



The Weblate Manual

Yayım 4.13

Michal Čihař

15 Haz 2022

1	User docs	1
1.1	Weblate basics	1
1.2	Registration and user profile	1
1.3	Translating using Weblate	10
1.4	Downloading and uploading translations	21
1.5	Sözlük	25
1.6	Checks and fixups	27
1.7	Searching	56
1.8	Çeviri iş akışları	63
1.9	Frequently Asked Questions	66
1.10	Supported file formats	74
1.11	Sürüm denetimi tümleşimi	95
1.12	Weblate's REST API	102
1.13	Weblate Client	149
1.14	Weblate's Python API	154
2	Administrator docs	157
2.1	Configuration instructions	157
2.2	Weblate deployments	221
2.3	Upgrading Weblate	222
2.4	Backing up and moving Weblate	230
2.5	Kimlik doğrulama	236
2.6	Erişim denetimi	247
2.7	Çeviri projeleri	256
2.8	Dil tanımları	274
2.9	Sürekli yerleştirme	277
2.10	Licensing translations	286
2.11	Çeviri işleyişi	288
2.12	Checks and fixups	294
2.13	Configuring automatic suggestions	305
2.14	Eklentiler	316
2.15	Çeviri Belleği	334
2.16	Yapılandırma	336
2.17	Sample configuration	361
2.18	Management commands	376
2.19	Duyurular	388
2.20	Component Lists	391
2.21	Optional Weblate modules	392
2.22	Customizing Weblate	397
2.23	Yönetim arayüzü	400
2.24	Getting support for Weblate	408

2.25	Legal documents	411
3	Contributor docs	413
3.1	Weblate'e Katkıda Bulunma	413
3.2	Weblate kodlarına katkıda bulunmaya başlama	415
3.3	Weblate kaynak kodları	419
3.4	Debugging Weblate	420
3.5	Weblate internals	422
3.6	Developing add-ons	423
3.7	Weblate frontend	425
3.8	Reporting issues in Weblate	426
3.9	Weblate testsuite and continuous integration	427
3.10	Data schemas	428
3.11	Releasing Weblate	432
3.12	Güvenlik ve gizlilik	433
3.13	Contributing to Weblate modules	433
3.14	Weblate hakkında	434
3.15	Lisans	435
4	Change History	436
4.1	Weblate 4.13	436
4.2	Weblate 4.12.2	437
4.3	Weblate 4.12.1	437
4.4	Weblate 4.12	437
4.5	Weblate 4.11.2	438
4.6	Weblate 4.11.1	438
4.7	Weblate 4.11	438
4.8	Weblate 4.10.1	439
4.9	Weblate 4.10	439
4.10	Weblate 4.9.1	440
4.11	Weblate 4.9	440
4.12	Weblate 4.8.1	441
4.13	Weblate 4.8	441
4.14	Weblate 4.7.2	442
4.15	Weblate 4.7.1	442
4.16	Weblate 4.7	442
4.17	Weblate 4.6.2	443
4.18	Weblate 4.6.1	443
4.19	Weblate 4.6	443
4.20	Weblate 4.5.3	444
4.21	Weblate 4.5.2	444
4.22	Weblate 4.5.1	445
4.23	Weblate 4.5	445
4.24	Weblate 4.4.2	446
4.25	Weblate 4.4.1	446
4.26	Weblate 4.4	446
4.27	Weblate 4.3.2	447
4.28	Weblate 4.3.1	448
4.29	Weblate 4.3	448
4.30	Weblate 4.2.2	449
4.31	Weblate 4.2.1	449
4.32	Weblate 4.2	449
4.33	Weblate 4.1.1	450
4.34	Weblate 4.1	450
4.35	Weblate 4.0.4	452
4.36	Weblate 4.0.3	452
4.37	Weblate 4.0.2	452
4.38	Weblate 4.0.1	453

4.39	Weblate 4.0	453
4.40	Weblate 3.x series	454
4.41	Weblate 2.x series	465
4.42	Weblate 1.x series	476
4.43	Weblate 0.x series	480
Python Modül Dizini		484
HTTP Routing Table		485
Dizin		488

1.1 Weblate basics

1.1.1 Proje ve bileşen yapısı

In Weblate translations are organized into projects and components. Each project can contain number of components and those contain translations into individual languages. The component corresponds to one translatable file (for example *GNU gettext* or *Android string resources*). The projects are there to help you organize component into logical sets (for example to group all translations used within one application).

Internally, each project has translations to common strings propagated across other components within it by default. This lightens the burden of repetitive and multi version translation. The translation propagation can be disabled per *Component configuration* using *Çeviri yaymaya izin ver* in case the translations should diverge.

Ayrıca bakınız:

[../devel/integration](#)

1.2 Registration and user profile

1.2.1 Kayıt

Everybody can browse projects, view translations or suggest translations by default. Only registered users are allowed to actually save changes, and are credited for every translation made.

You can register by following a few simple steps:

1. Fill out the registration form with your credentials.
2. Activate registration by following the link in the e-mail you receive.
3. Optionally adjust your profile to choose which languages you know.

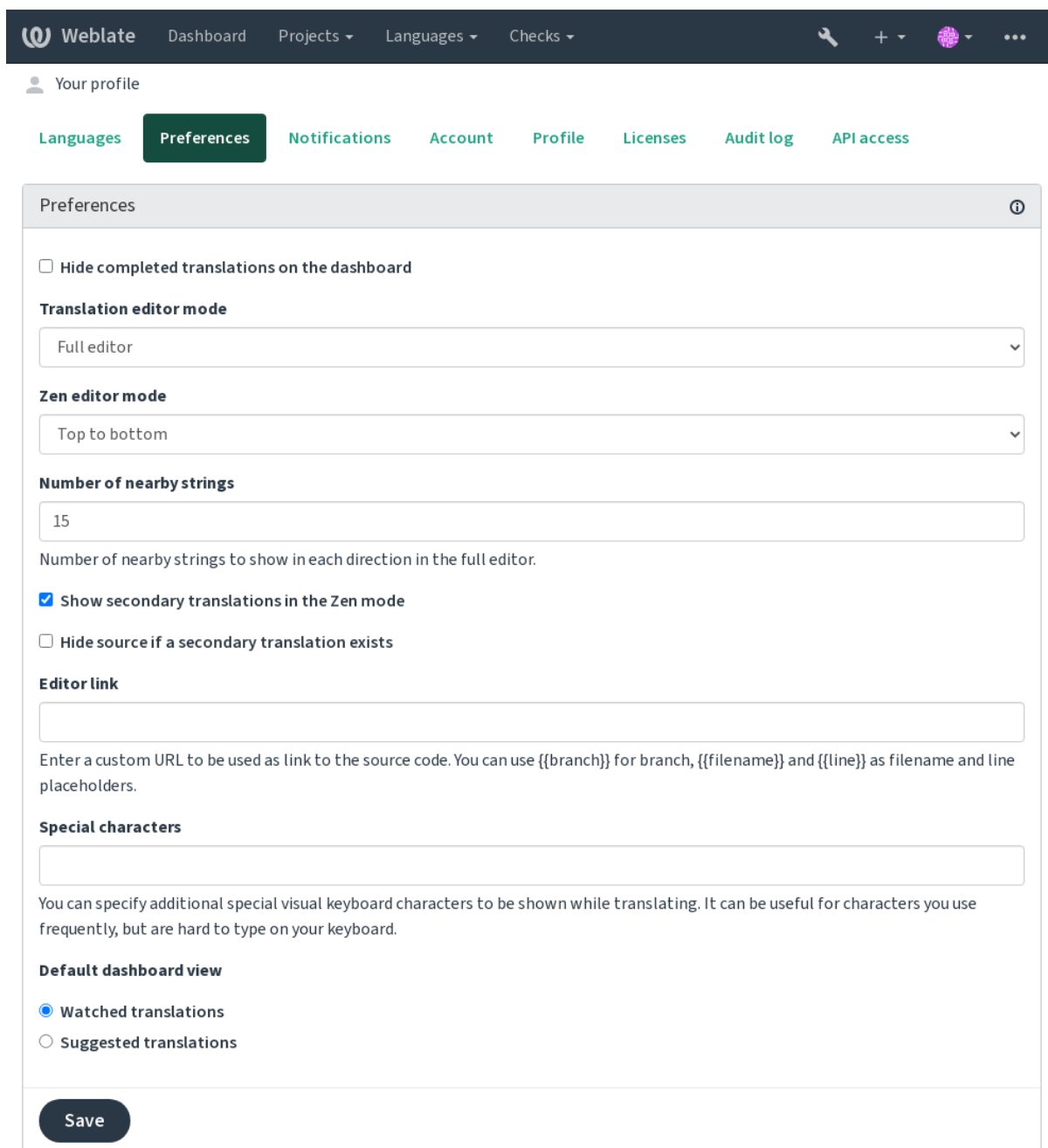
1.2.2 Panel

When you sign in, you will see an overview of projects and components, as well as their respective translation progression.

2.5 sürümünde geldi.

Components of projects you are watching are shown by default, and cross-referenced with your preferred languages.

İpucu: You can switch to different views using the navigation tabs.



The screenshot shows the Weblate user interface. At the top is a dark navigation bar with the Weblate logo, 'Dashboard', 'Projects', 'Languages', and 'Checks' menus. On the right are icons for settings, a plus sign, a user profile, and a menu. Below this is a 'Your profile' section with tabs for 'Languages', 'Preferences' (which is active), 'Notifications', 'Account', 'Profile', 'Licenses', 'Audit log', and 'API access'. The 'Preferences' panel is open, showing various settings. It includes a checkbox for 'Hide completed translations on the dashboard'. Under 'Translation editor mode', there is a dropdown menu set to 'Full editor'. Under 'Zen editor mode', there is a dropdown menu set to 'Top to bottom'. A text input field for 'Number of nearby strings' is set to '15', with a note below it: 'Number of nearby strings to show in each direction in the full editor.' There are checkboxes for 'Show secondary translations in the Zen mode' (checked) and 'Hide source if a secondary translation exists'. An 'Editor link' section has a text input field and a note: 'Enter a custom URL to be used as link to the source code. You can use {{branch}} for branch, {{filename}} and {{line}} as filename and line placeholders.' A 'Special characters' section has a text input field and a note: 'You can specify additional special visual keyboard characters to be shown while translating. It can be useful for characters you use frequently, but are hard to type on your keyboard.' Under 'Default dashboard view', there are two radio buttons: 'Watched translations' (selected) and 'Suggested translations'. At the bottom of the panel is a 'Save' button.

The menu has these options:

- *Projects > Browse all projects* in the main menu showing translation status for each project on the Weblate instance.
- Selecting a language in the main menu *Languages* will show translation status of all projects, filtered by one of your primary languages.
- *Watched translations* in the Dashboard will show translation status of only those projects you are watching, filtered by your primary languages.

In addition, the drop-down can also show any number of *component lists*, sets of project components preconfigured by the Weblate administrator, see [Component Lists](#).

You can configure your personal default dashboard view in the *Preferences* section of your user profile settings.

Not: When Weblate is configured for a single project using `SINGLE_PROJECT` in the `settings.py` file (see [Yapılandırma](#)), the dashboard will not be shown, as the user will be redirected to a single project or component instead.

1.2.3 Kullanıcı profili

The user profile is accessible by clicking your user icon in the top-right of the top menu, then the *Settings* menu.

The user profile contains your preferences. Name and e-mail address is used in VCS commits, so keep this info accurate.

Not: All language selections only offer currently translated languages.

İpucu: Request or add other languages you want to translate by clicking the button to make them available too.

Diller

1.2.4 Arayüz dili

Kullanıcı arayüzünü görüntülemek istediğiniz dili seçin.


Çevrilen diller

Choose which languages you prefer to translate, and they will be offered on the main page of watched projects, so that you have easier access to these all translations in each of those languages.


[Dashboard](#)
[Projects ▾](#)
[Languages ▾](#)
[Checks ▾](#)

 Dashboard

[Watched translations 13](#)
[Suggested translations 5](#)
[Insights ▾](#)
[Search](#)

Translation	Translated	Unfinished	Unfinished words	Checks	Suggestions
 WeblateOrg/Android — Czech    MIT	76%	3	3		
 WeblateOrg/Django — Hungarian   GPL-3.0	69%	8	109	1	
 WeblateOrg/Django — Czech    GPL-3.0	96%	1	12	4	
 WeblateOrg/Django — Hebrew    GPL-3.0	92%	2	15		
 WeblateOrg/Djangojs — Hungarian   GPL-3.0	96%	2	6		
 WeblateOrg/Djangojs — Hebrew    GPL-3.0	✓				
 WeblateOrg/Djangojs — Czech    GPL-3.0	✓				
 WeblateOrg/Language names — Czech    GPL-3.0	✓				
 WeblateOrg/Language names — Hungarian    GPL-3.0	81%	4	5		
 WeblateOrg/Language names — Hebrew    GPL-3.0	✓				
 WeblateOrg/WebplateOrg — Hungarian   GPL-3.0	100%				
 WeblateOrg/WebplateOrg — Czech   GPL-3.0	100%				
 WeblateOrg/WebplateOrg — Hebrew   GPL-3.0	100%				

Powered by Weblate 4.13
 [About Weblate](#)
[Legal](#)
[Contact](#)
[Documentation](#)
[Donate to Weblate](#)

İkincil diller

You can define which secondary languages are shown to you as a guide while translating. An example can be seen in the following image, where the Hebrew language is shown as secondarily:

The screenshot displays the Weblate web interface for a project named 'Django'. The user is in the 'Czech / Translate' view. The main translation area shows the source string 'Files' in English and its translation 'קבצים' in Hebrew. The Czech translation is 'Soubory'. The interface includes navigation buttons, a glossary, string information, and a list of nearby strings.

Translation Panel:

- Hebrew: קבצים
- English: Files
- Czech: Soubory
- Needs editing: ☐
- Buttons: Save and continue, Save and stay, Suggest, Skip

Glossary:

- English: No related strings found in the glossary.
- Czech: No related strings found in the glossary.
- Buttons: Add term to glossary

String information:

- Screenshot context: No screenshot currently associated. Buttons: Add screenshot
- Explanation: No explanation currently provided.
- Labels: No labels currently set.
- Flags: No flags currently set.
- Source string location: weblate/templates/translation.html:45, weblate/trans/forms.py:1404
- String age: 11 seconds ago
- Source string age: 11 seconds ago
- Translation file: weblate/locale/cs/LC_MESSAGES/django.po, string 1

Nearby strings: 16

History:

Language	Target string
Hebrew	קבצים
Hungarian	Fájlok
English	Files

1.2.5 Tercihler

Varsayılan panel görünümü

On the *Preferences* tab, you can pick which of the available dashboard views to present by default. If you pick the *Component list*, you have to select which component list will be displayed from the *Default component list* drop-down.

Ayrıca bakınız:

Component Lists

Düzenleyici bağlantısı

A source code link is shown in the web-browser configured in the *Component configuration* by default.

İpucu: By setting the *Editor link*, you use your local editor to open the VCS source code file of translated strings. You can use *Template markup*.

Usually something like `editor://open/?file={{filename}}&line={{line}}` is a good option.

Ayrıca bakınız:

You can find more info on registering custom URL protocols for the editor in the *Nette documentation*.

Özel karakterler

Additional special characters to include in the *Visual keyboard*.

1.2.6 Bildirimler


Subscribe to various notifications from the *Notifications* tab. Notifications for selected events on watched or administered projects will be sent to you per e-mail.





Some of the notifications are sent only for events in your languages (for example about new strings to translate), while some trigger at component level (for example merge errors). These two groups of notifications are visually separated in the settings.


You can toggle notifications for watched projects and administered projects and it can be further tweaked (or muted) per project and component. Visit the component overview page and select appropriate choice from the *Watching* menu.

In case *Automatically watch projects on contribution* is enabled you will automatically start watching projects upon translating a string. The default value depends on *DEFAULT_AUTO_WATCH*.

Not: You will not receive notifications for your own actions.


Dashboard
Projects
Languages
Checks


Your profile

Languages
Preferences
Notifications
Account
Profile
Licenses
Audit log
API access

Watched projects

☒ Automatically watch projects on contribution
Whenever you translate a string in a project, you will start watching it.

Watched projects

Search...

Available:

WebateOrg

Chosen:

WebateOrg

You can receive notifications for watched projects and they are shown on the dashboard by default.
Add all projects you want to translate to see them as watched projects on the dashboard.

Save

Notification settings

Other projects
Watched projects
Managed projects

Component wide notifications

You will receive a notification for every such event in your watched projects.

Repository failure

Do not notify

Repository operation

Do not notify

Component locking

Do not notify

Changed license

Do not notify

Parse error

Do not notify

Comment on own translation

Instant notification

Mentioned in comment

Instant notification

New language

Do not notify

New translation component

Do not notify

New announcement

Instant notification

New alert

Do not notify

Translation notifications

You will only receive these notifications for your translated languages in your watched projects.

New string

Do not notify

New contributor

Do not notify

New suggestion

Do not notify

New comment

Do not notify

Changed string

Do not notify

Translated string

Do not notify

Approved string

Do not notify

Pending suggestions

Do not notify

Unfinished strings

Do not notify

Save

Powered by Weblate 4.13

[About Weblate](#)
[Legal](#)
[Contact](#)
[Documentation](#)
[Donate to Weblate](#)

1.2.7 Hesap

The *Account* tab lets you set up basic account details, connect various services you can use to sign in into Weblate, completely remove your account, or download your user data (see [Weblate kullanıcı verilerini dışa aktarma](#)).

Not: The list of services depends on your Weblate configuration, but can be made to include popular sites such as GitLab, GitHub, Google, Facebook, or Bitbucket or other OAuth 2.0 providers.

 Weblate
 Dashboard Projects Languages Checks

+
 
 ...

 Your profile
 Languages Preferences Notifications **Account** Profile Licenses Audit log API access

Account

Username

Username may only contain letters, numbers or the following characters: @ . + - _

Full name



E-mail

You can add another e-mail address below.


Your name and e-mail will appear as commit authorship.

Save

Current user identities

Identity	User ID	Action
 Password	testuser	Change password
 E-mail	weblate@example.org	Disconnect
 Google	weblate@example.org	Disconnect
 GitHub	123456	Disconnect
 Bitbucket	weblate	Disconnect

Add new association


E-mail

Removal

Account removal deletes all your private data.

Remove my account

User data

You can download all your private data.

Download user data

1.2.8 Profil

Bu sayfadaki tüm alanlar isteğe bağlıdır ve istediğiniz zaman silinebilir ve bunları doldurarak, bu verileri kullanıcı profilinizin görüldüğü her yerde paylaşmamıza izin vermiş olursunuz.

Avatar can be shown for each user (depending on `ENABLE_AVATARS`). These images are obtained using <https://gravatar.com/>.

1.2.9 Lisanslar

1.2.10 API erişimi

You can get or reset your API access token [here](#).

1.2.11 Denetim günlüğü

Audit log keeps track of the actions performed with your account. It logs IP address and browser for every important action with your account. The critical actions also trigger a notification to a primary e-mail address.

Ayrıca bakınız:

Running behind reverse proxy

1.3 Translating using Weblate

Thank you for interest in translating using Weblate. Projects can either be set up for direct translation, or by way of accepting suggestions made by users without accounts.

Overall, there are two modes of translation:

- The project accepts direct translations
- The project only accepts suggestions, which are automatically validated once a defined number of votes is reached

Please see *Çeviri iş akışları* for more info on translation workflow.

Options for translation project visibility:

- Publicly visible
- Visible only to a certain group of translators

Ayrıca bakınız:

Erişim denetimi, Çeviri iş akışları

1.3.1 Çeviri projeleri

Translation projects hold related components; resources for the same software, book, or project.

Component	Translated	Unfinished	Unfinished words	Checks	Suggestions	Comments
Android	79%	30	30	3		
Language names	95%	4	5	5		
Glossary WeblateOrg	100%					

[Add new translation component](#)

Powered by Weblate 4.13 [About Weblate](#) [Legal](#) [Contact](#) [Documentation](#) [Donate to Weblate](#)

1.3.2 Translation links

Having navigated to a component, a set of links lead to its actual translation. The translation is further divided into individual checks, like *Untranslated strings* or *Unfinished strings*. If the whole project is translated, without error, *All strings* is still available. Alternatively you can use the search field to find a specific string or term.

W

Weblate

Dashboard

Projects

Languages

Checks

+

WebplateOrg

Django

Czech

translated 96%

Overview

Info

Search

Insights

Files

Tools

Manage

Share

Watching

Translation status

26

Strings

96%

183

Words

93%

Browse

Translate

Strings status

26

All strings — 183 words

Browse

Translate

Zen

25

Translated strings — 171 words

Browse

Translate

Zen

1

Unfinished strings — 12 words

Browse

Translate

Zen

1

Untranslated strings — 12 words

Browse

Translate

Zen

1

Unfinished strings without suggestions — 12 words

Browse

Translate

Zen

3

Strings with any failing checks — 11 words

Browse

Translate

Zen

3

Translated strings with any failing checks — 11 words

Browse

Translate

Zen

1

Failing check: Unchanged translation — 4 words

Browse

Translate

Zen

1

Failing check: Mismatched full stop — 4 words

Browse

Translate

Zen

1

Failing check: Python format — 3 words

Browse

Translate

Zen

Other components

Component	Translated	Unfinished	Unfinished words	Checks	Suggestions	Comments
<div>Android</div>	76%	3	3			
<div>Glossary</div>	100%					
<div>Djangojs</div>						
<div>Language names</div>						

Browse all components

Powered by Weblate 4.13

About Weblate

Legal

Contact

Documentation

Donate to Weblate

1.3.3 Öneriler

Not: Actual permissions might vary depending on your Weblate configuration.

Anonymous users can only (by default) forward suggestions. Doing so is still available to signed-in users, in cases where uncertainty about the translation arises, prompting other translators to review it.

The suggestions are scanned on a daily basis to remove duplicates and suggestions matching the current translation.

1.3.4 Yorumlar

Three types of comments can be posted: for translations, source strings, or to report source string bugs when this functionality is turned on using *Kaynak gözden geçirmelerini etkinleştir*. Choose the one suitable to topic you want to discuss. Source string comments are in any event good for providing feedback on the original string, for example that it should be rephrased or to ask questions about it.

Markdown söz dizimini tüm yorumlarda kullanabilir ve @kullanıcı_adı kullanarak diğer kullanıcılardan bahsedebilirsiniz.

Ayrıca bakınız:

report-source, *Source strings reviews*, *Kaynak gözden geçirmelerini etkinleştir*

1.3.5 Çeşitler

Variants are used to group different length variants of the string. The frontend of your project can then use different strings depending on the screen or window size.

Ayrıca bakınız:

variants, *Çeşitler*

1.3.6 Etiketler

Labels are used to categorize strings within a project to further customize the localization workflow (for example to define categories of strings).

Following labels are used by Weblate:

Otomatik olarak çevrildi

String was translated using *Otomatik çeviri*.

Kaynağı gözden geçirme gerekli

String was marked for review using *Source strings reviews*.

Ayrıca bakınız:

labels

1.3.7 Translating

On the translation page, the source string and an editing area for its translation are shown. Should the translation be plural, multiple source strings and editing areas are shown, each described and labeled in the amount of plural forms the translated language has.

All special whitespace characters are underlined in red and indicated with grey symbols. More than one subsequent space is also underlined in red to alert the translator to a potential formatting issue.

Various bits of extra info can be shown on this page, most of which coming from the project source code (like context, comments or where the message is being used). Translation fields for any secondary languages translators select in the preferences will be shown (see *İkincil diller*) above the source string.

Below the translation, translators will find suggestion made by others, to be accepted (✓), accepted with changes (⇒), or deleted (🗑).

Çoğullar

Words changing form to account of their numeric designation are called plurals. Each language has its own definition of plurals. English, for example, supports one. In the singular definition of for example “car”, implicitly one car is referenced, in the plural definition, “cars” two or more cars are referenced (or the concept of cars as a noun). Languages like for example Czech or Arabic have more plurals and also their rules for plurals are different.

Weblate has full support for each of these forms, in each respective language (by translating every plural separately). The number of fields and how it is in turn used in the translated application or project depends on the configured plural formula. Weblate shows the basic info, and the [Language Plural Rules](#) by the Unicode Consortium is a more detailed description.

Ayrıca bakınız:

Çoğul formülü

Alternative translations

4.13 sürümünde geldi.

Not: This is currently only supported with *Multivalue CSV file*.

With some formats, it is possible to have more translations for a single string. You can add more alternative translations using the *Tools* menu. Any blank alternative translations will be automatically removed upon saving.

Weblate
 Dashboard Projects Languages Checks

WeblateOrg / Django / Czech / Translate
 translated 96%

< 1 / 1 >

Custom search '%(count)s word'

Zen

Position and priority

Translation

English

Singular
 %(count)s word

Plural
 %(count)s words

Czech, One
 %(count)s slovo

Czech, Few
 %(count)s slova

Czech, Other
 %(count)s slov

Plural formula: (n==1) ? 0 : (n>=2 && n<=4) ? 1 : 2

☐ Needs editing

Save and continue Save and stay Suggest Skip

Nearby strings 20 Comments Automatic suggestions Other languages 3

History

New comment

Comment on this string for fellow translators and developers to read.

Scope
 Translation comment, discussions with other translators

Is your comment specific to this translation, or generic for all of them?

New comment

You can use Markdown and mention users by @username.

Save

Glossary

English Czech
 No related strings found in the glossary.
 Add term to glossary

String information

Screenshot context
 No screenshot currently associated.
 Add screenshot

Explanation
 No explanation currently provided.

Labels
 No labels currently set.

Flags
 python-format

Source string location
 weblate/templates/translation.html:149

String age
 6 seconds ago

Source string age
 6 seconds ago

Translation file
 weblate/locale/cs/LC_MESSAGE S/django.po, string 5

Klavye kısayolları

2.18 sürümünde değişti: The keyboard shortcuts have been revamped in 2.18 to less likely collide with browser or system defaults.

The following keyboard shortcuts can be utilized during translation:

Klavye kısayolu	Açıklama
Alt+Home	Geçerli aramadaki ilk çeviriye git.
Alt+End	Geçerli aramadaki son çeviriye git.
Alt+PageUp or Ctrl+↑ or Alt+↑ or Cmd+↑	Geçerli aramadaki önceki çeviriye git.
Alt+PageDown or Ctrl+↓ or Alt+↓ or Cmd+↓	Geçerli aramadaki sonraki çeviriye git.
Alt+Enter or Ctrl+Enter or Cmd+Enter	Submit current form; this is same as pressing <i>Save and continue</i> while editing translation.
Ctrl+Shift+Enter or Cmd+Shift+Enter	Düzenleme gerekli işaretini kaldır ve çeviriği gönder.
Ctrl+E or Cmd+E	Focus translation editor.
Ctrl+U or Cmd+U	Focus comment editor.
Ctrl+M or Cmd+M	Shows <i>Automatic suggestions</i> tab, see <i>Otomatik öneriler</i> .
Ctrl+1 to Ctrl+9 or Cmd+1 to Cmd+9	Copies placeable of given number from source string.
Ctrl+M+1 to 9 or Cmd+M+1 to 9	Copy the machine translation of given number to current translation.
Ctrl+I+1 to 9 or Cmd+I+1 to 9	Kusur denetimleri listesindeki bir öğeyi yok say.
Ctrl+J or Cmd+J	<i>Yakındaki dizgiler</i> sekmesini gösterir.
Ctrl+S or Cmd+S	Focus search field.
Ctrl+O or Cmd+O	Kaynak dizgiyi kopyala.
Ctrl+Y or Cmd+Y	Toggle the <i>Needs editing</i> checkbox.

Visual keyboard

A small visual keyboard row is shown just above the translation field. This can be useful to keep local punctuation in mind (as the row is local to each language), or have characters otherwise hard to type handy.

The shown symbols factor into three categories:

- User configured *Özel karakterler* defined in the *Kullanıcı profili*
- Per-language characters provided by Weblate (e.g. quotes or RTL specific characters)
- Characters configured using *SPECIAL_CHARS*

Weblate
 Dashboard Projects Languages Checks

WeblateOrg / Django / Hebrew / Translate
 translated 92%

1 / 26
 All strings
 Zen

Position and priority

Translation

English
Files

Hebrew
 NBS ... " ' , - - ZWNJ ZWJ LRM RLM LRE RLE PDF LRO RLO

Needs editing
 5/100 · 5 RTL LTR

Save and continue Save and stay Suggest Skip

Nearby strings 16 Comments Automatic suggestions Other languages 3

History

Language	Target string
Czech	Soubory
Hungarian	Fájlok
English	Files

Glossary

English Hebrew
 No related strings found in the glossary.
 Add term to glossary

String information

Screenshot context
 No screenshot currently associated.
 Add screenshot

Explanation
 No explanation currently provided.

Labels
 No labels currently set.

Flags
 No flags currently set.

Source string location
 weblate/templates/translation.html:45 · weblate/trans/forms.py:1404

String age
 12 seconds ago

Source string age
 12 seconds ago

Translation file
 weblate/locale/he/LC_MESSAGES/django.po, string 1

Translation context

This contextual description provides related info about the current string.

String attributes

Things like message ID, context (`msgctxt`) or location in source code.

Ekran görüntüleri

Screenshots can be uploaded to Weblate to better inform translators of where and how the string is used, see [Visual context for strings](#).

Yakındaki dizgiler

Displays neighbouring messages from the translation file. These are usually also used in a similar context and prove useful in keeping the translation consistent.

Diğer kullanımlar

In case a message appears in multiple places (e.g. multiple components), this tab shows all of them if they are found to be inconsistent (see [Tutarsız](#)). You can choose which one to use.

Çeviri Belleği

Look at similar strings translated in past, see [Çeviri Belleği](#).

Sözlük

Displays terms from the project glossary used in the current message.

En son değişiklikler

List of people whom have changed this message recently using Weblate.

Proje

Project info like instructions for translators, or a directory or link to the string in the version control system repository the project uses.

If you want direct links, the translation format has to support it.

Translation history

Every change is by default (unless turned off in component settings) saved in the database, and can be reverted. Optionally one can still also revert anything in the underlying version control system.

Translated string length

Weblate can limit the length of a translation in several ways to ensure the translated string is not too long:

- The default limitation for translation is ten times longer than the source string. This can be turned off by `LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH`. In case you are hitting this, it might be also caused by a monolingual translation erroneously set up as bilingual one, making Weblate mistaking the translation key for the actual source string. See [Bilingual and monolingual formats](#) for more info.
- Maximal length in characters defined by translation file or flag, see [Çevirinin en çok uzunluğu](#).
- Maximal rendered size in pixels defined by flags, see [Çeviri maksimum boyutu](#).

1.3.8 Otomatik öneriler

Based on configuration and your translated language, Weblate provides suggestions from several machine translation tools and *Çeviri Belleği*. All machine translations are available in a single tab of each translation page.

Ayrıca bakınız:

You can find the list of supported tools in *Configuring automatic suggestions*.

1.3.9 Otomatik çeviri

You can use automatic translation to bootstrap translation based on external sources. This tool is called *Automatic translation* accessible in the *Tools* menu, once you have selected a component and a language:

Automatic translation

Automatic translation takes existing translations in this project and pushes them to a different branch, to fix inconsistent translations in the current component. It can be used to push translations to a different branch, to fix inconsistent translations in the current component using translation memory. Automatic translation via machine translation uses active machine translation tools to get the best possible translations and applies them in this project.

Automatic translation mode

Add as suggestion

Search filter

Unfinished strings

Please note that translating all strings will discard all existing translations.

Source of automated translations

☐ Other translation components ☒ Machine translation

Machine translation engines

Search...

Available:

Chosen:

Score threshold

80

Apply

Two modes of operation are possible:

- Using other Weblate components as a source for translations.

- Using selected machine translation services with translations above a certain quality threshold.

You can also choose which strings are to be auto-translated.

Uyarı: Be mindful that this will overwrite existing translations if employed with wide filters such as *All strings*.

Useful in several situations like consolidating translation between different components (for example the application and its website) or when bootstrapping a translation for a new component using existing translations (translation memory).

The automatically translated strings are labelled *Automatically translated*.

Ayrıca bakınız:

Keeping translations same across components

1.3.10 Rate limiting

To avoid abuse of the interface, rate limiting is applied to several operations like searching, sending contact forms or translating. If affected by it, you are blocked for a certain period until you can perform the operation again.

Default limits and fine-tuning is described in the administrative manual, see *Rate limiting*.

1.3.11 Arama ve değiştirme

Change terminology effectively or perform bulk fixing of the strings using *Search and replace* in the *Tools* menu.

İpucu: Dizgileri karıştırma konusunda endişelenmeyin. Bu, asıl değişiklik onaylanmadan önce düzenlenen dizgilerin önizlemesini gösteren iki adımlı bir işlemdir.

1.3.12 Toplu düzenleme

Bulk editing allows performing one operation on number of strings. You define strings by searching for them and set up something to be done for matching ones. The following operations are supported:

- Changing string state (for example to approve all unreviewed strings).
- Adjust translation flags (see *Customizing behavior using flags*)
- Adjust string labels (see labels)

İpucu: This tool is called *Bulk edit* accessible in the *Tools* menu of each project, component or translation.

Ayrıca bakınız:

Bulk edit add-on

1.3.13 Matrix View

To compare different languages efficiently you can use the matrix view. It is available on every component page under the *Tools* menu. First select all languages you want to compare and confirm your selection, after that you can click on any translation to open and edit it quickly.

The matrix view is also a very good starting point to find missing translations in different languages and quickly add them from one view.

1.3.14 Zen Mode

The Zen editor can be enabled by clicking the *Zen* button on the top right while translating a component. It simplifies the layout and removes additional UI elements such as *Nearby strings* or the *Glossary*.

You can select the Zen editor as your default editor using the *Tercihler* tab on your *Kullanıcı profili*. Here you can also choose between having translations listed *Top to bottom* or *Side by side* depending on your personal preference.

1.4 Downloading and uploading translations

You can export files from a translation, make changes, and import them again. This allows working offline, and then merging changes back into the existing translation. This works even if it has been changed in the meantime.

Not: Available options might be limited by *access control* settings.

1.4.1 Downloading translations

From the project or component dashboard, translatable files can be downloaded in the *Files* menu.

The first option is to download the file in the original format as it is stored in the repository. In this case, any pending changes in the translation are getting committed and the up-to-date file is yield without any conversions.

You can also download the translation converted into one of the widely used localization formats. The converted files will be enriched with data provided in Weblate; such as additional context, comments or flags. Several file formats are available via the *Files* ↓ *Customize download* menu:

- gettext (PO)
- Gettext uzantılı XLIFF
- XLIFF 1.1
- TermBase eXchange
- Translation Memory eXchange
- gettext MO (only available when translation is using gettext PO)
- CSV
- Excel Open XML
- JSON (sadece tek dilli çeviriler için erişilebilir)
- Android Dizi Kaynağı (sadece tek dilli çeviriler için erişilebilir)
- iOS Dizileri (sadece tek dilli çeviriler için erişilebilir)

İpucu: The content available in the converted files differs based on file format features, you can find overview in *Translation types capabilities*.

Weblate
Dashboard
Projects
Languages
Checks

WeblateOrg / Django / Czech

translated 96%

Overview
Info
Search
Insights
Files
Tools
Manage
Share
Watching

Quick downloads
Download translation
Customize download
Upload translation

26	File in original format as translated in the repository									gettext PO file
26	All strings, converted files enriched with comments; suitable for offline translation	CSV	gettext MO	gettext PO	TBX	TMX	XLIFF 1.1 with gettext extensions	XLIFF 1.1	XLSX	
1	Unfinished strings, converted files enriched with comments; suitable for offline translation	CSV	gettext MO	gettext PO	TBX	TMX	XLIFF 1.1 with gettext extensions	XLIFF 1.1	XLSX	

Customize download

All strings

File format
☒ gettext PO
☐ XLIFF 1.1 with gettext extensions
☐ XLIFF 1.1
☐ TBX
☐ TMX
☐ gettext MO
☐ CSV
☐ XLSX
☐ JSON
☐ Android String Resource
☐ iOS strings

Download

Powered by Weblate 4.13
About Weblate
Legal
Contact
Documentation
Donate to Weblate

Ayrıca bakınız:

GET `/api/translations/(string:project)/(string:component)/(string:language)/file/`

1.4.2 Uploading translations

When you have made your changes, use *Upload translation* in the *Files* menu.

Webplate

Dashboard Projects Languages Checks

WebplateOrg / Django / Czech translated 96%

Overview Info Search Insights Files Tools Manage Share Watching

Upload

The uploaded file will be merged with the current file. Use the dropdown menu if you want to overwrite already translated strings.

File

Choose File No file chosen

File upload mode

☐ Add as translation
☐ Add as suggestion
☐ Add as translation needing edit
☐ Replace existing translation file

Processing of "Needs editing" strings

Do not import

Conflict handling

Change translated strings

Whether to overwrite existing translations if the string is already translated.

Author name

Weblate Test

Author e-mail

weblate@example.org

Upload

Powered by Weblate 4.13 [About Weblate](#) [Legal](#) [Contact](#) [Documentation](#) [Donate to Weblate](#)

Supported file formats

Any file in a supported file format can be uploaded, but it is still recommended to use the same file format as the one used for translation, otherwise some features might not be translated properly.

Ayrıca bakınız:

Supported file formats, Downloading and uploading translations

Import methods

These are the choices presented when uploading translation files:

Çeviri olarak ekle (**translate**)

Imported strings are added as translations to existing strings. This is the most common usecase, and the default behavior.

Only translations are used from the uploaded file and no additional content.

Öneri olarak ekle (**suggest**)

Imported strings are added as suggestions, do this when you want to have your uploaded strings reviewed.

Only translations are used from the uploaded file and no additional content.

Düzenlenmesi gereken çeviri olarak ekle (**fuzzy**)

Imported strings are added as translations needing edit. This can be useful when you want translations to be used, but also reviewed.

Only translations are used from the uploaded file and no additional content.

Mevcut çeviri dosyasını değiştir (**replace**)

Existing file is replaced with new content. This can lead to loss of existing translations, use with caution.

Kaynak dizgileri güncelle (**source**)

Updates source strings in bilingual translation file. This is similar to what *POT ile eşleşmesi için PO dosyalarını güncelle (msgmerge)* does.

Bu seçenek yalnızca bazı dosya biçimleri için desteklenir.

Yeni dizgiler ekle (**add**)

Adds new strings to the translation. It skips the one which already exist.

In case you want to both add new strings and update existing translations, upload the file second time with *Add as translation*.

This option is available only with *Dizgileri yönet* turned on.

Only source, translation and key (context) are used from the uploaded file.

Ayrıca bakınız:

```
POST /api/translations/(string:project)/(string:component)/  
(string:language)/file/
```

Conflicts handling

Defines how to deal with uploaded strings which are already translated.

Strings needing edit

There is also an option for how to handle strings needing edit in the imported file. Such strings can be handle in one of the three following ways: “Do not import”, “Import as string needing edit”, or “Import as translated”.

Overriding authorship

With admin permissions, you can also specify authorship of uploaded file. This can be useful in case you've received the file in another way and want to merge it into existing translations while properly crediting the actual author.

1.5 Sözlük

Each project can include one or more glossaries as a shorthand for storing terminology. Glossary easify maintaining consistency of the translation.

A glossary for each language can be managed on its own, but they are stored together as a single component which helps project admins and multilingual translators to maintain some cross-language consistency as well. Terms from the glossary containing words from the currently translated string are displayed in the sidebar of the translation editor.

1.5.1 Managing glossaries

4.5 sürümünde değişti: Glossaries are now regular translation components and you can use all Weblate features on them — commenting, storing in a remote repository, or adding explanations.

Use any component as a glossary by turning on *Sözlük olarak kullan*. You can create multiple glossaries for one project.

An empty glossary for a given project is automatically created with the project. Glossaries are shared among all components of the same project, and optionally with other projects using *Projelerde paylaş* from the respective glossary component.

The glossary component looks like any other component in Weblate with added colored label:

The screenshot shows the Weblate web interface for a project named 'Czech'. The top navigation bar includes 'Dashboard', 'Projects', 'Languages', and 'Checks'. The main header shows 'WeblateOrg / Glossary WeblateOrg / Czech' with a 'translated 100%' badge. Below the header, there are tabs for 'Overview', 'Info', 'Search', 'Insights', 'Files', 'Tools', and 'Share'. The 'Overview' tab is active.

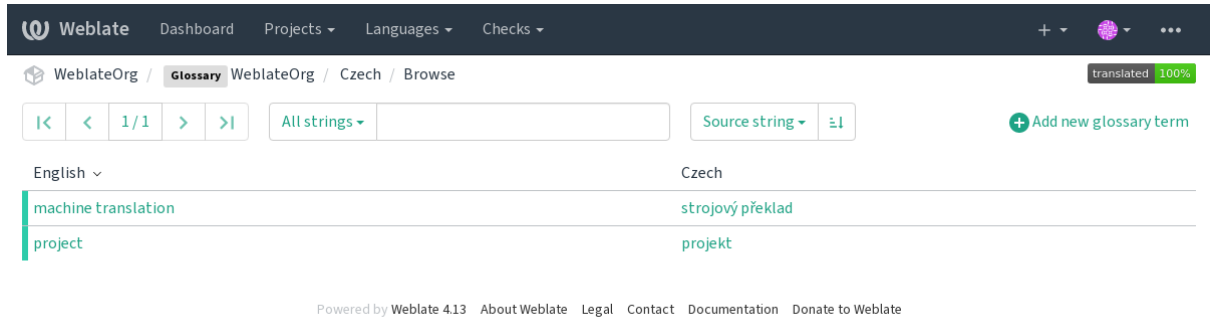
The 'Translation status' section shows two progress bars: '2 Strings' at 100% and '3 Words' at 100%. There are buttons for 'Add new glossary term', 'Browse', and 'Translate'.

The 'Strings status' section shows two items: 'All strings — 3 words' and 'Translated strings — 3 words'. Each item has buttons for 'Browse', 'Translate', and 'Zen'.

The 'Other components' section shows a table with columns: Component, Translated, Unfinished, Unfinished words, Checks, Suggestions, and Comments. The table lists two components: 'Django' (96% translated, 1 unfinished, 12 unfinished words, 3 checks) and 'Language names' (100% translated, 0 unfinished, 0 unfinished words, 0 checks). There is a 'Browse all components' button at the bottom.

Powered by Weblate 4.13 | [About Weblate](#) | [Legal](#) | [Contact](#) | [Documentation](#) | [Donate to Weblate](#)

Tüm sözlük terimlerine göz atabilirsiniz:

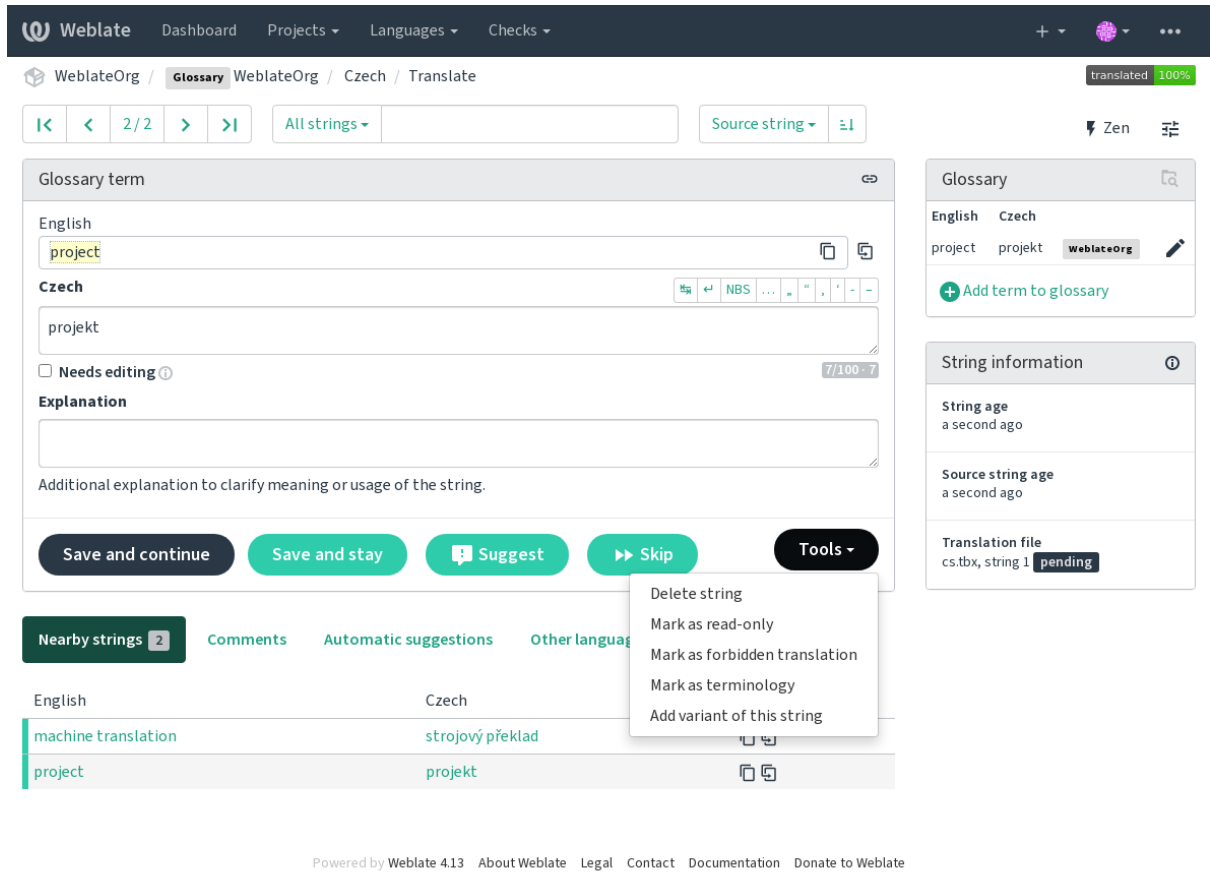


Powered by Weblate 4.13 About Weblate Legal Contact Documentation Donate to Weblate

veya bunları herhangi bir çeviri olarak düzenleyin.

1.5.2 Sözlük terimleri

Glossary terms are translated the same way regular strings are. You can toggle additional features using the *Tools* menu for each term.



Powered by Weblate 4.13 About Weblate Legal Contact Documentation Donate to Weblate

Untranslatable terms

4.5 sürümünde geldi.

Flagging certain glossary term translations `read-only` by bulk-editing, typing in the flag, or by using *Tools* ↓ *Mark as read-only* means they can not be translated. Use this for brand names or other terms that should not be changed in other languages. Such terms are visually highlighted in the glossary sidebar.

Ayrıca bakınız:

Customizing behavior using flags

Yasaklı çeviriler

4.5 sürümünde geldi.

Flagging certain glossary term translations as `forbidden`, by bulk-editing, typing in the flag, or by using *Tools* ↓ *Mark as forbidden translation* means they are **not** to be used. Use this to clarify translation when some words are ambiguous or could have unexpected meanings.

Ayrıca bakınız:

Customizing behavior using flags

Terminoloji

4.5 sürümünde geldi.

Flagging certain glossary terms as `terminology` by bulk-editing, typing in the flag, or by using *Tools* ↓ *Mark as terminology* adds entries for them to all languages in the glossary. Use this for important terms that should be well thought out, and retain a consistent meaning across all languages.

Ayrıca bakınız:

Customizing behavior using flags

Çeşitler

Variants are a generic way to group strings together. All term variants are listed in the glossary sidebar when translating.

İpucu: You can use this to add abbreviations or shorter expressions for a term.

Ayrıca bakınız:

variants

1.6 Checks and fixups

The quality checks help catch common translator errors, ensuring the translation is in good shape. The checks can be ignored in case of false positives.

Once submitting a translation with a failing check, this is immediately shown to the user:

[Dashboard](#)
[Projects](#)
[Languages](#)
[Checks](#)

[WeblateOrg](#) / [Django](#) / [Czech](#) / [Translate](#)
translated 96%

The translation has been saved, however there are some newly failing checks: Python format, Missing plurals

<<

<

1 / 1

>

>>

Custom search
'% (count)s word'

Position

Zen

Translation

English

Singular

% (count)s word

Plural

% (count)s words

Czech, One

Czech, Few

několik slov

Czech, Other

% (count)s slov

Plural formula: (n==1) ? 0 : (n>=2 && n<=4) ? 1 : 2

☐ Needs editing

Save and continue

Save and stay

Suggest

Skip

Things to check

Python format

Following format strings are missing: % (count)s

Dismiss

☐ For all languages

Missing plurals

Some plural forms are untranslated

Dismiss

☐ For all languages

Glossary

English

Czech

No related strings found in the glossary.

Add term to glossary

Nearby strings

Comments

Automatic suggestions

Other languages

History

New comment

Comment on this string for fellow translators and developers to read.

Scope

Translation comment, discussions with other translators

Is your comment specific to this translation, or generic for all of them?

New comment

You can use Markdown and mention users by @username.

Save

String information

Screenshot context

No screenshot currently associated.

Add screenshot

Explanation

No explanation currently provided.

Labels

No labels currently set.

Flags

python-format

Source string location

weblate/templates/translation.html:149

String age

10 seconds ago

Source string age

10 seconds ago

Translation file

weblate/locale/cs/LC_MESSAGES/django.po, string 5

pending

Powered by Weblate 4.13 [About Weblate](#) [Legal](#) [Contact](#) [Documentation](#) [Donate to Weblate](#)

28

Bölüm 1. User docs

1.6.1 Automatic fixups

In addition to *Quality checks*, Weblate can fix some common errors in translated strings automatically. Use it with caution to not have it add errors.

Ayrıca bakınız:

AUTOFIX_LIST

1.6.2 Quality checks

Weblate employs a wide range of quality checks on strings. The following section describes them all in further detail. There are also language specific checks. Please file a bug if anything is reported in error.

Ayrıca bakınız:

CHECK_LIST, *Customizing behavior using flags*

1.6.3 Translation checks

Executed upon every translation change, helping translators maintain good quality translations.

BBCode işaretleme

Özet

Çevirideki BBCode kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.markup.BBCodeCheck`

Flag to ignore

`ignore-bbcode`

BBCode represents simple markup, like for example highlighting important parts of a message in bold font, or italics. This check ensures they are also found in translation.

Not: The method for detecting BBCode is currently quite simple so this check might produce false positives.

Art arda yinelenen sözcükler

4.1 sürümünde geldi.

Özet

Metin aynı sözcüğü üst üste iki kez içeriyor:

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.duplicate.DuplicateCheck`

Flag to ignore

`ignore-duplicate`

Checks that no consecutive duplicate words occur in a translation. This usually indicates a mistake in the translation.

İpucu: This check includes language specific rules to avoid false positives. In case it triggers falsely in your case, let us know. See [Reporting issues in Weblate](#).

Sözlüğü takip etme

4.5 sürümünde geldi.

Özet

Çeviri, bir sözlükte tanımlanan terimleri takip etmiyor.

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.glossary.GlossaryCheck`

Flag to enable

`check-glossary`

Flag to ignore

`ignore-check-glossary`

This check has to be turned on using `check-glossary` flag (see [Customizing behavior using flags](#)). Please consider following prior to enabling it:

- It does exact string matching, the glossary is expected to contain terms in all variants.
- Checking each string against glossary is expensive, it will slow down any operation in Weblate which involves running checks like importing strings or translating.

Ayrıca bakınız:

[Sözlük](#), [Customizing behavior using flags](#), [Çeviri işaretleri](#)

Çift boşluk

Özet

Çeviri çift boşluk içeriyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.DoubleSpaceCheck`

Flag to ignore

`ignore-double-space`

Checks that double space is present in translation to avoid false positives on other space-related checks.

Check is false when double space is found in source meaning double space is intentional.

Formatted strings

Checks that formatting in strings are replicated between both source and translation. Omitting format strings in translation usually causes severe problems, so the formatting in strings should usually match the source.

Weblate supports checking format strings in several languages. The check is not enabled automatically, only if a string is flagged appropriately (e.g. *c-format* for C format). Gettext adds this automatically, but you will probably have to add it manually for other file formats or if your PO files are not generated by **xgettext**.

This can be done per unit (see [Additional info on source strings](#)) or in [Component configuration](#). Having it defined per component is simpler, but can lead to false positives in case the string is not interpreted as a formatting string, but format string syntax happens to be used.

İpucu: In case specific format check is not available in Weblate, you can use generic [Yer tutucular](#).

Besides checking, this will also highlight the formatting strings to easily insert them into translated strings:

Weblate
 Dashboard Projects Languages Checks

WeblateOrg / Django / Czech / Translate
 translated 96%

1 / 1

Custom search

'%(count)s word'

Position and priority

Zen

Translation

English

Singular

%(count)s word

Plural

%(count)s words

Czech, One

%(count)s slovo

Czech, Few

%(count)s slova

Czech, Other

%(count)s slov

Plural formula: (n==1) ? 0 : (n>2 && n<=4) ? 1 : 2

☐ Needs editing

Save and continue

Save and stay

Suggest

Skip

Nearby strings 20

Comments

Automatic suggestions

Other languages 3

History

None
 String updated in the repository
 WeblateOrg / Django — Czech

English

Singular

%(count)s word

Plural

%(count)s words

Czech Translated

One

%(count)s slovo

Few

%(count)s slova

Other

%(count)s slov

6 seconds ago

Browse all component changes

Powered by Weblate 4.13

About Weblate

Legal

Contact

Documentation

Donate to Weblate

Glossary

English Czech

No related strings found in the glossary.

Add term to glossary

String information

Screenshot context

No screenshot currently associated.

Add screenshot

Explanation

No explanation currently provided.

Labels

No labels currently set.

Flags

python-format

Source string location

weblate/templates/translation.html:149

String age

6 seconds ago

Source string age

6 seconds ago

Translation file

weblate/locale/cs/LC_MESSAGES/django.po, string 5

32

Bölüm 1. User docs

AngularJS ilişkilendirme dizgisi

Özet

AngularJS ilişkilendirme dizgileri kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.angularjs.AngularJSInterpolationCheck`

Flag to enable

`angularjs-format`

Flag to ignore

`ignore-angularjs-format`

Named format string example

Your balance is {{amount}} {{ currency }}

Ayrıca bakınız:

Formatted strings, *AngularJS text interpolation*

C biçimi

Özet

C biçimindeki dizgi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.format.CFormatCheck`

Flag to enable

`c-format`

Flag to ignore

`ignore-c-format`

Simple format string example

There are %d apples

Position format string example

Your balance is %1\$d %2\$s

Ayrıca bakınız:

Formatted strings,

C format strings, *C printf format*

C# biçimi

Özet

C# biçimindeki dizgi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.format.CSharpFormatCheck`

Flag to enable

c-sharp-format

Flag to ignore

ignore-c-sharp-format

Position format string example

There are {0} apples

Ayrıca bakınız:

[Formatted strings](#), [C# String Format](#)

ECMAScript şablon değişmezleri

Özet

ECMAScript şablon değişmezleri kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

weblate.checks.format.ESTemplateLiteralsCheck

Flag to enable

es-format

Flag to ignore

ignore-es-format

Interpolation example

There are \${number} apples

Ayrıca bakınız:

[Formatted strings](#), [Template literals](#)

i18next ilişkilendirmesi

4.0 sürümünde geldi.

Özet

I18next ilişkilendirmesi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

weblate.checks.format.I18NextInterpolationCheck

Flag to enable

i18next-interpolation

Flag to ignore

ignore-i18next-interpolation

Interpolation example

There are {{number}} apples

Nesting example

There are \$t(number) apples

Ayrıca bakınız:

[Formatted strings](#), [i18next interpolation](#)

ICU MessageFormat

4.9 sürümünde geldi.

Özet

ICU MessageFormat dizgilerinde sözdizimi hataları ve/veya yer tutucu uyumsuzlukları.

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.icu.ICUMessageFormatCheck`

Flag to enable

`icu-message-format`

Flag to ignore

`ignore-icu-message-format`

Interpolation example

There {number, plural, one {is one apple} other {are # apples}}.

This check has support for both pure ICU MessageFormat messages as well as ICU with simple XML tags. You can configure the behavior of this check by using `icu-flags: *`, either by opting into XML support or by disabling certain sub-checks. For example, the following flag enables XML support while disabling validation of plural sub-messages:

<code>xml</code>	Enable support for simple XML tags. By default, XML tags are parsed loosely. Stray < characters are ignored if they are not reasonably part of a tag.
<code>strict-xml</code>	Enable support for strict XML tags. All < characters must be escaped if they are not part of a tag.
<code>-highlight</code>	Disable highlighting placeholders in the editor.
<code>-require_other</code>	Disable requiring sub-messages to have an <code>other</code> selector.
<code>-submessage_selector</code>	Skip checking that sub-message selectors match the source.
<code>-types</code>	Skip checking that placeholder types match the source.
<code>-extra</code>	Skip checking that no placeholders are present that were not present in the source string.
<code>-missing</code>	Skip checking that no placeholders are missing that were present in the source string.

Additionally, when `strict-xml` is not enabled but `xml` is enabled, you can use the `icu-tag-prefix: PREFIX` flag to require that all XML tags start with a specific string. For example, the following flag will only allow XML tags to be matched if they start with `<x::`:

This would match `<x:link>click here</x:link>` but not `this`.

Ayrıca bakınız:

ICU MessageFormat sözdizimi, *Formatted strings*, *ICU: Formatting Messages*, *Format.JS: Message Syntax*

Java biçimi

Özet

Java biçimindeki dizgi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.format.JavaFormatCheck`

Flag to enable

`java-format`

Flag to ignore

`ignore-java-format`

Simple format string example

There are %d apples

Position format string example

Your balance is %1\$d %2\$s

Ayrıca bakınız:

[Formatted strings](#), [Java Format Strings](#)

Java MessageFormat

Özet

Java MessageFormat dizgisi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.format.JavaMessageFormatCheck`

Flag to enable unconditionally

`java-messageformat`

Flag to enable autodetection

`auto-java-messageformat` enables check only if there is a format string in the source

Flag to ignore

`ignore-java-messageformat`

Position format string example

There are {0} apples

Ayrıca bakınız:

[Formatted strings](#), [Java MessageFormat](#)

JavaScript biçimi

Özet

JavaScript biçimindeki dizgi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.format.JavaScriptFormatCheck`

Flag to enable

`javascript-format`

Flag to ignore

`ignore-javascript-format`

Simple format string example

There are %d apples

Ayrıca bakınız:

[Formatted strings](#), [JavaScript formatting strings](#)

Lua biçimi

Özet

Lua biçimindeki dizgi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.format.LuaFormatCheck`

Flag to enable

`lua-format`

Flag to ignore

`ignore-lua-format`

Simple format string example

There are %d apples

Ayrıca bakınız:

[Formatted strings](#), [Lua formatting strings](#)

Object Pascal biçimi

Özet

Object Pascal biçimindeki dizgi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.format.ObjectPascalFormatCheck`

Flag to enable

`object-pascal-format`

Flag to ignore

`ignore-object-pascal-format`

Simple format string example

There are %d apples

Ayrıca bakınız:

[Formatted strings](#), [Object Pascal formatting strings](#), [Free Pascal formatting strings](#) [Delphi formatting strings](#)

Yüzde yer tutucuları

4.0 sürümünde geldi.

Özet

Yüzde yer tutucuları kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.format.PercentPlaceholdersCheck`

Flag to enable

`percent-placeholders`

Flag to ignore

ignore-percent-placeholders

Simple format string example

There are %number% apples

Ayrıca bakınız:

[Formatted strings](#),

Perl biçimi

Özet

Perl biçimindeki dizgi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

weblate.checks.format.PperlFormatCheck

Flag to enable

perl-format

Flag to ignore

ignore-perl-format

Simple format string example

There are %d apples

Position format string example

Your balance is %1\$d %2\$s

Ayrıca bakınız:

[Formatted strings](#), [Perl sprintf](#), [Perl Format Strings](#)

PHP biçimi

Özet

PHP biçimindeki dizgi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

weblate.checks.format.PHPFormatCheck

Flag to enable

php-format

Flag to ignore

ignore-php-format

Simple format string example

There are %d apples

Position format string example

Your balance is %1\$d %2\$s

Ayrıca bakınız:

[Formatted strings](#), [PHP sprintf documentation](#), [PHP Format Strings](#)

Python ayraç biçimi

Özet

Python ayraç biçimindeki dizgi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.format.PythonBraceFormatCheck`

Flag to enable

`python-brace-format`

Flag to ignore

`ignore-python-brace-format`

Simple format string

There are {} apples

Named format string example

Your balance is {amount} {currency}

Ayrıca bakınız:

Formatted strings, *Python brace format*, *Python Format Strings*

Python biçimi

Özet

Python biçim dizgisi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.format.PythonFormatCheck`

Flag to enable

`python-format`

Flag to ignore

`ignore-python-format`

Simple format string

There are %d apples

Named format string example

Your balance is %(amount)d %(currency)s

Ayrıca bakınız:

Formatted strings, *Python string formatting*, *Python Format Strings*

Qt biçimi

Özet

Qt biçimindeki dizgi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.qt.QtFormatCheck`

Flag to enable

`qt-format`

Flag to ignore

`ignore-qt-format`

Position format string example

There are %1 apples

Ayrıca bakınız:

[Formatted strings](#), [Qt QString::arg\(\)](#)

Qt çoğul biçimi

Özet

Qt çoğul biçimindeki dizgi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.qt.QtPluralCheck`

Flag to enable

`qt-plural-format`

Flag to ignore

`ignore-qt-plural-format`

Plural format string example

There are %Ln apple(s)

Ayrıca bakınız:

[Formatted strings](#), [Qt i18n guide](#)

Ruby biçimi

Özet

Ruby biçim dizgisi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.ruby.RubyFormatCheck`

Flag to enable

`ruby-format`

Flag to ignore

`ignore-ruby-format`

Simple format string example

```
There are %d apples
```

Position format string example

```
Your balance is %1$f %2$s
```

Named format string example

```
Your balance is %+.2<amount>f %<currency>s
```

Named template string

```
Your balance is %{amount} %{currency}
```

Ayrıca bakınız:

Formatted strings, [Ruby Kernel#sprintf](#)

Şema biçimi**Özet**

Şema biçim dizgisi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

```
weblate.checks.format.SchemeFormatCheck
```

Flag to enable

```
scheme-format
```

Flag to ignore

```
ignore-scheme-format
```

Simple format string example

```
There are ~d apples
```

Ayrıca bakınız:

Formatted strings, [Srfi 28](#), [Chicken Scheme format](#), [Guile Scheme formatted output](#)

Vue I18n biçimlendirmesi**Özet**

Vue I18n biçimlendirmesi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

```
weblate.checks.format.VueFormattingCheck
```

Flag to enable

```
vue-format
```

Flag to ignore

```
ignore-vue-format
```

Adlandırılmış biçimlendirme

```
There are {count} apples
```

Rails i18n biçimlendirmesi

```
There are %{count} apples
```

Bağlantılı yerel iletiler

```
@:message.dio @:message.the_world!
```

Ayrıca bakınız:

[Formatted strings](#), [Vue I18n Formatting](#), [Vue I18n Linked locale messages](#)

Çevrilmiş

Özet

Bu dizgi geçmişte çevrilmiş

Kapsam

tüm dizgiler

Check class

`weblate.checks.consistency.TranslatedCheck`

Flag to ignore

`ignore-translated`

Means a string has been translated already. This can happen when the translations have been reverted in VCS or lost otherwise.

Tutarsız

Özet

Bu dizginin bu projede birden fazla çevirisi var ya da bazı bileşenlerde çevrilmedi.

Kapsam

tüm dizgiler

Check class

`weblate.checks.consistency.ConsistencyCheck`

Flag to ignore

`ignore-inconsistent`

Weblate checks translations of the same string across all translation within a project to help you keep consistent translations.

The check fails on differing translations of one string within a project. This can also lead to inconsistencies in displayed checks. You can find other translations of this string on the *Other occurrences* tab.

This check applies to all components in a project that have [Çeviri yaymaya izin ver](#) turned on.

İpucu: For performance reasons, the check might not find all inconsistencies, it limits number of matches.

Not: This check also fires in case the string is translated in one component and not in another. It can be used as a quick way to manually handle strings which are untranslated in some components just by clicking on the *Use this translation* button displayed on each line in the *Other occurrences* tab.

You can use [Otomatik çeviri](#) add-on to automate translating of newly added strings which are already translated in another component.

Ayrıca bakınız:

[Keeping translations same across components](#)

Kashida harfi kullanıldı

3.5 sürümünde geldi.

Özet

Dekoratif kashida harfleri kullanılmamalıdır

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.KashidaCheck`

Flag to ignore

`ignore-kashida`

The decorative Kashida letters should not be used in translation. These are also known as Tatweel.

Ayrıca bakınız:

[Kashida on Wikipedia](#)

Markdown bağlantıları

3.5 sürümünde geldi.

Özet

Markdown bağlantıları kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.markup.MarkdownLinkCheck`

Flag to enable

`md-text`

Flag to ignore

`ignore-md-link`

Markdown links do not match source.

Ayrıca bakınız:

[Markdown links](#)

Markdown başvuruları

3.5 sürümünde geldi.

Özet

Markdown bağlantı başvuruları kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.markup.MarkdownRefLinkCheck`

Flag to enable

`md-text`

Flag to ignore

`ignore-md-reflink`

Markdown link references do not match source.

Ayrıca bakınız:

[Markdown links](#)

Markdown sözdizimi

3.5 sürümünde geldi.

Özet

Markdown sözdizimi kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.markup.MarkdownSyntaxCheck`

Flag to enable

`md-text`

Flag to ignore

`ignore-md-syntax`

Markdown sözdizimi kaynakla eşleşmiyor

Ayrıca bakınız:

[Markdown span elements](#)

Çevirinin en çok uzunluğu

Özet

Çeviri verilen uzunluğu aşmamalı

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.MaxLengthCheck`

Flag to enable

`max-length`

Flag to ignore

`ignore-max-length`

Checks that translations are of acceptable length to fit available space. This only checks for the length of translation characters.

Unlike the other checks, the flag should be set as a `key:value` pair like `max-length:100`.

İpucu: This check looks at number of chars, what might not be the best metric when using proportional fonts to render the text. The *Çeviri maksimum boyutu* check does check actual rendering of the text.

The `replacements:` flag might be also useful to expand placeables before checking the string.

When `xml-text` flag is also used, the length calculation ignores XML tags.

Çeviri maksimum boyutu

Özet

Çevrilmiş metnin belirtilen boyutu aşmaması gerekir

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.render.MaxSizeCheck`

Flag to enable

`max-size`

Flag to ignore

`ignore-max-size`

3.7 sürümünde geldi.

Translation rendered text should not exceed given size. It renders the text with line wrapping and checks if it fits into given boundaries.

This check needs one or two parameters - maximal width and maximal number of lines. In case the number of lines is not provided, one line text is considered.

You can also configure used font by `font-*` directives (see [Customizing behavior using flags](#)), for example following translation flags say that the text rendered with ubuntu font size 22 should fit into two lines and 500 pixels:

```
max-size:500:2, font-family:ubuntu, font-size:22
```

İpucu: You might want to set `font-*` directives in [Component configuration](#) to have the same font configured for all strings within a component. You can override those values per string in case you need to customize it per string.

The `replacements: flag` might be also useful to expand placeables before checking the string.

When `xml-text` flag is also used, the length calculation ignores XML tags.

Ayrıca bakınız:

[Managing fonts](#), [Customizing behavior using flags](#), [Çevirinin en çok uzunluğu](#)

\n uyuşmadı

Özet

Çevirideki \n sayısı kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.EscapedNewlineCountingCheck`

Flag to ignore

`ignore-escaped-newline`

Usually escaped newlines are important for formatting program output. Check fails if the number of `\n` literals in translation do not match the source.

İki nokta üst üste uyuşmadı

Özet

Kaynağın ve çeviri iki nokta üst üste ile bitmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.EndColonCheck`

Flag to ignore

`ignore-end-colon`

Checks that colons are replicated between both source and translation. The presence of colons is also checked for various languages where they do not belong (Chinese or Japanese).

Ayrıca bakınız:

[Colon on Wikipedia](#)

Üç nokta uyuşmadı

Özet

Kaynağın ve çevirinin ikisinde üç nokta ile bitmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.EndEllipsisCheck`

Flag to ignore

`ignore-end-ellipsis`

Checks that trailing ellipses are replicated between both source and translation. This only checks for real ellipsis (...) not for three dots (. . .).

An ellipsis is usually rendered nicer than three dots in print, and sounds better with text-to-speech.

Ayrıca bakınız:

[Ellipsis on Wikipedia](#)

Ünlem işareti uyuşmadı

Özet

Kaynak ve çeviri bir ünlem işareti ile bitmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.EndExclamationCheck`

Flag to ignore

`ignore-end-exclamation`

Checks that exclamations are replicated between both source and translation. The presence of exclamation marks is also checked for various languages where they do not belong (Chinese, Japanese, Korean, Armenian, Limbu, Myanmar or Nko).

Ayrıca bakınız:

[Exclamation mark on Wikipedia](#)

Nokta işareti uyuşmadı

Özet

Kaynağın ve çevirinin ikisinde nokta ile bitmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.EndStopCheck`

Flag to ignore

`ignore-end-stop`

Checks that full stops are replicated between both source and translation. The presence of full stops is checked for various languages where they do not belong (Chinese, Japanese, Devanagari or Urdu).

Ayrıca bakınız:

[Full stop on Wikipedia](#)

Soru işareti uyuşmadı

Özet

Kaynak ve çeviri bir soru işareti ile bitmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.EndQuestionCheck`

Flag to ignore

`ignore-end-question`

Checks that question marks are replicated between both source and translation. The presence of question marks is also checked for various languages where they do not belong (Armenian, Arabic, Chinese, Korean, Japanese, Ethiopic, Vai or Coptic).

Ayrıca bakınız:

[Question mark on Wikipedia](#)

Noktalı virgöl uyuşmadı

Özet

Kaynağın ve çevirinin ikisinde noktalı virgöl ile bitmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.EndSemicolonCheck`

Flag to ignore

`ignore-end-semicolon`

Checks that semicolons at the end of sentences are replicated between both source and translation.

Ayrıca bakınız:

[Semicolon on Wikipedia](#)

Uyuşmayan satır sonları

Özet

Çevirideki yeni satır sayısı kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.NewLineCountCheck`

Flag to ignore

`ignore-newline-count`

Usually newlines are important for formatting program output. Check fails if the number of `\n` literals in translation do not match the source.

Eksik çoğul

Özet

Bazı çoğul biçimler çevrilmedi

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.consistency.PluralsCheck`

Flag to ignore

`ignore-plurals`

Checks that all plural forms of a source string have been translated. Specifics on how each plural form is used can be found in the string definition.

Failing to fill in plural forms will in some cases lead to displaying nothing when the plural form is in use.

Yer tutucular

3.9 sürümünde geldi.

Özet

Bazı yer tutucuları eksik çeviri

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.placeholders.PlaceholderCheck`

Flag to enable

`placeholders`

Flag to ignore

`ignore-placeholders`

4.3 sürümünde değişti: Düzenli ifadeyi yer tutucu olarak kullanabilirsiniz.

4.13 sürümünde değişti: With the `case-insensitive` flag, the placeholders are not case-sensitive.

Translation is missing some placeholders. These are either extracted from the translation file or defined manually using `placeholders` flag, more can be separated with colon, strings with space can be quoted:

```
placeholders:$URL$: $TARGET$: "some long text"
```

In case you have some syntax for placeholders, you can use a regular expression:

```
placeholders:r"%[^% ]%"
```

You can also have case insensitive placeholders:

```
placeholders:$URL$: $TARGET$, case-insensitive
```

Ayrıca bakınız:

Customizing behavior using flags

Noktalama aralığı

3.9 sürümünde geldi.

Özet

Çift noktalama işaretinden önce kırılmaz boşluk eksik

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.PunctuationSpacingCheck`

Flag to ignore

`ignore-punctuation-spacing`

Checks that there is non breakable space before double punctuation sign (exclamation mark, question mark, semicolon and colon). This rule is used only in a few selected languages like French or Breton, where space before double punctuation sign is a typographic rule.

Ayrıca bakınız:

[French and English spacing on Wikipedia](#)

Düzenli ifade

3.9 sürümünde geldi.

Özet

Çeviri düzenli ifadeyle eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.placeholders.RegexCheck`

Flag to enable

`regex`

Flag to ignore

`ignore-regex`

Translation does not match regular expression. The expression is either extracted from the translation file or defined manually using `regex` flag:

```
regex: ^foo|bar$
```

Aynı çoğullar

Özet

Bazı çoğul biçimler aynı şekilde çevrilmiş

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.consistency.SamePluralsCheck`

Flag to ignore

`ignore-same-plurals`

Check that fails if some plural forms are duplicated in the translation. In most languages they have to be different.

Baştaki yeni satır

Özet

Kaynağın ve çevirinin ikisinde yeni satır ile başlamıyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.BeginNewlineCheck`

Flag to ignore

`ignore-begin-newline`

Newlines usually appear in source strings for good reason, omissions or additions can lead to formatting problems when the translated text is put to use.

Ayrıca bakınız:

Sondaki yeni satır

Baştaki boşluk

Özet

Kaynağın ve çevirinin ikisinde aynı boşluk sayısıyla başlamıyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.BeginSpaceCheck`

Flag to ignore

`ignore-begin-space`

A space in the beginning of a string is usually used for indentation in the interface and thus important to keep.

Sondaki yeni satır

Özet

Kaynağın ve çevirinin ikisinde yeni satır ile bitmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.EndNewlineCheck`

Flag to ignore

`ignore-end-newline`

Newlines usually appear in source strings for good reason, omissions or additions can lead to formatting problems when the translated text is put to use.

Ayrıca bakınız:

Baştaki yeni satır

Sondaki boşluk

Özet

Kaynağın ve çevirinin ikisinde boşluk ile bitmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.EndSpaceCheck`

Flag to ignore

`ignore-end-space`

Checks that trailing spaces are replicated between both source and translation.

Trailing space is usually utilized to space out neighbouring elements, so removing it might break layout.

Değiştirilmemiş çeviri

Özet

Kaynak ve çeviri aynı

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.same.SameCheck`

Flag to ignore

`ignore-same`

Happens if the source and corresponding translation strings is identical, down to at least one of the plural forms. Some strings commonly found across all languages are ignored, and various markup is stripped. This reduces the number of false positives.

This check can help find strings mistakenly untranslated.

The default behavior of this check is to exclude words from the built-in blacklist from the checking. These are words which are frequently not being translated. This is useful to avoid false positives on short strings, which consist only of single word which is same in several languages. This blacklist can be disabled by adding `strict-same` flag to string or component.

Ayrıca bakınız:

Component configuration, Customizing behavior using flags

Güvenli olmayan HTML

3.9 sürümünde geldi.

Özet

Çeviri güvenli olmayan HTML işaretleme kullanıyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.markup.SafeHTMLCheck`

Flag to enable

`safe-html`

Flag to ignore

`ignore-safe-html`

The translation uses unsafe HTML markup. This check has to be enabled using `safe-html` flag (see *Customizing behavior using flags*). There is also accompanied autofixer which can automatically sanitize the markup.

İpucu: When `md-text` flag is also used, the Markdown style links are also allowed.

Ayrıca bakınız:

The HTML check is performed by the [Bleach](#) library developed by Mozilla.

URL

3.5 sürümünde geldi.

Özet

Çeviri bir URL içermiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.markup.URLCheck`

Flag to enable

`url`

Flag to ignore

`ignore-url`

The translation does not contain an URL. This is triggered only in case the unit is marked as containing URL. In that case the translation has to be a valid URL.

XML işaretlemesi

Özet

Çevirideki XML etiketleri kaynakla eşleşmiyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.markup.XMLTagsCheck`

Flag to ignore

`ignore-xml-tags`

This usually means the resulting output will look different. In most cases this is not a desired result from changing the translation, but occasionally it is.

Checks that XML tags are replicated between both source and translation.

Not: This check is disabled by the `safe-html` flag as the HTML cleanup done by it can produce HTML markup which is not valid XML.

XML sözdizimi

2.8 sürümünde geldi.

Özet

Çeviri geçerli XML değil

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.markup.XMLValidityCheck`

Flag to ignore

`ignore-xml-invalid`

The XML markup is not valid.

Not: This check is disabled by the `safe-html` flag as the HTML cleanup done by it can produce HTML markup which is not valid XML.

Sıfır genişlikli boşluk

Özet

Çeviri ekstra sıfır genişlikli boşluk karakteri içeriyor

Kapsam

çevrilmiş dizgiler

Check class

`weblate.checks.chars.ZeroWidthSpaceCheck`

Flag to ignore

`ignore-zero-width-space`

Zero-width space (<U+200B>) characters are used to break messages within words (word wrapping).

As they are usually inserted by mistake, this check is triggered once they are present in translation. Some programs might have problems when this character is used.

Ayrıca bakınız:

[Zero width space on Wikipedia](#)

1.6.4 Source checks

Source checks can help developers improve the quality of source strings.

Üç nokta

Özet

Dizgi, üç nokta karakteri (...) yerine üç tane nokta (...) kullanır

Kapsam

kaynak satırlar

Check class

`weblate.checks.source.EllipsisCheck`

Flag to ignore

`ignore-ellipsis`

This fails when the string uses three dots (. . .) when it should use an ellipsis character (...).

Using the Unicode character is in most cases the better approach and looks better rendered, and may sound better with text-to-speech.

Ayrıca bakınız:

[Ellipsis on Wikipedia](#)

ICU MessageFormat sözdizimi

4.9 sürümünde geldi.

Özet

ICU MessageFormat dizgilerinde sözdizimi hataları.

Kapsam

kaynak satırlar

Check class

`weblate.checks.icu.ICUSourceCheck`

Flag to enable

`icu-message-format`

Flag to ignore

`ignore-icu-message-format`

Ayrıca bakınız:

ICU MessageFormat

Uzun süre çevrilmeyen

4.1 sürümünde geldi.

Özet

Dizgi uzun zamandır çevrilmedi

Kapsam

kaynak satırlar

Check class

`weblate.checks.source.LongUntranslatedCheck`

Flag to ignore

`ignore-long-untranslated`

When the string has not been translated for a long time, it can indicate a problem in a source string making it hard to translate.

Birden çok kusur denetimi

Özet

Birkaç dildeki çevirilerin kusur denetimleri var

Kapsam

kaynak satırlar

Check class

`weblate.checks.source.MultipleFailingCheck`

Flag to ignore

`ignore-multiple-failures`

Numerous translations of this string have failing quality checks. This is usually an indication that something could be done to improve the source string.

This check failing can quite often be caused by a missing full stop at the end of a sentence, or similar minor issues which translators tend to fix in translation, while it would be better to fix it in the source string.

Birden çok adsız değişken

4.1 sürümünde geldi.

Özet

Dizgide birden çok adlandırılmamış değişken var, bu da çevirmenlerin bunları yeniden sıralamasını imkansız hale getirmektedir

Kapsam

kaynak satırlar

Check class

`weblate.checks.format.MultipleUnnamedFormatsCheck`

Flag to ignore

`ignore-unnamed-format`

There are multiple unnamed variables in the string, making it impossible for translators to reorder them.

Consider using named variables instead to allow translators to reorder them.

Çoğulsuz

Özet

Dizgi çoğul olarak kullanılır, ancak çoğul biçimleri kullanmaz

Kapsam

kaynak satırlar

Check class

`weblate.checks.source.OptionalPluralCheck`

Flag to ignore

`ignore-optional-plural`

The string is used as a plural, but does not use plural forms. In case your translation system supports this, you should use the plural aware variant of it.

For example with Gettext in Python it could be:

```
from gettext import ngettext
print(ngettext("Selected %d file", "Selected %d files", files) % files)
```

1.7 Searching

3.9 sürümünde geldi.

Advanced queries using boolean operations, parentheses, or field specific lookup can be used to find the strings you want.

When no field is defined, the lookup happens on source, target, and context strings.

Search

All strings ▾

Sort By ▾

Advanced query builder

Source strings ▾ Search for... ☐ Exact Add

String changed after ▾ mm/dd/yyyy ☐ Add

Query examples

Review strings changed by other users	<code>changed:>=2022-05-15 AND NOT changed_by:testuser</code>	Add
Translated strings	<code>state:>=translated</code>	Add
Strings with comments	<code>has:comment</code>	Add
Strings with any failing checks	<code>has:check</code>	Add
Strings with suggestions from others	<code>has:suggestion AND NOT suggestion_author:testuser</code>	Add
Approved strings with suggestions	<code>state:approved AND has:suggestion</code>	Add
All untranslated strings added the past month	<code>added:>=2022-05-15 AND state:<=needs-editing</code>	Add
Translated strings in a certain language	<code>is:translated AND language:cs</code>	Add

Search

Powered by Weblate 4.13 [About Weblate](#) [Legal](#) [Contact](#) [Documentation](#) [Donate to Weblate](#)

1.7.1 Simple search

Any phrase typed into the search box is split into words. Strings containing any of them are shown. To look for an exact phrase, put “the searchphrase” into quotes (both single (') and double (") quotes will work): "this is a quoted string" or 'another quoted string'.

1.7.2 Fields

source:TEXT

Source string case-insensitive search.

target:TEXT

Target string case-insensitive search.

context:TEXT

Context string case-insensitive search.

key:TEXT

Key string case-insensitive search.

note:TEXT

Source string description case-insensitive search.

location:TEXT

Location string case-insensitive search.

priority:NUMBER

String priority.

added:DATETIME

Timestamp for when the string was added to Weblate.

state:TEXT

State search (approved, translated, needs-editing, empty, read-only), supports *Field operators*.

pending:BOOLEAN

String pending for flushing to VCS.

has:TEXT

Search for string having attributes - plural, context, suggestion, comment, check, dismissed-check, translation, variant, screenshot, flags, explanation, glossary, note, label.

is:TEXT

Search for string states (pending, translated, untranslated).

language:TEXT

String target language.

component:TEXT

Component slug or name case-insensitive search, see *Component slug* and *Bileşen adı*.

project:TEXT

Project slug, see *URL kısaltması*.

changed_by:TEXT

String was changed by author with given username.

changed:DATETIME

String content was changed on date, supports *Field operators*.

change_time:DATETIME

String was changed on date, supports *Field operators*, unlike `changed` this includes event which don't change content and you can apply custom action filtering using `change_action`.

change_action:TEXT

Filters on change action, useful together with `change_time`. Accepts English name of the change action, either quoted and with spaces or lowercase and spaces replaced by a hyphen. See *Değişiklik arama* for examples.

check:TEXT

String has failing check.

dismissed_check:TEXT

String has dismissed check.

comment:TEXT

Search in user comments.

resolved_comment:TEXT

Search in resolved comments.

comment_author:TEXT

Filter by comment author.

suggestion:TEXT

Search in suggestions.

suggestion_author:TEXT

Filter by suggestion author.

explanation:METİN

Açıklamalarda ara.

label:TEXT

Search in labels.

screenshot:TEXT

Search in screenshots.

1.7.3 Boolean operators

You can combine lookups using AND, OR, NOT and parentheses to form complex queries. For example: `state:translated AND (source:hello OR source:bar)`

1.7.4 Field operators

You can specify operators, ranges or partial lookups for date or numeric searches:

state:> =translated

State is translated or better (approved).

changed:2019

Changed in year 2019.

changed:[2019-03-01 to 2019-04-01]

Changed between two given dates.

1.7.5 Exact operators

You can do an exact match query on different string fields using `=` operator. For example, to search for all source strings exactly matching `hello world`, use: `source:="hello world"`. For searching single word expressions, you can skip quotes. For example, to search for all source strings matching `hello`, you can use: `source:=hello`.

1.7.6 Değişiklik arama

4.4 sürümünde geldi.

Searching for history events can be done using `change_action` and `change_time` operators.

For example, searching for strings marked for edit in 2018 can be entered as `change_time:2018 AND change_action:marked-for-edit` or `change_time:2018 AND change_action:"Marked for edit"`.

1.7.7 Düzenli ifadeler

Anywhere text is accepted you can also specify a regular expression as `r"regexp"`.

For example, to search for all source strings which contain any digit between 2 and 5, use `source:r"[2-5]"`.

1.7.8 Predefined queries

You can select out of predefined queries on the search page, this allows you to quickly access the most frequent searches:

Weblate
 Dashboard Projects Languages Checks

WeblateOrg / Django / Czech / Translate
 translated 96%

1/1

Custom search '%(count)s word'

Zen

Position and priority

Translation

English

Singular
 %(count)s word

Plural
 %(count)s words

Czech, One

%(count)s slovo

Czech, Few

%(count)s slova

Czech, Other

%(count)s slov

Plural formula: (n==1) ? 0 : (n>=2 && n<=4) ? 1 : 2

☐ Needs editing

Save and continue

Save and stay

Suggest

Skip

15/140 · 15

14/140 · 15

Explanation
 No explanation currently provided.

Labels
 No labels currently set.

Flags
 python-format

Source string location
 weblate/templates/translation.html:149

String age
 6 seconds ago

Source string age
 6 seconds ago

Translation file
 weblate/locale/cs/LC_MESSAGE S/django.po, string 5

Nearby strings 20
 Comments
 Automatic suggestions
 Other languages 3

History

New comment

Comment on this string for fellow translators and developers to read.

Scope
 Translation comment, discussions with other translators

Is your comment specific to this translation, or generic for all of them?

New comment

You can use Markdown and mention users by @username.

Save

1.7.9 Ordering the results

There are many options to order the strings according to your needs:

The screenshot displays the Weblate web interface for the Django project, Czech language, and Translate view. The top navigation bar includes links for Dashboard, Projects, Languages, and Checks. The breadcrumb trail shows the path: WeblateOrg / Django / Czech / Translate. A status bar indicates that 96% of strings are translated.

The main content area shows a list of strings. The first string is "(...) instead of an ellipsis character (...)". A dropdown menu for "Position and priority" is open, showing the following options: Position and priority, Position, Priority, Labels, Source string, Target string, String age, Number of words, Number of comments, Number of failing checks, and Key. Below the string list, there are buttons for "Save and stay", "Suggest", and "Skip".

The right sidebar contains two sections: "Glossary" and "String information". The "Glossary" section shows a comparison between English and Czech, with a note that no related strings were found. The "String information" section provides details about the current string, including its source string location, age, and the translation file.

At the bottom of the interface, there is a "New comment" section for users to provide feedback or discuss the string. It includes a text area for the comment and a "Save" button.

1.8 Çeviri iş akışları

Using Weblate is a process that brings your users closer to you, by bringing you closer to your translators. It is up to you to decide how many of its features you want to make use of.

The following is not a complete list of ways to configure Weblate. You can base other workflows on the most usual examples listed here.

1.8.1 Translation access

The *access control* is not discussed in detail as a whole in the workflows, as most of its options can be applied to any workflow. Please consult the respective documentation on how to manage access to translations.

In the following chapters, *any user* means a user who has access to the translation. It can be any authenticated user if the project is public, or a user that has a *Translate* permission for the project.

1.8.2 Translation states

Each translated string can be in one of following states:

Çevrilmemiş

Translation is empty, it might or not be stored in the file, depending on the file format.

Düzenleme gerekli

Translation needs editing, this is usually the result of a source string change, fuzzy matching or translator action. The translation is stored in the file, depending on the file format it might be marked as needing edit (for example as it gets a `fuzzy` flag in the Gettext file).

Gözden geçirme için bekliyor

Translation is made, but not reviewed. It is stored in the file as a valid translation.

Onaylı

Translation has been approved in the review. It can no longer be changed by translators, but only by reviewers. Translators can only add suggestions to it.

This state is only available when reviews are enabled.

Öneriler

Suggestions are stored in Weblate only and not in the translation file.

The states are represented in the translation files when possible.

İpucu: In case file format you use does not support storing states, you might want to use *Değiştirilmemiş çevirileri “Düzenleme gerekli” olarak işaretle* add-on to flag unchanged strings as needing editing.

Ayrıca bakınız:

Translation types capabilities, Çeviri iş akışları

1.8.3 Direct translation

This is most usual setup for smaller teams, anybody can directly translate. This is also the default setup in Weblate.

- *Any user* can edit translations.
- Suggestions are optional ways to suggest changes, when translators are not sure about the change.

Setting	Value	Not
Gözden geçirmeleri etkinleştir	kapalı	Configured at project level.
Önerileri etkinleştir	açık	It is useful for users to be able to suggest when they are not sure.
Öneri oylaması	kapalı	
Önerileri kendiliğinden kabul et	0	
Translators group	<i>Kullanıcılar</i>	Ya da bkz <i>per-project access control</i> ile Çevir.
Reviewers group	YOK	Not used.

1.8.4 Peer review

With this workflow, anybody can add suggestions, and need approval from additional member(s) before it is accepted as a translation.

- *Any user* can add suggestions.
- *Any user* can vote for suggestions.
- Suggestions become translations when given a predetermined number of votes.

Setting	Value	Not
Gözden geçirmeleri etkinleştir	kapalı	Configured at project level.
Önerileri etkinleştir	açık	
Öneri oylaması	kapalı	
Önerileri kendiliğinden kabul et	1	You can set higher value to require more peer reviews.
Translators group	<i>Kullanıcılar</i>	Ya da bkz <i>per-project access control</i> ile Çevir.
Reviewers group	YOK	Not used, all translators review.

1.8.5 Dedicated reviewers

2.18 sürümünde geldi: The proper review workflow is supported since Weblate 2.18.

With dedicated reviewers you have two groups of users, one able to submit translations, and one able to review them to ensure translations are consistent and that the quality is good.

- *Any user* can edit unapproved translations.
- *Gözden geçiren* dizgileri onaylayabilir/onaylamayabilir.
- *Reviewer* can edit all translations (including approved ones).
- Suggestions can also be used to suggest changes for approved strings.

Setting	Value	Not
Gözden geçirmeleri etkinleştir	açık	Configured at project level.
Önerileri etkinleştir	kapalı	It is useful for users to be able to suggest when they are not sure.
Öneri oylaması	kapalı	
Önerileri kendiliğinden kabul et	0	
Translators group	<i>Kullanıcılar</i>	Ya da bkz <i>per-project access control</i> ile Çevir.
Reviewers group	<i>Gözden geçirenler</i>	Or Review with <i>per-project access control</i> .

1.8.6 Turning on reviews

Reviews can be turned on in the project configuration, from the *Workflow* subpage of project settings (to be found in the *Manage* → *Settings* menu):

The screenshot shows the Weblate interface with the 'Workflow' tab selected. The settings include:

- ☒ **Set "Language-Team" header** ⓘ
Lets Weblate update the "Language-Team" file header of your project.
- ☒ **Use shared translation memory** ⓘ
Uses the pool of shared translations between projects.
- ☒ **Contribute to shared translation memory** ⓘ
Contributes to the pool of shared translations between projects.
- ☒ **Enable hooks** ⓘ
Whether to allow updating this repository by remote hooks.
- Language aliases** ⓘ
Comma-separated list of language code mappings, for example: en_GB:en,en_US:en
- ☐ **Enable reviews** ⓘ
Requires dedicated reviewers to approve translations.
- ☐ **Enable source reviews** ⓘ
Requires dedicated reviewers to approve source strings.

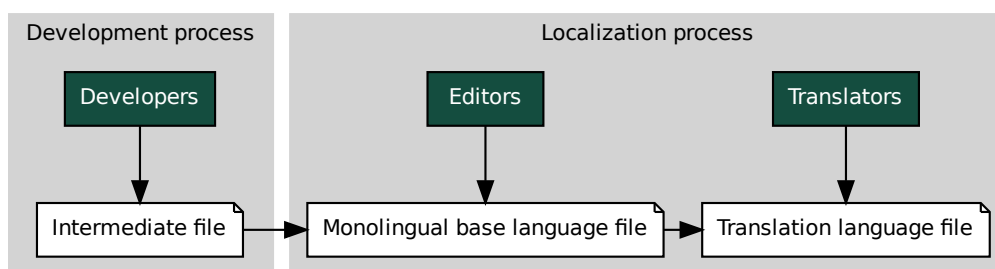
A 'Save' button is located at the bottom left of the settings area.

Not: Depending on Weblate configuration, the setting might not be available to you. For example on Hosted Weblate this is not available for projects hosted for free.

1.8.7 Quality gateway for the source strings

In many cases the original source language strings are coming from developers, because they write the code and provide initial strings. However developers are often not a native speakers in the source language and do not provide desired quality of the source strings. The intermediate translation can help you in addressing this - there is additional quality gateway for the strings between developers and translators and users.

By setting *Ara dil dosyası*, this file will be used as source for the strings, but it will be edited to source language to polish it. Once the string is ready in the source language, it will be also available for translators to translate into additional languages.



Ayrıca bakınız:

Ara dil dosyası, Tek dilli taban dil dosyası, Bilingual and monolingual formats

1.8.8 Source strings reviews

With *Kaynak gözden geçirmelerini etkinleştir* enabled, the review process can be applied on the source strings. Once enabled, users can report issues in the source strings. The actual process depends on whether you use bilingual or monolingual formats.

For monolingual formats, the source string review behaves similarly as with *Dedicated reviewers* - once issue is reported on the source string, it is marked as *Needs editing*.

The bilingual formats do not allow direct editing of the source strings (these are typically extracted directly from the source code). In this case *Source needs review* label is attached to strings reported by translators. You should review such strings and either edit them in the source or remove the label.

Ayrıca bakınız:

Bilingual and monolingual formats, Dedicated reviewers, labels, Yorumlar

1.9 Frequently Asked Questions

1.9.1 Yapılandırma

How to create an automated workflow?

Weblate can handle all the translation things semi-automatically for you. If you give it push access to your repository, the translations can happen without interaction, unless some merge conflict occurs.

1. Set up your Git repository to tell Weblate when there is any change, see *Bildirim kancaları* for info on how to do it.

2. Set a push URL at your *Component configuration* in Weblate, this allows Weblate to push changes to your repository.
3. Turn on *İşlemeye yolla* on your *Component configuration* in Weblate, this will make Weblate push changes to your repository whenever they happen at Weblate.

Ayrıca bakınız:

Sürekli yerelleştirme, Avoiding merge conflicts

How to access repositories over SSH?

Please see *Accessing repositories* for info on setting up SSH keys.

How to fix merge conflicts in translations?

Merge conflicts happen from time to time when the translation file is changed in both Weblate and the upstream repository concurrently. You can usually avoid this by merging Weblate translations prior to making changes in the translation files (e.g. before running msgmerge). Just tell Weblate to commit all pending translations (you can do it in *Repository maintenance* in the *Manage* menu) and merge the repository (if automatic push is not on).

If you've already encountered a merge conflict, the easiest way to solve all conflicts locally on your machine, is to add Weblate as a remote repository, merge it into upstream and fix any conflicts. Once you push changes back, Weblate will be able to use the merged version without any other special actions.

Not: Depending on your setup, access to the Weblate repository might require authentication. When using the built-in *Git exporter* in Weblate, you authenticate with your username and the API key.

```
# Commit all pending changes in Weblate, you can do this in the UI as well:
wlc commit
# Lock the translation in Weblate, again this can be done in the UI as well:
wlc lock
# Add Weblate as remote:
git remote add weblate https://hosted.weblate.org/git/project/component/
# You might need to include credentials in some cases:
git remote add weblate https://username:APIKEY@hosted.weblate.org/git/project/
↪component/

# Update weblate remote:
git remote update weblate

# Merge Weblate changes:
git merge weblate/main

# Resolve conflicts:
edit ...
git add ...
...
git commit

# Push changes to upstream repository, Weblate will fetch merge from there:
git push

# Open Weblate for translation:
wlc unlock
```

If you're using multiple branches in Weblate, you can do the same to all of them:

```
# Add and update Weblate remotes
git remote add weblate-one https://hosted.weblate.org/git/project/one/
git remote add weblate-second https://hosted.weblate.org/git/project/second/
git remote update weblate-one weblate-second

# Merge QA_4_7 branch:
git checkout QA_4_7
git merge weblate-one/QA_4_7
... # Resolve conflicts
git commit

# Merge main branch:
git checkout main
git merge weblate-second/main
... # Resolve conflicts
git commit

# Push changes to the upstream repository, Weblate will fetch the merge from there:
git push
```

In case of gettext PO files, there is a way to merge conflicts in a semi-automatic way:

Fetch and keep a local clone of the Weblate Git repository. Also get a second fresh local clone of the upstream Git repository (i. e. you need two copies of the upstream Git repository: An intact and a working copy):

```
# Add remote:
git remote add weblate /path/to/weblate/snapshot/

# Update Weblate remote:
git remote update weblate

# Merge Weblate changes:
git merge weblate/main

# Resolve conflicts in the PO files:
for PO in `find . -name '*.po'` ; do
    msgcat --use-first /path/to/weblate/snapshot/$PO\
                /path/to/upstream/snapshot/$PO -o $PO.merge
    msgmerge --previous --lang=${PO%.po} $PO.merge domain.pot -o $PO
    rm $PO.merge
    git add $PO
done
git commit

# Push changes to the upstream repository, Weblate will fetch merge from there:
git push
```

Ayrıca bakınız:

How to export the Git repository that Weblate uses?, Sürekli yerelleştirme, Avoiding merge conflicts, Weblate Client

How do I translate several branches at once?

Weblate supports pushing translation changes within one *Project configuration*. For every *Component configuration* which has it turned on (the default behavior), the change made is automatically propagated to others. This way translations are kept synchronized even if the branches themselves have already diverged quite a lot, and it is not possible to simply merge translation changes between them.

Once you merge changes from Weblate, you might have to merge these branches (depending on your development workflow) discarding differences:

```
git merge -s ours origin/maintenance
```

Ayrıca bakınız:

Keeping translations same across components

How to translate multi-platform projects?

Weblate supports a wide range of file formats (see *Supported file formats*) and the easiest approach is to use the native format for each platform.

Once you have added all platform translation files as components in one project (see *Adding translation projects and components*), you can utilize the translation propagation feature (turned on by default, and can be turned off in the *Component configuration*) to translate strings for all platforms at once.

Ayrıca bakınız:

Keeping translations same across components

How to export the Git repository that Weblate uses?

There is nothing special about the repository, it lives under the `DATA_DIR` directory and is named `vcs/<project>/<component>/`. If you have SSH access to this machine, you can use the repository directly.

For anonymous access, you might want to run a Git server and let it serve the repository to the outside world.

Alternatively, you can use *Git exporter* inside Weblate to automate this.

What are the options for pushing changes back upstream?

This heavily depends on your setup, Weblate is quite flexible in this area. Here are examples of some workflows used with Weblate:

- Weblate automatically pushes and merges changes (see *How to create an automated workflow?*).
- You manually tell Weblate to push (it needs push access to the upstream repository).
- Somebody manually merges changes from the Weblate git repository into the upstream repository.
- Somebody rewrites history produced by Weblate (e.g. by eliminating merge commits), merges changes, and tells Weblate to reset the content in the upstream repository.

Of course you are free to mix all of these as you wish.

How can I limit Weblate access to only translations, without exposing source code to it?

You can use [git submodule](#) for separating translations from source code while still having them under version control.

1. Create a repository with your translation files.
2. Add this as a submodule to your code:

```
git submodule add git@example.com:project-translations.git path/to/translations
```

3. Link Weblate to this repository, it no longer needs access to the repository containing your source code.
4. You can update the main repository with translations from Weblate by:

```
git submodule update --remote path/to/translations
```

Please consult the [git submodule](#) documentation for more details.

How can I check whether my Weblate is set up properly?

Weblate includes a set of configuration checks which you can see in the admin interface, just follow the *Performance report* link in the admin interface, or open the `/manage/performance/` URL directly.

Ayrıca bakınız:

[Monitoring Weblate](#), [Monitoring Celery status](#)

Why are all commits committed by Weblate <noreply@weblate.org>?

This is the default committer name, configured by `DEFAULT_COMMITTER_EMAIL` and `DEFAULT_COMMITTER_NAME`.

The author of every commit (if the underlying VCS supports it) is still recorded correctly as the user that made the translation.

For commits where no authorship is known (for example anonymous suggestions or machine translation results), the authorship is credited to the anonymous user (see [ANONYMOUS_USER_NAME](#)). You can change the name and e-mail in the management interface.

Ayrıca bakınız:

[Component configuration](#)

How to move files in the repository without losing history in Weblate?

To keep the history, comments, or screenshots linked to strings after changing the files location you need to ensure that these strings are never deleted in Weblate. These removals can happen in case the Weblate repository is updated, but the component configuration still points to the old files. This makes Weblate assume that it should delete all the translations.

The solution to this is to perform the operation in sync with Weblate:

1. Lock the affected component in Weblate.
2. Commit any pending changes and merge them into the upstream repository.
3. Disable receiving webhooks the *[Project configuration](#)*; this prevents Weblate from immediately seeing changes in the repository.
4. Do any needed changes in the repo (for example using `git mv`), push them to the upstream repository.
5. Change the *[Component configuration](#)* to match the new setup; upon changing configuration, Weblate will fetch the updated repository and notice the changed locations while keeping existing strings.

6. Unlock the component and re-enable hooks in the project configuration.

1.9.2 Usage

How do I review the translations of others?

- There are several review based workflows available in Weblate, see [Çeviri iş akışları](#).
- You can subscribe to any changes made in [Bildirimler](#) and then check others contributions as they come in by e-mail.
- There is a review tool available at the bottom of the translation view, where you can choose to browse translations made by others since a given date.

Ayrıca bakınız:

[Çeviri iş akışları](#)

How do I provide feedback on a source string?

On context tabs below translation, you can use the *Comments* tab to provide feedback on a source string, or discuss it with other translators.

Ayrıca bakınız:

report-source, [Yorumlar](#)

How can I use existing translations while translating?

- Paylaşılan çeviri belleği sayesinde, Weblate içindeki tüm çeviriler kullanılabilir.
- Mevcut çeviri belleği dosyalarını Weblate'e aktarabilirsiniz.
- Use the import functionality to load compendium as translations, suggestions or translations needing review. This is the best approach for a one-time translation using a compendium or a similar translation database.
- You can set up [tmserver](#) with all databases you have and let Weblate use it. This is good when you want to use it several times during translation.
- Another option is to translate all related projects in a single Weblate instance, which will make it automatically pick up translations from other projects as well.

Ayrıca bakınız:

[Configuring automatic suggestions](#), [Otomatik öneriler](#), [Çeviri Belleği](#)

Does Weblate update translation files besides translations?

Weblate tries to limit changes in translation files to a minimum. For some file formats it might unfortunately lead to reformatting the file. If you want to keep the file formatted your way, please use a pre-commit hook for that.

Ayrıca bakınız:

updating-target-files

Where do language definitions come from and how can I add my own?

The basic set of language definitions is included within Weblate and Translate-toolkit. This covers more than 150 languages and includes info about plural forms or text direction.

You are free to define your own languages in the administrative interface, you just need to provide info about it.

Ayrıca bakınız:

Dil tanımları

Can Weblate highlight changes in a fuzzy string?

Weblate supports this, however it needs the data to show the difference.

For Gettext PO files, you have to pass the parameter `--previous` to **msgmerge** when updating PO files, for example:

```
msgmerge --previous -U po/cs.po po/phpmyadmin.pot
```

For monolingual translations, Weblate can find the previous string by ID, so it shows the differences automatically.

Why does Weblate still show old translation strings when I've updated the template?

Weblate does not try to manipulate the translation files in any way other than allowing translators to translate. So it also does not update the translatable files when the template or source code have been changed. You simply have to do this manually and push changes to the repository, Weblate will then pick up the changes automatically.

Not: It is usually a good idea to merge changes done in Weblate before updating translation files, as otherwise you will usually end up with some conflicts to merge.

For example with gettext PO files, you can update the translation files using the **msgmerge** tool:

```
msgmerge -U locale/cs/LC_MESSAGES/django.mo locale/django.pot
```

In case you want to do the update automatically, you can install add-on *POT ile eşleşmesi için PO dosyalarını güncelle (msgmerge)*.

Ayrıca bakınız:

updating-target-files

1.9.3 Troubleshooting

Requests sometimes fail with “too many open files” error

This happens sometimes when your Git repository grows too much and you have many of them. Compressing the Git repositories will improve this situation.

The easiest way to do this is to run:

```
# Go to DATA_DIR directory
cd data/vcs
# Compress all Git repositories
for d in */* ; do
    pushd $d
    git gc
    popd
done
```

Ayrıca bakınız:[DATA_DIR](#)**When accessing the site I get a “Bad Request (400)” error**

This is most likely caused by an improperly configured `ALLOWED_HOSTS`. It needs to contain all hostnames you want to access on your Weblate. For example:

```
ALLOWED_HOSTS = ["weblate.example.com", "weblate", "localhost"]
```

Ayrıca bakınız:[Allowed hosts setup](#)**What does mean “There are more files for the single language (en)”?**

This typically happens when you have translation file for source language. Weblate keeps track of source strings and reserves source language for this. The additional file for same language is not processed.

- Kaynak dile çevirinin istenmesi durumunda, lütfen proje ayarlarındaki *Kaynak dil* seçeneğini değiştirin.
- Kaynak dilin çeviri dosyasına gerek olmaması durumunda, lütfen bunu depodan kaldırın.
- Kaynak dilin çeviri dosyasına ihtiyaç duyulduğu, ancak Weblate tarafından yok sayılmasının gerektiği durumda, lütfen hariç tutmak için *Dil süzgeci* seçeneğini ayarlayın.

İpucu: You might get similar error message for other languages as well. In that case the most likely reason is that several files map to single language in Weblate.

This can be caused by using obsolete language codes together with new one (j a and j p for Japanese) or including both country specific and generic codes (f r and f r_FR). See *Dil kodlarını ayırıştırma* for more details.

1.9.4 Özellikler

Does Weblate support other VCSes than Git and Mercurial?

Weblate currently does not have native support for anything other than *Git* (with extended support for *GitHub pull requests*, *Gerrit* and *Subversion*) and *Mercurial*, but it is possible to write backends for other VCSes.

You can also use *Git remote helpers* in Git to access other VCSes.

Weblate also supports VCS-less operation, see *Local files*.

Not: For native support of other VCSes, Weblate requires using distributed VCS, and could probably be adjusted to work with anything other than Git and Mercurial, but somebody has to implement this support.

Ayrıca bakınız:[Sürüm denetimi tümleşimi](#)

How does Weblate credit translators?

Every change made in Weblate is committed into VCS under the translators name. This way every single change has proper authorship, and you can track it down using the standard VCS tools you use for code.

Additionally, when the translation file format supports it, the file headers are updated to include the translator's name.

Ayrıca bakınız:

`list_translators`, `../devel/reporting`

Why does Weblate force showing all PO files in a single tree?

Weblate was designed in a way that every PO file is represented as a single component. This is beneficial for translators, so they know what they are actually translating.

4.2 sürümünde değişti: Çevirmenler, bir projenin tüm bileşenlerini bir bütün olarak belirli bir dile çevirebilir.

Why does Weblate use language codes such `sr_Latn` or `zh_Hant`?

These are language codes defined by [RFC 5646](#) to better indicate that they are really different languages instead previously wrongly used modifiers (for `@latin` variants) or country codes (for Chinese).

Weblate still understands legacy language codes and will map them to current one - for example `sr@latin` will be handled as `sr_Latn` or `zh@CN` as `zh_Hans`.

Not: Weblate defaults to POSIX style language codes with underscore, see [Dil tanımları](#) for more details.

Ayrıca bakınız:

[Dil tanımları](#), [Dil kodu stili](#), [Adding new translations](#)

1.10 Supported file formats

Weblate supports most translation format understood by [translate-toolkit](#), however each format being slightly different, some issues with formats that are not well tested can arise.

Ayrıca bakınız:

[Translation Related File Formats](#)

Not: When choosing a file format for your application, it's better to stick some well established format in the toolkit/platform you use. This way your translators can additionally use whatever tools they are used to, and will more likely contribute to your project.

1.10.1 Bilingual and monolingual formats

Both monolingual and bilingual formats are supported. Bilingual formats store two languages in single file—source and translation (typical examples are *GNU gettext*, *XLIFF* or *Apple iOS strings*). On the other side, monolingual formats identify the string by ID, and each language file contains only the mapping of those to any given language (typically *Android string resources*). Some file formats are used in both variants, see the detailed description below.

For correct use of monolingual files, Weblate requires access to a file containing complete list of strings to translate with their source—this file is called *Tek dilli taban dil dosyası* within Weblate, though the naming might vary in your paradigm.

Additionally this workflow can be extended by utilizing *Ara dil dosyası* to include strings provided by developers, but not to be used as is in the final strings.

1.10.2 Kendiliğinden algılama

Weblate can automatically detect several widespread file formats, but this detection can harm your performance and will limit features specific to given file format (for example automatic addition of new translations).

1.10.3 Translation types capabilities

Capabilities of all supported formats:

Format	Lingu- ality ^{Sayfa 76, 1}	Plu- rals ^{Sayfa 76, 2}	Descripti- ons ^{Sayfa 76, 3}	Conte- xt ^{Sayfa 76, 4}	Locati- on ^{Sayfa 76, 5}	Flags ^{Sayfa 76, 8}	Additi- onal sta- tes ^{Sayfa 76, 6}
<i>GNU gettext</i>	bilingual	yes	yes	yes	yes	yes ⁹	needs edi- ting
<i>Mono- lingual gettext</i>	mono	yes	yes	yes	yes	yes ⁹	needs edi- ting
<i>XLIFF</i>	both	yes	yes	yes	yes	yes ¹⁰	needs editing, approved
<i>Java properties</i>	both	no	yes	no	no	no	
<i>mi18n lang dosyaları</i>	mono	no	yes	no	no	no	
<i>GWT özel- likleri</i>	mono	yes	yes	no	no	no	
<i>Joomla translati- ons</i>	mono	no	yes	no	yes	no	
<i>Qt Linguist .ts</i>	both	yes	yes	no	yes	yes ^{Sayfa 76, 10}	needs edi- ting
<i>Android string resources</i>	mono	yes	yes ⁷	no	no	yes ^{Sayfa 76, 10}	
<i>Apple iOS strings</i>	both	no	yes	no	no	no	
<i>PHP dizgi- leri</i>	mono	no ¹¹	yes	no	no	no	
<i>JSON files</i>	mono	no	no	no	no	no	
<i>JSON i18next files</i>	mono	yes	no	no	no	no	
<i>go-i18n JSON files</i>	mono	yes	no	no	no	no	
<i>ARB File</i>	mono	yes	yes	no	no	no	
<i>WebEx- tension JSON</i>	mono	yes	yes	no	no	no	
<i>.XML reso- urce files</i>	mono	no	yes	no	no	yes ^{Sayfa 76, 10}	

sonraki sayfaya devam

Tablo 1 – önceki sayfadan devam

Format	Lingu- ality ¹ <small>Sayfa 76, 1</small>	Plurals ²	Descripti- ons ³	Context ⁴	Location ⁵	Flags ⁸	Additional states ⁶
<i>Resource- Dictionary files</i>	mono	no	no	no	no	yes ⁹ <small>Sayfa 76, 10</small>	
<i>CSV files</i>	both	no	yes	yes	yes	no	needs edi- ting
<i>YAML files</i>	mono	no	yes	no	no	no	
<i>Ruby YAML files</i>	mono	yes	yes	no	no	no	
<i>DTD files</i>	mono	no	no	no	no	no	
<i>Flat XML files</i>	mono	no	no	no	no	yes ¹⁰	
<i>Windows RC files</i>	mono	no	yes	no	no	no	
<i>Excel Open XML</i>	mono	no	yes	yes	yes	no	needs edi- ting
<i>Uygulama mağazası üst veri dosyaları</i>	mono	no	no	no	no	no	
<i>Subtitle fi- les</i>	mono	no	no	no	yes	no	
<i>HTML files</i>	mono	no	no	no	no	no	
<i>OpenDo- cument Format</i>	mono	no	no	no	no	no	
<i>IDML For- mat</i>	mono	no	no	no	no	no	
<i>INI transla- tions</i>	mono	no	no	no	no	no	
<i>Inno Setup INI çeviri- leri</i>	mono	no	no	no	no	no	
<i>TermBase eXchange biçimi</i>	bilingual	no	yes	no	no	yes ¹⁰	
<i>Metin dos- yaları</i>	mono	no	no	no	no	no	
<i>Stringsdict biçimi</i>	mono	yes	yes	no	no	no	
<i>Fluent biçimi</i>	mono	no ¹²	yes	no	no	no	

¹ See *Bilingual and monolingual formats*² Plurals are necessary to properly localize strings with variable count.³ Source string descriptions can be used to pass additional info about the string to translate.⁴ Context is used to differentiate identical strings used in different scopes (for example *Sun* can be used as an abbreviated name of the day “Sunday” or as the name of our closest star).⁵ Location of a string in source code might help proficient translators figure out how the string is used.⁸ See *Customizing behavior using flags*⁶ Additional states supported by the file format in addition to “Untranslated” and “Translated”.⁹ The gettext type comments are used as flags.¹⁰ The flags are extracted from the non-standard attribute `weblate-flags` for all XML based formats. Additionally `max-length:N` is supported through the `maxwidth` attribute as defined in the XLIFF standard, see *Specifying translation flags*.⁷ XML comment placed before the `<string>` element, parsed as a source string description.¹¹ The plurals are supported only for Laravel which uses in string syntax to define them, see *Localization in Laravel*.¹² Plurals are handled in the syntax of the strings and not exposed as plurals in Weblate.

Salt okunur dizgiler

3.10 sürümünde geldi.

Read-only strings from translation files will be included, but can not be edited in Weblate. This feature is natively supported by few formats (*XLIFF* and *Android string resources*), but can be emulated in others by adding a read-only flag, see *Customizing behavior using flags*.

1.10.4 GNU gettext

Özgür yazılım çevirmek için en yaygın olarak kullanılan biçim.

Contextual info stored in the file is supported by adjusting its headers or linking to corresponding source files.

The bilingual gettext PO file typically looks like this:

```
#: weblate/media/js/bootstrap-datepicker.js:1421
msgid "Monday"
msgstr "Pondělí"

#: weblate/media/js/bootstrap-datepicker.js:1421
msgid "Tuesday"
msgstr "Úterý"

#: weblate/accounts/avatar.py:163
msgctxt "No known user"
msgid "None"
msgstr "Žádný"
```

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	po/* .po
Tek dilli taban dil dosyası	<i>Empty</i>
Yeni çeviriler için şablon	po/messages .pot
Dosya biçimi	<i>Gettext PO file</i>

Ayrıca bakınız:

devel/gettext, devel/sphinx, [Gettext on Wikipedia](#), [PO Files](#), “*configure*” dosyasındaki *ALL_LINGUAS* değişkenini güncelle, *Gettext* çıktısını özelleştir, *LINGUAS* dosyasını güncelle, *MO* dosyaları üret, *POT* ile eşleşmesi için *PO* dosyalarını güncelle (*msgmerge*)

Monolingual gettext

Some projects decide to use gettext as monolingual formats—they code just the IDs in their source code and the string then needs to be translated to all languages, including English. This is supported, though you have to choose this file format explicitly when importing components into Weblate.

The monolingual gettext PO file typically looks like this:

```
#: weblate/media/js/bootstrap-datepicker.js:1421
msgid "day-monday"
msgstr "Pondělí"

#: weblate/media/js/bootstrap-datepicker.js:1421
msgid "day-tuesday"
msgstr "Úterý"

#: weblate/accounts/avatar.py:163
msgid "none-user"
msgstr "Žádný"
```

While the base language file will be:

```
#: weblate/media/js/bootstrap-datepicker.js:1421
msgid "day-monday"
msgstr "Monday"

#: weblate/media/js/bootstrap-datepicker.js:1421
msgid "day-tuesday"
msgstr "Tuesday"

#: weblate/accounts/avatar.py:163
msgid "none-user"
msgstr "None"
```

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	po/* .po
Tek dilli taban dil dosyası	po/en .po
Yeni çeviriler için şablon	po/messages .pot
Dosya biçimi	<i>Gettext PO file (monolingual)</i>

1.10.5 XLIFF

XML-based format created to standardize translation files, but in the end it is one of [many standards](#), in this area.

XML Localization Interchange File Format (XLIFF) is usually used as bilingual, but Weblate supports it as monolingual as well.

Weblate supports XLIFF in several variants:

XLIFF translation file

Simple XLIFF file where content of the elements is stored as plain text (all XML elements being escaped).

XLIFF with placeables support

Standard XLIFF supporting placeables and other XML elements.

XLIFF with gettext extensions

XLIFF enriched by [XLIFF 1.2 Representation Guide for Gettext PO](#) to support plurals.

Ayrıca bakınız:

[XML Localization Interchange File Format \(XLIFF\) specification](#), [XLIFF 1.2 Representation Guide for Gettext PO](#)

Translation states

3.3 sürümünde değişti: Weblate ignored the `state` attribute prior to the 3.3 release.

The `state` attribute in the file is partially processed and mapped to the “Needs edit” state in Weblate (the following states are used to flag the string as needing edit if there is a target present: `new`, `needs-translation`, `needs-adaptation`, `needs-l10n`). Should the `state` attribute be missing, a string is considered translated as soon as a `<target>` element exists.

If the translation string has `approved = "yes"`, it will also be imported into Weblate as “Approved”, anything else will be imported as “Waiting for review” (which matches the XLIFF specification).

While saving, Weblate doesn’t add those attributes unless necessary:

- The `state` attribute is only added in case string is marked as needing edit.
- The `approved` attribute is only added in case string has been reviewed.
- In other cases the attributes are not added, but they are updated in case they are present.

That means that when using the XLIFF format, it is strongly recommended to turn on the Weblate review process, in order to see and change the approved state of strings.

Similarly upon importing such files (in the upload form), you should choose *Import as translated* under *Processing of strings needing edit*.

Ayrıca bakınız:

Dedicated reviewers

Whitespace and newlines in XLIFF

Generally types or amounts of whitespace is not differentiated between in XML formats. If you want to keep it, you have to add the `xml:space="preserve"` flag to the string.

For example:

```
<trans-unit id="10" approved="yes">
  <source xml:space="preserve">hello</source>
  <target xml:space="preserve">Hello, world!
</target>
</trans-unit>
```

Specifying translation flags

You can specify additional translation flags (see *Customizing behavior using flags*) by using the `weblate-flags` attribute. Weblate also understands `maxwidth` and `font` attributes from the XLIFF specification:

```
<trans-unit id="10" maxwidth="100" size-unit="pixel" font="ubuntu;22:bold">
  <source>Hello %s</source>
</trans-unit>
<trans-unit id="20" maxwidth="100" size-unit="char" weblate-flags="c-format">
  <source>Hello %s</source>
</trans-unit>
```

The `font` attribute is parsed for font family, size and weight, the above example shows all of that, though only font family is required. Any whitespace in the font family is converted to underscore, so `Source Sans Pro` becomes `Source_Sans_Pro`, please keep that in mind when naming the font group (see *Managing fonts*).

Dizgi anahtarları

Weblate identifies the units in the XLIFF file by `resname` attribute in case it is present and falls back to `id` (together with `file` tag if present).

The `resname` attribute is supposed to be human friendly identifier of the unit making it more suitable for Weblate to display instead of `id`. The `resname` has to be unique in the whole XLIFF file. This is required by Weblate and is not covered by the XLIFF standard - it does not put any uniqueness restrictions on this attribute.

Typical Weblate <i>Component configuration</i> for bilingual XLIFF	
Dosya maskesi	<code>localizations/*.xliff</code>
Tek dilli taban dil dosyası	<i>Empty</i>
Yeni çeviriler için şablon	<code>localizations/en-US.xliff</code>
Dosya biçimi	<i>XLIFF Translation File</i>

Typical Weblate <i>Component configuration</i> for monolingual XLIFF	
Dosya maskesi	localizations/*.xliff
Tek dilli taban dil dosyası	localizations/en-US.xliff
Yeni çeviriler için şablon	localizations/en-US.xliff
Dosya biçimi	<i>XLIFF Translation File</i>

Ayrıca bakınız:

[XLIFF on Wikipedia](#), [XLIFF](#), [font attribute in XLIFF 1.2](#), [maxwidth attribute in XLIFF 1.2](#)

1.10.6 Java properties

Native Java format for translations.

Java properties are usually used as monolingual translations.

Weblate supports ISO-8859-1, UTF-8 and UTF-16 variants of this format. All of them support storing all Unicode characters, it is just differently encoded. In the ISO-8859-1, the Unicode escape sequences are used (for example `zkou\u0161ka`), all others encode characters directly either in UTF-8 or UTF-16.

Not: Loading escape sequences works in UTF-8 mode as well, so please be careful choosing the correct encoding set to match your application needs.

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	src/app/Bundle_*.properties
Tek dilli taban dil dosyası	src/app/Bundle.properties
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>Java Properties (ISO-8859-1)</i>

Ayrıca bakınız:

[Java properties on Wikipedia](#), [Mozilla and Java properties files](#), [mi18n lang dosyaları](#), [GWT özellikleri](#), [updating-target-files](#), [Format the Java properties file](#), [Çeviri dosyaslarını temizle](#)

1.10.7 mi18n lang dosyaları

4.7 sürümünde geldi.

File format used for JavaScript localization by [mi18n](#). Syntactically it matches *Java properties*.

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	*.lang
Tek dilli taban dil dosyası	en-US.lang
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>mi18n lang dosyası</i>

Ayrıca bakınız:

[mi18n](#), [Mozilla and Java properties files](#), [Java properties](#), [updating-target-files](#), [Format the Java properties file](#), [Çeviri dosyaslarını temizle](#)

1.10.8 GWT özellikleri

Native GWT format for translations.

GWT properties are usually used as monolingual translations.

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	src/app/Bundle_*.properties
Tek dilli taban dil dosyası	src/app/Bundle.properties
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>GWT Properties</i>

Ayrıca bakınız:

GWT localization guide, GWT Internationalization Tutorial, Mozilla and Java properties files, updating-target-files, *Format the Java properties file*, *Çeviri dosyaslarını temizle*

1.10.9 INI translations

4.1 sürümünde geldi.

INI file format for translations.

INI translations are usually used as monolingual translations.

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	language/*.ini
Tek dilli taban dil dosyası	language/en.ini
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>INI File</i>

Not: Weblate only extracts keys from sections within an INI file. In case your INI file lacks sections, you might want to use *Joomla translations* or *Java properties* instead.

Ayrıca bakınız:

INI Files, *Java properties*, *Joomla translations*, *Inno Setup INI çevirileri*

1.10.10 Inno Setup INI çevirileri

4.1 sürümünde geldi.

Çeviriler için Inno Setup INI dosya biçimi.

Inno Setup INI çevirileri genellikle tek dilli çeviriler olarak kullanılır.

Not: The only notable difference to *INI translations* is in supporting %n and %t placeholders for line break and tab.

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	language/*.isl
Tek dilli taban dil dosyası	language/en.isl
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>Inno Setup INI Dosyası</i>

Not: Only Unicode files (`.isl`) are currently supported, ANSI variant (`.isl`) is currently not supported.

Ayrıca bakınız:

INI Files, *Joomla translations*, *INI translations*

1.10.11 Joomla translations

2.12 sürümünde geldi.

Native Joomla format for translations.

Joomla translations are usually used as monolingual translations.

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	language/*/com_foobar.ini
Tek dilli taban dil dosyası	language/en-GB/com_foobar.ini
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>Joomla Language File</i>

Ayrıca bakınız:

Mozilla and Java properties files, *INI translations*, *Inno Setup INI çevirileri*

1.10.12 Qt Linguist .ts

Translation format used in Qt based applications.

Qt Linguist files are used as both bilingual and monolingual translations.

Typical Weblate <i>Component configuration</i> when using as bilingual	
Dosya maskesi	i18n/app.*.ts
Tek dilli taban dil dosyası	<i>Empty</i>
Yeni çeviriler için şablon	i18n/app.de.ts
Dosya biçimi	<i>Qt Linguist Translation File</i>

Typical Weblate <i>Component configuration</i> when using as monolingual	
Dosya maskesi	i18n/app.*.ts
Tek dilli taban dil dosyası	i18n/app.en.ts
Yeni çeviriler için şablon	i18n/app.en.ts
Dosya biçimi	<i>Qt Linguist Translation File</i>

Ayrıca bakınız:

Qt Linguist manual, Qt .ts, *Bilingual and monolingual formats*

1.10.13 Android string resources

Android specific file format for translating applications.

Android string resources are monolingual, the *Tek dilli taban dil dosyası* is stored in a different location from the other files – `res/values/strings.xml`.

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	<code>res/values-*/strings.xml</code>
Tek dilli taban dil dosyası	<code>res/values/strings.xml</code>
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>Android String Resource</i>

Ayrıca bakınız:

[Android string resources documentation](#), [Android string resources](#)

Not: Android *string-array* structures are not currently supported. To work around this, you can break your string arrays apart:

```
<string-array name="several_strings">
  <item>First string</item>
  <item>Second string</item>
</string-array>
```

become:

```
<string-array name="several_strings">
  <item>@string/several_strings_0</item>
  <item>@string/several_strings_1</item>
</string-array>
<string name="several_strings_0">First string</string>
<string name="several_strings_1">Second string</string>
```

The *string-array* that points to the *string* elements should be stored in a different file, and not be made available for translation.

This script may help pre-process your existing strings.xml files and translations: <https://gist.github.com/paour/11291062>

1.10.14 Apple iOS strings

File format typically used for translating Apple iOS applications, but also standardized by PWG 5100.13 and used on NeXTSTEP/OpenSTEP.

Apple iOS strings are usually used as monolingual.

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	<code>Resources/*.lproj/Localizable.strings</code>
Tek dilli taban dil dosyası	<code>Resources/en.lproj/Localizable.strings</code> or <code>Resources/Base.lproj/Localizable.strings</code>
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>iOS Strings (UTF-8)</i>

Ayrıca bakınız:

Stringsdict biçimi, Apple “strings files” documentation, Message Catalog File Format in PWG 5100.13, Mac OSX strings

1.10.15 PHP dizgileri

PHP translations are usually monolingual, so it is recommended to specify a base file with (what is most often the) English strings.

Example file:

```
<?php
$LANG['foo'] = 'bar';
$LANG['foo1'] = 'foo bar';
$LANG['foo2'] = 'foo bar baz';
$LANG['foo3'] = 'foo bar baz bag';
```

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	lang/*/texts.php
Tek dilli taban dil dosyası	lang/en/texts.php
Yeni çeviriler için şablon	lang/en/texts.php
Dosya biçimi	<i>PHP strings</i>

Laravel PHP dizgileri

4.1 sürümünde değişti.

The Laravel PHP localization files are supported as well with plurals:

```
<?php
return [
    'welcome' => 'Welcome to our application',
    'apples' => 'There is one apple|There are many apples',
];
```

Ayrıca bakınız:

PHP, Localization in Laravel

1.10.16 JSON files

2.0 sürümünde geldi.

2.16 sürümünde değişti: Since Weblate 2.16 and with *translate-toolkit* at-least 2.2.4, nested structure JSON files are supported as well.

4.3 sürümünde değişti: The structure of JSON file is properly preserved even for complex situations which were broken in prior releases.

JSON format is used mostly for translating applications implemented in JavaScript.

Weblate currently supports several variants of JSON translations:

- Simple key / value files, used for example by *vue-i18n* or *react-intl*.
- Files with nested keys.
- *JSON i18next files*
- *go-i18n JSON files*

- *WebExtension JSON*
- *ARB File*

JSON translations are usually monolingual, so it is recommended to specify a base file with (what is most often the) English strings.

Example file:

```
{
  "Hello, world!\n": "Ahoj světe!\n",
  "Orangutan has %d banana.\n": "",
  "Try Weblate at https://demo.weblate.org/!\n": "",
  "Thank you for using Weblate.": ""
}
```

Nested files are supported as well (see above for requirements), such a file can look like:

```
{
  "weblate": {
    "hello": "Ahoj světe!\n",
    "orangutan": "",
    "try": "",
    "thanks": ""
  }
}
```

İpucu: The *JSON file* and *JSON nested structure file* can both handle same type of files. Both preserve existing JSON structure when translating.

The only difference between them is when adding new strings using Weblate. The nested structure format parses the newly added key and inserts the new string into the matching structure. For example `app.name` key is inserted as:

```
{
  "app": {
    "name": "Weblate"
  }
}
```

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	langs/translation-*.json
Tek dilli taban dil dosyası	langs/translation-en.json
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>JSON nested structure file</i>

Ayrıca bakınız:

JSON, *updating-target-files*, *JSON çıktısını özelleştir*, *Çeviri dosyaslarını temizle*,

1.10.17 JSON i18next files

2.17 sürümünde değişti: Since Weblate 2.17 and with [translate-toolkit](#) at-least 2.2.5, i18next JSON files with plurals are supported as well.

[i18next](#) is an internationalization framework written in and for JavaScript. Weblate supports its localization files with features such as plurals.

i18next translations are monolingual, so it is recommended to specify a base file with (what is most often the) English strings.

Not: Weblate supports the i18next JSON v3 format. The v2 and v1 variants are mostly compatible, with exception of how plurals are handled.

The v4 variant uses different approach for storing plurals and is currently not supported.

Example file:

```
{
  "hello": "Hello",
  "apple": "I have an apple",
  "apple_plural": "I have {{count}} apples",
  "apple_negative": "I have no apples"
}
```

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	langs/*.json
Tek dilli taban dil dosyası	langs/en.json
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>i18next JSON file</i>

Ayrıca bakınız:

[JSON](#), [i18next JSON Format](#), [updating-target-files](#), [JSON çıktısını özelleştir](#), [Çeviri dosyalarını temizle](#)

1.10.18 go-i18n JSON files

4.1 sürümünde geldi.

go-i18n translations are monolingual, so it is recommended to specify a base file with (what is most often the) English strings.

Not: Weblate supports the go-i18n JSON v1 format, for flat JSON formats please use [JSON files](#). The v2 format with hash is currently not supported.

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	langs/*.json
Tek dilli taban dil dosyası	langs/en.json
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>go-i18n JSON file</i>

Ayrıca bakınız:

[JSON](#), [go-i18n](#), [updating-target-files](#), [JSON çıktısını özelleştir](#), [Çeviri dosyalarını temizle](#),

1.10.19 ARB File

4.1 sürümünde geldi.

ARB translations are monolingual, so it is recommended to specify a base file with (what is most often the) English strings.

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	lib/l10n/intl_*.arb
Tek dilli taban dil dosyası	lib/l10n/intl_en.arb
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>ARB file</i>

Ayrıca bakınız:

JSON, Application Resource Bundle Specification, Internationalizing Flutter apps, updating-target-files, *JSON çıktısını özelleştir, Çeviri dosyaslarını temizle*

1.10.20 WebExtension JSON

2.16 sürümünde geldi: This is supported since Weblate 2.16 and with [translate-toolkit](#) at-least 2.2.4.

File format used when translating extensions for Mozilla Firefox or Google Chromium.

Not: While this format is called JSON, its specification allows to include comments, which are not part of JSON specification. Weblate currently does not support file with comments.

Example file:

```
{
  "hello": {
    "message": "Ahoj světe!\n",
    "description": "Description",
    "placeholders": {
      "url": {
        "content": "$1",
        "example": "https://developer.mozilla.org"
      }
    }
  },
  "orangutan": {
    "message": "Orangutan has $coUnT$ bananas",
    "description": "Description",
    "placeholders": {
      "count": {
        "content": "$1",
        "example": "5"
      }
    }
  },
  "try": {
    "message": "",
    "description": "Description"
  },
  "thanks": {
    "message": "",
    "description": "Description"
  }
}
```

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	<code>_locales/*/messages.json</code>
Tek dilli taban dil dosyası	<code>_locales/en/messages.json</code>
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>WebExtension JSON file</i>

Ayrıca bakınız:

[JSON](#), [Google chrome.i18n](#), [Mozilla Extensions Internationalization](#)

1.10.21 .XML resource files

2.3 sürümünde geldi.

A .XML resource (.resx) file employs a monolingual XML file format used in Microsoft .NET applications. It is interchangeable with .resw, when using identical syntax to .resx.

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	<code>Resources/Language.*.resx</code>
Tek dilli taban dil dosyası	<code>Resources/Language.resx</code>
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>.NET kaynak dosyası</i>

Ayrıca bakınız:

[.NET Resource files \(.resx\)](#), [updating-target-files](#), [Çeviri dosyaslarını temizle](#)

1.10.22 ResourceDictionary files

4.13 sürümünde geldi.

ResourceDictionary is a monolingual XML file format used to package localizable string resources for Windows Presentation Foundation (WPF) applications.

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	<code>Languages/*.xaml</code>
Tek dilli taban dil dosyası	<code>Language/en.xaml</code>
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>ResourceDictionary file</i>

Ayrıca bakınız:

[Flat XML](#), [Flat XML files](#), [updating-target-files](#), [Çeviri dosyaslarını temizle](#)

1.10.23 CSV files

2.4 sürümünde geldi.

CSV files can contain a simple list of source and translation. Weblate supports the following files:

- Files with header defining fields (location, source, target, ID, fuzzy, context, translator_comments, developer_comments). This is the recommended approach, as it is the least error prone. Choose *CSV file* as a file format.
- Files with two fields—source and translation (in this order). Choose *Simple CSV file* as a file format.

- Headerless files with fields in order defined by the [translate-toolkit](#): location, source, target, ID, fuzzy, context, translator_comments, developer_comments. Choose *CSV file* as a file format.
- Remember to define *Tek dilli taban dil dosyası* when your files are monolingual (see *Bilingual and monolingual formats*).

İpucu: By default, the CSV format does autodetection of file encoding. This can be unreliable in some corner cases and causes performance penalty. Please choose file format variant with encoding to avoid this (for example *CSV file (UTF-8)*).

Uyarı: The CSV format currently automatically detects the dialect of the CSV file. In some cases the automatic detection might fail and you will get mixed results. This is especially true for CSV files with newlines in the values. As a workaround it is recommended to omit quoting characters.

Example file:

Thank you for using Weblate.,Děkujeme za použití Weblate.

Typical Weblate <i>Component configuration</i> for bilingual CSV	
Dosya maskesi	locale/*.csv
Tek dilli taban dil dosyası	<i>Empty</i>
Yeni çeviriler için şablon	locale/en.csv
Dosya biçimi	<i>CSV file</i>

Typical Weblate <i>Component configuration</i> for monolingual CSV	
Dosya maskesi	locale/*.csv
Tek dilli taban dil dosyası	locale/en.csv
Yeni çeviriler için şablon	locale/en.csv
Dosya biçimi	<i>Basit CSV dosyası</i>

Multivalue CSV file

4.13 sürümünde geldi.

This variant of the CSV files allows storing multiple translations per string.

Ayrıca bakınız:

[CSV](#)

1.10.24 YAML files

2.9 sürümünde geldi.

The plain YAML files with string keys and values. Weblate also extract strings from lists or dictionaries.

Example of a YAML file:

```
weblate:
  hello: ""
  orangutan: ""
  try: ""
  thanks: ""
```

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	translations/messages.*.yaml
Tek dilli taban dil dosyası	translations/messages.en.yaml
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>YAML file</i>

Ayrıca bakınız:

[YAML](#), *Ruby YAML files*

1.10.25 Ruby YAML files

2.9 sürümünde geldi.

Ruby i18n YAML files with language as root node.

Example Ruby i18n YAML file:

```
cs:
  weblate:
    hello: ""
    orangutan: ""
    try: ""
    thanks: ""
```

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	translations/messages.*.yaml
Tek dilli taban dil dosyası	translations/messages.en.yaml
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>Ruby YAML file</i>

Ayrıca bakınız:

[YAML](#), *YAML files*

1.10.26 DTD files

2.18 sürümünde geldi.

Example DTD file:

```
<!ENTITY hello "">
<!ENTITY orangutan "">
<!ENTITY try "">
<!ENTITY thanks "">
```

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	locale/*.dtd
Tek dilli taban dil dosyası	locale/en.dtd
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>DTD file</i>

Ayrıca bakınız:

[Mozilla DTD format](#)

1.10.27 Flat XML files

3.9 sürümünde geldi.

Example of a flat XML file:

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<root>
  <str key = "hello_world">Hello World!</str>
  <str key = "resource_key">Translated value.</str>
</root>
```

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	locale/*.xml
Tek dilli taban dil dosyası	locale/en.xml
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>Flat XML file</i>

Ayrıca bakınız:

[Flat XML](#)

1.10.28 Windows RC files

4.1 sürümünde değişti: Support for Windows RC files has been rewritten.

Not: Support for this format is currently in beta, feedback from testing is welcome.

Example Windows RC file:

```
LANGUAGE LANG_CZECH, SUBLANG_DEFAULT

STRINGTABLE
BEGIN
    IDS_MSG1          "Hello, world!\n"
    IDS_MSG2          "Orangutan has %d banana.\n"
    IDS_MSG3          "Try Weblate at http://demo.weblate.org/!\n"
    IDS_MSG4          "Thank you for using Weblate."
END
```

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	lang/*.rc
Tek dilli taban dil dosyası	lang/en-US.rc
Yeni çeviriler için şablon	lang/en-US.rc
Dosya biçimi	<i>RC file</i>

Ayrıca bakınız:

[Windows RC files](#)

1.10.29 Uygulama mağazası üst veri dosyaları

3.5 sürümünde geldi.

Metadata used for publishing apps in various app stores can be translated. Currently the following tools are compatible:

- [Triple-T gradle-play-publisher](#)
- [Fastlane](#)
- [F-Droid](#)

The metadata consists of several textfiles, which Weblate will present as separate strings to translate.

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	fastlane/android/metadata/*
Tek dilli taban dil dosyası	fastlane/android/metadata/en-US
Yeni çeviriler için şablon	fastlane/android/metadata/en-US
Dosya biçimi	<i>App store metadata files</i>

İpucu: In case you don't want to translate certain strings (for example changelogs), mark them read-only (see [Customizing behavior using flags](#)). This can be automated by the [Toplu düzenleme](#).

1.10.30 Subtitle files

3.7 sürümünde geldi.

Weblate çeşitli altyazı dosyalarını çevirebilir:

- SubRip subtitle file (*.srt)
- MicroDVD subtitle file (*.sub)
- Advanced Substation Alpha subtitles file (*.ass)
- Substation Alpha subtitle file (*.ssa)

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	path/*.srt
Tek dilli taban dil dosyası	path/en.srt
Yeni çeviriler için şablon	path/en.srt
Dosya biçimi	<i>SubRip subtitle file</i>

Ayrıca bakınız:

[Subtitles](#)

1.10.31 Excel Open XML

3.2 sürümünde geldi.

Excel Open XML (.xlsx) files can be imported and exported.

When uploading XLSX files for translation, be aware that only the active worksheet is considered, and there must be at least a column called `source` (which contains the source string) and a column called `target` (which contains the translation). Additionally there should be the column called `context` (which contains the context path of the translation string). If you use the XLSX download for exporting the translations into an Excel workbook, you already get a file with the correct file format.

1.10.32 HTML files

4.1 sürümünde geldi.

Not: Support for this format is currently in beta, feedback from testing is welcome.

The translatable content is extracted from the HTML files and offered for the translation.

Ayrıca bakınız:

[HTML](#)

1.10.33 Metin dosyaları

4.6 sürümünde geldi.

Not: Support for this format is currently in beta, feedback from testing is welcome.

The translatable content is extracted from the plain text files and offered for the translation. Each paragraph is translated as a separate string.

Bu biçimin üç çeşidi vardır:

- Düz metin dosyası
- DokuWiki metin dosyası
- MediaWiki metin dosyası

Ayrıca bakınız:

[Simple Text Documents](#)

1.10.34 OpenDocument Format

4.1 sürümünde geldi.

Not: Support for this format is currently in beta, feedback from testing is welcome.

The translatable content is extracted from the OpenDocument files and offered for the translation.

Ayrıca bakınız:

[OpenDocument Format](#)

1.10.35 IDML Format

4.1 sürümünde geldi.

Not: Support for this format is currently in beta, feedback from testing is welcome.

The translatable content is extracted from the Adobe InDesign Markup Language files and offered for the translation.

1.10.36 TermBase eXchange biçimi

4.5 sürümünde geldi.

TBX is an XML format for the exchange of terminology data.

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	tbx/* .tbx
Tek dilli taban dil dosyası	<i>Empty</i>
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>TermBase eXchange dosyası</i>

Ayrıca bakınız:

[TBX on Wikipedia](#), [TBX](#), [Sözlük](#)

1.10.37 Stringsdict biçimi

4.8 sürümünde geldi.

Not: Support for this format is currently in beta, feedback from testing is welcome.

XML based format used by Apple which is able to store plural forms of a string.

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	Resources/* .lproj/Localizable.stringsdict
Tek dilli taban dil dosyası	Resources/en.lproj/Localizable.stringsdict or Resources/Base.lproj/Localizable.stringsdict
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>Stringsdict dosyası</i>

Ayrıca bakınız:

[Apple iOS strings](#), [Stringsdict File Format](#)

1.10.38 Fluent biçimi

4.8 sürümünde geldi.

Not: Support for this format is currently in beta, feedback from testing is welcome.

Fluent is a monolingual text format that focuses on asymmetric localization: a simple string in one language can map to a complex multi-variant translation in another language.

Typical Weblate <i>Component configuration</i>	
Dosya maskesi	locales/* /messages.ftl
Tek dilli taban dil dosyası	locales/en/messages.ftl
Yeni çeviriler için şablon	<i>Empty</i>
Dosya biçimi	<i>Fluent file</i>

Ayrıca bakınız:

[Project Fluent website](#)

1.10.39 Diğer biçimleri destekleme

Most formats supported by [translate-toolkit](#) which support serializing can be easily supported, but they did not (yet) receive any testing. In most cases some thin layer is needed in Weblate to hide differences in behavior of different [translate-toolkit](#) storages.

To add support for a new format, the preferred approach is to first implement support for it in the [translate-toolkit](#).

Ayrıca bakınız:

[Translation Related File Formats](#)

1.11 Sürüm denetimi tümleşimi

Weblate currently supports [Git](#) (with extended support for [GitHub pull requests](#), [Gerrit](#) and [Subversion](#)) and [Mercurial](#) as version control back-ends.

1.11.1 Accessing repositories

The VCS repository you want to use has to be accessible to Weblate. With a publicly available repository you just need to enter the correct URL (for example <https://github.com/WeblateOrg/weblate.git>), but for private repositories or for push URLs the setup is more complex and requires authentication.

Accessing repositories from Hosted Weblate

For Hosted Weblate there is a dedicated push user registered on GitHub, Bitbucket, Codeberg and GitLab (with the username *weblate*, e-mail hosted@weblate.org and, named *Weblate push user*). You need to add this user as a collaborator and give it appropriate permission to your repository (read-only is okay for cloning, write is required for pushing). Depending on service and your organization settings, this happens immediately, or requires confirmation on the Weblate side.

The *weblate* user on GitHub accepts invitations automatically within five minutes. Manual processing might be needed on the other services, so please be patient.

Once the *weblate* user is added, you can configure *Kaynak kod deposu* and *Depo yollama URL'si* using the SSH protocol (for example [git@github.com:WeblateOrg/weblate.git](ssh://git@github.com:WeblateOrg/weblate.git)).

SSH repositories

The most frequently used method to access private repositories is based on SSH. Authorize the public Weblate SSH key (see [Weblate SSH key](#)) to access the upstream repository this way.

Uyarı: On GitHub, each key can only be used once, see [GitHub repositories](#) and [Accessing repositories from Hosted Weblate](#).

Weblate also stores the host key fingerprint upon first connection, and fails to connect to the host should it be changed later (see [Verifying SSH host keys](#)).

In case adjustment is needed, do so from the Weblate admin interface:

Weblate
Dashboard
Projects
Languages
Checks

+

Manage / SSH keys

Weblate status
Backups
Translation memory
Performance report
SSH keys
Alerts
Repositories
Users
Appearance
Tools
Billing

Public SSH key

Weblate uses SSH key to access remote repositories. The corresponding public key is found below, you can use it to grant Weblate access to a repository.

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQDCy3+xmtE5n9jvn2Jlr8dQPnOWIVZ7RBIgaVcLFv/n5SvQ3F8WYvQms0Ket1nOpK70ZaZQ6f14VmnCxxv/jjnNwhXzDyqVbxW
Weblate
```

Download private key

Known host keys

Hostname	Key type	Fingerprint
github.com	ssh-ed25519	+DIY3wwV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU
github.com	ssh-rsa	nThbg6kXUpJWGI7E1IGOCspRomTxdCARLviKw6E5SY8
github.com	ecdsa-sha2-nistp256	p2QAMXNIC1TJYWeIOttrVc98/R1BUFWu3/LiyKgUfQM

Add host key

To access SSH hosts, its host key needs to be verified. You can get the host key by entering a domain name or IP for the host in the form below.

Hostname
Port

Submit

Powered by Weblate 4.13
About Weblate
Legal
Contact
Documentation
Donate to Weblate

Weblate SSH key

The Weblate public key is visible to all users browsing the *About* page.

Admins can generate or display the public key currently used by Weblate in the connection (from *SSH keys*) on the admin interface landing page.

Not: The corresponding private SSH key can not currently have a password, so make sure it is well protected.

İpucu: Make a backup of the generated private Weblate SSH key.

Verifying SSH host keys

Weblate automatically stores the SSH host keys on first access and remembers them for further use.

In case you want to verify the key fingerprint before connecting to the repository, add the SSH host keys of the servers you are going to access in *Add host key*, from the same section of the admin interface. Enter the hostname you are going to access (e.g. `gitlab.com`), and press *Submit*. Verify its fingerprint matches the server you added.

The added keys with fingerprints are shown in the confirmation message:

Weblate
 Dashboard Projects Languages Checks

Manage / SSH keys

Added host key for github.com with fingerprint nThbg6kXUpJWGI7E1IGOCspRomTxdCARLviKw6E5SY8 (ssh-rsa), please verify that it is correct.

Added host key for github.com with fingerprint p2QAMXNIC1TJYWeIOttrVc98/R1BUFWu3/LiyKgUfQM (ecdsa-sha2-nistp256), please verify that it is correct.

Added host key for github.com with fingerprint +DiY3wvW6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU (ssh-ed25519), please verify that it is correct.

Weblate status Backups Translation memory Performance report **SSH keys** Alerts Repositories Users Appearance

Tools Billing

Public SSH key ⓘ

Weblate uses SSH key to access remote repositories. The corresponding public key is found below, you can use it to grant Weblate access to a repository.

ssh-rsa
 AAAAB3NzaC1yc2EAAAADAQABAAQDCy3+xmtE5n9jvn2JIr8dQPnOWIVZ7RBIgAVcLFv/n5SvQ3F8WYvQms0Ket1nOpK70ZaZQ6f14VmnCxxv/jjnNwhXzDyqVbxW
 Weblate

Download private key

Known host keys ⓘ

Hostname	Key type	Fingerprint
github.com	ssh-ed25519	+DiY3wvW6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU
github.com	ssh-rsa	nThbg6kXUpJWGI7E1IGOCspRomTxdCARLviKw6E5SY8
github.com	ecdsa-sha2-nistp256	p2QAMXNIC1TJYWeIOttrVc98/R1BUFWu3/LiyKgUfQM

Add host key ⓘ

To access SSH hosts, its host key needs to be verified. You can get the host key by entering a domain name or IP for the host in the form below.

Hostname Port

Submit

GitHub repositories

Access via SSH is possible (see [SSH repositories](#)), but in case you need to access more than one repository, you will hit a GitHub limitation on allowed SSH key usage (since each key can be used only once).

In case the *Yollama dali* is not set, the project is forked and changes pushed through a fork. In case it is set, changes are pushed to the upstream repository and chosen branch.

For smaller deployments, use HTTPS authentication with a personal access token and your GitHub account, see [Creating an access token for command-line use](#).

For bigger setups, it is usually better to create a dedicated user for Weblate, assign it the public SSH key generated in Weblate (see [Weblate SSH key](#)) and grant it access to all the repositories you want to translate. This approach is also used for Hosted Weblate, there is dedicated *weblate* user for that.

Ayrıca bakınız:

[Accessing repositories from Hosted Weblate](#)

Weblate internal URLs

Share one repository setup between different components by referring to its placement as `weblate://project/component` in other(linked) components. This way linked components use the VCS repository configuration of the main(referenced) component.

Uyarı: Removing main component also removes linked components.

Weblate automatically adjusts the repository URL when creating a component if it finds a component with a matching repository setup. You can override this in the last step of the component configuration.

Reasons to use this:

- Saves disk space on the server, the repository is stored just once.
- Makes the updates faster, only one repository is updated.
- There is just single exported repository with Weblate translations (see [Git exporter](#)).
- Some add-ons can operate on multiple components sharing one repository, for example [Git işlemlerini sıkıştır](#).

HTTPS repositories

To access protected HTTPS repositories, include the username and password in the URL. Don't worry, Weblate will strip this info when the URL is shown to users (if even allowed to see the repository URL at all).

For example the GitHub URL with authentication added might look like: `https://user:your_access_token@github.com/WeblateOrg/weblate.git`.

Not: If your username or password contains special characters, those have to be URL encoded, for example `https://user%40example.com:%24password%23@bitbucket.org/....`

Using proxy

If you need to access HTTP/HTTPS VCS repositories using a proxy server, configure the VCS to use it.

This can be done using the `http_proxy`, `https_proxy`, and `all_proxy` environment variables, (as described in the [cURL documentation](#)) or by enforcing it in the VCS configuration, for example:

```
git config --global http.proxy http://user:password@proxy.example.com:80
```

Not: The proxy configuration needs to be done under user running Weblate (see also [Filesystem permissions](#)) and with `HOME = $DATA_DIR/home` (see [DATA_DIR](#)), otherwise Git executed by Weblate will not use it.

Ayrıca bakınız:

The [cURL manpage](#), [Git config documentation](#)

1.11.2 Git

İpucu: Weblate needs Git 2.12 or newer.

Ayrıca bakınız:

See [Accessing repositories](#) for info on how to access different kinds of repositories.

Git (zorlama push ile)

This behaves exactly like Git itself, the only difference being that it always force pushes. This is intended only in the case of using a separate repository for translations.

Uyarı: Use with caution, as this easily leads to lost commits in your upstream repository.

Customizing Git configuration

Weblate invokes all VCS commands with `HOME = $DATA_DIR/home` (see [DATA_DIR](#)), therefore editing the user configuration needs to be done in `DATA_DIR/home/.git`.

Git remote helpers

You can also use Git [remote helpers](#) for additionally supporting other version control systems, but be prepared to debug problems this may lead to.

At this time, helpers for Bazaar and Mercurial are available within separate repositories on GitHub: [git-remote-hg](#) and [git-remote-bzr](#). Download them manually and put somewhere in your search path (for example `~/bin`). Make sure you have the corresponding version control systems installed.

Once you have these installed, such remotes can be used to specify a repository in Weblate.

To clone the `gnuhello` project from Launchpad using Bazaar:

```
bzr::lp:gnuhello
```

For the `hello` repository from `selenic.com` using Mercurial:

```
hg::http://selenic.com/repo/hello
```

Uyarı: The inconvenience of using Git remote helpers is for example with Mercurial, the remote helper sometimes creates a new tip when pushing changes back.

1.11.3 GitHub pull requests

2.3 sürümünde geldi.

This adds a thin layer atop *Git* using the *GitHub API* to allow pushing translation changes as pull requests, instead of pushing directly to the repository.

Git pushes changes directly to a repository, while *GitHub pull requests* creates pull requests. The latter is not needed for merely accessing Git repositories.

You need to configure API credentials (*GITHUB_CREDENTIALS*) in the Weblate settings to make this work. Once configured, you will see a *GitHub* option when selecting *Sürüm denetleme sistemi*.

Ayrıca bakınız:

Pushing changes from Weblate, GITHUB_USERNAME, GITHUB_TOKEN, GITHUB_CREDENTIALS

1.11.4 GitLab merge requests

3.9 sürümünde geldi.

This just adds a thin layer atop *Git* using the *GitLab API* to allow pushing translation changes as merge requests instead of pushing directly to the repository.

There is no need to use this to access Git repositories, ordinary *Git* works the same, the only difference is how pushing to a repository is handled. With *Git* changes are pushed directly to the repository, while *GitLab merge requests* creates merge request.

You need to configure API credentials (*GITLAB_CREDENTIALS*) in the Weblate settings to make this work. Once configured, you will see a *GitLab* option when selecting *Sürüm denetleme sistemi*.

Ayrıca bakınız:

Pushing changes from Weblate, GITLAB_USERNAME, GITLAB_TOKEN, GITLAB_CREDENTIALS

1.11.5 Gitea pull requests

4.12 sürümünde geldi.

This just adds a thin layer atop *Git* using the *Gitea API* to allow pushing translation changes as pull requests instead of pushing directly to the repository.

There is no need to use this to access Git repositories, ordinary *Git* works the same, the only difference is how pushing to a repository is handled. With *Git* changes are pushed directly to the repository, while *Gitea pull requests* creates pull requests.

You need to configure API credentials (*GITEA_CREDENTIALS*) in the Weblate settings to make this work. Once configured, you will see a *Gitea* option when selecting *Sürüm denetleme sistemi*.

Ayrıca bakınız:

Pushing changes from Weblate, GITEA_USERNAME, GITEA_TOKEN, GITEA_CREDENTIALS

1.11.6 Pagure merge requests

4.3.2 sürümünde geldi.

This just adds a thin layer atop [Git](#) using the [Pagure API](#) to allow pushing translation changes as merge requests instead of pushing directly to the repository.

There is no need to use this to access Git repositories, ordinary [Git](#) works the same, the only difference is how pushing to a repository is handled. With [Git](#) changes are pushed directly to the repository, while [Pagure merge requests](#) creates merge request.

You need to configure API credentials (`PAGURE_CREDENTIALS`) in the Weblate settings to make this work. Once configured, you will see a *Pagure* option when selecting *Sürüm denetleme sistemi*.

Ayrıca bakınız:

Pushing changes from Weblate, `PAGURE_USERNAME`, `PAGURE_TOKEN`, `PAGURE_CREDENTIALS`

1.11.7 Gerrit

2.2 sürümünde geldi.

Adds a thin layer atop [Git](#) using the [git-review](#) tool to allow pushing translation changes as Gerrit review requests, instead of pushing them directly to the repository.

The Gerrit documentation has the details on the configuration necessary to set up such repositories.

1.11.8 Mercurial

2.1 sürümünde geldi.

Mercurial is another VCS you can use directly in Weblate.

Not: It should work with any Mercurial version, but there are sometimes incompatible changes to the command-line interface which breaks Weblate integration.

Ayrıca bakınız:

See *Accessing repositories* for info on how to access different kinds of repositories.

1.11.9 Subversion

2.8 sürümünde geldi.

Weblate uses [git-svn](#) to interact with [subversion](#) repositories. It is a Perl script that lets subversion be used by a Git client, enabling users to maintain a full clone of the internal repository and commit locally.

Not: Weblate tries to detect Subversion repository layout automatically - it supports both direct URLs for branch or repositories with standard layout (branches/, tags/ and trunk/). More info about this is to be found in the [git-svn documentation](#). If your repository does not have a standard layout and you encounter errors, try including the branch name in the repository URL and leaving branch empty.

2.19 sürümünde değişti: Before this, only repositories using the standard layout were supported.

Subversion credentials

Weblate expects you to have accepted the certificate up-front (and your credentials if needed). It will look to insert them into the `DATA_DIR` directory. Accept the certificate by using `svn` once with the `$HOME` environment variable set to the `DATA_DIR`:

```
# Use DATA_DIR as configured in Weblate settings.py, it is /app/data in the Docker
HOME=${DATA_DIR}/home svn co https://svn.example.com/example
```

Ayrıca bakınız:

`DATA_DIR`

1.11.10 Local files

1.11.11 Git

İpucu: Underneath, this uses [Git](#). It requires Git installed and allows you to switch to using Git natively with full history of your translations.

3.8 sürümünde geldi.

Weblate can also operate without a remote VCS. The initial translations are imported by uploading them. Later you can replace individual files by file upload, or add translation strings directly from Weblate (currently available only for monolingual translations).

In the background Weblate creates a Git repository for you and all changes are tracked in. In case you later decide to use a VCS to store the translations, you already have a repository within Weblate can base your integration on.

1.12 Weblate's REST API

2.6 sürümünde geldi: The REST API is available since Weblate 2.6.

The API is accessible on the `/api/` URL and it is based on [Django REST framework](#). You can use it directly or by [Weblate Client](#).

1.12.1 Authentication and generic parameters

The public project API is available without authentication, though unauthenticated requests are heavily throttled (by default to 100 requests per day), so it is recommended to use authentication. The authentication uses a token, which you can get in your profile. Use it in the `Authorization` header:

ANY /

Generic request behaviour for the API, the headers, status codes and parameters here apply to all endpoints as well.

Query Parameters

- **format** – Response format (overrides [Accept](#)). Possible values depends on REST framework setup, by default `json` and `api` are supported. The latter provides web browser interface for API.
- **page** – Returns given page of paginated results (use `next` and `previous` fields in response to automate the navigation).

Request Headers

- [Accept](#) – the response content type depends on [Accept](#) header

- **Authorization** – optional token to authenticate as `Authorization: Token YOUR-TOKEN`

Response Headers

- **Content-Type** – this depends on **Accept** header of request
- **Allow** – list of allowed HTTP methods on object

Response JSON Object

- **detail** (*string*) – verbose description of the result (for HTTP status codes other than 200 OK)
- **count** (*int*) – total item count for object lists
- **next** (*string*) – next page URL for object lists
- **previous** (*string*) – previous page URL for object lists
- **results** (*array*) – results for object lists
- **url** (*string*) – URL to access this resource using API
- **web_url** (*string*) – URL to access this resource using web browser

Status Codes

- 200 OK – when request was correctly handled
- 201 Created – when a new object was created successfully
- 204 No Content – when an object was deleted successfully
- 400 Bad Request – when form parameters are missing
- 403 Forbidden – when access is denied
- 429 Too Many Requests – when throttling is in place

Authentication tokens

4.10 sürümünde değişti: Project scoped tokens were introduced in the 4.10 release.

Each user has his personal access token which can be obtained in the user profile. Newly generated user tokens have the `wlu_` prefix.

It is possible to create project scoped tokens for API access to given project only. These tokens can be identified by the `wlp_` prefix.

Authentication examples

Example request:

```
GET /api/ HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
Authorization: Token YOUR-TOKEN
```

Example response:

```
HTTP/1.0 200 OK
Date: Fri, 25 Mar 2016 09:46:12 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
Content-Language: en
Allow: GET, HEAD, OPTIONS

{
  "projects": "http://example.com/api/projects/",
  "components": "http://example.com/api/components/",
  "translations": "http://example.com/api/translations/",
  "languages": "http://example.com/api/languages/"
}
```

CURL example:

```
curl \
  -H "Authorization: Token TOKEN" \
  https://example.com/api/
```

Passing Parameters Examples

For the **POST** method the parameters can be specified either as form submission (*application/x-www-form-urlencoded*) or as JSON (*application/json*).

Form request example:

```
POST /api/projects/hello/repository/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/x-www-form-urlencoded
Authorization: Token TOKEN

operation =pull
```

JSON request example:

```
POST /api/projects/hello/repository/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20

{"operation": "pull"}
```

CURL example:

```
curl \
  -d operation=pull \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/components/hello/weblate/repository/
```

CURL JSON example:

```
curl \
  --data-binary '{"operation": "pull"}' \
  -H "Content-Type: application/json" \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/components/hello/weblate/repository/
```

API rate limiting

The API requests are rate limited; the default configuration limits it to 100 requests per day for anonymous users and 5000 requests per hour for authenticated users.

Rate limiting can be adjusted in the `settings.py`; see [Throttling in Django REST framework documentation](#) for more details how to configure it.

In the Docker container this can be configured using `WEBLATE_API_RATELIMIT_ANON` and `WEBLATE_API_RATELIMIT_USER`.

The status of rate limiting is reported in following headers:

X-RateLimit-Limit	Rate limiting limit of requests to perform
X-RateLimit-Remaining	Remaining limit of requests
X-RateLimit-Reset	Number of seconds until ratelimit window resets

4.1 sürümünde değişti: Added ratelimiting status headers.

Ayrıca bakınız:

Rate limiting, *Rate limiting*, `WEBLATE_API_RATELIMIT_ANON`, `WEBLATE_API_RATELIMIT_USER`

1.12.2 API Entry Point

GET /api/

The API root entry point.

Example request:

```
GET /api/ HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
Authorization: Token YOUR-TOKEN
```

Example response:

```
HTTP/1.0 200 OK
Date: Fri, 25 Mar 2016 09:46:12 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, HEAD, OPTIONS

{
  "projects": "http://example.com/api/projects/",
  "components": "http://example.com/api/components/",
  "translations": "http://example.com/api/translations/",
  "languages": "http://example.com/api/languages/"
}
```

1.12.3 Kullanıcılar

4.0 sürümünde geldi.

GET /api/users/

Returns a list of users if you have permissions to see manage users. If not, then you get to see only your own details.

Ayrıca bakınız:

Users object attributes are documented at [GET /api/users/\(str:username\)/](#).

POST /api/users/

Creates a new user.

Parameters

- **username** (*string*) – Kullanıcı adı
- **full_name** (*string*) – User full name
- **email** (*string*) – User email
- **is_superuser** (*boolean*) – Is user superuser? (optional)
- **is_active** (*boolean*) – Is user active? (optional)
- **is_bot** (*boolean*) – Is user bot? (optional) (used for project scoped tokens)

GET /api/users/(str: username) /

Returns information about users.

Parameters

- **username** (*string*) – User's username

Response JSON Object

- **username** (*string*) – username of a user
- **full_name** (*string*) – full name of a user
- **email** (*string*) – email of a user
- **is_superuser** (*boolean*) – whether the user is a super user
- **is_active** (*boolean*) – whether the user is active
- **is_bot** (*boolean*) – whether the user is bot (used for project scoped tokens)
- **date_joined** (*string*) – date the user is created
- **groups** (*array*) – link to associated groups; see [GET /api/groups/\(int:id\)/](#)

Example JSON data:

```
{
  "email": "user@example.com",
  "full_name": "Example User",
  "username": "exampleusername",
  "groups": [
    "http://example.com/api/groups/2/",
    "http://example.com/api/groups/3/"
  ],
  "is_superuser": true,
  "is_active": true,
  "is_bot": false,
  "date_joined": "2020-03-29T18:42:42.617681Z",
  "url": "http://example.com/api/users/exampleusername/",
}
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

"statistics_url": "http://example.com/api/users/exampleusername/statistics/"
}

```

PUT `/api/users/(str: username) /`

Changes the user parameters.

Parameters

- **username** (*string*) – User's username

Response JSON Object

- **username** (*string*) – username of a user
- **full_name** (*string*) – full name of a user
- **email** (*string*) – email of a user
- **is_superuser** (*boolean*) – whether the user is a super user
- **is_active** (*boolean*) – whether the user is active
- **is_bot** (*boolean*) – whether the user is bot (used for project scoped tokens)
- **date_joined** (*string*) – date the user is created

PATCH `/api/users/(str: username) /`

Changes the user parameters.

Parameters

- **username** (*string*) – User's username

Response JSON Object

- **username** (*string*) – username of a user
- **full_name** (*string*) – full name of a user
- **email** (*string*) – email of a user
- **is_superuser** (*boolean*) – whether the user is a super user
- **is_active** (*boolean*) – whether the user is active
- **is_bot** (*boolean*) – whether the user is bot (used for project scoped tokens)
- **date_joined** (*string*) – date the user is created

DELETE `/api/users/(str: username) /`

Deletes all user information and marks the user inactive.

Parameters

- **username** (*string*) – User's username

POST `/api/users/(str: username)/groups/`

Associate groups with a user.

Parameters

- **username** (*string*) – User's username

Form Parameters

- **string group_id** – The unique group ID

GET /api/users/ (str: *username*) /statistics/

List statistics of a user.

Parameters

- **username** (*string*) – User's username

Response JSON Object

- **translated** (*int*) – Kullanıcı tarafından yapılan çeviri sayısı
- **suggested** (*int*) – Kullanıcı tarafından yapılan öneri sayısı
- **uploaded** (*int*) – Kullanıcı tarafından yapılan yükleme sayısı
- **commented** (*int*) – Kullanıcı tarafından yapılan yorum sayısı
- **languages** (*int*) – Kullanıcının çevirebileceği dil sayısı

GET /api/users/ (str: *username*) /notifications/

List subscriptions of a user.

Parameters

- **username** (*string*) – User's username

POST /api/users/ (str: *username*) /notifications/

Associate subscriptions with a user.

Parameters

- **username** (*string*) – User's username

Request JSON Object

- **notification** (*string*) – Name of notification registered
- **scope** (*int*) – Scope of notification from the available choices
- **frequency** (*int*) – Frequency choices for notifications

GET /api/users/ (str: *username*) /notifications/
int: *subscription_id*/

Get a subscription associated with a user.

Parameters

- **username** (*string*) – User's username
- **subscription_id** (*int*) – Kayıtlı bildirim kimliği

PUT /api/users/ (str: *username*) /notifications/
int: *subscription_id*/

Edit a subscription associated with a user.

Parameters

- **username** (*string*) – User's username
- **subscription_id** (*int*) – Kayıtlı bildirim kimliği

Request JSON Object

- **notification** (*string*) – Name of notification registered
- **scope** (*int*) – Scope of notification from the available choices
- **frequency** (*int*) – Frequency choices for notifications

PATCH /api/users/ (str: *username*) /notifications/
int: *subscription_id*/

Edit a subscription associated with a user.

Parameters

- **username** (*string*) – User’s username
- **subscription_id** (*int*) – Kayıtlı bildirim kimliği

Request JSON Object

- **notification** (*string*) – Name of notification registered
- **scope** (*int*) – Scope of notification from the available choices
- **frequency** (*int*) – Frequency choices for notifications

DELETE /api/users/ (**str**: *username*) /notifications/
int: *subscription_id* /

Delete a subscription associated with a user.

Parameters

- **username** (*string*) – User’s username
- **subscription_id** – Name of notification registered
- **subscription_id** – int

1.12.4 Gruplar

4.0 sürümünde geldi.

GET /api/groups/

Returns a list of groups if you have permissions to see manage groups. If not, then you get to see only the groups the user is a part of.

Ayrıca bakınız:

Group object attributes are documented at [GET /api/groups/\(int:id\)/](#).

POST /api/groups/

Creates a new group.

Parameters

- **name** (*string*) – Grup adı
- **project_selection** (*int*) – Group of project selection from given options
- **language_selection** (*int*) – Group of languages selected from given options
- **defining_project** (*str*) – link to the defining project, used for *Proje başına erişim denetimini yönetme*; see [GET /api/projects/\(string:project\)/](#)

GET /api/groups/ (**int**: *id*) /

Returns information about group.

Parameters

- **id** (*int*) – Group’s ID

Response JSON Object

- **name** (*string*) – name of a group
- **project_selection** (*int*) – integer corresponding to group of projects
- **language_selection** (*int*) – integer corresponding to group of languages
- **roles** (*array*) – link to associated roles; see [GET /api/roles/\(int:id\)/](#)
- **projects** (*array*) – link to associated projects; see [GET /api/projects/\(string:project\)/](#)

- **components** (*array*) – link to associated components; see `GET /api/components/(string:project)/(string:component)/`
- **componentlists** (*array*) – link to associated componentlist; see `GET /api/component-lists/(str:slug)/`
- **defining_project** (*str*) – link to the defining project, used for *Proje başına erişim denetimini yönetme*; see `GET /api/projects/(string:project)/`

Example JSON data:

```
{
  "name": "Guests",
  "defining_project": null,
  "project_selection": 3,
  "language_selection": 1,
  "url": "http://example.com/api/groups/1/",
  "roles": [
    "http://example.com/api/roles/1/",
    "http://example.com/api/roles/2/"
  ],
  "languages": [
    "http://example.com/api/languages/en/",
    "http://example.com/api/languages/cs/"
  ],
  "projects": [
    "http://example.com/api/projects/demo1/",
    "http://example.com/api/projects/demo/"
  ],
  "componentlist": "http://example.com/api/component-lists/new/",
  "components": [
    "http://example.com/api/components/demo/weblate/"
  ]
}
```

PUT `/api/groups/(int: id)/`

Changes the group parameters.

Parameters

- **id** (*int*) – Group's ID

Response JSON Object

- **name** (*string*) – name of a group
- **project_selection** (*int*) – integer corresponding to group of projects
- **language_selection** (*int*) – integer corresponding to group of Languages

PATCH `/api/groups/(int: id)/`

Changes the group parameters.

Parameters

- **id** (*int*) – Group's ID

Response JSON Object

- **name** (*string*) – name of a group
- **project_selection** (*int*) – integer corresponding to group of projects
- **language_selection** (*int*) – integer corresponding to group of languages

DELETE `/api/groups/(int: id)/`

Deletes the group.

Parameters

- `id(int)` – Group’s ID

POST `/api/groups/(int: id)/roles/`

Associate roles with a group.

Parameters

- `id(int)` – Group’s ID

Form Parameters

- `string role_id` – The unique role ID

POST `/api/groups/(int: id)/components/`

Associate components with a group.

Parameters

- `id(int)` – Group’s ID

Form Parameters

- `string component_id` – The unique component ID

DELETE `/api/groups/(int: id)/components/`

`int: component_id`

Delete component from a group.

Parameters

- `id(int)` – Group’s ID
- `component_id(int)` – The unique component ID

POST `/api/groups/(int: id)/projects/`

Associate projects with a group.

Parameters

- `id(int)` – Group’s ID

Form Parameters

- `string project_id` – The unique project ID

DELETE `/api/groups/(int: id)/projects/`

`int: project_id`

Delete project from a group.

Parameters

- `id(int)` – Group’s ID
- `project_id(int)` – The unique project ID

POST `/api/groups/(int: id)/languages/`

Associate languages with a group.

Parameters

- `id(int)` – Group’s ID

Form Parameters

- `string language_code` – The unique language code

DELETE `/api/groups/(int: id)/languages/`

`string: language_code`

Delete language from a group.

Parameters

- **id** (*int*) – Group's ID
- **language_code** (*string*) – The unique language code

POST /api/groups/(int: id)/componentlists/

Associate componentlists with a group.

Parameters

- **id** (*int*) – Group's ID

Form Parameters

- **string component_list_id** – The unique componentlist ID

DELETE /api/groups/(int: id)/componentlists/

int: *component_list_id*

Delete componentlist from a group.

Parameters

- **id** (*int*) – Group's ID
- **component_list_id** (*int*) – The unique componentlist ID

1.12.5 Roller

GET /api/roles/

Returns a list of all roles associated with user. If user is superuser, then list of all existing roles is returned.

Ayrıca bakınız:

Roles object attributes are documented at [GET /api/roles/\(int:id\)/](#).

POST /api/roles/

Creates a new role.

Parameters

- **name** (*string*) – Role name
- **permissions** (*array*) – List of codenames of permissions

GET /api/roles/(int: id) /

Returns information about a role.

Parameters

- **id** (*int*) – Role ID

Response JSON Object

- **name** (*string*) – Role name
- **permissions** (*array*) – list of codenames of permissions

Example JSON data:

```
{
  "name": "Access repository",
  "permissions": [
    "vcs.access",
    "vcs.view"
  ],
  "url": "http://example.com/api/roles/1/",
}
```

PUT `/api/roles/(int: id) /`

Changes the role parameters.

Parameters

- **id** (*int*) – Role’s ID

Response JSON Object

- **name** (*string*) – Role name
- **permissions** (*array*) – list of codenames of permissions

PATCH `/api/roles/(int: id) /`

Changes the role parameters.

Parameters

- **id** (*int*) – Role’s ID

Response JSON Object

- **name** (*string*) – Role name
- **permissions** (*array*) – list of codenames of permissions

DELETE `/api/roles/(int: id) /`

Deletes the role.

Parameters

- **id** (*int*) – Role’s ID

1.12.6 Diller

GET `/api/languages/`

Returns a list of all languages.

Ayrıca bakınız:

Language object attributes are documented at [GET /api/languages/\(string: language\) /](#).

POST `/api/languages/`

Creates a new language.

Parameters

- **code** (*string*) – Dil adı
- **name** (*string*) – Dil adı
- **direction** (*string*) – Metin yönü
- **population** (*int*) – Konuşmacı sayısı
- **plural** (*object*) – Language plural formula and number

GET `/api/languages/(string: language) /`

Returns information about a language.

Parameters

- **language** (*string*) – Dil kodu

Response JSON Object

- **code** (*string*) – Dil kodu
- **direction** (*string*) – Metin yönü
- **plural** (*object*) – Object of language plural information

- **aliases** (*array*) – Array of aliases for language

Request JSON Object

- **population** (*int*) – Konuşmacı sayısı

Example JSON data:

```
{
  "code": "en",
  "direction": "ltr",
  "name": "English",
  "population": 159034349015,
  "plural": {
    "id": 75,
    "source": 0,
    "number": 2,
    "formula": "n != 1",
    "type": 1
  },
  "aliases": [
    "english",
    "en_en",
    "base",
    "source",
    "eng"
  ],
  "url": "http://example.com/api/languages/en/",
  "web_url": "http://example.com/languages/en/",
  "statistics_url": "http://example.com/api/languages/en/statistics/"
}
```

PUT /api/languages/ (**string:** *language*) /

Changes the language parameters.

Parameters

- **language** (*string*) – Language's code

Request JSON Object

- **name** (*string*) – Dil adı
- **direction** (*string*) – Metin yönü
- **population** (*int*) – Konuşmacı sayısı
- **plural** (*object*) – Language plural details

PATCH /api/languages/ (**string:** *language*) /

Changes the language parameters.

Parameters

- **language** (*string*) – Language's code

Request JSON Object

- **name** (*string*) – Dil adı
- **direction** (*string*) – Metin yönü
- **population** (*int*) – Konuşmacı sayısı
- **plural** (*object*) – Language plural details

DELETE /api/languages/ (**string:** *language*) /

Dili siler.

Parameters

- **language** (*string*) – Language’s code

GET /api/languages/ (**string**: *language*) /statistics/

Returns statistics for a language.

Parameters

- **language** (*string*) – Dil kodu

Response JSON Object

- **total** (*int*) – total number of strings
- **total_words** (*int*) – total number of words
- **last_change** (*timestamp*) – last changes in the language
- **recent_changes** (*int*) – total number of changes
- **translated** (*int*) – number of translated strings
- **translated_percent** (*float*) – percentage of translated strings
- **translated_words** (*int*) – number of translated words
- **translated_words_percent** (*int*) – percentage of translated words
- **translated_chars** (*int*) – number of translated characters
- **translated_chars_percent** (*int*) – percentage of translated characters
- **total_chars** (*int*) – number of total characters
- **fuzzy** (*int*) – belirsiz (düzenleme için işaretlenen) dizgilerin sayısı
- **fuzzy_percent** (*int*) – percentage of fuzzy (marked for edit) strings
- **failing** (*int*) – number of failing strings
- **failing** – percentage of failing strings

1.12.7 Projeler

GET /api/projects/

Returns a list of all projects.

Ayrıca bakınız:

Project object attributes are documented at [GET /api/projects/ \(*string*: *project*\) /](#).

POST /api/projects/

3.9 sürümünde geldi.

Creates a new project.

Parameters

- **name** (*string*) – Proje adı
- **slug** (*string*) – Project slug
- **web** (*string*) – Proje web sitesi

GET /api/projects/ (**string**: *project*) /

Returns information about a project.

Parameters

- **project** (*string*) – Proje URL’si kısaltması

Response JSON Object

- **name** (*string*) – proje adı
- **slug** (*string*) – project slug
- **web** (*string*) – project website
- **components_list_url** (*string*) – URL to components list; see `GET /api/projects/(string:project)/components/`
- **repository_url** (*string*) – URL to repository status; see `GET /api/projects/(string:project)/repository/`
- **changes_list_url** (*string*) – URL to changes list; see `GET /api/projects/(string:project)/changes/`
- **translation_review** (*boolean*) – *Gözden geçirmeleri etkinleştir*
- **source_review** (*boolean*) – *Kaynak gözden geçirmelerini etkinleştir*
- **set_language_team** (*boolean*) – *“Language-Team” (Çeviri-Takımı) başlığını ayarla*
- **enable_hooks** (*boolean*) – *Kancaları etkinleştir*
- **instructions** (*string*) – *Çeviri talimatları*
- **language_aliases** (*string*) – *Dil kod adları*

Example JSON data:

```
{
  "name": "Hello",
  "slug": "hello",
  "url": "http://example.com/api/projects/hello/",
  "web": "https://weblate.org/",
  "web_url": "http://example.com/projects/hello/"
}
```

PATCH `/api/projects/(string: project) /`

4.3 sürümünde geldi.

Edit a project by a **PATCH** request.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması

PUT `/api/projects/(string: project) /`

4.3 sürümünde geldi.

Edit a project by a **PUT** request.

Parameters

- **project** (*string*) – Proje URL’si kısaltması

DELETE `/api/projects/(string: project) /`

3.9 sürümünde geldi.

Deletes a project.

Parameters

- **project** (*string*) – Proje URL’si kısaltması

GET `/api/projects/(string: project) /changes/`

Returns a list of project changes. This is essentially a project scoped `GET /api/changes/` accepting same params.

Parameters

- **project** (*string*) – Proje URL'si kısaltması

Response JSON Object

- **results** (*array*) – array of component objects; see `GET /api/changes/(int:id)/`

GET /api/projects/(string: project)/repository/

Returns information about VCS repository status. This endpoint contains only an overall summary for all repositories for the project. To get more detailed status use `GET /api/components/(string:project)/(string:component)/repository/`.

Parameters

- **project** (*string*) – Proje URL'si kısaltması

Response JSON Object

- **needs_commit** (*boolean*) – whether there are any pending changes to commit
- **needs_merge** (*boolean*) – whether there are any upstream changes to merge
- **needs_push** (*boolean*) – whether there are any local changes to push

Example JSON data:

```
{
  "needs_commit": true,
  "needs_merge": false,
  "needs_push": true
}
```

POST /api/projects/(string: project)/repository/

Performs given operation on the VCS repository.

Parameters

- **project** (*string*) – Proje URL'si kısaltması

Request JSON Object

- **operation** (*string*) – Operation to perform: one of push, pull, commit, re-set, cleanup, file-sync

Response JSON Object

- **result** (*boolean*) – result of the operation

CURL example:

```
curl \
  -d operation=pull \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/projects/hello/repository/
```

JSON request example:

```
POST /api/projects/hello/repository/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20

{"operation": "pull"}
```

JSON response example:

```
HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, POST, HEAD, OPTIONS

{"result":true}
```

GET `/api/projects/(string: project)/components/`

Returns a list of translation components in the given project.

Parameters

- **project** (*string*) – Proje URL’si kısaltması

Response JSON Object

- **results** (*array*) – array of component objects; see `GET /api/components/(string:project)/(string:component)/`

POST `/api/projects/(string: project)/components/`

3.9 sürümünde geldi.

4.3 sürümünde değişti: The `zipfile` and `docfile` parameters are now accepted for VCS-less components, see [Local files](#).

4.6 sürümünde değişti: The cloned repositories are now automatically shared within a project using [Weblate internal URLs](#). Use `disable_autoshare` to turn off this.

Creates translation components in the given project.

İpucu: Use [Weblate internal URLs](#) when creating multiple components from a single VCS repository.

Not: Most of the component creation happens in the background. Check the `task_url` attribute of created component and follow the progress there.

Parameters

- **project** (*string*) – Proje URL’si kısaltması

Form Parameters

- **file zipfile** – ZIP file to upload into Weblate for translations initialization
- **file docfile** – Çevrilecek belge
- **boolean disable_autoshare** – Disables automatic repository sharing via [Weblate internal URLs](#).

Request JSON Object

- **object** – Component parameters, see `GET /api/components/(string:project)/(string:component)/`

Response JSON Object

- **result** (*object*) – Created component object; see `GET /api/components/(string:project)/(string:component)/`

JSON can not be used when uploading the files using the `zipfile` and `docfile` parameters. The data has to be uploaded as *multipart/form-data*.

CURL form request example:

```
curl \
  --form docfile=@strings.html \
  --form name=Weblate \
  --form slug=weblate \
  --form file_format=html \
  --form new_lang=add \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/projects/hello/components/
```

CURL JSON request example:

```
curl \
  --data-binary '{
    "branch": "main",
    "file_format": "po",
    "filemask": "po/*.po",
    "name": "Weblate",
    "slug": "weblate",
    "repo": "https://github.com/WeblateOrg/hello.git",
    "template": "",
    "new_base": "po/hello.pot",
    "vcs": "git"
  }' \
  -H "Content-Type: application/json" \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/projects/hello/components/
```

JSON request to create a new component from Git:

```
POST /api/projects/hello/components/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20

{
  "branch": "main",
  "file_format": "po",
  "filemask": "po/*.po",
  "name": "Weblate",
  "slug": "weblate",
  "repo": "https://github.com/WeblateOrg/hello.git",
  "template": "",
  "new_base": "po/hello.pot",
  "vcs": "git"
}
```

JSON request to create a new component from another one:

```
POST /api/projects/hello/components/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20

{
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

"file_format": "po",
"filemask": "po/*.po",
"name": "Weblate",
"slug": "weblate",
"repo": "weblate://weblate/hello",
"template": "",
"new_base": "po/hello.pot",
"vcs": "git"
}

```

JSON response example:

```

HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, POST, HEAD, OPTIONS

{
  "branch": "main",
  "file_format": "po",
  "filemask": "po/*.po",
  "git_export": "",
  "license": "",
  "license_url": "",
  "name": "Weblate",
  "slug": "weblate",
  "project": {
    "name": "Hello",
    "slug": "hello",
    "source_language": {
      "code": "en",
      "direction": "ltr",
      "population": 159034349015,
      "name": "English",
      "url": "http://example.com/api/languages/en/",
      "web_url": "http://example.com/languages/en/"
    },
    "url": "http://example.com/api/projects/hello/",
    "web": "https://weblate.org/",
    "web_url": "http://example.com/projects/hello/"
  },
  "repo": "file:///home/nijel/work/weblate-hello",
  "template": "",
  "new_base": "",
  "url": "http://example.com/api/components/hello/weblate/",
  "vcs": "git",
  "web_url": "http://example.com/projects/hello/weblate/"
}

```

GET /api/projects/(string: *project*)/languages/

Returns paginated statistics for all languages within a project.

3.8 sürümünde geldi.

Parameters

- **project** (*string*) – Proje URL'si kısaltması

Response JSON Object

- **results** (*array*) – array of translation statistics objects
- **language** (*string*) – language name
- **code** (*string*) – language code
- **total** (*int*) – total number of strings
- **translated** (*int*) – number of translated strings
- **translated_percent** (*float*) – percentage of translated strings
- **total_words** (*int*) – total number of words
- **translated_words** (*int*) – number of translated words
- **words_percent** (*float*) – percentage of translated words

GET /api/projects/ (*string: project*) /statistics/

Returns statistics for a project.

3.8 sürümünde geldi.

Parameters

- **project** (*string*) – Proje URL’si kısaltması

Response JSON Object

- **total** (*int*) – total number of strings
- **translated** (*int*) – number of translated strings
- **translated_percent** (*float*) – percentage of translated strings
- **total_words** (*int*) – total number of words
- **translated_words** (*int*) – number of translated words
- **words_percent** (*float*) – percentage of translated words

1.12.8 Bileşenler

İpucu: Use *POST* /api/projects/ (*string:project*) /components/ to create new components.

GET /api/components/

Returns a list of translation components.

Ayrıca bakınız:

Component object attributes are documented at *GET* /api/components/ (*string:project*) / (*string:component*) /.

GET /api/components/ (*string: project*) /
string: *component* /

Returns information about translation component.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması

Response JSON Object

- **project** (*object*) – the translation project; see *GET* /api/projects/ (*string:project*) /
- **name** (*string*) – *Bileşen adı*

- **slug** (*string*) – *Component slug*
- **vcs** (*string*) – *Sürüm denetleme sistemi*
- **repo** (*string*) – *Kaynak kod deposu*
- **git_export** (*string*) – *Dışa aktarılmış depo URL'si*
- **branch** (*string*) – *Depo dalı*
- **push_branch** (*string*) – *Yollama dalı*
- **filemask** (*string*) – *Dosya maskesi*
- **template** (*string*) – *Tek dilli taban dil dosyası*
- **edit_template** (*string*) – *Taban dosyayı düzenle*
- **intermediate** (*string*) – *Ara dil dosyası*
- **new_base** (*string*) – *Yeni çeviriler için şablon*
- **file_format** (*string*) – *Dosya biçimi*
- **license** (*string*) – *Çeviri lisansı*
- **agreement** (*string*) – *Katılımcı sözleşmesi*
- **new_lang** (*string*) – *Yeni çeviri ekleniyor*
- **language_code_style** (*string*) – *Dil kodu stili*
- **source_language** (*object*) – source language object; see `GET /api/languages/(string:language)/`
- **push** (*string*) – *Depo yollama URL'si*
- **check_flags** (*string*) – *Çeviri işaretleri*
- **priority** (*string*) – *Öncelik*
- **enforced_checks** (*string*) – *Zorunlu denetimler*
- **restricted** (*string*) – *Restricted access*
- **repoweb** (*string*) – *Depo tarayıcısı*
- **report_source_bugs** (*string*) – *Kaynak dizgi hata bildirme adresi*
- **merge_style** (*string*) – *Birleştirme stili*
- **commit_message** (*string*) – *Commit, add, delete, merge, add-on, and merge request messages*
- **add_message** (*string*) – *Commit, add, delete, merge, add-on, and merge request messages*
- **delete_message** (*string*) – *Commit, add, delete, merge, add-on, and merge request messages*
- **merge_message** (*string*) – *Commit, add, delete, merge, add-on, and merge request messages*
- **addon_message** (*string*) – *Commit, add, delete, merge, add-on, and merge request messages*
- **pull_message** (*string*) – *Commit, add, delete, merge, add-on, and merge request messages*
- **allow_translation_propagation** (*string*) – *Çeviri yaymaya izin ver*
- **enable_suggestions** (*string*) – *Önerileri etkinleştir*
- **suggestion_voting** (*string*) – *Öneri oylaması*

- **suggestion_autoaccept** (*string*) – Önerileri kendiliğinden kabul et
- **push_on_commit** (*string*) – İşlemeye yolla
- **commit_pending_age** (*string*) – İşlenecek değişikliklerin yaşı
- **auto_lock_error** (*string*) – Hata durumunda kilitle
- **language_regex** (*string*) – Dil süzgeci
- **variant_regex** (*string*) – Çeşitlerin düzenli ifadesi
- **repository_url** (*string*) – URL to repository status; see `GET /api/components/(string:project)/(string:component)/repository/`
- **translations_url** (*string*) – URL to translations list; see `GET /api/components/(string:project)/(string:component)/translations/`
- **lock_url** (*string*) – URL to lock status; see `GET /api/components/(string:project)/(string:component)/lock/`
- **changes_list_url** (*string*) – URL to changes list; see `GET /api/components/(string:project)/(string:component)/changes/`
- **task_url** (*string*) – URL to a background task (if any); see `GET /api/tasks/(str:uuid)/`

Example JSON data:

```
{
  "branch": "main",
  "file_format": "po",
  "filemask": "po/*.po",
  "git_export": "",
  "license": "",
  "license_url": "",
  "name": "Weblate",
  "slug": "weblate",
  "project": {
    "name": "Hello",
    "slug": "hello",
    "source_language": {
      "code": "en",
      "direction": "ltr",
      "population": 159034349015,
      "name": "English",
      "url": "http://example.com/api/languages/en/",
      "web_url": "http://example.com/languages/en/"
    },
    "url": "http://example.com/api/projects/hello/",
    "web": "https://weblate.org/",
    "web_url": "http://example.com/projects/hello/"
  },
  "source_language": {
    "code": "en",
    "direction": "ltr",
    "population": 159034349015,
    "name": "English",
    "url": "http://example.com/api/languages/en/",
    "web_url": "http://example.com/languages/en/"
  },
  "repo": "file:///home/nijel/work/weblate-hello",
  "template": "",
  "new_base": "",
  "url": "http://example.com/api/components/hello/weblate/",

```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

"vcs": "git",
"web_url": "http://example.com/projects/hello/weblate/"
}

```

PATCH `/api/components/(string: project) /`
string: `component/`

Edit a component by a **PATCH** request.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması
- **source_language** (*string*) – Project source language code (optional)

Request JSON Object

- **name** (*string*) – name of component
- **slug** (*string*) – slug of component
- **repo** (*string*) – VCS repository URL

CURL example:

```

curl \
  --data-binary '{"name": "new name"}' \
  -H "Content-Type: application/json" \
  -H "Authorization: Token TOKEN" \
  PATCH http://example.com/api/projects/hello/components/

```

JSON request example:

```

PATCH /api/projects/hello/components/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20

{
  "name": "new name"
}

```

JSON response example:

```

HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, POST, HEAD, OPTIONS

{
  "branch": "main",
  "file_format": "po",
  "filemask": "po/*.po",
  "git_export": "",
  "license": "",
  "license_url": "",
  "name": "new name",

```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

"slug": "weblate",
"project": {
  "name": "Hello",
  "slug": "hello",
  "source_language": {
    "code": "en",
    "direction": "ltr",
    "population": 159034349015,
    "name": "English",
    "url": "http://example.com/api/languages/en/",
    "web_url": "http://example.com/languages/en/"
  },
  "url": "http://example.com/api/projects/hello/",
  "web": "https://weblate.org/",
  "web_url": "http://example.com/projects/hello/"
},
"repo": "file:///home/nijel/work/weblate-hello",
"template": "",
"new_base": "",
"url": "http://example.com/api/components/hello/weblate/",
"vcs": "git",
"web_url": "http://example.com/projects/hello/weblate/"
}

```

PUT /api/components/(string: project) /
string: component/

Edit a component by a **PUT** request.

Parameters

- **project** (string) – Proje URL’si kısaltması
- **component** (string) – Bileşen URL’si kısaltması

Request JSON Object

- **branch** (string) – VCS repository branch
- **file_format** (string) – file format of translations
- **filemask** (string) – mask of translation files in the repository
- **name** (string) – name of component
- **slug** (string) – slug of component
- **repo** (string) – VCS repository URL
- **template** (string) – base file for monolingual translations
- **new_base** (string) – base file for adding new translations
- **vcs** (string) – version control system

DELETE /api/components/(string: project) /
string: component/

3.9 sürümünde geldi.

Deletes a component.

Parameters

- **project** (string) – Proje URL’si kısaltması
- **component** (string) – Bileşen URL’si kısaltması

GET /api/components/(string: project) /
string: component/changes/

Returns a list of component changes. This is essentially a component scoped `GET /api/changes/` accepting same params.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması

Response JSON Object

- **results** (*array*) – array of component objects; see `GET /api/changes/(int:id)/`

GET /api/components/(string: project) /
string: component/file/

4.9 sürümünde geldi.

Downloads all available translations associated with the component as an archive file using the requested format.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması

Query Parameters

- **format** (*string*) – The archive format to use; If not specified, defaults to zip; Supported formats: zip
- **q** (*string*) – Filter downloaded strings, see search.

GET /api/components/(string: project) /
string: component/screenshots/

Returns a list of component screenshots.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması

Response JSON Object

- **results** (*array*) – array of component screenshots; see `GET /api/screenshots/(int:id)/`

GET /api/components/(string: project) /
string: component/lock/

Returns component lock status.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması

Response JSON Object

- **locked** (*boolean*) – whether component is locked for updates

Example JSON data:

```
{
  "locked": false
}
```

POST /api/components/(string: project) /
string: component/lock/

Sets component lock status.

Response is same as `GET /api/components/(string:project)/(string:component)/lock/`.

Parameters

- **project** (*string*) – Proje URL'si kısaltması
- **component** (*string*) – Bileşen URL'si kısaltması

Request JSON Object

- **lock** – Boolean whether to lock or not.

CURL example:

```
curl \
  -d lock=true \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/components/hello/weblate/repository/
```

JSON request example:

```
POST /api/components/hello/weblate/repository/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20

{"lock": true}
```

JSON response example:

```
HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, POST, HEAD, OPTIONS

{"locked": true}
```

GET `/api/components/(string:project)/string:component/repository/`

Returns information about VCS repository status.

The response is same as for `GET /api/projects/(string:project)/repository/`.

Parameters

- **project** (*string*) – Proje URL'si kısaltması
- **component** (*string*) – Bileşen URL'si kısaltması

Response JSON Object

- **needs_commit** (*boolean*) – whether there are any pending changes to commit
- **needs_merge** (*boolean*) – whether there are any upstream changes to merge
- **needs_push** (*boolean*) – whether there are any local changes to push
- **remote_commit** (*string*) – Remote commit information

- **status** (*string*) – VCS repository status as reported by VCS
- **merge_failure** – Text describing merge failure or null if there is none

POST /api/components/ (**string**: *project*) /
string: *component/repository/*

Performs the given operation on a VCS repository.

See *POST /api/projects/(string:project)/repository/* for documentation.

Parameters

- **project** (*string*) – Proje URL'si kısaltması
- **component** (*string*) – Bileşen URL'si kısaltması

Request JSON Object

- **operation** (*string*) – Operation to perform: one of push, pull, commit, re-set, cleanup

Response JSON Object

- **result** (*boolean*) – result of the operation

CURL example:

```
curl \
  -d operation=pull \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/components/hello/weblate/repository/
```

JSON request example:

```
POST /api/components/hello/weblate/repository/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20

{"operation":"pull"}
```

JSON response example:

```
HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, POST, HEAD, OPTIONS

{"result":true}
```

GET /api/components/ (**string**: *project*) /
string: *component/monolingual_base/*

Downloads base file for monolingual translations.

Parameters

- **project** (*string*) – Proje URL'si kısaltması
- **component** (*string*) – Bileşen URL'si kısaltması

GET /api/components/ (**string**: *project*) /
string: *component/new_template/*

Downloads template file for new translations.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması

GET `/api/components/(string: project) /string: component/translations/`

Returns a list of translation objects in the given component.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması

Response JSON Object

- **results** (*array*) – array of translation objects; see `GET /api/translations/(string:project)/(string:component)/(string:language)/`

POST `/api/components/(string: project) /string: component/translations/`

Creates new translation in the given component.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması

Request JSON Object

- **language_code** (*string*) – translation language code; see `GET /api/languages/(string:language)/`

Response JSON Object

- **result** (*object*) – new translation object created

CURL example:

```
curl \
  -d language_code=cs \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/projects/hello/components/
```

JSON request example:

```
POST /api/projects/hello/components/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20

{"language_code": "cs"}
```

JSON response example:

```
HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

Content-Type: application/json
Content-Language: en
Allow: GET, POST, HEAD, OPTIONS

{
  "failing_checks": 0,
  "failing_checks_percent": 0,
  "failing_checks_words": 0,
  "filename": "po/cs.po",
  "fuzzy": 0,
  "fuzzy_percent": 0.0,
  "fuzzy_words": 0,
  "have_comment": 0,
  "have_suggestion": 0,
  "is_template": false,
  "is_source": false,
  "language": {
    "code": "cs",
    "direction": "ltr",
    "population": 1303174280,
    "name": "Czech",
    "url": "http://example.com/api/languages/cs/",
    "web_url": "http://example.com/languages/cs/"
  },
  "language_code": "cs",
  "id": 125,
  "last_author": null,
  "last_change": null,
  "share_url": "http://example.com/engage/hello/cs/",
  "total": 4,
  "total_words": 15,
  "translate_url": "http://example.com/translate/hello/weblate/cs/",
  "translated": 0,
  "translated_percent": 0.0,
  "translated_words": 0,
  "url": "http://example.com/api/translations/hello/weblate/cs/",
  "web_url": "http://example.com/projects/hello/weblate/cs/"
}

```

GET `/api/components/(string: project) /`
string: `component/statistics/`

Returns paginated statistics for all translations within component.

2.7 sürümünde geldi.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması

Response JSON Object

- **results** (*array*) – array of translation statistics objects; see `GET /api/translations/(string:project)/(string:component)/(string:language)/statistics/`

GET `/api/components/(string: project) /`
string: `component/links/`

Bir bileşenle bağlantılı projeleri döndürür.

4.5 sürümünde geldi.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması

Response JSON Object

- **projects** (*array*) – ilişkili projeler; bkz `GET /api/projects/(string:project)/`

POST `/api/components/(string: project) /string: component/links/`

Projeyi bir bileşenle ilişkilendir.

4.5 sürümünde geldi.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması

Form Parameters

- **string project_slug** – Project slug

DELETE `/api/components/(string: project) /string: component/links/string: project_slug/`

Bir projenin bir bileşenle ilişkisini kaldır.

4.5 sürümünde geldi.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması
- **project_slug** (*string*) – Kaldırılacak projenin kısaltması

1.12.9 Çeviriler

GET `/api/translations/`

Returns a list of translations.

Ayrıca bakınız:

Translation object attributes are documented at `GET /api/translations/(string:project)/(string:component)/(string:language)/`.

GET `/api/translations/(string: project) /string: component/string: language/`

Returns information about a translation.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması
- **language** (*string*) – Translation language code

Response JSON Object

- **component** (*object*) – component object; see `GET /api/components/(string:project)/(string:component)/`
- **failing_checks** (*int*) – kusur denetimleri olan dizgi sayısı
- **failing_checks_percent** (*float*) – kusur denetimleri olan dizgi yüzdesi

- **failing_checks_words** (*int*) – kusur denetimleri olan sözcük sayısı
- **filename** (*string*) – translation filename
- **fuzzy** (*int*) – belirsiz (düzenleme için işaretlenen) dizgilerin sayısı
- **fuzzy_percent** (*float*) – percentage of fuzzy (marked for edit) strings
- **fuzzy_words** (*int*) – number of words in fuzzy (marked for edit) strings
- **have_comment** (*int*) – number of strings with comment
- **have_suggestion** (*int*) – number of strings with suggestion
- **is_template** (*boolean*) – çevirinin tek dilli bir tabanı olup olmadığı
- **language** (*object*) – source language object; see [GET /api/languages/\(string:language\)/](#)
- **language_code** (*string*) – language code used in the repository; this can be different from language code in the language object
- **last_author** (*string*) – name of last author
- **last_change** (*timestamp*) – last change timestamp
- **revision** (*string*) – revision hash for the file
- **share_url** (*string*) – URL for sharing leading to engagement page
- **total** (*int*) – total number of strings
- **total_words** (*int*) – total number of words
- **translate_url** (*string*) – URL for translating
- **translated** (*int*) – number of translated strings
- **translated_percent** (*float*) – percentage of translated strings
- **translated_words** (*int*) – number of translated words
- **repository_url** (*string*) – URL to repository status; see [GET /api/translations/\(string:project\)/\(string:component\)/\(string:language\)/repository/](#)
- **file_url** (*string*) – URL to file object; see [GET /api/translations/\(string:project\)/\(string:component\)/\(string:language\)/file/](#)
- **changes_list_url** (*string*) – URL to changes list; see [GET /api/translations/\(string:project\)/\(string:component\)/\(string:language\)/changes/](#)
- **units_list_url** (*string*) – URL to strings list; see [GET /api/translations/\(string:project\)/\(string:component\)/\(string:language\)/units/](#)

Example JSON data:

```
{
  "component": {
    "branch": "main",
    "file_format": "po",
    "filemask": "po/*.po",
    "git_export": "",
    "license": "",
    "license_url": "",
    "name": "Weblate",
    "new_base": "",
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

    "project": {
      "name": "Hello",
      "slug": "hello",
      "source_language": {
        "code": "en",
        "direction": "ltr",
        "population": 159034349015,
        "name": "English",
        "url": "http://example.com/api/languages/en/",
        "web_url": "http://example.com/languages/en/"
      },
      "url": "http://example.com/api/projects/hello/",
      "web": "https://weblate.org/",
      "web_url": "http://example.com/projects/hello/"
    },
    "repo": "file:///home/nijel/work/weblate-hello",
    "slug": "weblate",
    "template": "",
    "url": "http://example.com/api/components/hello/weblate/",
    "vcs": "git",
    "web_url": "http://example.com/projects/hello/weblate/"
  },
  "failing_checks": 3,
  "failing_checks_percent": 75.0,
  "failing_checks_words": 11,
  "filename": "po/cs.po",
  "fuzzy": 0,
  "fuzzy_percent": 0.0,
  "fuzzy_words": 0,
  "have_comment": 0,
  "have_suggestion": 0,
  "is_template": false,
  "language": {
    "code": "cs",
    "direction": "ltr",
    "population": 1303174280,
    "name": "Czech",
    "url": "http://example.com/api/languages/cs/",
    "web_url": "http://example.com/languages/cs/"
  },
  "language_code": "cs",
  "last_author": "Weblate Admin",
  "last_change": "2016-03-07T10:20:05.499",
  "revision": "7ddfafe6daaf57fc8654cc852ea6be212b015792",
  "share_url": "http://example.com/engage/hello/cs/",
  "total": 4,
  "total_words": 15,
  "translate_url": "http://example.com/translate/hello/weblate/cs/",
  "translated": 4,
  "translated_percent": 100.0,
  "translated_words": 15,
  "url": "http://example.com/api/translations/hello/weblate/cs/",
  "web_url": "http://example.com/projects/hello/weblate/cs/"
}

```

DELETE /api/translations/(string: *project*) /
 string: *component*/string: *language*/

3.9 sürümünde geldi.

Deletes a translation.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması
- **language** (*string*) – Translation language code

GET /api/translations/ (**string:** *project*) /
string: *component* / **string:** *language/changes/*

Returns a list of translation changes. This is essentially a translations-scoped [GET /api/changes/](#) accepting the same parameters.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması
- **language** (*string*) – Translation language code

Response JSON Object

- **results** (*array*) – array of component objects; see [GET /api/changes/ \(int:id\)](#)

GET /api/translations/ (**string:** *project*) /
string: *component* / **string:** *language/units/*

Returns a list of translation units.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması
- **language** (*string*) – Translation language code
- **q** (*string*) – Search query string [Searching](#) (optional)

Response JSON Object

- **results** (*array*) – array of component objects; see [GET /api/units/ \(int:id\)](#)

POST /api/translations/ (**string:** *project*) /
string: *component* / **string:** *language/units/*

Add new unit.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması
- **language** (*string*) – Translation language code

Request JSON Object

- **key** (*string*) – Name of translation unit (used as key or context)
- **value** (*array*) – Source strings (use single string if not creating plural)

Response JSON Object

- **unit** (*object*) – newly created unit; see [GET /api/units/ \(int:id\)](#)

Ayrıca bakınız:

[Dizgileri yönet](#), adding-new-strings

POST /api/translations/ (**string:** *project*) /
string: *component* / **string:** *language/autotranslate/*

Trigger automatic translation.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması
- **language** (*string*) – Translation language code

Request JSON Object

- **mode** (*string*) – Otomatik çeviri kipi
- **filter_type** (*string*) – Automatic translation filter type
- **auto_source** (*string*) – Otomatik çeviri kaynağı - mt veya others
- **component** (*string*) – Projenin ek bileşenlere erişebilmesi için paylaşılan çeviri bel-
leğine katkıyı açın.
- **engines** (*array*) – Makine çevirisi motorları
- **threshold** (*string*) – Skor eşiği

GET /api/translations/ (**string:** *project*) /
string: *component* / **string:** *language* / **file** /

Download current translation file as it is stored in the VCS (without the `format` parameter) or converted to another format (see [Downloading translations](#)).

Not: This API endpoint uses different logic for output than rest of API as it operates on whole file rather than on data. Set of accepted `format` parameter differs and without such parameter you get translation file as stored in VCS.

Query Parameters

- **format** – File format to use; if not specified no format conversion happens; supported file formats: po, mo, xliff, xliff11, tbx, csv, xlsx, json, aresource, strings

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması
- **language** (*string*) – Translation language code

POST /api/translations/ (**string:** *project*) /
string: *component* / **string:** *language* / **file** /

Upload new file with translations.

Parameters

- **project** (*string*) – Proje URL’si kısaltması
- **component** (*string*) – Bileşen URL’si kısaltması
- **language** (*string*) – Translation language code

Form Parameters

- **string conflict** – How to deal with conflicts (ignore, replace-translated or replace-approved)
- **file file** – Uploaded file
- **string email** – Hazırlayanın e-postası
- **string author** – Hazırlayanın adı

- **string method** – Upload method (translate, approve, suggest, fuzzy, replace, source, add), see [Import methods](#)
- **string fuzzy** – Fuzzy (marked for edit) strings processing (*empty*, process, approve)

CURL example:

```
curl -X POST \
  -F file=@strings.xml \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/translations/hello/android/cs/file/
```

GET /api/translations/ (string: project) /
string: component/string: language/repository/

Returns information about VCS repository status.

The response is same as for `GET /api/components/(string:project)/(string:component)/repository/`.

Parameters

- **project** (*string*) – Proje URL'si kısaltması
- **component** (*string*) – Bileşen URL'si kısaltması
- **language** (*string*) – Translation language code

POST /api/translations/ (string: project) /
string: component/string: language/repository/

Performs given operation on the VCS repository.

See `POST /api/projects/(string:project)/repository/` for documentation.

Parameters

- **project** (*string*) – Proje URL'si kısaltması
- **component** (*string*) – Bileşen URL'si kısaltması
- **language** (*string*) – Translation language code

Request JSON Object

- **operation** (*string*) – Operation to perform: one of push, pull, commit, reset, cleanup

Response JSON Object

- **result** (*boolean*) – result of the operation

GET /api/translations/ (string: project) /
string: component/string: language/statistics/

Returns detailed translation statistics.

2.7 sürümünde geldi.

Parameters

- **project** (*string*) – Proje URL'si kısaltması
- **component** (*string*) – Bileşen URL'si kısaltması
- **language** (*string*) – Translation language code

Response JSON Object

- **code** (*string*) – language code
- **failing** (*int*) – number of failing checks

- **failing_percent** (*float*) – percentage of failing checks
- **fuzzy** (*int*) – belirsiz (düzenleme için işaretlenen) dizgilerin sayısı
- **fuzzy_percent** (*float*) – percentage of fuzzy (marked for edit) strings
- **total_words** (*int*) – total number of words
- **translated_words** (*int*) – number of translated words
- **last_author** (*string*) – name of last author
- **last_change** (*timestamp*) – date of last change
- **name** (*string*) – language name
- **total** (*int*) – total number of strings
- **translated** (*int*) – number of translated strings
- **translated_percent** (*float*) – percentage of translated strings
- **url** (*string*) – URL to access the translation (engagement URL)
- **url_translate** (*string*) – URL to access the translation (real translation URL)

1.12.10 Units

A *unit* is a single piece of a translation which pairs a source string with a corresponding translated string and also contains some related metadata. The term is derived from the [Translate Toolkit](#) and XLIFF.

2.10 sürümünde geldi.

GET /api/units/

Returns list of translation units.

Ayrıca bakınız:

Unit object attributes are documented at [GET /api/units/\(int:id\)/](#).

GET /api/units/(int: id) /

4.3 sürümünde değişti: The `target` and `source` are now arrays to properly handle plural strings.

Returns information about translation unit.

Parameters

- **id** (*int*) – Unit ID

Response JSON Object

- **translation** (*string*) – URL of a related translation object
- **source** (*array*) – source string
- **previous_source** (*string*) – previous source string used for fuzzy matching
- **target** (*array*) – target string
- **id_hash** (*string*) – unique identifier of the unit
- **content_hash** (*string*) – unique identifier of the source string
- **location** (*string*) – location of the unit in source code
- **context** (*string*) – translation unit context
- **note** (*string*) – translation unit note
- **flags** (*string*) – translation unit flags
- **state** (*int*) – unit state, 0 - untranslated, 10 - needs editing, 20 - translated, 30 - approved, 100 - read only

- **fuzzy** (*boolean*) – birimin belirsiz veya gözden geçirilmek üzere işaretlenmiş olup olmadığı
- **translated** (*boolean*) – birimin çevrilmiş olup olmadığı
- **approved** (*boolean*) – çevirinin onaylanmış olup olmadığı
- **position** (*int*) – unit position in translation file
- **has_suggestion** (*boolean*) – birimin önerileri olup olmadığı
- **has_comment** (*boolean*) – birimin yorumları olup olmadığı
- **has_failing_check** (*boolean*) – birimin kusur denetimleri olup olmadığı
- **num_words** (*int*) – number of source words
- **priority** (*int*) – translation priority; 100 is default
- **id** (*int*) – unit identifier
- **explanation** (*string*) – String explanation, available on source units, see [Additional info on source strings](#)
- **extra_flags** (*string*) – Additional string flags, available on source units, see [Customizing behavior using flags](#)
- **web_url** (*string*) – URL where the unit can be edited
- **source_unit** (*string*) – Source unit link; see `GET /api/units/(int:id)/`
- **pending** (*boolean*) – whether the unit is pending for write
- **timestamp** (*timestamp*) – string age

PATCH `/api/units/(int: id) /`

4.3 sürümünde geldi.

Çeviri birimi üzerinde kısmi güncelleme gerçekleştirir.

Parameters

- **id** (*int*) – Unit ID

Request JSON Object

- **state** (*int*) – unit state, 0 - untranslated, 10 - needs editing, 20 - translated, 30 - approved (need review workflow enabled, see [Dedicated reviewers](#))
- **target** (*array*) – target string
- **explanation** (*string*) – String explanation, available on source units, see [Additional info on source strings](#)
- **extra_flags** (*string*) – Additional string flags, available on source units, see [Customizing behavior using flags](#)

PUT `/api/units/(int: id) /`

4.3 sürümünde geldi.

Çeviri birimi üzerinde tam güncelleme gerçekleştirir.

Parameters

- **id** (*int*) – Unit ID

Request JSON Object

- **state** (*int*) – unit state, 0 - untranslated, 10 - needs editing, 20 - translated, 30 - approved (need review workflow enabled, see [Dedicated reviewers](#))
- **target** (*array*) – target string

- **explanation** (*string*) – String explanation, available on source units, see [Additional info on source strings](#)
- **extra_flags** (*string*) – Additional string flags, available on source units, see [Customizing behavior using flags](#)

DELETE /api/units/ (int: id) /

4.3 sürümünde geldi.

Bir çeviri birimini siler.

Parameters

- **id** (*int*) – Unit ID

1.12.11 Değişiklikler

2.10 sürümünde geldi.

GET /api/changes/

4.1 sürümünde değişti: Filtering of changes was introduced in the 4.1 release.

Returns a list of translation changes.

Ayrıca bakınız:

Change object attributes are documented at [GET /api/changes/\(int:id\)/](#).

Query Parameters

- **user** (*string*) – Username of user to filters
- **action** (*int*) – Action to filter, can be used several times
- **timestamp_after** (*timestamp*) – ISO 8601 formatted timestamp to list changes after
- **timestamp_before** (*timestamp*) – ISO 8601 formatted timestamp to list changes before

GET /api/changes/ (int: id) /

Returns information about translation change.

Parameters

- **id** (*int*) – Change ID

Response JSON Object

- **unit** (*string*) – URL of a related unit object
- **translation** (*string*) – URL of a related translation object
- **component** (*string*) – URL of a related component object
- **user** (*string*) – URL of a related user object
- **author** (*string*) – URL of a related author object
- **timestamp** (*timestamp*) – event timestamp
- **action** (*int*) – numeric identification of action
- **action_name** (*string*) – text description of action
- **target** (*string*) – event changed text or detail
- **id** (*int*) – change identifier

1.12.12 Ekran görüntüleri

2.14 sürümünde geldi.

GET /api/screenshots/

Returns a list of screenshot string information.

Ayrıca bakınız:

Screenshot object attributes are documented at `GET /api/screenshots/(int:id)/`.

GET /api/screenshots/(int: id) /

Returns information about screenshot information.

Parameters

- **id** (*int*) – Screenshot ID

Response JSON Object

- **name** (*string*) – name of a screenshot
- **component** (*string*) – URL of a related component object
- **file_url** (*string*) – URL to download a file; see `GET /api/screenshots/(int:id)/file/`
- **units** (*array*) – link to associated source string information; see `GET /api/units/(int:id)/`

GET /api/screenshots/(int: id) /file/

Download the screenshot image.

Parameters

- **id** (*int*) – Screenshot ID

POST /api/screenshots/(int: id) /file/

Replace screenshot image.

Parameters

- **id** (*int*) – Screenshot ID

Form Parameters

- **file image** – Uploaded file

CURL example:

```
curl -X POST \
  -F image=@image.png \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/screenshots/1/file/
```

POST /api/screenshots/(int: id) /units/

Associate source string with screenshot.

Parameters

- **id** (*int*) – Screenshot ID

Form Parameters

- **string unit_id** – Unit ID

Response JSON Object

- **name** (*string*) – name of a screenshot
- **translation** (*string*) – URL of a related translation object

- **file_url** (*string*) – URL to download a file; see `GET /api/screenshots/(int:id)/file/`
- **units** (*array*) – link to associated source string information; see `GET /api/units/(int:id)/`

**DELETE /api/screenshots/(int: id)/units/
int: unit_id**

Kaynak dizgi ile ekran görüntüsü ilişkilendirmesini kaldır.

Parameters

- **id** (*int*) – Screenshot ID
- **unit_id** – Kaynak dizgi birimi kimliği

POST /api/screenshots/

Yeni bir ekran görüntüsü oluşturur.

Form Parameters

- **file image** – Uploaded file
- **string name** – Ekran görüntüsü adı
- **string project_slug** – Project slug
- **string component_slug** – Component slug
- **string language_code** – Dil kodu

Response JSON Object

- **name** (*string*) – name of a screenshot
- **component** (*string*) – URL of a related component object
- **file_url** (*string*) – URL to download a file; see `GET /api/screenshots/(int:id)/file/`
- **units** (*array*) – link to associated source string information; see `GET /api/units/(int:id)/`

PATCH /api/screenshots/(int: id) /

Ekran görüntüsü ile ilgili kısmi bilgileri düzenle.

Parameters

- **id** (*int*) – Screenshot ID

Response JSON Object

- **name** (*string*) – name of a screenshot
- **component** (*string*) – URL of a related component object
- **file_url** (*string*) – URL to download a file; see `GET /api/screenshots/(int:id)/file/`
- **units** (*array*) – link to associated source string information; see `GET /api/units/(int:id)/`

PUT /api/screenshots/(int: id) /

Ekran görüntüsü ile ilgili tüm bilgileri düzenle.

Parameters

- **id** (*int*) – Screenshot ID

Response JSON Object

- **name** (*string*) – name of a screenshot

- **component** (*string*) – URL of a related component object
- **file_url** (*string*) – URL to download a file; see `GET /api/screenshots/(int:id)/file/`
- **units** (*array*) – link to associated source string information; see `GET /api/units/(int:id)/`

DELETE `/api/screenshots/(int: id) /`

Ekran görüntüsünü sil.

Parameters

- **id** (*int*) – Screenshot ID

1.12.13 Eklentiler

4.4.1 sürümünde geldi.

GET `/api/addons/`

Returns a list of add-ons.

Ayrıca bakınız:

Eklenti nesne nitelikleri `GET /api/addons/(int:id)/` adresinde belgelendirilmiştir.

GET `/api/addons/(int: id) /`

Returns information about add-on information.

Parameters

- **id** (*int*) – Eklenti kimliği

Response JSON Object

- **name** (*string*) – name of an add-on
- **component** (*string*) – URL of a related component object
- **configuration** (*object*) – Optional add-on configuration

Ayrıca bakınız:

[Eklentiler](#)

POST `/api/components/(string: project) /`
string: `component/addons/`

Creates a new add-on.

Parameters

- **project_slug** (*string*) – Project slug
- **component_slug** (*string*) – Component slug

Request JSON Object

- **name** (*string*) – name of an add-on
- **configuration** (*object*) – Optional add-on configuration

PATCH `/api/addons/(int: id) /`

Edit partial information about add-on.

Parameters

- **id** (*int*) – Eklenti kimliği

Response JSON Object

- **configuration** (*object*) – Optional add-on configuration

PUT `/api/addons/(int: id)/`

Edit full information about add-on.

Parameters

- **id** (*int*) – Eklenti kimliği

Response JSON Object

- **configuration** (*object*) – Optional add-on configuration

DELETE `/api/addons/(int: id)/`

Delete add-on.

Parameters

- **id** (*int*) – Eklenti kimliği

1.12.14 Bileşen listeleri

4.0 sürümünde geldi.

GET `/api/component-lists/`

Bileşen listelerinin bir listesini döndürür.

Ayrıca bakınız:

Bileşen listesi nesne nitelikleri `GET /api/component-lists/(str:slug)/` adresinde belgelendirilmiştir.

GET `/api/component-lists/(str: slug)/`

Bileşen listesi hakkında bilgi döndürür.

Parameters

- **slug** (*string*) – Bileşen listesi kısaltması

Response JSON Object

- **name** (*string*) – bir bileşen listesinin adı
- **slug** (*string*) – bir bileşen listesinin kısaltması
- **show_dashboard** (*boolean*) – bir panelde gösterilip gösterilmeyeceği
- **components** (*array*) – link to associated components; see `GET /api/components/(string:project)/(string:component)/`
- **auto_assign** (*array*) – otomatik atama kuralları

PUT `/api/component-lists/(str: slug)/`

Bileşen listesi parametrelerini değiştirir.

Parameters

- **slug** (*string*) – Bileşen listesi kısaltması

Request JSON Object

- **name** (*string*) – bir bileşen listesinin adı
- **slug** (*string*) – bir bileşen listesinin kısaltması
- **show_dashboard** (*boolean*) – bir panelde gösterilip gösterilmeyeceği

PATCH `/api/component-lists/ (str: slug) /`

Bileşen listesi parametrelerini değiştirir.

Parameters

- **slug** (*string*) – Bileşen listesi kısaltması

Request JSON Object

- **name** (*string*) – bir bileşen listesinin adı
- **slug** (*string*) – bir bileşen listesinin kısaltması
- **show_dashboard** (*boolean*) – bir panelde gösterilip gösterilmeyeceği

DELETE `/api/component-lists/ (str: slug) /`

Bileşen listesini siler.

Parameters

- **slug** (*string*) – Bileşen listesi kısaltması

POST `/api/component-lists/ (str: slug) /components/`

Bileşeni bir bileşen listesiyle ilişkilendirir.

Parameters

- **slug** (*string*) – Bileşen listesi kısaltması

Form Parameters

- **string component_id** – Bileşen kimliği

DELETE `/api/component-lists/ (str: slug) /components/
str: component_slug`

Bileşen listesinden bir bileşenin ilişkisini kaldırır.

Parameters

- **slug** (*string*) – Bileşen listesi kısaltması
- **component_slug** (*string*) – Component slug

1.12.15 Sözlük

4.5 sürümünde değişti: Glossaries are now stored as regular components, translations and strings, please use respective API instead.

1.12.16 Görevler

4.4 sürümünde geldi.

GET `/api/tasks/`

Görevlerin listesi şu anda kullanılamıyor.

GET `/api/tasks/ (str: uuid) /`

Bir görevle ilgili bilgileri döndürür

Parameters

- **uuid** (*string*) – Görev UUID'si

Response JSON Object

- **completed** (*boolean*) – Görevin tamamlanıp tamamlanmadığı
- **progress** (*int*) – Yüzde olarak görev ilerlemesi

- **result** (*object*) – Görev sonucu veya ilerleme ayrıntıları
- **log** (*string*) – Görev günlüğü

1.12.17 Metrics

GET /api/metrics/

Returns server metrics.

Response JSON Object

- **units** (*int*) – Birim sayısı
- **units_translated** (*int*) – Çevrilen birim sayısı
- **users** (*int*) – Kullanıcı sayısı
- **changes** (*int*) – Değişiklik sayısı
- **projects** (*int*) – Proje sayısı
- **components** (*int*) – Bileşen sayısı
- **translations** (*int*) – Çeviri sayısı
- **languages** (*int*) – Kullanılan dil sayısı
- **checks** (*int*) – Tetiklenen nitelik denetimleri sayısı
- **configuration_errors** (*int*) – Yapılandırma hataları sayısı
- **suggestions** (*int*) – Bekleyen öneri sayısı
- **celery_queues** (*object*) – Lengths of Celery queues, see [Background tasks using Celery](#)
- **name** (*string*) – Yapılandırılan sunucu adı

1.12.18 Bildirim kancaları

Notification hooks allow external applications to notify Weblate that the VCS repository has been updated.

You can use repository endpoints for projects, components and translations to update individual repositories; see [POST /api/projects/\(string:project\)/repository/](#) for documentation.

GET /hooks/update/(string: project) /
string: component/

2.6 sürümünden beri kullanım dışı: Please use [POST /api/components/\(string:project\)/\(string:component\)/repository/](#) instead which works properly with authentication for ACL limited projects.

Triggers update of a component (pulling from VCS and scanning for translation changes).

GET /hooks/update/(string: project) /

2.6 sürümünden beri kullanım dışı: Please use [POST /api/projects/\(string:project\)/repository/](#) instead which works properly with authentication for ACL limited projects.

Triggers update of all components in a project (pulling from VCS and scanning for translation changes).

POST /hooks/github/

Special hook for handling GitHub notifications and automatically updating matching components.

Not: GitHub includes direct support for notifying Weblate: enable Weblate service hook in repository settings and set the URL to the URL of your Weblate installation.

Ayrıca bakınız:

Automatically receiving changes from GitHub

For instruction on setting up GitHub integration

<https://docs.github.com/en/get-started/customizing-your-github-workflow/exploring-integrations/about-webhooks>

Generic information about GitHub Webhooks

ENABLE_HOOKS

For enabling hooks for whole Weblate

POST /hooks/gitlab/

Special hook for handling GitLab notifications and automatically updating matching components.

Ayrıca bakınız:

Automatically receiving changes from GitLab

For instruction on setting up GitLab integration

<https://docs.gitlab.com/ee/user/project/integrations/webhooks.html>

Generic information about GitLab Webhooks

ENABLE_HOOKS

For enabling hooks for whole Weblate

POST /hooks/bitbucket/

Special hook for handling Bitbucket notifications and automatically updating matching components.

Ayrıca bakınız:

Automatically receiving changes from Bitbucket

For instruction on setting up Bitbucket integration

<https://support.atlassian.com/bitbucket-cloud/docs/manage-webhooks/>

Generic information about Bitbucket Webhooks

ENABLE_HOOKS

For enabling hooks for whole Weblate

POST /hooks/pagure/

3.3 sürümünde geldi.

Special hook for handling Pagure notifications and automatically updating matching components.

Ayrıca bakınız:

Automatically receiving changes from Pagure

For instruction on setting up Pagure integration

https://docs.pagure.org/pagure/usage/using_webhooks.html

Generic information about Pagure Webhooks

ENABLE_HOOKS

For enabling hooks for whole Weblate

POST /hooks/azure/

3.8 sürümünde geldi.

Special hook for handling Azure DevOps notifications and automatically updating matching components.

Not: Please make sure that *Resource details to send* is set to *All*, otherwise Weblate will not be able to match your Azure repository.

Ayrıca bakınız:

Automatically receiving changes from Azure Repos

For instruction on setting up Azure integration

[https:](https://docs.microsoft.com/en-us/azure/devops/service-hooks/services/webhooks?view=azure-devops)

[//docs.microsoft.com/en-us/azure/devops/service-hooks/services/webhooks?view=azure-devops](https://docs.microsoft.com/en-us/azure/devops/service-hooks/services/webhooks?view=azure-devops)

Generic information about Azure DevOps Web Hooks

ENABLE_HOOKS

For enabling hooks for whole Weblate

POST /hooks/gitea/

3.9 sürümünde geldi.

Special hook for handling Gitea Webhook notifications and automatically updating matching components.

Ayrıca bakınız:

Automatically receiving changes from Gitea Repos

For instruction on setting up Gitea integration

<https://docs.gitea.io/en-us/webhooks/>

Generic information about Gitea Webhooks

ENABLE_HOOKS

For enabling hooks for whole Weblate

POST /hooks/gitee/

3.9 sürümünde geldi.

Special hook for handling Gitee Webhook notifications and automatically updating matching components.

Ayrıca bakınız:

Automatically receiving changes from Gitee Repos

For instruction on setting up Gitee integration

<https://gitee.com/help/categories/40>

Generic information about Gitee Webhooks

ENABLE_HOOKS

For enabling hooks for whole Weblate

1.12.19 Exports

Weblate provides various exports to allow you to further process the data.

GET /exports/stats/ (string: project) /
string: component/

Query Parameters

- **format** (*string*) – Output format: either json or csv

2.6 sürümünden beri kullanım dışı: Please use `GET /api/components/(string:project)/(string:component)/statistics/` and `GET /api/translations/(string:project)/(string:component)/(string:language)/statistics/` instead; it allows access to ACL controlled projects as well.

Retrieves statistics for given component in given format.

Example request:

```
GET /exports/stats/weblate/main/ HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
```

Example response:

```
HTTP/1.1 200 OK
Vary: Accept
Content-Type: application/json

[
  {
    "code": "cs",
    "failing": 0,
    "failing_percent": 0.0,
    "fuzzy": 0,
    "fuzzy_percent": 0.0,
    "last_author": "Michal Čihař",
    "last_change": "2012-03-28T15:07:38+00:00",
    "name": "Czech",
    "total": 436,
    "total_words": 15271,
    "translated": 436,
    "translated_percent": 100.0,
    "translated_words": 3201,
    "url": "http://hosted.weblate.org/engage/weblate/cs/",
    "url_translate": "http://hosted.weblate.org/projects/weblate/main/cs/"
  },
  {
    "code": "nl",
    "failing": 21,
    "failing_percent": 4.8,
    "fuzzy": 11,
    "fuzzy_percent": 2.5,
    "last_author": null,
    "last_change": null,
    "name": "Dutch",
    "total": 436,
    "total_words": 15271,
    "translated": 319,
    "translated_percent": 73.2,
    "translated_words": 3201,
    "url": "http://hosted.weblate.org/engage/weblate/nl/",
    "url_translate": "http://hosted.weblate.org/projects/weblate/main/nl/"
  },
  {
    "code": "el",
    "failing": 11,
    "failing_percent": 2.5,
    "fuzzy": 21,
    "fuzzy_percent": 4.8,
    "last_author": null,
    "last_change": null,
    "name": "Greek",
    "total": 436,
    "total_words": 15271,
    "translated": 312,
    "translated_percent": 71.6,
    "translated_words": 3201,
    "url": "http://hosted.weblate.org/engage/weblate/el/",
    "url_translate": "http://hosted.weblate.org/projects/weblate/main/el/"
  }
]
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

]

1.12.20 RSS bildirimleri

Çevirilerdeki değişiklikler RSS bildirimlerinde dışa aktarılır.

GET `/exports/rss/ (string: project) /`
string: `component/string: language/`

Bir çeviri için son değişiklikleri içeren RSS bildirimini alır.

GET `/exports/rss/ (string: project) /`
string: `component/`

Bir bileşen için son değişiklikleri içeren RSS bildirimini alır.

GET `/exports/rss/ (string: project) /`

Bir proje için son değişiklikleri içeren RSS bildirimini alır.

GET `/exports/rss/language/ (string: language) /`

Bir dil için son değişiklikleri içeren RSS bildirimini alır.

GET `/exports/rss/`

Weblate örneği için son değişiklikleri içeren RSS bildirimini alır.

Ayrıca bakınız:

[RSS on Wikipedia](#)

1.13 Weblate Client

2.7 sürümünde geldi: There has been full wlc utility support ever since Weblate 2.7. If you are using an older version some incompatibilities with the API might occur.

1.13.1 Installation

The Weblate Client is shipped separately and includes the Python module. To use the commands below, you need to install `wlc`:

```
pip3 install wlc
```

1.13.2 Docker kullanımı

The Weblate Client is also available as a Docker image.

The image is published on Docker Hub: <https://hub.docker.com/r/weblate/wlc>

Kurulum:

```
docker pull weblate/wlc
```

The Docker container uses Weblate's default settings and connects to the API deployed in localhost. The API URL and API_KEY can be configured through the arguments accepted by Weblate.

The command to launch the container uses the following syntax:

```
docker run --rm weblate/wlc [WLC_ARGS]
```

Example:

```
docker run --rm weblate/wlc --url https://hosted.weblate.org/api/ list-projects
```

You might want to pass your *Yapılandırma dosyaları* to the Docker container, the easiest approach is to add your current directory as /home/weblate volume:

```
docker run --volume $PWD:/home/weblate --rm weblate/wlc show
```

1.13.3 Getting started

The wlc configuration is stored in ~/.config/weblate (see *Yapılandırma dosyaları* for other locations), please create it to match your environment:

```
[weblate]
url = https://hosted.weblate.org/api/

[keys]
https://hosted.weblate.org/api/ = APIKEY
```

You can then invoke commands on the default server:

```
wlc ls
wlc commit sandbox/hello-world
```

Ayrıca bakınız:

Yapılandırma dosyaları

1.13.4 Synopsis

```
wlc [arguments] <command> [options]
```

Commands actually indicate which operation should be performed.

1.13.5 Açıklama

Weblate Client is a Python library and command-line utility to manage Weblate remotely using *Weblate's REST API*. The command-line utility can be invoked as **wlc** and is built-in on *wlc*.

Argümanlar

The program accepts the following arguments which define output format or which Weblate instance to use. These must be entered before any command.

--format {csv,json,text,html}

Specify the output format.

--url URL

Specify the API URL. Overrides any value found in the configuration file, see *Yapılandırma dosyaları*. The URL should end with /api/, for example https://hosted.weblate.org/api/.

--key KEY

Specify the API user key to use. Overrides any value found in the configuration file, see *Yapılandırma dosyaları*. You can find your key in your profile on Weblate.

--config PATH

Overrides the configuration file path, see *Yapılandırma dosyaları*.

--config-section SECTION

Overrides configuration file section in use, see *Yapılandırma dosyaları*.

Komutlar

Aşağıdaki komutlar kullanılabilir:

version

Geçerli sürümü yazdırır.

list-languages

Weblate'de kullanılan dilleri listeler.

list-projects

Weblate'deki projeleri listeler.

list-components

Weblate'deki bileşenleri listeler.

list-translations

Weblate'deki çevirileri listeler.

show

Weblate nesnesini (çeviri, bileşen veya proje) gösterir.

ls

Weblate nesnesini (çeviri, bileşen veya proje) listeler.

commit

Bir Weblate nesnesinde (çeviri, bileşen veya proje) yapılan değişiklikleri işler.

pull

Uzak depo değişikliklerini Weblate nesnesine (çeviri, bileşen veya proje) çeker.

push

Weblate nesnesindeki (çeviri, bileşen veya proje) değişiklikleri uzak depoya gönderir.

reset

0.7 sürümünde geldi: wlc 0.7'den beri desteklenmektedir.

Uzak depoya eşleşmesi için Weblate nesnesindeki (çeviri, bileşen veya proje) değişiklikleri sıfırlar.

cleanup

0.9 sürümünde geldi: wlc 0.9'dan beri desteklenmektedir.

Uzak depoya eşleşmesi için bir Weblate nesnesindeki (çeviri, bileşen veya proje) izlenmeyen değişiklikleri kaldırır.

repo

Belirli bir Weblate nesnesi (çeviri, bileşen veya proje) için depo durumunu görüntüler.

statistics

Belirli bir Weblate nesnesi (çeviri, bileşen veya proje) için ayrıntılı istatistikleri görüntüler.

lock-status

0.5 sürümünde geldi: wlc 0.5'ten beri desteklenmektedir.

Kilit durumunu görüntüler.

lock

0.5 sürümünde geldi: wlc 0.5'ten beri desteklenmektedir.

Bileşeni Weblate'de daha fazla çevrilmemesi için kilitler.

unlock

0.5 sürümünde geldi: wlc 0.5'ten beri desteklenmektedir.

Weblate bileşeninin çevirisinin kilidini açar.

changes

0.7 sürümünde geldi: wlc 0.7 ve Weblate 2.10'dan beri desteklenmektedir.

Belirli bir nesne için değişiklikleri görüntüler.

download

0.7 sürümünde geldi: wlc 0.7'den beri desteklenmektedir.

Bir çeviri dosyası indirir.

--convert

Dosya biçimini dönüştürür, belirtilmemişse sunucuda hiçbir dönüştürme olmaz ve dosya depoda olduğu gibi indirilir.

--output

Çıktının kaydedileceği dosyayı belirtir, belirtilmezse standart çıkışa yazdırılır.

upload

0.9 sürümünde geldi: wlc 0.9'dan beri desteklenmektedir.

Bir çeviri dosyasını karşıya yükler.

--overwrite

Yükleme sırasında mevcut çevirilerin üzerine yaz.

--input

İçeriğin okunduğu dosya, belirtilmemişse standart girişten okunur.

--method

Upload method to use, see [Import methods](#).

--fuzzy

Fuzzy (marked for edit) strings processing (*empty*, *process*, *approve*)

--author-name

Author name, to override currently authenticated user

--author-email

Author e-mail, to override currently authenticated user

İpucu: Her bir komutu çağırarak hakkında daha ayrıntılı bilgiyi `--help` seçeneğini kullanarak alabilirsiniz, örneğin:
`wlc ls --help`.

1.13.6 Yapılandırma dosyaları

.weblate, .weblate.ini, weblate.ini

1.6 sürümünde değişti: *.ini* uzantılı dosyalar da kabul edilir.

Proje başına yapılandırma dosyası

C:\Users\KULLANICIADI\AppData\weblate.ini

1.6 sürümünde geldi.

Windows'ta kullanıcı yapılandırma dosyası.

~/.config/weblate

Kullanıcı yapılandırma dosyası

/etc/xdg/weblate

Sistem genelinde yapılandırma dosyası

Program XDG tanımlamasını takip eder, böylece `XDG_CONFIG_HOME` veya `XDG_CONFIG_DIRS` ortam değişkenlerine göre yapılandırma dosyalarının yerleşimini ayarlayabilirsiniz. Windows'ta APPDATA dizini, yapılandırma dosyası için tercih edilen konumdur.

[weblate] bölümünde aşağıdaki ayarlar yapılandırılabilir (bunu *--config-section* ile özelleştirebilirsiniz):

key

Weblate'e erişmek için API ANAHTARI.

url

API sunucusu URL'si, öntanımlı olarak `http://127.0.0.1:8000/api/` şeklindedir.

translation

Öntanımlı çevirinin yolu - bileşen veya proje.

Yapılandırma dosyası bir INI dosyasıdır, örneğin:

```
[weblate]
url = https://hosted.weblate.org/api/
key = APIKEY
translation = weblate/application
```

Ek olarak API anahtarları [keys] bölümünde depolanabilir:

```
[keys]
https://hosted.weblate.org/api/ = APIKEY
```

This allows you to store keys in your personal settings, while using the `.weblate` configuration in the VCS repository so that `wlc` knows which server it should talk to.

1.13.7 Örnekler

Geçerli program sürümünü yazdır:

```
$ wlc version
version: 0.1
```

Tüm projeleri listele:

```
$ wlc list-projects
name: Hello
slug: hello
url: http://example.com/api/projects/hello/
web: https://weblate.org/
web_url: http://example.com/projects/hello/
```

Çeviri dosyasını karşıya yükle:

```
$ wlc upload project/component/language --input /tmp/hello.po
```

Ayrıca wlc'nin hangi proje üzerinde çalışması gerektiğini de belirleyebilirsiniz:

```
$ cat .weblate
[weblate]
url = https://hosted.weblate.org/api/
translation = weblate/application

$ wlc show
branch: main
file_format: po
source_language: en
filemask: weblate/locale/*/LC_MESSAGES/django.po
git_export: https://hosted.weblate.org/git/weblate/application/
license: GPL-3.0+
license_url: https://spdx.org/licenses/GPL-3.0+
name: Application
new_base: weblate/locale/django.pot
project: weblate
repo: git://github.com/WeblateOrg/weblate.git
slug: application
template:
url: https://hosted.weblate.org/api/components/weblate/application/
vcs: git
web_url: https://hosted.weblate.org/projects/weblate/application/
```

Bu kurulumla, geçerli projede bekleyen değişiklikleri işlemek kolaydır:

```
$ wlc commit
```

1.14 Weblate's Python API

1.14.1 Installation

The Python API is shipped separately, you need to install the *Weblate Client* (wlc) to have it.

```
pip install wlc
```

1.14.2 wlc

WeblateException

exception wlc.WeblateException

Base class for all exceptions.

Weblate

```
class wlc.Weblate (key="", url=None, config=None)
```

Parametreler

- **key** (*str*) – User key
- **url** (*str*) – API server URL, if not specified default is used
- **config** (*wlc.config.WeblateConfig*) – Configuration object, overrides any other parameters.

Access class to the API, define API key and optionally API URL.

```
get (path)
```

Parametreler

path (*str*) – Request path

Dönüş türü

object

Performs a single API GET call.

```
post (path, **kwargs)
```

Parametreler

path (*str*) – Request path

Dönüş türü

object

Performs a single API GET call.

1.14.3 wlc.config

WeblateConfig

```
class wlc.config.WeblateConfig (section='wlc')
```

Parametreler

section (*str*) – Configuration section to use

Configuration file parser following XDG specification.

```
load (path=None)
```

Parametreler

path (*str*) – Path from which to load configuration.

Loads configuration from a file, if none is specified, it loads from the *wlc* configuration file (`~/ .config/wlc`) placed in your XDG configuration path (`/etc/xdg/wlc`).

1.14.4 `wlc.main`

`wlc.main.main` (*settings* = *None*, *stdout* = *None*, *args* = *None*)

Parametreler

- **settings** (*list*) – Settings to override as list of tuples
- **stdout** (*object*) – stdout file object for printing output, uses `sys.stdout` as default
- **args** (*list*) – Command-line arguments to process, uses `sys.args` as default

Main entry point for command-line interface.

`@wlc.main.register_command` (*command*)

Decorator to register *Command* class in main parser used by *main()*.

Command

class `wlc.main.Command` (*args*, *config*, *stdout* = *None*)

Main class for invoking commands.

2.1 Configuration instructions

2.1.1 Installing Weblate

Installing using Docker

With dockerized Weblate deployment you can get your personal Weblate instance up and running in seconds. All of Weblate's dependencies are already included. PostgreSQL is set up as the default database.

Hardware requirements

Weblate should run on any contemporary hardware without problems, the following is the minimal configuration required to run Weblate on a single host (Weblate, database and webserver):

- 2 GB of RAM
- 2 CPU cores
- 1 GB of storage space

The more memory the better - it is used for caching on all levels (filesystem, database and Weblate).

Many concurrent users increases the amount of needed CPU cores. For hundreds of translation components at least 4 GB of RAM is recommended.

The typical database storage usage is around 300 MB per 1 million hosted words. Storage space needed for cloned repositories varies, but Weblate tries to keep their size minimal by doing shallow clones.

Not: Actual requirements for your installation of Weblate vary heavily based on the size of the translations managed in it.

Installation

The following examples assume you have a working Docker environment, with `docker-compose` installed. Please check the Docker documentation for instructions.

1. Clone the weblate-docker repo:

```
git clone https://github.com/WeblateOrg/docker-compose.git weblate-docker
cd weblate-docker
```

2. Create a `docker-compose.override.yml` file with your settings. See [Docker environment variables](#) for full list of environment variables.

```
version: '3'
services:
  weblate:
    ports:
      - 80:8080
    environment:
      WEBLATE_EMAIL_HOST: smtp.example.com
      WEBLATE_EMAIL_HOST_USER: user
      WEBLATE_EMAIL_HOST_PASSWORD: pass
      WEBLATE_SERVER_EMAIL: weblate@example.com
      WEBLATE_DEFAULT_FROM_EMAIL: weblate@example.com
      WEBLATE_SITE_DOMAIN: weblate.example.com
      WEBLATE_ADMIN_PASSWORD: password for the admin user
      WEBLATE_ADMIN_EMAIL: weblate.admin@example.com
```

Not: If `WEBLATE_ADMIN_PASSWORD` is not set, the admin user is created with a random password shown on first startup.

The provided example makes Weblate listen on port 80, edit the port mapping in the `docker-compose.override.yml` file to change it.

3. Start Weblate containers:

```
docker-compose up
```

Enjoy your Weblate deployment, it's accessible on port 80 of the `weblate` container.

2.15-2 sürümünde değişti: The setup has changed recently, priorly there was separate web server container, since 2.15-2 the web server is embedded in the Weblate container.

3.7.1-6 sürümünde değişti: In July 2019 (starting with the 3.7.1-6 tag), the containers are not running as a root user. This has changed the exposed port from 80 to 8080.

Ayrıca bakınız:

[Invoking management commands](#)

Choosing Docker hub tag

You can use following tags on Docker hub, see <https://hub.docker.com/r/weblate/weblate/tags/> for full list of available ones.

Etiket adı	Açıklama	Use case
latest	Weblate stable release, matches latest tagged release	Rolling updates in a production environment
<VERSION>-<PLATFORM>	Weblate kararlı sürümü	Well defined deploy in a production environment
edge	Weblate stable release with development changes in the Docker container (for example updated dependencies)	Rolling updates in a staging environment
edge-<DATE>-<PLATFORM>	Weblate stable release with development changes in the Docker container (for example updated dependencies)	Well defined deploy in a staging environment
bleeding	Development version Weblate from Git	Rolling updates to test upcoming Weblate features
bleeding-<DATE>-<PLATFORM>	Development version Weblate from Git	Well defined deploy to test upcoming Weblate features

Every image is tested by our CI before it gets published, so even the *bleeding* version should be quite safe to use.

Docker container with HTTPS support

Please see [Installation](#) for generic deployment instructions, this section only mentions differences compared to it.

Using own SSL certificates

3.8-3 sürümünde geldi.

In case you have own SSL certificate you want to use, simply place the files into the Weblate data volume (see [Docker container volumes](#)):

- `ssl/fullchain.pem` containing the certificate including any needed CA certificates
- `ssl/privkey.pem` containing the private key

Both of these files must be owned by the same user as the one starting the docker container and have file mask set to 600 (readable and writable only by the owning user).

Additionally, Weblate container will now accept SSL connections on port 4443, you will want to include the port forwarding for HTTPS in docker compose override:

```
version: '3'
services:
  weblate:
    ports:
      - 80:8080
      - 443:4443
```

If you already host other sites on the same server, it is likely ports 80 and 443 are used by a reverse proxy, such as NGINX. To pass the HTTPS connection from NGINX to the docker container, you can use the following configuration:

```
server {
    listen 443;
    listen [::]:443;

    server_name <SITE_URL>;
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

ssl_certificate /etc/letsencrypt/live/<SITE>/fullchain.pem;
ssl_certificate_key /etc/letsencrypt/live/<SITE>/privkey.pem;

location / {
    proxy_set_header HOST $host;
    proxy_set_header X-Forwarded-Proto https;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Host $server_name;
    proxy_pass https://127.0.0.1:<EXPOSED_DOCKER_PORT>;
}
}

```

Replace <SITE_URL>, <SITE> and <EXPOSED_DOCKER_PORT> with actual values from your environment.

Automatic SSL certificates using Let's Encrypt

In case you want to use [Let's Encrypt](#) automatically generated SSL certificates on public installation, you need to add a reverse HTTPS proxy an additional Docker container, [https-portal](#) will be used for that. This is made use of in the `docker-compose-https.yml` file. Then create a `docker-compose-https.override.yml` file with your settings:

```

version: '3'
services:
  weblate:
    environment:
      WEBLATE_EMAIL_HOST: smtp.example.com
      WEBLATE_EMAIL_HOST_USER: user
      WEBLATE_EMAIL_HOST_PASSWORD: pass
      WEBLATE_SITE_DOMAIN: weblate.example.com
      WEBLATE_ADMIN_PASSWORD: password for admin user
  https-portal:
    environment:
      DOMAINS: 'weblate.example.com -> http://weblate:8080'

```

Whenever invoking **docker-compose** you need to pass both files to it, and then do:

```

docker-compose -f docker-compose-https.yml -f docker-compose-https.override.yml
↩️ build
docker-compose -f docker-compose-https.yml -f docker-compose-https.override.yml up

```

Upgrading the Docker container

Usually it is good idea to only update the Weblate container and keep the PostgreSQL container at the version you have, as upgrading PostgreSQL is quite painful and in most cases does not bring many benefits.

4.10-1 sürümünde değişti: Since Weblate 4.10-1, the Docker container uses Django 4.0 what requires PostgreSQL 10 or newer, please upgrade it prior to upgrading Weblate. See [Upgrade from 4.9 to 4.10](#) and [Upgrading PostgreSQL container](#).

You can do this by sticking with the existing docker-compose and just pull the latest images and then restart:

```

# Fetch latest versions of the images
docker-compose pull
# Stop and destroy the containers
docker-compose down
# Spawn new containers in the background
docker-compose up -d

```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
# Follow the logs during upgrade
docker-compose logs -f
```

The Weblate database should be automatically migrated on first startup, and there should be no need for additional manual actions.

Not: Upgrades across major versions are not supported by Weblate. For example, if you are on 3.x series and want to upgrade to 4.x, first upgrade to the latest 4.0.x-y image (at time of writing this it is the 4.0.4-5), which will do the migration and then continue upgrading to newer versions.

You might also want to update the `docker-compose` repository, though it's not needed in most case. See [Upgrading PostgreSQL container](#) for upgrading the PostgreSQL server.

Upgrading PostgreSQL container

PostgreSQL containers do not support automatic upgrading between version, you need to perform the upgrade manually. Following steps show one of the options of upgrading.

Ayrıca bakınız:

<https://github.com/docker-library/postgres/issues/37>

1. Stop Weblate container:

```
docker-compose stop weblate cache
```

2. Backup the database:

```
docker-compose exec database pg_dumpall --clean --username weblate > backup.sql
```

3. Stop the database container:

```
docker-compose stop database
```

4. Remove the PostgreSQL volume:

```
docker-compose rm -v database
docker volume remove weblate_postgres-data
```

5. Adjust `docker-compose.yml` to use new PostgreSQL version.

6. Start the database container:

```
docker-compose up -d database
```

7. Restore the database from the backup:

```
cat backup.sql | docker-compose exec -T database psql --username weblate --
↳ dbname postgres
```

8. Start all remaining containers:

```
docker-compose up -d
```

Yönetici olarak oturum açma

After container setup, you can sign in as *admin* user with password provided in `WEBLATE_ADMIN_PASSWORD`, or a random password generated on first start if that was not set.

To reset *admin* password, restart the container with `WEBLATE_ADMIN_PASSWORD` set to new password.

Ayrıca bakınız:

`WEBLATE_ADMIN_PASSWORD`, `WEBLATE_ADMIN_NAME`, `WEBLATE_ADMIN_EMAIL`

Number of processes and memory consumption

The number of worker processes for both uWSGI and Celery is determined automatically based on number of CPUs. This works well for most cloud virtual machines as these typically have few CPUs and good amount of memory.

In case you have a lot of CPU cores and hit out of memory issues, try reducing number of workers:

```
environment:
  WEBLATE_WORKERS: 2
```

You can also fine-tune individual worker categories:

```
environment:
  WEB_WORKERS: 4
  CELERY_MAIN_OPTIONS: --concurrency 2
  CELERY_NOTIFY_OPTIONS: --concurrency 1
  CELERY_TRANSLATE_OPTIONS: --concurrency 1
```

Ayrıca bakınız:

`WEBLATE_WORKERS`, `CELERY_MAIN_OPTIONS`, `CELERY_NOTIFY_OPTIONS`, `CELERY_MEMORY_OPTIONS`, `CELERY_TRANSLATE_OPTIONS`, `CELERY_BACKUP_OPTIONS`, `CELERY_BEAT_OPTIONS`, `WEB_WORKERS`

Yatay ölçeklendirme

4.6 sürümünde geldi.

You can run multiple Weblate containers to scale the service horizontally. The `/app/data` volume has to be shared by all containers, it is recommended to use cluster filesystem such as GlusterFS for this. The `/app/cache` volume should be separate for each container.

Each Weblate container has defined role using `WEBLATE_SERVICE` environment variable. Please follow carefully the documentation as some of the services should be running just once in the cluster and the ordering of the services matters as well.

You can find example setup in the `docker-compose` repo as `docker-compose-split.yml`.

Docker environment variables

Many of Weblate's *Yapılandırma* can be set in the Docker container using environment variables:

Generic settings

WEBLATE_DEBUG

Configures Django debug mode using *DEBUG*.

Örnek:

```
environment:
  WEBLATE_DEBUG: 1
```

Ayrıca bakınız:

Disable debug mode

WEBLATE_LOGLEVEL

Configures the logging verbosity.

WEBLATE_LOGLEVEL_DATABASE

Configures the logging of the database queries verbosity.

WEBLATE_SITE_TITLE

Changes the site-title shown in the header of all pages.

WEBLATE_SITE_DOMAIN

Site etki alanını yapılandırır. Bu parametre gereklidir.

Ayrıca bakınız:

Set correct site domain, SITE_DOMAIN

WEBLATE_ADMIN_NAME

WEBLATE_ADMIN_EMAIL

Configures the site-admin's name and e-mail. It is used for both *ADMINS* setting and creating *admin* user (see *WEBLATE_ADMIN_PASSWORD* for more info on that).

Örnek:

```
environment:
  WEBLATE_ADMIN_NAME: Weblate admin
  WEBLATE_ADMIN_EMAIL: noreply@example.com
```

Ayrıca bakınız:

Yönetici olarak oturum açma, Properly configure admins, ADMINS

WEBLATE_ADMIN_PASSWORD

Sets the password for the *admin* user.

- If not set and *admin* user does not exist, it is created with a random password shown on first container startup.
- If not set and *admin* user exists, no action is performed.
- If set the *admin* user is adjusted on every container startup to match *WEBLATE_ADMIN_PASSWORD*, *WEBLATE_ADMIN_NAME* and *WEBLATE_ADMIN_EMAIL*.

Uyarı: It might be a security risk to store password in the configuration file. Consider using this variable only for initial setup (or let Weblate generate random password on initial startup) or for password recovery.

Ayrıca bakınız:

Yönetici olarak oturum açma, WEBLATE_ADMIN_PASSWORD, WEBLATE_ADMIN_PASSWORD_FILE, WEBLATE_ADMIN_NAME, WEBLATE_ADMIN_EMAIL

WEBLATE_ADMIN_PASSWORD_FILE

Sets the path to a file containing the password for the *admin* user.

Ayrıca bakınız:

WEBLATE_ADMIN_PASSWORD

WEBLATE_SERVER_EMAIL

The email address that error messages are sent from.

Ayrıca bakınız:

SERVER_EMAIL, *Configure e-mail sending*

WEBLATE_DEFAULT_FROM_EMAIL

Configures the address for outgoing e-mails.

Ayrıca bakınız:

DEFAULT_FROM_EMAIL, *Configure e-mail sending*

WEBLATE_CONTACT_FORM

Configures contact form behavior, see *CONTACT_FORM*.

WEBLATE_ALLOWED_HOSTS

Configures allowed HTTP hostnames using *ALLOWED_HOSTS*.

Defaults to * which allows all hostnames.

Örnek:

```
environment:
  WEBLATE_ALLOWED_HOSTS: weblate.example.com,example.com
```

Ayrıca bakınız:

ALLOWED_HOSTS, *Allowed hosts setup*, *Set correct site domain*

WEBLATE_REGISTRATION_OPEN

Configures whether registrations are open by toggling *REGISTRATION_OPEN*.

Örnek:

```
environment:
  WEBLATE_REGISTRATION_OPEN: 0
```

WEBLATE_REGISTRATION_ALLOW_BACKENDS

Configure which authentication methods can be used to create new account via *REGISTRATION_ALLOW_BACKENDS*.

Örnek:

```
environment:
  WEBLATE_REGISTRATION_OPEN: 0
  WEBLATE_REGISTRATION_ALLOW_BACKENDS: azuread-oauth2,azuread-tenant-
  ↪oauth2
```

WEBLATE_TIME_ZONE

Configures the used time zone in Weblate, see *TIME_ZONE*.

Not: To change the time zone of the Docker container itself, use the TZ environment variable.

Örnek:


```
environment:
  WEBLATE_TIME_ZONE: Europe/Prague
```

WEBLATE_ENABLE_HTTPS

Makes Weblate assume it is operated behind a reverse HTTPS proxy, it makes Weblate use HTTPS in e-mail and API links or set secure flags on cookies.

İpucu: Please see [ENABLE_HTTPS](#) documentation for possible caveats.

Not: This does not make the Weblate container accept HTTPS connections, you need to configure that as well, see [Docker container with HTTPS support](#) for examples.

Örnek:

```
environment:
  WEBLATE_ENABLE_HTTPS: 1
```

Ayrıca bakınız:

[ENABLE_HTTPS](#) Set correct site domain, [WEBLATE_SECURE_PROXY_SSL_HEADER](#)

WEBLATE_INTERLEDGER_PAYMENT_POINTERS

4.12.1 sürümünde geldi.

Lets Weblate set the *meta[name =monetization]* field in the head of the document. If multiple are specified, chooses one randomly.

Ayrıca bakınız:

[INTERLEDGER_PAYMENT_POINTERS](#)

WEBLATE_IP_PROXY_HEADER

Lets Weblate fetch the IP address from any given HTTP header. Use this when using a reverse proxy in front of the Weblate container.

Enables [IP_BEHIND_REVERSE_PROXY](#) and sets [IP_PROXY_HEADER](#).

Not: The format must conform to Django's expectations. Django [transforms](#) raw HTTP header names as follows:

- converts all characters to uppercase
- replaces any hyphens with underscores
- prepends HTTP_ prefix

So X-Forwarded-For would be mapped to HTTP_X_FORWARDED_FOR.

Örnek:

```
environment:
  WEBLATE_IP_PROXY_HEADER: HTTP_X_FORWARDED_FOR
```

WEBLATE_SECURE_PROXY_SSL_HEADER

A tuple representing a HTTP header/value combination that signifies a request is secure. This is needed when Weblate is running behind a reverse proxy doing SSL termination which does not pass standard HTTPS headers.

Örnek:

```
environment:
  WEBLATE_SECURE_PROXY_SSL_HEADER: HTTP_X_FORWARDED_PROTO,https
```

Ayrıca bakınız:

`SECURE_PROXY_SSL_HEADER`

WEBLATE_REQUIRE_LOGIN

Enables `REQUIRE_LOGIN` to enforce authentication on whole Weblate.

Örnek:

```
environment:
  WEBLATE_REQUIRE_LOGIN: 1
```

WEBLATE_LOGIN_REQUIRED_URLS_EXCEPTIONS

WEBLATE_ADD_LOGIN_REQUIRED_URLS_EXCEPTIONS

WEBLATE_REMOVE_LOGIN_REQUIRED_URLS_EXCEPTIONS

Adds URL exceptions for authentication required for the whole Weblate installation using `LOGIN_REQUIRED_URLS_EXCEPTIONS`.

You can either replace whole settings, or modify default value using ADD and REMOVE variables.

WEBLATE_GOOGLE_ANALYTICS_ID

Configures ID for Google Analytics by changing `GOOGLE_ANALYTICS_ID`.

WEBLATE_GITHUB_USERNAME

Configures GitHub username for GitHub pull-requests by changing `GITHUB_USERNAME`.

Ayrıca bakınız:

GitHub pull requests

WEBLATE_GITHUB_TOKEN

4.3 sürümünde geldi.

Configures GitHub personal access token for GitHub pull-requests via API by changing `GITHUB_TOKEN`.

Ayrıca bakınız:

GitHub pull requests

WEBLATE_GITLAB_USERNAME

Configures GitLab username for GitLab merge-requests by changing `GITLAB_USERNAME`

Ayrıca bakınız:

GitLab merge requests

WEBLATE_GITLAB_TOKEN

Configures GitLab personal access token for GitLab merge-requests via API by changing `GITLAB_TOKEN`

Ayrıca bakınız:

GitLab merge requests

WEBLATE_PAGURE_USERNAME

Configures Pagure username for Pagure merge-requests by changing `PAGURE_USERNAME`

Ayrıca bakınız:

Pagure merge requests

WEBLATE_PAGURE_TOKEN

Configures Pagure personal access token for Pagure merge-requests via API by changing *PAGURE_TOKEN*

Ayrıca bakınız:

Pagure merge requests

WEBLATE_DEFAULT_PULL_MESSAGE

Configures the default title and message for pull requests via API by changing *DEFAULT_PULL_MESSAGE*

Ayrıca bakınız:

DEFAULT_PULL_MESSAGE

WEBLATE_SIMPLIFY_LANGUAGES

Configures the language simplification policy, see *SIMPLIFY_LANGUAGES*.

WEBLATE_DEFAULT_ACCESS_CONTROL

Configures the default *Erişim denetimi* for new projects, see *DEFAULT_ACCESS_CONTROL*.

WEBLATE_DEFAULT_RESTRICTED_COMPONENT

Configures the default value for *Restricted access* for new components, see *DEFAULT_RESTRICTED_COMPONENT*.

WEBLATE_DEFAULT_TRANSLATION_PROPAGATION

Configures the default value for *Çeviri yaymaya izin ver* for new components, see *DEFAULT_TRANSLATION_PROPAGATION*.

WEBLATE_DEFAULT_COMMITER_EMAIL

Configures *DEFAULT_COMMITER_EMAIL*.

WEBLATE_DEFAULT_COMMITER_NAME

Configures *DEFAULT_COMMITER_NAME*.

WEBLATE_DEFAULT_SHARED_TM

Configures *DEFAULT_SHARED_TM*.

WEBLATE_AKISMET_API_KEY

Configures the Akismet API key, see *AKISMET_API_KEY*.

WEBLATE_GPG_IDENTITY

Configures GPG signing of commits, see *WEBLATE_GPG_IDENTITY*.

Ayrıca bakınız:

Signing Git commits with GnuPG

WEBLATE_URL_PREFIX

Configures URL prefix where Weblate is running, see *URL_PREFIX*.

WEBLATE_SILENCED_SYSTEM_CHECKS

Configures checks which you do not want to be displayed, see *SILENCED_SYSTEM_CHECKS*.

WEBLATE_CSP_SCRIPT_SRC**WEBLATE_CSP_IMG_SRC****WEBLATE_CSP_CONNECT_SRC****WEBLATE_CSP_STYLE_SRC**

WEBLATE_CSP_FONT_SRC

Allows to customize Content-Security-Policy HTTP header.

Ayrıca bakınız:

Content security policy, *CSP_SCRIPT_SRC*, *CSP_IMG_SRC*, *CSP_CONNECT_SRC*, *CSP_STYLE_SRC*, *CSP_FONT_SRC*

WEBLATE_LICENSE_FILTER

Configures *LICENSE_FILTER*.

WEBLATE_LICENSE_REQUIRED

Configures *LICENSE_REQUIRED*

WEBLATE_WEBSITE_REQUIRED

Configures *WEBSITE_REQUIRED*

WEBLATE_HIDE_VERSION

Configures *HIDE_VERSION*.

WEBLATE_BASIC_LANGUAGES

Configures *BASIC_LANGUAGES*.

WEBLATE_DEFAULT_AUTO_WATCH

Configures *DEFAULT_AUTO_WATCH*.

WEBLATE_RATELIMIT_ATTEMPTS

WEBLATE_RATELIMIT_LOCKOUT

WEBLATE_RATELIMIT_WINDOW

4.6 sürümünde geldi.

Oran sınırlayıcıyı yapılandırır.

İpucu: You can set configuration for any rate limiter scopes. To do that add `WEBLATE_` prefix to any of setting described in *Rate limiting*.

Ayrıca bakınız:

Rate limiting, *RATELIMIT_ATTEMPTS*, *RATELIMIT_WINDOW*, *RATELIMIT_LOCKOUT*

WEBLATE_API_RATELIMIT_ANON

WEBLATE_API_RATELIMIT_USER

4.11 sürümünde geldi.

Configures API rate limiting. Defaults to 100/day for anonymous and 5000/hour for authenticated users.

Ayrıca bakınız:

API rate limiting

WEBLATE_ENABLE_HOOKS

4.13 sürümünde geldi.

Configures *ENABLE_HOOKS*.

WEBLATE_ENABLE_AVATARS

4.6.1 sürümünde geldi.

Configures *ENABLE_AVATARS*.

WEBLATE_LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH

4.9 sürümünde geldi.

Configures *LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH*.

WEBLATE_SSH_EXTRA_ARGS

4.9 sürümünde geldi.

Configures *SSH_EXTRA_ARGS*.

WEBLATE_BORG_EXTRA_ARGS

4.9 sürümünde geldi.

Configures *BORG_EXTRA_ARGS*.

Automatic suggestion settings

4.13 sürümünde değişti: Automatic suggestion services are now configured in the user interface, see [Configuring automatic suggestions](#).

The existing environment variables are imported during the migration to Weblate 4.13, but changing them will not have any further effect.

Authentication settings**LDAP****WEBLATE_AUTH_LDAP_SERVER_URI****WEBLATE_AUTH_LDAP_USER_DN_TEMPLATE****WEBLATE_AUTH_LDAP_USER_ATTR_MAP****WEBLATE_AUTH_LDAP_BIND_DN****WEBLATE_AUTH_LDAP_BIND_PASSWORD****WEBLATE_AUTH_LDAP_BIND_PASSWORD_FILE**

Path to the file containing the LDAP server bind password.

Ayrıca bakınız:

WEBLATE_AUTH_LDAP_BIND_PASSWORD

WEBLATE_AUTH_LDAP_CONNECTION_OPTION_REFERRALS**WEBLATE_AUTH_LDAP_USER_SEARCH****WEBLATE_AUTH_LDAP_USER_SEARCH_FILTER****WEBLATE_AUTH_LDAP_USER_SEARCH_UNION****WEBLATE_AUTH_LDAP_USER_SEARCH_UNION_DELIMITER**

LDAP authentication configuration.

Example for direct bind:

```
environment:
  WEBLATE_AUTH_LDAP_SERVER_URI: ldap://ldap.example.org
  WEBLATE_AUTH_LDAP_USER_DN_TEMPLATE: uid =%(user)s,ou =People,dc =example,
  ↪dc =net
  # map weblate 'full_name' to ldap 'name' and weblate 'email' attribute to
  ↪'mail' ldap attribute.
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
# another example that can be used with OpenLDAP: 'full_name:cn,email:mail'
WEBLATE_AUTH_LDAP_USER_ATTR_MAP: full_name:name,email:mail
```

Example for search and bind:

```
environment:
  WEBLATE_AUTH_LDAP_SERVER_URI: ldap://ldap.example.org
  WEBLATE_AUTH_LDAP_BIND_DN: CN =ldap,CN =Users,DC =example,DC =com
  WEBLATE_AUTH_LDAP_BIND_PASSWORD: password
  WEBLATE_AUTH_LDAP_USER_ATTR_MAP: full_name:name,email:mail
  WEBLATE_AUTH_LDAP_USER_SEARCH: CN =Users,DC =example,DC =com
```

Example for union search and bind:

```
environment:
  WEBLATE_AUTH_LDAP_SERVER_URI: ldap://ldap.example.org
  WEBLATE_AUTH_LDAP_BIND_DN: CN =ldap,CN =Users,DC =example,DC =com
  WEBLATE_AUTH_LDAP_BIND_PASSWORD: password
  WEBLATE_AUTH_LDAP_USER_ATTR_MAP: full_name:name,email:mail
  WEBLATE_AUTH_LDAP_USER_SEARCH_UNION: ou =users,dc =example,
↪dc =com|ou =otherusers,dc =example,dc =com
```

Example with search and bind against Active Directory:

```
environment:
  WEBLATE_AUTH_LDAP_BIND_DN: CN =ldap,CN =Users,DC =example,DC =com
  WEBLATE_AUTH_LDAP_BIND_PASSWORD: password
  WEBLATE_AUTH_LDAP_SERVER_URI: ldap://ldap.example.org
  WEBLATE_AUTH_LDAP_CONNECTION_OPTION_REFERRALS: 0
  WEBLATE_AUTH_LDAP_USER_ATTR_MAP: full_name:name,email:mail
  WEBLATE_AUTH_LDAP_USER_SEARCH: CN =Users,DC =example,DC =com
  WEBLATE_AUTH_LDAP_USER_SEARCH_FILTER: (sAMAccountName =%(user)s)
```

Ayrıca bakınız:

LDAP authentication

GitHub

WEBLATE_SOCIAL_AUTH_GITHUB_KEY

WEBLATE_SOCIAL_AUTH_GITHUB_SECRET

WEBLATE_SOCIAL_AUTH_GITHUB_ORG_KEY

WEBLATE_SOCIAL_AUTH_GITHUB_ORG_SECRET

WEBLATE_SOCIAL_AUTH_GITHUB_ORG_NAME

WEBLATE_SOCIAL_AUTH_GITHUB_TEAM_KEY

WEBLATE_SOCIAL_AUTH_GITHUB_TEAM_SECRET

WEBLATE_SOCIAL_AUTH_GITHUB_TEAM_ID

Enables *GitHub authentication*.

Bitbucket

WEBLATE_SOCIAL_AUTH_BITBUCKET_OAUTH2_KEY

WEBLATE_SOCIAL_AUTH_BITBUCKET_OAUTH2_SECRET

WEBLATE_SOCIAL_AUTH_BITBUCKET_KEY

WEBLATE_SOCIAL_AUTH_BITBUCKET_SECRET

Enables *Bitbucket authentication*.

Facebook

WEBLATE_SOCIAL_AUTH_FACEBOOK_KEY

WEBLATE_SOCIAL_AUTH_FACEBOOK_SECRET

Enables *Facebook OAuth 2*.

Google

WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_KEY

WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET

WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_WHITELISTED_DOMAINS

WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_WHITELISTED_EMAILS

Enables *Google OAuth 2*.

GitLab

WEBLATE_SOCIAL_AUTH_GITLAB_KEY

WEBLATE_SOCIAL_AUTH_GITLAB_SECRET

WEBLATE_SOCIAL_AUTH_GITLAB_API_URL

Enables *GitLab OAuth 2*.

Azure Active Directory

WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_KEY

WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_SECRET

Enables Azure Active Directory authentication, see *Microsoft Azure Active Directory*.

Azure Active Directory with Tenant support

`WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_KEY`

`WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_SECRET`

`WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_TENANT_ID`

Enables Azure Active Directory authentication with Tenant support, see *Microsoft Azure Active Directory*.

Keycloak

`WEBLATE_SOCIAL_AUTH_KEYCLOAK_KEY`

`WEBLATE_SOCIAL_AUTH_KEYCLOAK_SECRET`

`WEBLATE_SOCIAL_AUTH_KEYCLOAK_PUBLIC_KEY`

`WEBLATE_SOCIAL_AUTH_KEYCLOAK_ALGORITHM`

`WEBLATE_SOCIAL_AUTH_KEYCLOAK_AUTHORIZATION_URL`

`WEBLATE_SOCIAL_AUTH_KEYCLOAK_ACCESS_TOKEN_URL`

`WEBLATE_SOCIAL_AUTH_KEYCLOAK_TITLE`

`WEBLATE_SOCIAL_AUTH_KEYCLOAK_IMAGE`

Enables Keycloak authentication, see [documentation](#).

Linux vendors

You can enable authentication using Linux vendors authentication services by setting following variables to any value.

`WEBLATE_SOCIAL_AUTH_FEDORA`

`WEBLATE_SOCIAL_AUTH_OPENSUSE`

`WEBLATE_SOCIAL_AUTH_UBUNTU`

Slack

`WEBLATE_SOCIAL_AUTH_SLACK_KEY`

`SOCIAL_AUTH_SLACK_SECRET`

Enables Slack authentication, see *Slack*.

OpenID Connect

4.13-1 sürümünde geldi.

`WEBLATE_SOCIAL_AUTH_OIDC_OIDC_ENDPOINT`

`WEBLATE_SOCIAL_AUTH_OIDC_KEY`

`WEBLATE_SOCIAL_AUTH_OIDC_SECRET`

WEBLATE_SOCIAL_AUTH_OIDC_USERNAME_KEY

Configures generic OpenID Connect intergration.

Ayrıca bakınız:

[OIDC \(OpenID Connect\)](#)

SAML

Self-signed SAML keys are automatically generated on first container startup. In case you want to use own keys, place the certificate and private key in `/app/data/ssl/saml.crt` and `/app/data/ssl/saml.key`.

WEBLATE_SAML_IDP_ENTITY_ID**WEBLATE_SAML_IDP_URL****WEBLATE_SAML_IDP_X509CERT****WEBLATE_SAML_IDP_IMAGE****WEBLATE_SAML_IDP_TITLE**

SAML Identity Provider settings, see [SAML authentication](#).

Other authentication settings**WEBLATE_NO_EMAIL_AUTH**

Disables e-mail authentication when set to any value. See [Turning off password authentication](#).

PostgreSQL database setup

The database is created by `docker-compose.yml`, so these settings affect both Weblate and PostgreSQL containers.

Ayrıca bakınız:

[Database setup for Weblate](#)

POSTGRES_PASSWORD

PostgreSQL password.

POSTGRES_PASSWORD_FILE

Path to the file containing the PostgreSQL password. Use as an alternative to `POSTGRES_PASSWORD`.

POSTGRES_USER

PostgreSQL username.

POSTGRES_DATABASE

PostgreSQL database name.

POSTGRES_HOST

PostgreSQL server hostname or IP address. Defaults to `database`.

POSTGRES_PORT

PostgreSQL server port. Defaults to none (uses the default value).

POSTGRES_SSL_MODE

Configure how PostgreSQL handles SSL in connection to the server, for possible choices see [SSL Mode Descriptions](#)

POSTGRES_ALTER_ROLE

Configures name of role to alter during migrations, see *Configuring Weblate to use PostgreSQL*.

POSTGRES_CONN_MAX_AGE

4.8.1 sürümünde geldi.

The lifetime of a database connection, as an integer of seconds. Use 0 to close database connections at the end of each request (this is the default behavior).

Enabling connection persistence will typically, cause more open connection to the database. Please adjust your database configuration prior enabling.

Example configuration:

```
environment:
  POSTGRES_CONN_MAX_AGE: 3600
```

Ayrıca bakınız:

`CONN_MAX_AGE`, *Persistent connections*

POSTGRES_DISABLE_SERVER_SIDE_CURSORS

4.9.1 sürümünde geldi.

Disable server side cursors in the database. This is necessary in some **pgbouncer** setups.

Example configuration:

```
environment:
  POSTGRES_DISABLE_SERVER_SIDE_CURSORS: 1
```

Ayrıca bakınız:

`DISABLE_SERVER_SIDE_CURSORS`, *Transaction pooling and server-side cursors*

Database backup settings

Ayrıca bakınız:

Dumped data for backups

WEBLATE_DATABASE_BACKUP

Configures the daily database dump using `DATABASE_BACKUP`. Defaults to `plain`.

Caching server setup

Using Redis is strongly recommended by Weblate and you have to provide a Redis instance when running Weblate in Docker.

Ayrıca bakınız:

Enable caching

REDIS_HOST

The Redis server hostname or IP address. Defaults to `cache`.

REDIS_PORT

The Redis server port. Defaults to `6379`.

REDIS_DB

The Redis database number, defaults to `1`.

REDIS_PASSWORD

The Redis server password, not used by default.

REDIS_PASSWORD_FILE

Path to the file containing the Redis server password.

Ayrıca bakınız:

`REDIS_PASSWORD`

REDIS_TLS

Enables using SSL for Redis connection.

REDIS_VERIFY_SSL

Can be used to disable SSL certificate verification for Redis connection.

Email server setup

To make outgoing e-mail work, you need to provide a mail server.

Example TLS configuration:

```
environment:
  WEBLATE_EMAIL_HOST: smtp.example.com
  WEBLATE_EMAIL_HOST_USER: user
  WEBLATE_EMAIL_HOST_PASSWORD: pass
```

Example SSL configuration:

```
environment:
  WEBLATE_EMAIL_HOST: smtp.example.com
  WEBLATE_EMAIL_PORT: 465
  WEBLATE_EMAIL_HOST_USER: user
  WEBLATE_EMAIL_HOST_PASSWORD: pass
  WEBLATE_EMAIL_USE_TLS: 0
  WEBLATE_EMAIL_USE_SSL: 1
```

Ayrıca bakınız:

Configuring outgoing e-mail

WEBLATE_EMAIL_HOST

Mail server hostname or IP address.

Ayrıca bakınız:

`WEBLATE_EMAIL_PORT`, `WEBLATE_EMAIL_USE_SSL`, `WEBLATE_EMAIL_USE_TLS`, `EMAIL_HOST`

WEBLATE_EMAIL_PORT

Mail server port, defaults to 25.

Ayrıca bakınız:

`EMAIL_PORT`

WEBLATE_EMAIL_HOST_USER

E-posta kimlik doğrulama kullanıcısı.

Ayrıca bakınız:

`EMAIL_HOST_USER`

WEBLATE_EMAIL_HOST_PASSWORD

E-posta kimlik doğrulama parolası.

Ayrıca bakınız:

`EMAIL_HOST_PASSWORD`

WEBLATE_EMAIL_HOST_PASSWORD_FILE

E-posta kimlik doğrulama parolasını içeren dosyanın yolu.

Ayrıca bakınız:

`WEBLATE_EMAIL_HOST_PASSWORD`

WEBLATE_EMAIL_USE_SSL

Whether to use an implicit TLS (secure) connection when talking to the SMTP server. In most e-mail documentation, this type of TLS connection is referred to as SSL. It is generally used on port 465. If you are experiencing problems, see the explicit TLS setting `WEBLATE_EMAIL_USE_TLS`.

4.11 sürümünde değişti: The SSL/TLS support is automatically enabled based on the `WEBLATE_EMAIL_PORT`.

Ayrıca bakınız:

`WEBLATE_EMAIL_PORT`, `WEBLATE_EMAIL_USE_TLS`, `EMAIL_USE_SSL`

WEBLATE_EMAIL_USE_TLS

Whether to use a TLS (secure) connection when talking to the SMTP server. This is used for explicit TLS connections, generally on port 587 or 25. If you are experiencing connections that hang, see the implicit TLS setting `WEBLATE_EMAIL_USE_SSL`.

4.11 sürümünde değişti: The SSL/TLS support is automatically enabled based on the `WEBLATE_EMAIL_PORT`.

Ayrıca bakınız:

`WEBLATE_EMAIL_PORT`, `WEBLATE_EMAIL_USE_SSL`, `EMAIL_USE_TLS`

WEBLATE_EMAIL_BACKEND

Configures Django back-end to use for sending e-mails.

Ayrıca bakınız:

Configure e-mail sending, `EMAIL_BACKEND`

WEBLATE_AUTO_UPDATE

Configures if and how Weblate should update repositories.

Ayrıca bakınız:

`AUTO_UPDATE`

Not: This is a Boolean setting (use "true" or "false").

Site tümleşimi

WEBLATE_GET_HELP_URL

Configures *GET_HELP_URL*.

WEBLATE_STATUS_URL

Configures *STATUS_URL*.

WEBLATE_LEGAL_URL

Configures *LEGAL_URL*.

WEBLATE_PRIVACY_URL

Configures *PRIVACY_URL*.

Error reporting

It is recommended to collect errors from the installation systematically, see *Collecting error reports*.

To enable support for Rollbar, set the following:

ROLLBAR_KEY

Your Rollbar post server access token.

ROLLBAR_ENVIRONMENT

Your Rollbar environment, defaults to `production`.

To enable support for Sentry, set following:

SENTRY_DSN

Your Sentry DSN.

SENTRY_ENVIRONMENT

Your Sentry Environment (optional).

Yerelleştirme CDN

WEBLATE_LOCALIZE_CDN_URL

WEBLATE_LOCALIZE_CDN_PATH

4.2.1 sürümünde geldi.

Configuration for *JavaScript yerelleştirme CDN*'i.

The *WEBLATE_LOCALIZE_CDN_PATH* is path within the container. It should be stored on the persistent volume and not in the transient storage.

One of possibilities is storing that inside the Weblate data dir:

```
environment:
  WEBLATE_LOCALIZE_CDN_URL: https://cdn.example.com/
  WEBLATE_LOCALIZE_CDN_PATH: /app/data/l10n-cdn
```

Not: You are responsible for setting up serving of the files generated by Weblate, it only does stores the files in configured location.

Ayrıca bakınız:

weblate-cdn, *LOCALIZE_CDN_URL*, *LOCALIZE_CDN_PATH*

Changing enabled apps, checks, add-ons or autofixes

3.8-5 sürümünde geldi.

The built-in configuration of enabled checks, add-ons or autofixes can be adjusted by the following variables:

WEBLATE_ADD_APPS

WEBLATE_REMOVE_APPS

WEBLATE_ADD_CHECK

WEBLATE_REMOVE_CHECK

WEBLATE_ADD_AUTOFIX

WEBLATE_REMOVE_AUTOFIX

WEBLATE_ADD_ADDONS

WEBLATE_REMOVE_ADDONS

Örnek:

```
environment:
  WEBLATE_REMOVE_AUTOFIX: weblate.trans.autofixes.whitespace.
  ↪ SameBookendingWhitespace
  WEBLATE_ADD_ADDONS: customize.addons.MyAddon, customize.addons.OtherAddon
```

Ayrıca bakınız:

CHECK_LIST, AUTOFIX_LIST, WEBLATE_ADDONS, INSTALLED_APPS

Konteyner ayarları

WEBLATE_WORKERS

4.6.1 sürümünde geldi.

Base number of worker processes running in the container. When not set it is determined automatically on container startup based on number of CPU cores available.

It is used to determine *CELERY_MAIN_OPTIONS*, *CELERY_NOTIFY_OPTIONS*, *CELERY_MEMORY_OPTIONS*, *CELERY_TRANSLATE_OPTIONS*, *CELERY_BACKUP_OPTIONS*, *CELERY_BEAT_OPTIONS*, and *WEB_WORKERS*. You can use these settings to fine-tune.

CELERY_MAIN_OPTIONS

CELERY_NOTIFY_OPTIONS

CELERY_MEMORY_OPTIONS

CELERY_TRANSLATE_OPTIONS

CELERY_BACKUP_OPTIONS

CELERY_BEAT_OPTIONS

These variables allow you to adjust Celery worker options. It can be useful to adjust concurrency (`--concurrency 16`) or use different pool implementation (`--pool =gevent`).

By default, the number of concurrent workers is based on *WEBLATE_WORKERS*.

Örnek:

```
environment:
  CELERY_MAIN_OPTIONS: --concurrency 16
```

Ayrıca bakınız:

Celery worker options, *Background tasks using Celery*

WEB_WORKERS

Configure how many uWSGI workers should be executed.

It defaults to `WEBLATE_WORKERS`.

Örnek:

```
environment:
  WEB_WORKERS: 32
```

WEBLATE_SERVICE

Defines which services should be executed inside the container. Use this for *Yatay ölçeklendirme*.

Aşağıdaki hizmetler tanımlıdır:

celery-beat

Celery task scheduler, only one instance should be running. This container is also responsible for the database structure migrations and it should be started prior others.

celery-backup

Celery worker for backups, only one instance should be running.

celery-celery

Generic Celery worker.

celery-memory

Çeviri Belleği Celery işleyicisi.

celery-notify

Bildirimler Celery işleyicisi.

celery-translate

Otomatik çeviri Celery işleyicisi.

web

Web server.

Docker container volumes

There are two volumes (data and cache) exported by the Weblate container. The other service containers (PostgreSQL or Redis) have their data volumes as well, but those are not covered by this document.

The data volume is used to store Weblate persistent data such as cloned repositories or to customize Weblate installation.

The placement of the Docker volume on host system depends on your Docker configuration, but usually it is stored in `/var/lib/docker/volumes/weblate-docker_weblate-data/_data/` (the path consist of name of your docker-compose directory, container, and volume names). In the container it is mounted as `/app/data`.

The cache volume is mounted as `/app/cache` and is used to store static files. Its content is recreated on container startup and the volume can be mounted using ephemeral filesystem such as *tmpfs*.

When creating the volumes manually, the directories should be owned by UID 1000 as that is user used inside the container.

Ayrıca bakınız:

[Docker volumes documentation](#)

Further configuration customization

You can further customize Weblate installation in the data volume, see *Docker container volumes*.

Custom configuration files

You can additionally override the configuration in `/app/data/settings-override.py` (see *Docker container volumes*). This is executed at the end of built-in settings, after all environment settings are loaded, and you can adjust or override them.

Replacing logo and other static files

3.8-5 sürümünde geldi.

The static files coming with Weblate can be overridden by placing into `/app/data/python/customize/static` (see *Docker container volumes*). For example creating `/app/data/python/customize/static/favicon.ico` will replace the favicon.

İpucu: The files are copied to the corresponding location upon container startup, so a restart of Weblate is needed after changing the content of the volume.

This approach can be also used to override Weblate templates. For example *Yasal* documents can be placed into `/app/data/python/customize/templates/legal/documents`.

Alternatively you can also include own module (see *Customizing Weblate*) and add it as separate volume to the Docker container, for example:

```
weblate:
  volumes:
    - weblate-data:/app/data
    - ./weblate_customization/weblate_customization:/app/data/python/weblate_
      ↪ customization
  environment:
    WEBLATE_ADD_APPS: weblate_customization
```

Adding own Python modules

3.8-5 sürümünde geldi.

You can place own Python modules in `/app/data/python/` (see *Docker container volumes*) and they can be then loaded by Weblate, most likely by using *Custom configuration files*.

Ayrıca bakınız:

Customizing Weblate

Configuring PostgreSQL server

The PostgreSQL container uses default PostgreSQL configuration and it won't effectively utilize your CPU cores or memory. It is recommended to customize the configuration to improve the performance.

The configuration can be adjusted as described in *Database Configuration* at https://hub.docker.com/_/postgres. The configuration matching your environment can be generated using <https://pgtune.leopard.in.ua/>.

Installing on Debian and Ubuntu

Hardware requirements

Weblate should run on any contemporary hardware without problems, the following is the minimal configuration required to run Weblate on a single host (Weblate, database and webserver):

- 2 GB of RAM
- 2 CPU cores
- 1 GB of storage space

The more memory the better - it is used for caching on all levels (filesystem, database and Weblate).

Many concurrent users increases the amount of needed CPU cores. For hundreds of translation components at least 4 GB of RAM is recommended.

The typical database storage usage is around 300 MB per 1 million hosted words. Storage space needed for cloned repositories varies, but Weblate tries to keep their size minimal by doing shallow clones.

Not: Actual requirements for your installation of Weblate vary heavily based on the size of the translations managed in it.

Installation

System requirements

Install the dependencies needed to build the Python modules (see *Software requirements*):

```
apt install \
  libxml2-dev libxslt-dev libfreetype6-dev libjpeg-dev libz-dev libyaml-dev \
  libffi-dev libcairo-dev gir1.2-pango-1.0 libgirepository1.0-dev \
  libacl1-dev libssl-dev libpq-dev libjpeg62-turbo-dev build-essential \
  python3-gdbm python3-dev python3-pip python3-virtualenv virtualenv git
```

Install wanted optional dependencies depending on features you intend to use (see *Optional dependencies*):

```
apt install tesseract-ocr libtesseract-dev liblibleptonica-dev
apt install libldap2-dev libldap-common libsasl2-dev
apt install libxmlsec1-dev
```

Optionally install software for running production server, see *Running server*, *Database setup for Weblate*, *Background tasks using Celery*. Depending on size of your installation you might want to run these components on dedicated servers.

The local installation instructions:

```
# Web server option 1: NGINX and uWSGI
apt install nginx uwsgi uwsgi-plugin-python3

# Web server option 2: Apache with ``mod_wsgi``
apt install apache2 libapache2-mod-wsgi-py3

# Caching backend: Redis
apt install redis-server

# Database server: PostgreSQL
apt install postgresql postgresql-contrib

# SMTP server
apt install exim4
```

Python modules

İpucu: We're using virtualenv to install Weblate in a separate environment from your system. If you are not familiar with it, check virtualenv [User Guide](#).

1. Create the virtualenv for Weblate:

```
virtualenv --python=python3 ~/weblate-env
```

2. Activate the virtualenv for Weblate:

```
. ~/weblate-env/bin/activate
```

3. Install Weblate including all optional dependencies:

```
# pkgconfig is needed to install borgbackup 1.2
pip install pkgconfig
# Install Weblate with all optional dependencies
pip install "Weblate[all]"
```

Please check *Optional dependencies* for fine-tuning of optional dependencies.

Not: On some Linux distributions running Weblate fails with libffi error:

```
ffi_prep_closure(): bad user_data (it seems that the version of the libffi_
→library seen at runtime is different from the 'ffi.h' file seen at compile-
→time)
```

This is caused by incompatibility of binary packages distributed via PyPI with the distribution. To address this, you need to rebuild the package on your system:

```
pip install --force-reinstall --no-binary :all: cffi
```

Configuring Weblate

Not: Following steps assume virtualenv used by Weblate is active (what can be done by `. ~/weblate-env/bin/activate`). In case this is not true, you will have to specify full path to **weblate** command as `~/weblate-env/bin/weblate`.

1. Copy the file `~/weblate-env/lib/python3.7/site-packages/weblate/settings_example.py` to `~/weblate-env/lib/python3.7/site-packages/weblate/settings.py`.
2. Adjust the values in the new `settings.py` file to your liking. You will need to provide at least the database credentials and Django secret key, but you will want more changes for production setup, see [Adjusting configuration](#).
3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check [Database setup for Weblate](#) for production ready setup):

```
weblate migrate
```

4. Create the administrator user account and copy the password it outputs to the clipboard, and also save it for later use:

```
weblate createadmin
```

5. Collect static files for web server (see [Running server](#) and [Serving static files](#)):

```
weblate collectstatic
```

6. Compress JavaScript and CSS files (optional, see [Compressing client assets](#)):

```
weblate compress
```

7. Start Celery workers. This is not necessary for development purposes, but strongly recommended otherwise. See [Background tasks using Celery](#) for more info:

```
~/weblate-env/lib/python3.7/site-packages/weblate/examples/celery start
```

8. Start the development server (see [Running server](#) for production setup):

```
weblate runserver
```

After installation

Congratulations, your Weblate server is now running and you can start using it.

- You can now access Weblate on `http://localhost:8000/`.
- Sign in with admin credentials obtained during installation or register with new users.
- You can now run Weblate commands using **weblate** command when Weblate virtualenv is active, see [Management commands](#).
- You can stop the test server with `Ctrl+C`.
- Review potential issues with your installation either on `/manage/performance/` URL (see [Yönetim arayüzü](#)) or using **weblate check --deploy**, see [Production setup](#).

Adding translation

1. Open the admin interface (<http://localhost:8000/create/project/>) and create the project you want to translate. See [Project configuration](#) for more details.

All you need to specify here is the project name and its website.

2. Create a component which is the real object for translation - it points to the VCS repository, and selects which files to translate. See [Component configuration](#) for more details.

The important fields here are: *Bileşen adı*, *Kaynak kod deposu*, and *Dosya maskesi* for finding translatable files. Weblate supports a wide range of formats including *GNU gettext*, *Android string resources*, *Apple iOS strings*, *Java properties*, *Stringsdict biçimi* or *Fluent biçimi*, see [Supported file formats](#) for more details.

3. Once the above is completed (it can be lengthy process depending on the size of your VCS repository, and number of messages to translate), you can start translating.

Installing on SUSE and openSUSE

Hardware requirements

Weblate should run on any contemporary hardware without problems, the following is the minimal configuration required to run Weblate on a single host (Weblate, database and webserver):

- 2 GB of RAM
- 2 CPU cores
- 1 GB of storage space

The more memory the better - it is used for caching on all levels (filesystem, database and Weblate).

Many concurrent users increases the amount of needed CPU cores. For hundreds of translation components at least 4 GB of RAM is recommended.

The typical database storage usage is around 300 MB per 1 million hosted words. Storage space needed for cloned repositories varies, but Weblate tries to keep their size minimal by doing shallow clones.

Not: Actual requirements for your installation of Weblate vary heavily based on the size of the translations managed in it.

Installation

System requirements

Install the dependencies needed to build the Python modules (see [Software requirements](#)):

```
zypper install \
  libxslt-devel libxml2-devel freetype-devel libjpeg-devel zlib-devel \
  libyaml-devel libffi-devel cairo-devel pango-devel \
  gobject-introspection-devel libacl-devel python3-pip python3-virtualenv \
  python3-devel git
```

Install wanted optional dependencies depending on features you intend to use (see [Optional dependencies](#)):

```
zypper install tesseract-ocr tesseract-devel leptonica-devel
zypper install libldap2-devel libsasl2-devel
zypper install libxmlsec1-devel
```

Optionally install software for running production server, see [Running server](#), [Database setup for Weblate](#), [Background tasks using Celery](#). Depending on size of your installation you might want to run these components on dedicated servers.

The local installation instructions:

```
# Web server option 1: NGINX and uWSGI
zypper install nginx uwsgi uwsgi-plugin-python3

# Web server option 2: Apache with ``mod_wsgi``
zypper install apache2 apache2-mod_wsgi

# Caching backend: Redis
zypper install redis-server

# Database server: PostgreSQL
zypper install postgresql postgresql-contrib

# SMTP server
zypper install postfix
```

Python modules

İpucu: We're using virtualenv to install Weblate in a separate environment from your system. If you are not familiar with it, check virtualenv [User Guide](#).

1. Create the virtualenv for Weblate:

```
virtualenv --python=python3 ~/weblate-env
```

2. Activate the virtualenv for Weblate:

```
. ~/weblate-env/bin/activate
```

3. Install Weblate including all optional dependencies:

```
# pkgconfig is needed to install borgbackup 1.2
pip install pkgconfig
# Install Weblate with all optional dependencies
pip install "Weblate[all]"
```

Please check [Optional dependencies](#) for fine-tuning of optional dependencies.

Not: On some Linux distributions running Weblate fails with libffi error:

```
ffi_prep_closure(): bad user_data (it seems that the version of the libffi_
↪library seen at runtime is different from the 'ffi.h' file seen at compile-
↪time)
```

This is caused by incompatibility of binary packages distributed via PyPI with the distribution. To address this, you need to rebuild the package on your system:

```
pip install --force-reinstall --no-binary :all: cffi
```

Configuring Weblate

Not: Following steps assume virtualenv used by Weblate is active (what can be done by `. ~/weblate-env/bin/activate`). In case this is not true, you will have to specify full path to **weblate** command as `~/weblate-env/bin/weblate`.

1. Copy the file `~/weblate-env/lib/python3.7/site-packages/weblate/settings_example.py` to `~/weblate-env/lib/python3.7/site-packages/weblate/settings.py`.
2. Adjust the values in the new `settings.py` file to your liking. You will need to provide at least the database credentials and Django secret key, but you will want more changes for production setup, see [Adjusting configuration](#).
3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check [Database setup for Weblate](#) for production ready setup):

```
weblate migrate
```

4. Create the administrator user account and copy the password it outputs to the clipboard, and also save it for later use:

```
weblate createadmin
```

5. Collect static files for web server (see [Running server](#) and [Serving static files](#)):

```
weblate collectstatic
```

6. Compress JavaScript and CSS files (optional, see [Compressing client assets](#)):

```
weblate compress
```

7. Start Celery workers. This is not necessary for development purposes, but strongly recommended otherwise. See [Background tasks using Celery](#) for more info:

```
~/weblate-env/lib/python3.7/site-packages/weblate/examples/celery start
```

8. Start the development server (see [Running server](#) for production setup):

```
weblate runserver
```

After installation

Congratulations, your Weblate server is now running and you can start using it.

- You can now access Weblate on `http://localhost:8000/`.
- Sign in with admin credentials obtained during installation or register with new users.
- You can now run Weblate commands using **weblate** command when Weblate virtualenv is active, see [Management commands](#).
- You can stop the test server with `Ctrl+C`.
- Review potential issues with your installation either on `/manage/performance/` URL (see [Yönetim arayüzü](#)) or using **weblate check --deploy**, see [Production setup](#).

Adding translation

1. Open the admin interface (<http://localhost:8000/create/project/>) and create the project you want to translate. See [Project configuration](#) for more details.

All you need to specify here is the project name and its website.

2. Create a component which is the real object for translation - it points to the VCS repository, and selects which files to translate. See [Component configuration](#) for more details.

The important fields here are: *Bileşen adı*, *Kaynak kod deposu*, and *Dosya maskesi* for finding translatable files. Weblate supports a wide range of formats including *GNU gettext*, *Android string resources*, *Apple iOS strings*, *Java properties*, *Stringsdict biçimi* or *Fluent biçimi*, see [Supported file formats](#) for more details.

3. Once the above is completed (it can be lengthy process depending on the size of your VCS repository, and number of messages to translate), you can start translating.

Installing on RedHat, Fedora and CentOS

Hardware requirements

Weblate should run on any contemporary hardware without problems, the following is the minimal configuration required to run Weblate on a single host (Weblate, database and webserver):

- 2 GB of RAM
- 2 CPU cores
- 1 GB of storage space

The more memory the better - it is used for caching on all levels (filesystem, database and Weblate).

Many concurrent users increases the amount of needed CPU cores. For hundreds of translation components at least 4 GB of RAM is recommended.

The typical database storage usage is around 300 MB per 1 million hosted words. Storage space needed for cloned repositories varies, but Weblate tries to keep their size minimal by doing shallow clones.

Not: Actual requirements for your installation of Weblate vary heavily based on the size of the translations managed in it.

Installation

System requirements

Install the dependencies needed to build the Python modules (see [Software requirements](#)):

```
dnf install \
    libxslt-devel libxml2-devel freetype-devel libjpeg-devel zlib-devel \
    libyaml-devel libffi-devel cairo-devel pango-devel \
    gobject-introspection-devel libacl-devel python3-pip python3-virtualenv \
    python3-devel git
```

Install wanted optional dependencies depending on features you intend to use (see [Optional dependencies](#)):

```
dnf install tesseract-langpack-eng tesseract-devel leptonica-devel
dnf install libldap2-devel libsasl2-devel
dnf install libxmlsec1-devel
```

Optionally install software for running production server, see [Running server](#), [Database setup for Weblate](#), [Background tasks using Celery](#). Depending on size of your installation you might want to run these components on dedicated servers.

The local installation instructions:

```
# Web server option 1: NGINX and uWSGI
dnf install nginx uwsgi uwsgi-plugin-python3

# Web server option 2: Apache with ``mod_wsgi``
dnf install apache2 apache2-mod_wsgi

# Caching backend: Redis
dnf install redis

# Database server: PostgreSQL
dnf install postgresql postgresql-contrib

# SMTP server
dnf install postfix
```

Python modules

İpucu: We're using virtualenv to install Weblate in a separate environment from your system. If you are not familiar with it, check virtualenv [User Guide](#).

1. Create the virtualenv for Weblate:

```
virtualenv --python=python3 ~/weblate-env
```

2. Activate the virtualenv for Weblate:

```
. ~/weblate-env/bin/activate
```

3. Install Weblate including all optional dependencies:

```
# pkgconfig is needed to install borgbackup 1.2
pip install pkgconfig
# Install Weblate with all optional dependencies
pip install "Weblate[all]"
```

Please check [Optional dependencies](#) for fine-tuning of optional dependencies.

Not: On some Linux distributions running Weblate fails with libffi error:

```
ffi_prep_closure(): bad user_data (it seems that the version of the libffi_
↪library seen at runtime is different from the 'ffi.h' file seen at compile-
↪time)
```

This is caused by incompatibility of binary packages distributed via PyPI with the distribution. To address this, you need to rebuild the package on your system:

```
pip install --force-reinstall --no-binary :all: cffi
```

Configuring Weblate

Not: Following steps assume virtualenv used by Weblate is active (what can be done by `. ~/weblate-env/bin/activate`). In case this is not true, you will have to specify full path to **weblate** command as `~/weblate-env/bin/weblate`.

1. Copy the file `~/weblate-env/lib/python3.7/site-packages/weblate/settings_example.py` to `~/weblate-env/lib/python3.7/site-packages/weblate/settings.py`.
2. Adjust the values in the new `settings.py` file to your liking. You will need to provide at least the database credentials and Django secret key, but you will want more changes for production setup, see [Adjusting configuration](#).
3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check [Database setup for Weblate](#) for production ready setup):

```
weblate migrate
```

4. Create the administrator user account and copy the password it outputs to the clipboard, and also save it for later use:

```
weblate createadmin
```

5. Collect static files for web server (see [Running server](#) and [Serving static files](#)):

```
weblate collectstatic
```

6. Compress JavaScript and CSS files (optional, see [Compressing client assets](#)):

```
weblate compress
```

7. Start Celery workers. This is not necessary for development purposes, but strongly recommended otherwise. See [Background tasks using Celery](#) for more info:

```
~/weblate-env/lib/python3.7/site-packages/weblate/examples/celery start
```

8. Start the development server (see [Running server](#) for production setup):

```
weblate runserver
```

After installation

Congratulations, your Weblate server is now running and you can start using it.

- You can now access Weblate on `http://localhost:8000/`.
- Sign in with admin credentials obtained during installation or register with new users.
- You can now run Weblate commands using **weblate** command when Weblate virtualenv is active, see [Management commands](#).
- You can stop the test server with `Ctrl+C`.
- Review potential issues with your installation either on `/manage/performance/` URL (see [Yönetim arayüzü](#)) or using **weblate check --deploy**, see [Production setup](#).

Adding translation

1. Open the admin interface (<http://localhost:8000/create/project/>) and create the project you want to translate. See [Project configuration](#) for more details.

All you need to specify here is the project name and its website.

2. Create a component which is the real object for translation - it points to the VCS repository, and selects which files to translate. See [Component configuration](#) for more details.

The important fields here are: *Bileşen adı*, *Kaynak kod deposu*, and *Dosya maskesi* for finding translatable files. Weblate supports a wide range of formats including *GNU gettext*, *Android string resources*, *Apple iOS strings*, *Java properties*, *Stringsdict biçimi* or *Fluent biçimi*, see [Supported file formats](#) for more details.

3. Once the above is completed (it can be lengthy process depending on the size of your VCS repository, and number of messages to translate), you can start translating.

Installing on macOS

Hardware requirements

Weblate should run on any contemporary hardware without problems, the following is the minimal configuration required to run Weblate on a single host (Weblate, database and webserver):

- 2 GB of RAM
- 2 CPU cores
- 1 GB of storage space

The more memory the better - it is used for caching on all levels (filesystem, database and Weblate).

Many concurrent users increases the amount of needed CPU cores. For hundreds of translation components at least 4 GB of RAM is recommended.

The typical database storage usage is around 300 MB per 1 million hosted words. Storage space needed for cloned repositories varies, but Weblate tries to keep their size minimal by doing shallow clones.

Not: Actual requirements for your installation of Weblate vary heavily based on the size of the translations managed in it.

Installation

System requirements

Install the dependencies needed to build the Python modules (see [Software requirements](#)):

```
brew install python pango cairo gobject-introspection libffi glib libyaml
pip3 install virtualenv
```

Make sure pip will be able to find the libffi version provided by homebrew — this will be needed during the installation build step.

```
export PKG_CONFIG_PATH="/usr/local/opt/libffi/lib/pkgconfig"
```

Install wanted optional dependencies depending on features you intend to use (see [Optional dependencies](#)):

```
brew install tesseract
```

Optionally install software for running production server, see [Running server](#), [Database setup for Weblate](#), [Background tasks using Celery](#). Depending on size of your installation you might want to run these components on dedicated servers.

The local installation instructions:

```
# Web server option 1: NGINX and uWSGI
brew install nginx uwsgi

# Web server option 2: Apache with `mod_wsgi`
brew install httpd

# Caching backend: Redis
brew install redis

# Database server: PostgreSQL
brew install postgresql
```

Python modules

İpucu: We're using virtualenv to install Weblate in a separate environment from your system. If you are not familiar with it, check virtualenv [User Guide](#).

1. Create the virtualenv for Weblate:

```
virtualenv --python=python3 ~/weblate-env
```

2. Activate the virtualenv for Weblate:

```
. ~/weblate-env/bin/activate
```

3. Install Weblate including all optional dependencies:

```
# pkgconfig is needed to install borgbackup 1.2
pip install pkgconfig
# Install Weblate with all optional dependencies
pip install "Weblate[all]"
```

Please check [Optional dependencies](#) for fine-tuning of optional dependencies.

Not: On some Linux distributions running Weblate fails with libffi error:

```
ffi_prep_closure(): bad user_data (it seems that the version of the libffi_
→library seen at runtime is different from the 'ffi.h' file seen at compile-
→time)
```

This is caused by incompatibility of binary packages distributed via PyPI with the distribution. To address this, you need to rebuild the package on your system:

```
pip install --force-reinstall --no-binary :all: cffi
```

Configuring Weblate

Not: Following steps assume virtualenv used by Weblate is active (what can be done by `. ~/weblate-env/bin/activate`). In case this is not true, you will have to specify full path to **weblate** command as `~/weblate-env/bin/weblate`.

1. Copy the file `~/weblate-env/lib/python3.7/site-packages/weblate/settings_example.py` to `~/weblate-env/lib/python3.7/site-packages/weblate/settings.py`.
2. Adjust the values in the new `settings.py` file to your liking. You will need to provide at least the database credentials and Django secret key, but you will want more changes for production setup, see [Adjusting configuration](#).
3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check [Database setup for Weblate](#) for production ready setup):

```
weblate migrate
```

4. Create the administrator user account and copy the password it outputs to the clipboard, and also save it for later use:

```
weblate createadmin
```

5. Collect static files for web server (see [Running server](#) and [Serving static files](#)):

```
weblate collectstatic
```

6. Compress JavaScript and CSS files (optional, see [Compressing client assets](#)):

```
weblate compress
```

7. Start Celery workers. This is not necessary for development purposes, but strongly recommended otherwise. See [Background tasks using Celery](#) for more info:

```
~/weblate-env/lib/python3.7/site-packages/weblate/examples/celery start
```

8. Start the development server (see [Running server](#) for production setup):

```
weblate runserver
```

After installation

Congratulations, your Weblate server is now running and you can start using it.

- You can now access Weblate on `http://localhost:8000/`.
- Sign in with admin credentials obtained during installation or register with new users.
- You can now run Weblate commands using **weblate** command when Weblate virtualenv is active, see [Management commands](#).
- You can stop the test server with `Ctrl+C`.
- Review potential issues with your installation either on `/manage/performance/` URL (see [Yönetim arayüzü](#)) or using **weblate check --deploy**, see [Production setup](#).

Adding translation

1. Open the admin interface (<http://localhost:8000/create/project/>) and create the project you want to translate. See [Project configuration](#) for more details.

All you need to specify here is the project name and its website.

2. Create a component which is the real object for translation - it points to the VCS repository, and selects which files to translate. See [Component configuration](#) for more details.

The important fields here are: *Bileşen adı*, *Kaynak kod deposu*, and *Dosya maskesi* for finding translatable files. Weblate supports a wide range of formats including *GNU gettext*, *Android string resources*, *Apple iOS strings*, *Java properties*, *Stringsdict biçimi* or *Fluent biçimi*, see [Supported file formats](#) for more details.

3. Once the above is completed (it can be lengthy process depending on the size of your VCS repository, and number of messages to translate), you can start translating.

Installing from sources

1. Please follow the installation instructions for your system first up to installing Weblate:

- [Installing on Debian and Ubuntu](#)
- [Installing on SUSE and openSUSE](#)
- [Installing on RedHat, Fedora and CentOS](#)

2. Grab the latest Weblate sources using Git (or download a tarball and unpack that):

```
git clone https://github.com/WeblateOrg/weblate.git weblate-src
```

Alternatively you can use released archives. You can download them from our website <<https://weblate.org/>>. Those downloads are cryptographically signed, please see [Verifying release signatures](#).

3. Install current Weblate code into the virtualenv:

```
. ~/weblate-env/bin/activate
pip install -e weblate-src
```

4. Copy `weblate/settings_example.py` to `weblate/settings.py`.
5. Adjust the values in the new `settings.py` file to your liking. You will need to provide at least the database credentials and Django secret key, but you will want more changes for production setup, see [Adjusting configuration](#).
6. Create the database used by Weblate, see [Database setup for Weblate](#).
7. Build Django tables, static files and initial data (see [Filling up the database](#) and [Serving static files](#)):

```
weblate migrate
weblate collectstatic
weblate compress
```

Not: This step should be repeated whenever you update the repository.

Installing on OpenShift

With the OpenShift Weblate template you can get your personal Weblate instance up and running in seconds. All of Weblate's dependencies are already included. PostgreSQL is set up as the default database and persistent volume claims are used.

You can find the template at <https://github.com/WeblateOrg/openshift/>.

Installation

The following examples assume you have a working OpenShift v3.x environment, with `oc` client tool installed. Please check the OpenShift documentation for instructions.

The `template.yml` is suited for running all components in OpenShift. There is also `template-external-postgresql.yml` which does not start a PostgreSQL server and allows you to configure external PostgreSQL server.

Web Console

Copy the raw content from [template.yml](#) and import them into your project, then use the `Create` button in the OpenShift web console to create your application. The web console will prompt you for the values for all of the parameters used by the template.

CLI

To upload the Weblate template to your current project's template library, pass the `template.yml` file with the following command:

```
$ oc create -f https://raw.githubusercontent.com/WeblateOrg/openshift/main/
↪template.yml \
  -n <PROJECT>
```

The template is now available for selection using the web console or the CLI.

Parameters

The parameters that you can override are listed in the parameters section of the template. You can list them with the CLI by using the following command and specifying the file to be used:

```
$ oc process --parameters -f https://raw.githubusercontent.com/WeblateOrg/
↪openshift/main/template.yml

# If the template is already uploaded
$ oc process --parameters -n <PROJECT> weblate
```

Tanımlama

You can also use the CLI to process templates and use the configuration that is generated to create objects immediately.

```
$ oc process -f https://raw.githubusercontent.com/WeblateOrg/openshift/main/
↪template.yml \
  -p APPLICATION_NAME=weblate \
  -p WEBLATE_VERSION=4.3.1-1 \
  -p WEBLATE_SITE_DOMAIN=weblate.app-openshift.example.com \
  -p POSTGRESQL_IMAGE=docker-registry.default.svc:5000/openshift/postgresql:9.6 \
  -p REDIS_IMAGE=docker-registry.default.svc:5000/openshift/redis:3.2 \
  | oc create -f
```

The Weblate instance should be available after successful migration and deployment at the specified `WEBLATE_SITE_DOMAIN` parameter.

After container setup, you can sign in as *admin* user with password provided in `WEBLATE_ADMIN_PASSWORD`, or a random password generated on first start if that was not set.

To reset *admin* password, restart the container with `WEBLATE_ADMIN_PASSWORD` set to new password in the respective Secret.

Kaldırma

```
$ oc delete all -l app=<APPLICATION_NAME>
$ oc delete configmap -l app= <APPLICATION_NAME>
$ oc delete secret -l app=<APPLICATION_NAME>
# ATTENTION! The following command is only optional and will permanently delete
↪all of your data.
$ oc delete pvc -l app=<APPLICATION_NAME>

$ oc delete all -l app=weblate \
  && oc delete secret -l app=weblate \
  && oc delete configmap -l app=weblate \
  && oc delete pvc -l app=weblate
```

Yapılandırma

By processing the template a respective ConfigMap will be created and which can be used to customize the Weblate image. The ConfigMap is directly mounted as environment variables and triggers a new deployment every time it is changed. For further configuration options, see *Docker environment variables* for full list of environment variables.

Installing on Kubernetes

Not: This guide is looking for contributors experienced with Kubernetes to cover the setup in more details.

With the Kubernetes Helm chart you can get your personal Weblate instance up and running in seconds. All of Weblate's dependencies are already included. PostgreSQL is set up as the default database and persistent volume claims are used.

You can find the chart at <https://github.com/WeblateOrg/helm/> and it can be displayed at <https://artifacthub.io/packages/helm/weblate/weblate>.

Installation

```
helm repo add weblate https://helm.weblate.org
helm install my-release weblate/weblate
```

Yapılandırma

For further configuration options, see *Docker environment variables* for full list of environment variables.

Depending on your setup and experience, choose an appropriate installation method for you:

- *Installing using Docker*, recommended for production setups.
- Virtualenv installation, recommended for production setups:
 - *Installing on Debian and Ubuntu*
 - *Installing on SUSE and openSUSE*
 - *Installing on RedHat, Fedora and CentOS*
 - *Installing on macOS*
- *Installing from sources*, recommended for development.
- *Installing on OpenShift*
- *Installing on Kubernetes*

2.1.2 Software requirements

Operating system

Weblate is known to work on Linux, FreeBSD and macOS. Other Unix like systems will most likely work too.

Weblate is not supported on Windows. But it may still work and patches are happily accepted.

Other services

Weblate is using other services for its operation. You will need at least following services running:

- PostgreSQL database server, see *Database setup for Weblate*.
- Redis server for cache and tasks queue, see *Background tasks using Celery*.
- SMTP server for outgoing e-mail, see *Configuring outgoing e-mail*.

Python dependencies

Weblate is written in *Python* and supports Python 3.6 or newer. You can install dependencies using pip or from your distribution packages, full list is available in `requirements.txt`.

Most notable dependencies:

Django

<https://www.djangoproject.com/>

Celery

<https://docs.celeryq.dev/>

Translate Toolkit

<https://toolkit.translatehouse.org/>

translation-finder

<https://github.com/WeblateOrg/translation-finder>

Python Social Auth

<https://python-social-auth.readthedocs.io/>

Django REST Framework

<https://www.django-rest-framework.org/>

Optional dependencies

Following modules are necessary for some Weblate features. You can find all of them in `requirements-optional.txt`.

Mercurial (optional for *Mercurial* repositories support)

<https://www.mercurial-scm.org/>

phply (optional for *PHP dizgileri*)

<https://github.com/viraptor/phply>

tesseract (optional for OCR in *Visual context for strings*)

<https://github.com/sirfz/tesseract>

python-akismet (optional for *Spam protection*)

<https://github.com/Nekmo/python-akismet>

ruamel.yaml (optional for *YAML files*)

<https://pypi.org/project/ruamel.yaml/>

Zeep (optional for *Microsoft Terminology*)

<https://docs.python-zeep.org/>

aeidon (optional for *Subtitle files*)

<https://pypi.org/project/aeidon/>

fluent.syntax (optional for *Fluent biçimi*)

<https://projectfluent.org/>

İpucu: When installing using pip, you can directly specify desired features when installing:

```
pip install "Weblate[PHP,Fluent]"
```

Or you can install Weblate with all optional features:

```
pip install "Weblate[all]"
```

Or you can install Weblate without any optional features:

```
pip install Weblate
```

Database backend dependencies

Weblate supports PostgreSQL, MySQL and MariaDB, see *Database setup for Weblate* and backends documentation for more details.

Other system requirements

The following dependencies have to be installed on the system:

Git

<https://git-scm.com/>

Pango, Cairo and related header files and GObject introspection data

<https://cairographics.org/>, <https://pango.gnome.org/>, see *Pango and Cairo*

git-review (optional for Gerrit support)

<https://pypi.org/project/git-review/>

git-svn (optional for Subversion support)

<https://git-scm.com/docs/git-svn>

tesseract and its data (optional for screenshots OCR)

<https://github.com/tesseract-ocr/tesseract>

licensee (optional for detecting license when creating component)

<https://github.com/licensee/licensee>

Build-time dependencies

To build some of the *Python dependencies* you might need to install their dependencies. This depends on how you install them, so please consult individual packages for documentation. You won't need those if using prebuilt *Wheels* while installing using *pip* or when you use distribution packages.

Pango and Cairo

3.7 sürümünde değişti.

Weblate uses Pango and Cairo for rendering bitmap widgets (see promotion) and rendering checks (see *Managing fonts*). To properly install Python bindings for those you need to install system libraries first - you need both Cairo and Pango, which in turn need GLib. All those should be installed with development files and GObject introspection data.

2.1.3 Verifying release signatures

Weblate release are cryptographically signed by the releasing developer. Currently this is Michal Čihař. Fingerprint of his PGP key is:

```
63CB 1DF1 EF12 CF2A C0EE 5A32 9C27 B313 42B7 511D
```

and you can get more identification information from [<https://keybase.io/nijel>](https://keybase.io/nijel).

You should verify that the signature matches the archive you have downloaded. This way you can be sure that you are using the same code that was released. You should also verify the date of the signature to make sure that you downloaded the latest version.

Each archive is accompanied with *.asc* files which contain the PGP signature for it. Once you have both of them in the same folder, you can verify the signature:

```
$ gpg --verify Weblate-3.5.tar.xz.asc
gpg: assuming signed data in 'Weblate-3.5.tar.xz'
gpg: Signature made Ne 3. března 2019, 16:43:15 CET
gpg:                using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: Can't check signature: public key not found
```

As you can see GPG complains that it does not know the public key. At this point you should do one of the following steps:

- Use *wkd* to download the key:

```
$ gpg --auto-key-locate wkd --locate-keys michal@cihar.com
pub  rsa4096 2009-06-17 [SC]
    63CB1DF1EF12CF2AC0EE5A329C27B31342B7511D
uid          [ultimate] Michal Čihař <michal@cihar.com>
uid          [ultimate] Michal Čihař <nijel@debian.org>
uid          [ultimate] [jpeg image of size 8848]
uid          [ultimate] Michal Čihař (Braiiins) <michal.cihar@braiins.cz>
sub  rsa4096 2009-06-17 [E]
sub  rsa4096 2015-09-09 [S]
```

- Download the keyring from [Michal's server](#), then import it with:

```
$ gpg --import wmxth3chu9jfxdxywj1skpmhsj311mzm
```

- Download and import the key from one of the key servers:

```
$ gpg --keyserver hkp://pgp.mit.edu --recv-keys 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: key 9C27B31342B7511D: "Michal Čihař <michal@cihar.com>" imported
gpg: Total number processed: 1
gpg:          unchanged: 1
```

This will improve the situation a bit - at this point you can verify that the signature from the given key is correct but you still can not trust the name used in the key:

```
$ gpg --verify Weblate-3.5.tar.xz.asc
gpg: assuming signed data in 'Weblate-3.5.tar.xz'
gpg: Signature made Ne 3. března 2019, 16:43:15 CET
gpg:          using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: Good signature from "Michal Čihař <michal@cihar.com>" [ultimate]
gpg:          aka "Michal Čihař <nijel@debian.org>" [ultimate]
gpg:          aka "[jpeg image of size 8848]" [ultimate]
gpg:          aka "Michal Čihař (Braiiins) <michal.cihar@braiins.cz>" [ultimate]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 63CB 1DF1 EF12 CF2A C0EE 5A32 9C27 B313 42B7 511D
```

The problem here is that anybody could issue the key with this name. You need to ensure that the key is actually owned by the mentioned person. The GNU Privacy Handbook covers this topic in the chapter [Validating other keys on your public keyring](#). The most reliable method is to meet the developer in person and exchange key fingerprints, however you can also rely on the web of trust. This way you can trust the key transitively through signatures of others, who have met the developer in person.

Once the key is trusted, the warning will not occur:

```
$ gpg --verify Weblate-3.5.tar.xz.asc
gpg: assuming signed data in 'Weblate-3.5.tar.xz'
gpg: Signature made Sun Mar  3 16:43:15 2019 CET
gpg:          using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: Good signature from "Michal Čihař <michal@cihar.com>" [ultimate]
gpg:          aka "Michal Čihař <nijel@debian.org>" [ultimate]
gpg:          aka "[jpeg image of size 8848]" [ultimate]
gpg:          aka "Michal Čihař (Braiiins) <michal.cihar@braiins.cz>" [ultimate]
```

Should the signature be invalid (the archive has been changed), you would get a clear error regardless of the fact that the key is trusted or not:

```
$ gpg --verify Weblate-3.5.tar.xz.asc
gpg: Signature made Sun Mar  3 16:43:15 2019 CET
gpg:                using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: BAD signature from "Michal Čihar <michal@cihar.com>" [ultimate]
```

2.1.4 Filesystem permissions

The Weblate process needs to be able to read and write to the directory where it keeps data - `DATA_DIR`. All files within this directory should be owned and writable by the user running all Weblate processes (typically WSGI and Celery, see *Running server* and *Background tasks using Celery*).

The default configuration places them in the same tree as the Weblate sources, however you might prefer to move these to a better location such as: `/var/lib/weblate`.

Weblate tries to create these directories automatically, but it will fail when it does not have permissions to do so.

You should also take care when running *Management commands*, as they should be ran under the same user as Weblate itself is running, otherwise permissions on some files might be wrong.

In the Docker container, all files in the `/app/data` volume have to be owned by the `weblate` user inside the container (UID 1000).

Ayrıca bakınız:

Serving static files

2.1.5 Database setup for Weblate

It is recommended to run Weblate with a PostgreSQL database server.

Ayrıca bakınız:

Use a powerful database engine, Databases, Migrating from other databases to PostgreSQL

PostgreSQL

PostgreSQL is usually the best choice for Django-based sites. It's the reference database used for implementing Django database layer.

Not: Weblate uses trigram extension which has to be installed separately in some cases. Look for `postgresql-contrib` or a similarly named package.

Ayrıca bakınız:

PostgreSQL notes

Creating a database in PostgreSQL

It is usually a good idea to run Weblate in a separate database, and separate user account:

```
# If PostgreSQL was not installed before, set the main password
sudo -u postgres psql postgres -c "\password postgres"

# Create a database user called "weblate"
sudo -u postgres createuser --superuser --pwprompt weblate

# Create the database "weblate" owned by "weblate"
sudo -u postgres createdb -E UTF8 -O weblate weblate
```

İpucu: If you don't want to make the Weblate user a superuser in PostgreSQL, you can omit that. In that case you will have to perform some of the migration steps manually as a PostgreSQL superuser in schema Weblate will use:

```
CREATE EXTENSION IF NOT EXISTS pg_trgm WITH SCHEMA weblate;
```

Configuring Weblate to use PostgreSQL

The `settings.py` snippet for PostgreSQL:

```
DATABASES = {
    "default": {
        # Database engine
        "ENGINE": "django.db.backends.postgresql",
        # Database name
        "NAME": "weblate",
        # Database user
        "USER": "weblate",
        # Name of role to alter to set parameters in PostgreSQL,
        # use in case role name is different than user used for authentication.
        # "ALTER_ROLE": "weblate",
        # Database password
        "PASSWORD": "password",
        # Set to empty string for localhost
        "HOST": "database.example.com",
        # Set to empty string for default
        "PORT": "",
    }
}
```

The database migration performs `ALTER ROLE` on database role used by Weblate. In most cases the name of the role matches username. In more complex setups the role name is different than username and you will get error about non-existing role during the database migration (`psycopg2.errors.UndefinedObject: role "weblate@hostname" does not exist`). This is known to happen with Azure Database for PostgreSQL, but it's not limited to this environment. Please set `ALTER_ROLE` to change name of the role Weblate should alter during the database migration.

MySQL and MariaDB

İpucu: Some Weblate features will perform better with *PostgreSQL*. This includes searching and translation memory, which both utilize full-text features in the database and PostgreSQL implementation is superior.

Weblate can be also used with MySQL or MariaDB, please see [MySQL notes](#) and [MariaDB notes](#) for caveats using Django with those. Because of the limitations it is recommended to use *PostgreSQL* for new installations.

Weblate requires MySQL at least 5.7.8 or MariaDB at least 10.2.7.

Following configuration is recommended for Weblate:

- Use the `utf8mb4` charset to allow representation of higher Unicode planes (for example emojis).
- Configure the server with `innodb_large_prefix` to allow longer indices on text fields.
- Set the isolation level to `READ COMMITTED`.
- The SQL mode should be set to `STRICT_TRANS_TABLES`.

MySQL 8.x, MariaDB 10.5.x or newer have reasonable default configuration so that no server tweaking should be necessary and all what is needed can be configured on the client side.

Below is an example `/etc/my.cnf.d/server.cnf` for a server with 8 GB of RAM. These settings should be sufficient for most installs. MySQL and MariaDB have tunables that will increase the performance of your server that are considered not necessary unless you are planning on having large numbers of concurrent users accessing the system. See the various vendors documentation on those details.

It is absolutely critical to reduce issues when installing that the setting `innodb_file_per_table` is set properly and MySQL/MariaDB restarted before you start your Weblate install.

```
[mysqld]
character-set-server = utf8mb4
character-set-client = utf8mb4
collation-server = utf8mb4_unicode_ci

datadir=/var/lib/mysql

log-error=/var/log/mariadb/mariadb.log

innodb_large_prefix=1
innodb_file_format=Barracuda
innodb_file_per_table=1
innodb_buffer_pool_size=2G
sql_mode=STRICT_TRANS_TABLES
```

İpucu: In case you are getting #1071 - Specified key was too long; max key length is 767 bytes error, please update your configuration to include the innodb settings above and restart your install.

İpucu: In case you are getting #2006 - MySQL server has gone away error, configuring `CONN_MAX_AGE` might help.

Weblate'i MySQL/MariaDB kullanması için yapılandırma

The `settings.py` snippet for MySQL and MariaDB:

```
DATABASES = {
    "default": {
        # Database engine
        "ENGINE": "django.db.backends.mysql",
        # Database name
        "NAME": "weblate",
        # Database user
        "USER": "weblate",
        # Database password
        "PASSWORD": "password",
        # Set to empty string for localhost
        "HOST": "127.0.0.1",
        # Set to empty string for default
        "PORT": "3306",
        # In case you wish to use additional
        # connection options
        "OPTIONS": {},
    }
}
```

You should also create the `weblate` user account in MySQL or MariaDB before you begin the install. Use the commands below to achieve that:

```
GRANT ALL ON weblate.* to 'weblate'@'localhost' IDENTIFIED BY 'password';
FLUSH PRIVILEGES;
```

2.1.6 Other configurations

Configuring outgoing e-mail

Weblate sends out e-mails on various occasions - for account activation and on various notifications configured by users. For this it needs access to an SMTP server.

The mail server setup is configured using these settings: `EMAIL_HOST`, `EMAIL_HOST_PASSWORD`, `EMAIL_USE_TLS`, `EMAIL_USE_SSL`, `EMAIL_HOST_USER` and `EMAIL_PORT`. Their names are quite self-explanatory, but you can find more info in the Django documentation.

İpucu: In case you get error about not supported authentication (for example SMTP AUTH extension not supported by server), it is most likely caused by using insecure connection and server refuses to authenticate this way. Try enabling `EMAIL_USE_TLS` in such case.

Ayrıca bakınız:

Not receiving e-mails from Weblate, Configuring outgoing e-mail in Docker container

Running behind reverse proxy

Several features in Weblate rely on being able to get client IP address. This includes *Rate limiting*, *Spam protection* or *Denetim günlüğü*.

In default configuration Weblate parses IP address from `REMOTE_ADDR` which is set by the WSGI handler.

In case you are running a reverse proxy, this field will most likely contain its address. You need to configure Weblate to trust additional HTTP headers and parse the IP address from these. This can not be enabled by default as it would allow IP address spoofing for installations not using a reverse proxy. Enabling `IP_BEHIND_REVERSE_PROXY` might be enough for the most usual setups, but you might need to adjust `IP_PROXY_HEADER` and `IP_PROXY_OFFSET` as well.

Another thing to take care of is the `Host` header. It should match to whatever is configured as `SITE_DOMAIN`. Additional configuration might be needed in your reverse proxy (for example use `ProxyPreserveHost On` for Apache or `proxy_set_header Host $host;` with nginx).

Ayrıca bakınız:

Spam protection, Rate limiting, Denetim günlüğü, IP_BEHIND_REVERSE_PROXY, IP_PROXY_HEADER, IP_PROXY_OFFSET, SECURE_PROXY_SSL_HEADER

HTTP proxy

Weblate does execute VCS commands and those accept proxy configuration from environment. The recommended approach is to define proxy settings in `settings.py`:

```
import os

os.environ["http_proxy"] = "http://proxy.example.com:8080"
os.environ["HTTPS_PROXY"] = "http://proxy.example.com:8080"
```

Ayrıca bakınız:

Proxy Environment Variables

2.1.7 Adjusting configuration

Ayrıca bakınız:

Sample configuration

Copy `weblate/settings_example.py` to `weblate/settings.py` and adjust it to match your setup. You will probably want to adjust the following options: `ADMINS`

List of site administrators to receive notifications when something goes wrong, for example notifications on failed merges, or Django errors.

Ayrıca bakınız:

`ADMINS`, *Properly configure admins*

`ALLOWED_HOSTS`

You need to set this to list the hosts your site is supposed to serve. For example:

```
ALLOWED_HOSTS = ["demo.weblate.org"]
```

Alternatively you can include wildcard:

```
ALLOWED_HOSTS = ["*"]
```

Ayrıca bakınız:

`ALLOWED_HOSTS`, `WEBLATE_ALLOWED_HOSTS`, *Allowed hosts setup*

`SESSION_ENGINE`

Configure how your sessions will be stored. In case you keep the default database backend engine, you should schedule: **`weblate clearsessions`** to remove stale session data from the database.

If you are using Redis as cache (see *Enable caching*) it is recommended to use it for sessions as well:

```
SESSION_ENGINE = "django.contrib.sessions.backends.cache"
```

Ayrıca bakınız:

Configuring the session engine, `SESSION_ENGINE`

`DATABASES`

Connectivity to database server, please check Django's documentation for more details.

Ayrıca bakınız:

Database setup for Weblate, `DATABASES`, *Databases*

`DEBUG`

Disable this for any production server. With debug mode enabled, Django will show backtraces in case of error to users, when you disable it, errors will be sent per e-mail to `ADMINS` (see above).

Debug mode also slows down Weblate, as Django stores much more info internally in this case.

Ayrıca bakınız:

`DEBUG`, *Disable debug mode*

`DEFAULT_FROM_EMAIL`

E-mail sender address for outgoing e-mail, for example registration e-mails.

Ayrıca bakınız:

`DEFAULT_FROM_EMAIL`

`SECRET_KEY`

Key used by Django to sign some info in cookies, see [Django secret key](#) for more info.

Ayrıca bakınız:

`SECRET_KEY`

`SERVER_EMAIL`

E-mail used as sender address for sending e-mails to the administrator, for example notifications on failed merges.

Ayrıca bakınız:

`SERVER_EMAIL`

2.1.8 Filling up the database

After your configuration is ready, you can run `weblate migrate` to create the database structure. Now you should be able to create translation projects using the admin interface.

In case you want to run an installation non interactively, you can use `weblate migrate --noinput`, and then create an admin user using `createadmin` command.

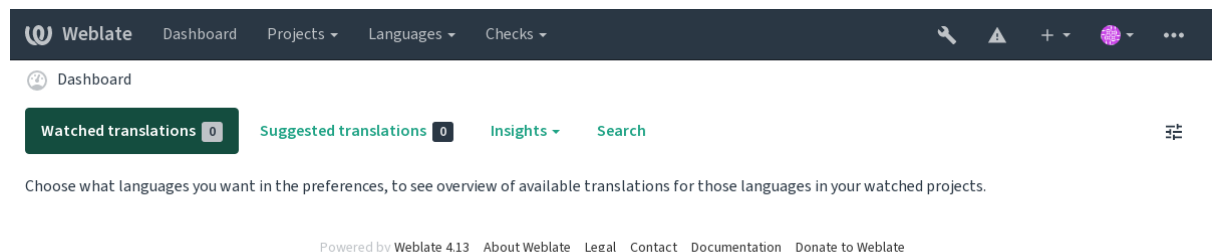
Once you are done, you should also check the *Performance report* in the admin interface, which will give you hints of potential non optimal configuration on your site.

Ayrıca bakınız:

Yapılandırma, Yetkiler ve yerleşik rollerin listesi

2.1.9 Production setup

For a production setup you should carry out adjustments described in the following sections. The most critical settings will trigger a warning, which is indicated by an exclamation mark in the top bar if signed in as a superuser:



It is also recommended to inspect checks triggered by Django (though you might not need to fix all of them):

```
weblate check --deploy
```

You can also review the very same checklist from the *Yönetim arayüzü*.

Ayrıca bakınız:

[Deployment checklist](#)

Disable debug mode

Disable Django’s debug mode (*DEBUG*) by:

```
DEBUG = False
```

With debug mode on, Django stores all executed queries and shows users backtraces of errors, which is not desired in a production setup.

Ayrıca bakınız:

Adjusting configuration

Properly configure admins

Set the correct admin addresses to the *ADMINS* setting to defining who will receive e-mails in case something goes wrong on the server, for example:

```
ADMINS = (("Your Name", "your_email@example.com"),)
```

Ayrıca bakınız:

Adjusting configuration

Set correct site domain

Adjust site name and domain in the admin interface, otherwise links in RSS or registration e-mails will not work. This is configured using *SITE_DOMAIN* which should contain site domain name.

4.2 sürümünde değişti: Prior to the 4.2 release the Django sites framework was used instead, please see [The “sites” framework](#).

Ayrıca bakınız:

Allowed hosts setup, Correctly configure HTTPS SITE_DOMAIN, WEBLATE_SITE_DOMAIN, ENABLE_HTTPS

Correctly configure HTTPS

It is strongly recommended to run Weblate using the encrypted HTTPS protocol. After enabling it, you should set *ENABLE_HTTPS* in the settings:

```
ENABLE_HTTPS = True
```

İpucu: You might want to set up HSTS as well, see [SSL/HTTPS](#) for more details.

Ayrıca bakınız:

ENABLE_HTTPS, Allowed hosts setup, Set correct site domain

Set properly `SECURE_HSTS_SECONDS`

If your site is served over SSL, you have to consider setting a value for `SECURE_HSTS_SECONDS` in the `settings.py` to enable HTTP Strict Transport Security. By default it's set to 0 as shown below.

```
SECURE_HSTS_SECONDS = 0
```

If set to a non-zero integer value, the `django.middleware.security.SecurityMiddleware` sets the HTTP Strict Transport Security header on all responses that do not already have it.

Uyarı: Setting this incorrectly can irreversibly (for some time) break your site. Read the [HTTP Strict Transport Security](#) documentation first.

Use a powerful database engine

- Please use PostgreSQL for a production environment, see [Database setup for Weblate](#) for more info.
- Use adjacent location for running the database server, otherwise the networking performance or reliability might ruin your Weblate experience.
- Check the database server performance or tweak its configuration, for example using [PGTune](#).

Ayrıca bakınız:

[Database setup for Weblate](#), [Migrating from other databases to PostgreSQL](#), [Adjusting configuration](#), [Databases](#)

Enable caching

If possible, use Redis from Django by adjusting the `CACHES` configuration variable, for example:

```
CACHES = {
    "default": {
        "BACKEND": "django_redis.cache.RedisCache",
        "LOCATION": "redis://127.0.0.1:6379/0",
        # If redis is running on same host as Weblate, you might
        # want to use unix sockets instead:
        # 'LOCATION': 'unix:///var/run/redis/redis.sock?db=0',
        "OPTIONS": {
            "CLIENT_CLASS": "django_redis.client.DefaultClient",
            "PARSER_CLASS": "redis.connection.HiredisParser",
        },
    },
}
```

İpucu: In case you change Redis settings for the cache, you might need to adjust them for Celery as well, see [Background tasks using Celery](#).

Ayrıca bakınız:

[Avatar ön bellekleme](#), Django's cache framework

Avatar önbellekleme

In addition to caching of Django, Weblate performs caching of avatars. It is recommended to use a separate, file-backed cache for this purpose:

```
CACHES = {
    "default": {
        # Default caching backend setup, see above
        "BACKEND": "django_redis.cache.RedisCache",
        "LOCATION": "unix:///var/run/redis/redis.sock?db=0",
        "OPTIONS": {
            "CLIENT_CLASS": "django_redis.client.DefaultClient",
            "PARSER_CLASS": "redis.connection.HiredisParser",
        },
    },
    "avatar": {
        "BACKEND": "django.core.cache.backends.filebased.FileBasedCache",
        "LOCATION": os.path.join(DATA_DIR, "avatar-cache"),
        "TIMEOUT": 604800,
        "OPTIONS": {
            "MAX_ENTRIES": 1000,
        },
    },
}
```

Ayrıca bakınız:

[ENABLE_AVATARS](#), [AVATAR_URL_PREFIX](#), [Avatars](#), [Enable caching](#), [Django's cache framework](#)

Configure e-mail sending

Weblate needs to send out e-mails on several occasions, and these e-mails should have a correct sender address, please configure [SERVER_EMAIL](#) and [DEFAULT_FROM_EMAIL](#) to match your environment, for example:

```
SERVER_EMAIL = "admin@example.org"
DEFAULT_FROM_EMAIL = "weblate@example.org"
```

Not: To disable sending e-mails by Weblate set [EMAIL_BACKEND](#) to `django.core.mail.backends.dummy.EmailBackend`.

This will disable *all* e-mail delivery including registration or password reset e-mails.

Ayrıca bakınız:

[Adjusting configuration](#), [Configuring outgoing e-mail](#), [EMAIL_BACKEND](#), [DEFAULT_FROM_EMAIL](#), [SERVER_EMAIL](#)

Allowed hosts setup

Django requires [ALLOWED_HOSTS](#) to hold a list of domain names your site is allowed to serve, leaving it empty will block any requests.

In case this is not configured to match your HTTP server, you will get errors like `Invalid HTTP_HOST header: '1.1.1.1'`. You may need to add `'1.1.1.1'` to [ALLOWED_HOSTS](#).

İpucu: On Docker container, this is available as [WEBLATE_ALLOWED_HOSTS](#).

Ayrıca bakınız:*ALLOWED_HOSTS, WEBLATE_ALLOWED_HOSTS, Set correct site domain***Django secret key**

The `SECRET_KEY` setting is used by Django to sign cookies, and you should really generate your own value rather than using the one from the example setup.

You can generate a new key using `weblate/examples/generate-secret-key` shipped with Weblate.

Ayrıca bakınız:*SECRET_KEY***Ana dizin**

2.1 sürümünde değişti: This is no longer required, Weblate now stores all its data in `DATA_DIR`.

The home directory for the user running Weblate should exist and be writable by this user. This is especially needed if you want to use SSH to access private repositories, but Git might need to access this directory as well (depending on the Git version you use).

You can change the directory used by Weblate in `settings.py`, for example to set it to configuration directory under the Weblate tree:

```
os.environ["HOME"] = os.path.join(BASE_DIR, "configuration")
```

Not: On Linux, and other UNIX like systems, the path to user's home directory is defined in `/etc/passwd`. Many distributions default to a non-writable directory for users used for serving web content (such as `apache`, `www-data` or `wwwrun`), so you either have to run Weblate under a different user, or change this setting.

Ayrıca bakınız:*Accessing repositories***Template loading**

It is recommended to use a cached template loader for Django. It caches parsed templates and avoids the need to do parsing with every single request. You can configure it using the following snippet (the `loaders` setting is important here):

```
TEMPLATES = [
    {
        "BACKEND": "django.template.backends.django.DjangoTemplates",
        "DIRS": [
            os.path.join(BASE_DIR, "templates"),
        ],
        "OPTIONS": {
            "context_processors": [
                "django.contrib.auth.context_processors.auth",
                "django.template.context_processors.debug",
                "django.template.context_processors.i18n",
                "django.template.context_processors.request",
                "django.template.context_processors.csrf",
                "django.contrib.messages.context_processors.messages",
                "weblate.trans.context_processors.weblate_context",
            ],
            "loaders": [
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
(
    "django.template.loaders.cached.Loader",
    [
        "django.template.loaders.filesystem.Loader",
        "django.template.loaders.app_directories.Loader",
    ],
),
],
},
]
```

Ayrıca bakınız:`django.template.loaders.cached.Loader`**Running maintenance tasks**

For optimal performance, it is good idea to run some maintenance tasks in the background. This is now automatically done by *Background tasks using Celery* and covers following tasks:

- Configuration health check (hourly).
- Committing pending changes (hourly), see *Lazy commits* and *commit_pending*.
- Updating component alerts (daily).
- Update remote branches (nightly), see *AUTO_UPDATE*.
- Translation memory backup to JSON (daily), see *dump_memory*.
- Fulltext and database maintenance tasks (daily and weekly tasks), see *cleanuptrans*.

3.2 sürümünde değişti: Since version 3.2, the default way of executing these tasks is using Celery and Weblate already comes with proper configuration, see *Background tasks using Celery*.

System locales and encoding

The system locales should be configured to UTF-8 capable ones. On most Linux distributions this is the default setting. In case it is not the case on your system, please change locales to UTF-8 variant.

For example by editing `/etc/default/locale` and setting there `LANG = "C.UTF-8"`.

In some cases the individual services have separate configuration for locales. This varies between distribution and web servers, so check documentation of your web server packages for that.

Apache on Ubuntu uses `/etc/apache2/envvars`:

```
export LANG='en_US.UTF-8'
export LC_ALL='en_US.UTF-8'
```

Apache on CentOS uses `/etc/sysconfig/httpd` (or `/opt/rh/httpd24/root/etc/sysconfig/httpd`):

```
LANG='en_US.UTF-8'
```

Using custom certificate authority

Weblate does verify SSL certificates during HTTP requests. In case you are using custom certificate authority which is not trusted in default bundles, you will have to add its certificate as trusted.

The preferred approach is to do this at system level, please check your distro documentation for more details (for example on debian this can be done by placing the CA certificate into `/usr/local/share/ca-certificates/` and running `update-ca-certificates`).

Once this is done, system tools will trust the certificate and this includes Git.

For Python code, you will need to configure requests to use system CA bundle instead of the one shipped with it. This can be achieved by placing following snippet to `settings.py` (the path is Debian specific):

```
import os

os.environ["REQUESTS_CA_BUNDLE"] = "/etc/ssl/certs/ca-certificates.crt"
```

Compressing client assets

Weblate comes with a bunch of JavaScript and CSS files. For performance reasons it is good to compress them before sending to a client. In default configuration this is done on the fly at cost of little overhead. On big installations, it is recommended to enable offline compression mode. This needs to be done in the configuration and the compression has to be triggered on every Weblate upgrade.

The configuration switch is simple by enabling `django.conf.settings.COMPRESS_OFFLINE` and configuring `django.conf.settings.COMPRESS_OFFLINE_CONTEXT` (the latter is already included in the example configuration):

```
COMPRESS_OFFLINE = True
```

On each deploy you need to compress the files to match current version:

```
weblate compress
```

İpucu: The official Docker image has this feature already enabled.

Ayrıca bakınız:

Common Deployment Scenarios, *Serving static files*

2.1.10 Running server

İpucu: In case you are not experienced with services described below, you might want to try *Installing using Docker*.

You will need several services to run Weblate, the recommended setup consists of:

- Database server (see *Database setup for Weblate*)
- Cache server (see *Enable caching*)
- Frontend web server for static files and SSL termination (see *Serving static files*)
- WSGI server for dynamic content (see *Sample configuration for NGINX and uWSGI*)
- Celery for executing background tasks (see *Background tasks using Celery*)

Not: There are some dependencies between the services, for example cache and database should be running when starting up Celery or uwsgi processes.

In most cases, you will run all services on single (virtual) server, but in case your installation is heavily loaded, you can split up the services. The only limitation on this is that Celery and Wsgi servers need access to `DATA_DIR`.

Not: The WSGI process has to be executed under the same user the Celery process, otherwise files in the `DATA_DIR` will be stored with mixed ownership, leading to runtime issues.

See also *Filesystem permissions* and *Background tasks using Celery*.

Running web server

Running Weblate is not different from running any other Django based program. Django is usually executed as uWSGI or fcgi (see examples for different web servers below).

For testing purposes, you can use the built-in web server in Django:

```
weblate runserver
```

Uyarı: DO NOT USE THIS SERVER IN A PRODUCTION SETTING. It has not gone through security audits or performance tests. See also Django documentation on `runserver`.

İpucu: The Django built-in server serves static files only with `DEBUG` enabled as it is intended for development only. For production use, please see wsgi setups in *Sample configuration for NGINX and uWSGI*, *Sample configuration for Apache*, *Sample configuration for Apache and Gunicorn*, and *Serving static files*.

Serving static files

2.4 sürümünde değişti: Prior to version 2.4, Weblate didn't properly use the Django static files framework and the setup was more complex.

Django needs to collect its static files in a single directory. To do so, execute `weblate collectstatic --noinput`. This will copy the static files into a directory specified by the `STATIC_ROOT` setting (this defaults to a `static` directory inside `DATA_DIR`).

It is recommended to serve static files directly from your web server, you should use that for the following paths:

/static/

Serves static files for Weblate and the admin interface (from defined by `STATIC_ROOT`).

/media/

Used for user media uploads (e.g. screenshots).

/favicon.ico

Should be rewritten to rewrite a rule to serve `/static/favicon.ico`.

Ayrıca bakınız:

Sample configuration for NGINX and uWSGI, *Sample configuration for Apache*, *Sample configuration for Apache and Gunicorn*, *Compressing client assets*, *How to deploy Django*, *How to deploy static files*

Content security policy

The default Weblate configuration enables `weblate.middleware.SecurityMiddleware` middleware which sets security related HTTP headers like `Content-Security-Policy` or `X-XSS-Protection`. These are by default set up to work with Weblate and its configuration, but this might need customization for your environment.

Ayrıca bakınız:

`CSP_SCRIPT_SRC`, `CSP_IMG_SRC`, `CSP_CONNECT_SRC`, `CSP_STYLE_SRC`, `CSP_FONT_SRC`

Sample configuration for NGINX and uWSGI

To run production webserver, use the wsgi wrapper installed with Weblate (in virtual env case it is installed as `~/weblate-env/lib/python3.7/site-packages/weblate/wsgi.py`). Don't forget to set the Python search path to your virtualenv as well (for example using `virtualenv = /home/user/weblate-env` in uWSGI).

The following configuration runs Weblate as uWSGI under the NGINX webserver.

Configuration for NGINX (also available as `weblate/examples/weblate.nginx.conf`):

```
# This example assumes Weblate is installed in virtualenv in /home/weblate/weblate-
→env
# and DATA_DIR is set to /home/weblate/data, please adjust paths to match your_
→setup.
server {
    listen 80;
    server_name weblate;
    # Not used
    root /var/www/html;

    location ~ ^/favicon.ico$ {
        # DATA_DIR/static/favicon.ico
        alias /home/weblate/data/static/favicon.ico;
        expires 30d;
    }

    location /static/ {
        # DATA_DIR/static/
        alias /home/weblate/data/static/;
        expires 30d;
    }

    location /media/ {
        # DATA_DIR/media/
        alias /home/weblate/data/media/;
        expires 30d;
    }

    location / {
        include uwsgi_params;
        # Needed for long running operations in admin interface
        uwsgi_read_timeout 3600;
        # Adjust based to uwsgi configuration:
        uwsgi_pass unix:///run/uwsgi/app/weblate/socket;
        # uwsgi_pass 127.0.0.1:8080;
    }
}
```

Configuration for uWSGI (also available as `weblate/examples/weblate.uwsgi.ini`):

```
# This example assumes Weblate is installed in virtualenv in /home/weblate/weblate-
↳env
# and DATA_DIR is set to /home/weblate/data, please adjust paths to match your
↳setup.
[uwsgi]
plugins      = python3
master       = true
protocol     = uwsgi
socket       = 127.0.0.1:8080
wsgi-file    = /home/weblate/weblate-env/lib/python3.9/site-packages/weblate/wsgi.
↳py

# Add path to Weblate checkout if you did not install
# Weblate by pip
# python-path = /path/to/weblate

# In case you're using virtualenv uncomment this:
virtualenv = /home/weblate/weblate-env

# Needed for OAuth/OpenID
buffer-size  = 8192

# Reload when consuming too much of memory
reload-on-rss = 250

# Increase number of workers for heavily loaded sites
workers      = 8

# Enable threads for Sentry error submission
enable-threads = true

# Child processes do not need file descriptors
close-on-exec = true

# Avoid default 0000 umask
umask = 0022

# Run as weblate user
uid = weblate
gid = weblate

# Enable harakiri mode (kill requests after some time)
# harakiri = 3600
# harakiri-verbose = true

# Enable uWSGI stats server
# stats = :1717
# stats-http = true

# Do not log some errors caused by client disconnects
ignore-sigpipe = true
ignore-write-errors = true
disable-write-exception = true
```

Ayrıca bakınız:

How to use Django with uWSGI

Sample configuration for Apache

It is recommended to use prefork MPM when using WSGI with Weblate.

The following configuration runs Weblate as WSGI, you need to have enabled `mod_wsgi` (available as `weblate/examples/apache.conf`):

```
#
# VirtualHost for Weblate
#
# This example assumes Weblate is installed in virtualenv in /home/weblate/weblate-
# ↪env
# and DATA_DIR is set to /home/weblate/data, please adjust paths to match your
# ↪setup.
#
<VirtualHost *:80>
    ServerAdmin admin@weblate.example.org
    ServerName weblate.example.org

    # DATA_DIR/static/favicon.ico
    Alias /favicon.ico /home/weblate/data/static/favicon.ico

    # DATA_DIR/static/
    Alias /static/ /home/weblate/data/static/
    <Directory /home/weblate/data/static/>
        Require all granted
    </Directory>

    # DATA_DIR/media/
    Alias /media/ /home/weblate/data/media/
    <Directory /home/weblate/data/media/>
        Require all granted
    </Directory>

    # Path to your Weblate virtualenv
    WSGIDaemonProcess weblate python-home=/home/weblate/weblate-env user=weblate
    ↪request-timeout=600
    WSGIProcessGroup weblate
    WSGIApplicationGroup %{GLOBAL}

    WSGIScriptAlias / /home/weblate/weblate-env/lib/python3.7/site-packages/
    ↪weblate/wsgi.py process-group=weblate
    WSGIPassAuthorization On

    <Directory /home/weblate/weblate-env/lib/python3.7/site-packages/weblate/>
        <Files wsgi.py>
            Require all granted
        </Files>
    </Directory>
</VirtualHost>
```

Not: Weblate requires Python 3, so please make sure you are running Python 3 variant of the `modwsgi`. Usually it is available as a separate package, for example `libapache2-mod-wsgi-py3`.

Ayrıca bakınız:

System locales and encoding, *How to use Django with Apache and mod_wsgi*

Sample configuration for Apache and Gunicorn

The following configuration runs Weblate in Gunicorn and Apache 2.4 (available as `weblate/examples/apache.gunicorn.conf`):

```
#
# VirtualHost for Weblate using gunicorn on localhost:8000
#
# This example assumes Weblate is installed in virtualenv in /home/weblate/weblate-
# ↪env
# and DATA_DIR is set to /home/weblate/data, please adjust paths to match your_
# ↪setup.
#
<VirtualHost *:443>
    ServerAdmin admin@weblate.example.org
    ServerName weblate.example.org

    # DATA_DIR/static/favicon.ico
    Alias /favicon.ico /home/weblate/data/static/favicon.ico

    # DATA_DIR/static/
    Alias /static/ /home/weblate/data/static/
    <Directory /home/weblate/data/static/>
        Require all granted
    </Directory>

    # DATA_DIR/media/
    Alias /media/ /home/weblate/data/media/
    <Directory /home/weblate/data/media/>
        Require all granted
    </Directory>

    SSLEngine on
    SSLCertificateFile /etc/apache2/ssl/https_cert.cert
    SSLCertificateKeyFile /etc/apache2/ssl/https_key.pem
    SSLProxyEngine On

    ProxyPass /favicon.ico !
    ProxyPass /static/ !
    ProxyPass /media/ !

    ProxyPass / http://localhost:8000/
    ProxyPassReverse / http://localhost:8000/
    ProxyPreserveHost On
</VirtualHost>
```

Ayrıca bakınız:

How to use Django with Gunicorn

Running Weblate under path

1.3 sürümünde geldi.

It is recommended to use prefork MPM when using WSGI with Weblate.

A sample Apache configuration to serve Weblate under `/weblate`. Again using `mod_wsgi` (also available as `weblate/examples/apache-path.conf`):

```
#
# VirtualHost for Weblate, running under /weblate path
#
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
# This example assumes Weblate is installed in virtualenv in /home/weblate/weblate-
↪env
# and DATA_DIR is set to /home/weblate/data, please adjust paths to match your
↪setup.
#
<VirtualHost *:80>
    ServerAdmin admin@weblate.example.org
    ServerName weblate.example.org

    # DATA_DIR/static/favicon.ico
    Alias /weblate/favicon.ico /home/weblate/data/static/favicon.ico

    # DATA_DIR/static/
    Alias /weblate/static/ /home/weblate/data/static/
    <Directory /home/weblate/data/static/>
        Require all granted
    </Directory>

    # DATA_DIR/media/
    Alias /weblate/media/ /home/weblate/data/media/
    <Directory /home/weblate/data/media/>
        Require all granted
    </Directory>

    # Path to your Weblate virtualenv
    WSGIDaemonProcess weblate python-home=/home/weblate/weblate-env user=weblate
↪request-timeout=600
    WSGIProcessGroup weblate
    WSGIApplicationGroup %{GLOBAL}

    WSGIScriptAlias /weblate /home/weblate/weblate-env/lib/python3.7/site-packages/
↪weblate/wsgi.py process-group=weblate
    WSGIPassAuthorization On

    <Directory /home/weblate/weblate-env/lib/python3.7/site-packages/weblate/>
        <Files wsgi.py>
            Require all granted
        </Files>
    </Directory>
</VirtualHost>
```

Additionally, you will have to adjust `weblate/settings.py`:

```
URL_PREFIX = "/weblate"
```

2.1.11 Background tasks using Celery

3.2 sürümünde geldi.

Weblate uses Celery to execute regular and background tasks. You are supposed to run a Celery service that will execute these. For example, it is responsible for handling following operations (this list is not complete):

- Receiving webhooks from external services (see *Bildirim kanalcıları*).
- Running regular maintenance tasks such as backups, cleanups, daily add-ons, or updates (see *Backing up and moving Weblate*, *BACKGROUND_TASKS*, *Eklentiler*).
- Running *Otomatik çeviri*.
- Sending digest notifications.

- Offloading expensive operations from the wsgi process.
- Committing pending changes (see *Lazy commits*).

A typical setup using Redis as a backend looks like this:

```
CELERY_TASK_ALWAYS_EAGER = False
CELERY_BROKER_URL = "redis://localhost:6379"
CELERY_RESULT_BACKEND = CELERY_BROKER_URL
```

Ayrıca bakınız:

Redis broker configuration in Celery

You should also start the Celery worker to process the tasks and start scheduled tasks, this can be done directly on the command-line (which is mostly useful when debugging or developing):

```
./weblate/examples/celery start
./weblate/examples/celery stop
```

Not: The Celery process has to be executed under the same user as the WSGI process, otherwise files in the *DATA_DIR* will be stored with mixed ownership, leading to runtime issues.

See also *Filesystem permissions* and *Running server*.

Executing Celery tasks in the wsgi using eager mode

Not: This will have severe performance impact on the web interface, and will break features depending on regular trigger (for example committing pending changes, digest notifications, or backups).

For development, you might want to use eager configuration, which does process all tasks in place:

```
CELERY_TASK_ALWAYS_EAGER = True
CELERY_BROKER_URL = "memory://"
CELERY_TASK_EAGER_PROPAGATES = True
```

Running Celery as system service

Most likely you will want to run Celery as a daemon and that is covered by *Daemonization*. For the most common Linux setup using systemd, you can use the example files shipped in the *examples* folder listed below.

Systemd unit to be placed as */etc/systemd/system/celery-weblate.service*:

```
[Unit]
Description=Celery Service (Weblate)
After=network.target

[Service]
Type=forking
User=weblate
Group=weblate
EnvironmentFile=/etc/default/celery-weblate
WorkingDirectory=/home/weblate
RuntimeDirectory=celery
RuntimeDirectoryPreserve=restart
LogsDirectory=celery
ExecStart=/bin/sh -c '${CELERY_BIN} multi start ${CELERYD_NODES} \'
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
-A ${CELERY_APP} --pidfile=${CELERYD_PID_FILE} \
--logfile=${CELERYD_LOG_FILE} --loglevel =${CELERYD_LOG_LEVEL} ${CELERYD_OPTS}'
ExecStop=/bin/sh -c '${CELERY_BIN} multi stopwait ${CELERYD_NODES} \
--pidfile=${CELERYD_PID_FILE}'
ExecReload=/bin/sh -c '${CELERY_BIN} multi restart ${CELERYD_NODES} \
-A ${CELERY_APP} --pidfile=${CELERYD_PID_FILE} \
--logfile=${CELERYD_LOG_FILE} --loglevel =${CELERYD_LOG_LEVEL} ${CELERYD_OPTS}'
```

[Install]

```
WantedBy=multi-user.target
```

Environment configuration to be placed as `/etc/default/celery-weblate`:

```
# Name of nodes to start
CELERYD_NODES="celery notify memory backup translate"

# Absolute or relative path to the 'celery' command:
CELERY_BIN="/home/weblate/weblate-env/bin/celery"

# App instance to use
# comment out this line if you don't use an app
CELERY_APP="weblate.utils"

# Extra command-line arguments to the worker,
# increase concurrency if you get weblate.E019
CELERYD_OPTS="--beat:celery --queues:celery =celery --prefetch-
↳multiplier:celery =4 \
--queues:notify =notify --prefetch-multiplier:notify =10 \
--queues:memory =memory --prefetch-multiplier:memory =10 \
--queues:translate =translate --prefetch-multiplier:translate =4 \
--concurrency:backup =1 --queues:backup =backup --prefetch-multiplier:backup =2
↳"

# Logging configuration
# - %n will be replaced with the first part of the nodename.
# - %I will be replaced with the current child process index
# and is important when using the prefork pool to avoid race conditions.
CELERYD_PID_FILE="/run/celery/weblate-%n.pid"
CELERYD_LOG_FILE="/var/log/celery/weblate-%n%I.log"
CELERYD_LOG_LEVEL="INFO"
```

Additional configuration to rotate Celery logs using **logrotate** to be placed as `/etc/logrotate.d/celery`:

```
/var/log/celery/*.log {
    weekly
    missingok
    rotate 12
    compress
    notifempty
}
```

Periodic tasks using Celery beat

Weblate comes with built-in setup for scheduled tasks. You can however define additional tasks in `settings.py`, for example see [Lazy commits](#).

The tasks are supposed to be executed by Celery beats daemon. In case it is not working properly, it might not be running or its database was corrupted. Check the Celery startup logs in such case to figure out root cause.

Monitoring Celery status

You can find current length of the Celery task queues in the [Yönetim arayüzü](#) or you can use `celery_queues` on the command-line. In case the queue will get too long, you will also get configuration error in the admin interface.

Uyarı: The Celery errors are by default only logged into Celery log and are not visible to user. In case you want to have overview on such failures, it is recommended to configure [Collecting error reports](#).

Ayrıca bakınız:

[Monitoring Weblate](#), [How can I check whether my Weblate is set up properly?](#), [Configuration and defaults](#), [Workers Guide](#), [Daemonization](#), [Monitoring and Management Guide](#), [celery_queues](#)

2.1.12 Monitoring Weblate

Weblate provides the `/healthz/` URL to be used in simple health checks, for example using Kubernetes. The Docker container has built-in health check using this URL.

For monitoring metrics of Weblate you can use `GET /api/metrics/` API endpoint.

Ayrıca bakınız:

[How can I check whether my Weblate is set up properly?](#), [Monitoring Celery status](#), [Weblate plugin for Munin](#)

2.1.13 Collecting error reports

Weblate, as any other software, can fail. In order to collect useful failure states we recommend to use third party services to collect such information. This is especially useful in case of failing Celery tasks, which would otherwise only report error to the logs and you won't get notified on them. Weblate has support for the following services:

Sentry

Weblate has built-in support for [Sentry](#). To use it, it's enough to set `SENTRY_DSN` in the `settings.py`:

```
SENTRY_DSN = "https://id@your.sentry.example.com/"
```

Rollbar

Weblate has built-in support for [Rollbar](#). To use it, it's enough to follow instructions for [Rollbar notifier for Python](#).

In short, you need to adjust `settings.py`:

```
# Add rollbar as last middleware:
MIDDLEWARE = [
    # ... other middleware classes ...
    "rollbar.contrib.django.middleware.RollbarNotifierMiddleware",
]
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
# Configure client access
ROLLBAR = {
    "access_token": "POST_SERVER_ITEM_ACCESS_TOKEN",
    "client_token": "POST_CLIENT_ITEM_ACCESS_TOKEN",
    "environment": "development" if DEBUG else "production",
    "branch": "main",
    "root": "/absolute/path/to/code/root",
}
```

Everything else is integrated automatically, you will now collect both server and client side errors.

2.1.14 Migrating Weblate to another server

Migrating Weblate to another server should be pretty easy, however it stores data in few locations which you should migrate carefully. The best approach is to stop Weblate for the migration.

Migrating database

Depending on your database backend, you might have several options to migrate the database. The most straightforward one is to dump the database on one server and import it on the new one. Alternatively you can use replication in case your database supports it.

The best approach is to use database native tools, as they are usually the most effective (e.g. **mysqldump** or **pg_dump**). If you want to migrate between different databases, the only option might be to use Django management to dump and import the database:

```
# Export current data
weblate dumpdata > /tmp/weblate.dump
# Import dump
weblate loaddata /tmp/weblate.dump
```

Migrating VCS repositories

The VCS repositories stored under `DATA_DIR` need to be migrated as well. You can simply copy them or use **rsync** to do the migration more effectively.

Other notes

Don't forget to move other services Weblate might have been using like Redis, Cron jobs or custom authentication backends.

2.2 Weblate deployments

Weblate can be easily installed in your cloud. Please find detailed guide for your platform:

- *Installing using Docker*
- *Installing on OpenShift*
- *Installing on Kubernetes*

2.2.1 Third-party deployments for Weblate

Not: Following deployments are not developed or supported by Weblate team. Parts of the setup might vary from what is described in this documentation.

Bitnami Weblate stack

Bitnami provides a Weblate stack for many platforms at <<https://bitnami.com/stack/weblate>>. The setup will be adjusted during installation, see <<https://bitnami.com/stack/weblate/README.txt>> for more documentation.

Weblate Cloudron Paketi

Cloudron is a platform for self-hosting web applications. Weblate installed with Cloudron will be automatically kept up-to-date. The package is maintained by the Cloudron team at their [Weblate package repo](#).



Weblate in YunoHost

The self-hosting project [YunoHost](#) provides a package for Weblate. Once you have your YunoHost installation, you may install Weblate as any other application. It will provide you with a fully working stack with backup and restoration, but you may still have to edit your settings file for specific usages.

You may use your administration interface, or this button (it will bring you to your server):



It also is possible to use the command-line interface:

```
yunohost app install https://github.com/YunoHost-Apps/weblate_ynh
```

2.3 Upgrading Weblate

2.3.1 Docker image upgrades

The official Docker image (see [Installing using Docker](#)) has all Weblate upgrade steps integrated. There are typically no manual steps needed besides pulling latest version.

Ayrıca bakınız:

[Upgrading the Docker container](#)

2.3.2 Generic upgrade instructions

Before upgrading, please check the current *Software requirements* as they might have changed. Once all requirements are installed or updated, please adjust your `settings.py` to match changes in the configuration (consult `settings_example.py` for correct values).

Always check *Version specific instructions* before upgrade. In case you are skipping some versions, please follow instructions for all versions you are skipping in the upgrade. Sometimes it's better to upgrade to some intermediate version to ensure a smooth migration. Upgrading across multiple releases should work, but is not as well tested as single version upgrades.

Not: It is recommended to perform a full database backup prior to upgrade so that you can roll back the database in case upgrade fails, see *Backing up and moving Weblate*.

1. Stop wsgi and Celery processes. The upgrade can perform incompatible changes in the database, so it is always safer to avoid old processes running while upgrading.
2. Upgrade Weblate code.

For pip installs it can be achieved by:

```
pip install -U "Weblate[all]"
```

If you don't want to install all of the optional dependencies do:

```
pip install -U Weblate
```

With Git checkout you need to fetch new source code and update your installation:

```
cd weblate-src
git pull
# Update Weblate inside your virtualenv
. ~/weblate-env/bin/pip install -e .
# Install dependencies directly when not using virtualenv
pip install --upgrade -r requirements.txt
# Install optional dependencies directly when not using virtualenv
pip install --upgrade -r requirements-optional.txt
```

3. New Weblate release might have new *Optional dependencies*, please check if they cover features you want.
4. Upgrade configuration file, refer to `settings_example.py` or *Version specific instructions* for needed steps.
5. Upgrade database structure:

```
weblate migrate --noinput
```

6. Collect updated static files (see *Running server* and *Serving static files*):

```
weblate collectstatic --noinput --clear
```

7. Compress JavaScript and CSS files (optional, see *Compressing client assets*):

```
weblate compress
```

8. If you are running version from Git, you should also regenerate locale files every time you are upgrading. You can do this by invoking:

```
weblate compilemessages
```

9. Verify that your setup is sane (see also *Production setup*):

```
weblate check --deploy
```

10. Restart Celery worker (see *Background tasks using Celery*).

2.3.3 Version specific instructions

Upgrade from 2.x

If you are upgrading from 2.x release, always first upgrade to 3.0.1 and then continue upgrading in the 3.x series. Upgrades skipping this step are not supported and will break.

Ayrıca bakınız:

Upgrade from 2.20 to 3.0 in [Weblate 3.0 documentation](#)

Upgrade from 3.x

If you are upgrading from 3.x release, always first upgrade to 4.0.4 or 4.1.1 and then continue upgrading in the 4.x series. Upgrades skipping this step are not supported and will break.

Ayrıca bakınız:

Upgrade from 3.11 to 4.0 in [Weblate 4.0 documentation](#)

Upgrade from 4.0 to 4.1

Please follow *Generic upgrade instructions* in order to perform update.

Notable configuration or dependencies changes:

- There are several changes in `settings_example.py`, most notable middleware changes, please adjust your settings accordingly.
- There are new file formats, you might want to include them in case you modified the `WEBLATE_FORMATS`.
- There are new quality checks, you might want to include them in case you modified the `CHECK_LIST`.
- There is change in `DEFAULT_THROTTLE_CLASSES` setting to allow reporting of rate limiting in the API.
- There are some new and updated requirements.
- There is a change in `INSTALLED_APPS`.
- The `MT_DEEPL_API_VERSION` setting has been removed in Version 4.7. The *DeepL* machine translation now uses the new `MT_DEEPL_API_URL` instead. You might need to adjust `MT_DEEPL_API_URL` to match your subscription.

Ayrıca bakınız:

[Generic upgrade instructions](#)

Upgrade from 4.1 to 4.2

Please follow *Generic upgrade instructions* in order to perform update.

Notable configuration or dependencies changes:

- Upgrade from 3.x releases is not longer supported, please upgrade to 4.0 or 4.1 first.
- There are some new and updated requirements.
- There are several changes in `settings_example.py`, most notable new middleware and changed application ordering.
- The keys for JSON based formats no longer include leading dot. The strings are adjusted during the database migration, but external components might need adjustment in case you rely on keys in exports or API.
- The Celery configuration was changed to no longer use `memory` queue. Please adjust your startup scripts and `CELERY_TASK_ROUTES` setting.
- The Weblate domain is now configured in the settings, see `SITE_DOMAIN` (or `WEBLATE_SITE_DOMAIN`). You will have to configure it before running Weblate.
- The username and email fields on user database now should be case insensitive unique. It was mistakenly not enforced with PostgreSQL.

Ayrıca bakınız:

Generic upgrade instructions

Upgrade from 4.2 to 4.3

Please follow *Generic upgrade instructions* in order to perform update.

Notable configuration or dependencies changes:

- There are some changes in quality checks, you might want to include them in case you modified the `CHECK_LIST`.
- The source language attribute was moved from project to a component what is exposed in the API. You will need to update *Weblate Client* in case you are using it.
- The database migration to 4.3 might take long depending on number of strings you are translating (expect around one hour of migration time per 100,000 source strings).
- There is a change in `INSTALLED_APPS`.
- There is a new setting `SESSION_COOKIE_AGE_AUTHENTICATED` which complements `SESSION_COOKIE_AGE`.
- In case you were using **hub** or **lab** to integrate with GitHub or GitLab, you will need to reconfigure this, see `GITHUB_CREDENTIALS` and `GITLAB_CREDENTIALS`.

4.3.1 sürümünde değişti:

- The Celery configuration was changed to add `memory` queue. Please adjust your startup scripts and `CELERY_TASK_ROUTES` setting.

4.3.2 sürümünde değişti:

- The `post_update` method of add-ons now takes extra `skip_push` parameter.

Ayrıca bakınız:

Generic upgrade instructions

Upgrade from 4.3 to 4.4

Please follow *Generic upgrade instructions* in order to perform update.

Notable configuration or dependencies changes:

- There is a change in `INSTALLED_APPS`, `weblate.configuration` has to be added there.
- Django 3.1 is now required.
- In case you are using MySQL or MariaDB, the minimal required versions have increased, see *MySQL and MariaDB*.

4.4.1 sürümünde değişti:

- *Monolingual gettext* now uses both `msgid` and `msgctxt` when present. This will change identification of translation strings in such files breaking links to Weblate extended data such as screenshots or review states. Please make sure you commit pending changes in such files prior upgrading and it is recommended to force loading of affected component using *loadpo*.
- Increased minimal required version of `translate-toolkit` to address several file format issues.

Ayrıca bakınız:

Generic upgrade instructions

Upgrade from 4.4 to 4.5

Please follow *Generic upgrade instructions* in order to perform update.

Notable configuration or dependencies changes:

- The migration might take considerable time if you had big glossaries.
- Sözlükler artık normal bileşenler olarak saklanıyor.
- The glossary API is removed, use regular translation API to access glossaries.
- There is a change in `INSTALLED_APPS` - `weblate.metrics` should be added.

4.5.1 sürümünde değişti:

- There is a new dependency on the *pyahocorasick* module.

Ayrıca bakınız:

Generic upgrade instructions

Upgrade from 4.5 to 4.6

Please follow *Generic upgrade instructions* in order to perform update.

Notable configuration or dependencies changes:

- There are new file formats, you might want to include them in case you modified the `WEBLATE_FORMATS`.
- API for creating components now automatically uses *Weblate internal URLs*, see `POST /api/projects/(string:project)/components/`.
- There is a change in dependencies and `PASSWORD_HASHERS` to prefer Argon2 for passwords hashing.

Ayrıca bakınız:

Generic upgrade instructions

Upgrade from 4.6 to 4.7

Please follow *Generic upgrade instructions* in order to perform update.

Notable configuration or dependencies changes:

- There are several changes in `settings_example.py`, most notable middleware changes (`MIDDLEWARE`), please adjust your settings accordingly.
- The *DeepL* machine translation now has a generic `MT_DEEPL_API_URL` setting to adapt to different subscription models more flexibly. The `MT_DEEPL_API_VERSION` setting is no longer used.
- Django 3.2 is now required.

Ayrıca bakınız:

Generic upgrade instructions

Upgrade from 4.7 to 4.8

Please follow *Generic upgrade instructions* in order to perform update.

There are no additional upgrade steps needed in this release.

Ayrıca bakınız:

Generic upgrade instructions

Upgrade from 4.8 to 4.9

Please follow *Generic upgrade instructions* in order to perform update.

- There is a change in storing metrics, the upgrade can take long time on larger sites.

Ayrıca bakınız:

Generic upgrade instructions

Upgrade from 4.9 to 4.10

Please follow *Generic upgrade instructions* in order to perform update.

- There is a change in per-project groups, the upgrade can take long time on sites with thousands of projects.
- Django 4.0 has made some incompatible changes, see [Backwards incompatible changes in 4.0](#). Weblate still supports Django 3.2 for now, in case any of these are problematic. Most notable changes which might affect Weblate:
 - Dropped support for PostgreSQL 9.6, Django 4.0 supports PostgreSQL 10 and higher.
 - Format of `CSRF_TRUSTED_ORIGINS` was changed.
- The Docker container now uses Django 4.0, see above for changes.

Ayrıca bakınız:

Generic upgrade instructions

Upgrade from 4.10 to 4.11

Please follow *Generic upgrade instructions* in order to perform update.

- Weblate now requires Python 3.7 or newer.
- The implementation of *Proje başına erişim denetimini yönetme* has changed, removing the project prefix from the group names. This affects API users.
- Weblate now uses `charset-normalizer` instead of `chardet` module for character set detection.
- **Changed in 4.11.1:** There is a change in `REST_FRAMEWORK` setting (removal of one of the backends in `DEFAULT_AUTHENTICATION_CLASSES`).

Ayrıca bakınız:

Generic upgrade instructions

Upgrade from 4.11 to 4.12

Please follow *Generic upgrade instructions* in order to perform update.

- There are no special steps required.

Ayrıca bakınız:

Generic upgrade instructions

Upgrade from 4.12 to 4.13

Please follow *Generic upgrade instructions* in order to perform update.

- The *Dil tanımları* are now automatically updated on upgrade, use `UPDATE_LANGUAGES` to disable that.
- Handling of context and location has been changed for *Windows RC files*, *HTML files*, *IDML Format*, and *Metin dosyaları* file formats. In most cases the context is now shown as location.
- The machine translation services are now configured using the user interface, settings from the configuration file will be imported during the database migration.

Ayrıca bakınız:

Generic upgrade instructions

2.3.4 Upgrading from Python 2 to Python 3

Weblate no longer supports Python older than 3.6. In case you are still running on older version, please perform migration to Python 3 first on existing version and upgrade later. See [Upgrading from Python 2 to Python 3 in the Weblate 3.11.1 documentation](#).

2.3.5 Migrating from other databases to PostgreSQL

If you are running Weblate on other database than PostgreSQL, you should consider migrating to PostgreSQL as Weblate performs best with it. The following steps will guide you in migrating your data between the databases. Please remember to stop both web and Celery servers prior to the migration, otherwise you might end up with inconsistent data.

Creating a database in PostgreSQL

It is usually a good idea to run Weblate in a separate database, and separate user account:

```
# If PostgreSQL was not installed before, set the main password
sudo -u postgres psql postgres -c "\password postgres"

# Create a database user called "weblate"
sudo -u postgres createuser -D -P weblate

# Create the database "weblate" owned by "weblate"
sudo -u postgres createdb -E UTF8 -O weblate weblate
```

Migrating using Django JSON dumps

The simplest approach for migration is to utilize Django JSON dumps. This works well for smaller installations. On bigger sites you might want to use pgloader instead, see [Migrating to PostgreSQL using pgloader](#).

1. Add PostgreSQL as additional database connection to the `settings.py`:

```
DATABASES = {
    "default": {
        # Database engine
        "ENGINE": "django.db.backends.mysql",
        # Database name
        "NAME": "weblate",
        # Database user
        "USER": "weblate",
        # Database password
        "PASSWORD": "password",
        # Set to empty string for localhost
        "HOST": "database.example.com",
        # Set to empty string for default
        "PORT": "",
        # Additional database options
        "OPTIONS": {
            # In case of using an older MySQL server, which has MyISAM as a
            ↳ default storage
            # 'init_command': 'SET storage_engine =INNODB',
            # Uncomment for MySQL older than 5.7:
            # 'init_command': "SET sql_mode = 'STRICT_TRANS_TABLES'",
            # If your server supports it, see the Unicode issues above
            "charset": "utf8mb4",
            # Change connection timeout in case you get MySQL gone away error:
            "connect_timeout": 28800,
        },
    },
    "postgresql": {
        # Database engine
        "ENGINE": "django.db.backends.postgresql",
        # Database name
        "NAME": "weblate",
        # Database user
        "USER": "weblate",
        # Database password
        "PASSWORD": "password",
        # Set to empty string for localhost
        "HOST": "database.example.com",
        # Set to empty string for default
        "PORT": "",
    },
}
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
} ,  
}
```

2. Run migrations and drop any data inserted into the tables:

```
weblate migrate --database=postgresql  
weblate sqlflush --database=postgresql | weblate dbshell --database=postgresql
```

3. Dump legacy database and import to PostgreSQL

```
weblate dumpdata --all --output weblate.json  
weblate loaddata weblate.json --database=postgresql
```

4. Adjust `DATABASES` to use just PostgreSQL database as default, remove legacy connection.

Weblate should be now ready to run from the PostgreSQL database.

Migrating to PostgreSQL using pgloader

The `pgloader` is a generic migration tool to migrate data to PostgreSQL. You can use it to migrate Weblate database.

1. Adjust your `settings.py` to use PostgreSQL as a database.
2. Migrate the schema in the PostgreSQL database:

```
weblate migrate  
weblate sqlflush | weblate dbshell
```

3. Run the `pgloader` to transfer the data. The following script can be used to migrate the database, but you might want to learn more about `pgloader` to understand what it does and tweak it to match your setup:

```
LOAD DATABASE  
FROM      mysql://weblate:password@localhost/weblate  
INTO      postgresql://weblate:password@localhost/weblate  
  
WITH include no drop, truncate, create no tables, create no indexes, no_  
→foreign keys, disable triggers, reset sequences, data only  
  
ALTER SCHEMA 'weblate' RENAME TO 'public'  
;
```

2.3.6 Migrating from Pootle

As Weblate was originally written as replacement from Pootle, it is supported to migrate user accounts from Pootle. You can dump the users from Pootle and import them using `importusers`.

2.4 Backing up and moving Weblate

2.4.1 Automated backup using BorgBackup

3.9 sürümünde geldi.

Weblate has built-in support for creating service backups using `BorgBackup`. Borg creates space-effective encrypted backups which can be safely stored in the cloud. The backups can be controlled in the management interface from the *Backups* tab.

4.4.1 sürümünde değişti: Both PostgreSQL and MySQL/MariaDB databases are included in the automated backups.

The backups using Borg are incremental and Weblate is configured to keep following backups:

- 14 gün geriye dönük günlük yedekler
- Weekly backups for 8 weeks back
- Monthly backups for 6 months back

Manage / Backups

Backup process triggered

[Weblate status](#)
[Backups](#)
[Translation memory](#)
[Performance report](#)
[SSH keys](#)
[Alerts](#)
[Repositories](#)
[Users](#)
[Appearance](#)

[Tools](#)
[Billing](#)

Backup service: /tmp/tmpmu9sjylwweblate

Backup service credentials	June 15, 2022
Backup repository	/tmp/tmpmu9sjylwweblate
Passphrase	uXk6XHUV01XR0cu^Qt(jyKXgG0f9G@SjYivKtv)76JY4NQ06\$K The passphrase is used to encrypt the backups and is necessary to restore them.
SSH key	Download private key The private key is needed to access the remote backup repository.

Deleted the oldest backups June 15, 2022
 Backup performed June 15, 2022
 Repository initialization June 15, 2022

[Turn off](#)
[Perform backup](#)
[Delete](#)

Activate support package

The support packages include priority e-mail support, or cloud backups of your Weblate installation.

Activation token

Please enter the activation token obtained when making the subscription.

[Activate](#)
[Purchase support package](#)

Add backup service

Backup repository URL

Use /path/to/repo for local backups or user@host:/path/to/repo or ssh://user@host:port/path/to/backups for remote SSH backups.

[Add](#)

Powered by Weblate 4.13 [About Weblate](#) [Legal](#) [Contact](#) [Documentation](#) [Donate to Weblate](#)

Borg encryption key

BorgBackup creates encrypted backups and you wouldn't be able to restore them without the passphrase. The passphrase is generated when adding a new backup service and you should copy it and keep it in a secure place.

If you are using *Weblate provisioned backup storage*, please backup your private SSH key too, as it's used to access your backups.

Ayrıca bakınız:

`borg init`

Customizing backup

- Veri tabanı yedeklemesi `DATABASE_BACKUP` aracılığıyla yapılandırılabilir.
- The backup creation can be customized using `BORG_EXTRA_ARGS`.

2.4.2 Weblate provisioned backup storage

The easiest way of backing up your Weblate instance is purchasing the *backup service* at weblate.org. This is how you get it running:

1. Purchase the *Backup service* on <https://weblate.org/support/#backup>.
2. Enter the obtained key in the management interface, see *Integrating support*.
3. Weblate connects to the cloud service and obtains access info for the backups.
4. Turn on the new backup configuration from the *Backups* tab.
5. Yedekleri geri yükleyebilmek için Borg kimlik bilgilerinizi yedekleyin, bkz: *Borg encryption key*.

İpucu: The manual step of turning everything on is there for your safety. Without your consent no data is sent to the backup repository obtained through the registration process.

2.4.3 Using custom backup storage

You can also use your own storage for the backups. SSH can be used to store backups in the remote destination, the target server needs to have **BorgBackup** installed.

Ayrıca bakınız:

General in the Borg documentation

Local filesystem

It is recommended to specify the absolute path for the local backup, for example `/path/to/backup`. The directory has to be writable by the user running Weblate (see *Filesystem permissions*). If it doesn't exist, Weblate attempts to create it but needs the appropriate permissions to do so.

İpucu: When running Weblate in Docker, please ensure the backup location is exposed as a volume from the Weblate container. Otherwise the backups will be discarded by Docker upon restarting the container it is in.

One option is to place backups into an existing volume, for example `/app/data/borgbackup`. This is an existing volume in the container.

You can also add a new container for the backups in the Docker Compose file for example by using `/borgbackup`:

```
services:
  weblate:
    volumes:
      - /home/weblate/data:/app/data
      - /home/weblate/borgbackup:/borgbackup
```

The directory where backups will be stored have to be owned by UID 1000, otherwise Weblate won't be able to write the backups there.

Uzaktan yedeklemeler

For creating remote backups, you will have to install [BorgBackup](#) onto another server that's accessible for your Weblate deployment via SSH using the Weblate SSH key:

1. Prepare a server where your backups will be stored.
2. Install the SSH server on it (you will get it by default with most Linux distributions).
3. Install [BorgBackup](#) on that server; most Linux distributions have packages available (see [Installation](#)).
4. Choose an existing user or create a new user that will be used for backing up.
5. Add Weblate SSH key to the user so that Weblate can SSH to the server without a password (see [Weblate SSH key](#)).
6. Configure the backup location in Weblate as `user@host:/path/to/backups` or `ssh://user@host:port/path/to/backups`.

İpucu: [Weblate provisioned backup storage](#) provides you automated remote backups without any effort.

Ayrıca bakınız:

[Weblate SSH key](#), [General](#)

2.4.4 Restoring from BorgBackup

1. Restore access to your backup repository and prepare your backup passphrase.
2. List all the backups on the server using `borg list REPOSITORY`.
3. Restore the desired backup to the current directory using `borg extract REPOSITORY::ARCHIVE`.
4. Restore the database from the SQL dump placed in the backup directory in the Weblate data dir (see [Dumped data for backups](#)).
5. Copy the Weblate configuration (`backups/settings.py`, see [Dumped data for backups](#)) to the correct location, see [Adjusting configuration](#).

When using Docker container, the settings file is already included in the container and you should restore the original environment variables. The `environment.yml` file might help you with this (see [Dumped data for backups](#)).

6. Copy the whole restored data dir to the location configured by `DATA_DIR`.

When using Docker container place the data into the data volume, see [Docker container volumes](#).

Please make sure the files have correct ownership and permissions, see [Filesystem permissions](#).

The Borg session might look like this:

```
$ borg list /tmp/xxx
Enter passphrase for key /tmp/xxx:
2019-09-26T14:56:08                               Thu, 2019-09-26 14:56:08_
→[de0e0f13643635d5090e9896bdaceb92a023050749ad3f3350e788f1a65576a5]
$ borg extract /tmp/xxx::2019-09-26T14:56:08
Enter passphrase for key /tmp/xxx:
```

Ayrıca bakınız:

[borg list](#), [borg extract](#)

2.4.5 Manual backup

Depending on what you want to save, back up the type of data Weblate stores in each respective place.

İpucu: If you are doing the manual backups, you might want to silence Weblate’s warning about a lack of backups by adding `weblate.I028` to `SILENCED_SYSTEM_CHECKS` in `settings.py` or `WEBLATE_SILENCED_SYSTEM_CHECKS` for Docker.

```
SILENCED_SYSTEM_CHECKS.append("weblate.I028")
```

Database

The actual storage location depends on your database setup.

İpucu: The database is the most important storage. Set up regular backups of your database. Without the database, all the translations are gone.

Native database backup

The recommended approach is to save a dump of the database using database-native tools such as `pg_dump` or `mysqldump`. It usually performs better than Django backup, and it restores complete tables with all their data.

You can restore this backup in a newer Weblate release, it will perform all the necessary migrations when running in `migrate`. Please consult [Upgrading Weblate](#) on more detailed info on how to upgrade between versions.

Django database backup

Alternatively, you can back up your database using Django’s `dumpdata` command. That way the backup is database agnostic and can be used in case you want to change the database backend.

Prior to restoring the database you need to be running exactly the same Weblate version the backup was made on. This is necessary as the database structure does change between releases and you would end up corrupting the data in some way. After installing the same version, run all database migrations using `migrate`.

Afterwards some entries will already be created in the database and you will have them in the database backup as well. The recommended approach is to delete such entries manually using the management shell (see [Invoking management commands](#)):

```
weblate shell
>>> from weblate.auth.models import User
>>> User.objects.get(username='anonymous').delete()
```

Dosyalar

If you have enough backup space, simply back up the whole `DATA_DIR`. This is a safe bet even if it includes some files you don't want. The following sections describe what you should back up and what you can skip in detail.

Dumped data for backups

4.7 sürümünde değişti: The environment dump was added as `environment.yml` to help in restoring in the Docker environments.

Stored in `DATA_DIR/backups`.

Weblate dumps various data here, and you can include these files for more complete backups. The files are updated daily (requires a running Celery beats server, see [Background tasks using Celery](#)). Currently, this includes:

- Weblate settings as `settings.py` (there is also expanded version in `settings-expanded.py`).
- PostgreSQL database backup as `database.sql`.
- Environment dump as `environment.yml`.

The database backups are saved as plain text by default, but they can also be compressed or entirely skipped using `DATABASE_BACKUP`.

To restore the database backup load it using database tools, for example:

```
psql --file=database.sql weblate
```

Version control repositories

Stored in `DATA_DIR/vcs`.

The version control repositories contain a copy of your upstream repositories with Weblate changes. If you have *İşlemeye yolla* enabled for all your translation components, all Weblate changes are included upstream. No need to back up the repositories on the Weblate side as they can be cloned again from the upstream location(s) with no data loss.

SSH and GPG keys

Stored in `DATA_DIR/ssh` and `DATA_DIR/home`.

If you are using SSH or GPG keys generated by Weblate, you should back up these locations. Otherwise you will lose the private keys and you will have to regenerate new ones.

User uploaded files

Stored in `DATA_DIR/media`.

You should back up all user uploaded files (e.g. *Visual context for strings*).

Celery tasks

The Celery task queue might contain some info, but is usually not needed for a backup. At most you will lose updates not yet been processed to translation memory. It is recommended to perform the fulltext or repository update upon restoration anyhow, so there is no problem in losing these.

Ayrıca bakınız:

Background tasks using Celery

Command-line for manual backup

Using a cron job, you can set up a Bash command to be executed on a daily basis, for example:

```
$ XZ_OPT="-9" tar -Jcf ~/backup/weblate-backup-$(date -u +%Y-%m-%d_%H%M%S).xz_
↪backups vcs ssh home media fonts secret
```

The string between the quotes after `XZ_OPT` allows you to choose your xz options, for instance the amount of memory used for compression; see <https://linux.die.net/man/1/xz>

You can adjust the list of folders and files to your needs. To avoid saving the translation memory (in backups folder), you can use:

```
$ XZ_OPT="-9" tar -Jcf ~/backup/weblate-backup-$(date -u +%Y-%m-%d_%H%M%S).xz_
↪backups/database.sql backups/settings.py vcs ssh home media fonts secret
```

2.4.6 Restoring manual backup

1. Restore all data you have backed up.
2. Update all repositories using *updategit*.

```
weblate updategit --all
```

2.4.7 Moving a Weblate installation

Relocate your installation to a different system by following the backing up and restoration instructions above.

Ayrıca bakınız:

Upgrading from Python 2 to Python 3, Migrating from other databases to PostgreSQL

2.5 Kimlik doğrulama

2.5.1 Kullanıcı kaydı

The default setup for Weblate is to use python-social-auth, a form on the website to handle registration of new users. After confirming their e-mail a new user can contribute or authenticate by using one of the third party services.

You can also turn off registration of new users using *REGISTRATION_OPEN*.

The authentication attempts are subject to *Rate limiting*.

2.5.2 Authentication backends

The built-in solution of Django is used for authentication, including various social options to do so. Using it means you can import the user database of other Django-based projects (see *Migrating from Pootle*).

Django can additionally be set up to authenticate against other means too.

Ayrıca bakınız:

Authentication settings describes how to configure authentication in the official Docker image.

2.5.3 Social authentication

Thanks to [Welcome to Python Social Auth's documentation!](#), Weblate support authentication using many third party services such as GitLab, Ubuntu, Fedora, etc.

Please check their documentation for generic configuration instructions in [Django Framework](#).

Not: By default, Weblate relies on third-party authentication services to provide a validated e-mail address. If some of the services you want to use don't support this, please enforce e-mail validation on the Weblate side by configuring `FORCE_EMAIL_VALIDATION` for them. For example:

```
SOCIAL_AUTH_OPENSUSE_FORCE_EMAIL_VALIDATION = True
```

Ayrıca bakınız:

[Pipeline](#)

Enabling individual backends is quite easy, it's just a matter of adding an entry to the `AUTHENTICATION_BACKENDS` setting and possibly adding keys needed for a given authentication method. Please note that some backends do not provide user e-mail by default, you have to request it explicitly, otherwise Weblate will not be able to properly credit contributions users make.

İpucu: Most of the authentication backends require HTTPS. Once HTTPS is enabled in your web server please configure Weblate to report it properly using `ENABLE_HTTPS`, or by `WEBLATE_ENABLE_HTTPS` in the Docker container.

Ayrıca bakınız:

[Python Social Auth backend](#)

OpenID authentication

For OpenID-based services it's usually just a matter of enabling them. The following section enables OpenID authentication for OpenSUSE, Fedora and Ubuntu:

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.email.EmailAuth",
    "social_core.backends.suse.OpenSUSEOpenId",
    "social_core.backends.ubuntu.UbuntuOpenId",
    "social_core.backends.fedora.FedoraOpenId",
    "weblate.accounts.auth.WeblateUserBackend",
)
```

Ayrıca bakınız:

[OpenID](#)

GitHub authentication

You need to register an OAuth application on GitHub and then tell Weblate all its secrets:

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.github.GithubOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_GITHUB_KEY = "GitHub Client ID"
SOCIAL_AUTH_GITHUB_SECRET = "GitHub Client Secret"
SOCIAL_AUTH_GITHUB_SCOPE = ["user:email"]
```

The GitHub should be configured to have callback URL as `https://example.com/accounts/complete/github/`.

There are similar authentication backends for GitHub for Organizations and GitHub for Teams. Their settings are named `SOCIAL_AUTH_GITHUB_ORG_*` and `SOCIAL_AUTH_GITHUB_TEAM_*`, and they require additional setting of the scope - `SOCIAL_AUTH_GITHUB_ORG_NAME` or `SOCIAL_AUTH_GITHUB_TEAM_ID`. Their callback URLs are `https://example.com/accounts/complete/github-org/` and `https://example.com/accounts/complete/github-teams/`.

Not: Weblate provided callback URL during the authentication includes configured domain. In case you get errors about URL mismatch, you might want to fix this, see [Set correct site domain](#).

Ayrıca bakınız:

GitHub

Bitbucket authentication

You need to register an application on Bitbucket and then tell Weblate all its secrets:

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.bitbucket.BitbucketOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_BITBUCKET_OAUTH2_KEY = "Bitbucket Client ID"
SOCIAL_AUTH_BITBUCKET_OAUTH2_SECRET = "Bitbucket Client Secret"
SOCIAL_AUTH_BITBUCKET_OAUTH2_VERIFIED_EMAILS_ONLY = True
```

Not: Weblate provided callback URL during the authentication includes configured domain. In case you get errors about URL mismatch, you might want to fix this, see [Set correct site domain](#).

Ayrıca bakınız:

Bitbucket

Google OAuth 2

To use Google OAuth 2, you need to register an application on <<https://console.developers.google.com/>> and enable the Google+ API.

The redirect URL is `https://WEBLATE_SERVER/accounts/complete/google-oauth2/`

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.google.GoogleOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_GOOGLE_OAUTH2_KEY = "Client ID"
SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET = "Client secret"
```

Not: Weblate provided callback URL during the authentication includes configured domain. In case you get errors about URL mismatch, you might want to fix this, see *Set correct site domain*.

Ayrıca bakınız:

[Google](#)

Facebook OAuth 2

As per usual with OAuth 2 services, you need to register your application with Facebook. Once this is done, you can set up Weblate to use it:

The redirect URL is `https://WEBLATE_SERVER/accounts/complete/facebook/`

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.facebook.FacebookOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_FACEBOOK_KEY = "key"
SOCIAL_AUTH_FACEBOOK_SECRET = "secret"
SOCIAL_AUTH_FACEBOOK_SCOPE = ["email", "public_profile"]
```

Not: Weblate provided callback URL during the authentication includes configured domain. In case you get errors about URL mismatch, you might want to fix this, see *Set correct site domain*.

Ayrıca bakınız:

[Facebook](#)

GitLab OAuth 2

For using GitLab OAuth 2, you need to register an application on <<https://gitlab.com/profile/applications>>.

The redirect URL is `https://WEBLATE_SERVER/accounts/complete/gitlab/` and ensure you mark the `read_user` scope.

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.gitlab.GitLabOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_GITLAB_KEY = "Application ID"
SOCIAL_AUTH_GITLAB_SECRET = "Secret"
SOCIAL_AUTH_GITLAB_SCOPE = ["read_user"]

# If you are using your own GitLab
# SOCIAL_AUTH_GITLAB_API_URL = 'https://gitlab.example.com/'
```

Not: Weblate provided callback URL during the authentication includes configured domain. In case you get errors about URL mismatch, you might want to fix this, see *Set correct site domain*.

Ayrıca bakınız:

GitLab

Microsoft Azure Active Directory

Weblate can be configured to use common or specific tenants for authentication.

The redirect URL is `https://WEBLATE_SERVER/accounts/complete/azuread-oauth2/` for common and `https://WEBLATE_SERVER/accounts/complete/azuread-tenant-oauth2/` for tenant-specific authentication.

```
# Azure AD common

# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.azuread.AzureADOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# OAuth2 keys
SOCIAL_AUTH_AZUREAD_OAUTH2_KEY = ""
SOCIAL_AUTH_AZUREAD_OAUTH2_SECRET = ""
```

```
# Azure AD Tenant

# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.azuread_tenant.AzureADTenantOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# OAuth2 keys
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_KEY = ""
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_SECRET = ""
# Tenant ID
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_TENANT_ID = ""
```

Not: Weblate provided callback URL during the authentication includes configured domain. In case you get errors about URL mismatch, you might want to fix this, see [Set correct site domain](#).

Ayrıca bakınız:

Microsoft Azure Active Directory

Slack

For using Slack OAuth 2, you need to register an application on <https://api.slack.com/apps>.

The redirect URL is `https://WEBLATE_SERVER/accounts/complete/slack/`.

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.slack.SlackOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_SLACK_KEY = ""
SOCIAL_AUTH_SLACK_SECRET = ""
```

Not: Weblate provided callback URL during the authentication includes configured domain. In case you get errors about URL mismatch, you might want to fix this, see [Set correct site domain](#).

Ayrıca bakınız:

Slack

Overriding authentication method names and icons

You can override the authentication method display name and icon using settings as `SOCIAL_AUTH_<NAME>_IMAGE` and `SOCIAL_AUTH_<NAME>_TITLE`. For example overriding naming for Auth0 would look like:

```
SOCIAL_AUTH_AUTH0_IMAGE = "custom.svg"
SOCIAL_AUTH_AUTH0_TITLE = "Custom auth"
```

Turning off password authentication

E-mail and password authentication can be turned off by removing `social_core.backends.email.EmailAuth` from `AUTHENTICATION_BACKENDS`. Always keep `weblate.accounts.auth.WeblateUserBackend` there, it is needed for core Weblate functionality.

Disabling e-mail authentication will disable all e-mail related functionality – user invitation or password reset feature.

Tüyo: You can still use password authentication for the admin interface, for users you manually create there. Just navigate to `/admin/login/`.

For example authentication using only the openSUSE Open ID provider can be achieved using the following:

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.suse.OpenSUSEOpenId",
    "weblate.accounts.auth.WeblateUserBackend",
)
```

2.5.4 Password authentication

The default `settings.py` comes with a reasonable set of `AUTH_PASSWORD_VALIDATORS`:

- Passwords can't be too similar to your other personal info.
- Passwords must contain at least 10 characters.
- Passwords can't be a commonly used password.
- Passwords can't be entirely numeric.
- Passwords can't consist of a single character or only whitespace.
- Passwords can't match a password you have used in the past.

You can customize this setting to match your password policy.

Additionally you can also install `django-zxcvbn-password` which gives quite realistic estimates of password difficulty and allows rejecting passwords below a certain threshold.

2.5.5 SAML authentication

4.1.1 sürümünde geldi.

Please follow the Python Social Auth instructions for configuration. Notable differences:

- Weblate supports single IDP which has to be called `weblate` in `SOCIAL_AUTH_SAML_ENABLED_IDPS`.
- The SAML XML metadata URL is `/accounts/metadata/saml/`.
- Following settings are automatically filled in: `SOCIAL_AUTH_SAML_SP_ENTITY_ID`, `SOCIAL_AUTH_SAML_TECHNICAL_CONTACT`, `SOCIAL_AUTH_SAML_SUPPORT_CONTACT`

Example configuration:

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.email.EmailAuth",
    "social_core.backends.saml.SAMLAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
# Social auth backends setup
SOCIAL_AUTH_SAML_SP_ENTITY_ID = f"https://{SITE_DOMAIN}/accounts/metadata/saml/"
SOCIAL_AUTH_SAML_SP_PUBLIC_CERT = "-----BEGIN CERTIFICATE-----"
SOCIAL_AUTH_SAML_SP_PRIVATE_KEY = "-----BEGIN PRIVATE KEY-----"
SOCIAL_AUTH_SAML_ENABLED_IDPS = {
    "weblate": {
        "entity_id": "https://idp.testshib.org/idp/shibboleth",
        "url": "https://idp.testshib.org/idp/profile/SAML2/Redirect/SSO",
        "x509cert": "MIIEDjCCAvagAwIBAgIBADA ... 8Bbn1+ev0peYzxFyF5sQA =",
        "attr_name": "full_name",
        "attr_username": "username",
        "attr_email": "email",
    }
}
SOCIAL_AUTH_SAML_ORG_INFO = {
    "en-US": {
        "name": "example",
        "displayname": "Example Inc.",
        "url": "http://example.com"
    }
}
SOCIAL_AUTH_SAML_TECHNICAL_CONTACT = {
    "givenName": "Tech Gal",
    "emailAddress": "technical@example.com"
}
SOCIAL_AUTH_SAML_SUPPORT_CONTACT = {
    "givenName": "Support Guy",
    "emailAddress": "support@example.com"
}
```

The default configuration extracts user details from following attributes, configure your IDP to provide them:

Attribute	SAML URI reference
Ad soyad	urn:oid:2.5.4.3
Ad	urn:oid:2.5.4.42
Last name	urn:oid:2.5.4.4
E-posta	urn:oid:0.9.2342.19200300.100.1.3
Kullanıcı adı	urn:oid:0.9.2342.19200300.100.1.1

İpucu: The example above and the Docker image define an IDP called weblate. You might need to configure this string as *Relay* in your IDP.

Ayrıca bakınız:

Configuring SAML in Docker, SAML

2.5.6 LDAP authentication

LDAP authentication can be best achieved using the *django-auth-ldap* package. You can install it via usual means:

```
# Using PyPI
pip install django-auth-ldap>=1.3.0

# Using apt-get
apt-get install python-django-auth-ldap
```

İpucu: This package is included in the Docker container, see *Installing using Docker*.

Not: There are some incompatibilities in the Python LDAP 3.1.0 module, which might prevent you from using that version. If you get error `AttributeError: 'module' object has no attribute '_trace_level'`, downgrading python-ldap to 3.0.0 might help.

Once you have the package installed, you can hook it into the Django authentication:

```
# Add LDAP backed, keep Django one if you want to be able to sign in
# even without LDAP for admin account
AUTHENTICATION_BACKENDS = (
    "django_auth_ldap.backend.LDAPBackend",
    "weblate.accounts.auth.WeblateUserBackend",
)

# LDAP server address
AUTH_LDAP_SERVER_URI = "ldaps://ldap.example.net"

# DN to use for authentication
AUTH_LDAP_USER_DN_TEMPLATE = "cn =%(user)s,o =Example"
# Depending on your LDAP server, you might use a different DN
# like:
# AUTH_LDAP_USER_DN_TEMPLATE = 'ou =users,dc =example,dc =com'

# List of attributes to import from LDAP upon sign in
# Weblate stores full name of the user in the full_name attribute
AUTH_LDAP_USER_ATTR_MAP = {
    "full_name": "name",
    # Use the following if your LDAP server does not have full name
    # Weblate will merge them later
    # 'first_name': 'givenName',
    # 'last_name': 'sn',
    # Email is required for Weblate (used in VCS commits)
    "email": "mail",
}

# Hide the registration form
REGISTRATION_OPEN = False
```

Not: You should remove `'social_core.backends.email.EmailAuth'` from the `AUTHENTICATION_BACKENDS` setting, otherwise users will be able to set their password in Weblate, and authenticate using that. Keeping `'weblate.accounts.auth.WeblateUserBackend'` is still needed in order to make permissions and facilitate anonymous users. It will also allow you to sign in using a local admin account, if you have created it (e.g. by using *createadmin*).

Using bind password

If you can not use direct bind for authentication, you will need to use search, and provide a user to bind for the search. For example:

```
import ldap
from django_auth_ldap.config import LDAPSearch

AUTH_LDAP_BIND_DN = ""
AUTH_LDAP_BIND_PASSWORD = ""
AUTH_LDAP_USER_SEARCH = LDAPSearch(
    "ou =users,dc =example,dc =com", ldap.SCOPE_SUBTREE, "(uid =%(user)s)"
)
```

Active Directory bütünleşmesi

```
import ldap
from django_auth_ldap.config import LDAPSearch, NestedActiveDirectoryGroupType

AUTH_LDAP_BIND_DN = "CN =ldap,CN =Users,DC =example,DC =com"
AUTH_LDAP_BIND_PASSWORD = "password"

# User and group search objects and types
AUTH_LDAP_USER_SEARCH = LDAPSearch(
    "CN =Users,DC =example,DC =com", ldap.SCOPE_SUBTREE, "(sAMAccountName =%(user)s)"
)

# Make selected group a superuser in Weblate
AUTH_LDAP_USER_FLAGS_BY_GROUP = {
    # is_superuser means user has all permissions
    "is_superuser": "CN =weblate_AdminUsers,OU =Groups,DC =example,DC =com",
}

# Map groups from AD to Weblate
AUTH_LDAP_GROUP_SEARCH = LDAPSearch(
    "OU =Groups,DC =example,DC =com", ldap.SCOPE_SUBTREE, "(objectClass =group)"
)
AUTH_LDAP_GROUP_TYPE = NestedActiveDirectoryGroupType()
AUTH_LDAP_FIND_GROUP_PERMS = True

# Optionally enable group mirroring from LDAP to Weblate
# AUTH_LDAP_MIRROR_GROUPS = True
```

Ayrıca bakınız:

Django Authentication Using LDAP, Authentication

2.5.7 CAS kimlik doğrulaması

CAS authentication can be achieved using a package such as *django-cas-ng*.

Step one is disclosing the e-mail field of the user via CAS. This has to be configured on the CAS server itself, and requires you run at least CAS v2 since CAS v1 doesn't support attributes at all.

Step two is updating Weblate to use your CAS server and attributes.

To install *django-cas-ng*:

```
pip install django-cas-ng
```

Once you have the package installed you can hook it up to the Django authentication system by modifying the `settings.py` file:

```
# Add CAS backed, keep the Django one if you want to be able to sign in
# even without LDAP for the admin account
AUTHENTICATION_BACKENDS = (
    "django_cas_ng.backends.CASBackend",
    "weblate.accounts.auth.WeblateUserBackend",
)

# CAS server address
CAS_SERVER_URL = "https://cas.example.net/cas/"

# Add django_cas_ng somewhere in the list of INSTALLED_APPS
INSTALLED_APPS = (... , "django_cas_ng")
```

Finally, a signal can be used to map the e-mail field to the user object. For this to work you have to import the signal from the *django-cas-ng* package and connect your code with this signal. Doing this in settings file can cause problems, therefore it's suggested to put it:

- In your app config's `django.apps.AppConfig.ready()` method
- In the project's `urls.py` file (when no models exist)

```
from django_cas_ng.signals import cas_user_authenticated
from django.dispatch import receiver

@receiver(cas_user_authenticated)
def update_user_email_address(sender, user=None, attributes=None, **kwargs):
    # If your CAS server does not always include the email attribute
    # you can wrap the next two lines of code in a try/catch block.
    user.email = attributes["email"]
    user.save()
```

Ayrıca bakınız:

Django CAS NG

2.5.8 Configuring third party Django authentication

Generally any Django authentication plugin should work with Weblate. Just follow the instructions for the plugin, just remember to keep the Weblate user backend installed.

Ayrıca bakınız:

LDAP authentication, CAS kimlik doğrulaması

Typically the installation will consist of adding an authentication backend to `AUTHENTICATION_BACKENDS` and installing an authentication app (if there is any) into `INSTALLED_APPS`:

```
AUTHENTICATION_BACKENDS = (
    # Add authentication backend here
    "weblate.accounts.auth.WeblateUserBackend",
)

INSTALLED_APPS += (
    # Install authentication app here
)
```

2.6 Erişim denetimi

Weblate, tüm örnek için veya sınırlı bir kapsamda kullanıcı izinlerini atamak için ayrıntılı bir yetki sistemiyle birlikte gelmektedir.

3.0 sürümünde değişti: Weblate 3.0'dan önce, yetki sistemi yalnızca Django ayrıcalık sistemine dayanıyordu, ancak şimdi özellikle Weblate için oluşturuldu. Daha eski bir şey kullanıyorsanız, lütfen kullandığınız belirli sürüme ait belgelere bakın.

2.6.1 Basit erişim denetimi

Eğer tüm Weblate kurulumunu yönetmiyorsanız ve sadece belirli projelere erişiminiz varsa ([Hosted Weblate](#) gibi), erişim denetim yönetim seçenekleriniz aşağıdaki ayarlarla kısıtlıdır. Eğer herhangi bir karışık kurulumla ihtiyacını yoksa, bu ayarlar sizin için yeterlidir.

Proje erişim denetimi

Not: Bu özellik, Barındırılan Weblate üzerinde Özgür planını kullanan projeler için kullanılabilir değildir.

Farklı bir *Erişim denetimi* ayarı seçerek kullanıcının bireysel projelere erişimini sınırlayabilirsiniz. Kullanılabilir seçenekler şunlardır:

Açık

Herkes görebilir, tüm oturum açmış kullanıcılar çevirebilir.

Korumalı

Herkes görebilir, ancak yalnızca seçili kullanıcılar çevirebilir.

Özel

Yalnızca seçili kullanıcılar görebilir ve çevirebilir.

Özel

Kullanıcı yönetimi özellikleri devre dışı bırakılacak; varsayılan olarak tüm kullanıcıların proje üzerinde herhangi bir işlem yapması yasaktır. Tüm izinleri *Custom access control* kullanarak ayarlamanız gerekecek.

Erişim denetimi, ilgili her projenin yapılandırmasının (*Yönet ↓ Ayarlar*) *Erişim* sekmesinde değiştirilebilir.

The screenshot shows the Weblate interface with the 'Access control' tab selected. The 'Protected' option is chosen, indicating that only chosen users can access the VCS repository. A message at the bottom indicates that the user does not have permission to change the project access control and suggests checking the billing status. A 'Save' button is visible at the bottom left.

Öntanımlı değer `DEFAULT_ACCESS_CONTROL` ile değiştirilebilir.

Not: *Özel* projeler için bile, projenizle ilgili bazı bilgiler açığa çıkacaktır: tüm örnek için istatistikler ve dil özeti, erişim denetimi ayarına rağmen tüm projelerin sayılarını içerecektir. Proje adınız ve diğer bilgileriniz bu şekilde ifşa edilemez.

Not: “Genel”, “Korumalı” ve “Özel” projelerde kullanıcılar için varsayılan olarak mevcut olan gerçek izinler kümesi, Weblate örneği yöneticisi tarafından özel ayarlar kullanılarak yeniden tanımlanabilir.

Uyarı: *Özel* erişim denetimini açarak, Weblate seçilen bir proje için oluşturduğu tüm *özel grupları* kaldıracaktır. Bunu tüm Weblate örneği için yönetici izni olmadan yapıyorsanız, projeyi yönetme erişiminizi anında kaybedersiniz.

Ayrıca bakınız:

[Erişim denetimi](#)

Proje başına erişim denetimini yönetme

Proje erişimini yönet ayrıcalığına sahip kullanıcılar (bkz. ayrıcalıklar), projelerdeki kullanıcıları takımlara ekleyerek yönetebilir. İlk takım grubu Weblate tarafından sağlanır, ancak daha ayrıntılı erişim denetimi sağlayan ek takımlar tanımlanabilir. Ekipleri dillerle sınırlayabilir ve onlara belirli erişim rolleri atayabilirsiniz (bkz. ayrıcalıklar).

Her proje için otomatik olarak aşağıdaki takımlar oluşturulur:

“Genel”, “Korumalı” ve “Özel” projeler için:

Yönetim

Proje için kullanılabilen tüm izinleri içerir.

Gözden geçir (yalnızca inceleme iş akışı açıksa)

Gözden geçirme sırasında çevirileri onaylayabilir.

Yalnızca “Korumalı” ve “Özel” projeler için:

Çevir

Projeyi çevirebilir ve çevrim dışı yapılan çevirileri yükleyebilir.

Kaynaklar

(*project settings* içinde izin verilirse) kaynak dizgileri ve kaynak dizgi bilgilerini düzenleyebilir.

Diller

Çevrilen dilleri yönetebilir (çeviri ekleyebilir veya kaldırabilir).

Sözlük

Sözlük yönetebilir (girişler ekleyebilir veya kaldırabilir, ayrıca yükleyebilir).

Bellek

Çeviri belleğini yönetebilir.

Ekran görüntüleri

Ekran görüntülerini yönetebilir (ekleyip kaldırabilir ve kaynak dizgilerle ilişkilendirebilir).

Otomatik çeviri

Otomatik çeviri kullanabilir.

VCS

VCS'yi yönetebilir ve dışı aktarılan depoya erişebilir.

Faturalandırma

Faturalandırma bilgilerine ve ayarlarına erişebilir (bkz: [Faturalandırma](#)).

Bu özellikler, projenin *Yönet* ↓ *Kullanıcılar* menüsünden erişilebilen *Erişim denetimi* sayfasında bulunmaktadır.

Yeni kullanıcı davetiyesi

Also, besides adding an existing user to the project, it is possible to invite new ones. Any new user will be created immediately, but the account will remain inactive until signing in with a link in the invitation sent via an e-mail. It is not required to have any site-wide privileges in order to do so, access management permission on the project's scope (e.g. a membership in the *Administration* team) would be sufficient.

İpucu: If the invited user missed the validity of the invitation, they can set their password using invited e-mail address in the password reset form as the account is created already.

3.11 sürümünde geldi: It is possible to resend the e-mail for user invitations (invalidating any previously sent invitation).

The same kind of invitations are available site-wide from the *management interface* on the *Users* tab.

Kullanıcıları engelleme

4.7 sürümünde geldi.

In case some users behave badly in your project, you have an option to block them from contributing. The blocked user still will be able to see the project if he has permissions for that, but he won't be able to contribute.

Proje başına izin yönetimi

Projelerinizi *Korumalı* veya *Özel* olarak ayarlayabilir ve Weblate kullanıcı arayüzünde her proje için *manage users*.

By default this prevents Weblate from granting access provided by *Users* and *Viewers* *default groups* due to these groups' own configuration. This doesn't prevent you from granting permissions to those projects site-wide by altering default groups, creating a new one, or creating additional custom settings for individual component as described in *Custom access control* below.

One of the main benefits of managing permissions through the Weblate user interface is that you can delegate it to other users without giving them the superuser privilege. In order to do so, add them to the *Administration* team of the project.

2.6.2 Custom access control

Not: Bu özellik, Barındırılan Weblate üzerinde Özgür planını kullanan projeler için kullanılabilir değildir.

İzin sistemi gruplara ve rollere dayanmaktadır, roller bir dizi izin tanımlamakta ve gruplar bunları kullanıcılara ve çevrilere bağlamaktadır, daha fazla ayrıntı için *Kullanıcılar, roller, gruplar ve izinler* bölümüne bakın.

The most powerful features of the Weblate's access control system for now are available only through the *Django admin interface*. You can use it to manage permissions of any project. You don't necessarily have to switch it to *Custom access control* to utilize it. However you must have superuser privileges in order to use it.

If you are not interested in details of implementation, and just want to create a simple-enough configuration based on the defaults, or don't have a site-wide access to the whole Weblate installation (like on *Hosted Weblate*), please refer to the *Basit erişim denetimi* section.

Yaygın kurulumlar

This section contains an overview of some common configurations you may be interested in.

Site çapında izin yönetimi

To manage permissions for a whole instance at once, add users to appropriate *default groups*:

- *Users* (this is done by default by the *automatic group assignment*).
- *Reviewers* (if you are using *review workflow* with dedicated reviewers).
- *Managers* (if you want to delegate most of the management operations to somebody else).

You should keep all projects configured as *Public* (see *Proje erişim denetimi*), otherwise the site-wide permissions provided by membership in the *Users* and *Reviewers* groups won't have any effect.

You may also grant some additional permissions of your choice to the default groups. For example, you may want to give a permission to manage screenshots to all the *Users*.

You can define some new custom groups as well. If you want to keep managing your permissions site-wide for these groups, choose an appropriate value for the *Project selection* (e.g. *All projects* or *All public projects*).

Diller, bileşenler veya projeler için özel izinler

You can create your own dedicated groups to manage permissions for distinct objects such as languages, components, and projects. Although these groups can only grant additional privileges, you can't revoke any permission granted by site-wide or per-project groups by adding another custom group.

Örnek:

If you want (for whatever reason) to allow translation to a specific language (lets say *Czech*) only to a closed set of reliable translators while keeping translations to other languages public, you will have to:

1. Remove the permission to translate *Czech* from all the users. In the default configuration this can be done by altering the *Users default group*.

Tablo 1: *Kullanıcılar grubu*

Dil seçimi	<i>Tanımlandığı gibi</i>
Diller	All but <i>Czech</i>

2. Add a dedicated group for *Czech* translators.

Tablo 2: *Çekçe çevirmenler grubu*

Roller	<i>Uzman kullanıcılar</i>
Proje seçimi	<i>Tüm genel projeler</i>
Dil seçimi	<i>Tanımlandığı gibi</i>
Diller	<i>Çekçe</i>

3. Add users you wish to give the permissions to into this group.

As you can see, permissions management this way is powerful, but can be quite a tedious job. You can't delegate it to another user, unless granting superuser permissions.

Kullanıcılar, roller, gruplar ve izinler

The authentication models consist of several objects:

Permission

Individual permission defined by Weblate. Permissions cannot be assigned to users. This can only be done through assignment of roles.

Role

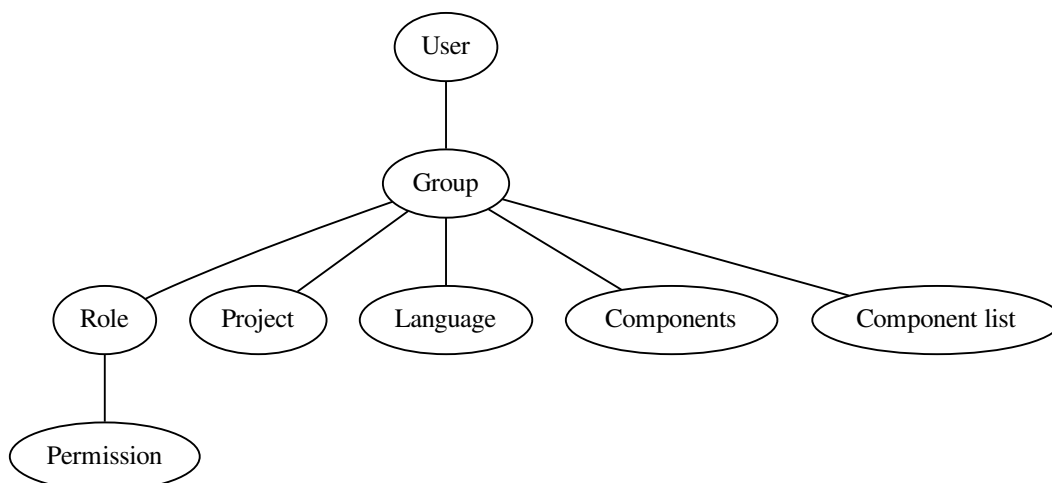
A role defines a set of permissions. This allows reuse of these sets in several places, making the administration easier.

User

User can belong to several groups.

Group

Group connect roles, users, and authentication objects (projects, languages, and component lists).



Not: A group can have no roles assigned to it, in that case access to browse the project by anyone is assumed (see below).

Bir projeye göz atmak için erişim

A user has to be a member of a group linked to the project, or any component inside that project. Having membership is enough, no specific permissions are needed to browse the project (this is used in the default *Viewers* group, see *Grup listesi*).

Bir bileşene göz atmak için erişim

A user can access unrestricted components once able to access the components' project (and will have all the permissions the user was granted for the project). With *Restricted access* turned on, access to the component requires explicit permissions for the component (or a component list the component is in).

Grupların kapsamı

The scope of the permission assigned by the roles in the groups are applied by the following rules:

- If the group specifies any *Component list*, all the permissions given to members of that group are granted for all the components in the component lists attached to the group, and an access with no additional permissions is granted for all the projects these components are in. *Components* and *Projects* are ignored.
- If the group specifies any *Components*, all the permissions given to the members of that group are granted for all the components attached to the group, and an access with no additional permissions is granted for all the projects these components are in. *Projects* are ignored.
- Otherwise, if the group specifies any *Projects*, either by directly listing them or by having *Projects selection* set to a value like *All public projects*, all those permissions are applied to all the projects, which effectively grants the same permissions to access all the projects *unrestricted components*.
- The restrictions imposed by a group's *Languages* are applied separately, when it's verified if a user has an access to perform certain actions. Namely, it's applied only to actions directly related to the translation process itself like reviewing, saving translations, adding suggestions, etc.

İpucu: Use *Language selection* or *Project selection* to automate inclusion of all languages or projects.

Örnek:

Let's say there is a project `foo` with the components: `foo/bar` and `foo/baz` and the following group:

Tablo 3: Group *Spanish Admin-Reviewers*

Roller	<i>Dizgileri gözden geçir, Depoyu yönet</i>
Bileşenler	<code>foo/bar</code>
Diller	<i>İspanyolca</i>

Members of that group will have following permissions (assuming the default role settings):

- General (browsing) access to the whole project `foo` including both components in it: `foo/bar` and `foo/baz`.
- Review strings in `foo/bar` Spanish translation (not elsewhere).
- Manage VCS for the whole `foo/bar` repository e.g. commit pending changes made by translators for all languages.

Kendiliğinden grup atamaları

On the bottom of the *Group* editing page in the *Django admin interface*, you can specify *Automatic group assignments*, which is a list of regular expressions used to automatically assign newly created users to a group based on their e-mail addresses. This assignment only happens upon account creation.

The most common use-case for the feature is to assign all new users to some default group. In order to do so, you will probably want to keep the default value (`^.*$`) in the regular expression field. Another use-case for this option might be to give some additional privileges to employees of your company by default. Assuming all of them use corporate e-mail addresses on your domain, this can be accomplished with an expression like `^.*@mycompany.com`.

Not: Automatic group assignment to *Users* and *Viewers* is always recreated when upgrading from one Weblate version to another. If you want to turn it off, set the regular expression to `^$` (which won't match anything).

Not: As for now, there is no way to bulk-add already existing users to some group via the user interface. For that, you may resort to using the *REST API*.

Varsayılan gruplar ve roller

After installation, a default set of groups is created (see *Grup listesi*).

These roles and groups are created upon installation. The built-in roles are always kept up to date by the database migration when upgrading. You can't actually change them, please define a new role if you want to define your own set of permissions.

Yetkiler ve yerleşik rollerin listesi

Kapsam	İzin	Roller
Faturalandırma (bkz: <i>Faturalandırma</i>)	Faturalandırma bilgisini görüntüle	<i>Yönetim, Faturalandırma</i>
Değişiklikler	Değişiklikleri indir	<i>Yönetim</i>
Yorumlar	Yorumu gönder	<i>Yönetim, Kaynağı düzenle, Uzman kullanıcı</i>
	Yorumu sil	<i>Yönetim</i>
	Resolve comment	<i>Yönetim, Dizgileri gözden geçir</i>
Bileşen	Bileşen ayarlarını düzenle	<i>Yönetim</i>
	Bileşeni kilitle, çevirileri engelle	<i>Yönetim</i>
Sözlük	Sözlük girişi ekle	<i>Yönetim, Sözlüğü yönet, Uzman kullanıcı</i>
	Sözlük girişini düzenle	<i>Yönetim, Sözlüğü yönet, Uzman kullanıcı</i>
	Sözlük girişini sil	<i>Yönetim, Sözlüğü yönet, Uzman kullanıcı</i>
	Sözlük girişlerini yükle	<i>Yönetim, Sözlüğü yönet, Uzman kullanıcı</i>
Otomatik öneriler	Otomatik önerileri kullan	<i>Yönetim, Kaynağı düzenle, Uzman kullanıcı</i>
Çeviri Belleği	Çeviri belleğini düzenle	<i>Yönetim, Çeviri belleğini yönet</i>
	Çeviri belleğini sil	<i>Yönetim, Çeviri belleğini yönet</i>
Projeler	Proje ayarlarını düzenle	<i>Yönetim</i>
	Proje erişimini yönet	<i>Yönetim</i>
Raporlar	Raporları indir	<i>Yönetim</i>
Ekran görüntüleri	Ekran görüntüsü ekle	<i>Yönetim, Ekran görüntülerini yönet</i>
	Ekran görüntüsünü düzenle	<i>Yönetim, Ekran görüntülerini yönet</i>
	Ekran görüntüsünü sil	<i>Yönetim, Ekran görüntülerini yönet</i>
Kaynak dizgiler	Ek dizgi bilgisini düzenle	<i>Yönetim, Kaynağı düzenle</i>
Dizgiler	Yeni dizgi ekle	<i>Yönetim</i>
	Bir dizgiyi kaldır	<i>Yönetim</i>
	Kusur denetimini yoksay	<i>Yönetim, Kaynağı düzenle, Uzman kullanıcı</i>
	Dizgileri düzenle	<i>Yönetim, Kaynağı düzenle, Uzman kullanıcı</i>
	Dizgileri gözden geçir	<i>Yönetim, Dizgileri gözden geçir</i>
	Öneriler zorunlu kıldığında dizgiyi düzenle	<i>Yönetim, Dizgileri gözden geçir</i>
	Kaynak dizgileri düzenle	<i>Yönetim, Kaynağı düzenle, Uzman kullanıcı</i>
Öneriler	Öneriyi kabul et	<i>Yönetim, Kaynağı düzenle, Uzman kullanıcı</i>
	Öneri ekle	<i>Yönetim, Kaynağı düzenle, Öneri ekle, Uzman kullanıcı</i>
	Öneriyi sil	<i>Yönetim, Uzman kullanıcı</i>
	Öneriye oy ver	<i>Yönetim, Kaynağı düzenle, Uzman kullanıcı</i>
Çeviriler	Çeviri için dil ekle	<i>Yönetim, Uzman kullanıcı, Dilleri yönet</i>
	Kendiliğinden çeviri gerçekleştir	<i>Yönetim, Otomatik çeviri</i>
	Varolan çeviriyi sil	<i>Yönetim, Dilleri yönet</i>
	Çeviri dosyasını indir	<i>Yönetim, Kaynağı düzenle, Depoya erişim, Uzman kullanıcı</i>
	Çeviri için birkaç dil ekleyin	<i>Yönetim, Dilleri yönet</i>
Yüklemeler	Yüklenmiş çevirinin hazırlayanını tanımlayın	<i>Yönetim</i>
	Yükleme ile varolan dizgilerin üzerine yaz	<i>Yönetim, Kaynağı düzenle, Uzman kullanıcı</i>
	Çevirileri yükle	<i>Yönetim, Kaynağı düzenle, Uzman kullanıcı</i>
VCS	İç depoya eriş	<i>Yönetim, Depoya erişim, Uzman kullanıcı, Dilleri yönet</i>
	Değişiklikleri iç depoya işle	<i>Yönetim, Depoyu yönet</i>
	İç depodan değişikliği yolla	<i>Yönetim, Depoyu yönet</i>
	İç depodaki değişiklikleri sıfırla	<i>Yönetim, Depoyu yönet</i>
	Yukarı akış depo konumunu görüntüle	<i>Yönetim, Depoya erişim, Uzman kullanıcı, Dilleri yönet</i>
	İç depoyu güncelle	<i>Yönetim, Depoyu yönet</i>
Site çapında yetkiler	Yönetim arayüzünü kullan	
	Yeni projeler ekle	
	Dil tanımları ekle	
	Dil tanımlarını yönet	
	Grupları yönet	
	Kullanıcıları yönet	
	Rolleri yönet	

Tablo 4 – önceki sayfadan devam

Kapsam	İzin	Roller
	Duyuruları yönet	
	Çeviri belleğini yönet	
	Manage machinery	
	Bileşen listelerini yönet	

Not: Site çapında yetkiler herhangi bir varsayılan role verilmez. Bunlar güçlüdür ve yetkili kullanıcı (superuser) durumuna oldukça yakındır. Çoğu Weblate kurulumunuzdaki tüm projelerini etkiler.

Grup listesi

Aşağıdaki gruplar kurulumun ardından (veya *setupgroups* çalıştırıldıktan sonra) oluşturulur ve bunları istediğiniz gibi değiştirebilirsiniz. Ancak, silerseniz veya yeniden adlandırırsanız taşıma işlemi onları yeniden oluşturacaktır.

Konuklar

Kimliği doğrulanmamış kullanıcılar için izinleri tanımlar.

Bu grup yalnızca anonim kullanıcıları içerir (bkz: *ANONYMOUS_USER_NAME*).

Kimliği doğrulanmamış kullanıcıların izinlerini sınırlamak için bu gruptaki rolleri kaldırabilirsiniz.

Varsayılan roller: *Öneri ekle*, *Depoya eriş*

Görüntüleyiciler

Bu rol, ortak projelerin tüm kullanıcılar için görünür olduğundan emin olunmasını sağlar. Varsayılan olarak tüm kullanıcılar bu grubun üyesidir.

By default, *automatic group assignment* makes all new accounts members of this group when they join.

Varsayılan roller: yok

Kullanıcılar

Tüm kullanıcılar için varsayılan grup.

By default, *automatic group assignment* makes all new accounts members of this group when they join.

Varsayılan roller: *Uzman kullanıcı*

Gözden geçirenler

Gözden geçirenler için grup (bkz: *Çeviri iş akışları*).

Varsayılan roller: *Dizgileri gözden geçir*

Yöneticiler

Yöneticiler için grup.

Varsayılan roller: *Yönetim*

Uyarı: Önceden tanımlanmış Weblate gruplarını ve kullanıcılarını asla kaldırmayın, bu beklenmedik sorunlara yol açabilir! Onları kullanmıyorsanız, bunun yerine tüm ayrıcalıklarını kaldırabilirsiniz.

2.6.3 Ek erişim kısıtlamaları

If you want to use your Weblate installation in a less public manner, i.e. allow new users on an invitational basis only, it can be done by configuring Weblate in such a way that only known users have an access to it. In order to do so, you can set `REGISTRATION_OPEN` to `False` to prevent registrations of any new users, and set `REQUIRE_LOGIN` to `/.*` to require signing in to access all the site pages. This is basically the way to lock your Weblate installation.

İpucu: You can use built-in *Yeni kullanıcı davetiyesi* to add new users.

2.7 Çeviri projeleri

2.7.1 Translation organization

Weblate organizes translatable VCS content of project/components into a tree-like structure.

- The bottom level object is *Project configuration*, which should hold all translations belonging together (for example translation of an application in several versions and/or accompanying documentation).
- On the level above, *Component configuration*, which is actually the component to translate, you define the VCS repository to use, and the mask of files to translate.
- Above *Component configuration* there are individual translations, handled automatically by Weblate as translation files (which match *Dosya maskesi* defined in *Component configuration*) appear in the VCS repository.

Weblate supports a wide range of translation formats (both bilingual and monolingual ones) supported by Translate Toolkit, see *Supported file formats*.

Not: You can share cloned VCS repositories using *Weblate internal URLs*. Using this feature is highly recommended when you have many components sharing the same VCS. It improves performance and decreases required disk space.


2.7.2 Adding translation projects and components

3.2 sürümünde değişti: An interface for adding projects and components is included, and you no longer have to use *Django yönetici arayüzü*.

3.4 sürümünde değişti: The process of adding components is now multi staged, with automated discovery of most parameters.

Based on your permissions, new translation projects and components can be created. It is always permitted for users with the *Add new projects* permission, and if your instance uses billing (e.g. like <https://hosted.weblate.org/> see *Faturalandırma*), you can also create those based on your plans allowance from the user account that manages billing.

You can view your current billing plan on a separate page:

 Weblate
 Dashboard Projects Languages Checks
 + 🌐 ...

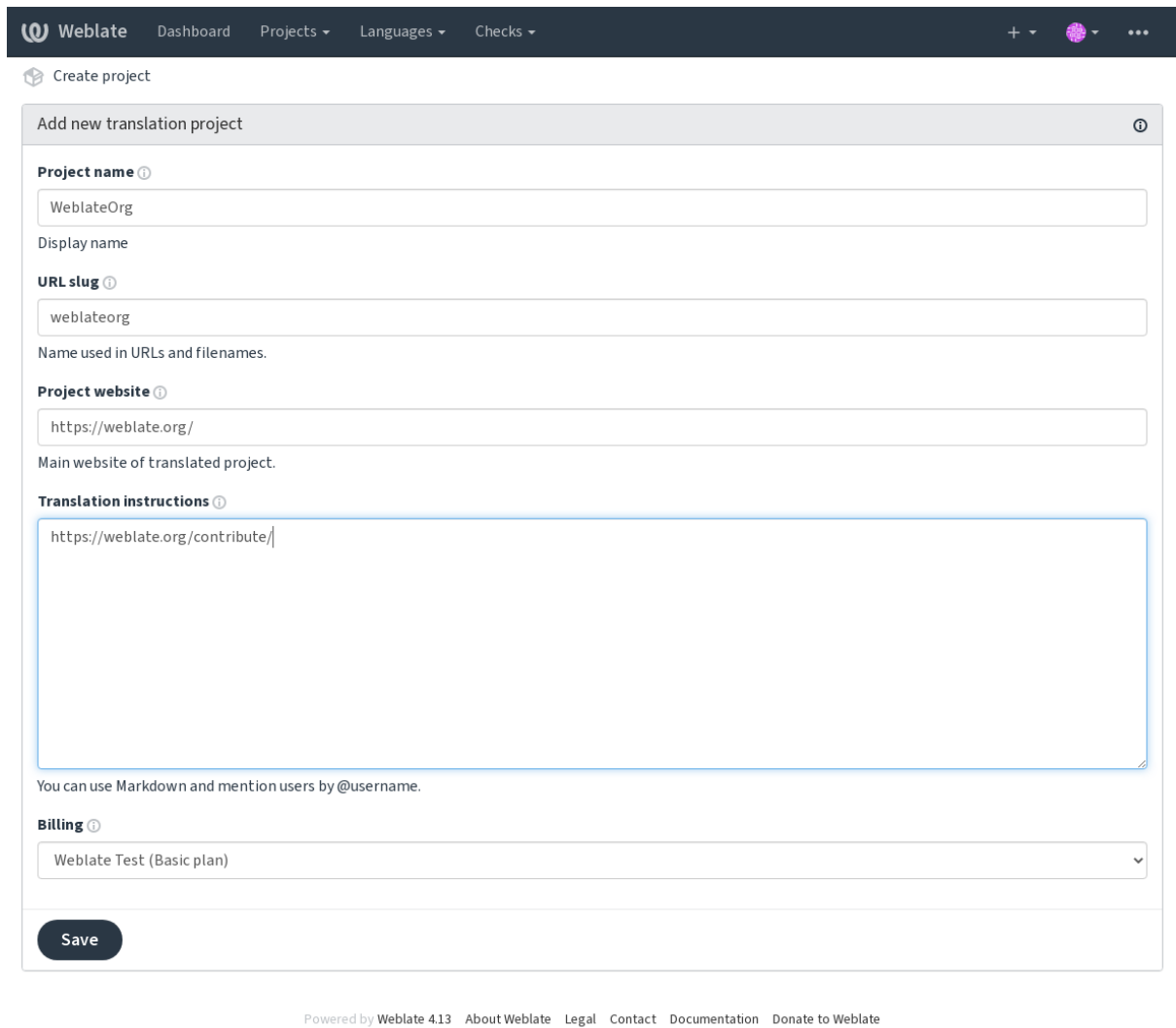
Your profile / Billing

Billing plan		ⓘ
Current plan	Basic plan (Active)	
Monthly price	19 EUR	
Yearly price	199 EUR	
Strings limit	Used 0	<div></div>
Languages limit	Used 0	<div></div>
Last invoice	2022-06-14 - 2022-06-16	
Projects limit	Used 0 of 1	<div></div>
Projects	No projects currently assigned! <button>Add new translation project</button>	
<div>Terminate billing plan</div>		

Invoices		
Invoice period	Invoice amount	Download invoice
06/14/2022 - 06/16/2022	19.0 EUR	Not available

Powered by Weblate 4.13 [About Weblate](#) [Legal](#) [Contact](#) [Documentation](#) [Donate to Weblate](#)

The project creation can be initiated from there, or using the menu in the navigation bar, filling in basic info about the translation project to complete addition of it:



The screenshot shows the 'Add new translation project' form in the Weblate interface. The form is titled 'Add new translation project' and includes several input fields and a text area. The 'Project name' field contains 'WeblateOrg'. The 'URL slug' field contains 'weblateorg'. The 'Project website' field contains 'https://weblate.org/'. The 'Translation instructions' text area contains 'https://weblate.org/contribute/'. The 'Billing' dropdown menu is set to 'Weblate Test (Basic plan)'. A 'Save' button is at the bottom of the form. Below the form, there is a footer with links: 'Powered by Weblate 4.13', 'About Weblate', 'Legal', 'Contact', 'Documentation', and 'Donate to Weblate'.

Webate Dashboard Projects Languages Checks

Create project

Add new translation project

Project name ①

WeblateOrg

Display name

URL slug ①

weblateorg

Name used in URLs and filenames.

Project website ①

https://weblate.org/

Main website of translated project.

Translation instructions ①

https://weblate.org/contribute/

You can use Markdown and mention users by @username.

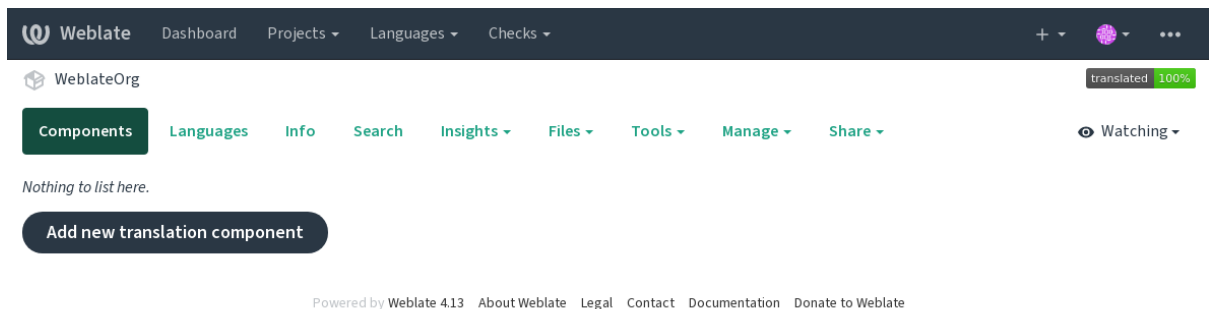
Billing ①

Weblate Test (Basic plan)

Save

Powered by Weblate 4.13 About Weblate Legal Contact Documentation Donate to Weblate

After creating the project, you are taken directly to the project page:



The screenshot shows the project page for 'WeblateOrg' in the Weblate interface. The page has a dark header with the Weblate logo and navigation links: 'Dashboard', 'Projects', 'Languages', and 'Checks'. Below the header, there is a section for 'WeblateOrg' with a 'translated 100%' badge. A navigation bar contains links: 'Components', 'Languages', 'Info', 'Search', 'Insights', 'Files', 'Tools', 'Manage', and 'Share'. A 'Watching' button is also present. Below the navigation bar, there is a message 'Nothing to list here.' and a button 'Add new translation component'. At the bottom, there is a footer with links: 'Powered by Weblate 4.13', 'About Weblate', 'Legal', 'Contact', 'Documentation', and 'Donate to Weblate'.

Webate Dashboard Projects Languages Checks

WeblateOrg translated 100%

Components Languages Info Search Insights Files Tools Manage Share

Watching

Nothing to list here.

Add new translation component

Powered by Weblate 4.13 About Weblate Legal Contact Documentation Donate to Weblate

Creating a new translation component can be initiated via a single click there. The process of creating a component is multi-staged and automatically detects most translation parameters. There are several approaches to creating component:

Sürüm denetiminden

Creates component from remote version control repository.

Mevcut bileşenden

Creates additional component to existing one by choosing different files.

Ek dal

Creates additional component to existing one, just for different branch.

Çeviri dosyalarını yükle

Upload translation files to Weblate in case you do not have version control or do not want to integrate it with Weblate. You can later update the content using the web interface or [Weblate's REST API](#).

Belgeyi çevir

Upload single document or translation file and translate that.

Sıfırdan başla

Create blank translation project and add strings manually.

Once you have existing translation components, you can also easily add new ones for additional files or branches using same repository.

First you need to fill in name and repository location:

Create component

From version control Upload translations files Translate document Start from scratch

Create a new translation component from remote version control system repository.

Component name ⓘ

Language names

Display name

URL slug ⓘ

language-names

Name used in URLs and filenames.

☐ Use as a glossary ⓘ

Project ⓘ

WeblateOrg

Source language ⓘ

English

Language used for source strings in all components

Version control system ⓘ

Git

Version control system to use to access your repository containing translations. You can also choose additional integration with third party providers to submit merge requests.

Source code repository ⓘ

https://github.com/WeblateOrg/demo.git

URL of a repository, use weblate://project/component to share it with other component.


Repository branch ⓘ

Repository branch to translate

Continue

Powered by Weblate 4.13 [About Weblate](#) [Legal](#) [Contact](#) [Documentation](#) [Donate to Weblate](#)

On the next page, you are presented with a list of discovered translatable resources:

 Weblate Dashboard Projects Languages Checks

Create component

Add new translation component ⓘ

Choose translation files to import ⓘ

☐ Specify configuration manually

☐ File format `Android String Resource`, File mask `app/src/main/res/values-*/strings.xml`

☐ File format `gettext PO file`, File mask `weblate/langdata/locale/*/LC_MESSAGES/django.po`


☐ File format `gettext PO file`, File mask `weblate/locale/*/LC_MESSAGES/django.po`

☐ File format `gettext PO file`, File mask `weblate/locale/*/LC_MESSAGES/djangojs.po`

Continue

Powered by Weblate 4.13 About Weblate Legal Contact Documentation Donate to Weblate

As a last step, you review the translation component info and fill in optional details:


Dashboard
Projects
Languages
Checks

Create component

Add new translation component

Project

WeblateOrg

Component name

Language names

Display name

URL slug

language-names

Name used in URLs and filenames.

Version control system

Git

Version control system to use to access your repository containing translations. You can also choose additional integration with third party providers to submit merge requests.

Source code repository

https://github.com/WeblateOrg/demo.git

URL of a repository, use weblate://project/component to share it with other component.

Repository branch

Repository branch to translate

Repository push URL

URL of a push repository, pushing is turned off if empty.

Push branch

Branch for pushing changes, leave empty to use repository branch

Repository browser

https://github.com/WeblateOrg/demo/blob/{{branch}}/{{filename}}#L{{line}}

Link to repository browser, use {{branch}} for branch, {{filename}} and {{line}} as filename and line placeholders. You might want to strip leading directory by using {{filename|parentdir}}.

File format

gettext PO file

File mask

app/src/main/res/values-*/strings.xmlweblate/langdata/locale/*/LC_MESSAGES/django.po

Path of files to translate relative to repository root, use * instead of language code, for example: po/* or locale/*/LC_MESSAGES/django.po.

Monolingual base language file

app/src/main/res/values/strings.xml

Filename of translation base file, containing all strings and their source; it is recommended for monolingual translation formats.

☒ **Edit base file**

Whether users will be able to edit the base file for monolingual translations.

Intermediate language file

Filename of intermediate translation file. In most cases this is a translation file provided by developers and is used when creating actual source strings.

Template for new translations

weblate/langdata/locale/django.pot

Filename of file used for creating new translations. For gettext choose .pot file.

Translation license

GNU General Public License v3.0 or later

Adding new translation

Create new language file

How to handle requests for creating new translations.

Language code style

Default based on the file format

Customize language code used to generate the filename for translations created by Weblate.

Language filter

^(cs|he|hu)\$

Regular expression used to filter translation files when scanning for file mask.

Source language

English

Language used for source strings in all components

☐ **Use as a glossary**

You will be able to edit more options in the component settings after creating it.

Save

Ayrıca bakınız:

Django yönetici arayüzü, Project configuration, Component configuration

2.7.3 Project configuration

Create a translation project and then add a new component for translation in it. The project is like a shelf, in which real translations are stacked. All components in the same project share suggestions and their dictionary; the translations are also automatically propagated through all components in a single project (unless turned off in the component configuration), see *Çeviri Belleği*.

Ayrıca bakınız:

/devel/integration

These basic attributes set up and inform translators of a project:

Proje adı

Verbose project name, used to display the project name.

URL kısaltması

Project name suitable for URLs.

Proje web sitesi

URL where translators can find more info about the project.

This is a required parameter unless turned off by *WEBSITE_REQUIRED*.

Çeviri talimatları

Text describing localization process in the project, and any other information useful for translators. Markdown can be used for text formatting or inserting links.

“Language-Team” (Çeviri-Takımı) başlığını ayarla

Whether Weblate should manage the Language-Team header (this is a *GNU gettext* only feature right now).

Paylaşılan çeviri belleği kullan

Whether to use shared translation memory, see *Paylaşılan çeviri belleği* for more details.

The default value can be changed by *DEFAULT_SHARED_TM*.

Paylaşılan çeviri belleğine katkıda bulunun

Whether to contribute to shared translation memory, see *Paylaşılan çeviri belleği* for more details.

The default value can be changed by `DEFAULT_SHARED_TM`.

Erişim denetimi

Configure per project access control, see *Proje erişim denetimi* for more details.

Öntanımlı değer `DEFAULT_ACCESS_CONTROL` ile değiştirilebilir.

Gözden geçirmeleri etkinleştir

Enable review workflow for translations, see *Dedicated reviewers*.

Kaynak gözden geçirmelerini etkinleştir

Enable review workflow for source strings, see *Source strings reviews*.

Ayrıca bakınız:

report-source, *Yorumlar*

Kancaları etkinleştir

Whether unauthenticated *Bildirim kancaları* are to be used for this repository.

Ayrıca bakınız:

Ara dil dosyası, *Quality gateway for the source strings*, *Bilingual and monolingual formats*, *Dil tanımları*

Dil kod adları

Define language codes mapping when importing translations into Weblate. Use this when language codes are inconsistent in your repositories and you want to get a consistent view in Weblate or in case you want to use non-standard naming of your translation files.

The typical use case might be mapping American English to English: `en_US:en`

Multiple mappings to be separated by comma: `en_GB:en, en_US:en`

Using non standard code: `ia_FOO:ia`

İpucu: The language codes are mapped when matching the translation files and the matches are case sensitive, so make sure you use the source language codes in same form as used in the filenames.

Ayrıca bakınız:

Dil kodlarını ayırıştırma

2.7.4 Component configuration

A component is a grouping of something for translation. You enter a VCS repository location and file mask for which files you want translated, and Weblate automatically fetches from this VCS, and finds all matching translatable files.

Ayrıca bakınız:

[/devel/integration](#)

You can find some examples of typical configurations in the [Supported file formats](#).

Not: It is recommended to keep translation components to a reasonable size - split the translation by anything that makes sense in your case (individual apps or add-ons, book chapters or websites).

Weblate easily handles translations with 10000s of strings, but it is harder to split work and coordinate among translators with such large translation components.

Should the language definition for a translation be missing, an empty definition is created and named as “cs_CZ (generated)”. You should adjust the definition and report this back to the Weblate authors, so that the missing languages can be included in next release.

The component contains all important parameters for working with the VCS, and for getting translations out of it:

Bileşen adı

Verbose component name, used to display the component name.

Component slug

Component name suitable for URLs.

Component project

[Project configuration](#) where the component belongs.

Sürüm denetleme sistemi

VCS to use, see [Sürüm denetimi tümleşimi](#) for details.

Ayrıca bakınız:

[Pushing changes from Weblate](#)

Kaynak kod deposu

VCS repository used to pull changes.

Ayrıca bakınız:

See [Accessing repositories](#) for more details on specifying URLs.

İpucu: This can either be a real VCS URL or `weblate://project/component` indicating that the repository should be shared with another component. See [Weblate internal URLs](#) for more details.

Depo yollama URL'si

Repository URL used for pushing. This setting is used only for *Git* and *Mercurial* and push support is turned off for these when this is empty.

For linked repositories, this is not used and setting from linked component applies.

Ayrıca bakınız:

See *Accessing repositories* for more details on how to specify a repository URL and *Pushing changes from Weblate* for more details on pushing changes from Weblate.

Depo tarayıcısı

URL of repository browser used to display source files (location of used messages). When empty, no such links will be generated. You can use *Template markup*.

For example on GitHub, use something like: `https://github.com/WeblateOrg/hello/blob/{{branch}}/{{filename}}#L{{line}}`

In case your paths are relative to different folder (path contains `..`), you might want to strip leading directory by `parentdir` filter (see *Template markup*): `https://github.com/WeblateOrg/hello/blob/{{branch}}/{{filename|parentdir}}#L{{line}}`

Dışa aktarılmış depo URL'si

URL where changes made by Weblate are exported. This is important when *Sürekli yerelleştirme* is not used, or when there is a need to manually merge changes. You can use *Git exporter* to automate this for Git repositories.

Depo dalı

Which branch to checkout from the VCS, and where to look for translations.

For linked repositories, this is not used and setting from linked component applies.

Yollama dalı

Branch for pushing changes, leave empty to use *Depo dalı*.

For linked repositories, this is not used and setting from linked component applies.

Not: This is currently only supported for Git, GitLab and GitHub, it is ignored for other VCS integrations.

Ayrıca bakınız:

Pushing changes from Weblate

Dosya maskesi

Mask of files to translate, including path. It should include one “*” replacing language code (see *Dil tanımları* for info on how this is processed). In case your repository contains more than one translation file (e.g. more gettext domains), you need to create a component for each of them.

For example `po/* .po` or `locale/*/LC_MESSAGES/django.po`.

In case your filename contains special characters such as `[,]`, these need to be escaped as `[[]` or `[]]`.

Ayrıca bakınız:

Bilingual and monolingual formats, What does mean “There are more files for the single language (en)”?

Tek dilli taban dil dosyası

Base file containing string definitions for *Monolingual components*.

Ayrıca bakınız:

Bilingual and monolingual formats, What does mean “There are more files for the single language (en)”?

Taban dosyayı düzenle

Whether to allow editing the base file for *Monolingual components*.

Ara dil dosyası

Intermediate language file for *Monolingual components*. In most cases this is a translation file provided by developers and is used when creating actual source strings.

When set, the source strings are based on this file, but all other languages are based on *Tek dilli taban dil dosyası*. In case the string is not translated into the source language, translating to other languages is prohibited. This provides *Quality gateway for the source strings*.

Ayrıca bakınız:

Quality gateway for the source strings, Bilingual and monolingual formats, What does mean “There are more files for the single language (en)”?

Yeni çeviriler için şablon

Base file used to generate new translations, e.g. `.pot` file with gettext.

İpucu: In many monolingual formats Weblate starts with empty file by default. Use this in case you want to have all strings present with empty value when creating new translation.

Ayrıca bakınız:

adding-translation, *Adding new translations, Yeni çeviri ekleniyor, Bilingual and monolingual formats, What does mean “There are more files for the single language (en)”?*

Dosya biçimi

Translation file format, see also *Supported file formats*.

Kaynak dizgi hata bildirme adresi

Email address used for reporting upstream bugs. This address will also receive notification about any source string comments made in Weblate.

Çeviri yaymaya izin ver

You can turn off propagation of translations to this component from other components within same project. This really depends on what you are translating, sometimes it's desirable to have make use of a translation more than once.

It's usually a good idea to turn this off for monolingual translations, unless you are using the same IDs across the whole project.

Default value can be changed by `DEFAULT_TRANSLATION_PROPAGATION`.

Ayrıca bakınız:

Keeping translations same across components

Önerileri etkinleştir

Whether translation suggestions are accepted for this component.

Öneri oylaması

Turns on vote casting for suggestions, see *Öneri oylaması*.

Önerileri kendiliğinden kabul et

Automatically accept voted suggestions, see *Öneri oylaması*.

Çeviri işaretleri

Customization of quality checks and other Weblate behavior, see *Customizing behavior using flags*.

Zorunlu denetimler

List of checks which can not be ignored, see *Enforcing checks*.

Not: Enforcing the check does not automatically enable it, you still should enabled it using *Customizing behavior using flags* in *Çeviri işaretleri* or *Additional info on source strings*.

Çeviri lisansı

License of the translation (does not need to be the same as the source code license).

Katılımcı sözleşmesi

Bir kullanıcının bu bileşeni çevirmeden önce onaylaması gereken kullanıcı sözleşmesi.

Yeni çeviri ekleniyor

How to handle requests for creation of new languages. Available options:

İletişim sorumlusu

User can select desired language and the project maintainers will receive a notification about this. It is up to them to add (or not) the language to the repository.

Çeviri talimatları URL'sini göster

User is presented a link to page which describes process of starting new translations. Use this in case more formal process is desired (for example forming a team of people before starting actual translation).

Yeni dil dosyası oluştur

User can select language and Weblate automatically creates the file for it and translation can begin.

Yeni çeviri ekleme devre dışı

There will be no option for user to start new translation.

İpucu: The project admins can add new translations even if it is disabled here when it is possible (either *Yeni çeviriler için şablon* or the file format supports starting from an empty file).

Ayrıca bakınız:

adding-translation, *Adding new translations*

Dizgileri yönet

4.5 sürümünde geldi.

Configures whether users in Weblate will be allowed to add new strings and remove existing ones. Adjust this to match your localization workflow - how the new strings are supposed to be introduced.

For bilingual formats, the strings are typically extracted from the source code (for example by using `xgettext`) and adding new strings in Weblate should be disabled (they would be discarded next time you update the translation files). In Weblate you can manage strings for every translation and it does not enforce the strings in all translations to be consistent.

For monolingual formats, the strings are managed only on source language and are automatically added or removed in the translations. The strings appear in the translation files once they are translated.

Ayrıca bakınız:

Bilingual and monolingual formats, `adding-new-strings`, `POST /api/translations/(string:project)/(string:component)/(string:language)/units/`

Dil kodu stili

Weblate tarafından oluşturulan çeviriler için dosya adını oluşturmak için kullanılan dil kodunu özelleştirin.

Ayrıca bakınız:

Adding new translations, Dil kodu, Dil kodlarını ayırıştırma

Birleştirme stili

You can configure how updates from the upstream repository are handled. The actual implementation depends on VCS, see *Sürüm denetimi tümleşimi*.

Yeniden konumla

Rebases Weblate commits on top of upstream repository on update. This provides clean history without extra merge commits.

Rebasing can cause you trouble in case of complicated merges, so carefully consider whether or not you want to enable them.

You might need to enable force pushing by choosing *Git (zorlama push ile)* as *Sürüm denetleme sistemi*, especially when pushing to a different branch.

Birleştir

Upstream repository changes are merged into Weblate one. This setting utilizes fast-forward when possible. This is the safest way, but might produce a lot of merge commits.

Hızlı ileri almadan birleştir

Upstream repository changes are merged into Weblate one with doing a merge commit every time (even when fast-forward would be possible). Every Weblate change will appear as a merge commit in Weblate repository.

Default value can be changed by `DEFAULT_MERGE_STYLE`.

Commit, add, delete, merge, add-on, and merge request messages

Message used when committing a translation, see *Template markup*.

Default value can be changed by `DEFAULT_ADD_MESSAGE`, `DEFAULT_ADDON_MESSAGE`, `DEFAULT_COMMIT_MESSAGE`, `DEFAULT_DELETE_MESSAGE`, `DEFAULT_MERGE_MESSAGE`, `DEFAULT_PULL_MESSAGE`.

İşlemeye yolla

Whether committed changes should be automatically pushed to the upstream repository. When enabled, the push is initiated once Weblate commits changes to its underlying repository (see *Lazy commits*). To actually enable pushing *Repository push URL* has to be configured as well.

İşlenecek değişikliklerin yaşı

Sets how old (in hours) changes have to be before they are committed by background task or the `commit_pending` management command. All changes in a component are committed once there is at least one change older than this period.

Default value can be changed by `COMMIT_PENDING_HOURS`.

İpucu: There are other situations where pending changes might be committed, see *Lazy commits*.

Hata durumunda kilitle

Locks the component (and linked components, see [Weblate internal URLs](#)) upon the first failed push or merge into its upstream repository, or pull from it. This avoids adding another conflicts, which would have to be resolved manually.

The component will be automatically unlocked once there are no repository errors left.

Kaynak dil

Language used for source strings. Change this if you are translating from something else than English.

İpucu: In case you are translating bilingual files from English, but want to be able to do fixes in the English translation as well, choose *English (Developer)* as a source language to avoid conflict between the name of the source language and the existing translation.

For monolingual translations, you can use intermediate translation in this case, see [Ara dil dosyası](#).

Dil süzgeci

Regular expression used to filter the translation when scanning for file mask. It can be used to limit the list of languages managed by Weblate.

Not: You need to list language codes as they appear in the filename.

Some examples of filtering:

Filter description	Düzenli ifade
Selected languages only	<code>^(cs de es)\$</code>
Exclude languages	<code>^(?! (it fr))\$</code> .+\$
Filter two letter codes only	<code>^[.]+\$</code>
Exclude non language files	<code>^(?! (blank))\$</code> .+\$
Include all files (default)	<code>^[^.] +\$</code>

Çeşitlerin düzenli ifadesi

Regular expression used to determine the variants of a string, see variants.

Not: Most of the fields can be edited by project owners or administrators, in the Weblate interface.

Ayrıca bakınız:

Does Weblate support other VCSes than Git and Mercurial?, alerts

Öncelik

Yüksek öncelikli bileşenler çevirmenlere ilk olarak sunulur.

Restricted access

By default the component is visible to anybody who has access to the project, even if the person can not perform any changes in the component. This makes it easier to keep translation consistency within the project.

Restricting access at a component, or component-list level takes over access permission to a component, regardless of project-level permissions. You will have to grant access to it explicitly. This can be done through granting access to a new user group and putting users in it, or using the default *custom* or *private* access control groups.

The default value can be changed in `DEFAULT_RESTRICTED_COMPONENT`.

İpucu: This applies to project admins as well — please make sure you will not loose access to the component after toggling the status.

Projelerde paylaş

You can choose additional projects where the component will be visible. Useful for shared libraries which you use in several projects.

Not: Sharing a component doesn't change its access control. It only makes it visible when browsing other projects. Users still need access to the actual component to browse or translate it.

Sözlük olarak kullan

4.5 sürümünde geldi.

Bu bileşenin sözlük olarak kullanılmasına izin verir. Nasıl listeleneceğini *Sözlük rengi* kullanarak yapılandırabilirsiniz.

The glossary will be accessible in all projects defined by *Projelerde paylaş*.

It is recommended to enable *Dizgileri yönet* on glossaries in order to allow adding new words to them.

Ayrıca bakınız:

Sözlük

Sözlük rengi

Display color for a glossary used when showing word matches.

2.7.5 Template markup

Weblate uses simple markup language in several places where text rendering is needed. It is based on *The Django template language*, so it can be quite powerful.

Currently it is used in:

- Commit message formatting, see *Component configuration*
- **Several add-ons**
 - *Bileşen keşfi*

- *İstatistik oluşturucu*
- *Eklentiden betikleri çalıştırma*

There following variables are available in the component templates:

```
{{ language_code }}
    Dil kodu

{{ language_name }}
    Dil adı

{{ component_name }}
    Bileşen adı

{{ component_slug }}
    Component slug

{{ project_name }}
    Proje adı

{{ project_slug }}
    Project slug

{{ url }}
    Translation URL

{{ filename }}
    Çeviri dosya adı

{{ stats }}
    Translation stats, this has further attributes, examples below.

{{ stats.all }}
    Total strings count

{{ stats.fuzzy }}
    Count of strings needing review

{{ stats.fuzzy_percent }}
    Percent of strings needing review

{{ stats.translated }}
    Translated strings count

{{ stats.translated_percent }}
    Translated strings percent

{{ stats.allchecks }}
    Number of strings with failing checks

{{ stats.allchecks_percent }}
    Percent of strings with failing checks

{{ author }}
    Author of current commit, available only in the commit scope.

{{ addon_name }}
    Name of currently executed add-on, available only in the add-on commit message.
```

The following variables are available in the repository browser or editor templates:

```
{{branch}}
    current branch

{{line}}
    line in file
```

```
{{filename}}
```

filename, you can also strip leading parts using the `parentdir` filter, for example `{{filename|parentdir}}`

You can combine them with filters:

```
{{ component|title }}
```

You can use conditions:

```
{% if stats.translated_percent > 80 %}Well translated!{% endif %}
```

There is additional tag available for replacing characters:

```
{% replace component "-" " " %}
```

You can combine it with filters:

```
{% replace component|capfirst "-" " " %}
```

There are also additional filter to manipulate with filenames:

```
Directory of a file: {{ filename|dirname }}
File without extension: {{ filename|striptext }}
File in parent dir: {{ filename|parentdir }}
It can be used multiple times: {{ filename|parentdir|parentdir }}
```

...and other Django template features.

2.7.6 Importing speed

Fetching VCS repository and importing translations to Weblate can be a lengthy process, depending on size of your translations. Here are some tips:

Optimize configuration

The default configuration is useful for testing and debugging Weblate, while for a production setup, you should do some adjustments. Many of them have quite a big impact on performance. Please check [Production setup](#) for more details, especially:

- Configure Celery for executing background tasks (see [Background tasks using Celery](#))
- [Enable caching](#)
- [Use a powerful database engine](#)
- [Disable debug mode](#)

Check resource limits

If you are importing huge translations or repositories, you might be hit by resource limitations of your server.

- Check the amount of free memory, having translation files cached by the operating system will greatly improve performance.
- Disk operations might be bottleneck if there is a lot of strings to process—the disk is pushed by both Weblate and the database.
- Additional CPU cores might help improve performance of background tasks (see [Background tasks using Celery](#)).

Disable unneeded checks

Some quality checks can be quite expensive, and if not needed, can save you some time during import if omitted. See [CHECK_LIST](#) for info on configuration.

2.7.7 Automatic creation of components

In case your project has dozen of translation files (e.g. for different gettext domains, or parts of Android apps), you might want to import them automatically. This can either be achieved from the command-line by using `import_project` or `import_json`, or by installing the *Bileşen keşfi* add-on.

To use the add-on, you first need to create a component for one translation file (choose the one that is the least likely to be renamed or removed in future), and install the add-on on this component.

For the management commands, you need to create a project which will contain all components and then run `import_project` or `import_json`.

Ayrıca bakınız:

Management commands, *Bileşen keşfi*

2.8 Dil tanımları

Farklı çevirileri doğru bir şekilde sunmak için dil adı, metin yönü, çoğul tanımları ve dil kodu hakkında bilgiler gereklidir.

2.8.1 Dil kodlarını ayırıştırma

While parsing translations, Weblate attempts to map language code (usually the ISO 639-1 one) from the *Dosya maskesi* to any existing language object.

Dil kod adları ile bu işlemeyi proje düzeyinde daha fazla ayarlayabilirsiniz.

Tam eşleşme bulunamazsa, mevcut bir dile en iyi şekilde uydurulmaya çalışılır. Aşağıdaki adımlar denenir:

- Büyük/küçük harfe duyarlı olmayan aramalar.
- Alt çizgileri ve kısa çizgileri normalleştirme.
- Yerleşik dil kod adlarında arama.
- Dil adına göre arama.
- Belirli bir dil için varsayılan ülke kodunu yok sayma—`cs_CZ` yerine `cs` seçme.

Bunun da başarısız olması durumunda, varsayılanlar kullanılarak (soldan sağa metin yönü, bir tane çoğul) yeni bir dil tanımı oluşturulacaktır. `xx_XX` koduyla otomatik olarak oluşturulan dil `xx_XX` (*generated*) olarak adlandırılacaktır. Bunu daha sonra yönetici arayüzünde değiştirmek (bkz: *Changing language definitions*) ve sorun izleyiciye bildirmek (bkz: *Weblate'e Katkıda Bulunma*) isteyebilirsiniz, böylece uygun tanımın gelecek Weblate sürümüne eklenmesini sağlayabilirsiniz.

İpucu: Dil olarak istenmeyen bir şey görürseniz, çevirileri ayırıştırırken bu tür bir dosyayı yok saymak için *Dil süzgeci* ayarlamak isteyebilirsiniz.

Ayrıca bakınız:

Dil kodu, *Adding new translations*

2.8.2 Changing language definitions

You can change language definitions in the languages interface (`/languages/` URL).

While editing, make sure all fields are correct (especially plurals and text direction), otherwise translators will be unable to properly edit those translations.

2.8.3 Yerleşik dil tanımları

Definitions for about 600 languages are included in Weblate and the list is extended in every release. Whenever Weblate is upgraded (more specifically whenever **weblate migrate** is executed, see [Generic upgrade instructions](#)) the database of languages is updated to include all language definitions shipped in Weblate.

This feature can be disabled using `UPDATE_LANGUAGES`. You can also enforce updating the database to match Weblate built-in data using `setuplang`.

Ayrıca bakınız:

[Yerleşik dil tanımlarını genişletme](#)

2.8.4 Belirsiz dil kodları ve makro diller

In many cases it is not a good idea to use macrolanguage code for a translation. The typical problematic case might be Kurdish language, which might be written in Arabic or Latin script, depending on actual variant. To get correct behavior in Weblate, it is recommended to use individual language codes only and avoid macrolanguages.

Ayrıca bakınız:

[Macrolanguages definition](#), [List of macrolanguages](#)

2.8.5 Dil tanımları

Each language consists of following fields:

Dil kodu

Code identifying the language. Weblate prefers two letter codes as defined by [ISO 639-1](#), but uses [ISO 639-2](#) or [ISO 639-3](#) codes for languages that do not have two letter code. It can also support extended codes as defined by [BCP 47](#).

Ayrıca bakınız:

[Dil kodlarını ayırıştırma](#), [Adding new translations](#)

Dil adı

Visible name of the language. The language names included in Weblate are also being localized depending on user interface language.

Metin yönü

Determines whether language is written right to left or left to right. This property is autodetected correctly for most of the languages.

Plural number

Number of plurals used in the language.

Çoğul formülü

Gettext compatible plural formula used to determine which plural form is used for given count.

Ayrıca bakınız:

[Çoğullar](#), GNU gettext utilities: Plural forms, [Language Plural Rules by the Unicode Consortium](#)

2.8.6 Adding new translations

2.18 sürümünde değişti: In versions prior to 2.18 the behaviour of adding new translations was file format specific.

Weblate can automatically start new translation for all of the file formats.

Some formats expect to start with an empty file and only translated strings to be included (for example *Android string resources*), while others expect to have all keys present (for example *GNU gettext*). The document-based formats (for example *OpenDocument Format*) start with a copy of the source document and all strings marked as needing editing. In some situations this really doesn't depend on the format, but rather on the framework you use to handle the translation (for example with *JSON files*).

When you specify *Yeni çeviriler için şablon* in *Component configuration*, Weblate will use this file to start new translations. Any exiting translations will be removed from the file when doing so.

When *Yeni çeviriler için şablon* is empty and the file format supports it, an empty file is created where new strings will be added once they are translated.

The *Dil kodu stili* allows you to customize language code used in generated filenames:

Dosya biçimine göre varsayılan

Dependent on file format, for most of them POSIX is used.

Ayrırcı olarak alt çizgi kullanan POSIX stili

Typically used by gettext and related tools, produces language codes like `pt_BR`.

Ülke kodu da dahil olmak üzere ayrırcı olarak alt çizgi kullanan POSIX stili

POSIX style language code including the country code even when not necessary (for example `cs_CZ`).

Ayrırcı olarak kısa çizgi kullanan BCP stili

Typically used on web platforms, produces language codes like `pt-BR`.

Ülke kodu da dahil olmak üzere ayrırcı olarak kısa çizgi kullanan BCP stili

BCP style language code including the country code even when not necessary (for example `cs-CZ`).

Android stili

Only used in Android apps, produces language codes like `pt-rBR`.

Java tarzı

Used by Java—mostly BCP with legacy codes for Chinese.

Linux stili

Locales as used by Linux, uses legacy codes for Chinese and POSIX style notation.

Additionally, any mappings defined in *Dil kod adları* are applied in reverse.

Not: Weblate recognizes any of these when parsing translation files, the above settings only influences how new files are created.

Ayrıca bakınız:

Dil kodu, Dil kodlarını ayırıştırma

2.9 Sürekli yerelleştirme

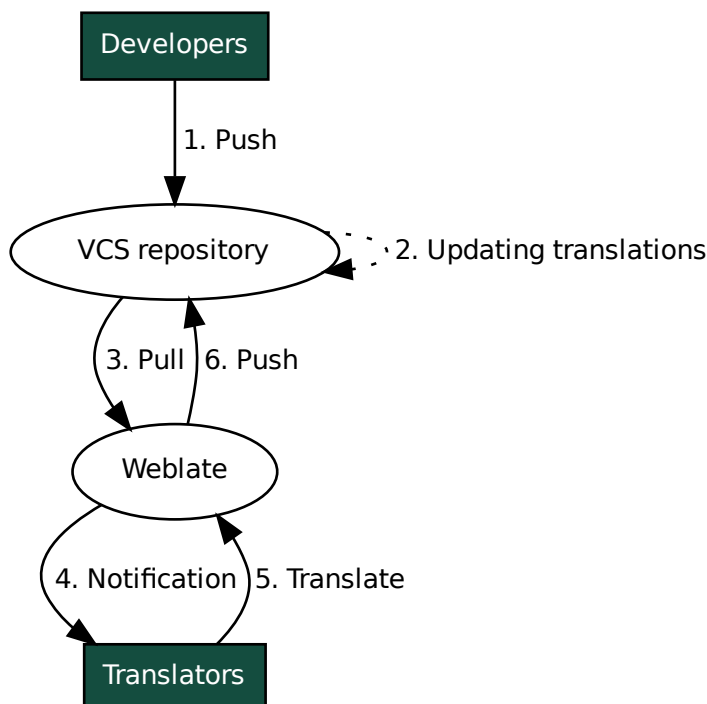
There is infrastructure in place so that your translation closely follows development. This way translators can work on translations the entire time, instead of working through huge amount of new text just prior to release.

Ayrıca bakınız:

/devel/integration describes basic ways to integrate your development with Weblate.

This is the process:

1. Developers make changes and push them to the VCS repository.
2. Optionally the translation files are updated (this depends on the file format, see *Why does Weblate still show old translation strings when I've updated the template?*).
3. Weblate pulls changes from the VCS repository, see *Updating repositories*.
4. Once Weblate detects changes in translations, translators are notified based on their subscription settings.
5. Translators submit translations using the Weblate web interface, or upload offline changes.
6. Once the translators are finished, Weblate commits the changes to the local repository (see *Lazy commits*) and pushes them back if it has permissions to do so (see *Pushing changes from Weblate*).



2.9.1 Updating repositories

You should set up some way of updating backend repositories from their source.

- Use *Bildirim kanalları* to integrate with most of common code hosting services:
 - *Automatically receiving changes from GitHub*
 - *Automatically receiving changes from GitLab*
 - *Automatically receiving changes from Bitbucket*
 - *Automatically receiving changes from Pagure*
 - *Automatically receiving changes from Azure Repos*
- Manually trigger update either in the repository management or using *Weblate's REST API* or *Weblate Client*
- Enable `AUTO_UPDATE` to automatically update all components on your Weblate instance
- Execute `updategit` (with selection of project or `--all` to update all)

Whenever Weblate updates the repository, the post-update addons will be triggered, see *Eklentiler*.

Avoiding merge conflicts

The merge conflicts from Weblate arise when same file was changed both in Weblate and outside it. There are two approaches to deal with that - avoid edits outside Weblate or integrate Weblate into your updating process, so that it flushes changes prior to updating the files outside Weblate.

The first approach is easy with monolingual files - you can add new strings within Weblate and leave whole editing of the files there. For bilingual files, there is usually some kind of message extraction process to generate translatable files from the source code. In some cases this can be split into two parts - one for the extraction generates template (for example gettext POT is generated using **xgettext**) and then further process merges it into actual translations (the gettext PO files are updated using **msgmerge**). You can perform the second step within Weblate and it will make sure that all pending changes are included prior to this operation.

The second approach can be achieved by using *Weblate's REST API* to force Weblate to push all pending changes and lock the translation while you are doing changes on your side.

The script for doing updates can look like this:

```
# Lock Weblate translation
wlc lock
# Push changes from Weblate to upstream repository
wlc push
# Pull changes from upstream repository to your local copy
git pull
# Update translation files, this example is for Django
./manage.py makemessages --keep-pot -a
git commit -m 'Locale updates' -- locale
# Push changes to upstream repository
git push
# Tell Weblate to pull changes (not needed if Weblate follows your repo
# automatically)
wlc pull
# Unlock translations
wlc unlock
```

If you have multiple components sharing same repository, you need to lock them all separately:

```
wlc lock foo/bar
wlc lock foo/baz
wlc lock foo/baj
```

Not: The example uses *Weblate Client*, which needs configuration (API keys) to be able to control Weblate remotely. You can also achieve this using any HTTP client instead of wlc, e.g. curl, see *Weblate's REST API*.

Ayrıca bakınız:

Weblate Client

Automatically receiving changes from GitHub

Weblate comes with native support for GitHub.

If you are using Hosted Weblate, the recommended approach is to install the *Weblate app*, that way you will get the correct setup without having to set much up. It can also be used for pushing changes back.

To receive notifications on every push to a GitHub repository, add the Weblate Webhook in the repository settings (*Webhooks*) as shown on the image below:

The screenshot shows the GitHub 'Add webhook' interface for the repository 'WeblateOrg / hello'. The left sidebar contains a menu with 'Options', 'Collaborators & teams', 'Branches', 'Webhooks' (highlighted), 'Integrations & services', 'Deploy keys', and 'Alerts'. The main content area is titled 'Webhooks / Add webhook' and includes the following fields and options:

- Payload URL:** A text input field containing 'https://hosted.weblate.org/hooks/github/'.
- Content type:** A dropdown menu set to 'application/x-www-form-urlencoded'.
- Secret:** An empty text input field.
- SSL verification:** A checkbox labeled 'By default, we verify SSL certificates when delivering payloads.' with a red 'Disable SSL verification' button.
- Which events would you like to trigger this webhook?:** Three radio button options: 'Just the push event.' (selected), 'Send me everything.', and 'Let me select individual events.'
- Active:** A checked checkbox with the text 'We will deliver event details when this hook is triggered.'
- Add webhook:** A green button at the bottom.

The footer of the page shows the GitHub logo, copyright information '© 2018 GitHub, Inc.', and links for 'Terms', 'Privacy', 'Security', 'Status', 'Help', 'Contact GitHub', 'API', 'Training', 'Shop', 'Blog', and 'About'.

For the payload URL, append `/hooks/github/` to your Weblate URL, for example for the Hosted Weblate service, this is `https://hosted.weblate.org/hooks/github/`.

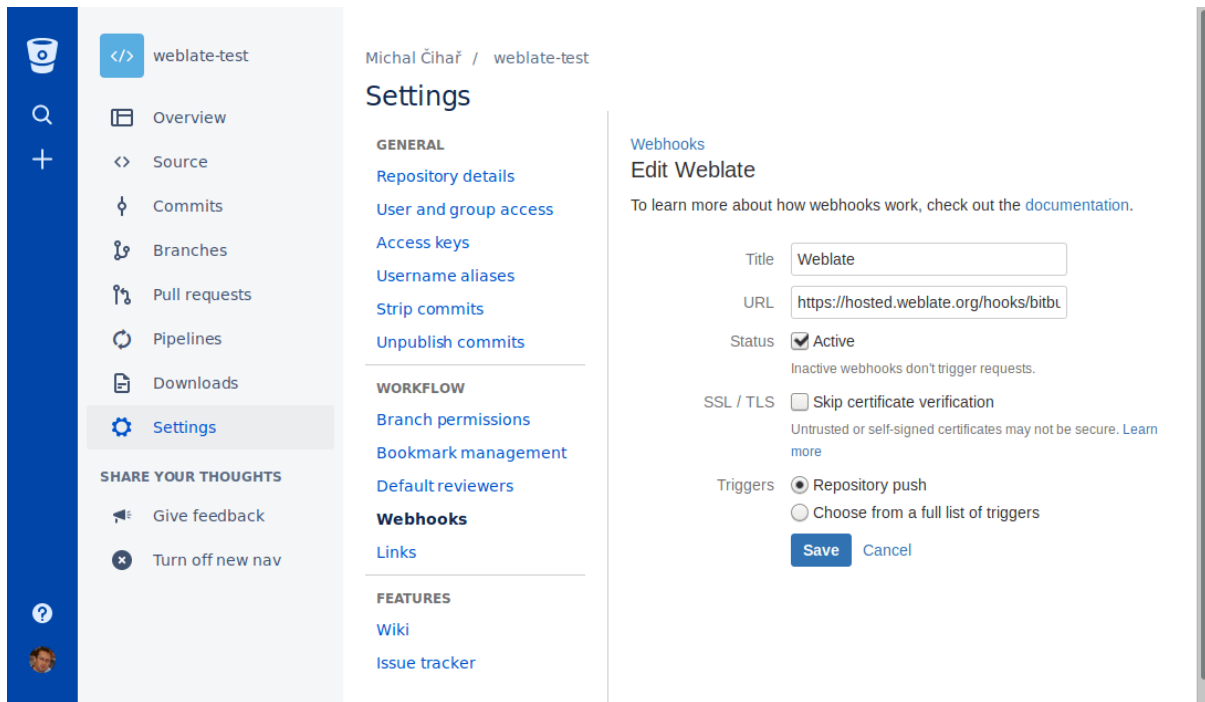
You can leave other values at default settings (Weblate can handle both content types and consumes just the *push* event).

Ayrıca bakınız:

POST /hooks/github/, Accessing repositories from Hosted Weblate

Automatically receiving changes from Bitbucket

Weblate has support for Bitbucket webhooks, add a webhook which triggers upon repository push, with destination to `/hooks/bitbucket/` URL on your Weblate installation (for example `https://hosted.weblate.org/hooks/bitbucket/`).

**Ayrıca bakınız:**

POST /hooks/bitbucket/, Accessing repositories from Hosted Weblate

Automatically receiving changes from GitLab

Weblate has support for GitLab hooks, add a project webhook with destination to `/hooks/gitlab/` URL on your Weblate installation (for example `https://hosted.weblate.org/hooks/gitlab/`).

Ayrıca bakınız:

POST /hooks/gitlab/, Accessing repositories from Hosted Weblate

Automatically receiving changes from Pagure

3.3 sürümünde geldi.

Weblate has support for Pagure hooks, add a webhook with destination to `/hooks/pagure/` URL on your Weblate installation (for example `https://hosted.weblate.org/hooks/pagure/`). This can be done in *Activate Web-hooks* under *Project options*:

The screenshot shows the Weblate interface for a project named 'nijel-test'. The top navigation bar includes 'Browse', 'Create', and a user profile icon. Below the project name, there are links for 'New Issue', 'Open PR', 'Fork', and 'Clone'. The left sidebar lists various settings categories: Project Settings, Project Details, Default Branch, Private Web Hook Key, API Keys, Project Options (selected), Public Notifications, Users & Groups, Deploy Keys, Hooks, Priorities, Roadmap, Close Status, Custom Issue Fields, Reports, Tags, Quick Replies, Regenerate Repos, Give Project, and Delete Project. The main content area is titled 'Project Options' and contains several checkboxes for enabling features like 'Activate always merge', 'Activate disable non fast-forward merges', 'Activate Enforce signed-off commits in pull-request', 'Activate fedmsg notifications', 'Activate Issue tracker', 'Activate Issue tracker read only', 'Activate Issues default to private', 'Activate Minimum score to merge pull-request' (set to -1), 'Activate notify on commit flag', 'Activate notify on pull-request flag', 'Activate Only assignee can merge pull-request', 'Activate open metadata access to all', 'Activate project documentation', 'Activate pull request access only', 'Activate pull requests', and 'Activate stomp notifications'. There is also a section for 'Activate Web-hooks' with a text input field containing 'https://hosted.weblate.org/hooks/pagure/' and buttons for 'Update' and 'Test web-hook'. At the bottom, there is a 'Learn more about' section with links to 'Flags', 'Tracker read-only', 'Pull-request access only', 'Roadmap on Issue page', and 'fedmsg notifications'.

Ayrıca bakınız:

POST /hooks/pagure/, Accessing repositories from Hosted Weblate

Automatically receiving changes from Azure Repos

3.8 sürümünde geldi.

Weblate has support for Azure Repos web hooks, add a webhook for *Code pushed* event with destination to `/hooks/azure/` URL on your Weblate installation (for example `https://hosted.weblate.org/hooks/azure/`). This can be done in *Service hooks* under *Project settings*.

Ayrıca bakınız:

Web hooks in Azure DevOps manual, *POST /hooks/azure/, Accessing repositories from Hosted Weblate*

Automatically receiving changes from Gitea Repos

3.9 sürümünde geldi.

Weblate has support for Gitea webhooks, add a *Gitea Webhook* for *Push events* event with destination to `/hooks/gitea/` URL on your Weblate installation (for example `https://hosted.weblate.org/hooks/gitea/`). This can be done in *Webhooks* under repository *Settings*.

Ayrıca bakınız:

Webhooks in Gitea manual, *POST /hooks/gitea/*, *Accessing repositories from Hosted Weblate*

Automatically receiving changes from Gitee Repos

3.9 sürümünde geldi.

Weblate has support for Gitee webhooks, add a *WebHook* for *Push* event with destination to `/hooks/gitee/` URL on your Weblate installation (for example `https://hosted.weblate.org/hooks/gitee/`). This can be done in *WebHooks* under repository *Management*.

Ayrıca bakınız:

Webhooks in Gitee manual, *POST /hooks/gitee/*, *Accessing repositories from Hosted Weblate*

Automatically updating repositories nightly

Weblate automatically fetches remote repositories nightly to improve performance when merging changes later. You can optionally turn this into doing nightly merges as well, by enabling *AUTO_UPDATE*.

2.9.2 Pushing changes from Weblate

Each translation component can have a push URL set up (see *Depo yollama URL'si*), and in that case Weblate will be able to push change to the remote repository. Weblate can be also be configured to automatically push changes on every commit (this is default, see *İşlemeye yolla*). If you do not want changes to be pushed automatically, you can do that manually under *Repository maintenance* or using API via *wlc push*.

The push options differ based on the *Sürüm denetimi tümleşimi* used, more details are found in that chapter.

In case you do not want direct pushes by Weblate, there is support for *GitHub pull requests*, *GitLab merge requests*, *Pagure merge requests* pull requests or *Gerrit* reviews, you can activate these by choosing *GitHub*, *GitLab*, *Gerrit* or *Pagure* as *Sürüm denetleme sistemi* in *Component configuration*.

Overall, following options are available with Git, GitHub and GitLab:

Desired setup	<i>Sürüm denetleme sistemi</i>	<i>Depo yollama URL'si</i>	<i>Yollama dali</i>
No push	<i>Git</i>	<i>empty</i>	<i>empty</i>
Push directly	<i>Git</i>	SSH URL	<i>empty</i>
Ayrı bir dala yolla	<i>Git</i>	SSH URL	Branch name
GitHub pull request from fork	<i>GitHub pull requests</i>	<i>empty</i>	<i>empty</i>
GitHub pull request from branch	<i>GitHub pull requests</i>	SSH URL ¹	Branch name
GitLab merge request from fork	<i>GitLab merge requests</i>	<i>empty</i>	<i>empty</i>
GitLab merge request from branch	<i>GitLab merge requests</i>	SSH URL ¹	Branch name
Pagure merge request from fork	<i>Pagure merge requests</i>	<i>empty</i>	<i>empty</i>
Pagure merge request from branch	<i>Pagure merge requests</i>	SSH URL ¹	Branch name

Not: You can also enable automatic pushing of changes after Weblate commits, this can be done in *İşlemeye yolla*.

¹ Can be empty in case *Kaynak kod deposu* supports pushing.

Ayrıca bakınız:

See [Accessing repositories](#) for setting up SSH keys, and [Lazy commits](#) for info about when Weblate decides to commit changes.

Protected branches

If you are using Weblate on protected branch, you can configure it to use pull requests and perform actual review on the translations (what might be problematic for languages you do not know). An alternative approach is to waive this limitation for the Weblate push user.

For example on GitHub this can be done in the repository configuration:

☒ **Require pull request reviews before merging**
When enabled, all commits must be made to a non-protected branch and submitted via a pull request with the required number of approving reviews and no changes requested before it can be merged into a branch that matches this rule.

Required approving reviews: **1** ▼


☐ **Dismiss stale pull request approvals when new commits are pushed**
New reviewable commits pushed to a matching branch will dismiss pull request review approvals.


☐ **Require review from Code Owners**
Require an approved review in pull requests including files with a designated code owner.

☒ **Restrict who can dismiss pull request reviews**
Specify people or teams allowed to dismiss pull request reviews.

Q Search for people or teams

People and teams that can dismiss reviews.

 **Organization and repository administrators**
These members can always dismiss.

 **weblate**
Weblate push user ×

2.9.3 Interacting with others

Weblate makes it easy to interact with others using its API.

Ayrıca bakınız:

Weblate's REST API

2.9.4 Lazy commits

The behaviour of Weblate is to group commits from the same author into one commit if possible. This greatly reduces the number of commits, however you might need to explicitly tell it to do the commits in case you want to get the VCS repository in sync, e.g. for merge (this is by default allowed for the *Managers* group, see *Yetkiler ve yerleşik rollerin listesi*).

The changes in this mode are committed once any of the following conditions are fulfilled:

- Somebody else changes an already changed string.
- A merge from upstream occurs.
- An explicit commit is requested.
- Change is older than period defined as *İşlenecek değişikliklerin yaşı* on *Component configuration*.

İpucu: Commits are created for every component. So in case you have many components you will still see lot of commits. You might utilize *Git işlemlerini sıkıştır* add-on in that case.

If you want to commit changes more frequently and without checking of age, you can schedule a regular task to perform a commit:

```
CELERY_BEAT_SCHEDULE = {
    # Unconditionally commit all changes every 2 minutes
    "commit": {
        "task": "weblate.trans.tasks.commit_pending",
        # Omitting hours will honor per component settings,
        # otherwise components with no changes older than this
        # won't be committed
        "kwargs": {"hours": 0},
        # How frequently to execute the job in seconds
        "schedule": 120,
    }
}
```

2.9.5 Processing repository with scripts

The way to customize how Weblate interacts with the repository is *Eklentiler*. Consult *Eklentiden betikleri çalıştırma* for info on how to execute external scripts through add-ons.

2.9.6 Keeping translations same across components

Once you have multiple translation components, you might want to ensure that the same strings have same translation. This can be achieved at several levels.

Translation propagation

With *Çeviri yaymaya izin ver* enabled (what is the default, see *Component configuration*), all new translations are automatically done in all components with matching strings. Such translations are properly credited to currently translating user in all components.

Not: The translation propagation requires the key to be match for monolingual translation formats, so keep that in mind when creating translation keys.

Consistency check

The *Tutarsız* check fires whenever the strings are different. You can utilize this to review such differences manually and choose the right translation.

Otomatik çeviri

Automatic translation based on different components can be way to synchronize the translations across components. You can either trigger it manually (see *Otomatik çeviri*) or make it run automatically on repository update using add-on (see *Otomatik çeviri*).

2.10 Licensing translations

You can specify which license translations are contributed under. This is especially important to do if translations are open to the public, to stipulate what they can be used for.

You should specify *Component configuration* license info. You should avoid requiring a contributor license agreement, though it is possible.


2.10.1 License info


Upon specifying license info (license name and URL), this info is shown in the translation info section of the respective *Component configuration*.

Usually this is best place to post licensing info if no explicit consent is required. If your project or translation is not libre you most probably need prior consent.

2.10.2 Katılımcı sözleşmesi













If you specify a contributor license agreement, only users who have agreed to it will be able to contribute. This is a clearly visible step when accessing the translation:


Dashboard
Projects ▾
Languages ▾
Checks ▾


WeblateOrg / Language names
translated 95%

Contribution to this translation requires you to agree with a contributor agreement.
 View contributor agreement

Languages
Info
Alerts
Search
Insights ▾
Files ▾
Tools ▾
Manage ▾
Share ▾
Watching ▾

Language	Translated	Unfinished	Unfinished words	Checks	Suggestions	Comments
 Czech   GPL-3.0	✓					
 Hebrew   GPL-3.0	✓					
 Hungarian   GPL-3.0	81%	4	5			
 English   GPL-3.0	✓					


Start new translation


Powered by Weblate 4.13
 [About Weblate](#)
[Legal](#)
[Contact](#)
[Documentation](#)
[Donate to Weblate](#)

The entered text is formatted into paragraphs and external links can be included. HTML markup can not be used.

2.10.3 User licenses

Any user can review all translation licenses of all public projects on the instance from their profile:

 Weblate Dashboard Projects Languages Checks

 Your profile

Languages Preferences Notifications Account Profile **Licenses** Audit log API access

Licenses

Please pay attention to the licensing info, as this specifies how translations can be used.

By registering you agree to use your name and e-mail in the commits, and provide your contribution under the license defined by each localization project.

You have agreed to the following as a contributor:

- [WeblateOrg/Language names](#)

Licenses for individual translations

GNU General Public License v3.0 or later GPL-3.0 ⓘ

[WeblateOrg/WeblateOrg](#) [WeblateOrg/Djangojs](#) [WeblateOrg/Django](#) [WeblateOrg/Language names](#)

MIT License MIT ⓘ

[WeblateOrg/Android](#)

Powered by Weblate 4.13 About Weblate Legal Contact Documentation Donate to Weblate

2.11 Çeviri işleyişi

2.11.1 Öneri oylaması

Everyone can add suggestions by default, to be accepted by signed in users. Suggestion voting can be used to make use of a string when more than one signed-in user agrees, by setting up the *Component configuration* with *Suggestion voting* to turn on voting, and *Autoaccept suggestions* to set a threshold for accepted suggestions (this includes a vote from the user making the suggestion if it is cast).

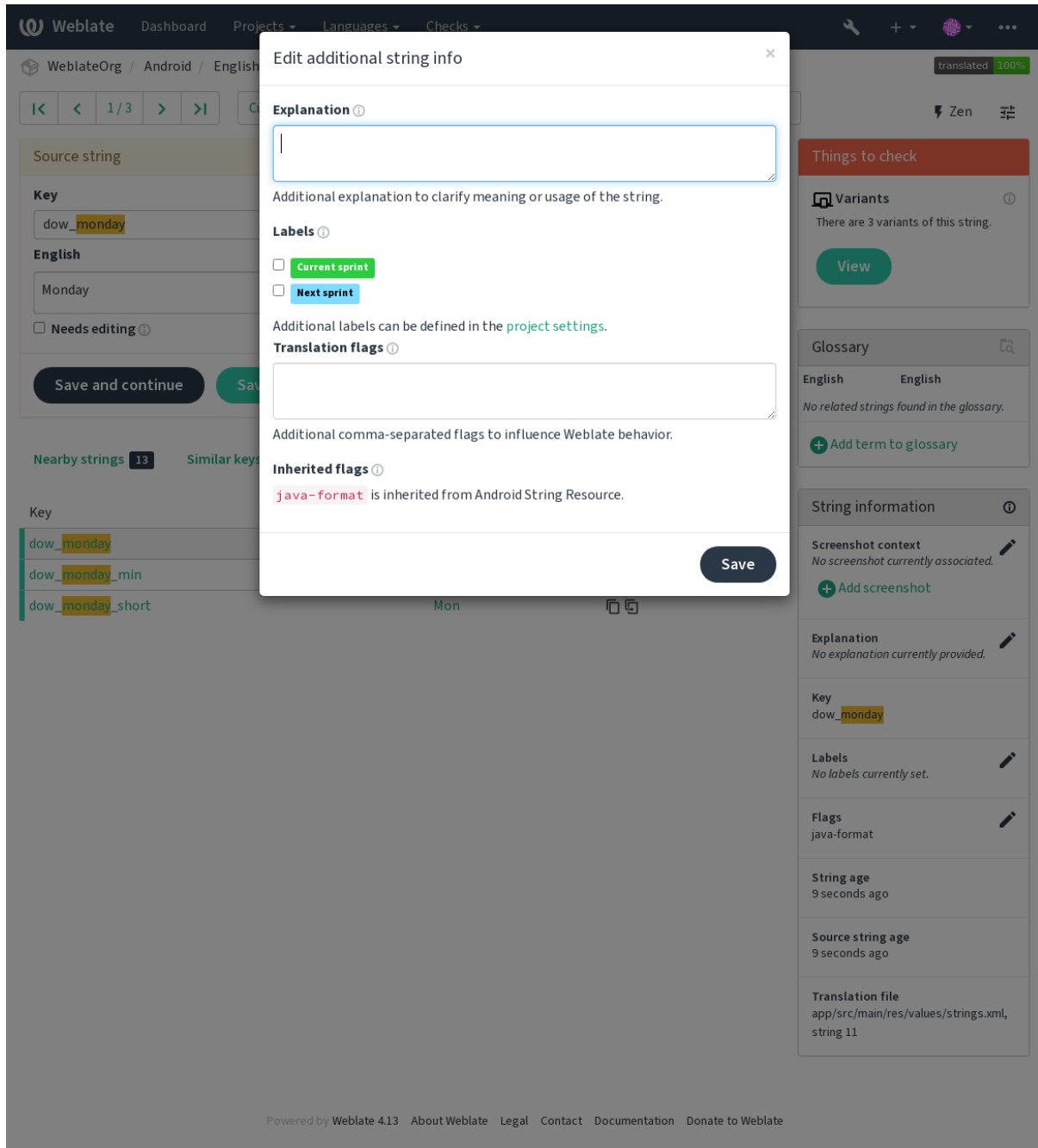
Not: Once automatic acceptance is set up, normal users lose the privilege to directly save translations or accept suggestions. This can be overridden with the *Edit string when suggestions are enforced permission*.

You can combine these with *access control* into one of the following setups:

- Users suggest and vote for suggestions and a limited group controls what is accepted. - Turn on voting. - Turn off automatic acceptance. - Don't let users save translations.
- Users suggest and vote for suggestions with automatic acceptance once the defined number of them agree. - Turn on voting. - Set the desired number of votes for automatic acceptance.
- Optional voting for suggestions. (Can optionally be used by users when they are unsure about a translation by making multiple suggestions.) - Only turn on voting.

2.11.2 Additional info on source strings

Enhance the translation process by adding additional info to the strings including explanations, string priorities, check flags and visual context. Some of that info may be extracted from the translation files and some may be added by editing the additional string info:



Access this directly from the translation interface by clicking the “Edit” icon next to *Screenshot context* or *Flags*.

WeblateOrg / Django / Czech / Translate
translated 96%

<

>

11 / 26

All strings ▾

Position and priority ▾

Zen

Translation

Explanation

Help text for automatic translation tool

English

Automatic translation via machine translation uses active machine translation engines to get the best possible translations and applies them in this project.

Czech

Automatický překlad prostřednictvím strojového překladu používá aktivní enginy strojového překladu pro získání nejlepších možných překladů a použije je na tento projekt.

☐ Needs editing ⓘ

Save and continue

Save and stay

Suggest

Skip

Glossary

English	Czech
machine translation	strojový překlad
project	projekt

+ Add term to glossary

String information

Screenshot context

No screenshot currently associated.

+ Add screenshot

Explanation

Help text for automatic translation tool

Labels

No labels currently set.

Flags

No flags currently set.

Source string location

weblate/templates/translation.html:212

String age

2 seconds ago

Source string age

2 seconds ago

Translation file

weblate/locale/cs/LC_MESSAGES/django.po, string 11

Nearby strings 26

Comments Automatic suggestions Other languages 4 History

Context	English	Czech	Actions
	Files	Soubory	
	Automatic translation	Automatický překlad	
	Add new translation string	Add new translation string	
	Translation status	Stav překladu	
	%{count}s word	%{count}s slovo	
	Other components	Další součásti	
	Translation file	Soubor s překladem	
	Download	Stáhnout	
	Browse all translation changes	Procházet všechny změny v překladu.	
	Automatic translation takes existing translations in this project and applies them to the current component. It can be used to push translations to a different branch, to fix inconsistent translations or to translate a new component using translation memory.	Automatický překlad použije stávající překlady v projektu na tuto součást. Může být užitečný pro sloučení překladů z jiné větve, opravu nekonzistentních překladů nebo překlad nové součásti pomocí překladové paměti.	
	Automatic translation via machine translation uses active machine translation engines to get the best possible translations and applies them in this project.	Automatický překlad prostřednictvím strojového překladu používá aktivní enginy strojového překladu pro získání nejlepších možných překladů a použije je na tento projekt.	
	You can add new translation string here, it will automatically appear in all translations.	Zde můžete přidat nový řetězec k překladu, automaticky se objeví ve všech jazycích.	
	The uploaded file will be merged with the current translation. In case you want to overwrite already translated strings, don't forget to enable it.	Nahráný soubor bude sloučen se stávajícími překlady. Pokud chcete přepsat již přeložené řetězce, nezapomeňte to povolit.	
	The uploaded file will be merged with the current translation.	Nahráný soubor bude sloučen se stávajícími překlady.	
	The fulltext search might not work properly as the fulltext index for this translation is not yet up to date.	Fulltextové vyhledávání nemusí fungovat správně, protože fulltextový index pro tento překlad ještě není plně zpracován.	
	Review	Kontrola	
	Review translations touched by other users.	Zkontrolovat překlady od ostatních uživatelů.	
	Start review	Začít kontrolu	
	Percent	Procenta	
	Total	Celkem	
	Failing check	Neúspěšných kontrol	
	Last activity	Poslední aktivity	
	Last change	Poslední změna	
	Last author	Poslední autor	
	What is %s?	Kolik to je?	
	Question for a mathematics-based CAPTCHA, the %s is an arithmetic problem		
	The string uses three dots (...) instead of an ellipsis character (...)		

Powered by Weblate 4.13 About Weblate Legal Contact Documentation Donate to Weblate

Strings prioritization

2.0 sürümünde geldi.

String priority can be changed to offer higher priority strings for translation earlier by using the `priority` flag.

İpucu: This can be used to order the flow of translation in a logical manner.

Ayrıca bakınız:

Quality checks

Çeviri işaretleri

2.4 sürümünde geldi.

3.3 sürümünde değişti: Previously called *Quality checks flags*, it no longer configures only checks.

Customization of quality checks and other Weblate behavior, see *Customizing behavior using flags*.

The string flags are also inherited from the *Çeviri işaretleri* at *Component configuration* and flags from the translation file (see *Supported file formats*).

Ayrıca bakınız:

Quality checks, Customizing behavior using flags

Açıklama

4.1 sürümünde değişti: In previous versions this has been called *Extra context*.

Use the explanation to clarify scope or usage of the translation. You can use Markdown to include links and other markup.

Visual context for strings

2.9 sürümünde geldi.

You can upload a screenshot showing a given source string in use within your program. This helps translators understand where it is used, and how it should be translated.

The uploaded screenshot is shown in the translation context sidebar:

Weblate
Dashboard
Projects ▾
Languages ▾
Checks ▾

WeblateOrg /
 Django /
 Czech /
 Translate

translated 96%

<<

<

11 / 26

>

>>

All strings ▾

Position and priority ▾

Zen

Translation

Explanation

Help text for automatic translation tool

English

Automatic translation via **machine translation** uses active **machine translation** engines to get the best possible translations and applies them in this **project**.

Czech

Automatický překlad prostřednictvím strojového překladu používá aktivní enginy strojového překladu pro získání nejlepších možných překladů a použije je na tento projekt.

☐ Needs editing ⓘ

169/1570 · 157

Save and continue

Save and stay

Suggest

Skip

Nearby strings 26
Comments
Automatic suggestions
Other languages 4
History

Translation memory

Search

Translation

Source

Origin

Similarity

Glossary

English

Czech

machine translation

strojový překlad

project

projekt

Add term to glossary

String information ⓘ

Screenshot context

Add screenshot

Explanation

Help text for automatic translation tool

Labels

No labels currently set.

Flags

No flags currently set.

Source string location

weblate/templates/translation.html:212

String age

5 seconds ago

Source string age

6 seconds ago

Translation file

weblate/locale/cs/LC_MESSAGES/django.po, string 11

In addition to *Additional info on source strings*, screenshots have a separate management interface under the *Tools* menu. Upload screenshots, assign them to source strings manually, or use optical character recognition to do so.

Once a screenshot is uploaded, this interface handles management and source string association:

Weblate
 Dashboard Projects Languages Checks

WeblateOrg / Django / Screenshots / Automatic translation

Screenshot has been uploaded, you can now assign it to source strings.

Assigned source strings

English	Location	Assigned screenshots	Actions
No matching strings found.			

Screenshot is shown to add visual context for all listed source strings.

Assign source strings

English	Location	Assigned screenshots	Actions
No matching strings found.			

Image

Source string

Hello, world!

One
 Orangutan has %d banana.

Other
 Orangutan has %d bananas.

Try Weblate at <http://demo.weblate.org/>!

Thank you for using Weblate.

Screenshot is shown to add visual context for all listed source strings.

Edit screenshot

Screenshot name

Image

Currently: [screenshots/screenshot.png](#)

Change:

No file chosen

Upload JPEG or PNG images up to 2000x2000 pixels.

Screenshot details

Created	now
Uploaded by	testuser
Language	English

Delete screenshot

Deleting screenshot will remove it from all associated source strings.

2.12 Checks and fixups

2.12.1 Custom automatic fixups

You can also implement your own automatic fixup in addition to the standard ones and include them in `AUTO-FIX_LIST`.

The automatic fixes are powerful, but can also cause damage; be careful when writing one.

For example, the following automatic fixup would replace every occurrence of the string `foo` in a translation with `bar`:

```
#
# Copyright © 2012-2022 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program. If not, see <https://www.gnu.org/licenses/>.
#

from django.utils.translation import gettext_lazy as _

from weblate.trans.autofixes.base import AutoFix

class ReplaceFooWithBar(AutoFix):
    """Replace foo with bar."""

    name = _("Foobar")

    def fix_single_target(self, target, source, unit):
        if "foo" in target:
            return target.replace("foo", "bar"), True
        return target, False
```

To install custom checks, provide a fully-qualified path to the Python class in the `AUTOFIX_LIST`, see *Custom quality checks, add-ons and auto-fixes*.

2.12.2 Customizing behavior using flags

You can fine-tune the Weblate behavior by using flags. This can be done on the source string level (see *Additional info on source strings*), or in the *Component configuration* (*Çeviri işaretleri*). Some file formats also allow to specify flags directly in the format (see *Supported file formats*).

The flags are comma-separated, the parameters are separated with colon. You can use quotes to include whitespace or special chars in the string. For example:

```
placeholders:"special:value":"other value", regex:.*
```

Both single and double quotes are accepted, special characters are being escaped using backslash:

```
placeholders:"quoted \"string\"":'single \'quoted\''
```

Here is a list of flags currently accepted:

rst-text

Treat a text as an reStructuredText document, affects *Değiştirilmemiş çeviri*.

dos-eol

Uses DOS end-of-line markers instead of Unix ones (`\r\n` instead of `\n`).

read-only

The string is read-only and should not be edited in Weblate, see *Salt okunur dizgiler*.

priority:N

Priority of the string. Higher priority strings are presented first for translation. The default priority is 100, the higher priority a string has, the earlier it is offered for translation.

max-length:N

Limit the maximal length for a string to N characters, see *Çevirinin en çok uzunluğu*.

xml-text

Treat text as XML document, affects *XML sözdizimi* and *XML işaretlemesi*.

font-family:NAME

Define font-family for rendering checks, see *Managing fonts*.

font-weight:WEIGHT

Define font-weight for rendering checks, see *Managing fonts*.

font-size:SIZE

Define font-size for rendering checks, see *Managing fonts*.

font-spacing:SPACING

Define letter spacing for rendering checks, see *Managing fonts*.

icu-flags:FLAGS

ICU MessageFormat nitelik denetiminin davranışını özelleştirmek için bayraklar tanımla.

icu-tag-prefix:PREFIX

Set a required prefix for XML tags for the *ICU MessageFormat* quality check.

placeholders:NAME:NAME2:...

Placeholder strings expected in translation, see *Yer tutucular*.

replacements:FROM:TO:FROM2:TO2...

Replacements to perform when checking resulting text parameters (for example in *Çeviri maksimum boyutu* or *Çevirinin en çok uzunluğu*). The typical use case for this is to expand placeables to ensure that the text fits even with long values, for example: `replacements:%s:"John Doe"`.

variants:SOURCE

Mark this string as a variant of string with matching source. See variants.

regex:REGEX

Çeviri dosyalarını eşleştirmek için düzenli ifade, bakınız *Düzenli ifade*.

forbidden

Indicates forbidden translation in a glossary, see *Yasaklı çeviriler*.

strict-same

Make “Unchanged translation” avoid using built-in words blacklist, see *Değiştirilmemiş çeviri*.

check-glossary

Sözlüğü takip etme nitelik denetimini etkinleştir.

angularjs-format

AngularJS ilişkilendirme dizgisi nitelik denetimini etkinleştir.

c-format

C biçimi nitelik denetimini etkinleştir.

c-sharp-format

C# biçimi nitelik denetimini etkinleştir.

es-format

ECMAScript şablon değişmezleri nitelik denetimini etkinleştir.

i18next-interpolation

Enable the *i18next ilişkilendirmesi* quality check.

icu-message-format

ICU MessageFormat nitelik denetimini etkinleştir.

java-format

Java biçimi nitelik denetimini etkinleştir.

java-messageformat

Java MessageFormat nitelik denetimini etkinleştir.

javascript-format

JavaScript biçimi nitelik denetimini etkinleştir.

lua-format

Lua biçimi nitelik denetimini etkinleştir.

object-pascal-format

Object Pascal biçimi nitelik denetimini etkinleştir.

percent-placeholders

Enable the *Yüzde yer tutucuları* quality check.

perl-format

Perl biçimi nitelik denetimini etkinleştir.

php-format

PHP biçimi nitelik denetimini etkinleştir.

python-brace-format

Enable the *Python ayracı biçimi* quality check.

python-format

Enable the *Python biçimi* quality check.

qt-format

Qt biçimi nitelik denetimini etkinleştir.

qt-plural-format

Qt çoğul biçimi nitelik denetimini etkinleştir.

ruby-format

Ruby biçimi nitelik denetimini etkinleştir.

scheme-format

Şema biçimi nitelik denetimini etkinleştir.

vue-format

Vue I18n biçimlendirmesi nitelik denetimini etkinleştir.

md-text

Treat text as a Markdown document. Enable *Markdown bağlantıları*, *Markdown başvuruları*, and *Markdown sözdizimi* quality checks.

case-insensitive

Adjust checks behavior to be case-insensitive. Currently affects only *Yer tutucular* quality check.

safe-html

Güvenli olmayan HTML nitelik denetimini etkinleştir.

url

The string should consist of only a URL. Enable the [URL](#) quality check.

ignore-all-checks

Ignore all quality checks.

ignore-bbcode

BBCode işaretlemesi nitelik denetimini atla.

ignore-duplicate

Art arda yinelenen sözcükler nitelik denetimini atla.

ignore-check-glossary

Sözlüğü takip etme nitelik denetimini atla.

ignore-double-space

Çift boşluk nitelik denetimini atla.

ignore-angularjs-format

AngularJS ilişkilendirme dizgisi nitelik denetimini atla.

ignore-c-format

C biçimi nitelik denetimini atla.

ignore-c-sharp-format

C# biçimi nitelik denetimini atla.

ignore-es-format

ECMAScript şablon değişmezleri nitelik denetimini atla.

ignore-i18next-interpolation

check-i18next-format nitelik denetimini atla.

ignore-icu-message-format

ICU MessageFormat nitelik denetimini atla.

ignore-java-format

Java biçimi nitelik denetimini atla.

ignore-java-messageformat

Java MessageFormat nitelik denetimini atla.

ignore-javascript-format

JavaScript biçimi nitelik denetimini atla.

ignore-lua-format

Lua biçimi nitelik denetimini atla.

ignore-object-pascal-format

Object Pascal biçimi nitelik denetimini atla.

ignore-percent-placeholders

Yüzde yer tutucuları nitelik denetimini atla.

ignore-perl-format

Perl biçimi nitelik denetimini atla.

ignore-php-format

PHP biçimi nitelik denetimini atla.

ignore-python-brace-format

Python ayracı biçimi nitelik denetimini atla.

ignore-python-format

Python biçimi nitelik denetimini atla.

ignore-qt-format

Qt biçimi nitelik denetimini atla.

ignore-qt-plural-format

Qt çoğul biçimi nitelik denetimini atla.

ignore-ruby-format

Ruby biçimi nitelik denetimini atla.

ignore-scheme-format

Şema biçimi nitelik denetimini atla.

ignore-vue-format

Vue I18n biçimlendirmesi nitelik denetimini atla.

ignore-translated

Çevrilmiş nitelik denetimini atla.

ignore-inconsistent

Tutarsız nitelik denetimini atla.

ignore-kashida

Kashida harfi kullanıldı nitelik denetimini atla.

ignore-md-link

Markdown bağlantıları nitelik denetimini atla.

ignore-md-reflink

Markdown başvuruları nitelik denetimini atla.

ignore-md-syntax

Markdown sözdizimi nitelik denetimini atla.

ignore-max-length

Çevirinin en çok uzunluğu nitelik denetimini atla.

ignore-max-size

Çeviri maksimum boyutu nitelik denetimini atla.

ignore-escaped-newline

\n uyuşmadı nitelik denetimini atla.

ignore-end-colon

İki nokta üst üste uyuşmadı nitelik denetimini atla.

ignore-end-ellipsis

Üç nokta uyuşmadı nitelik denetimini atla.

ignore-end-exclamation

Ünlem işareti uyuşmadı nitelik denetimini atla.

ignore-end-stop

Nokta işareti uyuşmadı nitelik denetimini atla.

ignore-end-question

Soru işareti uyuşmadı nitelik denetimini atla.

ignore-end-semicolon

Noktalı virgül uyuşmadı nitelik denetimini atla.

ignore-newline-count

Uyuşmayan satır sonları nitelik denetimini atla.

ignore-plurals

Eksik çoğul nitelik denetimini atla.

ignore-placeholders

Yer tutucular nitelik denetimini atla.

ignore-punctuation-spacing

Noktalama aralığı nitelik denetimini atla.

ignore-regex

Düzenli ifade nitelik denetimini atla.

ignore-same-plurals

Aynı çoğullar nitelik denetimini atla.

ignore-begin-newline

Baştaki yeni satır nitelik denetimini atla.

ignore-begin-space

Baştaki boşluk nitelik denetimini atla.

ignore-end-newline

Sondaki yeni satır nitelik denetimini atla.

ignore-end-space

Sondaki boşluk nitelik denetimini atla.

ignore-same

Değiştirilmemiş çeviri nitelik denetimini atla.

ignore-safe-html

Güvenli olmayan HTML nitelik denetimini atla.

ignore-url

URL nitelik denetimini atla.

ignore-xml-tags

XML işaretlemesi nitelik denetimini atla.

ignore-xml-invalid

XML söz dizimi nitelik denetimini atla.

ignore-zero-width-space

Sıfır genişlikli boşluk nitelik denetimini atla.

ignore-ellipsis

Üç nokta nitelik denetimini atla.

ignore-icu-message-format-syntax

ICU MessageFormat söz dizimi nitelik denetimini atla.

ignore-long-untranslated

Uzun süre çevrilmeyen nitelik denetimini atla.

ignore-multiple-failures

Birden çok kusur denetimi nitelik denetimini atla.

ignore-unnamed-format

Birden çok adsız değişken nitelik denetimini atla.

ignore-optional-plural

Çoğulsuz nitelik denetimini atla.

Not: Generally the rule is named `ignore-*` for any check, using its identifier, so you can use this even for your custom checks.

These flags are understood both in *Component configuration* settings, per source string settings and in the translation file itself (for example in GNU gettext).

2.12.3 Enforcing checks

3.11 sürümünde geldi.

You can configure a list of checks which can not be ignored by setting *Zorunlu denetimler* in *Component configuration*. Each listed check can not be dismissed in the user interface and any string failing this check is marked as *Needs editing* (see *Translation states*).

2.12.4 Managing fonts


3.7 sürümünde geldi.


İpucu: Fonts uploaded into Weblate are used purely for purposes of the *Çeviri maksimum boyutu* check, they do not have an effect in Weblate user interface.

The *Çeviri maksimum boyutu* check used to calculate dimensions of the rendered text needs font to be loaded into Weblate and selected using a translation flag (see *Customizing behavior using flags*).

Weblate font management tool in *Fonts* under the *Manage* menu of your translation project provides interface to upload and manage fonts. TrueType or OpenType fonts can be uploaded, set up font-groups and use those in the check.

The font-groups allow you to define different fonts for different languages, which is typically needed for non-latin languages:

 Weblate
 Dashboard Projects Languages Checks

 WeblateOrg / Font groups / default-font

Font group

Name	default-font		
Default font	Source Sans 3 Bold		
Japanese	language override	Droid Sans Fallback Regular	Remove
Korean	language override	Droid Sans Fallback Regular	Remove

Delete

Add language override

Language

Font

Save

Edit font group

Font group name

default-font

Identifier you will use in checks to select this font group. Avoid whitespaces and special characters.

Default font

Source Sans 3 Bold

Default font is used unless per language override matches.



Save

Powered by Weblate 4.13
 [About Weblate](#)
[Legal](#)
[Contact](#)
[Documentation](#)
[Donate to Weblate](#)

The font-groups are identified by name, which can not contain whitespace or special characters, so that it can be easily used in the check definition:

W Weblate

DashboardProjectsLanguagesChecks

+

WebOrg / Fonts

Font groupsFonts

Group name	Default font	Language overrides	
default-font	Source Sans 3 Bold	Japanese: Droid Sans Fallback Regular Korean: Droid Sans Fallback Regular	Edit

Add font group

Font group name

Identifier you will use in checks to select this font group. Avoid whitespaces and special characters.

Default font

Default font is used unless per language override matches.



Save

Powered by Weblate 4.13About WeblateLegalContactDocumentationDonate to Weblate


Font-family and style is automatically recognized after uploading them:

W Weblate

DashboardProjectsLanguagesChecks

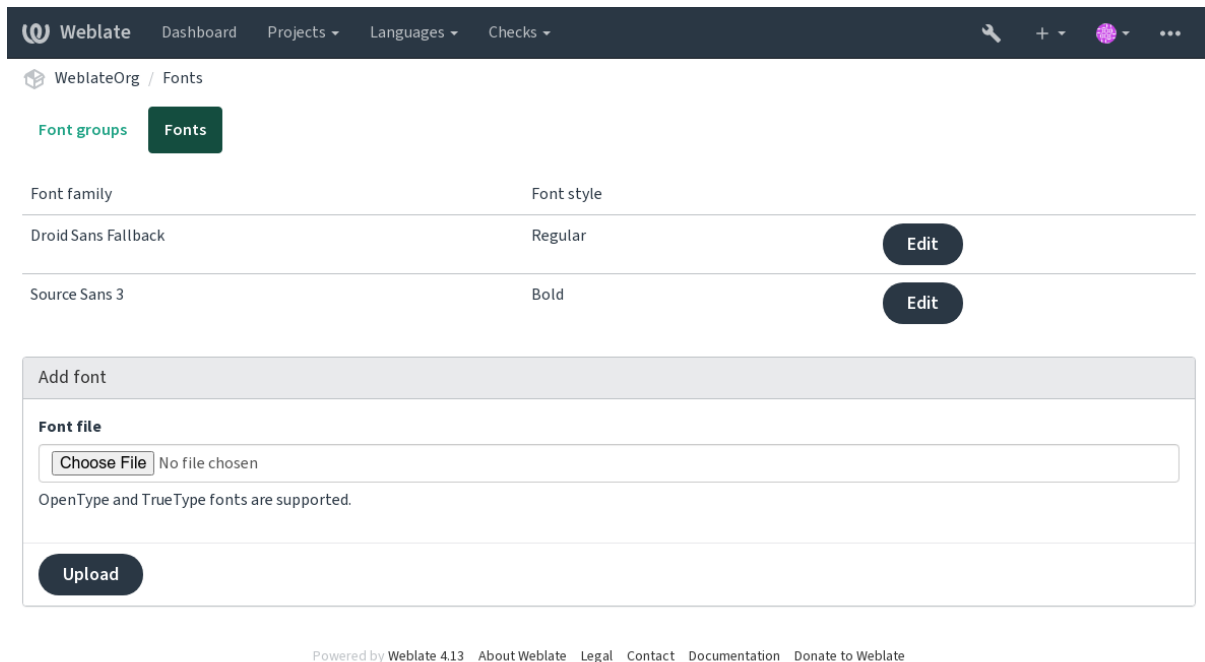
+

WebOrg / Fonts / Droid Sans Fallback Regular

Font	
Font family	Droid Sans Fallback
Font style	Regular
File size	3939852
Created	now
Uploaded by	 testuser
Used in groups	
Delete	

Powered by Weblate 4.13About WeblateLegalContactDocumentationDonate to Weblate

You can have a number of fonts loaded into Weblate:



To use the fonts for checking the string length, pass it the appropriate flags (see [Customizing behavior using flags](#)). You will probably need the following ones:

max-size:500

Defines maximal width in pixels.

font-family:ubuntu

Defines font group to use by specifying its identifier.

font-size:22

Defines font size in pixels.

2.12.5 Writing own checks

A wide range of quality checks are built-in, (see [Quality checks](#)), though they might not cover everything you want to check. The list of performed checks can be adjusted using [CHECK_LIST](#), and you can also add custom checks.

1. Subclass the `weblate.checks.Check`
2. Set a few attributes.
3. Implement either the `check` (if you want to deal with plurals in your code) or the `check_single` method (which does it for you).

Some examples:

To install custom checks, provide a fully-qualified path to the Python class in the [CHECK_LIST](#), see [Custom quality checks, add-ons and auto-fixes](#).

Checking translation text does not contain “foo”

This is a pretty simple check which just checks whether the translation is missing the string “foo”.

```
#
# Copyright © 2012-2022 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program. If not, see <https://www.gnu.org/licenses/>.
#
"""Simple quality check example."""

from django.utils.translation import gettext_lazy as _

from weblate.checks.base import TargetCheck

class FooCheck(TargetCheck):

    # Used as identifier for check, should be unique
    # Has to be shorter than 50 characters
    check_id = "foo"

    # Short name used to display failing check
    name = _("Foo check")

    # Description for failing check
    description = _("Your translation is foo")

    # Real check code
    def check_single(self, source, target, unit):
        return "foo" in target
```

Checking that Czech translation text plurals differ

Check using language info to verify the two plural forms in Czech language are not same.

```
#
# Copyright © 2012-2022 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program. If not, see <https://www.gnu.org/licenses/>.
#
"""Quality check example for Czech plurals."""

from django.utils.translation import gettext_lazy as _

from weblate.checks.base import TargetCheck

class PluralCzechCheck(TargetCheck):

    # Used as identifier for check, should be unique
    # Has to be shorter than 50 characters
    check_id = "foo"

    # Short name used to display failing check
    name = _("Foo check")

    # Description for failing check
    description = _("Your translation is foo")

    # Real check code
    def check_target_unit(self, sources, targets, unit):
        if self.is_language(unit, ("cs",)):
            return targets[1] == targets[2]
        return False


    def check_single(self, source, target, unit):
        """We don't check target strings here."""
        return False


```

2.13 Configuring automatic suggestions

4.13 sürümünde değişti: Prior to Weblate 4.13, the services were configured in the *Yapılandırma*.

The support for several machine translation and translation memory services is built-in. Each service can be turned on by the administrator for whole site or at the project settings:


Dashboard
Projects ▾
Languages ▾
Checks ▾


WeblateOrg / Automatic suggestions

Configured automatic suggestion services ⓘ

There are no services currently installed.

Available automatic suggestion services ⓘ

AWS ⓘ	Install
Amagama ⓘ	Install
Apertium APy ⓘ	Install
Baidu ⓘ	Install
DeepL ⓘ	Install
Glosbe ⓘ	Install
Google Translate ⓘ	Install
Google Translate API v3 ⓘ	Install
LibreTranslate ⓘ	Install
Microsoft Terminology ⓘ	Install
Microsoft Translator ⓘ	Install
ModernMT ⓘ	Install
MyMemory ⓘ	Install
Netease Sight ⓘ	Install
SAP Translation Hub ⓘ	Install
Weblate ⓘ	Install
Weblate Translation Memory ⓘ	Install
Yandex ⓘ	Install
Youdao Zhiyun ⓘ	Install
tmserver ⓘ	Install

Some services will ask for additional configuration during installation.

Powered by Weblate 4.13
 [About Weblate](#)
[Legal](#)
[Contact](#)
[Documentation](#)
[Donate to Weblate](#)

Not: They come subject to their terms of use, so ensure you are allowed to use them how you want.

The services translate from the source language as configured at *Component configuration*, see *Kaynak dil*.

Ayrıca bakınız:

Otomatik öneriler

2.13.1 Amagama

Service ID

amagama

Yapılandırma

This service has no configuration.

Special installation of *tmserver* run by the authors of Virtaal.

Ayrıca bakınız:

Installing amaGama, Amagama, amaGama Translation Memory

2.13.2 Apertium APy

Service ID

apertium-apy

Yapılandırma

url	API URL'si	
-----	------------	--

A libre software machine translation platform providing translations to a limited set of languages.

The recommended way to use Apertium is to run your own Apertium-APy server.

Ayrıca bakınız:

[Apertium website](#), [Apertium APy documentation](#)

2.13.3 AWS

3.1 sürümünde geldi.

Service ID

aws

Yapılandırma

key	Erişim anahtarı kimliği	
secret	API gizli anahtarı	
region	Bölge adı	

Amazon Translate is a neural machine translation service for translating text to and from English across a breadth of supported languages.

Ayrıca bakınız:

[Amazon Translate Documentation](#)

2.13.4 Baidu

3.2 sürümünde geldi.

Service ID

baidu

Yapılandırma

key	İstemci kimliği	
secret	İstemci gizli anahtarı	

Machine translation service provided by Baidu.

This service uses an API and you need to obtain an ID and API key from Baidu to use it.

Ayrıca bakınız:

[Baidu Translate API](#)

2.13.5 DeepL

2.20 sürümünde geldi.

Service ID

deepl

Yapılandırma

url	API URL'si	
key	API anahtarı	

DeepL is paid service providing good machine translation for a few languages. You need to purchase *DeepL API* subscription or you can use legacy *DeepL Pro (classic)* plan.

API URL to use with the DeepL service. At the time of writing, there is the v1 API as well as a free and a paid version of the v2 API.

`https://api.deepl.com/v2/` (default in Weblate)

Is meant for API usage on the paid plan, and the subscription is usage-based.

`https://api-free.deepl.com/v2/`

Is meant for API usage on the free plan, and the subscription is usage-based.

`https://api.deepl.com/v1/`

Is meant for CAT tools and is usable with a per-user subscription.

Previously Weblate was classified as a CAT tool by DeepL, so it was supposed to use the v1 API, but now is supposed to use the v2 API. Therefore it defaults to v2, and you can change it to v1 in case you have an existing CAT subscription and want Weblate to use that.

The easiest way to find out which one to use is to open an URL like the following in your browser:

https://api.deepl.com/v2/translate?text=Hello&target_lang=FR&auth_key=XXX

Replace the XXX with your auth_key. If you receive a JSON object which contains “Bonjour”, you have the correct URL; if not, try the other three.

Ayrıca bakınız:

[DeepL website](#), [DeepL pricing](#), [DeepL API documentation](#)

2.13.6 Glosbe

Service ID

glosbe

Yapılandırma

This service has no configuration.

Free dictionary and translation memory for almost every living language.

The API is gratis to use, but usage of the translations is subject to the license of the used data source. There is a limit of calls that may be done from one IP in a set period of time, to prevent abuse.

Ayrıca bakınız:

[Glosbe website](#)

2.13.7 Google Translate

Service ID

google-translate

Yapılandırma

key	API anahtarı	
-----	--------------	--

Machine translation service provided by Google.

This service uses the Google Translation API, and you need to obtain an API key and turn on billing in the Google API console.

Ayrıca bakınız:

[Google translate documentation](#)

2.13.8 Google Translate API v3

Service ID

google-translate-api-v3

Yapılandırma

credentials	Google Çeviri hizmeti hesap bilgileri	
project	Google Çeviri projesi	
location	Google Çeviri konumu	

Machine translation service provided by Google Cloud services.

Ayrıca bakınız:

[Google translate documentation](#), [Getting started with authentication on Google Cloud](#), [Creating Google Translate project](#), [Google Cloud App Engine locations](#)

2.13.9 LibreTranslate

4.7.1 sürümünde geldi.

Service ID

libretranslate

Yapılandırma

url	API URL'si	
key	API anahtarı	

LibreTranslate is a free and open-source service for machine translations. The public instance requires an API key, but LibreTranslate can be self-hosted and there are several mirrors available to use the API for free.

<https://libretranslate.com/> (official public instance)

Requires an API key to use outside of the website.

Ayrıca bakınız:

[LibreTranslate website](#), [LibreTranslate repository](#), [LibreTranslate mirrors](#)

2.13.10 Microsoft Terminology

2.19 sürümünde geldi.

Service ID

microsoft-terminology

Yapılandırma

This service has no configuration.

The Microsoft Terminology Service API allows you to programmatically access the terminology, definitions and user interface (UI) strings available in the Language Portal through a web service.

Ayrıca bakınız:

[Microsoft Terminology Service API](#)

2.13.11 Microsoft Translator

2.10 sürümünde geldi.

Service ID

microsoft-translator

Yapılandırma

key	API anahtarı	
endpoint	Uygu- lama noktası URL'si	
base_url	Uygu- lama tabanlı URL	Kullanılabilir seçenekler: api.cognitive.microsofttranslator.com – Global (non-regional) api-apc.cognitive.microsofttranslator.com – Asia Pacific api-eur.cognitive.microsofttranslator.com – Europe api-nam.cognitive.microsofttranslator.com – North America api.translator.azure.cn – China
region	Uygu- lama bölgesi	

Machine translation service provided by Microsoft in Azure portal as a one of Cognitive Services.

Weblate implements Translator API V3.

Translator Text API V2

The key you use with Translator API V2 can be used with API 3.

Translator Text API V3

You need to register at Azure portal and use the key you obtain there. With new Azure keys, you also need to set `region` to locale of your service.

İpucu: For Azure China, please use your endpoint from the Azure Portal.

Ayrıca bakınız:

Cognitive Services - Text Translation API, Microsoft Azure Portal, Base URLs, “Authenticating with a Multi-service resource” “Authenticating with an access token” section

2.13.12 ModernMT

4.2 sürümünde geldi.

Service ID

modernmt

Yapılandırma

url	API URL'si	
key	API anahtarı	

Ayrıca bakınız:

[ModernMT API](#),

2.13.13 MyMemory

Service ID

mymemory

Yapılandırma

email	İletişim e-postası	
username	Kullanıcı adı	
key	API anahtarı	

Huge translation memory with machine translation.

Free, anonymous usage is currently limited to 100 requests/day, or to 1000 requests/day when you provide a contact e-mail address in `email`. You can also ask them for more.

Ayrıca bakınız:

[MyMemory website](#)

2.13.14 Netease Sight

3.3 sürümünde geldi.

Service ID

netease-sight

Yapılandırma

key	İstemci kimliği	
secret	İstemci gizli anahtarı	

NetEase tarafından sağlanan makine çevirisi hizmeti.

This service uses an API, and you need to obtain key and secret from NetEase.

Ayrıca bakınız:

[NetEase Sight Translation Platform](#)

2.13.15 SAP Translation Hub

Service ID

sap-translation-hub

Yapılandırma

url	API URL'si	
key	API anahtarı	
username	SAP kullanıcı adı	
password	SAP parolası	
enable_mt	Makine çevirisini etkinleştir	

Machine translation service provided by SAP.

You need to have a SAP account (and the SAP Translation Hub enabled in the SAP Cloud Platform) to use this service.

You can also configure whether to also use machine translation services, in addition to the term database.

Not: To access the Sandbox API, you need to set `url` and `key`.

To access the productive API, you need to set `url`, `username` and `password`.

Ayrıca bakınız:

[SAP Translation Hub API](#)

2.13.16 tmserver

Service ID

tmserver

Yapılandırma

url	API URL'si	
-----	------------	--

You can run your own translation memory server by using the one bundled with Translate-toolkit and let Weblate talk to it. You can also use it with an amaGama server, which is an enhanced version of tmserver.

1. First you will want to import some data to the translation memory:

```
build_tmdb -d /var/lib/tm/db -s en -t cs locale/cs/LC_MESSAGES/django.po
build_tmdb -d /var/lib/tm/db -s en -t de locale/de/LC_MESSAGES/django.po
build_tmdb -d /var/lib/tm/db -s en -t fr locale/fr/LC_MESSAGES/django.po
```

2. Start tmserver to listen to your requests:

```
tmserver -d /var/lib/tm/db
```

3. Configure Weblate to talk to it, the default URL is `http://localhost:8888/tmserver/`.

Ayrıca bakınız:

[tmserver](#) [Installing amaGama](#), [Amagama](#), [Amagama Translation Memory](#)

2.13.17 Weblate

Service ID

weblate

Yapılandırma

This service has no configuration.

Weblate machine translation service can provide translations for strings that are already translated inside Weblate. It looks for exact matches in the existing strings.

2.13.18 Weblate Translation Memory

2.20 sürümünde geldi.

Service ID

`weblate-translation-memory`

Yapılandırma

This service has no configuration.

Use [Çeviri Belleği](#) as a machine translation service. Any string that has been translated in past (or uploaded to the translation memory) can be translated in this way.

2.13.19 Yandex

Service ID

`yandex`

Yapılandırma

key	API anahtarı	
-----	--------------	--

Machine translation service provided by Yandex.

This service uses a Translation API, and you need to obtain an API key from Yandex.

Ayrıca bakınız:

[Yandex Translate API](#), Powered by [Yandex.Translate](#)

2.13.20 Youdao Zhiyun

3.2 sürümünde geldi.

Service ID

`youdao-zhiyun`

Yapılandırma

key	İstemci kimliği	
secret	İstemci gizli anahtarı	

Machine translation service provided by Youdao.

This service uses an API, and you need to obtain an ID and an API key from Youdao.

Ayrıca bakınız:

[Youdao Zhiyun Natural Language Translation Service](#)

2.13.21 Custom machine translation

You can also implement your own machine translation services using a few lines of Python code. This example implements machine translation in a fixed list of languages using dictionary Python module:

```
#
# Copyright © 2012–2022 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program. If not, see <https://www.gnu.org/licenses/>.
#
"""Machine translation example."""

import dictionary

from weblate.machinery.base import MachineTranslation

class SampleTranslation(MachineTranslation):
    """Sample machine translation interface."""

    name = "Sample"

    def download_languages(self):
        """Return list of languages your machine translation supports."""
        return {"cs"}

    def download_translations(
        self,
        source,
        language,
        text: str,
        unit,
        user,
        search: bool,
        threshold: int = 75,
    ):
        """Return tuple with translations."""
        for t in dictionary.translate(text):
            yield {"text": t, "quality": 100, "service": self.name, "source": text}
```


You can list your own class in `WEBLATE_MACHINERY` and Weblate will start using that.

2.14 Eklentiler

2.19 sürümünde geldi.

Eklentiler, çeviri iş akışını özelleştirmenin ve otomatikleştirmenin yollarını sağlar. Yöneticiler, her bir çeviri bileşenin *Yönet* ↓ *Eklentiler* menüsünden eklenti ekleyebilir ve onları yönetebilir.

İpucu: You can also configure add-ons using [API](#), `DEFAULT_ADDONS`, or `install_addon`.



[Weblate](#)
[Dashboard](#)
[Projects](#)
[Languages](#)
[Checks](#)

[WeblateOrg](#) / [Language names](#) / [Add-ons](#)

Installed add-ons ⓘ


There are no add-ons currently installed.

Available add-ons ⓘ


Automatic translation ⓘ

Automatically translates strings using machine translation or other components.


Install


Add missing languages ⓘ

Ensures a consistent set of languages is used for all components within a project.

project wide


Install


Component discovery ⓘ

Automatically adds or removes project components based on file changes in the version control system.


repository wide

Install


Bulk edit ⓘ


Bulkedit flags, labels, or states of strings.

Install


Statistics generator ⓘ


Generates a file containing detailed info about the translation status.

Install


Prefill translation with source ⓘ


Fills in translation strings with source string.

Install


Pseudolocale generation ⓘ


Generates a translation by adding prefix and suffix to source strings automatically.

Install


Contributors in comment ⓘ


Updates the comment part of the PO file header to include contributor names and years of contributions.

Install


Customize gettext output ⓘ


Allows customization of gettext output behavior, for example line wrapping.

Install


Generate MO files ⓘ


Automatically generates a MO file for every changed PO file.

Install


Update PO files to match POT (msgmerge) ⓘ

Updates all PO files (as configured by "File mask") to match the POT file (as configured by "Template for new translations") using msgmerge.


Install


Squash Git commits ⓘ

Squash Git commits prior to pushing changes.

repository wide


Install


Stale comment removal ⓘ

Set a timeframe for removal of comments.

project wide

Install


Stale suggestion removal ⓘ

Set a timeframe for removal of suggestions.

project wide

Install

Some add-ons will ask for additional configuration during installation.

2.14.1 Yerleşik eklentiler

Otomatik çeviri

3.9 sürümünde geldi.

Eklenti kimliği

`weblate.autotranslate.autotranslate`

Yapılandırma

<code>mode</code>	Otomatik çeviri kipi	Kullanılabilir seçenekler: suggest – Öneri olarak ekle translate – Çeviri olarak ekle fuzzy – Düzenleme ihtiyacı var olarak ekle
<code>filter</code>	Aramaya süzgeci	Lütfen tüm dizgileri çevirmenin varolan tüm çevirileri yoksayacağını unutmayın. Kullanılabilir seçenekler: all – Tüm dizgiler nottranslated – Çevrilmemiş dizgiler todo – Tamamlanmamış dizgiler fuzzy – Düzenleme için işaretlenen dizgiler check:inconsistent – Kusur denetimi: Tutarsız
<code>auto</code>	Otomatik çeviri kaynağı	Kullanılabilir seçenekler: others – Diğer çeviri bileşenleri mt – Makine çevirisi
<code>component</code>	Bileşenler	Kaynak olarak kullanılacak bileşeni girin, şu anki projedeki tüm bileşenleri kullanmak için boş bırakın.
<code>engine</code>	Makine çevirisi motorları	
<code>threshold</code>	Skor eşiği	

Triggers

bileşen güncellemesi, günlük

Makine çevirisini veya diğer bileşenleri kullanarak dizgileri kendiliğinden çevirir.

Ne zaman tetiklenir:

- Bileşende yeni dizgiler görüldüğünde.
- Once in a month for every component, this can be configured using `BACKGROUND_TASKS`.

Ayrıca bakınız:

Otomatik çeviri, Keeping translations same across components

JavaScript yerelleştirme CDN'i

4.2 sürümünde geldi.

Eklenti kimliği

`weblate.cdn.cdnjs`

Yapılandırma

<code>thres-</code> <code>hold</code>	Çeviri eşiği	Çevirilerin dahil edilmesi için eşik.
<code>css_select</code>	CSS seçici	Yerelleştirilebilir öğeleri algılamak için CSS seçici.
<code>co-</code> <code>okie_name</code>	Dil tanımlama bilgi- si adı	Dil tercihini saklayan tanımlama bilgisi adı.
<code>files</code>	HTML dosyaların- dan dizgileri çıkar	Çevrilebilir dizgiler için ayrıştırılacak şu anki depodaki veya uzak URL'lerdeki dosya adlarının listesi.

Triggers

`daily`, `repository post-commit`, `repository post-update`

JavaScript veya HTML yerelleştirmesinde kullanılmak üzere içerik dağıtım ağına çevirileri yayınlar.

Statik HTML sayfalarını yerelleştirmek veya JavaScript kodunda yerelleştirmeyi yüklemek için kullanılabilir.

Bileşeniniz için, bunları yerelleştirmek için HTML sayfalarına ekleyebileceğiniz benzersiz bir URL oluşturur. Daha fazla ayrıntı için bkz: `weblate-cdn`.

Ayrıca bakınız:

`cdn-addon-config`, `weblate-cdn`, `cdn-addon-extract`, `cdn-addon-html`

Boş dizgileri kaldır

4.4 sürümünde geldi.

Eklenti kimliği

`weblate.cleanup.blank`

Yapılandırma

Bu eklentinin yapılandırması yok.

Triggers

`repository post-commit`, `repository post-update`

Çeviri dosyalarından çevirisi olmayan dizgileri kaldırır.

Çeviri dosyalarında boş dizgilerin görünmemesi için bunu kullanın (örneğin yerelleştirme kütüphaneniz kaynak dizgiyi kullanmak yerine bunları boş dizgiler olarak görüntülüyorsa).

Ayrıca bakınız:

Does Weblate update translation files besides translations?

Çeviri dosyaslarını temizle

Eklenti kimliği

`weblate.cleanup.generic`

Yapılandırma

Bu eklentinin yapılandırması yok.

Triggers

repository pre-commit, repository post-update

Tüm çeviri dosyalarını tek dilli taban dosyayla eşleşmesi için güncelle. Çoğu dosya biçimi için bu, taban dosyada artık bulunmayan eski çeviri anahtarlarını kaldırmak anlamına gelir.

Ayrıca bakınız:

Does Weblate update translation files besides translations?

Eksik dilleri ekle

Eklenti kimliği

`weblate.consistency.languages`

Yapılandırma

Bu eklentinin yapılandırması yok.

Triggers

daily, repository post-add

Proje içindeki tüm bileşenler için tutarlı bir dil kümesinin kullanılmasını sağlar.

Eksik diller her 24 saatte bir ve Weblate'e yeni diller eklendiğinde denetlenir.

Diğerlerinden farklı olarak, bu eklenti tüm projeyi etkiler.

İpucu: Yeni eklenen dizgileri *Otomatik çeviri* ile otomatik olarak çevirin.

Bileşen keşfi

Ekleni kimliği

weblate.discovery.discovery

Yapılandırma

match	Çeviri dosyalarına karşı eşleştirmek için düzenli ifade	
file_format	Dosya biçimi	
name_template	Bileşen adını özelleştir	
base_file_template	Tek dil taban dosya adını tanımla	İki dilli çeviri dosyaları için boş bırakın.
new_base_file_template	Yeni çeviriler için taban dosyayı tanımla	Yeni çevirileri oluşturmak için kullanılan dosyanın dosya adı. Gettext için .pot dosyasını seçin.
intermediate_template	Ara dil dosyası	Ara çeviri dosyasının dosya adı. Çoğu durumda bu geliştiriciler tarafından sağlanan bir çeviri dosyasıdır ve gerçek kaynak dizgiler oluştururken kullanılır.
language_regex	Dil süzgeci	Dosya maskesi taranırken çeviri dosyalarını süzmek için kullanılan düzenli ifade.
copy_addons	Ana bileşenden yeni oluşturulanlara ekleri çoğalt	
remove	Var olmayan dosyalar için bileşenleri kaldır	
confirm	Yukarıdaki eşleşmelerin doğru görüldüğünü onaylıyorum	

Triggers

depo güncellenmesinden sonra

Sürüm denetim sistemindeki dosya değişikliklerine göre proje bileşenlerini kendiliğinden ekler veya kaldırır.

Triggered each time the VCS is updated, and otherwise similar to the `import_project` management command. This way you can track multiple translation components within one VCS.

The matching is done using regular expressions enabling complex configuration, but some knowledge is required to do so. Some examples for common use cases can be found in the add-on help section.

Once you hit *Save*, a preview of matching components will be presented, from where you can check whether the configuration actually matches your needs:

Weblate

Dashboard

Projects

Languages

Checks

+

WeblateOrg

Language names

Add-ons

Component discovery

Configure add-on

Please review and confirm the matched components.

Component

Matched files

The following components would be created

Djangojs

weblate/locale/he/LC_MESSAGES/djangojs.po (he)

weblate/locale/cs/LC_MESSAGES/djangojs.po (cs)

weblate/locale/hu/LC_MESSAGES/djangojs.po (hu)

Django

weblate/locale/hu/LC_MESSAGES/django.po (hu)

weblate/locale/cs/LC_MESSAGES/django.po (cs)

weblate/locale/he/LC_MESSAGES/django.po (he)

☐ I confirm the above matches look correct

Regular expression to match translation files against

weblate/locale/(?P<language>[^/]+)/LC_MESSAGES/(?P<component>[^\.]*)\.po

File format

gettext PO file

Customize the component name

{{ component|title }}

Define the monolingual base filename

Leave empty for bilingual translation files.

Define the base file for new translations

weblate/locale/{{ component }}.pot

Filename of file used for creating new translations. For gettext choose .pot file.

Language filter

^(cs|he|hu)\$

Regular expression to filter translation files against when scanning for file mask.

☒ Clone add-ons from the main component to the newly created ones

☐ Remove components for inexistent files

The regular expression to match translation files has to contain two named groups to match component and language, some examples:

Regular expression	Example matched files	Description
<code>(?P<language>[^/.]*)/(?P<component>[^\.]*)\.po</code>	<code>cs/application.po</code> <code>cs/website.po</code> <code>de/application.po</code> <code>de/website.po</code>	One folder per language containing translation files for components.
<code>locale/(?P<language>[^/.]*)/LC_MESSAGES/(?P<component>[^\.]*)\.po</code>	<code>locale/cs/LC_MESSAGES/application.po</code> <code>locale/cs/LC_MESSAGES/website.po</code> <code>locale/de/LC_MESSAGES/application.po</code> <code>locale/de/LC_MESSAGES/website.po</code>	Usual structure for storing gettext PO files.
<code>src/locale/(?P<component>[^\.]*)\. (?P<language>[^/.]*)\.po</code>	<code>src/locale/application.cs.po</code> <code>src/locale/website.cs.po</code> <code>src/locale/application.de.po</code> <code>src/locale/website.de.po</code>	Using both component and language name within filename.
<code>locale/(?P<language>[^/.]*)/(?P<component>[^\.]*)/(?P=language)\.po</code>	<code>locale/cs/application/cs.po</code> <code>locale/cs/website/cs.po</code> <code>locale/de/application/de.po</code> <code>locale/de/website/de.po</code>	Using language in both path and filename.
<code>res/values-(?P<language>[^/.]*)/strings-(?P<component>[^\.]*)\.xml</code>	<code>res/values-cs/strings-about.xml</code> <code>res/values-cs/strings-help.xml</code> <code>res/values-de/strings-about.xml</code> <code>res/values-de/strings-help.xml</code>	Android resource strings, split into several files.

You can use Django template markup in both component name and the monolingual base filename, for example:

`{{ component }}`

Component filename match

`{{ component|title }}`

Component filename with upper case first letter

Save

Powered by Weblate 4.13

About Weblate

Legal

Contact

Documentation

Donate to Weblate

İpucu: Component discovery add-on uses *Weblate internal URLs*. It's a convenient way to share VCS setup between multiple components. Linked components use the local repository of the main component set up by filling `weblate://project/main-component` into the *Kaynak kod deposu* field (in *Manage* ↓ *Settings* ↓ *Version control system*) of each respective component. This saves time with configuration and system resources too.

Ayrıca bakınız:

Template markup

Toplu düzenleme

3.11 sürümünde geldi.

Eklenti kimliği

`weblate.flags.bulk`

Yapılandırma

q	Sorgu	
state	Ayarlanacak durum	Kullanılabilir seçenekler: -1 – Değiştirmeyin 10 – Düzenleme gerekli 20 – Çevrildi 30 – Onaylandı
add_flags	Eklenecek çeviri işaretleri	
remove_flags	Kaldırılacak çeviri işaretleri	
add_labels	Eklenecek etiketler	
remove_labels	Kaldırılacak etiketler	

Triggers

bileşen güncellemesi

Dizgilerin işaretlerini, etiketlerini veya durumlarını toplu düzenleyin.

`NOT has:label` arama sorgusuyla başlayarak etiketlemeyi otomatikleştirin ve tüm dizgiler gerekli tüm etiketlere sahip olana kadar etiketler ekleyin. Weblate üst verileri için diğer otomatik işlemler de yapılabilir.

Örnekler:

Tablo 5: Yeni dizgileri otomatik olarak etiketle

Arama sorgusu	<code>NOT has:label</code>
Eklenecek etiketler	<code>en son</code>

Tablo 6: Tüm Uygulama mağazası üst veri dosyaları değişiklik günlüğü dizgilerini salt okunur olarak işaretler

Arama sorgusu	<code>language:en AND key:changelogs/</code>
Eklenecek çeviri işaretleri	<code>read-only</code>

Ayrıca bakınız:

Toplu düzenleme, Customizing behavior using flags, labels

Değiştirilmemiş çevirileri “Düzenleme gerekli” olarak işaretle

3.1 sürümünde geldi.

Eklenti kimliği

`weblate.flags.same_edit`

Yapılandırma

Bu eklentinin yapılandırması yok.

Triggers

birim oluşturduktan sonra

VCS’den yeni bir çevrilebilir dizgi içe aktarıldığında ve bir kaynak dizgiyle eşleştğinde, Weblate’te düzenleme gerekli olarak işaretlenir. Özellikle çevrilmemiş dizgiler için kaynak dizgileri içeren dosya biçimleri için kullanışlıdır.

İpucu: Ayrıca, [Çeviri işaretleri](#) için `strict-same` bayrağı ekleyerek *Değiştirilmemiş çeviri* denetimini sıkılaştırmak isteyebilirsiniz.

Ayrıca bakınız:

[Translation states](#)

Yeni kaynak dizgileri “Düzenleme gerekli” olarak işaretle

Eklenti kimliği

`weblate.flags.source_edit`

Yapılandırma

Bu eklentinin yapılandırması yok.

Triggers

birim oluşturduktan sonra

Yeni bir kaynak dizgi VCS’den içe her aktarıldığında, Weblate’de düzenleme gerekli olarak işaretlenir. Bu yolla geliştiriciler tarafından yazılan kaynak dizgiler kolayca süzülür ve düzenleyebilirsiniz.

Ayrıca bakınız:

[Translation states](#)

Yeni çevirileri “Düzenleme gerekli” olarak işaretle

Eklenti kimliği

`weblate.flags.target_edit`

Yapılandırma

Bu eklentinin yapılandırması yok.

Triggers

birim oluşturduktan sonra

Yeni bir çevrilebilir dizgi, VCS’den içeri her aktarıldığında, Weblate’de düzenleme gerekli olarak işaretlenir. Bu yolla geliştiricilerin oluşturduğu çevirileri kolayca süzer ve düzenleyebilirsiniz.

Ayrıca bakınız:

[Translation states](#)

İstatistik oluşturucu

Eklenti kimliği

weblate.generate.generate

Yapılandırma

filename	Oluşturulan dosyanın adı	
template	Oluşturulan dosyanın içeriği	

Triggers

depo işlenmesinden önce

Çeviri durumu hakkında ayrıntılı bilgi içeren bir dosya oluşturur.

Hem dosya adında hem de içerikte bir Django şablonu kullanabilirsiniz, ayrıntılı bir işaretleme açıklaması için [Template markup](#) bölümüne bakın.

Örneğin, her çeviri için bir özet dosyası oluşturma:

Oluşturulan dosyanın adı

locale/{{ language_code }}.json

İçerik

```
{
  "language": "{{ language_code }}",
  "strings": "{{ stats.all }}",
  "translated": "{{ stats.translated }}",
  "last_changed": "{{ stats.last_changed }}",
  "last_author": "{{ stats.last_author }}"
}
```

Ayrıca bakınız:

[Template markup](#)

Çeviriyi kaynakla önceden doldur

4.11 sürümünde geldi.

Eklenti kimliği

weblate.generate.prefill

Yapılandırma

Bu eklentinin yapılandırması yok.

Triggers

bileşen güncellemesi, günlük

Çeviri dizgilerini kaynak dizgiyle doldurur.

Bileşendeki tüm çevrilmemiş dizgiler kaynak dizgiyle doldurulacak ve düzenleme gerekli olarak işaretlenecektir. Çeviri dosyalarında boş dizgi kabul edilmediğinde bunu kullanın.

Sözde yerel oluşturma

4.5 sürümünde geldi.

Eklenti kimliği

`weblate.generate.pseudolocale`

Yapılandırma

<code>source</code>	Kaynak dizgiler	
<code>target</code>	Hedef çeviri	Bu çevirideki tüm dizgilerin üzerine yazılacaktır
<code>prefix</code>	Sabit dizgi öneki	
<code>var_prefix</code>	Değişken dizgi öneki	
<code>suffix</code>	Sabit dizgi soneki	
<code>var_suffix</code>	Değişken dizgi soneki	
<code>var_multiplier</code>	Değişken kısım çarpanı	Kaynak dizginin uzunluğuna bağlı olarak değişken kısmın kaç kez tekrarlanacağı.

Triggers

bileşen güncellemesi, günlük

Kaynak dizgilere kendiliğinden ön ek ve son ek ekleyerek çeviri oluşturur.

Pseudolocales are useful to find strings that are not prepared for localization. This is done by altering all translatable source strings to make it easy to spot unaltered strings when running the application in the pseudolocale language.

Finding strings whose localized counterparts might not fit the layout is also possible.

Using the variable parts makes it possible to look for strings which might not fit into the user interface after the localization - it extends the text based on the source string length. The variable parts are repeated by length of the text multiplied by the multiplier. For example `Hello world` with variable suffix `_` and variable multiplier of 1 becomes `Hello world_____` - the suffix is repeated once for each character in the source string.

Dizgiler aşağıdaki şekil kullanılarak oluşturulacaktır:

Fixed string prefix Variable string prefix Source string Variable string suffix Fixed string suffix

İpucu: You can use real languages for testing, but there are dedicated pseudolocales available in Weblate - *en_XA* and *ar_XB*.

İpucu: You can use this add-on to start translation to a new locale of an existing language or similar language. Once you add the translation to the component, follow to the add-on. *Example:* If you have *fr* and want to start *fr_CA* translation, simply set *fr* as the source, *fr_CA* as the target, and leave the prefix and suffix blank.

Uninstall the add-on once you have the new translation filled to prevent Weblate from changing the translations made after the copying.

Yorum içindeki katılımcılar

Ekenti kimliği

weblate.gettext.authors

Yapılandırma

Bu eklentinin yapılandırması yok.

Triggers

depo işlenmesinden önce

PO dosyası başlığının yorum bölümünü, katılımcı adlarını ve katkı yıllarını içerecek şekilde günceller.

PO dosya başlığı şöyle görünecektir:

```
# Michal Čihař <michal@cihar.com>, 2012, 2018, 2019, 2020.
# Pavel Borecki <pavel@example.com>, 2018, 2019.
# Filip Hron <filip@example.com>, 2018, 2019.
# anonymous <noreply@weblate.org>, 2019.
```

“configure” dosyasındaki ALL_LINGUAS değişkenini güncelle

Ekenti kimliği

weblate.gettext.configure

Yapılandırma

Bu eklentinin yapılandırması yok.

Triggers

depo eklenmesinden sonra, günlük

Yeni bir çeviri eklendiğinde ALL_LINGUAS değişkenini configure, configure.in veya herhangi bir configure.ac dosyasında günceller.

Gettext çıktısını özelleştir

Ekenti kimliği

weblate.gettext.customize

Yapılandırma

width	Uzun	Varsayılan olarak gettext satırları 77 karakterde ve yeni satırlarda kaydırır, –no-wrap parametresi ile kaydırma yalnızca yeni satırlarda yapılır.
	satır-ları	Kullanılabilir seçenekler:
	kay-dır	77 – Satırları, 77 karakterde ve yeni satırlarda kaydır
		65535 – Satırları yalnızca yeni satırlarda kaydır
		–1 – Satır kaydırma yok

Triggers

depolama yüklendikten sonra

Gettext çıktısı davranışının özelleştirilmesine izin verir, örneğin satır kaydırma.

Aşağıdaki seçenekleri sunar:

- Satırları, 77 karakterde ve yeni satırlarda kaydır
- Satırları yalnızca yeni satırlarda kaydır
- Satır kaydırma yok

Not: Varsayılan olarak gettext satırları 77 karakterde ve yeni satırlarda kaydırır. `--no-wrap` parametresi ile kaydırma yalnızca yeni satırlarda yapılır.

LINGUAS dosyasını güncelle

Eklenti kimliği

`weblate.gettext.linguas`

Yapılandırma

Bu eklentinin yapılandırması yok.

Triggers

depo eklenmesinden sonra, günlük

Yeni bir çeviri eklendiğinde LINGUAS dosyasını günceller.

MO dosyaları üret

Eklenti kimliği

`weblate.gettext.mo`

Yapılandırma

path	Oluşturulan MO dosyasının yolu	Belirtilmemişse, PO dosyasının konumu kullanılır.
------	--------------------------------	---

Triggers

depo işlenmesinden önce

Her değiştirilen PO dosyası için kendiliğinden MO dosyası üretir.

The location of the generated MO file can be customized and the field for it uses *Template markup*.

POT ile eşleşmesi için PO dosyalarını güncelle (msgmerge)

Eklenti kimliği

`weblate.gettext.msgmerge`

Yapılandırma

previous	Çevrilen dizgilerin önceki msgid'lerini sakla	
no_location	Çevrilen dizgilerin konumlarını kaldır	
fuzzy	Belirsiz eşleşmeyi kullan	

Triggers

depo güncellenmesinden sonra

Tüm PO dosyalarını (*Dosya maskesi* ile yapılandırıldığı gibi) **msgmerge** komutunu kullanarak POT dosyasıyla eşleşecek şekilde (*Yeni çeviriler için şablon* ile yapılandırıldığı gibi) günceller.

Yukarı akış deposundan yeni değişiklikler çekildiğinde tetiklenir. Eklenti yapılandırması aracılığıyla msgmerge komut satırı seçeneklerinin çoğu yapılandırılabilir.

Ayrıca bakınız:

Does Weblate update translation files besides translations?

Git işlemlerini sıkıştır

Ekenti kimliği

weblate.git.squash

Yapılandırma

squash	Sıkıştırmayı işle	Kullanılabilir seçenekler: all – Tüm işlemler bir arada language – Dil başına file – Dosya başına author – Yazar başına
append-trailer	Sıkıştırılmış işlemlerin iletisine art-bilgileri ekle	‘Ortak yazar: ...’ gibi bir işleme iletisinin başka bir serbest biçim parçasının sonunda, artbilgi satırları RFC 822 e-posta üstbilgilerine benzeyen satırlardır.
commit-message	İletiyi işle	Bu işleme iletisi, sıkıştırılmış işleme iletilerinden gelen birleşik işleme iletileri yerine kullanılacaktır.

Triggers

depo işlenmesinden sonra

Değişiklikleri yollamadan önce Git işlemlerini sıkıştırın.

Git işlemleri, aşağıdaki modlardan birinde değişiklikleri yollamadan önce sıkıştırılabilir:

- Tüm işlemler bir arada
- Dil başına
- Dosya başına
- Hazırlayan başına

Orijinal işleme iletileri tutulur, ancak *Hazırlayan başına* seçilmediği veya işleme iletisi eklemek üzere özelleştirilmediği sürece yazarlık kaybedilir.

The original commit messages can optionally be overridden with a custom commit message.

Trailers (commit lines like `Co-authored-by: ...`) can optionally be removed from the original commit messages and appended to the end of the squashed commit message. This also generates proper `Co-authored-by: credit` for every translator.

JSON çıktısını özelleştir

Ekenti kimliği

weblate.json.customize

Yapılandırma

sort_keys	JSON anahtarlarını sırala	
indent	JSON girintisi	
style	JSON girinti stili	Kullanılabilir seçenekler: spaces – Spaces tabs – Tabs

Triggers

depolama yüklendikten sonra

JSON çıktısı davranışının ayarlanmasına izin verir, örneğin girinti veya sıralama.

Format the Java properties file

Ekleni kimliđi

`weblate.properties.sort`

Yapılandırma

Bu eklentinin yapılandırması yok.

Triggers

depo işlenmesinden önce

Formats and sorts the Java properties file.

- Consolidates newlines to Unix ones.
- Uppercase formatting of Unicode escape sequences (in case they are present).
- Strips blank lines and comments.
- Sorts the strings by the keys.
- Drops duplicate strings.

Eski yorum kaldırma

3.7 sürümünde geldi.

Ekleni kimliđi

`weblate.removal.comments`

Yapılandırma

age	Saklanacak günler	
-----	-------------------	--

Triggers

günlük

Yorumların kaldırılması için bir zaman dilimi ayarla.

Bu, güncelliđini yitirmiş olabilecek eski yorumları kaldırmak için faydalı olabilir. Yorumun eski olması önemlerini yitirdikleri anlamına gelmediđi için dikkatli kullanın.

Eski öneri kaldırma

3.7 sürümünde geldi.

Ekleni kimliđi

`weblate.removal.suggestions`

Yapılandırma

age	Saklanacak gün-ler	
vo-tes	Oy verme eşiđi	Kaldırma eşiđi. Bu alanın oylamanın kapatılması ile etkisi yok-tur.

Triggers

günlük

Önerilerin kaldırılması için bir zaman dilimi ayarlayın.

Belirli bir zaman diliminde yeterince olumlu oy almayan önerileri kaldırmak için öneri oylamayla (bkz: [Peer review](#)) bağlantılı olarak çok faydalı olabilir.

RESX dosyalarını güncelle

3.9 sürümünde geldi.

Eklenti kimliği

`weblate.resx.update`

Yapılandırma

Bu eklentinin yapılandırması yok.

Triggers

depo güncellenmesinden sonra

Tek dilli yukarı akış tabanlı dosya ile eşleştirmek için tüm çeviri dosyalarını güncelleyin. Kullanılmayan dizgiler kaldırılır ve kaynak dizginin kopyaları olarak yenileri eklenir.

İpucu: Yalnızca eski çeviri anahtarlarını kaldırmak istiyorsanız *Çeviri dosyaslarını temizle* kullanın.

Ayrıca bakınız:

Does Weblate update translation files besides translations?

YAML çıktısını özelleştir

3.10.2 sürümünde geldi.

Eklenti kimliği

`weblate.yaml.customize`

Yapılandırma

<code>indent</code>	YAML girintisi	
<code>width</code>	Uzun satırları kaydır	Kullanılabilir seçenekler: 80 – Satırları, 80 karakterde kaydır 100 – Satırları, 100 karakterde kaydır 120 – Satırları, 120 karakterde kaydır 180 – Satırları, 180 karakterde kaydır 65535 – Satır kaydırma yok
<code>line_breaks</code>	Satır sonları	Kullanılabilir seçenekler: dos – DOS (\r\n) unix – UNIX (\n) mac – MAC (\r)

Triggers

depolama yüklendikten sonra

YAML çıktısı davranışının ayarlanmasına izin verir, örneğin satır uzunluğu veya yeni satırlar.

2.14.2 Eklenti listesini özelleştirme

The list of add-ons is configured by `WEBLATE_ADDONS`. To add another add-on, simply include the absolute class name in this setting.

2.14.3 Eklenti yazma

You can write your own add-ons too, create a subclass of `weblate.addons.base.BaseAddon` to define the add-on metadata, and then implement a callback to do the processing.

Ayrıca bakınız:

Developing add-ons

2.14.4 Eklentiden betikleri çalıştırma

Add-ons can also be used to execute external scripts. This used to be integrated in Weblate, but now you have to write some code to wrap your script with an add-on.

```
#
# Copyright © 2012-2022 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program. If not, see <https://www.gnu.org/licenses/>.
#
"""Example pre commit script."""

from django.utils.translation import gettext_lazy as _

from weblate.addons.events import EVENT_PRE_COMMIT
from weblate.addons.scripts import BaseScriptAddon

class ExamplePreAddon(BaseScriptAddon):
    # Event used to trigger the script
    events = (EVENT_PRE_COMMIT,)
    # Name of the addon, has to be unique
    name = "weblate.example.pre"
    # Verbose name and long description
    verbose = _("Execute script before commit")
    description = _("This add-on executes a script.")

    # Script to execute
    script = "/bin/true"
    # File to add in commit (for pre commit event)
    # does not have to be set
    add_file = "po/{{ language_code }}.po"
```


For installation instructions see *Custom quality checks, add-ons and auto-fixes*.

The script is executed with the current directory set to the root of the VCS repository for any given component.

Additionally, the following environment variables are available:

WL_VCS

Kullanılan sürüm denetimi sistemi.

WL_REPO

Yukarı akış deposu URL'si.

WL_PATH

VCS deposunun mutlak yolu.

WL_BRANCH

2.11 sürümünde geldi.

Geçerli bileşende yapılandırılan depo dalı.

WL_FILEMASK

Geçerli bileşen için dosya maskesi.

WL_TEMPLATE

Tek dilli çeviriler için şablonun dosya adı (boş olabilir).

WL_NEW_BASE

2.14 sürümünde geldi.

Yeni çeviriler oluşturmak için kullanılan dosyanın adı (boş olabilir).

WL_FILE_FORMAT

Geçerli bileşende kullanılan dosya biçimi.

WL_LANGUAGE

İşlenen geçerli çevirinin dili (bileşen düzeyinde kancalar için kullanılamaz).

WL_PREVIOUS_HEAD

Güncellemeden sonraki eski HEAD (yalnızca güncelleme sonrası kancayı çalıştırdıktan sonra kullanılabilir).

WL_COMPONENT_SLUG

3.9 sürümünde geldi.

URL oluşturmak için kullanılan bileşen kısaltması.

WL_PROJECT_SLUG

3.9 sürümünde geldi.

URL oluşturmak için kullanılan proje kısaltması.

WL_COMPONENT_NAME

3.9 sürümünde geldi.

Bileşen adı.

WL_PROJECT_NAME

3.9 sürümünde geldi.

Proje adı.

WL_COMPONENT_URL

3.9 sürümünde geldi.

Bileşen URL'si.

WL_ENGAGE_URL

3.9 sürümünde geldi.

Proje etkileşim URL'si.

Ayrıca bakınız:

Component configuration

Güncelleme sonrası depo işleme

Can be used to update translation files when the VCS upstream source changes. To achieve this, please remember Weblate only sees files committed to the VCS, so you need to commit changes as a part of the script.

For example with Gulp you can do it using following code:

```
#!/bin/sh
gulp --gulpfile gulp-i18n-extract.js
git commit -m 'Update source strings' src/languages/en.lang.json
```

Depoya işlenmeden önce çevirilerin işlemde geçirilmesi

Use the commit script to automatically change a translation before it is committed to the repository.

It is passed as a single parameter consisting of the filename of a current translation.

2.15 Çeviri Belleği

2.20 sürümünde geldi.

Weblate comes with a built-in translation memory consisting of the following:

- Manually imported translation memory (see *User interface*).
- Automatically stored translations performed in Weblate (depending on *Translation memory scopes*).
- Automatically imported past translations.

Content in the translation memory can be applied one of two ways:

- Manually, *Otomatik öneriler* view while translating.
- Automatically, by translating strings using *Otomatik çeviri*, or *Otomatik çeviri* add-on.

For installation tips, see *Weblate Translation Memory*, which is turned on by default.

2.15.1 Translation memory scopes

3.2 sürümünde geldi: In earlier versions translation memory could be only loaded from a file corresponding to the current imported translation memory scope.

The translation memory scopes are there to allow both privacy and sharing of translations, to suit the desired behavior.

Imported translation memory

Importing arbitrary translation memory data using the `import_memory` command makes memory content available to all users and projects.

Per user translation memory

Stores all user translations automatically in the personal translation memory of each respective user.

Per project translation memory

All translations within a project are automatically stored in a project translation memory only available for this project.

Paylaşılan çeviri belleği

All translation within projects with shared translation memory turned on are stored in a shared translation memory available to all projects.

Please consider carefully whether to turn this feature on for shared Weblate installations, as it can have severe implications:

- The translations can be used by anybody else.
- This might lead to disclosing secret information.

2.15.2 Managing translation memory

User interface

3.2 sürümünde geldi.

In the basic user interface you can manage per user and per project translation memories. It can be used to download, wipe or import translation memory.

İpucu: Translation memory in JSON can be imported into Weblate, TMX is provided for interoperability with other tools.

Ayrıca bakınız:

Weblate Çeviri Belleği Şeması

testuser / Translation memory

Translation memory status ⓘ

Number of your entries	0	Download as JSON	Download as TMX	Delete
Total number of entries	0			

Import translation memory

File

Choose File No file chosen

You can upload a TMX or JSON file.

Upload

Powered by Weblate 4.13 About Weblate Legal Contact Documentation Donate to Weblate

Yönetim arayüzü

There are several management commands to manipulate the translation memory content. These operate on the translation memory as whole, unfiltered by scopes (unless requested by parameters):

dump_memory

Exports the memory into JSON

import_memory

Imports TMX or JSON files into the translation memory

2.16 Yapılandırma

All settings are stored in `settings.py` (as is usual for Django).

Not: After changing any of these settings, you need to restart Weblate - both WSGI and Celery processes.

In case it is run as `mod_wsgi`, you need to restart Apache to reload the configuration.

Ayrıca bakınız:

Please also check [Django's documentation](#) for parameters configuring Django itself.

2.16.1 AKISMET_API_KEY

Weblate can use Akismet to check incoming anonymous suggestions for spam. Visit akismet.com to purchase an API key and associate it with a site.

2.16.2 ANONYMOUS_USER_NAME

Username of users that are not signed in.

Ayrıca bakınız:

Erişim denetimi

2.16.3 AUDITLOG_EXPIRY

3.6 sürümünde geldi.

How many days Weblate should keep audit logs, which contain info about account activity.

Defaults to 180 days.

2.16.4 AUTH_LOCK_ATTEMPTS

2.14 sürümünde geldi.

Maximum number of failed authentication attempts before rate limiting is applied.

This is currently applied in the following locations:

- Sign in. Deletes the account password, preventing the user from signing in without requesting a new password.
- Password reset. Prevents new e-mails from being sent, avoiding spamming users with too many password reset attempts.

Defaults to 10.

Ayrıca bakınız:

Rate limiting

2.16.5 AUTO_UPDATE

3.2 sürümünde geldi.

3.11 sürümünde değişti: The original on/off option was changed to differentiate which strings are accepted.

Updates all repositories on a daily basis.

İpucu: Useful if you are not using *Bildirim kancaları* to update Weblate repositories automatically.

Not: On/off options exist in addition to string selection for backward compatibility.

Options are:

"none"

No daily updates.

"remote" also False

Only update remotes.

"full" also True

Update remotes and merge working copy.

Not: This requires that *Background tasks using Celery* is working, and will take effect after it is restarted.

2.16.6 AVATAR_URL_PREFIX

Prefix for constructing avatar URLs as: `${AVATAR_URL_PREFIX}/avatar/${MAIL_HASH}?${PARAMS}`. The following services are known to work:

Gravatar (default), as per <https://gravatar.com/>

```
AVATAR_URL_PREFIX = 'https://www.gravatar.com/'
```

Libravatar, as per <https://www.libravatar.org/>

```
AVATAR_URL_PREFIX = 'https://www.libravatar.org/'
```

Ayrıca bakınız:

Avatar ön bellekleme, `ENABLE_AVATARS`, `Avatars`

2.16.7 AUTH_TOKEN_VALID

2.14 sürümünde geldi.

How long the authentication token and temporary password from password reset e-mails is valid for. Set in number of seconds, defaulting to 172800 (2 days).

2.16.8 AUTH_PASSWORD_DAYS

2.15 sürümünde geldi.

How many days using the same password should be allowed.

Not: Password changes made prior to Weblate 2.15 will not be accounted for in this policy.

Defaults to 180 days.

2.16.9 AUTOFIX_LIST

List of automatic fixes to apply when saving a string.

Not: Provide a fully-qualified path to the Python class that implementing the autofixer interface.

Available fixes:

`weblate.trans.autofixes.whitespace.SameBookendingWhitespace`

Matches whitespace at the start and end of the string to the source.

`weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis`

Replaces trailing dots (...) if the source string has a corresponding ellipsis (...).

`weblate.trans.autofixes.chars.RemoveZeroSpace`

Removes zero-width space characters if the source does not contain any.

`weblate.trans.autofixes.chars.RemoveControlChars`

Removes control characters if the source does not contain any.

`weblate.trans.autofixes.html.BleachHTML`

Removes unsafe HTML markup from strings flagged as `safe-html` (see *Güvenli olmayan HTML*).

You can select which ones to use:

```
AUTOFIX_LIST = (
    "weblate.trans.autofixes.whitespace.SameBookendingWhitespace",
    "weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis",
)
```

Ayrıca bakınız:*Automatic fixups, Custom automatic fixups*

2.16.10 BACKGROUND_TASKS

4.5.2 sürümünde geldi.

Defines how often lengthy maintenance tasks should be triggered for a component.

Right now this controls:

- *Otomatik çeviri* add-on
- *Checks and fixups* recalculation

Olası seçenekler:

- monthly (this is the default)
- weekly
- daily
- never

Not: Increasing the frequency is not recommended when Weblate contains thousands of components.

2.16.11 BASE_DIR

Base directory where Weblate sources are located. Used to derive several other paths by default:

- *DATA_DIR*

Default value: Top level directory of Weblate sources.

2.16.12 BASIC_LANGUAGES

4.4 sürümünde geldi.

List of languages to offer users for starting new translation. When not specified built-in list is used which includes all commonly used languages, but without country specific variants.

This only limits non privileged users to add unwanted languages. The project admins are still presented with full selection of languages defined in Weblate.

Not: This does not define new languages for Weblate, it only filters existing ones in the database.

Örnek:

```
BASIC_LANGUAGES = {"cs", "it", "ja", "en"}
```

Ayrıca bakınız:*Dil tanımları*

2.16.13 BORG_EXTRA_ARGS

4.9 sürümünde geldi.

You can pass additional arguments to **borg create** when built-in backups are triggered.

Örnek:

```
BORG_EXTRA_ARGS = ["--exclude", "vcs/"]
```

Ayrıca bakınız:

Backing up and moving Weblate, `borg create`

2.16.14 CSP_SCRIPT_SRC, CSP_IMG_SRC, CSP_CONNECT_SRC, CSP_STYLE_SRC, CSP_FONT_SRC

Customize Content-Security-Policy header for Weblate. The header is automatically generated based on enabled integrations with third-party services (Matomo, Google Analytics, Sentry, ...).

All these default to empty list.

Örnek:

```
# Enable Cloudflare Javascript optimizations
CSP_SCRIPT_SRC = ["ajax.cloudflare.com"]
```

Ayrıca bakınız:

Content security policy, Content Security Policy (CSP)

2.16.15 CHECK_LIST

List of quality checks to perform on a translation.

Not: Provide a fully-qualified path to the Python class implementing the check interface.

Adjust the list of checks to include ones relevant to you.

All built-in *Quality checks* are turned on by default, from where you can change these settings. By default they are commented out in *Sample configuration* so that default values are used. New checks then carried out for each new Weblate version.

You can turn off all checks:

```
CHECK_LIST = ()
```

You can turn on only a few:

```
CHECK_LIST = (
    "weblate.checks.chars.BeginNewlineCheck",
    "weblate.checks.chars.EndNewlineCheck",
    "weblate.checks.chars.MaxLengthCheck",
)
```

Not: Changing this setting only affects newly changed translations, existing checks will still be stored in the database. To also apply changes to the stored translations, run *updatechecks*.

Ayrıca bakınız:

Quality checks, Customizing behavior using flags

2.16.16 COMMENT_CLEANUP_DAYS

3.6 sürümünde geldi.

Delete comments after a given number of days. Defaults to `None`, meaning no deletion at all.

2.16.17 COMMIT_PENDING_HOURS

2.10 sürümünde geldi.

Number of hours between committing pending changes by way of the background task.

Ayrıca bakınız:

Component configuration, İşlenecek değişikliklerin yaşı, Running maintenance tasks, `commit_pending`

2.16.18 CONTACT_FORM

4.6 sürümünde geldi.

Configures how e-mail from the contact form is being sent. Choose a configuration that matches your mail server configuration.

"reply-to"

The sender is used in as *Reply-To*, this is the default behaviour.

"from"

The sender is used in as *From*. Your mail server needs to allow sending such e-mails.

2.16.19 DATA_DIR

The folder Weblate stores all data in. It contains links to VCS repositories, a fulltext index and various configuration files for external tools.

The following subdirectories usually exist:

home

Home directory used for invoking scripts.

ssh

SSH keys and configuration.

static

Default location for static Django files, specified by `STATIC_ROOT`. See *Serving static files*.

The Docker container uses a separate volume for this, see *Docker container volumes*.

media

Default location for Django media files, specified by `MEDIA_ROOT`. Contains uploaded screenshots, see *Visual context for strings*.

vcs

Çeviriler için sürüm denetimi depoları.

backups

Daily backup data, please check *Dumped data for backups* for details.

celery

Celery scheduler data, see *Background tasks using Celery*.

fonts:

User-uploaded fonts, see [Managing fonts](#).

Not: This directory has to be writable by Weblate. Running it as uWSGI means the `www-data` user should have write access to it.

The easiest way to achieve this is to make the user the owner of the directory:

```
sudo chown www-data:www-data -R $DATA_DIR
```

Defaults to `$BASE_DIR/data`.

Ayrıca bakınız:

[BASE_DIR](#), [Filesystem permissions](#), [Backing up and moving Weblate](#)

2.16.20 DATABASE_BACKUP

3.1 sürümünde geldi.

Whether the database backups should be stored as plain text, compressed or skipped. The authorized values are:

- "plain"
- "compressed"
- "none"

Ayrıca bakınız:

[Backing up and moving Weblate](#)

2.16.21 DEFAULT_ACCESS_CONTROL

3.3 sürümünde geldi.

The default access control setting for new projects:

- 0
Public
- 1
Protected
- 100
Private
- 200
Custom

Use *Custom* if you are managing ACL manually, which means not relying on the internal Weblate management.

Ayrıca bakınız:

[Proje erişim denetimi](#), [Erişim denetimi](#)

2.16.22 DEFAULT_AUTO_WATCH

4.5 sürümünde geldi.

Configures whether *Automatically watch projects on contribution* should be turned on for new users. Defaults to `True`.

Ayrıca bakınız:

Bildirimler

2.16.23 DEFAULT_RESTRICTED_COMPONENT

4.1 sürümünde geldi.

The default value for component restriction.

Ayrıca bakınız:

Restricted access, Grupların kapsamı

2.16.24 DEFAULT_ADD_MESSAGE, DEFAULT_ADDON_MESSAGE, DEFAULT_COMMIT_MESSAGE, DEFAULT_DELETE_MESSAGE, DEFAULT_MERGE_MESSAGE

Default commit messages for different operations, please check *Component configuration* for details.

Ayrıca bakınız:

Template markup, Component configuration, Commit, add, delete, merge, add-on, and merge request messages

2.16.25 DEFAULT_ADDONS

Default add-ons to install on every created component.

Not: This setting affects only newly created components.

Example:

```
DEFAULT_ADDONS = {
    # Add-on with no parameters
    "weblate.flags.target_edit": {},
    # Add-on with parameters
    "weblate.autotranslate.autotranslate": {
        "mode": "suggest",
        "filter_type": "todo",
        "auto_source": "mt",
        "component": "",
        "engines": ["weblate-translation-memory"],
        "threshold": "80",
    },
}
```

Ayrıca bakınız:

install_addon, Eklentiler, WEBLATE_ADDONS

2.16.26 DEFAULT_COMMITER_EMAIL

2.4 sürümünde geldi.

Committer e-mail address defaulting to `noreply@weblate.org`.

Ayrıca bakınız:

DEFAULT_COMMITER_NAME

2.16.27 DEFAULT_COMMITER_NAME

2.4 sürümünde geldi.

Committer name defaulting to `Weblate`.

Ayrıca bakınız:

DEFAULT_COMMITER_EMAIL

2.16.28 DEFAULT_LANGUAGE

4.3.2 sürümünde geldi.

Default source language to use for example in *Kaynak dil*.

Defaults to *en*. The matching language object needs to exist in the database.

Ayrıca bakınız:

Dil tanımları, Kaynak dil

2.16.29 DEFAULT_MERGE_STYLE

3.4 sürümünde geldi.

Merge style for any new components.

- *rebase* - default
- *merge*

Ayrıca bakınız:

Component configuration, Birleştirme stili

2.16.30 DEFAULT_SHARED_TM

3.2 sürümünde geldi.

Configures default value of *Paylaşılan çeviri belleği kullan* and *Paylaşılan çeviri belleğine katkıda bulunun*.

2.16.31 DEFAULT_TRANSLATION_PROPAGATION

2.5 sürümünde geldi.

Default setting for translation propagation, defaults to `True`.

Ayrıca bakınız:

Component configuration, Çeviri yaymaya izin ver

2.16.32 DEFAULT_PULL_MESSAGE

Configures the default title and message for pull requests.

2.16.33 ENABLE_AVATARS

Whether to turn on Gravatar-based avatars for users. By default this is on.

Avatars are fetched and cached on the server, lowering the risk of leaking private info, speeding up the user experience.

Ayrıca bakınız:

Avatar ön bellekleme, AVATAR_URL_PREFIX, Avatars

2.16.34 ENABLE_HOOKS

Whether to enable anonymous remote hooks.

Ayrıca bakınız:

Bildirim kancaları

2.16.35 ENABLE_HTTPS

Whether to send links to Weblate as HTTPS or HTTP. This setting affects sent e-mails and generated absolute URLs.

In the default configuration this is also used for several Django settings related to HTTPS - it enables secure cookies, toggles HSTS or enables redirection to HTTPS URL.

The HTTPS redirection might be problematic in some cases and you might hit issue with infinite redirection in case you are using a reverse proxy doing SSL termination which does not correctly pass protocol headers to Django. Please tweak your reverse proxy configuration to emit `X-Forwarded-Proto` or `Forwarded` headers or configure `SECURE_PROXY_SSL_HEADER` to let Django correctly detect the SSL status.

Ayrıca bakınız:

`SESSION_COOKIE_SECURE`, `CSRF_COOKIE_SECURE`, `SECURE_SSL_REDIRECT`, `SECURE_PROXY_SSL_HEADER` *Set correct site domain*

2.16.36 ENABLE_SHARING

Turn on/off the *Share* menu so users can share translation progress on social networks.

2.16.37 GET_HELP_URL

4.5.2 sürümünde geldi.

URL where support for your Weblate instance can be found.

2.16.38 GITEA_CREDENTIALS

4.12 sürümünde geldi.

List for credentials for Gitea servers.

İpucu: Use this in case you want Weblate to interact with more of them, for single Gitea endpoint stick with *GITEA_USERNAME* and *GITEA_TOKEN*.

```
GITEA_CREDENTIALS = {
  "try.gitea.io": {
    "username": "weblate",
    "token": "your-api-token",
  },
  "gitea.example.com": {
    "username": "weblate",
    "token": "another-api-token",
  },
}
```

2.16.39 GITEA_USERNAME

4.12 sürümünde geldi.

Gitea username used to send pull requests for translation updates.

Ayrıca bakınız:

GITEA_CREDENTIALS, *Gitea pull requests*

2.16.40 GITEA_TOKEN

4.12 sürümünde geldi.

Gitea personal access token used to make API calls to send pull requests for translation updates.

Ayrıca bakınız:

GITEA_CREDENTIALS, *Gitea pull requests*, *Creating a Gitea personal access token*

2.16.41 GITLAB_CREDENTIALS

4.3 sürümünde geldi.

List for credentials for GitLab servers.

İpucu: Use this in case you want Weblate to interact with more of them, for single GitLab endpoint stick with *GITLAB_USERNAME* and *GITLAB_TOKEN*.

```
GITLAB_CREDENTIALS = {
  "gitlab.com": {
    "username": "weblate",
    "token": "your-api-token",
  },
  "gitlab.example.com": {
    "username": "weblate",
    "token": "another-api-token",
  },
}
```

2.16.42 GITLAB_USERNAME

GitLab username used to send merge requests for translation updates.

Ayrıca bakınız:

GITLAB_CREDENTIALS, *GitLab merge requests*

2.16.43 GITLAB_TOKEN

4.3 sürümünde geldi.

GitLab personal access token used to make API calls to send merge requests for translation updates.

Ayrıca bakınız:

GITLAB_CREDENTIALS, *GitLab merge requests*, *GitLab: Personal access token*

2.16.44 GITHUB_CREDENTIALS

4.3 sürümünde geldi.

List for credentials for GitHub servers.

İpucu: Use this in case you want Weblate to interact with more of them, for single GitHub endpoint stick with *GITHUB_USERNAME* and *GITHUB_TOKEN*.

```
GITHUB_CREDENTIALS = {
  "api.github.com": {
    "username": "weblate",
    "token": "your-api-token",
  },
  "github.example.com": {
    "username": "weblate",
    "token": "another-api-token",
  },
}
```

2.16.45 GITHUB_USERNAME

GitHub username used to send pull requests for translation updates.

Ayrıca bakınız:

GITHUB_CREDENTIALS, *GitHub pull requests*

2.16.46 GITHUB_TOKEN

4.3 sürümünde geldi.

GitHub personal access token used to make API calls to send pull requests for translation updates.

Ayrıca bakınız:

GITHUB_CREDENTIALS, *GitHub pull requests*, *Creating a GitHub personal access token*

2.16.47 GOOGLE_ANALYTICS_ID

Google Analytics ID to turn on monitoring of Weblate using Google Analytics.

2.16.48 HIDE_REPO_CREDENTIALS

Hide repository credentials from the web interface. In case you have repository URL with user and password, Weblate will hide it when related info is shown to users.

For example instead of `https://user:password@git.example.com/repo.git` it will show just `https://git.example.com/repo.git`. It tries to clean up VCS error messages too in a similar manner.

Not: This is turned on by default.

2.16.49 HIDE_VERSION

4.3.1 sürümünde geldi.

Hides version information from unauthenticated users. This also makes all documentation links point to latest version instead of the documentation matching currently installed version.

Hiding version is recommended security practice in some corporations, but it doesn't prevent attacker to figure out version by probing the behavior.

Not: Bu, varsayılan olarak kapalıdır.

2.16.50 INTERLEDGER_PAYMENT_POINTERS

4.12.1 sürümünde geldi.

List of Interledger Payment Pointers (ILPs) for Web Monetization.

If multiple are specified, probabilistic revenue sharing is achieved by selecting one randomly.

Please check <<https://webmonetization.org/>> for more details.

İpucu: The default value lets users fund Weblate itself.

2.16.51 IP_BEHIND_REVERSE_PROXY

2.14 sürümünde geldi.

Indicates whether Weblate is running behind a reverse proxy.

If set to `True`, Weblate gets IP address from a header defined by `IP_PROXY_HEADER`.

Uyarı: Ensure you are actually using a reverse proxy and that it sets this header, otherwise users will be able to fake the IP address.

Not: This is not on by default.

Ayrıca bakınız:

Running behind reverse proxy, Rate limiting, IP_PROXY_HEADER, IP_PROXY_OFFSET

2.16.52 IP_PROXY_HEADER

2.14 sürümünde geldi.

Indicates which header Weblate should obtain the IP address from when `IP_BEHIND_REVERSE_PROXY` is turned on.

Defaults to `HTTP_X_FORWARDED_FOR`.

Ayrıca bakınız:

Running behind reverse proxy, Rate limiting, SECURE_PROXY_SSL_HEADER, IP_BEHIND_REVERSE_PROXY, IP_PROXY_OFFSET

2.16.53 IP_PROXY_OFFSET

2.14 sürümünde geldi.

Indicates which part of `IP_PROXY_HEADER` is used as client IP address.

Depending on your setup, this header might consist of several IP addresses, (for example `X-Forwarded-For: a, b, client-ip`) and you can configure which address from the header is used as client IP address here.

Uyarı: Setting this affects the security of your installation, you should only configure it to use trusted proxies for determining IP address.

Defaults to 0.

Ayrıca bakınız:

Running behind reverse proxy, Rate limiting, SECURE_PROXY_SSL_HEADER, IP_BEHIND_REVERSE_PROXY, IP_PROXY_HEADER

2.16.54 LEGAL_URL

3.5 sürümünde geldi.

URL where your Weblate instance shows its legal documents.

İpucu: Useful if you host your legal documents outside Weblate for embedding them inside Weblate, please check [Yasal](#) for details.

Example:

```
LEGAL_URL = "https://weblate.org/terms/"
```

Ayrıca bakınız:

[PRIVACY_URL](#)

2.16.55 LICENSE_EXTRA

Additional licenses to include in the license choices.

Not: Each license definition should be tuple of its short name, a long name and an URL.

For example:

```
LICENSE_EXTRA = [
    (
        "AGPL-3.0",
        "GNU Affero General Public License v3.0",
        "https://www.gnu.org/licenses/agpl-3.0-standalone.html",
    ),
]
```

2.16.56 LICENSE_FILTER

4.3 sürümünde değişti: Setting this to blank value now disables license alert.

Filter list of licenses to show. This also disables the license alert when set to empty.

Not: This filter uses the short license names.

For example:

```
LICENSE_FILTER = {"AGPL-3.0", "GPL-3.0-or-later"}
```

Following disables the license alert:

```
LICENSE_FILTER = set()
```

Ayrıca bakınız:

[alerts](#)

2.16.57 LICENSE_REQUIRED

Defines whether the license attribute in *Component configuration* is required.

Not: This is off by default.

2.16.58 LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH

Whether the length of a given translation should be limited. The restriction is the length of the source string × 10 characters.

İpucu: Set this to `False` to allow longer translations (up to 10,000 characters) irrespective of source string length.

Not: Defaults to `True`.

2.16.59 LOCALIZE_CDN_URL and LOCALIZE_CDN_PATH

These settings configure the *JavaScript yerelleştirme CDN'i* add-on. `LOCALIZE_CDN_URL` defines root URL where the localization CDN is available and `LOCALIZE_CDN_PATH` defines path where Weblate should store generated files which will be served at the `LOCALIZE_CDN_URL`.

İpucu: On Hosted Weblate, this uses `https://weblate-cdn.com/`.

Ayrıca bakınız:

JavaScript yerelleştirme CDN'i

2.16.60 LOGIN_REQUIRED_URLS

A list of URLs you want to require signing in. (Besides the standard rules built into Weblate).

İpucu: This allows you to password protect a whole installation using:

```
LOGIN_REQUIRED_URLS = (r"/(.*)$",)
REST_FRAMEWORK["DEFAULT_PERMISSION_CLASSES"] = [
    "rest_framework.permissions.IsAuthenticated"
]
```

İpucu: It is desirable to lock down API access as well, as shown in the above example.

Ayrıca bakınız:

`REQUIRE_LOGIN`

2.16.61 LOGIN_REQUIRED_URLS_EXCEPTIONS

List of exceptions for `LOGIN_REQUIRED_URLS`. If not specified, users are allowed to access the sign in page.

Some of exceptions you might want to include:

```
LOGIN_REQUIRED_URLS_EXCEPTIONS = (  
    r"/accounts/(.*)$", # Required for sign in  
    r"/static/(.*)$", # Required for development mode  
    r"/widgets/(.*)$", # Allowing public access to widgets  
    r"/data/(.*)$", # Allowing public access to data exports  
    r"/hooks/(.*)$", # Allowing public access to notification hooks  
    r"/api/(.*)$", # Allowing access to API  
    r"/js/i18n/$", # JavaScript localization  
)
```

2.16.62 MATOMO_SITE_ID

ID of a site in Matomo (formerly Piwik) you want to track.

Not: This integration does not support the Matomo Tag Manager.

Ayrıca bakınız:

`MATOMO_URL`

2.16.63 MATOMO_URL

Full URL (including trailing slash) of a Matomo (formerly Piwik) installation you want to use to track Weblate use. Please check <<https://matomo.org/>> for more details.

İpucu: This integration does not support the Matomo Tag Manager.

For example:

```
MATOMO_SITE_ID = 1  
MATOMO_URL = "https://example.matomo.cloud/"
```

Ayrıca bakınız:

`MATOMO_SITE_ID`

2.16.64 NEARBY_MESSAGES

How many strings to show around the currently translated string. This is just a default value, users can adjust this in *Kullanıcı profili*.

2.16.65 DEFAULT_PAGE_LIMIT

4.7 sürümünde geldi.

Default number of elements to display when pagination is active.

2.16.66 PAGURE_CREDENTIALS

4.3.2 sürümünde geldi.

List for credentials for Pagure servers.

İpucu: Use this in case you want Weblate to interact with more of them, for single Pagure endpoint stick with [PAGURE_USERNAME](#) and [PAGURE_TOKEN](#).

```
PAGURE_CREDENTIALS = {
    "pagure.io": {
        "username": "weblate",
        "token": "your-api-token",
    },
    "pagure.example.com": {
        "username": "weblate",
        "token": "another-api-token",
    },
}
```

2.16.67 PAGURE_USERNAME

4.3.2 sürümünde geldi.

Çeviri güncellemeleri için birleştirme istekleri gönderirken kullanılacak Pagure kullanıcı adı.

Ayrıca bakınız:

[PAGURE_CREDENTIALS](#), [Pagure merge requests](#)

2.16.68 PAGURE_TOKEN

4.3.2 sürümünde geldi.

Pagure personal access token used to make API calls to send merge requests for translation updates.

Ayrıca bakınız:

[PAGURE_CREDENTIALS](#), [Pagure merge requests](#), [Pagure API](#)

2.16.69 PRIVACY_URL

4.8.1 sürümünde geldi.

URL where your Weblate instance shows its privacy policy.

İpucu: Useful if you host your legal documents outside Weblate for embedding them inside Weblate, please check [Yasal](#) for details.

Example:

```
PRIVACY_URL = "https://weblate.org/terms/"
```

Ayrıca bakınız:

LEGAL_URL

2.16.70 RATELIMIT_ATTEMPTS

3.2 sürümünde geldi.

Maximum number of authentication attempts before rate limiting is applied.

Defaults to 5.

Ayrıca bakınız:

Rate limiting, RATELIMIT_WINDOW, RATELIMIT_LOCKOUT

2.16.71 RATELIMIT_WINDOW

3.2 sürümünde geldi.

How long authentication is accepted after rate limiting applies.

An amount of seconds defaulting to 300 (5 minutes).

Ayrıca bakınız:

Rate limiting, RATELIMIT_ATTEMPTS, RATELIMIT_LOCKOUT

2.16.72 RATELIMIT_LOCKOUT

3.2 sürümünde geldi.

How long authentication is locked after rate limiting applies.

An amount of seconds defaulting to 600 (10 minutes).

Ayrıca bakınız:

Rate limiting, RATELIMIT_ATTEMPTS, RATELIMIT_WINDOW

2.16.73 REGISTRATION_ALLOW_BACKENDS

4.1 sürümünde geldi.

List of authentication backends to allow registration from. This only limits new registrations, users can still authenticate and add authentication using all configured authentication backends.

It is recommended to keep *REGISTRATION_OPEN* enabled while limiting registration backends, otherwise users will be able to register, but Weblate will not show links to register in the user interface.

Example:

```
REGISTRATION_ALLOW_BACKENDS = ["azuread-oauth2", "azuread-tenant-oauth2"]
```

İpucu: The backend names match names used in URL for authentication.

Ayrıca bakınız:

REGISTRATION_OPEN, Kimlik doğrulama

2.16.74 REGISTRATION_CAPTCHA

A value of either `True` or `False` indicating whether registration of new accounts is protected by CAPTCHA. This setting is optional, and a default of `True` will be assumed if it is not supplied.

If turned on, a CAPTCHA is added to all pages where a users enters their e-mail address:

- New account registration.
- Password recovery.
- Adding e-mail to an account.
- Contact form for users that are not signed in.

2.16.75 REGISTRATION_EMAIL_MATCH

2.17 sürümünde geldi.

Allows you to filter which e-mail addresses can register.

Defaults to `.*`, which allows any e-mail address to be registered.

You can use it to restrict registration to a single e-mail domain:

```
REGISTRATION_EMAIL_MATCH = r"^.*@weblate\.org$"
```

2.16.76 REGISTRATION_OPEN

Whether registration of new accounts is currently permitted. This optional setting can remain the default `True`, or changed to `False`.

This setting affects built-in authentication by e-mail address or through the Python Social Auth (you can whitelist certain back-ends using [REGISTRATION_ALLOW_BACKENDS](#)).

Not: If using third-party authentication methods such as [LDAP authentication](#), it just hides the registration form, but new users might still be able to sign in and create accounts.

Ayrıca bakınız:

[REGISTRATION_ALLOW_BACKENDS](#), [REGISTRATION_EMAIL_MATCH](#), [Kimlik doğrulama](#)

2.16.77 REPOSITORY_ALERT_THRESHOLD

4.0.2 sürümünde geldi.

Threshold for triggering an alert for outdated repositories, or ones that contain too many changes. Defaults to 25.

Ayrıca bakınız:

[alerts](#)

2.16.78 REQUIRE_LOGIN

4.1 sürümünde geldi.

This enables `LOGIN_REQUIRED_URLS` and configures REST framework to require authentication for all API endpoints.

Not: This is implemented in the *Sample configuration*. For Docker, use `WEBLATE_REQUIRE_LOGIN`.

2.16.79 SENTRY_DSN

3.9 sürümünde geldi.

Sentry DSN to use for *Collecting error reports*.

Ayrıca bakınız:

[Django integration for Sentry](#)

2.16.80 SESSION_COOKIE_AGE_AUTHENTICATED

4.3 sürümünde geldi.

Set session expiry for authenticated users. This complements `SESSION_COOKIE_AGE` which is used for unauthenticated users.

Ayrıca bakınız:

`SESSION_COOKIE_AGE`

2.16.81 SIMPLIFY_LANGUAGES

Use simple language codes for default language/country combinations. For example an `fr_FR` translation will use the `fr` language code. This is usually the desired behavior, as it simplifies listing languages for these default combinations.

Turn this off if you want to different translations for each variant.

2.16.82 SITE_DOMAIN

Configures site domain. This is necessary to produce correct absolute links in many scopes (for example activation e-mails, notifications or RSS feeds).

In case Weblate is running on non-standard port, include it here as well.

Örnekler:

```
# Production site with domain name
SITE_DOMAIN = "weblate.example.com"

# Local development with IP address and port
SITE_DOMAIN = "127.0.0.1:8000"
```

Not: This setting should only contain the domain name. For configuring protocol, (enabling and enforcing HTTPS) use `ENABLE_HTTPS` and for changing URL, use `URL_PREFIX`.

İpucu: On a Docker container, the site domain is configured through `WEBLATE_ALLOWED_HOSTS`.

Ayrıca bakınız:

Set correct site domain, Allowed hosts setup, Correctly configure HTTPS `WEBLATE_SITE_DOMAIN`, `ENABLE_HTTPS`

2.16.83 SITE_TITLE

Site title to be used for the website and sent e-mails.

2.16.84 SPECIAL_CHARS

Additional characters to include in the visual keyboard, *Visual keyboard*.

The default value is:

```
SPECIAL_CHARS = ("\t", "\n", "\u00a0", "...")
```

2.16.85 SINGLE_PROJECT

3.8 sürümünde geldi.

Redirects users directly to a project or component instead of showing the dashboard. You can either set it to `True` and in this case it only works in case there is actually only single project in Weblate. Alternatively set the project slug, and it will redirect unconditionally to this project.

3.11 sürümünde değişti: The setting now also accepts a project slug, to force displaying that single project.

Example:

```
SINGLE_PROJECT = "test"
```

2.16.86 SSH_EXTRA_ARGS

4.9 sürümünde geldi.

Allows to add custom parameters when Weblate is invoking SSH. This is useful when connecting to servers using legacy encryption or other non-standard features.

For example when SSH connection in Weblate fails with *Unable to negotiate with legacyhost: no matching key exchange method found. Their offer: diffie-hellman-group1-sha1*, you can enable that using:

```
SSH_EXTRA_ARGS = "-oKexAlgorithms =+diffie-hellman-group1-sha1"
```

İpucu: The string is evaluated by shell, so make sure to quote any whitespace and special characters.

Ayrıca bakınız:

OpenSSH Legacy Options

2.16.87 STATUS_URL

The URL where your Weblate instance reports its status.

2.16.88 SUGGESTION_CLEANUP_DAYS

3.2.1 sürümünde geldi.

Automatically deletes suggestions after a given number of days. Defaults to `None`, meaning no deletions.

2.16.89 UPDATE_LANGUAGES

4.3.2 sürümünde geldi.

Controls whether languages database should be updated when running database migration and is enabled by default. This setting has no effect on invocation of `setuplang`.

Uyarı: The languages display might become inconsistent with this. Weblate language definitions extend over time and it will not display language code for the defined languages.

Ayrıca bakınız:

Yerleşik dil tanımları

2.16.90 URL_PREFIX

This setting allows you to run Weblate under some path (otherwise it relies on being run from the webserver root).

Not: To use this setting, you also need to configure your server to strip this prefix. For example with WSGI, this can be achieved by setting `WSGIScriptAlias`.

İpucu: The prefix should start with a `/`.

Example:

```
URL_PREFIX = "/translations"
```

Not: This setting does not work with Django's built-in server, you would have to adjust `urls.py` to contain this prefix.

2.16.91 VCS_BACKENDS

Configuration of available VCS backends.

Not: Weblate tries to use all supported back-ends you have the tools for.

İpucu: You can limit choices or add custom VCS back-ends by using this.

```
VCS_BACKENDS = ("weblate.vcs.git.GitRepository",)
```

Ayrıca bakınız:

Sürüm denetimi tümleşimi

2.16.92 VCS_CLONE_DEPTH

3.10.2 sürümünde geldi.

Configures how deep cloning of repositories Weblate should do.

Not: Currently this is only supported in *Git*. By default Weblate does shallow clones of the repositories to make cloning faster and save disk space. Depending on your usage (for example when using custom *Eklentiler*), you might want to increase the depth or turn off shallow clones completely by setting this to 0.

İpucu: In case you get fatal: protocol error: expected old/new/ref, got 'shallow <commit hash>' error when pushing from Weblate, turn off shallow clones completely by setting:

```
VCS_CLONE_DEPTH = 0
```

2.16.93 WEBLATE_ADDONS

List of add-ons available for use. To use them, they have to be enabled for a given translation component. By default this includes all built-in add-ons, when extending the list you will probably want to keep existing ones enabled, for example:

```
WEBLATE_ADDONS = (
    # Built-in add-ons
    "weblate.addons.gettext.GenerateMoAddon",
    "weblate.addons.gettext.UpdateLinguasAddon",
    "weblate.addons.gettext.UpdateConfigureAddon",
    "weblate.addons.gettext.MsgmergeAddon",
    "weblate.addons.gettext.GettextCustomizeAddon",
    "weblate.addons.gettext.GettextAuthorComments",
    "weblate.addons.cleanup.CleanupAddon",
    "weblate.addons.consistency.LanguangeConsistencyAddon",
    "weblate.addons.discovery.DiscoveryAddon",
    "weblate.addons.flags.SourceEditAddon",
    "weblate.addons.flags.TargetEditAddon",
    "weblate.addons.flags.SameEditAddon",
    "weblate.addons.flags.BulkEditAddon",
    "weblate.addons.generate.GenerateFileAddon",
    "weblate.addons.json.JSONCustomizeAddon",
    "weblate.addons.properties.PropertiesSortAddon",
    "weblate.addons.git.GitSquashAddon",
    "weblate.addons.removal.RemoveComments",
    "weblate.addons.removal.RemoveSuggestions",
    "weblate.addons.resx.ResxUpdateAddon",
    "weblate.addons.autotranslate.AutoTranslateAddon",
    "weblate.addons.yaml.YAMLCustomizeAddon",
    "weblate.addons.cdn.CDNJSAddon",
    # Add-on you want to include
    "weblate.addons.example.ExampleAddon",
)
```

Not: Removing the add-on from the list does not uninstall it from the components. Weblate will crash in that case. Please uninstall add-on from all components prior to removing it from this list.

Ayrıca bakınız:

Eklentiler, `DEFAULT_ADDONS`

2.16.94 WEBLATE_EXPORTERS

4.2 sürümünde geldi.

List of a available exporters offering downloading translations or glossaries in various file formats.

Ayrıca bakınız:

Supported file formats

2.16.95 WEBLATE_FORMATS

3.0 sürümünde geldi.

List of file formats available for use.

Not: The default list already has the common formats.

Ayrıca bakınız:

Supported file formats

2.16.96 WEBLATE_MACHINERY

4.13 sürümünde geldi.

List of machinery services available for use.

Ayrıca bakınız:

Configuring automatic suggestions

2.16.97 WEBLATE_GPG_IDENTITY

3.1 sürümünde geldi.

Identity used by Weblate to sign Git commits, for example:

```
WEBLATE_GPG_IDENTITY = "Weblate <weblate@example.com>"
```

The Weblate GPG keyring is searched for a matching key (home/ .gnupg under `DATA_DIR`). If not found, a key is generated, please check *Signing Git commits with GnuPG* for more details.

Ayrıca bakınız:

Signing Git commits with GnuPG

2.16.98 WEBSITE_REQUIRED

Defines whether *Proje web sitesi* has to be specified when creating a project. Turned on by default as that suits public server setups.

2.17 Sample configuration

The following example is shipped as `weblate/settings_example.py` with Weblate:

```
#
# Copyright © 2012-2022 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program. If not, see <https://www.gnu.org/licenses/>.
#

import os
import platform
from logging.handlers import SysLogHandler

# Title of site to use
SITE_TITLE = "Weblate"

# Site domain
SITE_DOMAIN = ""

# Whether site uses https
ENABLE_HTTPS = False

#
# Django settings for Weblate project.
#

DEBUG = True

ADMINS = (
    # ("Your Name", "your_email@example.com"),
)

MANAGERS = ADMINS

DATABASES = {
    "default": {
        # Use "postgresql" or "mysql".
        "ENGINE": "django.db.backends.postgresql",
        # Database name.
        "NAME": "weblate",
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

# Database user.
"USER": "weblate",
# Name of role to alter to set parameters in PostgreSQL,
# use in case role name is different than user used for authentication.
# "ALTER_ROLE": "weblate",
# Database password.
"PASSWORD": "",
# Set to empty string for localhost.
"HOST": "127.0.0.1",
# Set to empty string for default.
"PORT": "",
# Customizations for databases.
"OPTIONS": {
    # In case of using an older MySQL server,
    # which has MyISAM as a default storage
    # "init_command": "SET storage_engine =INNODB",
    # Uncomment for MySQL older than 5.7:
    # "init_command": "SET sql_mode = 'STRICT_TRANS_TABLES'",
    # Set emoji capable charset for MySQL:
    # "charset": "utf8mb4",
    # Change connection timeout in case you get MySQL gone away error:
    # "connect_timeout": 28800,
},
# Persistent connections
"CONN_MAX_AGE": 0,
# Disable server-side cursors, might be needed with pgbouncer
"DISABLE_SERVER_SIDE_CURSORS": False,
}
}

```

```
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
```

```
# Data directory
```

```
DATA_DIR = os.path.join(BASE_DIR, "data")
```

```
# Local time zone for this installation. Choices can be found here:
```

```
# http://en.wikipedia.org/wiki/List\_of\_tz\_zones\_by\_name
```

```
# although not all choices may be available on all operating systems.
```

```
# In a Windows environment this must be set to your system time zone.
```

```
TIME_ZONE = "UTC"
```

```
# Language code for this installation. All choices can be found here:
```

```
# http://www.i18nguy.com/unicode/language-identifiers.html
```

```
LANGUAGE_CODE = "en-us"
```

```
LANGUAGES = (
```

```
    ("ar", "العربية"),
```

```
    ("az", "Azərbaycan"),
```

```
    ("be", "Беларуская"),
```

```
    ("be@latin", "Biełaruskaja"),
```

```
    ("bg", "Български"),
```

```
    ("br", "Brezhoneg"),
```

```
    ("ca", "Català"),
```

```
    ("cs", "Čeština"),
```

```
    ("da", "Dansk"),
```

```
    ("de", "Deutsch"),
```

```
    ("en", "English"),
```

```
    ("el", "Ελληνικά"),
```

```
    ("en-gb", "English (United Kingdom)"),
```

```
    ("es", "Español"),
```

```
    ("fi", "Suomi"),
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

("fr", "Français"),
("gl", "Galego"),
("he", "עברית"),
("hu", "Magyar"),
("hr", "Hrvatski"),
("id", "Indonesia"),
("is", "Íslenska"),
("it", "Italiano"),
("ja", "日本語"),
("kab", "Taqbaylit"),
("kk", "Қазақ тілі"),
("ko", "한국어"),
("nb", "Norsk bokmål"),
("nl", "Nederlands"),
("pl", "Polski"),
("pt", "Português"),
("pt-br", "Português brasileiro"),
("ro", "Română"),
("ru", "Русский"),
("sk", "Slovenčina"),
("sl", "Slovenščina"),
("sq", "Shqip"),
("sr", "Српски"),
("sr-latn", "Srpski"),
("sv", "Svenska"),
("th", "ไทย"),
("tr", "Türkçe"),
("uk", "Українська"),
("zh-hans", "简体中文"),
("zh-hant", "繁體中文"),
)

SITE_ID = 1

# If you set this to False, Django will make some optimizations so as not
# to load the internationalization machinery.
USE_I18N = True

# If you set this to False, Django will not format dates, numbers and
# calendars according to the current locale.
USE_L10N = True

# If you set this to False, Django will not use timezone-aware datetimes.
USE_TZ = True

# Type of automatic primary key, introduced in Django 3.2
DEFAULT_AUTO_FIELD = "django.db.models.AutoField"

# URL prefix to use, please see documentation for more details
URL_PREFIX = ""

# Absolute filesystem path to the directory that will hold user-uploaded files.
MEDIA_ROOT = os.path.join(DATA_DIR, "media")

# URL that handles the media served from MEDIA_ROOT. Make sure to use a
# trailing slash.
MEDIA_URL = f"{URL_PREFIX}/media/"

# Absolute path to the directory static files should be collected to.
# Don't put anything in this directory yourself; store your static files
# in apps' "static/" subdirectories and in STATICFILES_DIRS.

```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

STATIC_ROOT = os.path.join(DATA_DIR, "static")

# URL prefix for static files.
STATIC_URL = f"{URL_PREFIX}/static/"

# Additional locations of static files
STATICFILES_DIRS = (
    # Put strings here, like "/home/html/static" or "C:/www/django/static".
    # Always use forward slashes, even on Windows.
    # Don't forget to use absolute paths, not relative paths.
)

# List of finder classes that know how to find static files in
# various locations.
STATICFILES_FINDERS = (
    "django.contrib.staticfiles.finders.FileSystemFinder",
    "django.contrib.staticfiles.finders.AppDirectoriesFinder",
    "compressor.finders.CompressorFinder",
)

# Make this unique, and don't share it with anybody.
# You can generate it using weblate/examples/generate-secret-key
SECRET_KEY = ""

TEMPLATES = [
    {
        "BACKEND": "django.template.backends.django.DjangoTemplates",
        "OPTIONS": {
            "context_processors": [
                "django.contrib.auth.context_processors.auth",
                "django.template.context_processors.debug",
                "django.template.context_processors.i18n",
                "django.template.context_processors.request",
                "django.template.context_processors.csrf",
                "django.contrib.messages.context_processors.messages",
                "weblate.trans.context_processors.weblate_context",
            ],
        },
        "APP_DIRS": True,
    }
]

# GitHub username and token for sending pull requests.
# Please see the documentation for more details.
GITHUB_USERNAME = None
GITHUB_TOKEN = None

# GitLab username and token for sending merge requests.
# Please see the documentation for more details.
GITLAB_USERNAME = None
GITLAB_TOKEN = None

# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.email.EmailAuth",
    # "social_core.backends.google.GoogleOAuth2",
    # "social_core.backends.github.GithubOAuth2",
    # "social_core.backends.bitbucket.BitbucketOAuth2",
    # "social_core.backends.suse.OpenSUSEOpenId",
    # "social_core.backends.ubuntu.UbuntuOpenId",

```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

# "social_core.backends.fedora.FedoraOpenId",
# "social_core.backends.facebook.FacebookOAuth2",
"weblate.accounts.auth.WeblateUserBackend",
)

# Custom user model
AUTH_USER_MODEL = "weblate_auth.User"

# Social auth backends setup
SOCIAL_AUTH_GITHUB_KEY = ""
SOCIAL_AUTH_GITHUB_SECRET = ""
SOCIAL_AUTH_GITHUB_SCOPE = ["user:email"]

SOCIAL_AUTH_GITHUB_ORG_KEY = ""
SOCIAL_AUTH_GITHUB_ORG_SECRET = ""
SOCIAL_AUTH_GITHUB_ORG_NAME = ""

SOCIAL_AUTH_GITHUB_TEAM_KEY = ""
SOCIAL_AUTH_GITHUB_TEAM_SECRET = ""
SOCIAL_AUTH_GITHUB_TEAM_ID = ""

SOCIAL_AUTH_BITBUCKET_OAUTH2_KEY = ""
SOCIAL_AUTH_BITBUCKET_OAUTH2_SECRET = ""
SOCIAL_AUTH_BITBUCKET_OAUTH2_VERIFIED_EMAILS_ONLY = True

SOCIAL_AUTH_FACEBOOK_KEY = ""
SOCIAL_AUTH_FACEBOOK_SECRET = ""
SOCIAL_AUTH_FACEBOOK_SCOPE = ["email", "public_profile"]
SOCIAL_AUTH_FACEBOOK_PROFILE_EXTRA_PARAMS = {"fields": "id,name,email"}

SOCIAL_AUTH_GOOGLE_OAUTH2_KEY = ""
SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET = ""

# Social auth settings
SOCIAL_AUTH_PIPELINE = (
    "social_core.pipeline.social_auth.social_details",
    "social_core.pipeline.social_auth.social_uid",
    "social_core.pipeline.social_auth.auth_allowed",
    "social_core.pipeline.social_auth.social_user",
    "weblate.accounts.pipeline.store_params",
    "weblate.accounts.pipeline.verify_open",
    "social_core.pipeline.user.get_username",
    "weblate.accounts.pipeline.require_email",
    "social_core.pipeline.mail.mail_validation",
    "weblate.accounts.pipeline.revoke_mail_code",
    "weblate.accounts.pipeline.ensure_valid",
    "weblate.accounts.pipeline.remove_account",
    "social_core.pipeline.social_auth.associate_by_email",
    "weblate.accounts.pipeline.reauthenticate",
    "weblate.accounts.pipeline.verify_username",
    "social_core.pipeline.user.create_user",
    "social_core.pipeline.social_auth.associate_user",
    "social_core.pipeline.social_auth.load_extra_data",
    "weblate.accounts.pipeline.cleanup_next",
    "weblate.accounts.pipeline.user_full_name",
    "weblate.accounts.pipeline.store_email",
    "weblate.accounts.pipeline.notify_connect",
    "weblate.accounts.pipeline.password_reset",
)
SOCIAL_AUTH_DISCONNECT_PIPELINE = (
    "social_core.pipeline.disconnect.allowed_to_disconnect",

```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

    "social_core.pipeline.disconnect.get_entries",
    "social_core.pipeline.disconnect.revoke_tokens",
    "weblate.accounts.pipeline.cycle_session",
    "weblate.accounts.pipeline.adjust_primary_mail",
    "weblate.accounts.pipeline.notify_disconnect",
    "social_core.pipeline.disconnect.disconnect",
    "weblate.accounts.pipeline.cleanup_next",
)

# Custom authentication strategy
SOCIAL_AUTH_STRATEGY = "weblate.accounts.strategy.WeblateStrategy"

# Raise exceptions so that we can handle them later
SOCIAL_AUTH_RAISE_EXCEPTIONS = True

SOCIAL_AUTH_EMAIL_VALIDATION_FUNCTION = "weblate.accounts.pipeline.send_validation"
SOCIAL_AUTH_EMAIL_VALIDATION_URL = f"{URL_PREFIX}/accounts/email-sent/"
SOCIAL_AUTH_LOGIN_ERROR_URL = f"{URL_PREFIX}/accounts/login/"
SOCIAL_AUTH_EMAIL_FORM_URL = f"{URL_PREFIX}/accounts/email/"
SOCIAL_AUTH_NEW_ASSOCIATION_REDIRECT_URL = f"{URL_PREFIX}/accounts/profile/#account
↪"
SOCIAL_AUTH_PROTECTED_USER_FIELDS = ("email",)
SOCIAL_AUTH_SLUGIFY_USERNAMES = True
SOCIAL_AUTH_SLUGIFY_FUNCTION = "weblate.accounts.pipeline.slugify_username"

# Password validation configuration
AUTH_PASSWORD_VALIDATORS = [
    {
        "NAME": "django.contrib.auth.password_validation.
↪UserAttributeSimilarityValidator" # noqa: E501, pylint: disable =line-too-long
    },
    {
        "NAME": "django.contrib.auth.password_validation.MinimumLengthValidator",
        "OPTIONS": {"min_length": 10},
    },
    {"NAME": "django.contrib.auth.password_validation.CommonPasswordValidator"},
    {"NAME": "django.contrib.auth.password_validation.NumericPasswordValidator"},
    {"NAME": "weblate.accounts.password_validation.CharsPasswordValidator"},
    {"NAME": "weblate.accounts.password_validation.PastPasswordsValidator"},
    # Optional password strength validation by django-zxcvbn-password
    # {
    #     "NAME": "zxcvbn_password.ZXCVBNValidator",
    #     "OPTIONS": {
    #         "min_score": 3,
    #         "user_attributes": ("username", "email", "full_name")
    #     }
    # },
]

# Password hashing (prefer Argon)
PASSWORD_HASHERS = [
    "django.contrib.auth.hashers.Argon2PasswordHasher",
    "django.contrib.auth.hashers.PBKDF2PasswordHasher",
    "django.contrib.auth.hashers.PBKDF2SHA1PasswordHasher",
    "django.contrib.auth.hashers.BCryptSHA256PasswordHasher",
]

# Allow new user registrations
REGISTRATION_OPEN = True

# Shortcut for login required setting

```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

REQUIRE_LOGIN = False

# Middleware
MIDDLEWARE = [
    "weblate.middleware.RedirectMiddleware",
    "weblate.middleware.ProxyMiddleware",
    "django.middleware.security.SecurityMiddleware",
    "django.contrib.sessions.middleware.SessionMiddleware",
    "django.middleware.csrf.CsrfViewMiddleware",
    "weblate.accounts.middleware.AuthenticationMiddleware",
    "django.contrib.messages.middleware.MessageMiddleware",
    "django.middleware.clickjacking.XFrameOptionsMiddleware",
    "social_django.middleware.SocialAuthExceptionMiddleware",
    "weblate.accounts.middleware.RequireLoginMiddleware",
    "weblate.api.middleware.ThrottlingMiddleware",
    "weblate.middleware.SecurityMiddleware",
    "weblate.wladmin.middleware.ManageMiddleware",
]

ROOT_URLCONF = "weblate.urls"

# Django and Weblate apps
INSTALLED_APPS = [
    # Weblate apps on top to override Django locales and templates
    "weblate.addons",
    "weblate.auth",
    "weblate.checks",
    "weblate.formats",
    "weblate.glossary",
    "weblate.machinery",
    "weblate.trans",
    "weblate.lang",
    "weblate_language_data",
    "weblate.memory",
    "weblate.screenshots",
    "weblate.fonts",
    "weblate.accounts",
    "weblate.configuration",
    "weblate.utils",
    "weblate.vcs",
    "weblate.wladmin",
    "weblate.metrics",
    "weblate",
    # Optional: Git exporter
    "weblate.gitexport",
    # Standard Django modules
    "django.contrib.auth",
    "django.contrib.contenttypes",
    "django.contrib.sessions",
    "django.contrib.messages",
    "django.contrib.staticfiles",
    "django.contrib.admin.apps.SimpleAdminConfig",
    "django.contrib.admindocs",
    "django.contrib.sitemaps",
    "django.contrib.humanize",
    # Third party Django modules
    "social_django",
    "crispy_forms",
    "compressor",
    "rest_framework",
    "rest_framework.authtoken",

```

(sonraki sayfaya devam)

```

    "django_filters",
]

# Custom exception reporter to include some details
DEFAULT_EXCEPTION_REPORTER_FILTER = "weblate.trans.debug.
↳ WeblateExceptionReporterFilter"

# Default logging of Weblate messages
# - to syslog in production (if available)
# - otherwise to console
# - you can also choose "logfile" to log into separate file
# after configuring it below

# Detect if we can connect to syslog
HAVE_SYSLOG = False
if platform.system() != "Windows":
    try:
        handler = SysLogHandler(address="/dev/log", facility=SysLogHandler.LOG_
↳ LOCAL2)
        handler.close()
        HAVE_SYSLOG = True
    except OSError:
        HAVE_SYSLOG = False

if DEBUG or not HAVE_SYSLOG:
    DEFAULT_LOG = "console"
else:
    DEFAULT_LOG = "syslog"
DEFAULT_LOGLEVEL = "DEBUG" if DEBUG else "INFO"

# A sample logging configuration. The only tangible logging
# performed by this configuration is to send an email to
# the site admins on every HTTP 500 error when DEBUG=False.
# See http://docs.djangoproject.com/en/stable/topics/logging for
# more details on how to customize your logging configuration.
LOGGING = {
    "version": 1,
    "disable_existing_loggers": True,
    "filters": {"require_debug_false": {"()": "django.utils.log.RequireDebugFalse"}
↳ },
    "formatters": {
        "syslog": {"format": "weblate[%(process)d]: %(levelname)s %(message)s"},
        "simple": {"format": "[%asctime)s: %(levelname)s/%(process)s] %(message)s
↳ "},
        "logfile": {"format": "%(asctime)s %(levelname)s %(message)s"},
        "django.server": {
            "()": "django.utils.log.ServerFormatter",
            "format": "[%server_time)s] %(message)s",
        },
    },
    "handlers": {
        "mail_admins": {
            "level": "ERROR",
            "filters": ["require_debug_false"],
            "class": "django.utils.log.AdminEmailHandler",
            "include_html": True,
        },
        "console": {
            "level": "DEBUG",
            "class": "logging.StreamHandler",
            "formatter": "simple",

```

(önceki sayfadan devam)

```

    },
    "django.server": {
        "level": "INFO",
        "class": "logging.StreamHandler",
        "formatter": "django.server",
    },
    "syslog": {
        "level": "DEBUG",
        "class": "logging.handlers.SysLogHandler",
        "formatter": "syslog",
        "address": "/dev/log",
        "facility": SysLogHandler.LOG_LOCAL2,
    },
    # Logging to a file
    # "logfile": {
    #     "level": "DEBUG",
    #     "class": "logging.handlers.RotatingFileHandler",
    #     "filename": "/var/log/weblate/weblate.log",
    #     "maxBytes": 100000,
    #     "backupCount": 3,
    #     "formatter": "logfile",
    # },
    },
    "loggers": {
        "django.request": {
            "handlers": ["mail_admins", DEFAULT_LOG],
            "level": "ERROR",
            "propagate": True,
        },
        "django.server": {
            "handlers": ["django.server"],
            "level": "INFO",
            "propagate": False,
        },
    },
    # Logging database queries
    # "django.db.backends": {
    #     "handlers": [DEFAULT_LOG],
    #     "level": "DEBUG",
    # },
    "weblate": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
    # Logging VCS operations
    "weblate.vcs": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
    # Python Social Auth
    "social": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
    # Django Authentication Using LDAP
    "django_auth_ldap": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
    # SAML IdP
    "djangosaml2idp": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
    },
}

# Remove syslog setup if it's not present
if not HAVE_SYSLOG:
    del LOGGING["handlers"]["syslog"]

# List of machine translations
MT_SERVICES = (
    #     "weblate.machinery.apertium.ApertiumAPYTranslation",
    #     "weblate.machinery.baidu.BaiduTranslation",
    #     "weblate.machinery.deepl.DeepLTranslation",
    #     "weblate.machinery.glosbe.GlosbeTranslation",

```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

# "weblate.machinery.google.GoogleTranslation",
# "weblate.machinery.googlev3.GoogleV3Translation",
# "weblate.machinery.libretranslate.LibreTranslateTranslation",
# "weblate.machinery.microsoft.MicrosoftCognitiveTranslation",
# "weblate.machinery.microsoftterminology.MicrosoftTerminologyService",
# "weblate.machinery.modernmt.ModernMTTranslation",
# "weblate.machinery.mymemory.MyMemoryTranslation",
# "weblate.machinery.netease.NeteaseSightTranslation",
# "weblate.machinery.tmserver.AmagamaTranslation",
# "weblate.machinery.tmserver.TMServerTranslation",
# "weblate.machinery.yandex.YandexTranslation",
# "weblate.machinery.saptranslationhub.SAPTranslationHub",
# "weblate.machinery.youdao.YoudaoTranslation",
"weblate.machinery.weblatetm.WeblateTranslation",
"weblate.memory.machine.WeblateMemory",
)

# Machine translation API keys

# URL of the Apertium APY server
MT_APERTIUM_APY = None

# DeepL API key
MT_DEEPL_KEY = None

# LibreTranslate
MT_LIBRETRANSLATE_API_URL = None
MT_LIBRETRANSLATE_KEY = None

# Microsoft Cognitive Services Translator API, register at
# https://portal.azure.com/
MT_MICROSOFT_COGNITIVE_KEY = None
MT_MICROSOFT_REGION = None

# ModernMT
MT_MODERNMT_KEY = None

# MyMemory identification email, see
# https://mymemory.translated.net/doc/spec.php
MT_MYMEMORY_EMAIL = None

# Optional MyMemory credentials to access private translation memory
MT_MYMEMORY_USER = None
MT_MYMEMORY_KEY = None

# Google API key for Google Translate API v2
MT_GOOGLE_KEY = None

# Google Translate API3 credentials and project id
MT_GOOGLE_CREDENTIALS = None
MT_GOOGLE_PROJECT = None

# Baidu app key and secret
MT_BAIDU_ID = None
MT_BAIDU_SECRET = None

# Youdao Zhiyun app key and secret
MT_YOUDAO_ID = None
MT_YOUDAO_SECRET = None

# Netease Sight (Jianwai) app key and secret

```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

MT_NETEASE_KEY = None
MT_NETEASE_SECRET = None

# API key for Yandex Translate API
MT_YANDEX_KEY = None

# tmserver URL
MT_TMSERVER = None

# SAP Translation Hub
MT_SAP_BASE_URL = None
MT_SAP_SANDBOX_APIKEY = None
MT_SAP_USERNAME = None
MT_SAP_PASSWORD = None
MT_SAP_USE_MT = True

# Use HTTPS when creating redirect URLs for social authentication, see
# documentation for more details:
# https://python-social-auth-docs.readthedocs.io/en/latest/configuration/settings.
# ↪html#processing-redirects-and-urlopen
SOCIAL_AUTH_REDIRECT_IS_HTTPS = ENABLE_HTTPS

# Make CSRF cookie HttpOnly, see documentation for more details:
# https://docs.djangoproject.com/en/1.11/ref/settings/#csrf-cookie-httponly
CSRF_COOKIE_HTTPONLY = True
CSRF_COOKIE_SECURE = ENABLE_HTTPS
# Store CSRF token in session
CSRF_USE_SESSIONS = True
# Customize CSRF failure view
CSRF_FAILURE_VIEW = "weblate.trans.views.error.csrf_failure"
SESSION_COOKIE_SECURE = ENABLE_HTTPS
SESSION_COOKIE_HTTPONLY = True
# SSL redirect
SECURE_SSL_REDIRECT = ENABLE_HTTPS
# Sent referrrrer only for same origin links
SECURE_REFERRER_POLICY = "same-origin"
# SSL redirect URL exemption list
SECURE_REDIRECT_EXEMPT = (r"healthz/$",) # Allowing HTTP access to health check
# Session cookie age (in seconds)
SESSION_COOKIE_AGE = 1000
SESSION_COOKIE_AGE_AUTHENTICATED = 1209600
SESSION_COOKIE_SAMESITE = "Lax"
# Increase allowed upload size
DATA_UPLOAD_MAX_MEMORY_SIZE = 50000000

# Apply session coookie settings to language cookie as ewll
LANGUAGE_COOKIE_SECURE = SESSION_COOKIE_SECURE
LANGUAGE_COOKIE_HTTPONLY = SESSION_COOKIE_HTTPONLY
LANGUAGE_COOKIE_AGE = SESSION_COOKIE_AGE_AUTHENTICATED * 10
LANGUAGE_COOKIE_SAMESITE = SESSION_COOKIE_SAMESITE

# Some security headers
SECURE_BROWSER_XSS_FILTER = True
X_FRAME_OPTIONS = "DENY"
SECURE_CONTENT_TYPE_NOSNIFF = True

# Optionally enable HSTS
SECURE_HSTS_SECONDS = 31536000 if ENABLE_HTTPS else 0
SECURE_HSTS_PRELOAD = ENABLE_HTTPS
SECURE_HSTS_INCLUDE_SUBDOMAINS = ENABLE_HTTPS

```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

# HTTPS detection behind reverse proxy
SECURE_PROXY_SSL_HEADER = None

# URL of login
LOGIN_URL = f"{URL_PREFIX}/accounts/login/"

# URL of logout
LOGOUT_URL = f"{URL_PREFIX}/accounts/logout/"

# Default location for login
LOGIN_REDIRECT_URL = f"{URL_PREFIX}/"

# Anonymous user name
ANONYMOUS_USER_NAME = "anonymous"

# Reverse proxy settings
IP_PROXY_HEADER = "HTTP_X_FORWARDED_FOR"
IP_BEHIND_REVERSE_PROXY = False
IP_PROXY_OFFSET = 0

# Sending HTML in mails
EMAIL_SEND_HTML = True

# Subject of emails includes site title
EMAIL_SUBJECT_PREFIX = f"[{SITE_TITLE}] "

# Enable remote hooks
ENABLE_HOOKS = True

# By default the length of a given translation is limited to the length of
# the source string * 10 characters. Set this option to False to allow longer
# translations (up to 10.000 characters)
LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH = True

# Use simple language codes for default language/country combinations
SIMPLIFY_LANGUAGES = True

# Render forms using bootstrap
CRISPY_TEMPLATE_PACK = "bootstrap3"

# List of quality checks
# CHECK_LIST = (
#     "weblate.checks.same.SameCheck",
#     "weblate.checks.chars.BeginNewlineCheck",
#     "weblate.checks.chars.EndNewlineCheck",
#     "weblate.checks.chars.BeginSpaceCheck",
#     "weblate.checks.chars.EndSpaceCheck",
#     "weblate.checks.chars.DoubleSpaceCheck",
#     "weblate.checks.chars.EndStopCheck",
#     "weblate.checks.chars.EndColonCheck",
#     "weblate.checks.chars.EndQuestionCheck",
#     "weblate.checks.chars.EndExclamationCheck",
#     "weblate.checks.chars.EndEllipsisCheck",
#     "weblate.checks.chars.EndSemicolonCheck",
#     "weblate.checks.chars.MaxLengthCheck",
#     "weblate.checks.chars.KashidaCheck",
#     "weblate.checks.chars.PunctuationSpacingCheck",
#     "weblate.checks.format.PythonFormatCheck",
#     "weblate.checks.format.PythonBraceFormatCheck",
#     "weblate.checks.format.PHPFormatCheck",
#     "weblate.checks.format.CFormatCheck",

```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

# "weblate.checks.format.PperlFormatCheck",
# "weblate.checks.format.JavaScriptFormatCheck",
# "weblate.checks.format.LuaFormatCheck",
# "weblate.checks.format.ObjectPascalFormatCheck",
# "weblate.checks.format.SchemeFormatCheck",
# "weblate.checks.format.CSharpFormatCheck",
# "weblate.checks.format.JavaFormatCheck",
# "weblate.checks.format.JavaMessageFormatCheck",
# "weblate.checks.format.PercentPlaceholdersCheck",
# "weblate.checks.format.VueFormattingCheck",
# "weblate.checks.format.I18NextInterpolationCheck",
# "weblate.checks.format.ESTemplateLiteralsCheck",
# "weblate.checks.angularjs.AngularJSInterpolationCheck",
# "weblate.checks.icu.ICUMessageFormatCheck",
# "weblate.checks.icu.ICUSourceCheck",
# "weblate.checks.qt.QtFormatCheck",
# "weblate.checks.qt.QtPluralCheck",
# "weblate.checks.ruby.RubyFormatCheck",
# "weblate.checks.consistency.PluralsCheck",
# "weblate.checks.consistency.SamePluralsCheck",
# "weblate.checks.consistency.ConsistencyCheck",
# "weblate.checks.consistency.TranslatedCheck",
# "weblate.checks.chars.EscapedNewLineCountingCheck",
# "weblate.checks.chars.NewLineCountCheck",
# "weblate.checks.markup.BBCodeCheck",
# "weblate.checks.chars.ZeroWidthSpaceCheck",
# "weblate.checks.render.MaxSizeCheck",
# "weblate.checks.markup.XMLValidityCheck",
# "weblate.checks.markup.XMLTagsCheck",
# "weblate.checks.markup.MarkdownRefLinkCheck",
# "weblate.checks.markup.MarkdownLinkCheck",
# "weblate.checks.markup.MarkdownSyntaxCheck",
# "weblate.checks.markup.URLCheck",
# "weblate.checks.markup.SafeHTMLCheck",
# "weblate.checks.placeholders.PlaceholderCheck",
# "weblate.checks.placeholders.RegexCheck",
# "weblate.checks.duplicate.DuplicateCheck",
# "weblate.checks.source.OptionalPluralCheck",
# "weblate.checks.source.EllipsisCheck",
# "weblate.checks.source.MultipleFailingCheck",
# "weblate.checks.source.LongUntranslatedCheck",
# "weblate.checks.format.MultipleUnnamedFormatsCheck",
# "weblate.checks.glossary.GlossaryCheck",
# )

# List of automatic fixups
# AUTOFIX_LIST = (
#     "weblate.trans.autofixes.whitespace.SameBookendingWhitespace",
#     "weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis",
#     "weblate.trans.autofixes.chars.RemoveZeroSpace",
#     "weblate.trans.autofixes.chars.RemoveControlChars",
# )

# List of enabled addons
# WEBLATE_ADDONS = (
#     "weblate.addons.gettext.GenerateMoAddon",
#     "weblate.addons.gettext.UpdateLinguasAddon",
#     "weblate.addons.gettext.UpdateConfigureAddon",
#     "weblate.addons.gettext.MsgmergeAddon",
#     "weblate.addons.gettext.GettextCustomizeAddon",
#     "weblate.addons.gettext.GettextAuthorComments",

```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

# "weblate.addons.cleanup.CleanupAddon",
# "weblate.addons.cleanup.RemoveBlankAddon",
# "weblate.addons.consistency.LanguaugeConsistencyAddon",
# "weblate.addons.discovery.DiscoveryAddon",
# "weblate.addons.autotranslate.AutoTranslateAddon",
# "weblate.addons.flags.SourceEditAddon",
# "weblate.addons.flags.TargetEditAddon",
# "weblate.addons.flags.SameEditAddon",
# "weblate.addons.flags.BulkEditAddon",
# "weblate.addons.generate.GenerateFileAddon",
# "weblate.addons.generate.PseudolocaleAddon",
# "weblate.addons.generate.PrefillAddon",
# "weblate.addons.json.JSONCustomizeAddon",
# "weblate.addons.properties.PropertiesSortAddon",
# "weblate.addons.git.GitSquashAddon",
# "weblate.addons.removal.RemoveComments",
# "weblate.addons.removal.RemoveSuggestions",
# "weblate.addons.resx.ResxUpdateAddon",
# "weblate.addons.yaml.YAMLCustomizeAddon",
# "weblate.addons.cdn.CDNJSAddon",
# )

# E-mail address that error messages come from.
SERVER_EMAIL = "noreply@example.com"

# Default email address to use for various automated correspondence from
# the site managers. Used for registration emails.
DEFAULT_FROM_EMAIL = "noreply@example.com"

# List of URLs your site is supposed to serve
ALLOWED_HOSTS = ["*"]

# Configuration for caching
CACHES = {
    "default": {
        "BACKEND": "django_redis.cache.RedisCache",
        "LOCATION": "redis://127.0.0.1:6379/1",
        # If redis is running on same host as Weblate, you might
        # want to use unix sockets instead:
        # "LOCATION": "unix:///var/run/redis/redis.sock?db=1",
        "OPTIONS": {
            "CLIENT_CLASS": "django_redis.client.DefaultClient",
            "PARSER_CLASS": "redis.connection.HiredisParser",
            # If you set password here, adjust CELERY_BROKER_URL as well
            "PASSWORD": None,
            "CONNECTION_POOL_KWARGS": {},
        },
        "KEY_PREFIX": "weblate",
    },
    "avatar": {
        "BACKEND": "django.core.cache.backends.filebased.FileBasedCache",
        "LOCATION": os.path.join(DATA_DIR, "avatar-cache"),
        "TIMEOUT": 86400,
        "OPTIONS": {"MAX_ENTRIES": 1000},
    },
}

# Store sessions in cache
SESSION_ENGINE = "django.contrib.sessions.backends.cache"
# Store messages in session
MESSAGE_STORAGE = "django.contrib.messages.storage.session.SessionStorage"

```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```

# REST framework settings for API
REST_FRAMEWORK = {
    # Use Django's standard `django.contrib.auth` permissions,
    # or allow read-only access for unauthenticated users.
    "DEFAULT_PERMISSION_CLASSES": [
        # Require authentication for login required sites
        "rest_framework.permissions.IsAuthenticated"
        if REQUIRE_LOGIN
        else "rest_framework.permissions.IsAuthenticatedOrReadOnly"
    ],
    "DEFAULT_AUTHENTICATION_CLASSES": (
        "rest_framework.authentication.TokenAuthentication",
        "weblate.api.authentication.BearerAuthentication",
        "rest_framework.authentication.SessionAuthentication",
    ),
    "DEFAULT_THROTTLE_CLASSES": (
        "weblate.api.throttling.UserRateThrottle",
        "weblate.api.throttling.AnonRateThrottle",
    ),
    "DEFAULT_THROTTLE_RATES": {"anon": "100/day", "user": "5000/hour"},
    "DEFAULT_PAGINATION_CLASS": ("rest_framework.pagination.PageNumberPagination"),
    "PAGE_SIZE": 20,
    "VIEW_DESCRIPTION_FUNCTION": "weblate.api.views.get_view_description",
    "UNAUTHENTICATED_USER": "weblate.auth.models.get_anonymous",
}

# Fonts CDN URL
FONTS_CDN_URL = None

# Django compressor offline mode
COMPRESS_OFFLINE = False
COMPRESS_OFFLINE_CONTEXT = [
    {"fonts_cdn_url": FONTS_CDN_URL, "STATIC_URL": STATIC_URL, "LANGUAGE_BIDI": ↵
↵ True},
    {"fonts_cdn_url": FONTS_CDN_URL, "STATIC_URL": STATIC_URL, "LANGUAGE_BIDI": ↵
↵ False},
]

# Require login for all URLs
if REQUIRE_LOGIN:
    LOGIN_REQUIRED_URLS = (r"/(.*)$",)

# In such case you will want to include some of the exceptions
# LOGIN_REQUIRED_URLS_EXCEPTIONS = (
#     rf"{URL_PREFIX}/accounts/(.*)$", # Required for login
#     rf"{URL_PREFIX}/admin/login/(.*)$", # Required for admin login
#     rf"{URL_PREFIX}/static/(.*)$", # Required for development mode
#     rf"{URL_PREFIX}/widgets/(.*)$", # Allowing public access to widgets
#     rf"{URL_PREFIX}/data/(.*)$", # Allowing public access to data exports
#     rf"{URL_PREFIX}/hooks/(.*)$", # Allowing public access to notification hooks
#     rf"{URL_PREFIX}/healthz/$", # Allowing public access to health check
#     rf"{URL_PREFIX}/api/(.*)$", # Allowing access to API
#     rf"{URL_PREFIX}/js/i18n/$", # JavaScript localization
#     rf"{URL_PREFIX}/contact/$", # Optional for contact form
#     rf"{URL_PREFIX}/legal/(.*)$", # Optional for legal app
#     rf"{URL_PREFIX}/avatar/(.*)$", # Optional for avatars
# )

# Silence some of the Django system checks
SILENCED_SYSTEM_CHECKS = [

```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
# We have modified django.contrib.auth.middleware.AuthenticationMiddleware
# as weblate.accounts.middleware.AuthenticationMiddleware
"admin.E408"
]

# Celery worker configuration for testing
# CELERY_TASK_ALWAYS_EAGER = True
# CELERY_BROKER_URL = "memory://"
# CELERY_TASK_EAGER_PROPAGATES = True
# Celery worker configuration for production
CELERY_TASK_ALWAYS_EAGER = False
CELERY_BROKER_URL = "redis://localhost:6379"
CELERY_RESULT_BACKEND = CELERY_BROKER_URL

# Celery settings, it is not recommended to change these
CELERY_WORKER_MAX_MEMORY_PER_CHILD = 200000
CELERY_BEAT_SCHEDULE_FILENAME = os.path.join(DATA_DIR, "celery", "beat-schedule")
CELERY_TASK_ROUTES = {
    "weblate.trans.tasks.auto_translate*": {"queue": "translate"},
    "weblate.accounts.tasks.notify_*": {"queue": "notify"},
    "weblate.accounts.tasks.send_mails": {"queue": "notify"},
    "weblate.utils.tasks.settings_backup": {"queue": "backup"},
    "weblate.utils.tasks.database_backup": {"queue": "backup"},
    "weblate.wladmin.tasks.backup": {"queue": "backup"},
    "weblate.wladmin.tasks.backup_service": {"queue": "backup"},
    "weblate.memory.tasks.*": {"queue": "memory"},
}

# Enable plain database backups
DATABASE_BACKUP = "plain"

# Enable auto updating
AUTO_UPDATE = False

# PGP commits signing
WEBLATE_GPG_IDENTITY = None

# Third party services integration
MATOMO_SITE_ID = None
MATOMO_URL = None
GOOGLE_ANALYTICS_ID = None
SENTRY_DSN = None
SENTRY_ENVIRONMENT = SITE_DOMAIN
AKISMET_API_KEY = None
```

2.18 Management commands

Not: Running management commands under a different user than the one running your webserver can result in files getting wrong permissions, please check *Filesystem permissions* for more details.

You will find basic management commands (available as `./manage.py` in the Django sources, or as an extended set in a script called **weblate** installable atop Weblate).

2.18.1 Invoking management commands

As mentioned before, invocation depends on how you installed Weblate.

If using virtualenv for Weblate, you can either specify the full path to **weblate**, or activate the virtualenv prior to invoking it:

```
# Direct invocation
~/weblate-env/bin/weblate

# Activating virtualenv adds it to search path
. ~/weblate-env/bin/activate
weblate
```

If you are using source code directly (either from a tarball or Git checkout), the management script is `./manage.py` available in the Weblate sources. To run it:

```
python ./manage.py list_versions
```

If you've installed Weblate using the pip or pip3 installer, or by using the `./setup.py` script, the **weblate** is installed to your path (or virtualenv path), from where you can use it to control Weblate:

```
weblate list_versions
```

For the Docker image, the script is installed like above, and you can run it using **docker exec**:

```
docker exec --user weblate <container> weblate list_versions
```

For **docker-compose** the process is similar, you just have to use **docker-compose exec**:

```
docker-compose exec --user weblate weblate weblate list_versions
```

In case you need to pass it a file, you can temporary add a volume:

```
docker-compose exec --user weblate /tmp:/tmp weblate weblate importusers /tmp/
↪users.json
```

Ayrıca bakınız:

Installing using Docker, Installing on Debian and Ubuntu, Installing on SUSE and openSUSE, Installing on RedHat, Fedora and CentOS, Installing from sources

2.18.2 add_suggestions

weblate add_suggestions <project> <component> <language> <file>

2.5 sürümünde geldi.

Imports a translation from the file to use as a suggestion for the given translation. It skips duplicated translations; only different ones are added.

--author USER@EXAMPLE.COM

E-mail of author for the suggestions. This user has to exist prior to importing (you can create one in the admin interface if needed).

Example:

```
weblate --author michal@cihar.com add_suggestions weblate application cs /tmp/
↪suggestions-cs.po
```

2.18.3 auto_translate

weblate auto_translate <project> <component> <language>

2.5 sürümünde geldi.

4.6 sürümünde değişti: Çeviri modu için parametre eklendi.

Performs automatic translation based on other component translations.

--source PROJECT/COMPONENT

Specifies the component to use as source available for translation. If not specified all components in the project are used.

--user USERNAME

Specify username listed as author of the translations. “Anonymous user” is used if not specified.

--overwrite

Whether to overwrite existing translations.

--inconsistent

Whether to overwrite existing translations that are inconsistent (see *Tutarsız*).

--add

Automatically add language if a given translation does not exist.

--mt MT

Use machine translation instead of other components as machine translations.

--threshold THRESHOLD

Similarity threshold for machine translation, defaults to 80.

--mode MODE

Specify translation mode, default is `translate` but `fuzzy` or `suggest` can be used.

Example:

```
weblate auto_translate --user nijel --inconsistent --source weblate/application_
↪weblate website cs
```

Ayrıca bakınız:

Otomatik çeviri

2.18.4 celery_queues

weblate celery_queues

3.7 sürümünde geldi.

Displays length of Celery task queues.

2.18.5 checkgit

weblate checkgit <project|project/component>

Prints current state of the back-end Git repository.

You can either define which project or component to update (for example `weblate/application`), or use `--all` to update all existing components.

2.18.6 commitgit

weblate commitgit <project|project/component>

Commits any possible pending changes to the back-end Git repository.

You can either define which project or component to update (for example `weblate/application`), or use `--all` to update all existing components.

2.18.7 commit_pending

weblate commit_pending <project|project/component>

Commits pending changes older than a given age.

You can either define which project or component to update (for example `weblate/application`), or use `--all` to update all existing components.

--age HOURS

Age in hours for committing. If not specified the value configured in *Component configuration* is used.

Not: This is automatically performed in the background by Weblate, so there no real need to invoke this manually, besides forcing an earlier commit than specified by *Component configuration*.

Ayrıca bakınız:

Running maintenance tasks, `COMMIT_PENDING_HOURS`

2.18.8 cleanuptrans

weblate cleanuptrans

Cleans up orphaned checks and translation suggestions. There is normally no need to run this manually, as the cleanups happen automatically in the background.

Ayrıca bakınız:

Running maintenance tasks

2.18.9 cleanup_ssh_keys

weblate cleanup_ssh_keys

4.9.1 sürümünde geldi.

Performs cleanup of stored SSH host keys:

- Removes deprecated RSA keys for GitHub which might cause issues connecting to GitHub.
- Removes duplicate entries in host keys.

Ayrıca bakınız:

SSH repositories

2.18.10 createadmin

weblate createadmin

Creates an `admin` account with a random password, unless it is specified.

--password PASSWORD

Provides a password on the command-line, to not generate a random one.

--no-password

Do not set password, this can be useful with `--update`.

--username USERNAME

Use the given name instead of `admin`.

--email USER@EXAMPLE.COM

Specify the admin e-mail address.

--name

Specify the admin name (visible).

--update

Update the existing user (you can use this to change passwords).

2.9 sürümünde değişti: Added parameters `--username`, `--email`, `--name` and `--update`.

2.18.11 dump_memory

weblate dump_memory

2.20 sürümünde geldi.

Export a JSON file containing Weblate Translation Memory content.

Ayrıca bakınız:

Çeviri Belleği, Weblate Çeviri Belleği Şeması

2.18.12 dumpuserdata

weblate dumpuserdata <file.json>

Dumps userdata to a file for later use by *importuserdata*.

İpucu: This comes in handy when migrating or merging Weblate instances.

2.18.13 import_demo

weblate import_demo

4.1 sürümünde geldi.

Creates a demo project with components based on <<https://github.com/WeblateOrg/demo>>. Make sure the celery tasks are running before running this command.

This can be useful when developing Weblate.

2.18.14 import_json

weblate import_json <json-file>

2.7 sürümünde geldi.

Batch import of components based on JSON data.

The imported JSON file structure pretty much corresponds to the component object (see *GET /api/components/(string:project)/(string:component)/*). You have to include the name and filemask fields.

--project PROJECT

Specifies where the components will be imported from.

--main-component COMPONENT

Use the given VCS repository from this component for all of them.

--ignore

Skip (already) imported components.

--update

Update (already) imported components.

2.9 sürümünde değişti: The parameters *--ignore* and *--update* are there to deal with already imported components.

Example of JSON file:

```
[
  {
    "slug": "po",
    "name": "Gettext PO",
    "file_format": "po",
    "filemask": "po/*.po",
    "new_lang": "none"
  },
  {
    "name": "Android",
    "filemask": "android/values-*/strings.xml",
```

(sonraki sayfaya devam)

```
"template": "android/values/strings.xml",
"repo": "weblate://test/test",
"file_format": "aresource"
}
]
```

Ayrıca bakınız:

import_memory

2.18.15 import_memory

weblate import_memory <file>

2.20 sürümünde geldi.

Imports a TMX or JSON file into the Weblate translation memory.

--language-map LANGMAP

Allows mapping languages in the TMX to the Weblate translation memory. The language codes are mapped after normalization usually done by Weblate.

--language-map en_US:en will for example import all en_US strings as en ones.

This can be useful in case your TMX file locales happen not to match what you use in Weblate.

Ayrıca bakınız:

Çeviri Belleği, Weblate Çeviri Belleği Şeması

2.18.16 import_project

weblate import_project <project> <gitrepo> <branch> <filemask>

3.0 sürümünde değişti: The import_project command is now based on the *Bileşen keşfi* add-on, leading to some changes in behavior and what parameters are accepted.

Batch imports components into project based on the file mask.

<project> names an existing project, into which the components are to be imported.

The <gitrepo> defines the Git repository URL to use, and <branch> signifies the Git branch. To import additional translation components from an existing Weblate component, use a *weblate://<project>/<component>* URL for the <gitrepo>.

The <filemask> defines file discovery for the repository. It can be either be made simple using wildcards, or it can use the full power of regular expressions.

The simple matching uses ** for component name and * for language, for example: **/* .po

The regular expression has to contain groups named *component* and *language*. For example: (?P<language>[^/]*) / (?P<component>[^-/*]*) \.po

The import matches existing components based on files and adds the ones that do not exist. It does not change already existing ones.

--name-template TEMPLATE

Customize the name of a component using Django template syntax.

For example: Documentation: {{ component }}

--base-file-template TEMPLATE

Customize the base file for monolingual translations.

For example: `{{ component }}/res/values/string.xml`

--new-base-template TEMPLATE

Customize the base file for addition of new translations.

For example: `{{ component }}/ts/en.ts`

--file-format FORMAT

You can also specify the file format to use (see *Supported file formats*), the default is auto-detection.

--language-regex REGEX

You can specify language filtering (see *Component configuration*) with this parameter. It has to be a valid regular expression.

--main-component

You can specify which component will be chosen as the main one—the one actually containing the VCS repository.

--license NAME

Specify the overall, project or component translation license.

--license-url URL

Specify the URL where the translation license is to be found.

--vcs NAME

In case you need to specify which version control system to use, you can do it here. The default version control is Git.

To give you some examples, let's try importing two projects.

First The Debian Handbook translations, where each language has separate a folder with the translations of each chapter:

```
weblate import_project \
  debian-handbook \
  git://anonscm.debian.org/debian-handbook/debian-handbook.git \
  squeeze/master \
  '*/**.po'
```

Then the Tanaguru tool, where the file format needs be specified, along with the base file template, and how all components and translations are located in single folder:

```
weblate import_project \
  --file-format=properties \
  --base-file-template=web-app/tgol-web-app/src/main/resources/i18n/%s-I18N.
↪properties \
  tanaguru \
  https://github.com/Tanaguru/Tanaguru \
  master \
  web-app/tgol-web-app/src/main/resources/i18n/**-I18N_*.properties
```

More complex example of parsing of filenames to get the correct component and language out of a filename like `src/security/Numerous_security_holes_in_0.10.1.de.po`:

```
weblate import_project \
  tails \
  git://git.tails.boum.org/tails master \
  'wiki/src/security/(?P<component>.*)(?P<language>[^\.]*)\.po$'
```

Filtering only translations in a chosen language:

```
./manage import_project \
  --language-regex '^(cs|sk)$' \
  weblate \
  https://github.com/WeblateOrg/weblate.git \
  'weblate/locale/*/LC_MESSAGES/**/*.po'
```

Importing Sphinx documentation split to multiple files:

```
$ weblate import_project --name-template 'Documentation: %s' \
  --file-format po \
  project https://github.com/project/docs.git master \
  'docs/locale/*/LC_MESSAGES/**/*.po'
```

Importing Sphinx documentation split to multiple files and directories:

```
$ weblate import_project --name-template 'Directory 1: %s' \
  --file-format po \
  project https://github.com/project/docs.git master \
  'docs/locale/*/LC_MESSAGES/dir1/**/*.po'
$ weblate import_project --name-template 'Directory 2: %s' \
  --file-format po \
  project https://github.com/project/docs.git master \
  'docs/locale/*/LC_MESSAGES/dir2/**/*.po'
```

Ayrıca bakınız:

More detailed examples can be found in the starting chapter, alternatively you might want to use *import_json*.

2.18.17 importuserdata

weblate importuserdata <file.json>

Imports user data from a file created by *dumpuserdata*

2.18.18 importusers

weblate importusers --check <file.json>

Imports users from JSON dump of the Django auth_users database.

--check

With this option it will just check whether a given file can be imported and report possible conflicts arising from usernames or e-mails.

You can dump users from the existing Django installation using:

```
weblate dumpdata auth.User > users.json
```

2.18.19 install_addon

3.2 sürümünde geldi.

weblate install_addon --addon ADDON <project|project/component>

Installs an add-on to a set of components.

--addon ADDON

Name of the add-on to install. For example `weblate.gettext.customize`.

--configuration CONFIG

Bir eklentinin JSON biçimindeki yapılandırması.

--update

Mevcut eklenti yapılandırmasını güncelle.

You can either define which project or component to install the add-on in (for example `weblate/application`), or use `--all` to include all existing components.

To install *Gettext çıktısını özelleştir* for all components:

```
weblate install_addon --addon weblate.gettext.customize --config '{"width": -1}' --
↪update --all
```

Ayrıca bakınız:

Eklentiler

2.18.20 list_languages

weblate list_languages <locale>

Lists supported languages in MediaWiki markup - language codes, English names and localized names.

This is used to generate `<https://wiki.110n.cz/Slovn%C3%ADk_s_n%C3%A1lvy_jazyk%C5%AF>`.

2.18.21 list_translators

weblate list_translators <project|project/component>

Lists translators by contributed language for the given project:

```
[French]
Jean Dupont <jean.dupont@example.com>
[English]
John Doe <jd@example.com>
```

--language-code

List names by language code instead of language name.

You can either define which project or component to use (for example `weblate/application`), or use `--all` to list translators from all existing components.

2.18.22 list_versions

weblate list_versions

Lists all Weblate dependencies and their versions.

2.18.23 loadpo

weblate loadpo <project|project/component>

Reloads translations from disk (for example in case you have done some updates in the VCS repository).

--force

Force update, even if the files should be up-to-date.

--lang LANGUAGE

Limit processing to a single language.

You can either define which project or component to update (for example `weblate/application`), or use `--all` to update all existing components.

Not: You seldom need to invoke this, Weblate will automatically load changed files for every VCS update. This is needed in case you manually changed an underlying Weblate VCS repository or in some special cases following an upgrade.

2.18.24 lock_translation

weblate lock_translation <project|project/component>

Prevents further translation of a component.

İpucu: Useful in case you want to do some maintenance on the underlying repository.

You can either define which project or component to update (for example `weblate/application`), or use `--all` to update all existing components.

Ayrıca bakınız:

`unlock_translation`

2.18.25 move_language

weblate move_language source target

3.0 sürümünde geldi.

Allows you to merge language content. This is useful when updating to a new version which contains aliases for previously unknown languages that have been created with the *(generated)* suffix. It moves all content from the *source* language to the *target* one.

Example:

```
weblate move_language cze cs
```

After moving the content, you should check whether there is anything left (this is subject to race conditions when somebody updates the repository meanwhile) and remove the *(generated)* language.

2.18.26 pushgit

weblate pushgit <project|project/component>

Pushes committed changes to the upstream VCS repository.

--force-commit

Force commits any pending changes, prior to pushing.

You can either define which project or component to update (for example `weblate/application`), or use `--all` to update all existing components.

Not: Weblate pushes changes automatically if *İşlemeye yolla* in *Component configuration* is turned on, which is the default.

2.18.27 unlock_translation

weblate unlock_translation <project|project/component>

Unlocks a given component, making it available for translation.

İpucu: Useful in case you want to do some maintenance on the underlying repository.

You can either define which project or component to update (for example `weblate/application`), or use `--all` to update all existing components.

Ayrıca bakınız:

lock_translation

2.18.28 setupgroups

weblate setupgroups

Configures default groups and optionally assigns all users to that default group.

--no-privs-update

Turns off automatic updating of existing groups (only adds new ones).

--no-projects-update

Prevents automatic updates of groups for existing projects. This allows adding newly added groups to existing projects, see *Proje erişim denetimi*.

Ayrıca bakınız:

Yetkiler ve yerleşik rollerin listesi

2.18.29 setuplang

weblate setuplang

Updates list of defined languages in Weblate.

--no-update

Turns off automatic updates of existing languages (only adds new ones).

2.18.30 updatechecks

weblate updatechecks <project|project/component>

Updates all checks for all strings.

İpucu: Useful for upgrades which do major changes to checks.

You can either define which project or component to update (for example `weblate/application`), or use `--all` to update all existing components.

2.18.31 updategit

weblate updategit <project|project/component>

Fetches remote VCS repositories and updates the internal cache.

You can either define which project or component to update (for example `weblate/application`), or use `--all` to update all existing components.

Not: Usually it is better to configure hooks in the repository to trigger *Bildirim kancaları*, instead of regular polling by `updategit`.

2.19 Duyurular

4.0 sürümünde değişti: In prior releases this feature was called whiteboard messages.

Provide info to your translators by posting announcements, site-wide, per project, component, or language.

Announce the purpose, deadlines, status, or specify targets for translation.

The users will receive notification on the announcements for watched projects (unless they opt out).

This can be useful for various things from announcing the purpose of the website to specifying targets for translations.

The announcements can be posted on each level in the *Manage* menu, using *Post announcement*:

Webplate

Dashboard

Projects ▾

Languages ▾

Checks ▾

WebplateOrg

translated 90%

Translations will be used only if they reach 60%.

Components

Languages

Info

Search

Insights ▾

Files ▾

Tools ▾

Manage ▾

Share ▾

Not watching ▾

Post announcement

Message

You can use Markdown and mention users by @username.

Category

Info (light blue) ▾

Category defines color used for the message.

Expiry date

mm/dd/yyyy

The message will be not shown after this date. Use it to announce string freeze and translation deadline for next release.

☒ Notify users

The message is shown for all translations within the project, until its given expiry, or permanently until it is deleted.

Add

Powered by Weblate 4.13 [About Weblate](#) [Legal](#) [Contact](#) [Documentation](#) [Donate to Weblate](#)

It can be also added using the admin interface:

Weblate administration

WELCOME, **WEBLATE TEST**. [RETURN TO WEBLATE](#) / [DOCUMENTATION](#) / [CHANGE PASSWORD](#) / [SIGN OUT](#)

[Home](#) · [Weblate translations](#) · [Announcements](#) · [Add Announcement](#)




Add Announcement




Required fields are marked in bold.




Message:


Translations will be used only if they reach 60%.


You can use Markdown and mention users by @username.

Project: WeblateOrg   

Component: -----   

Language: -----   

Category: Info (light blue) 
Category defines color used for the message.

Expiry date: [Today](#) 
The message will be not shown after this date. Use it to announce string freeze and translation deadline for next release.

☒ Notify users

Save and add another Save and continue editing SAVE

The announcements are then shown based on their specified context:

No context specified

Shown on dashboard (landing page).

Project specified

Shown within the project, including all its components and translations.

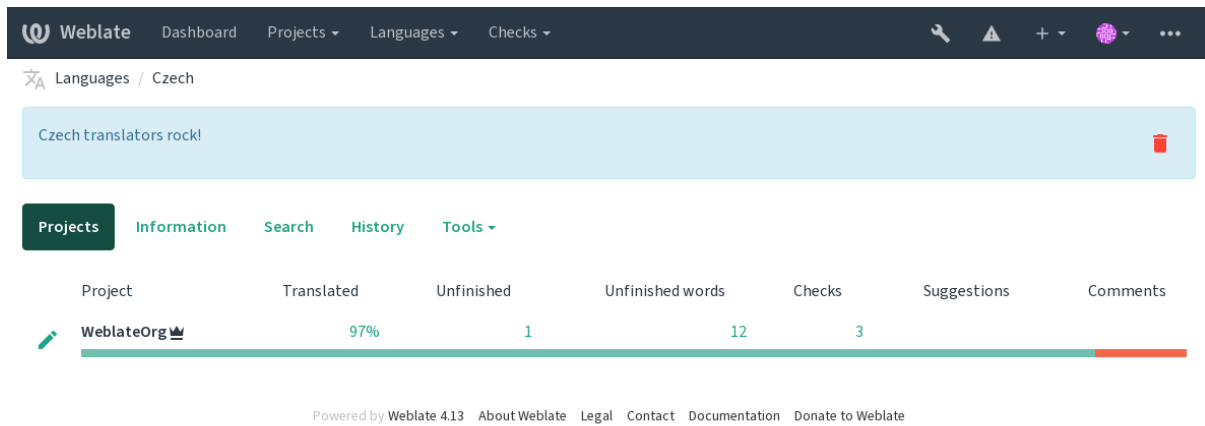
Component specified

Shown for a given component and all its translations.

Language specified

Shown on the language overview and all translations in that language.

This is how it looks on the language overview page:



2.20 Component Lists

Specify multiple lists of components to appear as options on the user dashboard, from which users can pick one as their default view. See *Panel* to learn more.

2.20 sürümünde değişti: A status will be presented for each component list presented on the dashboard.

The names and content of component lists can be specified in the admin interface, in *Component lists* section. Each component list must have a name that is displayed to the user, and a slug representing it in the URL.

2.13 sürümünde değişti: Change dashboard settings for anonymous users from the admin interface, altering what dashboard is presented to unauthenticated users.

2.20.1 Automatic component lists

2.13 sürümünde geldi.

Add components to the list automatically based on their slug by creating *Automatic component list assignment* rules.

- Useful for maintaining component lists for large installations, or in case you want to have one component list with all components on your Weblate installation.

İpucu: Make a component list containing all the components of your Weblate installation.

1. Define *Automatic component list assignment* with `^.*$` as regular expression in both the project and the component fields, as shown on this image:

Weblate administration
WELCOME, **WEBLATE TEST**. [RETURN TO WEBLATE](#) / [DOCUMENTATION](#) / [CHANGE PASSWORD](#) / [SIGN OUT](#)

Home · Weblate translations · Component lists · Add Component list

Add Component list

Required fields are marked in bold.

Component list name: Display name

URL slug: Name used in URLs and filenames.

☒ Show on dashboard
When enabled this component list will be shown as a tab on the dashboard

Components:

Available components ⓘ

- WeblateOrg/Django
- WeblateOrg/Language names
- WeblateOrg/WeblateOrg

Choose all ⓘ

Chosen components ⓘ

Remove all ⓘ

Hold down "Control", or "Command" on a Mac, to select more than one.

AUTOMATIC COMPONENT LIST ASSIGNMENTS

PROJECT REGULAR EXPRESSION ⓘ	COMPONENT REGULAR EXPRESSION ⓘ	DELETE? ⓘ
<input type="text" value="^.*\$"/>	<input type="text" value="^.*\$"/>	<input type="button" value="✕"/>

+ Add another Automatic component list assignment

Save and add another

Save and continue editing

SAVE

2.21 Optional Weblate modules

Several optional modules are available for your setup.

2.21.1 Git exporter

2.10 sürümünde geldi.

Provides you read-only access to the underlying Git repository using HTTP(S).

Installation

1. Add `weblate.gitexport` to installed apps in `settings.py`:

```
INSTALLED_APPS += ("weblate.gitexport",)
```

2. Export existing repositories by migrating your database after installation:

```
weblate migrate
```

Usage

The module automatically hooks into Weblate and sets the exported repository URL in the *Component configuration*. The repositories are accessible under the `/git/` part of the Weblate URL, for example `https://example.org/git/weblate/main/`.

Repositories for publicly available projects can be cloned without authentication:

```
git clone 'https://example.org/git/weblate/main/'
```

Access to browse the repositories with restricted access (with *Private access control* or when `REQUIRE_LOGIN` is enabled) requires an API token which can be obtained in your *user profile*:

```
git clone 'https://user:KEY@example.org/git/weblate/main/'
```

İpucu: By default members or *Users* group and anonymous user have access to the repositories for public projects via *Access repository* and *Power user* roles.

2.21.2 Faturalandırma

2.4 sürümünde geldi.

This is used on [Hosted Weblate](#) to define billing plans, track invoices and usage limits.

Installation

1. Add `weblate.billing` to installed apps in `settings.py`:

```
INSTALLED_APPS += ("weblate.billing",)
```

2. Run the database migration to optionally install additional database structures for the module:

```
weblate migrate
```

Usage

After installation you can control billing in the admin interface. Users with billing enabled will get new *Billing* tab in their *Kullanıcı profili*.

The billing module additionally allows project admins to create new projects and components without being superusers (see *Adding translation projects and components*). This is possible when following conditions are met:

- The billing is in its configured limits (any overusage results in blocking of project/component creation) and paid (if its price is non zero)
- The user is admin of existing project with billing or user is owner of billing (the latter is necessary when creating new billing for users to be able to import new projects).

Upon project creation user is able to choose which billing should be charged for the project in case he has access to more of them.

2.21.3 Yasal

2.15 sürümünde geldi.

This is used on *Hosted Weblate* to provide required legal documents. It comes provided with blank documents, and you are expected to fill out the following templates in the documents:

legal/documents/tos.html

Terms of service document

legal/documents/privacy.html

Privacy policy document

legal/documents/summary.html

Short overview of the terms of service and privacy policy

Not: Legal documents for the Hosted Weblate service are available in this Git repository <<https://github.com/WeblateOrg/wllegal/tree/main/wllegal/templates/legal/documents>>.

Most likely these will not be directly usable to you, but might come in handy as a starting point if adjusted to meet your needs.

Installation

1. Add `weblate.legal` to installed apps in `settings.py`:

```
INSTALLED_APPS += ("weblate.legal",)

# Optional:

# Social auth pipeline to confirm TOS upon registration/subsequent sign in
SOCIAL_AUTH_PIPELINE += ("weblate.legal.pipeline.tos_confirm",)

# Middleware to enforce TOS confirmation of signed in users
MIDDLEWARE += [
    "weblate.legal.middleware.RequireTOSMiddleware",
]
```

2. Run the database migration to optionally install additional database structures for the module:

```
weblate migrate
```

3. Edit the legal documents in the `weblate/legal/templates/legal/` folder to match your service.

Usage

After installation and editing, the legal documents are shown in the Weblate UI.

2.21.4 Avatars

Avatars are downloaded and cached server-side to reduce information leaks to the sites serving them by default. The built-in support for fetching avatars from e-mails addresses configured for it can be turned off using `ENABLE_AVATARS`.

Weblate currently supports:

- Gravatar
- Libravatar

Ayrıca bakınız:

Avatar ön bellekleme, `AVATAR_URL_PREFIX`, `ENABLE_AVATARS`

2.21.5 Spam protection

You can protect against spamming by users by using the [Akismet](#) service.

1. Install the `akismet` Python module (this is already included in the official Docker image).
2. Obtain the Akismet API key.
3. Store it as `AKISMET_API_KEY` or `WEBLATE_AKISMET_API_KEY` in Docker.

Following content is sent to Akismet for checking:

- Kimliği doğrulanmamış kullanıcılardan öneriler
- Proje ve bileşen açıklamaları ve bağlantılar

Not: This (among other things) relies on IP address of the client, please see *Running behind reverse proxy* for properly configuring that.

Ayrıca bakınız:

Running behind reverse proxy, `AKISMET_API_KEY`, `WEBLATE_AKISMET_API_KEY`

2.21.6 Signing Git commits with GnuPG


3.1 sürümünde geldi.

All commits can be signed by the GnuPG key of the Weblate instance.

1. Turn on `WEBLATE_GPG_IDENTITY`. (Weblate will generate a GnuPG key when needed and will use it to sign all translation commits.)

This feature needs GnuPG 2.1 or newer installed.

You can find the key in the `DATA_DIR` and the public key is shown on the “About” page:


[Weblate](#)
[Dashboard](#)
[Projects](#)
[Languages](#)
[Checks](#)
[Register](#)
[Sign in](#)

[About Weblate](#) / [Weblate keys](#)

[About Weblate](#)
[Statistics](#)
[Keys](#)

SSH key

Weblate uses SSH key to access remote repositories. The corresponding public key is found below, you can use it to grant Weblate access to a repository.

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQDCy3+xmtE5n9jvn2Jlr8dQPnOWIVZ7RBIgaVcLFv/n5SvQ3F8WvQms0Ket1nOpK70ZaZQ6f14VmnCxxv/jjnNwhXzDyqVbxW+I
Weblate
```

Commit signing

All commits made with Weblate are signed with the GPG key CCC7D05B7BEC017B5B2FC2DB1ABC0646F0E9CED5, for which the corresponding public key is found below.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQGNBGGKpR4BDADmx90jAxX1EdzkuRLj6ONwldx+MLDI5Wc/KOr1Q4C0T0Ejk3/
t2+3ulGF2CIRAI3REKGV9z5qrb0NycUOwdKP5Rx+1kya6A7pLY1b610TnnXkSRiv
VW6wVrUPeun+kMBmr50ebWiOTm3lZRSedJNR6ROfaTfWmldtu9jm70+T1BxKJ+l
e2dgh8HnEky7nA0cs4QDHdHv62oawQ4QLh/iQM2s4XkDil1IN2ooOHwL/KUtr
8NsHoJOO40WuTh5kDWgSeQFQqdm0JeaXAtei7E70P2rkCWooo5jaaFkk7UDa7TND
0x/Me5tZGAKP248i7Zhd+9TonjwStqVctW0bhSNlpSDF72FicdX+EYZC8pWDJfLh
lg51kiVclthQQzq9n4B+W85topUh5DO0XQcnHRVnbfnHEY0Ut9teCtgIT003zo8E
1IXv5DiHibpVGjgbVv2vSvn5p9Lft3qq9gBFDIO8sYD6MCJDjoqQ/p782NIOryG
dZtnhxU2O6xlyosAEQEAAbQdV2VibGF0ZSA8d2VibGF0ZUBleGFtcGxlLmNvbT6J
Ac4EEwEKADgWIQTmx9Bbe+wBe1svwtsavAZG8OnO1QUCYqm9HglbAwULCQgHAGYV
CgklCwIEFglDAQleAQIXgAAKCAAvAZG8OnO1U/5C/41TnbrVF0cQjqK0dbGW6jl
DDrav3oJcoAjrjCaCBp0nngy94IXyYLGpzR61Ai3JzeHwlaufcm48bE7Pv/u9h7
-----
```

Powered by Weblate 4.13
 [About Weblate](#)
[Legal](#)
[Contact](#)
[Documentation](#)
[Donate to Weblate](#)

2. Alternatively you can also import existing keys into Weblate, just set `HOME = $DATA_DIR/home` when invoking `gpg`.

Ayrıca bakınız:

`WEBLATE_GPG_IDENTITY`

2.21.7 Rate limiting

3.2 sürümünde değişti: The rate limiting now accepts more fine-grained configuration.

4.6 sürümünde değişti: The rate limiting no longer applies to superusers.

Several operations in Weblate are rate limited. At most `RATELIMIT_ATTEMPTS` attempts are allowed within `RATELIMIT_WINDOW` seconds. The user is then blocked for `RATELIMIT_LOCKOUT`. There are also settings specific to scopes, for example `RATELIMIT_CONTACT_ATTEMPTS` or `RATELIMIT_TRANSLATE_ATTEMPTS`. The table below is a full list of available scopes.

The following operations are subject to rate limiting:

Adı	Kapsam	Allowed at-tempts	Ratelimit win-dow	Lockout peri-od
Kayıt	REGISTRATION	5	300	600
Sending message to admins	MESSAGE	5	300	600
Password authentication on sign in	LOGIN	5	300	600
Sitewide search	SEARCH	6	60	60
Translating	TRANSLATE	30	60	600
Adding to glossary	GLOSSARY	30	60	600
Yeni bir dile çeviri başlatma	LANGUAGE	2	300	600
Yeni proje oluşturma	PROJECT	5	600	600

If a user fails to sign in `AUTH_LOCK_ATTEMPTS` times, password authentication will be turned off on the account until having gone through the process of having its password reset.

The settings can be also applied in the Docker container by adding `WEBLATE_` prefix to the setting name, for example `RATELIMIT_ATTEMPTS` becomes `WEBLATE_RATELIMIT_ATTEMPTS`.

The API has separate rate limiting settings, see [API rate limiting](#).

Ayrıca bakınız:

[Rate limiting](#), [Running behind reverse proxy](#), [API rate limiting](#)

2.21.8 Fedora Mesajlaşma tümleşimi

Fedora Mesajlaşma, Weblate'te gerçekleşen tüm değişiklikler için AMQP tabanlı bir yayıncıdır. Bunu kullanarak Weblate'te meydana gelen değişikliklere ek hizmetler bağlayabilirsiniz.

Fedora Mesajlaşma tümleşimi ayrı bir Python modülü olan `weblate-fedora-messaging` olarak kullanılabilir. Kurulum talimatları için lütfen <https://github.com/WeblateOrg/fedora_messaging/> sayfasına bakın.

2.22 Customizing Weblate

Extend and customize using Django and Python. Contribute your changes upstream so that everybody can benefit. This reduces your maintenance costs; code in Weblate is taken care of when changing internal interfaces or refactoring the code.

Uyarı: Neither internal interfaces nor templates are considered a stable API. Please review your own customizations for every upgrade, the interfaces or their semantics might change without notice.

Ayrıca bakınız:

[Weblate'e Katkıda Bulunma](#)

2.22.1 Creating a Python module

If you are not familiar with Python, you might want to look into [Python For Beginners](#), explaining the basics and pointing to further tutorials.

To write some custom Python code (called a module), a place to store it is needed, either in the system path (usually something like `/usr/lib/python3.7/site-packages/`) or in the Weblate directory, which is also added to the interpreter search path.

Better yet, turn your customization into a proper Python package:

1. Create a folder for your package (we will use `weblate_customization`).
2. Within it, create a `setup.py` file to describe the package:

```
from setuptools import setup

setup(
    name="weblate_customization",
    version="0.0.1",
    author="Your name",
    author_email="yourname@example.com",
    description="Sample Custom check for Weblate.",
    license="GPLv3+",
    keywords="Weblate check example",
    packages=["weblate_customization"],
)
```

3. Create a folder for the Python module (also called `weblate_customization`) for the customization code.
4. Within it, create a `__init__.py` file to make sure Python can import the module.
5. This package can now be installed using `pip install -e`. More info to be found in [Editable installs](#).
6. Once installed, the module can be used in the Weblate configuration (for example `weblate_customization.checks.FooCheck`).

Your module structure should look like this:

```
weblate_customization
├── setup.py
└── weblate_customization
    ├── __init__.py
    ├── addons.py
    └── checks.py
```

You can find an example of customizing Weblate at <https://github.com/WeblateOrg/customize-example>, it covers all the topics described below.

2.22.2 Changing the logo

1. Create a simple Django app containing the static files you want to overwrite (see [Creating a Python module](#)).

Branding appears in the following files:

icons/weblate.svg

Logo shown in the navigation bar.

logo-*.png

Web icons depending on screen resolution and web-browser.

favicon.ico

Web icon used by legacy browsers.

weblate-*.png

Avatars for bots or anonymous users. Some web-browsers use these as shortcut icons.

email-logo.png

Used in notifications e-mails.

2. Add it to `INSTALLED_APPS`:

```
INSTALLED_APPS = (
    # Add your customization as first
    "weblate_customization",
    # Weblate apps are here...
)
```

3. Run `weblate collectstatic --noinput`, to collect static files served to clients.

Ayrıca bakınız:

How to manage static files (e.g. images, JavaScript, CSS), *Serving static files*

2.22.3 Custom quality checks, add-ons and auto-fixes

To install your code for *Custom automatic fixups*, *Writing own checks* or *Eklenti yazma* in Weblate:

1. Place the files into your Python module containing the Weblate customization (see *Creating a Python module*).
2. Add its fully-qualified path to the Python class in the dedicated settings (`WEBLATE_ADDONS`, `CHECK_LIST` or `AUTOFIX_LIST`):

```
# Checks
CHECK_LIST += ("weblate_customization.checks.FooCheck",)

# Autofixes
AUTOFIX_LIST += ("weblate_customization.autofix.FooFixer",)

# Add-ons
WEBLATE_ADDONS += ("weblate_customization.addons.ExamplePreAddon",)
```

Ayrıca bakınız:

Custom automatic fixups, *Writing own checks*, *Eklenti yazma*, *Eklentiden betikleri çalıştırma*

2.23 Yönetim arayüzü

The management interface offer administration settings under the `/manage/` URL. It is available for users signed in with admin privileges, accessible by using the wrench icon top right:

The screenshot shows the Weblate management interface. At the top, there's a navigation bar with 'Weblate' logo, 'Dashboard', 'Projects', 'Languages', and 'Checks'. On the right, there's a wrench icon, a plus sign, a user profile icon, and a menu icon. Below the navigation bar, there's a 'Manage' section with a sub-menu containing 'Weblate status', 'Backups', 'Translation memory', 'Performance report', 'SSH keys', 'Alerts', 'Repositories', 'Users', and 'Appearance'. The 'Weblate status' section is active and shows the following information:

- Weblate support status** (with an info icon)
- Weblate version**: 4.13 — `ff7eb62199eab8bd19df9fcb6d826db46a905cd`
- Support status**: Community support
- Buttons: [Purchase support package](#) and [Donate to Weblate](#)

Below this, there's an 'Activate support package' section (with an info icon) containing:

- A description: 'The support packages include priority e-mail support, or cloud backups of your Weblate installation.'
- Activation token**: A text input field.
- A note: 'Please enter the activation token obtained when making the subscription.'
- Buttons: [Activate](#) and [Purchase support package](#)

At the bottom, there's a footer with the text: 'Powered by Weblate 4.13 About Weblate Legal Contact Documentation Donate to Weblate'.

It includes basic overview of your Weblate:

- Destek durumu, bkz: [Getting support for Weblate](#)
- Yedekler, bkz: [Backing up and moving Weblate](#)
- Paylaşılan çeviri belleği, bkz: [Çeviri Belleği](#)
- Weblate durumunu ve Celery kuyruklarının uzunluğunu gözden geçirmek için performans raporu
- SSH anahtar yönetimi, bkz: [SSH repositories](#)
- Tüm bileşenler için uyarılara genel bakış, bkz: alerts

2.23.1 Django yönetici arayüzü

Uyarı: Kullanımı önerilmediğinden gelecekte kaldırılacaktır; çoğu özellik doğrudan Weblate'den yönetilebilir.

Burada kullanıcılar, çeviriler ve diğer ayarlar gibi veri tabanında depolanan nesneleri yönetebilirsiniz:

Weblate administration

WELCOME **WEBLATE TEST** / [RETURN TO WEBLATE](#) / [DOCUMENTATION](#) / [CHANGE PASSWORD](#) / [SIGN OUT](#)

Site administration

REPORTS

[Weblate support status](#)[Status of repositories](#)[SSH keys](#)[Performance report](#)[Translation memory](#)

ACCOUNTS

[Audit log entries](#) [+ Add](#) [Change](#)[User profiles](#) [+ Add](#) [Change](#)[Verified e-mails](#) [+ Add](#) [Change](#)

AUTH TOKEN

[Tokens](#) [+ Add](#) [Change](#)

AUTHENTICATION

[Groups](#) [+ Add](#) [Change](#)[Roles](#) [+ Add](#) [Change](#)[Users](#) [+ Add](#) [Change](#)

BILLING

[Billing plans](#) [+ Add](#) [Change](#)[Customer billings](#) [+ Add](#) [Change](#)[Invoices](#) [+ Add](#) [Change](#)

CHECKS

[Quality checks](#) [+ Add](#) [Change](#)

FONTS

[Font groups](#) [+ Add](#) [Change](#)[Fonts](#) [+ Add](#) [Change](#)

LEGAL

[TOS agreements](#) [+ Add](#) [Change](#)

PYTHON SOCIAL AUTH

[Associations](#) [+ Add](#) [Change](#)[Nonces](#) [+ Add](#) [Change](#)[User social auths](#) [+ Add](#) [Change](#)

SCREENSHOTS

[Screenshots](#) [+ Add](#) [Change](#)

TRANSLATION MEMORY

[Translation memory entries](#) [+ Add](#) [Change](#)

WEBLATE CONFIGURATION

[Settings](#) [+ Add](#) [Change](#)

WEBLATE LANGUAGES

[Languages](#) [+ Add](#) [Change](#)

WEBLATE TRANSLATIONS

[Announcements](#) [+ Add](#) [Change](#)[Component lists](#) [+ Add](#) [Change](#)[Components](#) [+ Add](#) [Change](#)[Contributor agreements](#) [+ Add](#) [Change](#)[History events](#) [+ Add](#) [Change](#)[Projects](#) [+ Add](#) [Change](#)[String comments](#) [+ Add](#) [Change](#)[String suggestions](#) [+ Add](#) [Change](#)[Strings](#) [+ Add](#) [Change](#)[Translations](#) [+ Add](#) [Change](#)

Recent actions

[My actions](#)

None available

In the *Reports* section, you can check the status of your site, tweak it for *Production setup*, or manage SSH keys used to access *Accessing repositories*.

Manage database objects under any of the sections. The most interesting one is probably *Weblate translations*, where you can manage translatable projects, see *Project configuration* and *Component configuration*.

Weblate languages holds language definitions, explained further in *Dil tanımları*.

Adding a project

Adding a project serves as container for all components. Usually you create one project for one piece of software, or book (See *Project configuration* for info on individual parameters):

Weblate administration

WELCOME, **WEBLATE TEST** / [RETURN TO WEBLATE](#) / [DOCUMENTATION](#) / [CHANGE PASSWORD](#) / [SIGN OUT](#)[Home](#) · [Weblate translations](#) · [Projects](#) · [Add Project](#)

Add Project

Required fields are marked in bold.

Project name:	<input type="text" value="WeblateOrg"/>
	Display name

URL slug:	<input type="text" value="weblateorg"/>
	Name used in URLs and filenames.

Project website:	<input type="text" value="https://weblate.org/"/>
	Main website of translated project.

Translation instructions:	<div><input type="text" value="https://weblate.org/contribute/"/></div>
	You can use Markdown and mention users by @username.

☒ **Set "Language-Team" header**
Lets Weblate update the "Language-Team" file header of your project.

☒ **Use shared translation memory**
Uses the pool of shared translations between projects.

☒ **Contribute to shared translation memory**
Contributes to the pool of shared translations between projects.

Access control:	<div>Protected ▾</div>
	How to restrict access to this project is detailed in the documentation.

☐ **Enable reviews**
Requires dedicated reviewers to approve translations.

☐ **Enable source reviews**
Requires dedicated reviewers to approve source strings.

☒ **Enable hooks**
Whether to allow updating this repository by remote hooks.

Language aliases:	<div></div>
	Comma-separated list of language code mappings, for example: en_GB:en,en_US:en

Machinery settings:	<div>⌀</div>
---------------------	--------------

Ayrıca bakınız:

Project configuration

Bilingual components

Once you have added a project, translation components can be added to it. (See [Component configuration](#) for info regarding individual parameters):

[illegible]

Ayrıca bakınız:

Component configuration, Bilingual and monolingual formats

Monolingual components

For easier translation of these, provide a template file containing the mapping of message IDs to its respective source language (usually English). (See *Component configuration* for info regarding individual parameters):

[illegible]

Ayrıca bakınız:

Component configuration, Bilingual and monolingual formats

2.24 Getting support for Weblate

Weblate is copylefted libre software with community support. Subscribers receive priority support at no extra charge. Prepaid help packages are available for everyone. You can find more info about current support offerings at <https://weblate.org/support/>.

2.24.1 Integrating support

3.8 sürümünde geldi.

Purchased support packages can optionally be integrated into your Weblate [subscription management](#) interface, from where you will find a link to it. Basic instance details about your installation are also reported back to Weblate this way.

Manage

Weblate status Backups Translation memory Performance report SSH keys Alerts Repositories Users Appearance

Tools Billing

Weblate support status ⓘ

Weblate version 4.13 — ff7eb62199eab8bd19df9fcbdd826db46a905cd

Support status Community support

Purchase support package Donate to Weblate

Activate support package ⓘ

The support packages include priority e-mail support, or cloud backups of your Weblate installation.

Activation token

Please enter the activation token obtained when making the subscription.

Activate Purchase support package

Powered by Weblate 4.13 About Weblate Legal Contact Documentation Donate to Weblate

2.24.2 Data submitted to the Weblate

- URL where your Weblate instance is configured
- Your site title
- The Weblate version you are running
- Tallies of some objects in your Weblate database (projects, components, languages, source strings and users)
- The public SSH key of your instance

Additionally, when *Weblate'i Keşfedin* is turned on:

- List of public projects (name, URL and website)

No other data is submitted.

2.24.3 Integration services

- See if your support package is still valid
- *Weblate provisioned backup storage*
- *Weblate’i Keşfedin*

İpucu: Purchased support packages are already activated upon purchase, and can be used without integrating them.

2.24.4 Weblate’i Keşfedin

4.5.2 sürümünde geldi.


Not: Bu özellik şu anda erken beta aşamasındadır.

Weblate’i Keşfet, kullanıcıların Weblate sunucularını ve topluluklarını bulmasını kolaylaştıran bir katılım hizmetidir. Kullanıcılar <<https://weblate.org/discover/>> adresinde kayıtlı hizmetlere göz atabilir ve orada katkıda bulunacak projeleri bulabilir.

Listelenme

İpucu: Weblate’i Keşfet’e katılmak, Weblate’in sunucunuz hakkında bazı bilgiler göndermesini sağlar, lütfen bkz: *Data submitted to the Weblate*.

Weblate’i Keşfet’te sunucunuzu etkin bir destek aboneliği ile listelemek için (bkz: *Integrating support*) yönetim panelinden bunu açmanız yeterlidir:



 Weblate


Dashboard

Projects ▾

Languages ▾

Checks ▾

 + ▾  ▾ ...

 Manage

Weblate status

Backups

Translation memory

Performance report

SSH keys

Alerts

Repositories

Users

Appearance

Tools

Billing

Weblate support status ⓘ

Weblate version4.13 — [ff7eb62199eab8bd19df9fcbd6d826db46a905cd](#)

Support statusCommunity support

Discover WeblateYour Weblate is not listed on weblate.org [Browse discovery](#)

Enable discovery

Manage support package

Purchase support package

Donate to Weblate

Activate support package ⓘ

The support packages include priority e-mail support, or cloud backups of your Weblate installation.

Activation token

Please enter the activation token obtained when making the subscription.

Activate

Purchase support package

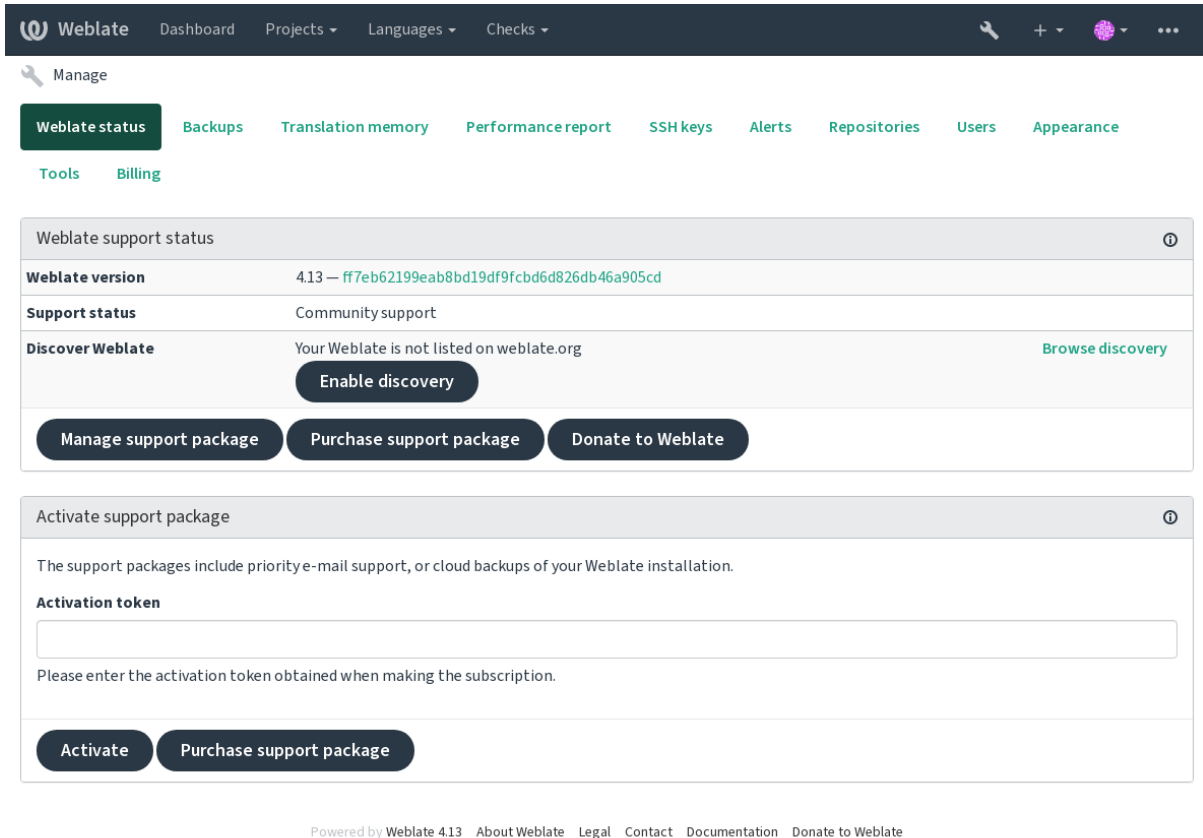
Powered by Weblate 4.13 [About Weblate](#) [Legal](#) [Contact](#) [Documentation](#) [Donate to Weblate](#)

Weblate'i Keşfet'te sunucunuzu destek aboneliği olmadan listeleme:

1. <https://weblate.org/user/> adresinde kaydolun
2. Weblate sunucunuzu <https://weblate.org/subscription/discovery/> adresindeki keşif veri tabanına kaydedin
3. Weblate'inizde hizmet etkinleştirmesini onaylayın ve *Keşfi etkinleştir* düğmesini kullanarak Weblate yönetim sayfanızda keşif listesini açın:

410

Bölüm 2. Administrator docs



Manage

Weblate status Backups Translation memory Performance report SSH keys Alerts Repositories Users Appearance

Tools Billing

Weblate support status ⓘ

Weblate version 4.13 — ff7eb62199eab8bd19df9fcbdd6d826db46a905cd

Support status Community support

Discover Weblate Your Weblate is not listed on weblate.org [Browse discovery](#)

[Enable discovery](#)

[Manage support package](#) [Purchase support package](#) [Donate to Weblate](#)

Activate support package ⓘ

The support packages include priority e-mail support, or cloud backups of your Weblate installation.

Activation token

Please enter the activation token obtained when making the subscription.

[Activate](#) [Purchase support package](#)

Powered by Weblate 4.13 [About Weblate](#) [Legal](#) [Contact](#) [Documentation](#) [Donate to Weblate](#)

Listelemeyi özelleştirme

`<https://weblate.org/user/>` adresinde bir metin ve resim (570 x 260 piksel) sağlayarak listelemeyi özelleştirebilirsiniz.

2.25 Legal documents

Not: Herein you will find various legal information you might need to operate Weblate in certain legal jurisdictions. It is provided as a means of guidance, without any warranty of accuracy or correctness. It is ultimately your responsibility to ensure that your use of Weblate complies with all applicable laws and regulations.

2.25.1 ITAR and other export controls

Weblate can be run within your own datacenter or virtual private cloud. As such, it can be used to store ITAR or other export-controlled information, however, end users are responsible for ensuring such compliance.

The Hosted Weblate service has not been audited for compliance with ITAR or other export controls, and does not currently offer the ability to restrict translations access by country.

2.25.2 US encryption controls

Weblate does not contain any cryptographic code, but might be subject export controls as it uses third party components utilizing cryptography for authentication, data-integrity and -confidentiality.

Most likely Weblate would be classified as ECCN 5D002 or 5D992 and, as publicly available libre software, it should not be subject to EAR (see [Encryption items NOT Subject to the EAR](#)).

Software components used by Weblate (listing only components related to cryptographic function):

Python

See https://wiki.python.org/moin/PythonSoftwareFoundationLicenseFaq#Is_Python_subject_to_export_laws.3F

GnuPG

Optionally used by Weblate

Git

Optionally used by Weblate

curl

Used by Git

OpenSSL

Used by Python and cURL

The strength of encryption keys depends on the configuration of Weblate and the third party components it interacts with, but in any decent setup it will include all export restricted cryptographic functions:

- In excess of 56 bits for a symmetric algorithm
- Factorisation of integers in excess of 512 bits for an asymmetric algorithm
- Computation of discrete logarithms in a multiplicative group of a finite field of size greater than 512 bits for an asymmetric algorithm
- Discrete logarithms in a group different than above in excess of 112 bits for an asymmetric algorithm

Weblate doesn't have any cryptographic activation feature, but it can be configured in a way where no cryptography code would be involved. The cryptographic features include:

- Accessing remote servers using secure protocols (HTTPS)
- Generating signatures for code commits (PGP)

Ayrıca bakınız:

[Export Controls \(EAR\) on Open Source Software](#)

3.1 Weblate'e Katkıda Bulunma

Weblate'i iyileştirmenin düzinelerce yolu vardır. Kodlama, grafik tasarım, belgelendirme, sponsorluk veya fikir olsun, kendinizi rahat hissedeceğiniz birini seçebilirsiniz:

- *Reporting issues in Weblate*
- *Weblate kodlarına katkıda bulunmaya başlama*
- *Contributing to Weblate modules*
- *Weblate'i çevirme*
- *Weblate belgelendirmesine katkıda bulunun*
- *Weblate tartışmaları*
- *Weblate'in geliştirilmesine maddi destek verme*

3.1.1 Weblate'i çevirme

Weblate, Weblate'in kendisi kullanılarak sürekli olarak [çevrilmektedir](#). Weblate'i mümkün olduğu kadar çok dilde kullanılabilir hale getirme çabasında yer almaktan çekinmeyin. Bu, Weblate'i kullanıcılarına yaklaştırmaktadır!

Kaynak dizgide olası bir hata bulursanız, bunu Weblate düzenleyicisinde bir yorumla işaretleyebilirsiniz. Bu şekilde tartışılabilir ve düzeltilebilir. Eminseniz, *Kaynak dizgi konumu* bölümündeki bağlantıya tıklayabilir ve düzeltmenizle birlikte bir çekme isteği gönderebilirsiniz.

3.1.2 Weblate belgelendirmesine katkıda bulunun

İstedığınız belgelendirme sayfasını iyileştirebilirsiniz. Bunu, sayfanın sağ üst köşesindeki *GitHub'da düzenle* düğmesine tıklayarak kolayca yapın.

Lütfen yazarken şu yönergelerle uyun:

1. Geçerliyse, belgelendirmenin bir kısmını kaldırmayın.
2. Açık ve kolay anlaşılır bir dil kullanın. Teknoloji belgeleri yazıyorsunuz, şiir değil. Tüm belgelendirme okuyucularının ana dili İngilizce değildir, düşünceli olun.
3. Don't be afraid to ask if you are not certain. If you have to ask about some feature while editing, don't change its docs before you have the answer. This means: You change or ask. Don't do both at the same time.
4. Belgeleri takip ederken açıklanan eylemleri gerçekleştirerek değişikliklerinizi doğrulayın.
5. Gözden geçirmeyi ve birleştirmeyi daha kolay ve daha hızlı hale getirmek için çekme isteğini küçük parçalar halindeki değişikliklerle gönderin.
6. Büyük bir makaleyi yeniden yazmak ve onun yapısını değiştirmek istiyorsanız, bunu iki adımda yapın:
 1. Yeniden yazın
 2. Yeniden yazma gözden geçirildikten, iyileştirildikten ve birleştirildikten sonra, başka bir çekme isteğinde paragrafların yapısını değiştirin.

İpucu: Belgeleri çevirebilirsiniz.

3.1.3 Yerleşik dil tanımlarını genişletme

Dil tanımları `weblate-language-data` deposunda bulunmaktadır.

Eksik dil tanımlarını `languages.csv` dosyasına ekleyebilirsiniz, diğer dosyalar bu dosyadan oluşturulmaktadır.

3.1.4 Weblate tartışmaları

Bir fikriniz varsa ve bir soruna uygun olup olmadığından emin değilseniz endişelenmeyin. Topluluğa *GitHub tartışmalarında* katılabilirsiniz.

3.1.5 Weblate'in geliştirilmesine maddi destek verme

Weblate'in geliştirilmesini *bağış sayfasında* destekleyebilirsiniz. Burada toplanan fonlar, özgür yazılım projelerine için ücretsiz barındırma sağlanması ve Weblate'in daha da geliştirilmesi için kullanılmaktadır. Fonlama hedefleri ve gururlu bir fon sağlayıcı olarak alacağınız ödüller gibi seçenekler için lütfen *'bağış sayfasına'* bakın.

Supporters who have funded Weblate

List of Weblate supporters:

- Yashiro Ccs
- Cheng-Chia Tseng
- Timon Reinhard
- Cassidy James
- Loic Dachary
- Marozed

- <https://freedombox.org/>
- GNU Solidario (GNU Health)
- BallotReady
- Richard Nespithal
- MyExpenses.Mobi

Do you want to be in the list? Please see options on the [Donate to Weblate](#).

3.2 Weblate kodlarına katkıda bulunmaya başlama

Understand the Weblate source code by going through *Weblate kaynak kodları*, *Weblate frontend* and *Weblate internals*.

3.2.1 Starting with the codebase

Familiarize yourself with the Weblate codebase, by having a go at the bugs labelled [good first issue](#).

3.2.2 Weblate'i yerel olarak çalıştırma

Weblate geliştirmeye başlamak için en rahat yaklaşım *Installing from sources* bölümündeki talimatları izlemektir. Bu, size düzenlenebilir Weblate kaynaklarını içeren bir sanal ortam sağlayacaktır.

1. Clone the Weblate source code:

```
git clone https://github.com/WeblateOrg/weblate.git
cd weblate
```

2. Create a virtualenv:

```
virtualenv .venv
.venv/bin/activate
```

3. Install Weblate (for this you need some system dependencies, see *Installing from sources*):

```
pip install -e .
```

3. Geliştirme için faydalı olacak tüm bağımlılıkları kurun:

```
pip install -r requirements-dev.txt
```

4. Bir geliştirme sunucusu başlatın:

```
weblate runserver
```

5. Depending on your configuration, you might also want to start Celery workers:

```
./weblate/examples/celery start
```

6. To run a test (see *Local testing* for more details):

```
. scripts/test-database
./manage.py test
```

Ayrıca bakınız:

Installing from sources

3.2.3 Weblate'i Docker'da yerel olarak çalıştırma

If you have Docker and docker-compose installed, you can spin up the development environment by simply running:

```
./rundev.sh
```

It will create a development Docker image and start it. Weblate is running on <http://127.0.0.1:8080/> and you can sign in as the user `admin` using `admin` as the password. The new installation is empty, so you might want to continue with [Adding translation projects and components](#).

The `Dockerfile` and `docker-compose.yml` for this are located in the `dev-docker` directory.

The script also accepts some parameters, to execute tests, run it with the `test` parameter and then specify any `test` parameters, for example running only tests in the `weblate.machine` module:

```
./rundev.sh test --failfast weblate.machine
```

Not: Be careful that your Docker containers are up and running before running the tests. You can check that by running the `docker ps` command.

To display the logs:

```
./rundev.sh logs
```

To stop the background containers, run:

```
./rundev.sh stop
```

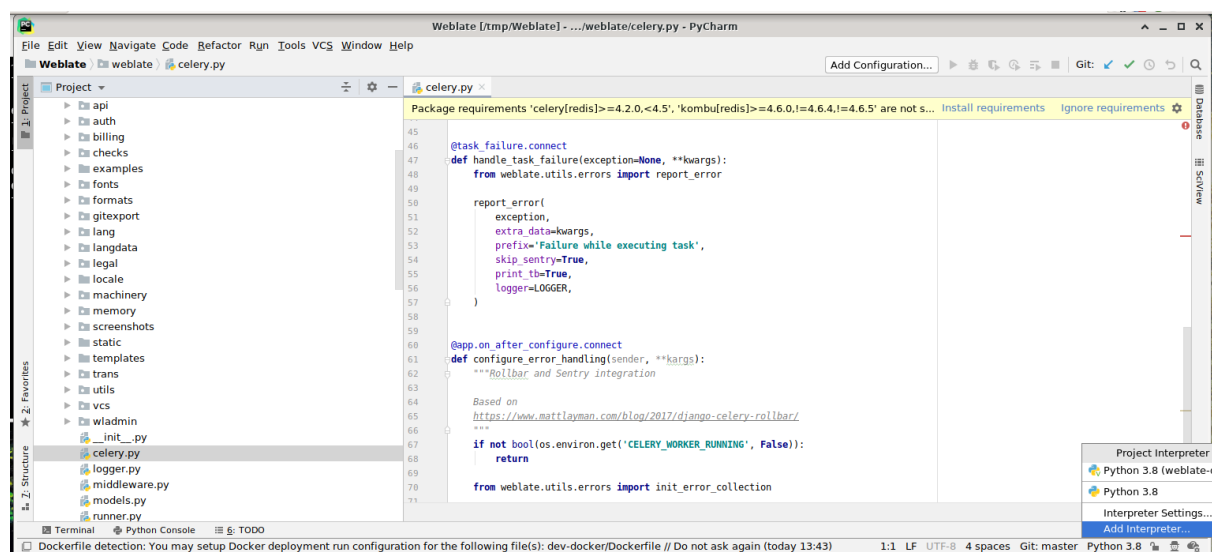
Running the script without arguments will re-create the Docker container and restart it.

Not: This is not a suitable setup for production, as it includes several hacks which are insecure, but they make development easier.

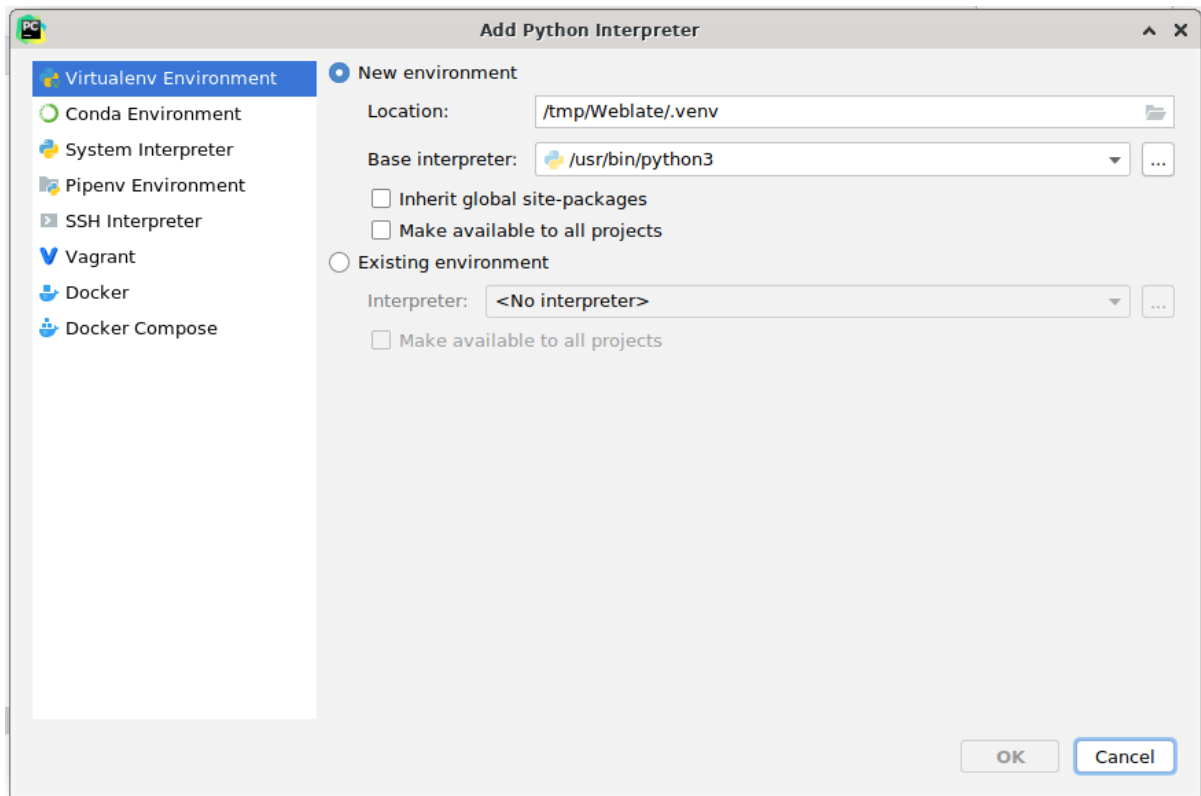
3.2.4 Coding Weblate with PyCharm

PyCharm is a known IDE for Python, here are some guidelines to help you set up your Weblate project in it.

Considering you have just cloned the GitHub repository to a folder, just open it with PyCharm. Once the IDE is open, the first step is to specify the interpreter you want to use:

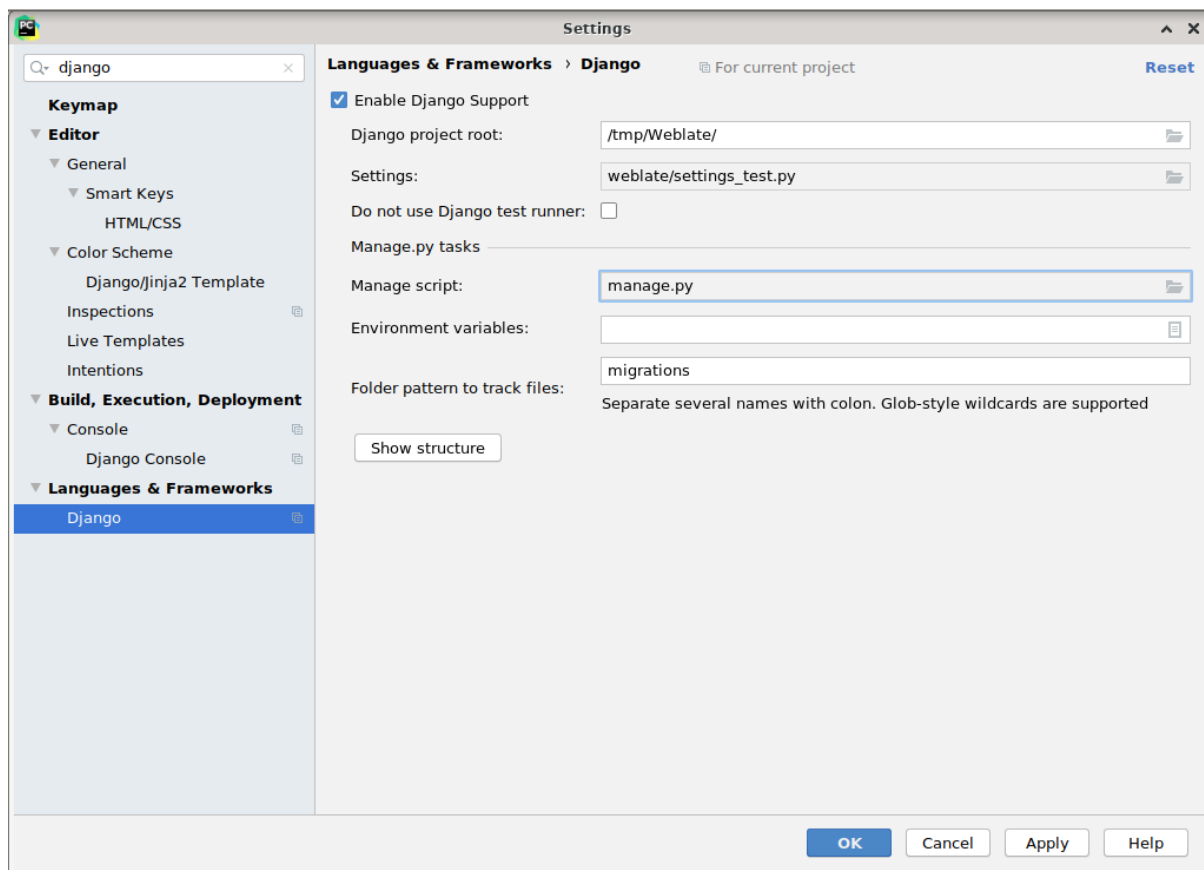


You can either choose to let PyCharm create the virtualenv for you, or select an already existing one:



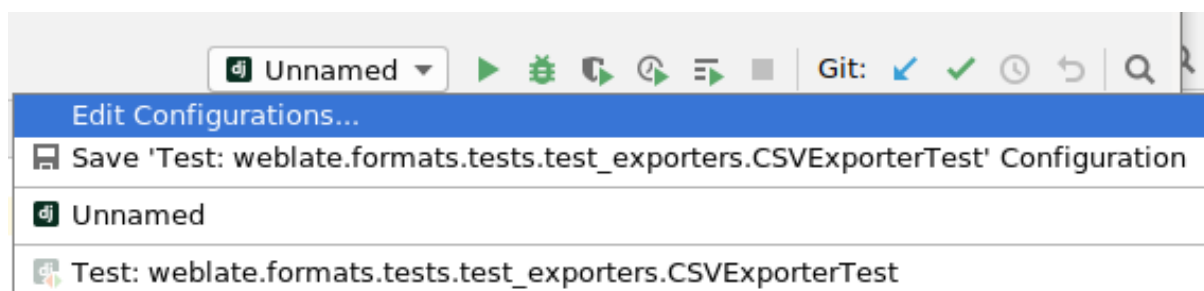
Don't forget to install the dependencies once the interpreter is set: Either through the console (the console from the IDE will directly use your virtualenv by default), or through the interface when you get a warning about missing dependencies.

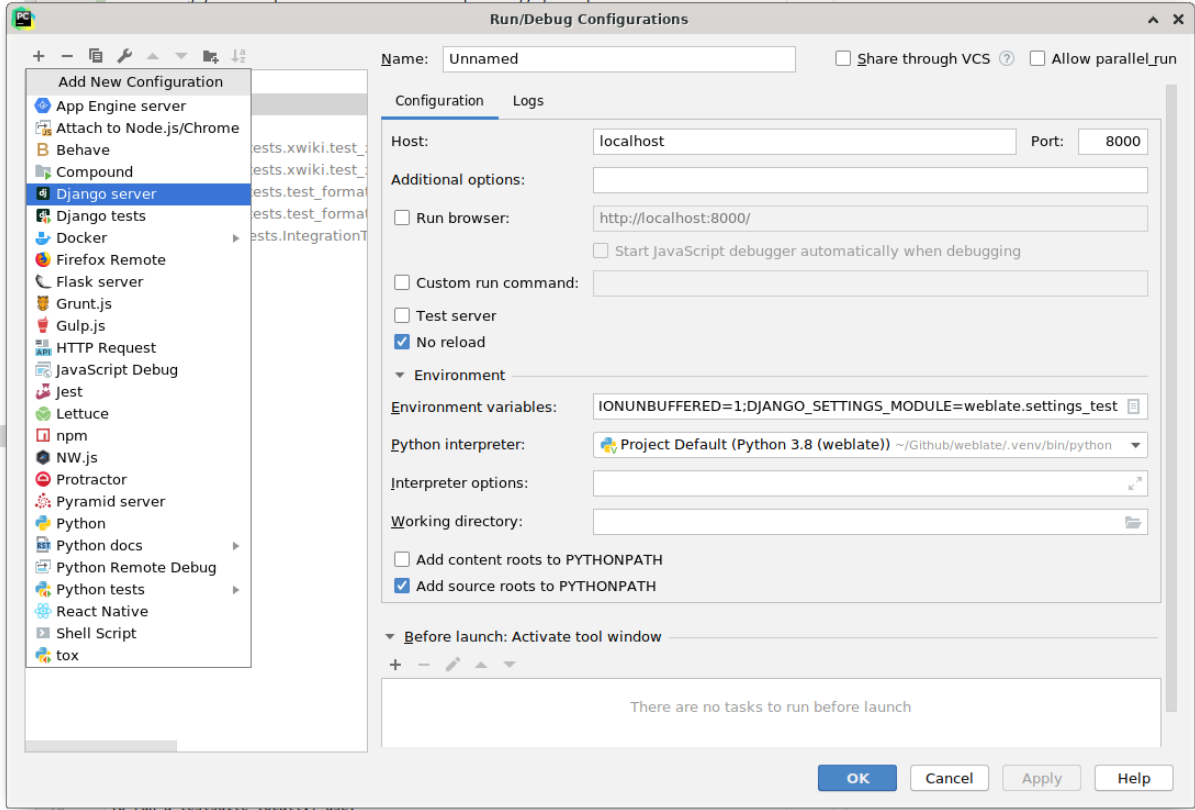
The second step is to set the right info to use Django natively inside PyCharm: The idea is to be able to immediately trigger the unit tests in the IDE. For that you need to specify the root path of the Django project and the path to its settings:



Be careful, the *Django project root* is the actual root of the repository, not the Weblate sub-directory. About the settings, you could use the `weblate/settings_test.py` from the repository, but you could create your own setting and set it there.

The last step is to run the server and to put breakpoints in the code to be able to debug it. This is done by creating a new *Django Server* configuration:





İpucu: Be careful with the property called *No reload*: It prevents the server from being reloaded live if you modify files. This allows the existing debugger breakpoints to persist, when they normally would be discarded upon reloading the server.

3.2.5 Bootstrapping your devel instance

You might want to use `import_demo` to create demo translations and `createadmin` to make an admin user.

3.3 Weblate kaynak kodları

Weblate, [GitHub](#) üzerinde geliştirilmektedir. Kodu çatallayabilir ve çekme isteklerini açabilirsiniz. Başka herhangi bir biçimdeki yamalar da kabul edilir.

Ayrıca bakınız:

Weblate'in içeriden nasıl görüldüğünü görmek için [Weblate internals](#) bölümüne bakın.

3.3.1 Coding guidelines

Any code for Weblate should be written with [Security by Design Principles](#) in mind.

Any code should come with documentation explaining the behavior. Don't forget documenting methods, complex code blocks, or user visible features.

Any new code should utilize [PEP 484](#) type hints. We're not checking this in our CI yet as existing code does not yet include them.

3.3.2 Coding standard and linting the code

The code should follow PEP-8 coding guidelines and should be formatted using **black** code formatter.

To check the code quality, you can use **flake8**, the recommended plugins are listed in `.pre-commit-config.yaml` and its configuration is placed in `setup.cfg`.

The easiest approach to enforce all this is to install [pre-commit](#). The repository contains configuration for it to verify the committed files are sane. After installing it (it is already included in the `requirements-lint.txt`) turn it on by running `pre-commit install` in Weblate checkout. This way all your changes will be automatically checked.

You can also trigger check manually, to check all files run:

```
pre-commit run --all
```

3.4 Debugging Weblate

Bugs can behave as application crashes or as various misbehavior. You are welcome to collect info on any such issue and submit it to the [issue tracker](#).

3.4.1 Hata ayıklama kipi

Turning on debug mode will make the exceptions show in the web browser. This is useful to debug issues in the web interface, but not suitable for a production environment because it has performance consequences and might leak private data.

In a production environment, use [ADMINS](#) to receive e-mails containing error reports, or configure error collection using a third-party service.

Ayrıca bakınız:

[Disable debug mode](#), [Properly configure admins](#), [Collecting error reports](#)

3.4.2 Weblate logs

Weblate can produce detailed logs of what is going on in the background. In the default configuration it uses syslog and that makes the log appear either in `/var/log/messages` or `/var/log/syslog` (depending on your syslog daemon configuration).

The Celery process (see [Background tasks using Celery](#)) usually produces its own logs as well. The example system-wide setups logs to several files under `/var/log/celery/`.

Docker containers log to their output (as per usual in the Docker world), so you can look at the logs using `docker-compose logs`.

Ayrıca bakınız:

[Sample configuration](#) contains `LOGGING` configuration.

3.4.3 Not processing background tasks

A lot of things are done in the background by Celery workers. If things like sending out e-mails or component removal does not work, there might be a related issue.

Bu durumda denetlenecek olanlar:

- Check that the Celery process is running, see [Background tasks using Celery](#)
- Check the Celery queue status, either in [Yönetim arayüzü](#), or using `celery_queues`
- Look in the Celery logs for errors (see [Weblate logs](#))

3.4.4 Not receiving e-mails from Weblate

You can verify whether outgoing e-mail is working correctly by using the `sendtestemail` management command (see [Invoking management commands](#) for instructions on how to invoke it in different environments) or by using [Yönetim arayüzü](#) under the *Tools* tab.

These send e-mails directly, so this verifies that your SMTP configuration is correct (see [Configuring outgoing e-mail](#)). Most of the e-mails from Weblate are however sent in the background and there might be some issues with Celery involved as well, please see [Not processing background tasks](#) for debugging that.

3.4.5 Analyzing application crashes

In case the application crashes, it is useful to collect as much info about the crash as possible. This can be achieved by using third-party services which can collect such info automatically. You can find info on how to set this up in [Collecting error reports](#).

3.4.6 Silent failures

Lots of tasks are offloaded to Celery for background processing. Failures are not shown in the user interface, but appear in the Celery logs. Configuring [Collecting error reports](#) helps you to notice such failures easier.

3.4.7 Performance issues

In case Weblate performs badly in some scenario, please collect the relevant logs showing the issue, and anything that might help figuring out where the code might be improved.

In case some requests take too long without any indication, you might want to install `dogslow` along with [Collecting error reports](#) and get pinpointed and detailed tracebacks in the error collection tool.

In case the slow performance is linked to the database, you can also enable logging of all database queries using following configuration after enabling `DEBUG`:

```
LOGGING["loggers"]["django.db.backends"] = {"handlers": ["console"], "level":
↪ "DEBUG" }
```

3.5 Weblate internals

Not: This chapter will give you basic overview of Weblate internals.

Weblate derives most of its code structure from, and is based on [Django](#).

3.5.1 Directory structure

Quick overview of directory structure of Weblate main repository:

docs

Source code for this documentation, which can be built using [Sphinx](#).

dev-docker

Docker code to run development server, see [Weblate'i Docker'da yerel olarak çalıştırma](#).

weblate

Source code of Weblate as a [Django](#) application, see [Weblate internals](#).

weblate/static

Client files (CSS, Javascript and images), see [Weblate frontend](#).

3.5.2 Modules

Weblate consists of several Django applications (some optional, see [Optional Weblate modules](#)):

accounts

User account, profiles and notifications.

addons

Add-ons to tweak Weblate behavior, see [Eklentiler](#).

api

API based on [Django REST framework](#).

auth

Authentication and permissions.

billing

The optional [Faturalandırma](#) module.

checks

Translation string [Quality checks](#) module.

fonts

Font rendering checks module.

formats

File format abstraction layer based on [translate-toolkit](#).

gitexport

The optional [Git exporter](#) module.

lang

Module defining language and plural models.

legal

	The optional <i>Yasal</i> module.
<code>machinery</code>	Integration of machine translation services.
<code>memory</code>	Dahili çeviri belleği, bkz <i>Çeviri Belleği</i> .
<code>screenshots</code>	Screenshots management and OCR module.
<code>trans</code>	Main module handling translations.
<code>utils</code>	Various helper utilities.
<code>vcs</code>	Version control system abstraction.
<code>wladmin</code>	Django admin interface customization.

3.6 Developing add-ons

Eklentiler Weblate’de yerelleştirme iş akışını özelleştirmenin bir yoludur.

```
class weblate.addons.base.BaseAddon (storage=None)
    Weblate eklentileri için temel sınıf.

    classmethod can_install (component, user)
        Eklentinin verilen bileşenle uyumlu olup olmadığını denetle.

    configure (settings)
        Yapılandırmayı kaydet.

    daily (component)
        Günlük tetiklenen kanca.

    classmethod get_add_form (user, component, **kwargs)
        Yeni eklenti eklemek için yapılandırma formunu döndür.

    get_settings_form (user, **kwargs)
        Bu eklenti için yapılandırma formunu döndür.

    post_add (translation)
        Yeni çeviri eklendikten sonra tetiklenen kanca.

    post_commit (component)
        Değişiklikler depoya işlendikten sonra tetiklenen kanca.

    post_push (component)
        Depo yukarı akışa yollandıktan sonra tetiklenen kanca.

    post_update (component, previous_head: str, skip_push: bool)
        Depo yukarı akıştan güncellendikten sonra tetiklenen kanca.
```

Parametreler

- **previous_head** (*str*) – Güncellemeden önce deponun HEAD'i, ilk klonlamada boş olabilir.
- **skip_push** (*bool*) – Eklenti işleminin değişiklikleri yukarı akışa yollamayı atlayıp atlamayacağı. Genellikle bunu temeldeki yöntemlere `commit_and_push` veya `commit_pending` olarak iletebilirsiniz.

pre_commit (*translation, author*)

Değişiklikler depoya işlenmeden önce tetiklenen kanca.

pre_push (*component*)

Depo yukarı akışa yollanmadan önce tetiklenen kanca.

pre_update (*component*)

Depo yukarı akıştan güncellenmeden önce tetiklenen kanca.

save_state ()

Eklenti durum bilgilerini kaydet.

store_post_load (*translation, store*)

Bir dosya ayrıştırıldıktan sonra tetiklenen kanca.

Argüman olarak bir dosya biçimi sınıfının bir örneğini alır.

Bu, örneğin dosyanın nasıl kaydedileceğini ayarlamak gibi dosya biçimi sınıfı parametrelerini değiştirmek için faydalıdır.

unit_pre_create (*unit*)

Yeni birim oluşturulmadan önce tetiklenen kanca.

Here is an example add-on:

```
#
# Copyright © 2012–2022 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program. If not, see <https://www.gnu.org/licenses/>.
#

from django.utils.translation import gettext_lazy as _

from weblate.addons.base import BaseAddon
from weblate.addons.events import EVENT_PRE_COMMIT

class ExampleAddon(BaseAddon):
    # Filter for compatible components, every key is
    # matched against property of component
    compat = {"file_format": {"po", "po-mono"}}
    # List of events add-on should receive
    events = (EVENT_PRE_COMMIT,)
```

(sonraki sayfaya devam)

(önceki sayfadan devam)

```
# Add-on unique identifier
name = "weblate.example.example"
# Verbose name shown in the user interface
verbose = _("Example add-on")
# Detailed add-on description
description = _("This add-on does nothing it is just an example.")

# Callback to implement custom behavior
def pre_commit(self, translation, author):
    return
```

3.7 Weblate frontend

The frontend is currently built using Bootstrap, jQuery and few third party libraries.

3.7.1 Desteklenen tarayıcılar

Weblate supports the latest, stable releases of all major browsers and platforms.

Alternative browsers which use the latest version of WebKit, Blink, or Gecko, whether directly or via the platform's web view API, are not explicitly supported. However, Weblate should (in most cases) display and function correctly in these browsers as well.

Older browsers might work, but some features might be limited.

3.7.2 Dependency management

The yarn package manager is used to update third party libraries. The configuration lives in `scripts/yarn` and there is a wrapper script `scripts/yarn-update` to upgrade the libraries, build them and copy to correct locations in `weblate/static/vendor`, where all third partly frontend code is located. The Weblate specific code should be placed directly in `weblate/static` or feature specific subdirectories (for example `weblate/static/editor`).

Adding new third-party library typically consists of:

```
# Add a yarn package
yarn --cwd scripts/yarn add PACKAGE
# Edit the script to copy package to the static folder
edit scripts/yarn-update
# Run the update script
./scripts/yarn-update
# Add files to git
git add .
```

3.7.3 Coding style

Weblate relies on [Prettier](#) for the code formatting for both JavaScript and CSS files.

We also use [ESLint](#) to check the JavaScript code.

3.7.4 Yerelleştirme

Should you need any user visible text in the frontend code, it should be localizable. In most cases all you need is to wrap your text inside `gettext` function, but there are more complex features available:

```
document.write(gettext('this is to be translated'));

var object_count = 1 // or 0, or 2, or 3, ...
s = gettext('literal for the singular case',
            'literal for the plural case', object_count);

fmts = gettext('There is %s object. Remaining: %s',
               'There are %s objects. Remaining: %s', 11);
s = interpolate(fmts, [11, 20]);
// s is 'There are 11 objects. Remaining: 20'
```

Ayrıca bakınız:

[Translation topic in the Django documentation](#)

3.7.5 Icons

Weblate currently uses material design icons. In case you are looking for new symbol, check [Material Design Icons](#) or [Material Design Resources](#).

Additionally, there is `scripts/optimize-svg` to reduce size of the SVG as most of the icons are embedded inside the HTML to allow styling of the paths.

3.8 Reporting issues in Weblate

Weblate [hata takibi](#) GitHub üzerinde barındırılıyor.

Feel welcome to report any issues you have, or suggest improvement for Weblate there. There are various templates prepared to comfortably guide you through the issue report.

If what you have found is a security issue in Weblate, please consult the [Güvenlik sorunları](#) section below.

If you are not sure about your bug report or feature request, you can try [Weblate tartışmaları](#).

3.8.1 Güvenlik sorunları

In order to give the community time to respond and upgrade, you are strongly urged to report all security issues privately. HackerOne is used to handle security issues, and can be reported directly at [HackerOne](#). Once you submit it there, community has limited but enough time to solve the incident.

Alternatively, report to security@weblate.org, which ends up on HackerOne as well.

If you don't want to use HackerOne, for whatever reason, you can send the report by e-mail to michal@cihar.com. You can choose to encrypt it using this PGP key `3CB 1DF1 EF12 CF2A C0EE 5A32 9C27 B313 42B7 511D`. You can also get the PGP key from [Keybase](#).

Not: Weblate depends on third-party components for many things. In case you find a vulnerability affecting one of those components in general, please report it directly to the respective project.

Some of these are:

- [Django](#)
- [Django REST framework](#)

- [Python Social Auth](#)

3.9 Weblate testsuite and continuous integration

Testsuites exist for most of the current code, increase coverage by adding testcases for any new functionality, and verify that it works.

3.9.1 Continuous integration

Current test results can be found on [GitHub Actions](#) and coverage is reported on [Codecov](#).

There are several jobs to verify different aspects:

- Unit tests
- Documentation build and external links
- Migration testing from all supported releases
- Code linting
- Setup verification (ensures that generated dist files do not miss anything and can be tested)

The configuration for the CI is in `.github/workflows` directory. It heavily uses helper scripts stored in `ci` directory. The scripts can be also executed manually, but they require several environment variables, mostly defining Django settings file to use and database connection. The example definition of that is in `scripts/test-database`:

```
# Simple way to configure test database from environment

# Database backend to use postgresql / mysql / mariadb
export CI_DATABASE=${1:-postgresql}

# Database server configuration
export CI_DB_USER=weblate
export CI_DB_PASSWORD=weblate
export CI_DB_HOST=127.0.0.1

# Django settings module to use
export DJANGO_SETTINGS_MODULE=weblate.settings_test
```

The simple execution can look like:

```
. scripts/test-database
./ci/run-migrate
./ci/run-test
./ci/run-docs
```

3.9.2 Local testing

To run a testsuite locally, use:

```
DJANGO_SETTINGS_MODULE=weblate.settings_test ./manage.py test
```

İpucu: You will need a database (PostgreSQL) server to be used for tests. By default Django creates separate database to run tests with `test_` prefix, so in case your settings is configured to use `weblate`, the tests will use `test_weblate` database. See [Database setup for Weblate](#) for setup instructions.

The `weblate/settings_test.py` is used in CI environment as well (see [Continuous integration](#)) and can be tuned using environment variables:

```
# Simple way to configure test database from environment

# Database backend to use postgresql / mysql / mariadb
export CI_DATABASE=${1:-postgresql}

# Database server configuration
export CI_DB_USER=weblate
export CI_DB_PASSWORD=weblate
export CI_DB_HOST=127.0.0.1

# Django settings module to use
export DJANGO_SETTINGS_MODULE=weblate.settings_test
```

Prior to running tests you should collect static files as some tests rely on them being present:

```
DJANGO_SETTINGS_MODULE=weblate.settings_test ./manage.py collectstatic
```

You can also specify individual tests to run:

```
DJANGO_SETTINGS_MODULE=weblate.settings_test ./manage.py test weblate.gitexport
```

İpucu: The tests can also be executed inside developer docker container, see [Weblate'i Docker'da yerel olarak çalıştırma](#).

Ayrıca bakınız:

See [Testing in Django](#) for more info on running and writing tests for Django.

3.10 Data schemas

Weblate uses [JSON Schema](#) to define layout of external JSON files.

3.10.1 Weblate Çeviri Belleği Şeması

https://weblate.org/schemas/weblate-memory.schema.json			
tür	array		
items	Çeviri Belleği Ögesi		
	tür	nesne	
	özellikler		
	• category	Dizgi Kategorisi	
		1 is global, 2 is shared, 10000000+ are project specific, 20000000+ are user specific	
		tür	integer
		örnekler	1
		minimum	0
		öntanımlı	1
	• origin	The String Origin	
		Dosya adı veya bileşen adı	
		tür	dizgi
örnekler		test.tmx	
		project/component	
öntanımlı			

[sonraki sayfaya devam](#)

Tablo 1 – önceki sayfadan devam

tanımlar	• source	<i>Kaynak Dizgi</i>	
		tür	<i>dizgi</i>
		örnekler	Merhaba
		minLength	1
		öntanımlı	
	• source_language	<i>Kaynak Dili</i>	
		ISO 639-1 / ISO 639-2 / IETF BCP 47	
		tür	<i>dizgi</i>
		örnekler	en
		pattern	^[^]+\$
		öntanımlı	
	• target	<i>Hedef Dizgi</i>	
		tür	<i>dizgi</i>
		örnekler	Ahoj
		minLength	1
		öntanımlı	
	• target_language	<i>Hedef Dili</i>	
		ISO 639-1 / ISO 639-2 / IETF BCP 47	
		tür	<i>dizgi</i>
		örnekler	cs
		pattern	^[^]+\$
		öntanımlı	
	additionalProperties		False

Ayrıca bakınız:

Çeviri Belleği, dump_memory, import_memory

3.10.2 Weblate kullanıcı verilerini dışa aktarma

https://weblate.org/schemas/weblate-userdata.schema.json		
tür	<i>nesne</i>	
özellikler		
• basic	<i>Temel</i>	
	tür	<i>nesne</i>
	özellikler	
	• username	<i>Kullanıcı adı</i>
		tür <i>dizgi</i>
		örnekler yönetici
		öntanımlı
	• full_name	<i>Ad soyad</i>
		tür <i>dizgi</i>
		örnekler Weblate Yöneticisi
		öntanımlı
	• email	<i>E-posta</i>
		tür <i>dizgi</i>
		örnekler noreply@example.com
		öntanımlı
	• date_joined	<i>Katıldığı tarih</i>
		tür <i>dizgi</i>
		örnekler 2019-11-18T18:53:54.862Z
		öntanımlı
• profile	<i>Profil</i>	
	tür	<i>nesne</i>

sonraki sayfaya devam

Tablo 2 – önceki sayfadan devam

özellikler			
• language	Dil		
	tür	dizgi	
	örnekler	cs	
	pattern	^.*\$	
• suggested	öntanımlı		
	Önerilen dizgi sayısı		
	tür	integer	
	örnekler	1	
• translated	öntanımlı	0	
	Çevrilen dizgi sayısı		
	tür	integer	
	örnekler	24	
• uploaded	öntanımlı	0	
	Yüklenen ekran görüntüsü sayısı		
	tür	integer	
	örnekler	1	
• hi-de_completed	öntanımlı	0	
	Tamamlanan çevirileri panelde gizle		
	tür	boolean	
	örnekler	False	
• secon-dary_in_zen	öntanımlı	True	
	İkincil çevirileri Zen kipinde göster		
	tür	boolean	
	örnekler	True	
• hi-de_source_secondary	öntanımlı	True	
	İkincil çeviri varsa kaynağı gizle		
	tür	boolean	
	örnekler	False	
• editor_link	öntanımlı	True	
	Düzenleyici bağlantısı		
	tür	dizgi	
	örnekler		
• transla-te_mode	pattern	^.*\$	
	öntanımlı		
	Çeviri düzenleyici kipi		
	tür	integer	
• zen_mode	örnekler	0	
	öntanımlı	0	
	Zen düzenleyici kipi		
	tür	integer	
• speci-al_chars	örnekler	0	
	öntanımlı	0	
	Özel karakterler		
	tür	dizgi	
• dashbo-ard_view	örnekler	^.*\$	
	öntanımlı		
	Varsayılan panel görünümü		
	tür	integer	
• dashbo-ard_component_list	örnekler	1	
	öntanımlı	0	
	Varsayılan bileşen listesi		
	tür	null	
• dashbo-ard_component_list	anyOf	tür	null
		tür	integer

sonraki sayfaya devam

Tablo 2 – önceki sayfadan devam

	• languages	Çevrilen diller			
		tür	array		
		öntanımlı			
		items	Dil kodu		
			tür	dizgi	
			örnekler	cs	
			pattern	^.*\$	
			öntanımlı		
		• secondary_languages	İkincil diller		
			tür	array	
	öntanımlı				
	items		Dil kodu		
			tür	dizgi	
			örnekler	sk	
			pattern	^.*\$	
			öntanımlı		
	• watched	İzlenen projeler			
		tür	array		
		öntanımlı			
		items	Proje kısaltması		
			tür	dizgi	
			örnekler	weblate	
			pattern	^.*\$	
	öntanımlı				
• auditlog	Denetim günlüğü				
	tür	array			
	öntanımlı				
	items	Ögeler			
		tür	nesne		
		özellikler			
		• address	IP adresi		
			tür	dizgi	
			örnekler	127.0.0.1	
			pattern	^.*\$	
			öntanımlı		
		• user_agent	Kullanıcı tanıtıcı		
			tür	dizgi	
			örnekler	PC / Linux / Firefox 70.0	
			pattern	^.*\$	
			öntanımlı		
		• timestamp	Zaman damgası		
			tür	dizgi	
			örnekler	2019-11-18T18:58:30.845Z	
			pattern	^.*\$	
			öntanımlı		
		• activity	Etkinlik		
			tür	dizgi	
	örnekler		oturum aç		
pattern	^.*\$				
öntanımlı					
tanımlar					

Ayrıca bakınız:

Kullanıcı profili, *dumpuserdata*

3.11 Releasing Weblate

3.11.1 Releasing schedule

Weblate has two month release cycle for releases (x.y). These are usually followed by a bunch of bugfix releases to fix issues which slip into them (x.y.z).

The change in the major version indicates that the upgrade process can not skip this version - you always have to upgrade to x.0 before upgrading to higher x.y releases.

Ayrıca bakınız:

Upgrading Weblate

3.11.2 Release planning

The features for upcoming releases are collected using GitHub milestones, you can see our roadmap at <https://github.com/WeblateOrg/weblate/milestones>.

3.11.3 Release process

Things to check prior to release:

1. Check newly translated languages by `./scripts/list-translated-languages`.
2. Set final version by `./scripts/prepare-release`.
3. Make sure screenshots are up to date `make -j 12 -C docs update-screenshots`.
4. Merge any possibly pending translations `wlc push; git remote update; git merge origin/weblate`

Perform the release:

5. Create a release `./scripts/create-release --tag` (see below for requirements).

Post release manual steps:

6. Update Docker image.
7. Close GitHub milestone.
8. Once the Docker image is tested, add a tag and push it.
9. Update Helm chart to new version.
10. Include new version in `.github/workflows/migrations.yml` to cover it in migration testing.
11. Increase version in the website download links.
12. Increase version in the repository by `./scripts/set-version`.

To create tags using the `./scripts/create-release` script you will need following:

- GnuPG with private key used to sign the release
- Push access to Weblate git repositories (it pushes tags)
- Configured **hub** tool and access to create releases on the Weblate repo
- SSH access to Weblate download server (the Website downloads are copied there)

3.12 Güvenlik ve gizlilik

Tüyo: Weblate’de güvenlik, kullanıcılarımızın gizliliğine değer veren bir ortam sağlar.

Development of Weblate adheres to the [Best Practices of the Linux Foundation’s Core Infrastructure Initiative](#).

Ayrıca bakınız:

Güvenlik sorunları

3.12.1 Tracking dependencies for vulnerabilities

Security issues in our dependencies are monitored using [Dependabot](#). This covers the Python and JavaScript libraries, and the latest stable release has its dependencies updated to avoid vulnerabilities.

İpucu: There might be vulnerabilities in third-party libraries which do not affect Weblate, so those are not addressed by releasing bugfix versions of Weblate.

3.12.2 Docker container security

The Docker containers are regularly scanned using [Anchore](#) and [Trivy](#) security scanners.

This allows us to detect vulnerabilities early and release improvements quickly.

You can get the results of these scans at GitHub — they are stored as artifacts on our CI in the SARIF format (Static Analysis Results Interchange Format).

Ayrıca bakınız:

Continuous integration

3.13 Contributing to Weblate modules

Besides the main repository, Weblate consists of several Python modules. All these follow same structure and this documentation covers them all.

For example, this covers:

- [wlc](#), Python client library, see *Weblate Client*
- [translation-finder](#), used to discover translatable files in the repository
- [language-data](#), language definitions for Weblate, see *Dil tanımları*

3.13.1 Coding guidelines

Any code for Weblate should be written with [Security by Design Principles](#) in mind.

Any code should come with documentation explaining the behavior. Don’t forget documenting methods, complex code blocks, or user visible features.

Any new code should utilize [PEP 484](#) type hints. We’re not checking this in our CI yet as existing code does not yet include them.

3.13.2 Running tests

The tests are executed using `py.test`. First you need to install test requirements:

```
pip install -r requirements-test.txt
```

You can then execute the testsuite in the repository checkout:

```
py.test
```

Ayrıca bakınız:

The CI integration is very similar to [Weblate testsuite and continuous integration](#).

3.13.3 Coding standard and linting the code

The code should follow PEP-8 coding guidelines and should be formatted using **black** code formatter.

To check the code quality, you can use **flake8**, the recommended plugins are listed in `.pre-commit-config.yaml` and its configuration is placed in `setup.cfg`.

The easiest approach to enforce all this is to install `pre-commit`. The repository contains configuration for it to verify the committed files are sane. After installing it (it is already included in the `requirements-lint.txt`) turn it on by running `pre-commit install` in Weblate checkout. This way all your changes will be automatically checked.

You can also trigger check manually, to check all files run:

```
pre-commit run --all
```

Ayrıca bakınız:

[Weblate kaynak kodları](#)

3.14 Weblate hakkında

3.14.1 Proje hedefleri

Web-based continuous localization tool with tight *Sürüm denetimi tümleşimi* supporting a wide range of *file formats*, making it easy for translators to contribute.

3.14.2 Proje adı

“Weblate”, “web” ve “translate” kelimelerinin birleşimidir.

3.14.3 Proje web sitesi

<https://weblate.org> adresi açılış sayfasıdır ve <https://hosted.weblate.org> adresinde bulutta barındırılan bir hizmet vardır. Belgelendirme <https://docs.weblate.org> adresinde okunabilir.

3.14.4 Proje logoları

Proje logoları ve diğer grafikler <https://github.com/WeblateOrg/graphics> adresinde bulunmaktadır.

3.14.5 Liderlik

Bu proje, michal@cihar.com adresinden ulaşılabilen Michal Čihař tarafından sürdürülmektedir.

3.14.6 Yazarlar

Weblate, Michal Čihař tarafından başlatıldı. 2012 yılında kurulduğundan bu yana binlerce kişi katkıda bulundu.

3.15 Lisans

Copyright © 2012–2022 Michal Čihař michal@cihar.com

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <<https://www.gnu.org/licenses/>>.

4.1 Weblate 4.13

Released on June 15th 2022.

- Dil adlarını güncelleme davranışı değiştirildi.
- Proje listesine sayfalandırma eklendi.
- Yeni birimler oluşturma API'si artık yeni oluşturulan birim hakkında bilgi veriyor.
- Bileşen keşfi artık bir ara dilin yapılandırılmasını destekliyor.
- CSV biçimlerine sabit kodlama çeşitleri eklendi.
- Temeldeki uygulamaya daha iyi uyması için bazı biçimler için bağlam ve konumun işlenmesi değiştirildi.
- ResourceDictionary biçimi için destek eklendi.
- Renk körleri için ilerleme çubuğu renkleri iyileştirildi.
- Dizgi kaldırmada türevlerin temizlenmesi düzeltildi.
- Django 4.1 ile uyumluluk.
- XLIFF içinde kaçırılan XML öğelerini depolamak için destek eklendi.
- Yer tutucu denetim hatalarının biçimlendirilmesi iyileştirildi.
- /.well-known/change-password /accounts/password/ adresine yönlendirildi.
- Makine çevirisi hizmetleri artık her proje için ayrı olarak yapılandırılabilir.
- Added separate permission for resolving comments and grant it to the *Review strings* role.
- Added support for storing alternative translations in the CSV file.
- The placeholders check can now be case-insensitive as well.

[Ayrıntılı tüm değişiklikler.](#)

4.2 Weblate 4.12.2

11 Mayıs 2022 tarihinde yayınlandı.

- Bazı bileşenler için proje çeviri belleğinin yeniden oluşturulması düzeltildi.
- Bileşenleri çevrilmemiş dizgilere göre sıralama düzeltildi.
- Yeni dil eklenirken olası çeviri kaybı düzeltildi.
- Taşıma işlemleri sırasında Weblate SSH anahtarının oluşturulduğundan emin olunması sağlandı.

[Ayrıntılı tüm değişiklikler.](#)

4.3 Weblate 4.12.1

29 Nisan 2022 tarihinde yayınlandı.

- Çekme isteği ileti başlığı düzeltildi.
- Fluent biçiminde söz dizimi hatalarının ele alınması iyileştirildi.
- Bildirim e-postalarında avatar görüntülenmesi düzeltildi.
- İnternette para kazanma desteği eklendi.
- Çevirileri kaldırırken eski kaynak dizgilerinin kaldırılması düzeltildi.

[Ayrıntılı tüm değişiklikler.](#)

4.4 Weblate 4.12

20 Nisan 2022 tarihinde yayınlandı.

- *Nokta işareti uyuşmadı* için Amharca desteği eklendi.
- *Soru işareti uyuşmadı* için Birmanca desteği eklendi.
- *Sözde yerel oluşturma* eklentisinin seçenekleri genişletildi.
- Bir dizgideki tüm nitelik denetimlerini yoksaymak için `ignore-all-checks` bayrağı eklendi.
- *Sözde yerel oluşturma* eklentisinin başarısız denetimleri tetiklemesinden kaçınıldı.
- *Gitea pull requests* için destek eklendi.
- *Dil kodu stili* için Linux stili dil kodu eklendi.
- Proje çeviri belleğini yeniden oluşturma desteği eklendi.
- Bir dosyadan bileşenler oluşturmak API iyileştirildi.
- Diğer çevrilere kopyala ve çoğalt düğmeleri eklendi.
- Birleştirme isteği iletisi bileşen düzeyinde yapılandırılabilir yapıldı.
- XML etiketlerinde en fazla uzunluk kısıtlama davranışı iyileştirildi.
- Fluent dosyalarının ek yorumlarla yüklenmesi düzeltildi.

[Ayrıntılı tüm değişiklikler.](#)

4.5 Weblate 4.11.2

4 Mart 2022 tarihinde yayınlandı.

- İkili sürümdeki bozuk MO dosyaları düzeltildi.

Ayrıntılı tüm değişiklikler.

4.6 Weblate 4.11.1

4 Mart 2022 tarihinde yayınlandı.

- Git ve Mercurial için argümanların eksik temizlenmesi düzeltildi - CVE-2022-23915, daha fazla ayrıntı için [GHSA-3872-f48p-pxqj](#) adresine bakın.
- CSV dosyalarından belirsiz dizgilerin yüklenmesi düzeltildi.
- API kullanarak takım oluşturma desteği eklendi.
- Kullanıcı bahsetme önerilerinin gösterilmesi düzeltildi.
- Proje belirteçlerine erişim artık özelleştirilebilir.

Ayrıntılı tüm değişiklikler.

4.7 Weblate 4.11

25 Şubat 2022 tarihinde yayınlandı.

- XSS - CVE-2022-24710 düzeltmeleri, daha fazla ayrıntı için [GHSA-6jp6-9rf9-gc66](#) adresine bakın.
- API kullanarak eklenti kurulumu düzeltildi.
- *Eylem gerektiren dizgiler*, *Tamamlanmamış dizgiler* olarak yeniden adlandırıldı.
- *ICU MessageFormat sözdizimi* kaynaklı yanlış pozitifler düzeltildi.
- Diğer kullanımlar listesinde kilit ve katılımcı sözleşmesini belirt.
- Eski dizgiler veya eksik çoğullarla PO dosyalarının güncellenmesi düzeltildi.
- Gerrit ile sıkıştırma eklentisi uyumluluğu iyileştirildi.
- Kullanıcı dillerini [Accept-Language](#) başlığına göre otomatik olarak başlat.
- Dizgi kaldırmada hataları ele alma iyileştirildi.
- Weblate now requires Python 3.7 or newer.
- Proje belirteci kimlik doğrulaması ile bazı yazma işlemleri düzeltildi.
- Depodaki dizgiler değiştiğinde dizgi durumu izleme düzeltildi.
- Depodaki dizgi değişikliklerini izle.
- Gezinmeyi iyileştirmek için çeviriler listesinde sabitlenen başlık.
- *Java properties* içindeki çevrilmeyen dizgiler düzeltildi.
- ASCII olmayan dal adlarıyla Git işlemi düzeltildi.
- Yeni eklenti *Çeviriyi kaynakla önceden doldur*.
- *Hızlı ileri almadan birleştir* *Birleştirme stili* eklendi.
- Yeni eklenen dizgilerde *Otomatik çeviri* eklentisinin tetiklenmesi düzeltildi.
- Birmanca için noktalama denetimleri iyileştirildi.

- Kullanıcılara erişim vermek için proje düzeyinde özel takımlar tanımlama desteği eklendi, bkz: [Proje başına erişim denetimini yönetme](#).
- Uyarılara belge bağlantıları eklendi.
- Docker konteyneri, gerektiğinde giden e-posta için TLS/SSL'i otomatik olarak etkinleştirir.
- Çözümlemiş yorumları aramak için destek eklendi.
- borgbackup 1.2 desteği eklendi.
- *Otomatik olarak çevrildi* etiketinin uygulanması düzeltildi.

Ayrıntılı tüm değişiklikler.

4.8 Weblate 4.10.1

22 Aralık 2021 tarihinde yayınlandı.

- Django 4.0'a yükseltmeyle gelen değişiklikler belgelendirildi.
- *Otomatik olarak çevrildi* etiketinin görüntülenmesi düzeltildi.
- Paylaşılan bir depoya sahip bileşenlerde dalın API ile görüntülenmesi düzeltildi.
- Başarısız yollama uyarısında çözümleme iyileştirildi.
- Değişikliklere göz atarken sayfayı elle düzenleme düzeltildi.
- *Kashida harfi kullanıldı* doğruluğu iyileştirildi.
- Weblate Docker konteyneri artık Python 3.10 kullanıyor.

Ayrıntılı tüm değişiklikler.

4.9 Weblate 4.10

16 Aralık 2021 tarihinde yayınlandı.

- DeepL ile resmîlik ve yer tutucuları desteği eklendi.
- Toplu düzenleme ile arama ve değiştirme artık proje ve dil düzeyinde kullanılabilir.
- Arama ve değiştirme için filtreleme eklendi.
- Düzeltildi: “Otomatik çeviri gerçekleştir” ayrıcalığı artık *Diller* grubunun bir parçası değil.
- “Otomatik çeviri gerçekleştir”, *Yönetim* ve yeni *Otomatik çeviri* grubunda.
- Özel karakterlerle XLSX dosyalarının oluşturulması düzeltildi.
- GitHub kimlik doğrulama arka ucuna, kullanıcının belirli bir GitHub kuruluşuna veya ekibine ait olup olmadığını denetleme desteği eklendi.
- API'ye iletilen geçersiz parametrelerle ilgili geri bildirim iyileştirildi.
- API için proje kapsamında erişim belirteçleri için destek eklendi.
- Bazı durumlarda dizgi kaldırma düzeltildi.
- Yeni eklenen dizgilerin çevrilmesi düzeltildi.
- Süzülmesini kolaylaştırmak için otomatik olarak çevrilen dizgileri etiketle.

Ayrıntılı tüm değişiklikler.

4.10 Weblate 4.9.1

19 Kasım 2021 tarihinde yayınlandı.

- Şablon değiştirildikten sonra tek dilli dosyaların yüklenmesi düzeltildi.
- Bayraklardaki boşlukların işlenmesi iyileştirildi.
- İndirme API'sine filtreleme desteği eklendi.
- Yeni çeviriler eklerken istatistiklerin görüntülenmesi düzeltildi.
- GitHub SSH anahtar değişikliği ile ilgili sorunların üstesinden gelindi.

[Ayrıntılı tüm değişiklikler.](#)

4.11 Weblate 4.9

10 Kasım 2021 tarihinde yayınlandı.

- Geçmişte olaylar için daha fazla ayrıntı sağla.
- Geçmişin işlenmesi iyileştirildi.
- Çeviri sayfalarının performansı iyileştirildi.
- Çeviri dosyası indirmelerini kısıtlama desteği eklendi.
- `safe-html` artık `md-text` ile kullanıldığında Markdown'ı anlayabilir.
- `max-length` etiketi artık `xml-text` ile kullanıldığında XML işaretlemesini yok sayar.
- *Çeviri maksimum boyutu* içinde oluşturulan metinlerin boyutları düzeltildi.
- Yaklaşan Google politikası değişikliklerine yardımcı olmak için uygulama mağazası başlığı uzunluğu 30'a düşürüldü.
- `SSH_EXTRA_ARGS` aracılığıyla SSH çağırma özelleştirme desteği eklendi.
- ICU MessageFormat için denetimler eklendi.
- Makine çevirisi arka uçlarında hata durumu ele alınması iyileştirildi.
- Dizgilerdeki olağan dışı boşluk karakterlerini vurgula.
- Düzenleme yaparken çevrilen dizgide kalma seçeneği eklendi.
- `BORG_EXTRA_ARGS` aracılığıyla Borg çağırma özelleştirme desteği eklendi.
- Tek dilli çeviriler için MO dosyalarının oluşturulması düzeltildi.
- Tüm bileşen çevirilerini ZIP dosyası olarak indirmek için API uç noktası eklendi.
- Python 3.10 desteği eklendi.
- Yönetim arayüzünden e-posta davetini yeniden gönderme desteği eklendi.

[Ayrıntılı tüm değişiklikler.](#)

4.12 Weblate 4.8.1

10 Eylül 2021 tarihinde yayınlandı.

- Django yönetici arayüzünde kullanıcı kaldırma düzeltildi.
- Eklenti parametreleri daha ayrıntılı olarak belgelendirildi.
- Sözlükte JavaScript hatası düzeltildi.
- Tutarlılık denetimlerinde eşleşme sayısına sınır ekle.
- Makine çevirilerinde yer tutucuların işlenmesi iyileştirildi.
- API kullanarak eklenti oluşturma düzeltildi.
- Altbilgiye gizlilik ilkesi bağlantısı eklemek için `PRIVACY_URL` ayarı eklendi.
- Üye e-posta adreslerini proje yöneticilerinden gizle.
- Çakışma durumunda gettext PO birleştirmesi iyileştirildi.
- Sözlük vurgulaması iyileştirildi.
- XML denetimleriyle `safe-html` bayrak davranışı iyileştirildi.
- Bağlantılı bileşenler için işleme mesajları düzeltildi.

Ayrıntılı tüm değişiklikler.

4.13 Weblate 4.8

21 Ağustos 2021 tarihinde yayınlandı.

- Apple stringsdict biçimi için destek eklendi.
- Tam arama işlemi artık PostgreSQL ile büyük/küçük harfe duyarlı.
- Bazı durumlarda sözlük açıklamalarının kaydedilmesi düzeltildi.
- Belgelendirme iyileştirmeleri.
- Performans iyileştirmeleri.
- Gerrit ile sıkıştırma eklentisi uyumluluğu iyileştirildi.
- Tek dilli sözlük bileşenlerine dizgi ekleme sorunu düzeltildi.
- Çeşitlerin işlenmesi performansı iyileştirildi.
- Sıkıştırma eklentisinin, bazen yukarı akış değişikliklerini ayrıştırmayı atlaması düzeltildi.
- İndirmeler için dosya uzantısını koru.
- Fluent biçimi için destek eklendi.
- JSON biçimlerini girintilemek için sekmeleri kullanma desteği eklendi.

Ayrıntılı tüm değişiklikler.

4.14 Weblate 4.7.2

15 Temmuz 2021 tarihinde yayınlandı.

- Bir projede daha fazla dil takma adınının yapılandırılmasını destekle.
- API'de arama dizgisi doğrulaması düzeltildi.
- Bir etki alanı değişikliğinden sonra Git dışı aktarıcı URL'leri düzeltildi.
- Windows RC dosyaları için temizleme eklentisi düzeltildi.
- XLIFF güncellemesinde olası çökme düzeltildi.

[Ayrıntılı tüm değişiklikler.](#)

4.15 Weblate 4.7.1

30 Haziran 2021 tarihinde yayınlandı.

- Sözlüğe terim ekleme açılır penceresi iyileştirildi.
- LibreTranslate makine çevirisi hizmeti için destek eklendi.
- Yeni projeler oluştururken hız sınırlaması eklendi.
- Dosya güncellemelerinin performansı iyileştirildi.

[Ayrıntılı tüm değişiklikler.](#)

4.16 Weblate 4.7

17 Haziran 2021 tarihinde yayınlandı.

- Yapılandırma durum denetimi iyileştirildi.
- gettext PO'da kullanılan `object-pascal-format` için destek eklendi, bkz: *Object Pascal biçimi*.
- Amacı daha iyi açıklamak için *Nearby keys*, *Similar keys* olarak yeniden adlandırıldı.
- *mi18n lang dosyaları* için destek eklendi.
- SAML kimlik doğrulama bütünleştirmesi iyileştirildi.
- Uç durumları daha iyi işlemek için *Gerrit* bütünleştirmesi düzeltildi.
- Weblate artık Django 3.2 gerektiriyor.
- E-posta kimlik doğrulaması devre dışı bırakıldığında kullanıcıları davet etme sorunu düzeltildi.
- Dil tanımları iyileştirildi.
- Kullanıcıların bir projeye katkıda bulunmalarını engellemek için destek eklendi.
- Sözlük dillerinin otomatik olarak oluşturulması düzeltildi.
- Eklentiler hakkında belgelendirme genişletildi.
- Bağlantılı depolara sahip bileşenler için performans iyileştirmeleri.
- Ücretsiz DeepL API'si için destek eklendi.
- Kullanıcı yönetimi artık Django yönetici arayüzüne ihtiyaç duymuyor.

[Ayrıntılı tüm değişiklikler.](#)

4.17 Weblate 4.6.2

8 Mayıs 2021 tarihinde yayınlandı.

- Paylaşılan bileşen projeler arasında taşındıktan sonra oluşan çökme düzeltildi.
- Boş özellikler dosyalarına yeni dizgilerin eklenmesi düzeltildi.
- Sağdan sola dillerinde kopya simgesi hizalaması düzeltildi.
- Bilgi sekmesinde dizgi istatistikleri genişletildi.
- Git'te yok sayılan çeviri dosyalarının işlenmesi düzeltildi.
- Ölçüm performansı iyileştirildi.
- Sözlükleri kaydetmedeki olası hata düzeltildi.
- Farklı çoğul kurallara sahip dillerde tutarlılık denetimi davranışı düzeltildi.

[Ayrıntılı tüm değişiklikler.](#)

4.18 Weblate 4.6.1

2 Mayıs 2021 tarihinde yayınlandı.

- Eski istenmeyen ileti koruma kodunu kaldırıldı.
- Kaynak çoğul denetimi doğruluğu iyileştirildi.
- Docker'da kullanıcı arayüzü dillerinin listesi güncellendi.
- Çekme istekleri oluştururken alınan hata mesajları iyileştirildi.
- Pagure'de çekme istekleri oluşturma sorunu düzeltildi.
- Otomatik olarak kurulan eklentilerin tetiklenmesi düzeltildi.
- Yükseltme sırasında olası önbelleğe alma sorunları düzeltildi.
- Yükleme yapılarak tek dilli çevirilere yeni birimler eklenmesi düzeltildi.

[Ayrıntılı tüm değişiklikler.](#)

4.19 Weblate 4.6

19 Nisan 2021 tarihinde yayınlandı.

- `auto_translate` yönetim komutu artık çeviri modunu belirtmek için bir parametreye sahip.
- *Metin dosyaları* için destek eklendi.
- Tüm nesneler için eğilimler ve ölçümler eklendi.
- Added support for directly copying text from secondary languages.
- Değişikliklere göz atarken tarih filtreleme eklendi.
- Etkinlik çizelgeleri iyileştirildi.
- İletişim formu e-postaları için gönderen artık yapılandırılabilir.
- Bileşen oluşturma API'sinde parametre doğrulaması iyileştirildi.
- The rate limiting no longer applies to superusers.
- Otomatik çeviri eklentisi performansı ve güvenilirliği iyileştirildi.

- Docker konteynerinde artık oran sınırlaması özelleştirilebilir.
- Bileşen oluşturma API'si artık otomatik olarak *Weblate internal URLs* kullanıyor.
- Dizgileri listelerken durumun gösterilmesi basitleştirildi.
- Parola karma artık varsayılan olarak Argon2'yi kullanıyor.
- Çeviri durumunu gösteren ilerleme çubukları basitleştirildi.
- Amacı netleştirmek için *Eksik dilleri ekle* yeniden adlandırıldı.
- Dizgi durumunu XLIFF olarak kaydetme düzeltildi.
- Dil genelinde arama eklendi.
- Docker dağıtımında *Yatay ölçeklendirme* için ilk destek.

Ayrıntılı tüm değişiklikler.

4.20 Weblate 4.5.3

1 Nisan 2021 tarihinde yayınlandı.

- Ölçümlerin toplanması düzeltildi.
- Dizgiler eklerken olası çökme düzeltildi.
- Arama sorgusu örnekleri iyileştirildi.
- Değiştirme yüklemesinde yeni eklenen dizgilerin olası kaybı düzeltildi.

4.21 Weblate 4.5.2

26 Mart 2021 tarihinde yayınlandı.

- Otomatik çeviri için yapılandırılabilir zamanlama.
- Lua biçim denetimi eklendi.
- *Art arda yinelenen sözcükler* denetiminde biçim dizgilerini yok say.
- Çeviri sayfasından ekran görüntüsü yüklemeye izin ver.
- Depo bakımına zorunlu dosya eşzamanlaması eklendi.
- Daha uzun kodlu diller için otomatik öneriler düzeltildi.
- Yeni dizgiler ekleme performansı iyileştirildi.
- Nitelik denetimlerinde çeşitli hata düzeltmeleri.
- Çeşitli performans iyileştirmeleri.
- *Weblate'i Keşfedin* ile bütünleştirme eklendi.
- Salt okunur dizgilerle denetim davranışı düzeltildi.

Ayrıntılı tüm değişiklikler.

4.22 Weblate 4.5.1

5 Mart 2021 tarihinde yayınlandı.

- Bazı uç durumlarda sözlük işaretlerinin düzenlenmesi düzeltildi.
- Birkaç sayfanın performansını artırmak için ölçümlerin kullanımı genişletildi.
- TMX dosyalarında doğru kaynak dili kaydet.
- API kullanarak tek dilli PO yüklemelerini daha iyi yönet.
- Sözlük bileşenlerinde uyarı davranışı iyileştirildi.
- Markdown bağlantı denetimleri iyileştirildi.
- İçerik haritalarında sözlük ve kaynak dili belirt.
- Büyük projelerin bileşen listesinin sayfalandırılması.
- Çeviri, bileşen veya proje kaldırma performansı iyileştirildi.
- Toplu düzenleme performansı iyileştirildi.
- ODF dosyaları için “Düzenleme gerekli” ve “Onaylandı” durumlarını koruma düzeltildi.
- Çeviri dosyası indirmelerini özelleştirme arayüzü iyileştirildi

Ayrıntılı tüm değişiklikler.

4.23 Weblate 4.5

19 Şubat 2021 tarihinde yayınlandı.

- gettext PO’da kullanılan `lua-format` için destek eklendi.
- Bir bileşeni projeler arasında paylaşmak için destek eklendi.
- Birden çok biçim işaretiyle birden çok adsız değişken denetimi davranışı düzeltildi.
- Posta listesi alanı projelerden kaldırıldı, bunun yerine çevirmenler için genel talimatların kullanılması önerilir.
- Sözde yerel oluşturma eklentisi eklendi.
- TermBase eXchange dosyaları için destek eklendi.
- Bir işaret kullanarak dizgi çeşitlerini el ile tanımlamak için destek eklendi.
- Tutarlılık denetimlerinin performansı iyileştirildi.
- Uzun dizgiler için çeviri belleği performansı iyileştirildi.
- Açıklamalarda arama desteği eklendi.
- Dizgiler artık iki dilli biçimlerde da eklenebilir ve kaldırılabilir.
- Amazon Translate makine çevirisinde desteklenen dillerin listesini genişlet.
- Java Özellikleri için Java MessageFormat denetimlerini otomatik olarak etkinleştir.
- Bir çeviriye yeni dizgiler eklemek için yeni bir yükleme yöntemi eklendi.
- Çeviriye göz atmak için basit bir arayüz eklendi.
- Sözlükler artık normal bileşenler olarak saklanıyor.
- Artık bileşen API’si kullanıldığı için sözlüklere özel API kaldırıldı.
- Bazı bayrakları değiştirmek için basitleştirilmiş arayüz eklendi.
- Sözlükte çevrilemeyen veya yasak terimler için destek eklendi.

- Bir sözlükte terminoloji tanımlama desteği eklendi.
- Görsel klavye için daha fazla alan elde etmek için metin yönü geçişi taşındı.
- Kullanıcının katkıda bulunduğu projeleri otomatik olarak izleme seçeneği eklendi.
- Çevirinin sözlükle eşleşip eşleşmediği denetimi eklendi.
- Gezinme metni rengini özelleştirme desteği eklendi.

Ayrıntılı tüm değişiklikler.

4.24 Weblate 4.4.2

14 Ocak 2021 tarihinde yayınlandı.

- Dağıtılan bir MO dosyasının bozulması düzeltildi.

4.25 Weblate 4.4.1

13 Ocak 2021 tarihinde yayınlandı.

- Çoğul değişikliklerini geri alma düzeltildi.
- Proje ayarları için yardım görüntüleme düzeltildi.
- Kullanıcıların yönetilmesi iyileştirildi.
- Tek dilli PO dosyalarında bağlamın işlenmesi iyileştirildi.
- HTML, ODF, IDML ve Windows RC biçimleriyle temizleme eklentisi davranışı düzeltildi.
- CSV dosyalarından konumun ayrıştırılması düzeltildi.
- Dosya indirmeleri için içerik sıkıştırması kullan.
- ZIP dosyasından içe aktarmada kullanıcı deneyimi iyileştirildi.
- Yüklemler için dosya biçimlerinin algılanması iyileştirildi.
- Pagure’de yinelenen çekme isteklerinden kaçın.
- Hayalet çevirileri görüntüleme performansı iyileştirildi.
- Yerel tarayıcı metin alanını kullanması için çeviri düzenleyicisi yeniden uygulandı.
- Temizleme eklentisinin yeni dizgiler eklemeyi bozması düzeltildi.
- Eklentiler için API eklendi.

Ayrıntılı tüm değişiklikler.

4.26 Weblate 4.4

15 Aralık 2020 tarihinde yayınlandı.

- Bileşen oluştururken doğrulama iyileştirildi.
- Weblate artık Django 3.1 gerektiriyor.
- Yönetim arayüzünde görünüm özelleştirme desteği eklendi.
- Toplu düzenlemede salt okunur durum işleme düzeltildi.
- CodeMirror bütünleştirilmesi iyileştirildi.

- Çeviri dosyalarından boş dizgileri kaldırmak için eklenti eklendi.
- Çeviriler için artık CodeMirror düzenleyicisi kullanılıyor.
- XML, HTML, Markdown ve reStructuredText için çeviri düzenleyicide söz dizimi vurgulama.
- Çeviri düzenleyicide yerleştirilebilir öğeleri vurgula.
- Standart olmayan dil kodları için destek iyileştirildi.
- Belirsiz dil kodlarını kullanırken uyarı eklendi.
- Kullanıcıya artık yeni bir çeviri eklerken filtrelenmiş bir dil listesi sunuluyor.
- Geçmişte değişiklikler için arama yetenekleri genişletildi.
- Fatura ayrıntı sayfaları ve özgür barındırma iş akışı iyileştirildi.
- Çeviri istatistikleri API'si genişletildi.
- Çeviri yaparken “diğer çeviriler” sekmesi iyileştirildi.
- Görevler API'si eklendi.
- Dosya yüklemelerinin performansı iyileştirildi.
- Kullanıcı tarafından tanımlanan özel karakterlerin gösterilmesi iyileştirildi.
- Otomatik çeviri performansı iyileştirildi.
- Kullanıcı arayüzünde birkaç küçük iyileştirme.
- ZIP indirmelerinin adlandırılması iyileştirildi.
- İzlenmeyen projelerde bildirim alma seçeneği eklendi.

[Ayrıntılı tüm değişiklikler.](#)

4.27 Weblate 4.3.2

4 Kasım 2020 tarihinde yayınlandı.

- Fixed crash on certain component file masks.
- Ard arda yinelenen sözcüklerin doğruluğu iyileştirildi.
- Pagure çekme istekleri için destek eklendi.
- Başarısız kayıtlar için hata mesajları iyileştirildi.
- Geliştirici yorumlarını Markdown olarak görüntüleme geri alındı.
- “master” dalından farklı bir varsayılan dala sahip Git depolarının kurulumu basitleştirildi.
- Yeni oluşturulan iç depolar artık varsayılan dal olarak main kullanıyor.
- reStructuredText çevirirken değiştirilmemiş çevirinin yanlış algılanma oranı azaltıldı.
- Bazı durumlarda CodeMirror görüntüleme sorunları düzeltildi.
- Anlamını netleştirmek için Şablon grubu “Kaynaklar” olarak yeniden adlandırıldı.
- Daha uzun yolları olan depolarda GitLab çekme istekleri düzeltildi.

[Ayrıntılı tüm değişiklikler.](#)

4.28 Weblate 4.3.1

21 Ekim 2020 tarihinde yayınlandı.

- Otomatik çeviri performansı iyileştirildi.
- Kimliği doğrulanmış kullanıcılar için oturum sona ermesi düzeltildi.
- Sürüm bilgilerini gizleme desteği eklendi.
- Bitbucket Sunucusu ile kanca uyumluluğu iyileştirildi.
- Çeviri belleği güncellemelerinin performansı iyileştirildi.
- Bellek kullanımı azaltıldı.
- Matris görünümünün performansı iyileştirildi.
- Bir projeden kullanıcı kaldırmadan önce onay eklendi.

[Ayrıntılı tüm değişiklikler.](#)

4.29 Weblate 4.3

15 Ekim 2020 tarihinde yayınlandı.

- Kullanıcı istatistikleri API'ye dahil edildi.
- Sayfalandırılmış sayfalardaki bileşen sıralaması düzeltildi.
- Bir sözlük için kaynak dili tanımlama.
- GitHub ve GitLab çekme istekleri desteği yeniden yazıldı.
- Öneriyi kaldırdıktan sonraki istatistik sayıları düzeltildi.
- Açık kullanıcı profili genişletildi.
- Zorunlu denetimlerin yapılandırması düzeltildi.
- Yerleşik yedeklemelerle ilgili belgeler iyileştirildi.
- Kaynak dil özniteliği projeden bileşene taşındı.
- Vue I18n biçimlendirmesi denetimi eklendi.
- Genel yer tutucu denetimi artık düzenli ifadeleri destekliyor.
- Matris kipinin görünümü iyileştirildi.
- Makine çevirisi artık otomatik öneriler olarak adlandırılıyor.
- Birden çok GitLab veya GitHub örneğiyle etkileşim için destek eklendi.
- Proje güncellemelerini, birim güncellemeleri ve kaldırmalarını ve sözlükleri kapsayacak şekilde API genişletildi.
- Birim API'si artık çoğul dizgileri düzgün şekilde işliyor.
- Bileşen oluşturma artık ZIP dosyası veya belge yüklemesini işleyebilir.
- API yanıt durum kodları birleştirildi.
- Katılımcı sözleşmesinde Markdown desteği.
- Kaynak dizgi izleme iyileştirildi.
- JSON, YAML ve CSV biçimleri uyumluluğu iyileştirildi.
- Dizgileri kaldırma desteği eklendi.
- Dosya indirmelerinin performansı iyileştirildi.

- Depo yönetimi görünümü iyileştirildi.
- Android için java biçimini otomatik olarak etkinleştirme.
- Yerelleştirilmiş ekran görüntüleri desteği eklendi.
- Python 3.9 desteği eklendi.
- Belirli koşullar altında HTML dosyalarının çevrilmesi düzeltildi.

Ayrıntılı tüm değişiklikler.

4.30 Weblate 4.2.2

2 Eylül 2020 tarihinde yayınlandı.

- JSON biçimleri için kaynak dizgilerinin eşleşmesi düzeltildi.
- Bazı kimlik doğrulama yapılandırmaları için oturum açma yeniden yönlendirmesi düzeltildi.
- Grup eşitleme ile LDAP kimlik doğrulaması düzeltildi.
- Otomatik çeviri ilerlemesini bildirirken çökme düzeltildi.
- Artbilgiler etkinken Git işleme toparlaması düzeltildi.
- API kullanarak yerel VCS bileşenleri oluşturma düzeltildi.

4.31 Weblate 4.2.1

21 Ağustos 2020 tarihinde yayınlandı.

- Android kaynaklarında bazı yerel ayarlar için çoğulların kaydedilmesi düzeltildi.
- Bazı XLIFF dosyaları için temizleme eklentisindeki çökme düzeltildi.
- Docker kalıbında yerelleştirme CDN'si kurmaya izin ver.

4.32 Weblate 4.2

18 Ağustos 2020 tarihinde yayınlandı.

- Kullanıcı sayfaları iyileştirildi ve kullanıcı listesi eklendi.
- 3.x sürümlerinden geçiş desteği bırakıldı, 4.1 veya 4.0'dan geçiş yapın.
- Birkaç tek dilli biçime dışa aktarım eklendi.
- Etkinlik çizelgeleri iyileştirildi.
- Görüntülenen yakındaki dizgilerin sayısı yapılandırılabilir.
- Depo hataları yaşayan bileşenleri kilitleme desteği eklendi.
- Ana gezinme basitleştirildi (düğmeler simgelerle değiştirildi).
- Google Çeviri bütünleştirmesinde dil kodu işlenmesi iyileştirildi.
- Git toparlama eklentisi Co-authored-by : artbilgileri oluşturabilir.
- Arama sorgusu ayrıştırıcısı iyileştirildi.
- Biçim dizgileri denetimlerinden kullanıcı geri bildirimi iyileştirildi.
- Toplu durum değişikliklerinin performansı iyileştirildi.

- Proje veya bileşen yeniden adlandırıldıktan sonra uyumluluk yeniden yönlendirmeleri eklendi.
- Dizgi onayı, bileşen kilitleme ve lisans değişikliği için bildirimler eklendi.
- ModernMT için destek eklendi.
- Dosya yüklemede onaylanan çevirilerin üzerine yazmaktan kaçınmak için izin ver.
- Bazı uyumluluk URL yönlendirmeleri için destek bırakıldı.
- ECMAScript şablon değişmezleri denetimi eklendi.
- Bir bileşeni izleme seçeneği eklendi.
- JSON birim anahtarlarının başındaki nokta kaldırıldı.
- Çeviri belleği için ayrı Celery kuyruğu kaldırıldı.
- Tüm bileşenlerin bir dili bir kerede çevirmesine izin ver.
- Content-Security-Policy HTTP başlıklarını yapılandırmaya izin ver.
- Proje düzeyinde dillere kod adı vermek için destek eklendi.
- HTML veya JavaScript yerelleştirmesine yardımcı olmak için yeni eklenti, bkz: *JavaScript yerelleştirme CDN'i*.
- Weblate etki alanı artık ayarlarda yapılandırılıyor, bkz: *SITE_DOMAIN*.
- Bileşene ve projeye göre arama desteği ekle.

4.33 Weblate 4.1.1

19 Haziran 2020 tarihinde yayınlandı.

- Docker'da otomatik düzeltme veya eklenti yapılandırmasının değişmesi düzeltildi.
- “Hakkında” sayfasındaki olası çökme düzeltildi.
- Bayt koduna derlenmiş yerel ayar dosyalarının kurulumu iyileştirildi.
- Sözlüğe sözcük ekleme sorunu düzeltildi.
- Makineler için klavye kısayolları düzeltildi.
- Bazı kurulumlarda günlük olaylarının atılmasına neden olan hata ayıklama çıktısı kaldırıldı.
- Proje listesindeki kilit göstergesi düzeltildi.
- Bazı kurulumlarda GPG anahtarlarının listelenmesi düzeltildi.
- Hangi DeepL API sürümünün kullanılacağı seçeneği eklendi.
- SAML Hizmet Sağlayıcısı olarak davranmak için destek eklendi, bkz: *SAML authentication*.

4.34 Weblate 4.1

Released on June 15th 2020.

- Added support for creating new translations with included country code.
- Added support for searching source strings with screenshot.
- Extended info available in the stats insights.
- Improved search editing on “Translate” pages.
- Improve handling of concurrent repository updates.
- Include source language in project creation form.

- Include changes count in credits.
- Fixed UI language selection in some cases.
- Allow to whitelist registration methods with registrations closed.
- Improved lookup of related terms in glossary.
- Improved translation memory matches.
- Group same machinery results.
- Add direct link to edit screenshot from translate page.
- Improved removal confirmation dialog.
- Include templates in ZIP download.
- Add support for Markdown and notification configuration in announcements.
- Extended details in check listings.
- Added support for new file formats: *Laravel PHP dizgileri*, *HTML files*, *OpenDocument Format*, *IDML Format*, *Windows RC files*, *INI translations*, *Inno Setup INI çevirileri*, *GWT özellikleri*, *go-i18n JSON files*, *ARB File*.
- Consistently use dismissed as state of dismissed checks.
- Etkinleştirilecek varsayılan eklentileri yapılandırmak için destek eklendi.
- Fixed editor keyboard shortcut to dismiss checks.
- Improved machine translation of strings with placeholders.
- Show ghost translation for user languages to ease starting them.
- Improved language code parsing.
- Show translations in user language first in the list.
- Renamed shapings to more generic name variants.
- Added new quality checks: *Birden çok adsız değişken*, *Uzun süre çevrilmeyen*, *Art arda yinelenen sözcükler*.
- Reintroduced support for wiping translation memory.
- Fixed option to ignore source checks.
- Added support for configuring different branch for pushing changes.
- API now reports rate limiting status in the HTTP headers.
- Added support for Google Translate V3 API (Advanced).
- Added ability to restrict access on component level.
- Added support for whitespace and other special chars in translation flags, see *Customizing behavior using flags*.
- Always show rendered text check if enabled.
- API now supports filtering of changes.
- Added support for sharing glossaries between projects.

4.35 Weblate 4.0.4

7 Mayıs 2020 tarihinde yayınlandı.

- Fixed testsuite execution on some Python 3.8 environments.
- Typo fixes in the documentation.
- Fixed creating components using API in some cases.
- Fixed JavaScript errors breaking mobile navigation.
- Fixed crash on displaying some checks.
- Fixed screenshots listing.
- Fixed monthly digest notifications.
- Fixed intermediate translation behavior with units non existing in translation.

4.36 Weblate 4.0.3

2 Mayıs 2020 tarihinde yayınlandı.

- Fixed possible crash in reports.
- User mentions in comments are now case insensitive.
- Fixed PostgreSQL migration for non superusers.
- Fixed changing the repository URL while creating component.
- Fixed crash when upstream repository is gone.

4.37 Weblate 4.0.2

Released on April 27th 2020.

- Improved performance of translation stats.
- Improved performance of changing labels.
- Toplu düzenleme performansı iyileştirildi.
- Improved translation memory performance.
- Fixed possible crash on component deletion.
- Fixed displaying of translation changes in some corner cases.
- Improved warning about too long celery queue.
- Fixed possible false positives in the consistency check.
- Fixed deadlock when changing linked component repository.
- Included edit distance in changes listing and CSV and reports.
- Avoid false positives of punctuation spacing check for Canadian French.
- Fixed XLIFF export with placeholders.
- Fixed false positive with zero width check.
- Improved reporting of configuration errors.
- Fixed bilingual source upload.

- Automatically detect supported languages for DeepL machine translation.
- Fixed progress bar display in some corner cases.
- Fixed some checks triggering on non translated strings.

4.38 Weblate 4.0.1

Released on April 16th 2020.

- Fixed package installation from PyPI.

4.39 Weblate 4.0

Released on April 16th 2020.

- Weblate now requires Python 3.6 or newer.
- Added management overview of component alerts.
- Added component alert for broken repository browser URLs.
- Improved sign in and registration pages.
- Project access control and workflow configuration integrated to project settings.
- Added check and highlighter for `il8next` interpolation and nesting.
- Added check and highlighter for percent placeholders.
- Önerilerin kusur denetimlerini göster.
- Record source string changes in history.
- Upgraded Microsoft Translator to version 3 API.
- Reimplemented translation memory backend.
- Added support for several `is :` lookups in *Searching*.
- Allow to make *Değiştirilmemiş çeviri* avoid internal blacklist.
- Improved comments extraction from monolingual po files.
- Renamed whiteboard messages to announcements.
- Fixed occasional problems with registration mails.
- Improved LINGUAS update add-on to handle more syntax variants.
- Fixed editing monolingual XLIFF source file.
- Added support for exact matching in *Searching*.
- Extended API to cover screenshots, users, groups, componentlists and extended creating projects.
- Add support for source upload on bilingual translations.
- Added support for intermediate language from developers.
- Added support for source strings review.
- Extended download options for platform wide translation memory.

4.40 Weblate 3.x series

4.40.1 Weblate 3.11.3

Released on March 11th 2020.

- Fixed searching for fields with certain priority.
- Fixed predefined query for recently added strings.
- Fixed searching returning duplicate matches.
- Fixed notifications rendering in Gmail.
- Fixed reverting changes from the history.
- Added links to events in digest notifications.
- Fixed email for account removal confirmation.
- Added support for Slack authentication in Docker container.
- Avoid sending notifications for not subscribed languages.
- Include Celery queues in performance overview.
- Eklentiler için belge bağlantıları düzeltildi.
- Reduced false negatives for unchanged translation check.
- Raised bleach dependency to address CVE-2020-6802.
- Fixed listing project level changes in history.
- Fixed stats invalidation in some corner cases.
- Fixed searching for certain string states.
- Improved format string checks behavior on missing percent.
- Fixed authentication using some third party providers.

4.40.2 Weblate 3.11.2

Released on February 22nd 2020.

- Fixed rendering of suggestions.
- Fixed some strings wrongly reported as having no words.

4.40.3 Weblate 3.11.1

Released on February 20th 2020.

- Documented Celery setup changes.
- Improved filename validation on component creation.
- Fixed minimal versions of some dependencies.
- Fixed adding groups with certain Django versions.
- Fixed manual pushing to upstream repository.
- Improved glossary matching.

4.40.4 Weblate 3.11

Released on February 17th 2020.

- Allow using VCS push URL during component creation via API.
- Rendered width check now shows image with the render.
- Fixed links in notifications e-mails.
- Improved look of plaintext e-mails.
- Display ignored checks and allow to make them active again.
- Display nearby keys on monolingual translations.
- Dizgi şekillerini grupta destegi eklendi.
- Recommend upgrade to new Weblate versions in the system checks.
- Provide more detailed analysis for duplicate language alert.
- Include more detailed license info on the project pages.
- Automatically unshallow local copies if needed.
- Fixed download of strings needing action.
- New alert to warn about using the same file mask twice.
- Improve XML placeables extraction.
- The `SINGLE_PROJECT` can now enforce redirection to chosen project.
- Added option to resolve comments.
- Added bulk editing of flags.
- Added support for labels.
- Toplu düzenleme eklentisi eklendi.
- Added option for *Enforcing checks*.
- Increased default validity of confirmation links.
- Improved Matomo integration.
- Fixed *Çevrilmiş* to correctly handle source string change.
- Extended automatic updates configuration by `AUTO_UPDATE`.
- LINGUAS add-ons now do full sync of translations in Weblate.

4.40.5 Weblate 3.10.3

Released on January 18th 2020.

- Support for translate-toolkit 2.5.0.

4.40.6 Weblate 3.10.2

Released on January 18th 2020.

- Add lock indication to projects.
- Fixed CSS bug causing flickering in some web browsers.
- Fixed searching on systems with non-English locales.
- Improved repository matching for GitHub and Bitbucket hooks.
- Fixed data migration on some Python 2.7 installations.
- Allow configuration of Git shallow cloning.
- Improved background notification processing.
- Fixed broken form submission when navigating back in web browser.
- New add-on to configure YAML formatting.
- Fixed same plurals check to not fire on single plural form languages.
- Fixed regex search on some fields.

4.40.7 Weblate 3.10.1

Released on January 9th 2020.

- Extended API with translation creation.
- Fixed several corner cases in data migrations.
- Compatibility with Django 3.0.
- Veri temizleme performansı iyileştirildi.
- Added support for customizable security.txt.
- Improved breadcrumbs in changelog.
- Improved translations listing on dashboard.
- Improved HTTP responses for webhooks.
- Added support for GitLab merge requests in Docker container.

4.40.8 Weblate 3.10

Released on December 20th 2019.

- Improved application user interface.
- Added doublespace check.
- Fixed creating new languages.
- Avoid sending auditlog notifications to deleted e-mails.
- Salt okunur dizgiler için destek eklendi.
- Added support for Markdown in comments.
- Allow placing translation instruction text in project info.
- Add copy to clipboard for secondary languages.
- Improved support for Mercurial.
- Improved Git repository fetching performance.

- Add search lookup for age of string.
- Show source language for all translations.
- Show context for nearby strings.
- Added support for notifications on repository operations.
- Improved translation listings.
- Extended search capabilities.
- Added support for automatic translation strings marked for editing.
- Avoid sending duplicate notifications for linked component alerts.
- Improve default merge request message.
- Better indicate string state in Zen mode.
- Added support for more languages in Yandex Translate.
- Improved look of notification e-mails.
- Provide choice for translation license.

4.40.9 Weblate 3.9.1

Released on October 28th 2019.

- Remove some unneeded files from backups.
- Fixed potential crash in reports.
- Fixed cross database migration failure.
- Added support for force pushing Git repositories.
- Reduced risk of registration token invalidation.
- Fixed account removal hitting rate limiter.
- Added search based on priority.
- Fixed possible crash on adding strings to JSON file.
- Safe HTML check and fixup now honor source string markup.
- Avoid sending notifications to invited and deleted users.
- Fix SSL connection to redis in Celery in Docker container.

4.40.10 Weblate 3.9

Released on October 15th 2019.

- Include Weblate metadata in downloaded files.
- Improved UI for failing checks.
- Indicate missing strings in format checks.
- Separate check for French punctuation spacing.
- Add support for fixing some of quality checks errors.
- Add separate permission to create new projects.
- Extend stats for char counts.
- Improve support for Java style language codes.

- Added new generic check for placeholders.
- Added support for WebExtension JSON placeholders.
- Added support for flat XML format.
- Extended API with project, component and translation removal and creation.
- Added support for Gitea and Gitee webhooks.
- Added new custom regex based check.
- Allow to configure contributing to shared translation memory.
- Added ZIP download for more translation files.
- Make XLIFF standard compliant parsing of maxwidth and font.
- Added new check and fixer for safe HTML markup for translating web applications.
- Add component alert on unsupported configuration.
- İlk çevirileri başlatmak için otomatik çeviri eklentisi eklendi.
- Extend automatic translation to add suggestions.
- Display add-on parameters on overview.
- Sentry is now supported through modern Sentry SDK instead of Raven.
- Changed example settings to be better fit for production environment.
- Added automated backups using BorgBackup.
- Split cleanup add-on for RESX to avoid unwanted file updates.
- Added advanced search capabilities.
- Allow users to download their own reports.
- Added localization guide to help configuring components.
- Added support for GitLab merge requests.
- Improved display of repository status.
- Perform automated translation in the background.

4.40.11 Weblate 3.8

Released on August 15th 2019.

- Added support for simplified creating of similar components.
- Added support for parsing translation flags from the XML based file formats.
- Log exceptions into Celery log.
- Depo kapsamındaki eklentilerin performansı iyileştirildi.
- Improved look of notification e-mails.
- Fixed password reset behavior.
- Improved performance on most of translation pages.
- Fixed listing of languages not known to Weblate.
- Keşfedilen bileşenlere eklentileri klonlamak için destek eklendi.
- Add support for replacing file content with uploaded.
- Add support for translating non VCS based content.
- Added OpenGraph widget image to use on social networks.

- Added support for animated screenshots.
- Improved handling of monolingual XLIFF files.
- Avoid sending multiple notifications for single event.
- Add support for filtering changes.
- Extended predefined periods for reporting.
- Added webhook support for Azure Repos.
- New opt-in notifications on pending suggestions or untranslated strings.
- Add one click unsubscribe link to notification e-mails.
- Fixed false positives with Has been translated check.
- New management interface for admins.
- String priority can now be specified using flags.
- Added language management views.
- Add checks for Qt library and Ruby format strings.
- Added configuration to better fit single project installations.
- Notify about new string on source string change on monolingual translations.
- Added separate view for translation memory with search capability.

4.40.12 Weblate 3.7.1

Released on June 28th 2019.

- Documentation updates.
- Fixed some requirements constraints.
- Updated language database.
- Localization updates.
- Various user interface tweaks.
- Improved handling of unsupported but discovered translation files.
- More verbosely report missing file format requirements.

4.40.13 Weblate 3.7

Released on June 21st 2019.

- Added separate Celery queue for notifications.
- Use consistent look with application for API browsing.
- Include approved stats in the reports.
- Report progress when updating translation component.
- Allow to abort running background component update.
- Extend template language for filename manipulations.
- Use templates for editor link and repository browser URL.
- Indicate max length and current characters count when editing translation.
- Improved handling of abbreviations in unchanged translation check.

- Refreshed landing page for new contributors.
- msgmerge eklentisini yapılandırmak için destek eklendi.
- Delay opening SMTP connection when sending notifications.
- Improved error logging.
- MO oluşturma eklentisinde özel konuma izin verildi.
- Added add-ons to cleanup old suggestions or comments.
- Added option to enable horizontal mode in the Zen editor.
- Improved import performance with many linked components.
- Fixed examples installation in some cases.
- Improved rendering of alerts in changes.
- Added new horizontal stats widget.
- Improved format strings check on plurals.
- Added font management tool.
- New check for rendered text dimensions.
- Added support for subtitle formats.
- Include overall completion stats for languages.
- Added reporting at project and global scope.
- Improved user interface when showing translation status.
- New Weblate logo and color scheme.
- New look of bitmap badges.

4.40.14 Weblate 3.6.1

Released on April 26th 2019.

- Improved handling of monolingual XLIFF files.
- Fixed digest notifications in some corner cases.
- Fixed add-on script error alert.
- Fixed generating MO file for monolingual PO files.
- Fixed display of uninstalled checks.
- Indicate administered projects on project listing.
- Allow update to recover from missing VCS repository.

4.40.15 Weblate 3.6

Released on April 20th 2019.

- Add support for downloading user data.
- Add-ons are now automatically triggered upon installation.
- Improved instructions for resolving merge conflicts.
- Cleanup add-on is now compatible with app store metadata translations.
- Configurable language code syntax when adding new translations.

- Warn about using Python 2 with planned termination of support in April 2020.
- Extract special characters from the source string for visual keyboard.
- Extended contributor stats to reflect both source and target counts.
- Admins and consistency add-ons can now add translations even if disabled for users.
- Fixed description of toggle disabling Language–Team header manipulation.
- Notify users mentioned in comments.
- Removed file format autodetection from component setup.
- Fixed generating MO file for monolingual PO files.
- Added digest notifications.
- Added support for muting component notifications.
- Added notifications for new alerts, whiteboard messages or components.
- Notifications for administered projects can now be configured.
- Improved handling of three letter language codes.

4.40.16 Weblate 3.5.1

Released on March 10th 2019.

- Fixed Celery systemd unit example.
- Fixed notifications from HTTP repositories with login.
- Fixed race condition in editing source string for monolingual translations.
- Include output of failed add-on execution in the logs.
- Improved validation of choices for adding new language.
- Allow to edit file format in component settings.
- Update installation instructions to prefer Python 3.
- Performance and consistency improvements for loading translations.
- Make Microsoft Terminology service compatible with current Zeep releases.
- Localization updates.

4.40.17 Weblate 3.5

Released on March 3rd 2019.

- Improved performance of built-in translation memory.
- Added interface to manage global translation memory.
- Improved alerting on bad component state.
- Added user interface to manage whiteboard messages.
- Eklenti işleme mesajı artık yapılandırılabilir.
- Reduce number of commits when updating upstream repository.
- Fixed possible metadata loss when moving component between projects.
- Improved navigation in the Zen mode.
- Added several new quality checks (Markdown related and URL).

- Added support for app store metadata files.
- Added support for toggling GitHub or Gerrit integration.
- Added check for Kashida letters.
- Added option to squash commits based on authors.
- Improved support for XLSX file format.
- Compatibility with Tesseract 4.0.
- Billing add-on now removes projects for unpaid billings after 45 days.

4.40.18 Weblate 3.4

Released on January 22nd 2019.

- Added support for XLIFF placeholders.
- Celery can now utilize multiple task queues.
- Added support for renaming and moving projects and components.
- Include characters counts in reports.
- Added guided adding of translation components with automatic detection of translation files.
- Customizable merge commit messages for Git.
- Added visual indication of component alerts in navigation.
- Improved performance of loading translation files.
- Yollamadan önce Git işlemlerini toparlamak için yeni eklenti.
- Improved displaying of translation changes.
- Changed default merge style to rebase and made that configurable.
- Better handle private use subtags in language code.
- Improved performance of fulltext index updates.
- Extended file upload API to support more parameters.

4.40.19 Weblate 3.3

Released on November 30th 2018.

- Added support for component and project removal.
- Improved performance for some monolingual translations.
- Added translation component alerts to highlight problems with a translation.
- Expose XLIFF string rename as context when available.
- Added support for XLIFF states.
- Added check for non writable files in DATA_DIR.
- Improved CSV export for changes.

4.40.20 Weblate 3.2.2

Released on October 20th 2018.

- Remove no longer needed Babel dependency.
- Updated language definitions.
- Eklentiler, LDAP ve Celery için belgeler iyileştirildi.
- Fixed enabling new dos-eol and auto-java-messageformat flags.
- Fixed running setup.py test from PyPI package.
- Improved plurals handling.
- Fixed translation upload API failure in some corner cases.
- Fixed updating Git configuration in case it was changed manually.

4.40.21 Weblate 3.2.1

Released on October 10th 2018.

- Document dependency on backports.csv on Python 2.7.
- Fix running tests under root.
- Improved error handling in gitexport module.
- Fixed progress reporting for newly added languages.
- Correctly report Celery worker errors to Sentry.
- Fixed creating new translations with Qt Linguist.
- Fixed occasional fulltext index update failures.
- Improved validation when creating new components.
- Added support for cleanup of old suggestions.

4.40.22 Weblate 3.2

Released on October 6th 2018.

- Add install_addon management command for automated add-on installation.
- Allow more fine grained ratelimit settings.
- Added support for export and import of Excel files.
- Improve component cleanup in case of multiple component discovery add-ons.
- Rewritten Microsoft Terminology machine translation backend.
- Weblate now uses Celery to offload some processing.
- Improved search capabilities and added regular expression search.
- Added support for Youdao Zhiyun API machine translation.
- Added support for Baidu API machine translation.
- Integrated maintenance and cleanup tasks using Celery.
- Improved performance of loading translations by almost 25%.
- Removed support for merging headers on upload.
- Removed support for custom commit messages.

- Configurable editing mode (zen/full).
- Added support for error reporting to Sentry.
- Added support for automated daily update of repositories.
- Added support for creating projects and components by users.
- Dahili çeviri belleği artık yapılan çevirileri otomatik olarak saklıyor.
- Users and projects can import their existing translation memories.
- Better management of related strings for screenshots.
- Added support for checking Java MessageFormat.

See [3.2 milestone on GitHub](#) for detailed list of addressed issues.

4.40.23 Weblate 3.1.1

Released on July 27th 2018.

- Fix testsuite failure on some setups.

4.40.24 Weblate 3.1

Released on July 27th 2018.

- Upgrades from older version than 3.0.1 are not supported.
- Allow to override default commit messages from settings.
- Improve webhooks compatibility with self hosted environments.
- Added support for Amazon Translate.
- Compatibility with Django 2.1.
- Django system checks are now used to diagnose problems with installation.
- Removed support for soon shutdown libavatar service.
- Değiştirilmemiş çevirileri “Düzenleme gerekli” olarak işaretlemek için yeni eklenti.
- Add support for jumping to specific location while translating.
- Downloaded translations can now be customized.
- Improved calculation of string similarity in translation memory matches.
- Added support by signing Git commits by GnuPG.

4.40.25 Weblate 3.0.1

Released on June 10th 2018.

- Fixed possible migration issue from 2.20.
- Localization updates.
- Removed obsolete hook examples.
- Improved caching documentation.
- Fixed displaying of admin documentation.
- Improved handling of long language names.

4.40.26 Weblate 3.0

Released on June 1st 2018.

- Rewritten access control.
- Several code cleanups that lead to moved and renamed modules.
- New add-on for automatic component discovery.
- The `import_project` management command has now slightly different parameters.
- Added basic support for Windows RC files.
- New add-on to store contributor names in PO file headers.
- The per component hook scripts are removed, use add-ons instead.
- Add support for collecting contributor agreements.
- Access control changes are now tracked in history.
- Bir projedeki tüm bileşenlerin aynı çevirilere sahip olmasını sağlamak için yeni eklenti.
- Support for more variables in commit message templates.
- Add support for providing additional textual context.

4.41 Weblate 2.x series

4.41.1 Weblate 2.20

Released on April 4th 2018.

- Improved speed of cloning subversion repositories.
- Changed repository locking to use third party library.
- Added support for downloading only strings needing action.
- Added support for searching in several languages at once.
- New add-on to configure gettext output wrapping.
- New add-on to configure JSON formatting.
- Added support for authentication in API using RFC 6750 compatible Bearer authentication.
- Added support for automatic translation using machine translation services.
- Added support for HTML markup in whiteboard messages.
- Added support for mass changing state of strings.
- Translate-toolkit at least 2.3.0 is now required, older versions are no longer supported.
- Dahili çeviri belleği eklendi.
- Added componentlists overview to dashboard and per component list overview pages.
- Added support for DeepL machine translation service.
- Machine translation results are now cached inside Weblate.
- İşlenmiş değişiklikleri yeniden sıralama desteği eklendi.

4.41.2 Weblate 2.19.1

Released on February 20th 2018.

- Fixed migration issue on upgrade from 2.18.
- Improved file upload API validation.

4.41.3 Weblate 2.19

Released on February 15th 2018.

- Fixed imports across some file formats.
- Display human friendly browser information in audit log.
- Added TMX exporter for files.
- Various performance improvements for loading translation files.
- Added option to disable access management in Weblate in favor of Django one.
- Improved glossary lookup speed for large strings.
- Compatibility with django_auth_ldap 1.3.0.
- Configuration errors are now stored and reported persistently.
- Honor ignore flags in whitespace autofixer.
- Improved compatibility with some Subversion setups.
- Dahili makine çevirisi hizmeti geliştirildi.
- Added support for SAP Translation Hub service.
- Added support for Microsoft Terminology service.
- Removed support for advertisement in notification e-mails.
- Improved translation progress reporting at language level.
- Improved support for different plural formulas.
- Added support for Subversion repositories not using stdlayout.
- Çeviri iş akışlarını özelleştirmek için eklentiler eklendi.

4.41.4 Weblate 2.18

Released on December 15th 2017.

- Extended contributor stats.
- Improved configuration of special characters virtual keyboard.
- Added support for DTD file format.
- Changed keyboard shortcuts to less likely collide with browser/system ones.
- Improved support for approved flag in XLIFF files.
- Added support for not wrapping long strings in gettext PO files.
- Added button to copy permalink for current translation.
- Dropped support for Django 1.10 and added support for Django 2.0.
- Removed locking of translations while translating.
- Added support for adding new strings to monolingual translations.

- Added support for translation workflows with dedicated reviewers.

4.41.5 Weblate 2.17.1

Released on October 13th 2017.

- Fixed running testsuite in some specific situations.
- Locales updates.

4.41.6 Weblate 2.17

Released on October 13th 2017.

- Weblate by default does shallow Git clones now.
- Improved performance when updating large translation files.
- Added support for blocking certain e-mails from registration.
- Users can now delete their own comments.
- Added preview step to search and replace feature.
- Client side persistence of settings in search and upload forms.
- Extended search capabilities.
- More fine grained per project ACL configuration.
- Default value of BASE_DIR has been changed.
- Added two step account removal to prevent accidental removal.
- Project access control settings is now editable.
- Added optional spam protection for suggestions using Akismet.

4.41.7 Weblate 2.16

Released on August 11th 2017.

- Various performance improvements.
- Added support for nested JSON format.
- Added support for WebExtension JSON format.
- Fixed git exporter authentication.
- Improved CSV import in certain situations.
- Improved look of Other translations widget.
- The max-length checks is now enforcing length of text in form.
- Make the commit_pending age configurable per component.
- Various user interface cleanups.
- Fixed component/project/site wide search for translations.

4.41.8 Weblate 2.15

Released on June 30th 2017.

- Show more related translations in other translations.
- Add option to see translations of current string to other languages.
- Use 4 plural forms for Lithuanian by default.
- Fixed upload for monolingual files of different format.
- Improved error messages on failed authentication.
- Keep page state when removing word from glossary.
- Added direct link to edit secondary language translation.
- Added Perl format quality check.
- Added support for rejecting reused passwords.
- Extended toolbar for editing RTL languages.

4.41.9 Weblate 2.14.1

Released on May 24th 2017.

- Fixed possible error when paginating search results.
- Fixed migrations from older versions in some corner cases.
- Fixed possible CSRF on project watch and unwatch.
- The password reset no longer authenticates user.
- Fixed possible CAPTCHA bypass on forgotten password.

4.41.10 Weblate 2.14

Released on May 17th 2017.

- Add glossary entries using AJAX.
- The logout now uses POST to avoid CSRF.
- The API key token reset now uses POST to avoid CSRF.
- Weblate sets Content-Security-Policy by default.
- The local editor URL is validated to avoid self-XSS.
- The password is now validated against common flaws by default.
- Notify users about important activity with their account such as password change.
- The CSV exports now escape potential formulas.
- Various minor improvements in security.
- The authentication attempts are now rate limited.
- Suggestion content is stored in the history.
- Store important account activity in audit log.
- Ask for password confirmation when removing account or adding new associations.
- Show time when suggestion has been made.
- There is new quality check for trailing semicolon.

- Ensure that search links can be shared.
- Included source string information and screenshots in the API.
- Allow to overwrite translations through API upload.

4.41.11 Weblate 2.13.1

Released on Apr 12th 2017.

- Fixed listing of managed projects in profile.
- Fixed migration issue where some permissions were missing.
- Fixed listing of current file format in translation download.
- Return HTTP 404 when trying to access project where user lacks privileges.

4.41.12 Weblate 2.13

Released on Apr 12th 2017.

- Fixed quality checks on translation templates.
- Added quality check to trigger on losing translation.
- Add option to view pending suggestions from user.
- Add option to automatically build component lists.
- Default dashboard for unauthenticated users can be configured.
- Add option to browse 25 random strings for review.
- History now indicates string change.
- Better error reporting when adding new translation.
- Added per language search within project.
- Group ACLs can now be limited to certain permissions.
- The per project ACLs are now implemented using Group ACL.
- Added more fine grained privileges control.
- Various minor UI improvements.

4.41.13 Weblate 2.12

Released on Mar 3rd 2017.

- Improved admin interface for groups.
- Added support for Yandex Translate API.
- Improved speed of site wide search.
- Added project and component wide search.
- Added project and component wide search and replace.
- Improved rendering of inconsistent translations.
- Added support for opening source files in local editor.
- Added support for configuring visual keyboard with special characters.
- Improved screenshot management with OCR support for matching source strings.

- Default commit message now includes translation information and URL.
- Added support for Joomla translation format.
- Improved reliability of import across file formats.

4.41.14 Weblate 2.11

Released on Jan 31st 2017.

- Include language detailed information on language page.
- Mercurial backend improvements.
- Added option to specify translation component priority.
- More consistent usage of Group ACL even with less used permissions.
- Added WL_BRANCH variable to hook scripts.
- Improved developer documentation.
- Better compatibility with various Git versions in Git exporter add-on.
- Included per project and component stats.
- Added language code mapping for better support of Microsoft Translate API.
- Moved fulltext cleanup to background job to make translation removal faster.
- Fixed displaying of plural source for languages with single plural form.
- Improved error handling in import_project.
- Various performance improvements.

4.41.15 Weblate 2.10.1

Released on Jan 20th 2017.

- Do not leak account existence on password reset form (CVE-2017-5537).

4.41.16 Weblate 2.10

Released on Dec 15th 2016.

- Added quality check to check whether plurals are translated differently.
- Fixed GitHub hooks for repositories with authentication.
- Added optional Git exporter module.
- Support for Microsoft Cognitive Services Translator API.
- Simplified project and component user interface.
- Added automatic fix to remove control characters.
- Added per language overview to project.
- Added support for CSV export.
- Added CSV download for stats.
- Added matrix view for quick overview of all translations.
- Added basic API for changes and strings.
- Added support for Apertium APy server for machine translations.

4.41.17 Weblate 2.9

Released on Nov 4th 2016.

- Extended parameters for createadmin management command.
- Extended import_json to be able to handle with existing components.
- Added support for YAML files.
- Project owners can now configure translation component and project details.
- Use “Watched” instead of “Subscribed” projects.
- Projects can be watched directly from project page.
- Added multi language status widget.
- Highlight secondary language if not showing source.
- Record suggestion deletion in history.
- Improved UX of languages selection in profile.
- Fixed showing whiteboard messages for component.
- Keep preferences tab selected after saving.
- Show source string comment more prominently.
- Automatically install Gettext PO merge driver for Git repositories.
- Added search and replace feature.
- Added support for uploading visual context (screenshots) for translations.

4.41.18 Weblate 2.8

Released on Aug 31st 2016.

- Belgelendirme iyileştirmeleri.
- Translations.
- Updated bundled JavaScript libraries.
- Added list_translators management command.
- Django 1.8 is no longer supported.
- Fixed compatibility with Django 1.10.
- Added Subversion support.
- Separated XML validity check from XML mismatched tags.
- Fixed API to honor HIDE_REPO_CREDENTIALS settings.
- Show source change in Zen mode.
- Alt+PageUp/PageDown/Home/End now works in Zen mode as well.
- Add tooltip showing exact time of changes.
- Add option to select filters and search from translation page.
- Added UI for translation removal.
- Improved behavior when inserting placeables.
- Fixed auto locking issues in Zen mode.

4.41.19 Weblate 2.7

Released on Jul 10th 2016.

- Removed Google web translate machine translation.
- Improved commit message when adding translation.
- Fixed Google Translate API for Hebrew language.
- Compatibility with Mercurial 3.8.
- Added import_json management command.
- Correct ordering of listed translations.
- Show full suggestion text, not only a diff.
- Extend API (detailed repository status, statistics, ...).
- Testsuite no longer requires network access to test repositories.

4.41.20 Weblate 2.6

Released on Apr 28th 2016.

- Fixed validation of components with language filter.
- Improved support for XLIFF files.
- Fixed machine translation for non English sources.
- Added REST API.
- Django 1.10 compatibility.
- Added categories to whiteboard messages.

4.41.21 Weblate 2.5

Released on Mar 10th 2016.

- Fixed automatic translation for project owners.
- Improved performance of commit and push operations.
- New management command to add suggestions from command-line.
- Added support for merging comments on file upload.
- Added support for some GNU extensions to C printf format.
- Belgelendirme iyileştirmeleri.
- Added support for generating translator credits.
- Added support for generating contributor stats.
- Site wide search can search only in one language.
- Improve quality checks for Armenian.
- Support for starting translation components without existing translations.
- Support for adding new translations in Qt TS.
- Improved support for translating PHP files.
- Performance improvements for quality checks.
- Kusur denetimleri için site genelinde arama düzeltildi.

- Added option to specify source language.
- Improved support for XLIFF files.
- Extended list of options for `import_project`.
- Improved targeting for whiteboard messages.
- Support for automatic translation across projects.
- Optimized fulltext search index.
- Added management command for auto translation.
- Added placeables highlighting.
- Added keyboard shortcuts for placeables, checks and machine translations.
- Improved translation locking.
- Added quality check for AngularJS interpolation.
- Added extensive group based ACLs.
- Clarified terminology on strings needing edit (formerly fuzzy).
- Clarified terminology on strings needing action and untranslated strings.
- Support for Python 3.
- Dropped support for Django 1.7.
- Dropped dependency on msginit for creating new gettext PO files.
- Added configurable dashboard views.
- Improved notifications on parse errors.
- Added option to import components with duplicate name to `import_project`.
- Improved support for translating PHP files.
- Added XLIFF export for dictionary.
- Added XLIFF and gettext PO export for all translations.
- Belgelendirme iyileştirmeleri.
- Added support for configurable automatic group assignments.
- Improved adding of new translations.

4.41.22 Weblate 2.4

Released on Sep 20th 2015.

- Improved support for PHP files.
- Ability to add ACL to anonymous user.
- Improved configurability of `import_project` command.
- Added CSV dump of history.
- Avoid copy/paste errors with whitespace characters.
- Added support for Bitbucket webhooks.
- Tighter control on fuzzy strings on translation upload.
- Several URLs have changed, you might have to update your bookmarks.
- Hook scripts are executed with VCS root as current directory.
- Hook scripts are executed with environment variables describing current component.

- Add management command to optimize fulltext index.
- Added support for error reporting to Rollbar.
- Projects now can have multiple owners.
- Project owners can manage themselves.
- Added support for `javascript-format` used in gettext PO.
- Support for adding new translations in XLIFF.
- Improved file format autodetection.
- Extended keyboard shortcuts.
- Improved dictionary matching for several languages.
- Improved layout of most of pages.
- Support for adding words to dictionary while translating.
- Added support for filtering languages to be managed by Weblate.
- Added support for translating and importing CSV files.
- Rewritten handling of static files.
- Direct login/registration links to third-party service if that's the only one.
- Commit pending changes on account removal.
- Add management command to change site name.
- Add option to configure default committer.
- Add hook after adding new translation.
- Add option to specify multiple files to add to commit.

4.41.23 Weblate 2.3

Released on May 22nd 2015.

- Dropped support for Django 1.6 and South migrations.
- Support for adding new translations when using Java Property files.
- Allow to accept suggestion without editing.
- Improved support for Google OAuth 2.0.
- Added support for Microsoft .resx files.
- Tuned default robots.txt to disallow big crawling of translations.
- Simplified workflow for accepting suggestions.
- Added project owners who always receive important notifications.
- Allow to disable editing of monolingual template.
- More detailed repository status view.
- Direct link for editing template when changing translation.
- Allow to add more permissions to project owners.
- Allow to show secondary language in Zen mode.
- Support for hiding source string in favor of secondary language.

4.41.24 Weblate 2.2

Released on Feb 19th 2015.

- Performans iyileştirmeleri.
- Fulltext search on location and comments fields.
- Yeni SVG/JavaScript tabanlı etkinlik çizelgeleri.
- Support for Django 1.8.
- Support for deleting comments.
- Added own SVG badge.
- Added support for Google Analytics.
- Improved handling of translation filenames.
- Added support for monolingual JSON translations.
- Record component locking in a history.
- Support for editing source (template) language for monolingual translations.
- Added basic support for Gerrit.

4.41.25 Weblate 2.1

Released on Dec 5th 2014.

- Added support for Mercurial repositories.
- Replaced Glyphicon font by Awesome.
- Added icons for social authentication services.
- Better consistency of button colors and icons.
- Belgelendirme iyileştirmeleri.
- Various bugfixes.
- Automatic hiding of columns in translation listing for small screens.
- Changed configuration of filesystem paths.
- Improved SSH keys handling and storage.
- Improved repository locking.
- Customizable quality checks per source string.
- Allow to hide completed translations from dashboard.

4.41.26 Weblate 2.0

Released on Nov 6th 2014.

- New responsive UI using Bootstrap.
- Rewritten VCS backend.
- Belgelendirme iyileştirmeleri.
- Added whiteboard for site wide messages.
- Configurable strings priority.
- Added support for JSON file format.

- Fixed generating mo files in certain cases.
- Added support for GitLab notifications.
- Added support for disabling translation suggestions.
- Django 1.7 support.
- ACL projects now have user management.
- Extended search possibilities.
- Give more hints to translators about plurals.
- Fixed Git repository locking.
- Compatibility with older Git versions.
- Improved ACL support.
- Added buttons for per language quotes and other special characters.
- Support for exporting stats as JSONP.

4.42 Weblate 1.x series

4.42.1 Weblate 1.9

Released on May 6th 2014.

- Django 1.6 compatibility.
- No longer maintained compatibility with Django 1.4.
- Management commands for locking/unlocking translations.
- Improved support for Qt TS files.
- Users can now delete their account.
- Avatars can be disabled.
- Merged first and last name attributes.
- Avatars are now fetched and cached server side.
- Added support for shields.io badge.

4.42.2 Weblate 1.8

Released on November 7th 2013.

- Please check manual for upgrade instructions.
- Nicer listing of project summary.
- Better visible options for sharing.
- More control over anonymous users privileges.
- Supports login using third party services, check manual for more details.
- Users can login by e-mail instead of username.
- Belgelendirme iyileştirmeleri.
- Improved source strings review.
- Searching across all strings.

- Better tracking of source strings.
- Captcha protection for registration.

4.42.3 Weblate 1.7

Released on October 7th 2013.

- Please check manual for upgrade instructions.
- Support for checking Python brace format string.
- Per component customization of quality checks.
- Detailed per translation stats.
- Changed way of linking suggestions, checks and comments to strings.
- Users can now add text to commit message.
- Support for subscribing on new language requests.
- Support for adding new translations.
- Widgets and charts are now rendered using Pillow instead of Pango + Cairo.
- Add status badge widget.
- Dropped invalid text direction check.
- Changes in dictionary are now logged in history.
- Çeviri görünümü için performans iyileştirmeleri.

4.42.4 Weblate 1.6

Released on July 25th 2013.

- Nicer error handling on registration.
- Browsing of changes.
- Fixed sorting of machine translation suggestions.
- Improved support for MyMemory machine translation.
- Added support for Amagama machine translation.
- Various optimizations on frequently used pages.
- Highlights searched phrase in search results.
- Support for automatic fixups while saving the message.
- Tracking of translation history and option to revert it.
- Added support for Google Translate API.
- Added support for managing SSH host keys.
- Various form validation improvements.
- Various quality checks improvements.
- Performance improvements for import.
- Added support for voting on suggestions.
- Cleanup of admin interface.

4.42.5 Weblate 1.5

Released on April 16th 2013.

- Please check manual for upgrade instructions.
- Added public user pages.
- Better naming of plural forms.
- Added support for TBX export of glossary.
- Added support for Bitbucket notifications.
- Activity charts are now available for each translation, language or user.
- Extended options of `import_project` admin command.
- Compatible with Django 1.5.
- Avatars are now shown using libavatar.
- Added possibility to pretty print JSON export.
- Various performance improvements.
- Indicate failing checks or fuzzy strings in progress bars for projects or languages as well.
- Added support for custom pre-commit hooks and committing additional files.
- Rewritten search for better performance and user experience.
- New interface for machine translations.
- Added support for monolingual po files.
- Extend amount of cached metadata to improve speed of various searches.
- Now shows word counts as well.

4.42.6 Weblate 1.4

Released on January 23rd 2013.

- Fixed deleting of checks/comments on string deletion.
- Added option to disable automatic propagation of translations.
- Added option to subscribe for merge failures.
- Correctly import on projects which needs custom ttkit loader.
- Added sitemaps to allow easier access by crawlers.
- Provide direct links to string in notification e-mails or feeds.
- Various improvements to admin interface.
- Provide hints for production setup in admin interface.
- Added per language widgets and engage page.
- Improved translation locking handling.
- Show code snippets for widgets in more variants.
- Indicate failing checks or fuzzy strings in progress bars.
- More options for formatting commit message.
- Fixed error handling with machine translation services.
- Improved automatic translation locking behaviour.

- Support for showing changes from previous source string.
- Added support for substring search.
- Various quality checks improvements.
- Support for per project ACL.
- Basic code coverage by unit tests.

4.42.7 Weblate 1.3

Released on November 16th 2012.

- Compatibility with PostgreSQL database backend.
- Removes languages removed in upstream git repository.
- Improved quality checks processing.
- Added new checks (BBCode, XML markup and newlines).
- Support for optional rebasing instead of merge.
- Possibility to relocate Weblate (for example to run it under /weblate path).
- Support for manually choosing file type in case autodetection fails.
- Better support for Android resources.
- Support for generating SSH key from web interface.
- More visible data exports.
- New buttons to enter some special characters.
- Support for exporting dictionary.
- Support for locking down whole Weblate installation.
- Checks for source strings and support for source strings review.
- Support for user comments for both translations and source strings.
- Better changes log tracking.
- Changes can now be monitored using RSS.
- Improved support for RTL languages.

4.42.8 Weblate 1.2

Released on August 14th 2012.

- Weblate now uses South for database migration, please check upgrade instructions if you are upgrading.
- Fixed minor issues with linked git repos.
- New introduction page for engaging people with translating using Weblate.
- Added widgets which can be used for promoting translation projects.
- Added option to reset repository to origin (for privileged users).
- Project or component can now be locked for translations.
- Possibility to disable some translations.
- Configurable options for adding new translations.
- Configuration of git commits per project.

- Simple antispam protection.
- Better layout of main page.
- Support for automatically pushing changes on every commit.
- Support for e-mail notifications of translators.
- List only used languages in preferences.
- Improved handling of not known languages when importing project.
- Support for locking translation by translator.
- Optionally maintain `Language-Team` header in po file.
- Include some statistics in about page.
- Supports (and requires) django-registration 0.8.
- Kusur denetimleri olan dizgilerin sayısının önbellege alınması.
- Checking of requirements during setup.
- Belgelendirme iyileştirmeleri.

4.42.9 Weblate 1.1

Released on July 4th 2012.

- Improved several translations.
- Better validation while creating component.
- Added support for shared git repositories across components.
- Do not necessary commit on every attempt to pull remote repo.
- Added support for offloading indexing.

4.42.10 Weblate 1.0

Released on May 10th 2012.

- Improved validation while adding/saving component.
- Experimental support for Android component files (needs patched ttkit).
- Updates from hooks are run in background.
- Improved installation instructions.
- Improved navigation in dictionary.

4.43 Weblate 0.x series

4.43.1 Weblate 0.9

Released on April 18th 2012.

- Fixed import of unknown languages.
- Improved listing of nearby messages.
- Improved several checks.
- Documentation updates.

- Added definition for several more languages.
- Various code cleanups.
- Belgelendirme iyileştirmeleri.
- Changed file layout.
- Update helper scripts to Django 1.4.
- Improved navigation while translating.
- Better handling of po file renames.
- Better validation while creating component.
- Integrated full setup into syncdb.
- Added list of recent changes to all translation pages.
- Check for untranslated strings ignores format string only messages.

4.43.2 Weblate 0.8

Released on April 3rd 2012.

- Replaced own full text search with Whoosh.
- Various fixes and improvements to checks.
- New command updatechecks.
- Lot of translation updates.
- Added dictionary for storing most frequently used terms.
- Added /admin/report/ for overview of repositories status.
- Machine translation services no longer block page loading.
- Management interface now contains also useful actions to update data.
- Records log of changes made by users.
- Ability to postpone commit to Git to generate less commits from single user.
- Possibility to browse failing checks.
- Automatic translation using already translated strings.
- New about page showing used versions.
- Django 1.4 compatibility.
- Ability to push changes to remote repo from web interface.
- Added review of translations done by others.

4.43.3 Weblate 0.7

Released on February 16th 2012.

- Direct support for GitHub notifications.
- Added support for cleaning up orphaned checks and translations.
- Displays nearby strings while translating.
- Displays similar strings while translating.
- Improved searching for string.

4.43.4 Weblate 0.6

Released on February 14th 2012.

- Added various checks for translated messages.
- Tunable access control.
- Improved handling of translations with new lines.
- Added client side sorting of tables.
- Please check upgrading instructions in case you are upgrading.

4.43.5 Weblate 0.5

Released on February 12th 2012.

- **Support for machine translation using following online services:**
 - Apertium
 - Microsoft Translator
 - MyMemory
- Several new translations.
- Improved merging of upstream changes.
- Better handle concurrent git pull and translation.
- Propagating works for fuzzy changes as well.
- Propagating works also for file upload.
- Fixed file downloads while using FastCGI (and possibly others).

4.43.6 Weblate 0.4

Released on February 8th 2012.

- Added usage guide to documentation.
- Fixed API hooks not to require CSRF protection.

4.43.7 Weblate 0.3

Released on February 8th 2012.

- Better display of source for plural translations.
- New documentation in Sphinx format.
- Displays secondary languages while translating.
- Improved error page to give list of existing projects.
- New per language stats.

4.43.8 Weblate 0.2

Released on February 7th 2012.

- Improved validation of several forms.
- Warn users on profile upgrade.
- Remember URL for login.
- Naming of text areas while entering plural forms.
- Automatic expanding of translation area.

4.43.9 Weblate 0.1

Released on February 6th 2012.

- Initial release.

W

`wlc`, [154](#)

`wlc.config`, [155](#)

`wlc.main`, [156](#)

HTTP Routing Table

/	GET /api/components/(string:project)/(string:component:slug)/, 130
ANY /, 102	GET /api/components/(string:project)/(string:component:slug)/, 126
/api	GET /api/components/(string:project)/(string:component:slug)/, 128
GET /api/, 105	GET /api/components/(string:project)/(string:component:slug)/, 128
/api/addons	GET /api/components/(string:project)/(string:component:slug)/, 127
GET /api/addons/, 142	GET /api/components/(string:project)/(string:component:slug)/, 126
GET /api/addons/(int:id)/, 142	GET /api/components/(string:project)/(string:component:slug)/, 130
PUT /api/addons/(int:id)/, 143	GET /api/components/(string:project)/(string:component:slug)/, 129
DELETE /api/addons/(int:id)/, 143	POST /api/components/(string:project)/(string:component:slug)/, 142
PATCH /api/addons/(int:id)/, 142	POST /api/components/(string:project)/(string:component:slug)/, 131
/api/changes	POST /api/components/(string:project)/(string:component:slug)/, 126
GET /api/changes/, 139	POST /api/components/(string:project)/(string:component:slug)/, 128
GET /api/changes/(int:id)/, 139	POST /api/components/(string:project)/(string:component:slug)/, 129
/api/component-lists	PUT /api/components/(string:project)/(string:component:slug)/, 125
GET /api/component-lists/, 143	DELETE /api/components/(string:project)/(string:component:slug)/, 125
GET /api/component-lists/(str:slug)/, 143	DELETE /api/components/(string:project)/(string:component:slug)/, 131
POST /api/component-lists/(str:slug)/components/, 144	PATCH /api/components/(string:project)/(string:component:slug)/, 124
PUT /api/component-lists/(str:slug)/, 143	/api/groups
DELETE /api/component-lists/(str:slug)/, 144	GET /api/groups/, 109
DELETE /api/component-lists/(str:slug)/components/(string:component:slug)/, 144	GET /api/groups/(int:id)/, 109
PATCH /api/component-lists/(str:slug)/, 143	POST /api/groups/, 109
/api/components	POST /api/groups/(int:id)/changes/, 112
GET /api/components/, 121	POST /api/groups/(int:id)/components/, 111
GET /api/components/(string:project)/(string:component:slug)/, 121	
GET /api/components/(string:project)/(string:component:slug)/, 125	
GET /api/components/(string:project)/(string:component:slug)/, 126	

POST /api/groups/(int:id)/languages/, 111
 PATCH /api/projects/(string:project)/, 116

POST /api/groups/(int:id)/projects/, 111

POST /api/groups/(int:id)/roles/, 111

PUT /api/groups/(int:id)/, 110

DELETE /api/groups/(int:id)/, 110

DELETE /api/groups/(int:id)/componentlinks/(int:componentlink_id)/, 112

DELETE /api/groups/(int:id)/components/(int:component_id)/, 111
 PATCH /api/components/(int:id)/, 113

DELETE /api/groups/(int:id)/languages/(string:language)/, 111

DELETE /api/groups/(int:id)/projects/(int:project_id)/, 111

PATCH /api/groups/(int:id)/, 110

/api/languages

GET /api/languages/, 113

GET /api/languages/(string:language)/, 113

GET /api/languages/(string:language)/statistics/, 115

POST /api/languages/, 113

PUT /api/languages/(string:language)/, 114

DELETE /api/languages/(string:language)/, 114

PATCH /api/languages/(string:language)/, 114

/api/metrics

GET /api/metrics/, 145

/api/projects

GET /api/projects/, 115

GET /api/projects/(string:project)/, 115

GET /api/projects/(string:project)/changes/, 116

GET /api/projects/(string:project)/components/, 118

GET /api/projects/(string:project)/languages/, 120

GET /api/projects/(string:project)/repository/, 117

GET /api/projects/(string:project)/statistics/, 121

POST /api/projects/, 115

POST /api/projects/(string:project)/components/, 118

POST /api/projects/(string:project)/repository/, 117

PUT /api/projects/(string:project)/, 116

DELETE /api/projects/(string:project)/, 116

/api/roles

GET /api/roles/, 112

GET /api/roles/(int:id)/, 112

POST /api/roles/, 112

DELETE /api/roles/(int:id)/, 113

PATCH /api/roles/(int:id)/, 113

/api/screenshots

GET /api/screenshots/, 140

GET /api/screenshots/(int:id)/, 140

GET /api/screenshots/(int:id)/file/, 140

POST /api/screenshots/, 141

POST /api/screenshots/(int:id)/file/, 140

POST /api/screenshots/(int:id)/units/, 140

DELETE /api/screenshots/(int:id)/, 141

DELETE /api/screenshots/(int:id)/units/(int:unit_id)/, 142

PATCH /api/screenshots/(int:id)/, 141

DELETE /api/screenshots/(int:id)/units/(int:unit_id)/, 142

PATCH /api/screenshots/(int:id)/, 141

/api/tasks

GET /api/tasks/, 144

GET /api/tasks/(str:uuid)/, 144

/api/translations

GET /api/translations/, 131

GET /api/translations/(string:project)/(string:component)/, 131

GET /api/translations/(string:project)/(string:component)/, 134

GET /api/translations/(string:project)/(string:component)/, 134

GET /api/translations/(string:project)/(string:component)/, 135

GET /api/translations/(string:project)/(string:component)/, 136

GET /api/translations/(string:project)/(string:component)/, 136

GET /api/translations/(string:project)/(string:component)/, 134

POST /api/translations/(string:project)/(string:component)/, 135

POST /api/translations/(string:project)/(string:component)/, 136

POST /api/translations/(string:project)/(string:component)/, 134

DELETE /api/translations/(string:project)/(string:component)/, 133

/api/units

GET /api/units/, 137

GET /api/units/(int:id)/, 137
 PUT /api/units/(int:id)/, 138
 DELETE /api/units/(int:id)/, 139
 PATCH /api/units/(int:id)/, 138

/api/users

GET /api/users/, 106
 GET /api/users/(str:username)/, 106
 GET /api/users/(str:username)/notifications/,
 108
 GET /api/users/(str:username)/notifications/(int:subscription_id)/,
 108
 GET /api/users/(str:username)/statistics/,
 107
 POST /api/users/, 106
 POST /api/users/(str:username)/groups/,
 107
 POST /api/users/(str:username)/notifications/,
 108
 PUT /api/users/(str:username)/, 107
 PUT /api/users/(str:username)/notifications/(int:subscription_id)/,
 108
 DELETE /api/users/(str:username)/, 107
 DELETE /api/users/(str:username)/notifications/(int:subscription_id)/,
 109
 PATCH /api/users/(str:username)/, 107
 PATCH /api/users/(str:username)/notifications/(int:subscription_id)/,
 108

/exports

GET /exports/rss/, 149
 GET /exports/rss/(string:project)/, 149
 GET /exports/rss/(string:project)/(string:component)/,
 149
 GET /exports/rss/(string:project)/(string:component)/(string:language)/,
 149
 GET /exports/rss/language/(string:language)/,
 149
 GET /exports/stats/(string:project)/(string:component)/,
 147

/hooks

GET /hooks/update/(string:project)/,
 145
 GET /hooks/update/(string:project)/(string:component)/,
 145
 POST /hooks/azure/, 146
 POST /hooks/bitbucket/, 146
 POST /hooks/gitea/, 147
 POST /hooks/gitee/, 147
 POST /hooks/github/, 145
 POST /hooks/gitlab/, 146
 POST /hooks/pagure/, 146

Semboller

.XML resource file
 file format, 88

--add
 auto_translate komut satırı seçeneği, 378

--addon
 install_addon komut satırı seçeneği, 385

--age
 commit_pending komut satırı seçeneği, 379

--author
 add_suggestions komut satırı seçeneği, 377

--author-email
 wlc komut satırı seçeneği, 152

--author-name
 wlc komut satırı seçeneği, 152

--base-file-template
 import_project komut satırı seçeneği, 382

--check
 importusers komut satırı seçeneği, 384

--config
 wlc komut satırı seçeneği, 150

--config-section
 wlc komut satırı seçeneği, 151

--configuration
 install_addon komut satırı seçeneği, 385

--convert
 wlc komut satırı seçeneği, 152

--email
 createadmin komut satırı seçeneği, 380

--file-format
 import_project komut satırı seçeneği, 383

--force
 loadpo komut satırı seçeneği, 386

--force-commit
 pushgit komut satırı seçeneği, 387

--format
 wlc komut satırı seçeneği, 150

--fuzzy
 wlc komut satırı seçeneği, 152

--ignore
 import_json komut satırı seçeneği, 381

--inconsistent
 auto_translate komut satırı seçeneği, 378

--input
 wlc komut satırı seçeneği, 152

--key
 wlc komut satırı seçeneği, 150

--lang
 loadpo komut satırı seçeneği, 386

--language-code
 list_translators komut satırı seçeneği, 385

--language-map
 import_memory komut satırı seçeneği, 382

--language-regex
 import_project komut satırı seçeneği, 383

--license
 import_project komut satırı seçeneği, 383

--license-url
 import_project komut satırı seçeneği, 383

--main-component
 import_json komut satırı seçeneği, 381

--method
 import_project komut satırı seçeneği, 383

--mode
 auto_translate komut satırı seçeneği, 378

--mt

auto_translate komut satırı seçeneği, 378
 --name
 createadmin komut satırı seçeneği, 380
 --name-template
 import_project komut satırı seçeneği, 382
 --new-base-template
 import_project komut satırı seçeneği, 383
 --no-password
 createadmin komut satırı seçeneği, 380
 --no-privs-update
 setupgroups komut satırı seçeneği, 387
 --no-projects-update
 setupgroups komut satırı seçeneği, 387
 --no-update
 setuplang komut satırı seçeneği, 388
 --output
 wlc komut satırı seçeneği, 152
 --overwrite
 auto_translate komut satırı seçeneği, 378
 wlc komut satırı seçeneği, 152
 --password
 createadmin komut satırı seçeneği, 380
 --project
 import_json komut satırı seçeneği, 381
 --source
 auto_translate komut satırı seçeneği, 378
 --threshold
 auto_translate komut satırı seçeneği, 378
 --update
 createadmin komut satırı seçeneği, 380
 import_json komut satırı seçeneği, 381
 install_addon komut satırı seçeneği, 385
 --url
 wlc komut satırı seçeneği, 150
 --user
 auto_translate komut satırı seçeneği, 378
 --username
 createadmin komut satırı seçeneği, 380
 --vcs
 import_project komut satırı seçeneği, 383

A

add_suggestions
 weblate admin command, 377
 add_suggestions komut satırı seçeneği
 --author, 377
 ADMINS
 setting, 204
 AKISMET_API_KEY
 setting, 336
 ALLOWED_HOSTS
 setting, 204
 Android
 file format, 83
 ANONYMOUS_USER_NAME
 setting, 336
 API, 102, 149, 154
 Apple strings
 file format, 83
 ARB
 file format, 87
 AUDITLOG_EXPIRY
 setting, 337
 AUTH_LOCK_ATTEMPTS
 setting, 337
 AUTH_TOKEN_VALID
 setting, 338
 auto_translate
 weblate admin command, 378
 auto_translate komut satırı seçeneği
 --add, 378
 --inconsistent, 378
 --mode, 378
 --mt, 378
 --overwrite, 378
 --source, 378
 --threshold, 378
 --user, 378
 AUTO_UPDATE
 setting, 337
 AUTOFIX_LIST
 setting, 338
 AVATAR_URL_PREFIX
 setting, 337

B

BACKGROUND_TASKS
 setting, 339
 BASE_DIR
 setting, 339
 BaseAddon (*weblate.addons.base* içindeki sınıf), 423
 BASIC_LANGUAGES
 setting, 339
 bilingual
 translation, 74
 BORG_EXTRA_ARGS
 setting, 339

C

`can_install()` (*weblate.addons.base.BaseAddon sınıf yöntemi*), 423

CELERY_BACKUP_OPTIONS, 162, 178

CELERY_BEAT_OPTIONS, 162, 178

CELERY_MAIN_OPTIONS, 162, 178

CELERY_MEMORY_OPTIONS, 162, 178

CELERY_NOTIFY_OPTIONS, 162, 178

celery_queues

- weblate admin command, 378

CELERY_TRANSLATE_OPTIONS, 162, 178

changes

- wlc komut satırı seçeneği, 152

CHECK_LIST

- setting, 340

checkgit

- weblate admin command, 379

cleanup

- wlc komut satırı seçeneği, 151

cleanup_ssh_keys

- weblate admin command, 380

cleanuptrans

- weblate admin command, 379

Comma separated values

- file format, 88

Command (*wlc.main içindeki sınıf*), 156

COMMENT_CLEANUP_DAYS

- setting, 341

commit

- wlc komut satırı seçeneği, 151

commit_pending

- weblate admin command, 379

commit_pending komut satırı seçeneği

- age, 379

COMMIT_PENDING_HOURS

- setting, 341

commitgit

- weblate admin command, 379

configure() (*weblate.addons.base.BaseAddon yöntemi*), 423

CONTACT_FORM

- setting, 341

createadmin

- weblate admin command, 380

createadmin komut satırı seçeneği

- email, 380
- name, 380
- no-password, 380
- password, 380
- update, 380
- username, 380

CSP_CONNECT_SRC

- setting, 340

CSP_FONT_SRC

- setting, 340

CSP_IMG_SRC

- setting, 340

CSP_SCRIPT_SRC

- setting, 340

CSP_STYLE_SRC

- setting, 340

CSV

- file format, 88

D

`daily()` (*weblate.addons.base.BaseAddon yöntemi*), 423

DATA_DIR

- setting, 341

DATABASE_BACKUP

- setting, 342

DATABASES

- setting, 204

DEBUG

- setting, 204

DEFAULT_ACCESS_CONTROL

- setting, 342

DEFAULT_ADD_MESSAGE

- setting, 343

DEFAULT_ADDON_MESSAGE

- setting, 343

DEFAULT_ADDONS

- setting, 343

DEFAULT_AUTO_WATCH

- setting, 342

DEFAULT_COMMIT_MESSAGE

- setting, 343

DEFAULT_COMMITER_EMAIL

- setting, 343

DEFAULT_COMMITER_NAME

- setting, 344

DEFAULT_DELETE_MESSAGE

- setting, 343

DEFAULT_FROM_EMAIL

- setting, 204

DEFAULT_LANGUAGE

- setting, 344

DEFAULT_MERGE_MESSAGE

- setting, 343

DEFAULT_MERGE_STYLE

- setting, 344

DEFAULT_PAGE_LIMIT

- setting, 352

DEFAULT_PULL_MESSAGE

- setting, 345

DEFAULT_RESTRICTED_COMPONENT

- setting, 343

DEFAULT_SHARED_TM

- setting, 344

DEFAULT_TRANSLATION_PROPAGATION

- setting, 344

download

- wlc komut satırı seçeneği, 152

DTD

- file format, 90

dump_memory

weblate admin command, 380
 dumpuserdata
 weblate admin command, 381

E

ENABLE_AVATARS
 setting, 345
 ENABLE_HOOKS
 setting, 345
 ENABLE_HTTPS
 setting, 345
 ENABLE_SHARING
 setting, 345

F

file format
 .XML resource file, 88
 Android, 83
 Apple strings, 83
 ARB, 87
 Comma separated values, 88
 CSV, 88
 DTD, 90
 gettext, 77
 go-i18n, 86
 GWT properties, 81
 i18next, 86
 INI translations, 81
 Java properties, 80
 Joomla translations, 82
 JSON, 84
 mi18n lang, 80
 PHP strings, 84
 PO, 77
 Qt, 82
 RC, 91
 ResourceDictionary, 88
 RESX, 88
 Ruby YAML, 90
 Ruby YAML Ain't Markup Language, 90
 string resources, 83
 TS, 82
 WPF, 88
 XLIFF, 78
 XML, 91
 YAML, 89
 YAML Ain't Markup Language, 89

G

get() (*wlc. Weblate yöntemi*), 155
 get_add_form() (*weblate.addons.base.BaseAddon sınıf yöntemi*), 423
 GET_HELP_URL
 setting, 345
 get_settings_form() (*weblate.addons.base.BaseAddon yöntemi*), 423
 gettext
 file format, 77

GITEA_CREDENTIALS
 setting, 346
 GITEA_TOKEN
 setting, 346
 GITEA_USERNAME
 setting, 346
 GITHUB_CREDENTIALS
 setting, 347
 GITHUB_TOKEN
 setting, 348
 GITHUB_USERNAME
 setting, 347
 GITLAB_CREDENTIALS
 setting, 346
 GITLAB_TOKEN
 setting, 347
 GITLAB_USERNAME
 setting, 347
 go-i18n
 file format, 86
 GOOGLE_ANALYTICS_ID
 setting, 348
 GWT properties
 file format, 81

H

HIDE_REPO_CREDENTIALS
 setting, 348
 HIDE_VERSION
 setting, 348

I

i18next
 file format, 86
 import_demo
 weblate admin command, 381
 import_json
 weblate admin command, 381
 import_json komut satırı seçeneği
 --ignore, 381
 --main-component, 381
 --project, 381
 --update, 381
 import_memory
 weblate admin command, 382
 import_memory komut satırı seçeneği
 --language-map, 382
 import_project
 weblate admin command, 382
 import_project komut satırı seçeneği
 --base-file-template, 382
 --file-format, 383
 --language-regex, 383
 --license, 383
 --license-url, 383
 --main-component, 383
 --name-template, 382
 --new-base-template, 383

- vcs, 383
- importuserdata
 - weblate admin command, 384
- importusers
 - weblate admin command, 384
- importusers komut satırı seçeneği
 - check, 384
- INI translations
 - file format, 81
- install_addon
 - weblate admin command, 385
- install_addon komut satırı seçeneği
 - addon, 385
 - configuration, 385
 - update, 385
- INTERLEDGER_PAYMENT_POINTERS
 - setting, 348
- iOS
 - translation, 83
- IP_BEHIND_REVERSE_PROXY
 - setting, 348
- IP_PROXY_HEADER
 - setting, 349
- IP_PROXY_OFFSET
 - setting, 349

J

- Java properties
 - file format, 80
- Joomla translations
 - file format, 82
- JSON
 - file format, 84

L

- LEGAL_URL
 - setting, 349
- LICENSE_EXTRA
 - setting, 350
- LICENSE_FILTER
 - setting, 350
- LICENSE_REQUIRED
 - setting, 350
- LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH
 - setting, 351
- list_languages
 - weblate admin command, 385
- list_translators
 - weblate admin command, 385
- list_translators komut satırı seçeneği
 - language-code, 385
- list_versions
 - weblate admin command, 386
- list-components
 - wlc komut satırı seçeneği, 151
- list-languages
 - wlc komut satırı seçeneği, 151

- list-projects
 - wlc komut satırı seçeneği, 151
- list-translations
 - wlc komut satırı seçeneği, 151
- load() (*wlc.config.WeblateConfig yöntemi*), 155
- loadpo
 - weblate admin command, 386
- loadpo komut satırı seçeneği
 - force, 386
 - lang, 386
- LOCALIZE_CDN_PATH
 - setting, 351
- LOCALIZE_CDN_URL
 - setting, 351
- lock
 - wlc komut satırı seçeneği, 151
- lock_translation
 - weblate admin command, 386
- lock-status
 - wlc komut satırı seçeneği, 151
- LOGIN_REQUIRED_URLS
 - setting, 351
- LOGIN_REQUIRED_URLS_EXCEPTIONS
 - setting, 351
- ls
 - wlc komut satırı seçeneği, 151

M

- main() (*wlc.main modülü içinde*), 156
- MATOMO_SITE_ID
 - setting, 352
- MATOMO_URL
 - setting, 352
- mi18n lang
 - file format, 80
- modülü
 - wlc, 154
 - wlc.config, 155
 - wlc.main, 156
- monolingual
 - translation, 74
- move_language
 - weblate admin command, 386

N

- NEARBY_MESSAGES
 - setting, 352

O

- ortam değişkeni
 - CELERY_BACKUP_OPTIONS, 162, 178
 - CELERY_BEAT_OPTIONS, 162, 178
 - CELERY_MAIN_OPTIONS, 162, 178
 - CELERY_MEMORY_OPTIONS, 162, 178
 - CELERY_NOTIFY_OPTIONS, 162, 178
 - CELERY_TRANSLATE_OPTIONS, 162, 178
 - POSTGRES_ALTER_ROLE, 173
 - POSTGRES_CONN_MAX_AGE, 174

POSTGRES_DATABASE, 173
 POSTGRES_DISABLE_SERVER_SIDE_CURSORS, 174
 POSTGRES_HOST, 173
 POSTGRES_PASSWORD, 173
 POSTGRES_PASSWORD_FILE, 173
 POSTGRES_PORT, 173
 POSTGRES_SSL_MODE, 173
 POSTGRES_USER, 173
 REDIS_DB, 174
 REDIS_HOST, 174
 REDIS_PASSWORD, 174, 175
 REDIS_PASSWORD_FILE, 175
 REDIS_PORT, 174
 REDIS_TLS, 175
 REDIS_VERIFY_SSL, 175
 ROLLBAR_ENVIRONMENT, 177
 ROLLBAR_KEY, 177
 SENTRY_DSN, 177
 SENTRY_ENVIRONMENT, 177
 SOCIAL_AUTH_SLACK_SECRET, 172
 WEB_WORKERS, 162, 178, 179
 WEBLATE_ADD_ADDONS, 178
 WEBLATE_ADD_APPS, 178
 WEBLATE_ADD_AUTOFIX, 178
 WEBLATE_ADD_CHECK, 178
 WEBLATE_ADD_LOGIN_REQUIRED_URLS_EXCEPTED, 166
 WEBLATE_ADMIN_EMAIL, 162, 163
 WEBLATE_ADMIN_NAME, 162, 163
 WEBLATE_ADMIN_PASSWORD, 158, 162--164
 WEBLATE_ADMIN_PASSWORD_FILE, 163
 WEBLATE_AKISMET_API_KEY, 167, 395
 WEBLATE_ALLOWED_HOSTS, 164, 204, 208, 209, 357
 WEBLATE_API_RATELIMIT_ANON, 105, 168
 WEBLATE_API_RATELIMIT_USER, 105, 168
 WEBLATE_AUTH_LDAP_BIND_DN, 169
 WEBLATE_AUTH_LDAP_BIND_PASSWORD, 169
 WEBLATE_AUTH_LDAP_BIND_PASSWORD_FILE, 169
 WEBLATE_AUTH_LDAP_CONNECTION_OPTION_REFUSAL, 169
 WEBLATE_AUTH_LDAP_SERVER_URI, 169
 WEBLATE_AUTH_LDAP_USER_ATTR_MAP, 169
 WEBLATE_AUTH_LDAP_USER_DN_TEMPLATE, 169
 WEBLATE_AUTH_LDAP_USER_SEARCH, 169
 WEBLATE_AUTH_LDAP_USER_SEARCH_FILTER, 169
 WEBLATE_AUTH_LDAP_USER_SEARCH_UNION, 169
 WEBLATE_AUTH_LDAP_USER_SEARCH_UNION_DELETE, 169
 WEBLATE_AUTO_UPDATE, 176
 WEBLATE_BASIC_LANGUAGES, 168
 WEBLATE_BORG_EXTRA_ARGS, 169
 WEBLATE_CONTACT_FORM, 164
 WEBLATE_CSP_CONNECT_SRC, 167
 WEBLATE_CSP_FONT_SRC, 167
 WEBLATE_CSP_IMG_SRC, 167
 WEBLATE_CSP_SCRIPT_SRC, 167
 WEBLATE_CSP_STYLE_SRC, 167
 WEBLATE_DATABASE_BACKUP, 174
 WEBLATE_DEBUG, 163
 WEBLATE_DEFAULT_ACCESS_CONTROL, 167
 WEBLATE_DEFAULT_AUTO_WATCH, 168
 WEBLATE_DEFAULT_COMMITER_EMAIL, 167
 WEBLATE_DEFAULT_COMMITER_NAME, 167
 WEBLATE_DEFAULT_FROM_EMAIL, 164
 WEBLATE_DEFAULT_PULL_MESSAGE, 167
 WEBLATE_DEFAULT_RESTRICTED_COMPONENT, 167
 WEBLATE_DEFAULT_SHARED_TM, 167
 WEBLATE_DEFAULT_TRANSLATION_PROPAGATION, 167
 WEBLATE_EMAIL_BACKEND, 176
 WEBLATE_EMAIL_HOST, 175
 WEBLATE_EMAIL_HOST_PASSWORD, 175, 176
 WEBLATE_EMAIL_HOST_PASSWORD_FILE, 176
 WEBLATE_EMAIL_HOST_USER, 175
 WEBLATE_EMAIL_PORT, 175, 176
 WEBLATE_EMAIL_USE_SSL, 175, 176
 WEBLATE_EMAIL_USE_TLS, 175, 176
 WEBLATE_ENABLE_AVATARS, 168
 WEBLATE_ENABLE_HOOKS, 168
 WEBLATE_ENABLE_HTTPS, 165, 237
 WEBLATE_GET_HELP_URL, 177
 WEBLATE_GITHUB_TOKEN, 166
 WEBLATE_GITHUB_USERNAME, 166
 WEBLATE_GITLAB_TOKEN, 166
 WEBLATE_GITLAB_USERNAME, 166
 WEBLATE_GOOGLE_ANALYTICS_ID, 166
 WEBLATE_GPG_IDENTITY, 167
 WEBLATE_HIDE_VERSION, 168
 WEBLATE_INTERLEDGER_PAYMENT_POINTERS, 165
 WEBLATE_IP_PROXY_HEADER, 165
 WEBLATE_LEGAL_URL, 177
 WEBLATE_LICENSE_FILTER, 168
 WEBLATE_LICENSE_REQUIRED, 168
 WEBLATE_LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH, 168
 WEBLATE_LOCALIZE_CDN_PATH, 177
 WEBLATE_LOCALIZE_CDN_URL, 177
 WEBLATE_LOGIN_REQUIRED_URLS_EXCEPTIONS, 166
 WEBLATE_LOGLEVEL, 163
 WEBLATE_LOGLEVEL_DATABASE, 163
 WEBLATE_NO_EMAIL_AUTH, 173
 WEBLATE_PAGURE_TOKEN, 166
 WEBLATE_PAGURE_USERNAME, 166
 WEBLATE_PRIVACY_URL, 177

WEBLATE_RATELIMIT_ATTEMPTS, 168, 397
 WEBLATE_RATELIMIT_LOCKOUT, 168
 WEBLATE_RATELIMIT_WINDOW, 168
 WEBLATE_REGISTRATION_ALLOW_BACKENDS, 164
 WEBLATE_REGISTRATION_OPEN, 164
 WEBLATE_REMOVE_ADDONS, 178
 WEBLATE_REMOVE_APPS, 178
 WEBLATE_REMOVE_AUTOFIX, 178
 WEBLATE_REMOVE_CHECK, 178
 WEBLATE_REMOVE_LOGIN_REQUIRED_URLS_EXCEPT, 166
 WEBLATE_REQUIRE_LOGIN, 166, 356
 WEBLATE_SAML_IDP_ENTITY_ID, 173
 WEBLATE_SAML_IDP_IMAGE, 173
 WEBLATE_SAML_IDP_TITLE, 173
 WEBLATE_SAML_IDP_URL, 173
 WEBLATE_SAML_IDP_X509CERT, 173
 WEBLATE_SECURE_PROXY_SSL_HEADER, 165
 WEBLATE_SERVER_EMAIL, 164
 WEBLATE_SERVICE, 162, 179
 WEBLATE_SILENCED_SYSTEM_CHECKS, 167, 234
 WEBLATE_SIMPLIFY_LANGUAGES, 167
 WEBLATE_SITE_DOMAIN, 163, 206, 225, 357
 WEBLATE_SITE_TITLE, 163
 WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_KEY, 171
 WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_SECRET, 171
 WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_KEY, 172
 WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_SECRET, 172
 WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_TENANT_ID, 172
 WEBLATE_SOCIAL_AUTH_BITBUCKET_KEY, 171
 WEBLATE_SOCIAL_AUTH_BITBUCKET_OAUTH2_KEY, 171
 WEBLATE_SOCIAL_AUTH_BITBUCKET_OAUTH2_SECRET, 171
 WEBLATE_SOCIAL_AUTH_BITBUCKET_SECRET, 171
 WEBLATE_SOCIAL_AUTH_FACEBOOK_KEY, 171
 WEBLATE_SOCIAL_AUTH_FACEBOOK_SECRET, 171
 WEBLATE_SOCIAL_AUTH_FEDORA, 172
 WEBLATE_SOCIAL_AUTH_GITHUB_KEY, 170
 WEBLATE_SOCIAL_AUTH_GITHUB_ORG_KEY, 170
 WEBLATE_SOCIAL_AUTH_GITHUB_ORG_NAME, 170
 WEBLATE_SOCIAL_AUTH_GITHUB_ORG_SECRET, 170
 WEBLATE_SOCIAL_AUTH_GITHUB_SECRET, 170
 WEBLATE_SOCIAL_AUTH_GITHUB_TEAM_ID, 170
 WEBLATE_SOCIAL_AUTH_GITHUB_TEAM_KEY, 170
 WEBLATE_SOCIAL_AUTH_GITHUB_TEAM_SECRET, 170
 WEBLATE_SOCIAL_AUTH_GITLAB_API_URL, 171
 WEBLATE_SOCIAL_AUTH_GITLAB_KEY, 171
 WEBLATE_SOCIAL_AUTH_GITLAB_SECRET, 171
 WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_KEY, 171
 WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET, 171
 WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_WHITELISTED, 171
 WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_WHITELISTED, 171
 WEBLATE_SOCIAL_AUTH_KEYCLOAK_ACCESS_TOKEN_URL, 172
 WEBLATE_SOCIAL_AUTH_KEYCLOAK_ALGORITHM, 172
 WEBLATE_SOCIAL_AUTH_KEYCLOAK_AUTHORIZATION_URL, 172
 WEBLATE_SOCIAL_AUTH_KEYCLOAK_IMAGE, 172
 WEBLATE_SOCIAL_AUTH_KEYCLOAK_KEY, 172
 WEBLATE_SOCIAL_AUTH_KEYCLOAK_PUBLIC_KEY, 172
 WEBLATE_SOCIAL_AUTH_KEYCLOAK_SECRET, 172
 WEBLATE_SOCIAL_AUTH_KEYCLOAK_TITLE, 172
 WEBLATE_SOCIAL_AUTH_OIDC_KEY, 172
 WEBLATE_SOCIAL_AUTH_OIDC_OIDC_ENDPOINT, 172
 WEBLATE_SOCIAL_AUTH_OIDC_SECRET, 172
 WEBLATE_SOCIAL_AUTH_OIDC_USERNAME_KEY, 172
 WEBLATE_SOCIAL_AUTH_OPENSUSE, 172
 WEBLATE_SOCIAL_AUTH_SLACK_KEY, 172
 WEBLATE_SOCIAL_AUTH_UBUNTU, 172
 WEBLATE_SSH_EXTRA_ARGS, 169
 WEBLATE_STATUS_URL, 177
 WEBLATE_TIME_ZONE, 164
 WEBLATE_URL_PREFIX, 167
 WEBLATE_WEBSITE_REQUIRED, 168
 WEBLATE_WORKERS, 162, 178, 179
 WL_BRANCH, 333
 WL_COMPONENT_NAME, 333
 WL_COMPONENT_SLUG, 333
 WL_COMPONENT_URL, 333
 WL_ENGAGE_URL, 333
 WL_FILE_FORMAT, 333

WL_FILEMASK, 333
 WL_LANGUAGE, 333
 WL_NEW_BASE, 333
 WL_PATH, 333
 WL_PREVIOUS_HEAD, 333
 WL_PROJECT_NAME, 333
 WL_PROJECT_SLUG, 333
 WL_REPO, 333
 WL_TEMPLATE, 333
 WL_VCS, 333

P

PAGURE_CREDENTIALS
 setting, 353
 PAGURE_TOKEN
 setting, 353
 PAGURE_USERNAME
 setting, 353
 PHP strings
 file format, 84
 PIWIK_SITE_ID
 setting, 352
 PIWIK_URL
 setting, 352
 PO
 file format, 77
 post() (*wlc. Weblate yöntemi*), 155
 post_add() (*weblate.addons.base.BaseAddon yöntemi*), 423
 post_commit() (*weblate.addons.base.BaseAddon yöntemi*), 423
 post_push() (*weblate.addons.base.BaseAddon yöntemi*), 423
 post_update() (*weblate.addons.base.BaseAddon yöntemi*), 423
 pre_commit() (*weblate.addons.base.BaseAddon yöntemi*), 424
 pre_push() (*weblate.addons.base.BaseAddon yöntemi*), 424
 pre_update() (*weblate.addons.base.BaseAddon yöntemi*), 424
 PRIVACY_URL
 setting, 353
 pull
 wlc komut satırı seçeneği, 151
 push
 wlc komut satırı seçeneği, 151
 pushgit
 weblate admin command, 387
 pushgit komut satırı seçeneği
 --force-commit, 387
 Python, 154
 Python Geliştirme Önerileri
 PEP 484, 420, 433

Q

Qt
 file format, 82

R

RATELIMIT_ATTEMPTS
 setting, 354
 RATELIMIT_LOCKOUT
 setting, 354
 RATELIMIT_WINDOW
 setting, 354
 RC
 file format, 91
 REDIS_PASSWORD, 175
 register_command() (*wlc.main modülü içinde*), 156
 REGISTRATION_ALLOW_BACKENDS
 setting, 354
 REGISTRATION_CAPTCHA
 setting, 354
 REGISTRATION_EMAIL_MATCH
 setting, 355
 REGISTRATION_OPEN
 setting, 355
 repo
 wlc komut satırı seçeneği, 151
 REPOSITORY_ALERT_THRESHOLD
 setting, 355
 REQUIRE_LOGIN
 setting, 355
 reset
 wlc komut satırı seçeneği, 151
 ResourceDictionary
 file format, 88
 REST, 102
 RESX
 file format, 88
 RFC
 RFC 5646, 74
 Ruby YAML
 file format, 90
 Ruby YAML Ain't Markup Language
 file format, 90

S

save_state() (*weblate.addons.base.BaseAddon yöntemi*), 424
 SECRET_KEY
 setting, 204
 SENTRY_DSN
 setting, 356
 SERVER_EMAIL
 setting, 205
 SESSION_COOKIE_AGE_AUTHENTICATED
 setting, 356
 SESSION_ENGINE
 setting, 204
 setting
 ADMINS, 204
 AKISMET_API_KEY, 336
 ALLOWED_HOSTS, 204
 ANONYMOUS_USER_NAME, 336

AUDITLOG_EXPIRY, 337
 AUTH_LOCK_ATTEMPTS, 337
 AUTH_TOKEN_VALID, 338
 AUTO_UPDATE, 337
 AUTOFIX_LIST, 338
 AVATAR_URL_PREFIX, 337
 BACKGROUND_TASKS, 339
 BASE_DIR, 339
 BASIC_LANGUAGES, 339
 BORG_EXTRA_ARGS, 339
 CHECK_LIST, 340
 COMMENT_CLEANUP_DAYS, 341
 COMMIT_PENDING_HOURS, 341
 CONTACT_FORM, 341
 CSP_CONNECT_SRC, 340
 CSP_FONT_SRC, 340
 CSP_IMG_SRC, 340
 CSP_SCRIPT_SRC, 340
 CSP_STYLE_SRC, 340
 DATA_DIR, 341
 DATABASE_BACKUP, 342
 DATABASES, 204
 DEBUG, 204
 DEFAULT_ACCESS_CONTROL, 342
 DEFAULT_ADD_MESSAGE, 343
 DEFAULT_ADDON_MESSAGE, 343
 DEFAULT_ADDONS, 343
 DEFAULT_AUTO_WATCH, 342
 DEFAULT_COMMIT_MESSAGE, 343
 DEFAULT_COMMITER_EMAIL, 343
 DEFAULT_COMMITER_NAME, 344
 DEFAULT_DELETE_MESSAGE, 343
 DEFAULT_FROM_EMAIL, 204
 DEFAULT_LANGUAGE, 344
 DEFAULT_MERGE_MESSAGE, 343
 DEFAULT_MERGE_STYLE, 344
 DEFAULT_PAGE_LIMIT, 352
 DEFAULT_PULL_MESSAGE, 345
 DEFAULT_RESTRICTED_COMPONENT, 343
 DEFAULT_SHARED_TM, 344
 DEFAULT_TRANSLATION_PROPAGATION, 344
 ENABLE_AVATARS, 345
 ENABLE_HOOKS, 345
 ENABLE_HTTPS, 345
 ENABLE_SHARING, 345
 GET_HELP_URL, 345
 GITEA_CREDENTIALS, 346
 GITEA_TOKEN, 346
 GITEA_USERNAME, 346
 GITHUB_CREDENTIALS, 347
 GITHUB_TOKEN, 348
 GITHUB_USERNAME, 347
 GITLAB_CREDENTIALS, 346
 GITLAB_TOKEN, 347
 GITLAB_USERNAME, 347
 GOOGLE_ANALYTICS_ID, 348
 HIDE_REPO_CREDENTIALS, 348
 HIDE_VERSION, 348
 INTERLEDGER_PAYMENT_POINTERS, 348
 IP_BEHIND_REVERSE_PROXY, 348
 IP_PROXY_HEADER, 349
 IP_PROXY_OFFSET, 349
 LEGAL_URL, 349
 LICENSE_EXTRA, 350
 LICENSE_FILTER, 350
 LICENSE_REQUIRED, 350
 LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH, 351
 LOCALIZE_CDN_PATH, 351
 LOCALIZE_CDN_URL, 351
 LOGIN_REQUIRED_URLS, 351
 LOGIN_REQUIRED_URLS_EXCEPTIONS, 351
 MATOMO_SITE_ID, 352
 MATOMO_URL, 352
 NEARBY_MESSAGES, 352
 PAGURE_CREDENTIALS, 353
 PAGURE_TOKEN, 353
 PAGURE_USERNAME, 353
 PIWIK_SITE_ID, 352
 PIWIK_URL, 352
 PRIVACY_URL, 353
 RATELIMIT_ATTEMPTS, 354
 RATELIMIT_LOCKOUT, 354
 RATELIMIT_WINDOW, 354
 REGISTRATION_ALLOW_BACKENDS, 354
 REGISTRATION_CAPTCHA, 354
 REGISTRATION_EMAIL_MATCH, 355
 REGISTRATION_OPEN, 355
 REPOSITORY_ALERT_THRESHOLD, 355
 REQUIRE_LOGIN, 355
 SECRET_KEY, 204
 SENTRY_DSN, 356
 SERVER_EMAIL, 205
 SESSION_COOKIE_AGE_AUTHENTICATED, 356
 SESSION_ENGINE, 204
 SIMPLIFY_LANGUAGES, 356
 SINGLE_PROJECT, 357
 SITE_DOMAIN, 356
 SITE_TITLE, 357
 SPECIAL_CHARS, 357
 SSH_EXTRA_ARGS, 357
 STATUS_URL, 357
 SUGGESTION_CLEANUP_DAYS, 358
 UPDATE_LANGUAGES, 358
 URL_PREFIX, 358
 VCS_BACKENDS, 358
 VCS_CLONE_DEPTH, 359
 WEBLATE_ADDONS, 359
 WEBLATE_EXPORTERS, 360
 WEBLATE_FORMATS, 360
 WEBLATE_GPG_IDENTITY, 360
 WEBLATE_MACHINERY, 360
 WEBSITE_REQUIRED, 360
 setupgroups

weblate admin command, 387

setupgroups komut satırı seçeneği

- no-privs-update, 387
- no-projects-update, 387

setuplang

- weblate admin command, 388

setuplang komut satırı seçeneği

- no-update, 388

show

- wlc komut satırı seçeneği, 151

SIMPLIFY_LANGUAGES

- setting, 356

SINGLE_PROJECT

- setting, 357

SITE_DOMAIN

- setting, 356

SITE_TITLE

- setting, 357

SPECIAL_CHARS

- setting, 357

SSH_EXTRA_ARGS

- setting, 357

statistics

- wlc komut satırı seçeneği, 151

STATUS_URL

- setting, 357

store_post_load()

- (*weblate.addons.base.BaseAddon yöntemi*), 424

string resources

- file format, 83

SUGGESTION_CLEANUP_DAYS

- setting, 358

T

translation

- bilingual, 74
- iOS, 83
- monolingual, 74

TS

- file format, 82

U

unit_pre_create()

- (*weblate.addons.base.BaseAddon yöntemi*), 424

unlock

- wlc komut satırı seçeneği, 152

unlock_translation

- weblate admin command, 387

UPDATE_LANGUAGES

- setting, 358

updatechecks

- weblate admin command, 388

updategit

- weblate admin command, 388

upload

- wlc komut satırı seçeneği, 152

URL_PREFIX

- setting, 358

V

VCS_BACKENDS

- setting, 358

VCS_CLONE_DEPTH

- setting, 359

version

- wlc komut satırı seçeneği, 151

W

WEB_WORKERS, 162, 178

Weblate (*wlc içindeki sınıf*), 155

weblate admin command

- add_suggestions, 377
- auto_translate, 378
- celery_queues, 378
- checkgit, 379
- cleanup_ssh_keys, 380
- cleanuptrans, 379
- commit_pending, 379
- commitgit, 379
- createadmin, 380
- dump_memory, 380
- dumpuserdata, 381
- import_demo, 381
- import_json, 381
- import_memory, 382
- import_project, 382
- importuserdata, 384
- importusers, 384
- install_addon, 385
- list_languages, 385
- list_translators, 385
- list_versions, 386
- loadpo, 386
- lock_translation, 386
- move_language, 386
- pushgit, 387
- setupgroups, 387
- setuplang, 388
- unlock_translation, 387
- updatechecks, 388
- updategit, 388

WEBLATE_ADDONS

- setting, 359

WEBLATE_ADMIN_EMAIL, 162, 163

WEBLATE_ADMIN_NAME, 162, 163

WEBLATE_ADMIN_PASSWORD, 158, 162--164

WEBLATE_ADMIN_PASSWORD_FILE, 163

WEBLATE_AKISMET_API_KEY, 395

WEBLATE_ALLOWED_HOSTS, 204, 208, 209, 357

WEBLATE_API_RATELIMIT_ANON, 105

WEBLATE_API_RATELIMIT_USER, 105

WEBLATE_AUTH_LDAP_BIND_PASSWORD, 169

WEBLATE_EMAIL_HOST_PASSWORD, 176

WEBLATE_EMAIL_PORT, 175, 176

WEBLATE_EMAIL_USE_SSL, 175, 176

WEBLATE_EMAIL_USE_TLS, 175, 176

WEBLATE_ENABLE_HTTPS, 237

WEBLATE_EXPORTERS
 setting, 360
WEBLATE_FORMATS
 setting, 360
WEBLATE_GPG_IDENTITY
 setting, 360
WEBLATE_LOCALIZE_CDN_PATH, 177
WEBLATE_MACHINERY
 setting, 360
WEBLATE_RATELIMIT_ATTEMPTS, 397
WEBLATE_REQUIRE_LOGIN, 356
WEBLATE_SECURE_PROXY_SSL_HEADER, 165
WEBLATE_SERVICE, 162
WEBLATE_SILENCED_SYSTEM_CHECKS, 234
WEBLATE_SITE_DOMAIN, 206, 225, 357
WEBLATE_WORKERS, 162, 178, 179
WeblateConfig (*wlc.config içindeki sınıf*), 155
WeblateException, 154
WEBSITE_REQUIRED
 setting, 360
wlc, 149
 modülü, 154
wlc komut satırı seçeneği
 --author-email, 152
 --author-name, 152
 --config, 150
 --config-section, 151
 --convert, 152
 --format, 150
 --fuzzy, 152
 --input, 152
 --key, 150
 --method, 152
 --output, 152
 --overwrite, 152
 --url, 150
 changes, 152
 cleanup, 151
 commit, 151
 download, 152
 list-components, 151
 list-languages, 151
 list-projects, 151
 list-translations, 151
 lock, 151
 lock-status, 151
 ls, 151
 pull, 151
 push, 151
 repo, 151
 reset, 151
 show, 151
 statistics, 151
 unlock, 152
 upload, 152
 version, 151
wlc.config
 modülü, 155

wlc.main
 modülü, 156
WPF
 file format, 88

X

XLIFF
 file format, 78
XML
 file format, 91

Y

YAML
 file format, 89
YAML Ain't Markup Language
 file format, 89