# Dokumentime Përdoruesi

1 Dokumentime Përdoruesi
1.1 Gjëra Elementare Webtranslate
1.2 Regjistrimi dhe Profil Përdoruesi
1.3 Përkthimi duke përdorur Webtranslate-in
1.4 Shkarkimi dhe Ngarkimi Përkkthimesh
1.5 Fjalorth
1.6 Checks and Fixups
1.7 Kërkimi
1.8 Rjedha Puni Përkkthimi
1.9 Pyetjet Bëra Rëndom
1.10 Formate të Mbuluar Kartelash
1.11 Integrim Kontrolli Versioni
1.12 Weblate’s REST API
1.13 Client Weblate
1.14 API Python e Webtranslate-it

2 Dokumentime Përgjegjisë
2.1 Udhëzime Formësimi
2.2 Sendësrtime Weblate
2.3 Upgrading Weblate
2.4 Kopjerujtje dhe Zhvendasje e Weblate
2.5 Mirefillsësim
2.6 Kontroll Hyrjesh
2.7 Projekte Përkkthimi
2.8 Language Definitions
2.9 Përkkthimi në Vazhdimeshi
2.10 Licencim Përkkthimesh
2.11 Procesi i Përkkthimit
2.12 Checks and Fixups
2.13 Përkkthimi nga Makina
2.14 Shtesa
2.15 Kujtesë Përkkthimesh
2.16 Formësim
2.17 Sample Configuration
2.18 Management Commands
2.19 Lajmërime
2.20 Lista Përbërësish
2.21 Optional Webtranslate Modules
2.22 Përshtatje e Webtranslate-it
2.23 Ndërfaqe Administrimi
2.24 Getting Support for Webtranslate
3 Dokumentime kontribuesi

3.1 Dhënie ndihmese të Weblate

3.2 Starting contributing code to Weblate

3.3 Kod burim i Weblate-it

3.4 Diagnostikim Weblate-it

3.5 Gjërat e brendshme të Weblate-it

3.6 Developing addons

3.7 Pjesa e dukshmë e Weblate-it

3.8 Njoftim problemesh në Weblate

3.9 Weblate testsuite and continuous integration

3.10 Skema të dhënahash

3.11 Releasing Weblate

3.12 Security and privacy

3.13 Rreth Weblate-it

3.14 Licencë

4 Historik Ndryshimesh

4.1 Weblate 4.9

4.2 Weblate 4.8.1

4.3 Weblate 4.8

4.4 Weblate 4.7.2

4.5 Weblate 4.7.1

4.6 Weblate 4.7

4.7 Weblate 4.6.2

4.8 Weblate 4.6.1

4.9 Weblate 4.6

4.10 Weblate 4.5.3

4.11 Weblate 4.5.2

4.12 Weblate 4.5.1

4.13 Weblate 4.5

4.14 Weblate 4.4.2

4.15 Weblate 4.4.1

4.16 Weblate 4.4

4.17 Weblate 4.3.2

4.18 Weblate 4.3.1

4.19 Weblate 4.3

4.20 Weblate 4.2.2

4.21 Weblate 4.2.1

4.22 Weblate 4.2

4.23 Weblate 4.1.1

4.24 Weblate 4.1

4.25 Weblate 4.0.4

4.26 Weblate 4.0.3

4.27 Weblate 4.0.2

4.28 Weblate 4.0.1

4.29 Weblate 4.0

4.30 Weblate 3.x series

4.31 Weblate 2.x series

4.32 Seritë Weblate 1.x

4.33 Weblate 0.x series

Tregues Modulesh Python

HTTP Routing Table

Tregues
1.1 Gjëra elementare Weblate

1.1.1 Project and component structure

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 Burime vargjesh Android). 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 Lejo përhapje përkthimesh in case the translations should diverge.

Shihni edhe:
../devel/integration

1.2 Regjistrim dhe profil përdoruesi

1.2.1 Regjistrim

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. Plotësoni formularin e regjistrimit me të dhënat tuaja.
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 Pult

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

Të reja në versionin 2.5.

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

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

Menuja ka këto mundësi:

- 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 *Lista Përbërësish*.

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

---

**Shënim:** When Weblate is configured for a single project using *SINGLE_PROJECT* in the *settings.py* file (see *Formësim*), the dashboard will not be shown, as the user will be redirected to a single project or component instead.

---

### 1.2.3 Profil përdoruesi

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.

**Shënim:** All language selections only offer currently translated languages.

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

---

**Gjuhë**

### 1.2.4 Interface language

Choose the language you want to display the UI in.
Gjuhë të përkthyera

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.
Gjuhë dytësore

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:

1.2. Regjistrim dhe profil përdoruesi
1.2.5 Parapëlqime

Pamje parazgjedhje e pultit

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.

Shihni edhe:

Lista Përbërešísh

Lidhje përpunuesi

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

Ndihmë: 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.

Shihni edhe:

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

1.2.6 Njoftime

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.

Shënim: S’dotë merrri njoftime për veprimet tuaja.
1.2. Regjistrim dhe profil përdoruesi
1.2.7 Llogari

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 *Eksportim të dhënashe përduresë në Weblate*).

**Shënim:** 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.
1.2. Regjistrimi dhe profil përdoruesi
1.2.8 Profil

Krejt fushat në këtë faqe janë opsionale dhe mund të fshihen kur të doni, dhe duke i plotësuar, na japni pranimin për t’i ndarë këto të dhëna me të tjerë kudo ku shfaqet profili juaj i përdoruesit.

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

1.2.9 Licenca

1.2.10 Përdorim API

You can get or reset your API access token here.

1.2.11 Auditim regjistr

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.

Shihni edhe:

Running behind reverse proxy

1.3 Përkthim duke përdorur Weblate-in

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.

Përgjithësisht, ka dy mënra përkthimi:

- Projekti pranon përkthime të drejtpërdrejtë
- The project only accepts suggestions, which are automatically validated once a defined number of votes is reached

Please see Rrjedha pune përkthimi for more info on translation workflow.

Mundësi për dukshmëri projekti përkthimi:

- Publicly visible and anybody can contribute
- Visible only to a certain group of translators

Shihni edhe:

Kontroll hyrjesh, Rrjedha pune përkthimi

1.3.1 Projekte përkthimi

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

Having navigated to a component, a set of links lead to its actual translation. The translation is further divided into individual checks, like Not translated strings or Strings needing action. 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.
1.3.3 Sugjerime

**Shënimi:** 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 Komente

Three types of comments can be posted: for translations, source strings, or to report source string bugs when this functionality is turned on using *Aktivizo shqyrtime burimi*. 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.

You can use Markdown syntax in all comments and mention other users using `@mention`.

Shihni edhe:

report-source, *Shqyrtim vargjesh burim, Aktivizo shqyrtime burimi*

1.3.5 Variante

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.

Shihni edhe:

variants, *Variante*

1.3.6 Etiketa

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

Shihni edhe:

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 *Gjuhë dytësore*) above the source string.

Below the translation, translators will find suggestion made by others, to be accepted (√), accepted with changes (✏), or deleted (🗑).
Shihni edhe:

Formulë shumësi

Kapitulli 1. Dokumentime përdoruesi
Shkurtore tastiere

Ndryshuar në versionin 2.18: The keyboard shortcuts have been revamped in 2.18 to less likely collide with browser or system defaults.

Gjatë përkhimit mund të përdoren shkurtoret vijuese të tastierës:

<table>
<thead>
<tr>
<th>Keyboard shortcut</th>
<th>Përshkrim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alt+Home</td>
<td>Navigate to first translation in current search.</td>
</tr>
<tr>
<td>Alt+End</td>
<td>Navigate to last translation in current search.</td>
</tr>
<tr>
<td>Alt+PageUp or Ctrl ↑ or Alt ↑ or Cmd ↑</td>
<td>Navigate to previous translation in current search.</td>
</tr>
<tr>
<td>Ctrl+PageDown or Ctrl+↓ or Alt+↓ or Cmd+↓</td>
<td>Navigate to next translation in current search.</td>
</tr>
<tr>
<td>Alt+Enter or Ctrl+Enter or Cmd+Enter</td>
<td>Ruaje përkhimin e tanishëm.</td>
</tr>
<tr>
<td>Ctrl+Shift+Enter or Cmd+Shift+Enter</td>
<td>Unmark translation as needing edit and submit it.</td>
</tr>
<tr>
<td>Ctrl+E or Cmd+E</td>
<td>Fokuso përpunues përkhimesh.</td>
</tr>
<tr>
<td>Ctrl+U or Cmd+U</td>
<td>Fokuso përpunues komentesh.</td>
</tr>
<tr>
<td>Ctrl+M or Cmd+M</td>
<td>Shows Automatic suggestions tab, see Sugjerime të automatizuara.</td>
</tr>
<tr>
<td>Ctrl+1 to Ctrl+9 or Cmd+1 to Cmd+9</td>
<td>Copies placeable of given number from source string.</td>
</tr>
<tr>
<td>Ctrl+M+1 to 9 or Cmd+M+1 to 9</td>
<td>Copy the machine translation of given number to current translation.</td>
</tr>
<tr>
<td>Ctrl+I+1 to 9 or Cmd+I+1 to 9</td>
<td>Ignore one item in the list of failing checks.</td>
</tr>
<tr>
<td>Ctrl+J or Cmd+J</td>
<td>Shows the Nearby strings tab.</td>
</tr>
<tr>
<td>Ctrl+S or Cmd+S</td>
<td>Focus search field.</td>
</tr>
<tr>
<td>Ctrl+O or Cmd+O</td>
<td>Copy source string.</td>
</tr>
<tr>
<td>Ctrl+Y or Cmd+Y</td>
<td>Toggle the Needs editing flag.</td>
</tr>
</tbody>
</table>

Tastierë pamore

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 characters defined in the Profil përdoruesi
- Per-language characters provided by Weblate (e.g. quotes or RTL specific characters)
- Characters configured using SPECIAL_CHARS
Kontekstpërkthimi

This contextual description provides related info about the current string.

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

Foto ekrani Screenshots can be uploaded to Weblate to better inform translators of where and how the string is used, see Visual context for strings.

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

Hasje të tjera 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 Jo i njëtruajshëm). You can choose which one to use.
Kujtesë përkkthimesh  Look at similar strings translated in past, see Kujtesë Përkkthimesh.

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

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

Projekt  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.

Historik përkkthimi

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.

Gjatësievargutë përkkthyer

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 \texttt{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 \texttt{Formate dygjuhësh dhe njëgjuhësh} for more info.
- Maximal length in characters defined by translation file or flag, see Gjatësi maksimum e përkkthimit.
- Maximal rendered size in pixels defined by flags, see Madhësi maksimum e përkkthimit.

1.3.8 Sugjerime të automatizuara

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

Shihni edhe:  You can find the list of supported tools in Përkkthim nga makina.

1.3.9 Përkkthim i automatizuar

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:
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.

**Sinjalizim:** 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).

**Shihni edhe:**

*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 Kërkonidhezëvendësoni

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

Ndihmëz: Don’t worry about messing up the strings. This is a two-step process showing a preview of edited strings before the actual change is confirmed.

1.3.12 Përpunim në masë

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)

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

Shihni edhe:

Bulk edit addon

1.4 Shkarkim dhe ngarkim përthimesh

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.

Shënim: Available options might be limited by access control settings.

1.4.1 Shkarkim përthimesh

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:

• PO gettext
• XLIFF me zgjerime gettext
• XLIFF 1.1
• TermBase eXchange
• Translation Memory eXchange
• gettext MO (only available when translation is using gettext PO)
• CSV
• Excel Open XML
• JSON (only available for monolingual translations)
• Android String Resource (only available for monolingual translations)
• iOS strings (only available for monolingual translations)

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

Shihni edhe:

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

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

### Formate të mbuluar kartelash

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.

**Shihni edhe:**

*Formate të mbuluar kartelash*

The uploaded file is merged to update the translation, overwriting existing entries by default (this can be turned off or on in the upload dialog).
Import methods

These are the choices presented when uploading translation files:

**Shtoje si përkthim (translate)** Imported translations are added as translations. This is the most common usecase, and the default behavior.

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

**Shtoje si sugjerim (suggest)** Imported translations 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.

**Shtoje si përkthim që lyp përpunim (fuzzy)** Imported translations 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.

**Zëvendëso kartelë ekzistuese përkthimi (replace)** Existing file is replaced with new content. This can lead to loss of existing translations, use with caution.

**Përditëso vargje burim (source)** Updates source strings in bilingual translation file. This is similar to what *Përditësoni kartela PO, që të përputhen me POT (msgmerge)* does.

This option is supported only for some file formats.

**Add new strings (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 *Administroni vargje* turned on.

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

Shihni edhe:

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

Trajitim përplasje

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

Vargje që lypin përpunim

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 Fjalorth

Each project can include one or more glossaries as a shorthand for storing terminology. Glossary easiy 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 Administrim fjalorthësh

Ndryshuar në versionin 4.5: 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 Përdore si fjalorth. 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 Ndaje te projekte from the respective glossary
component.

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

You can browse all glossary terms:
or edit them as any translations.

1.5.2 Glossary terms

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

Untranslatable terms

Të reja në versionin 4.5.

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.

Shihni edhe:

Customizing behavior using flags
Forbidden translations

Të reja në versionin 4.5.

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.

Shihni edhe:
Customizing behavior using flags

Terminologji

Të reja në versionin 4.5.

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.

Shihni edhe:
Customizing behavior using flags

Variante

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

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

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

<table>
<thead>
<tr>
<th>Singular</th>
<th>Plural</th>
</tr>
</thead>
<tbody>
<tr>
<td>%(\text{count})'s word</td>
<td>%(\text{count})'s words</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Czech, One</th>
</tr>
</thead>
<tbody>
<tr>
<td>práve slov</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Czech, Few</th>
</tr>
</thead>
<tbody>
<tr>
<td>několik slov</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Czech, Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>%(\text{count})'s slov</td>
</tr>
</tbody>
</table>

**Plural formula**: \(n=1\)? 0; \((n=2 \& n\neq 4)\)? 1: 2

- Needs editing

**Glossary**

<table>
<thead>
<tr>
<th>English</th>
<th>Czech</th>
</tr>
</thead>
<tbody>
<tr>
<td>No related strings found in the glossary.</td>
<td></td>
</tr>
</tbody>
</table>

**String information**

- No screenshot currently associated.
- No explanation currently provided.
- No labels currently set.
- python-format
- Source string location: [webate/templates/translation.html:110]
- String age: 20 seconds ago
- Source string age: 21 seconds ago
- Translation file: [webate/locale/cs/LC_MESSAGE5/5djangos PANEL]
1.6.1 Automatic fixups

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

Shihni edhe: AUTOFIX_LIST

1.6.2 Kontrole cilësie

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.

Shihni edhe: CHECK_LIST, Customizing behavior using flags

1.6.3 Kontrole përkthimi

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

Markup BBCode

**Summary** BBcode te përkthimi s’përputhet me burimin

**Fokus** translated strings

**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.

Shënim: The method for detecting BBcode is currently quite simple so this check might produce false positives.

Fjalë të njëpasnjëshme të përsëdytura

Të reja në versionin 4.1.

**Summary** Teksti përmban të njëjtën fjalë dy herë, njëra pas tjetrës:

**Fokus** translated strings

**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.

Ndihmëz: This check includes language specific rules to avoid false positives. In case it triggers falsely in your case, let us know. See Njofëm problemesh në Weblate.
S'ndjek fjalorthin
Të reja në versionin 4.5.

Summary  Përkthimis'ndjek zërat e përcaktuar në fjalorth.
Fokus translated strings
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.

Shihni edhe:
Fjalorth, Customizing behavior using flags, Flamuj përkthimi

Hapësirë dyshe

Summary  Përkthimi përmban hapësira dyshe
Fokus translated strings
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.

Vargje të formatuar

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.

Ndihmëz: In case specific format check is not available in Weblate, you can use generic ` Vendmbajtëse.`

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

**Summary**  Vargu i ndërshtënies AngulaJS s’përputhet me burimin

**Fokus**  translated strings

**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 }}`

Shihni edhe:

*Vargje të formatuar*, AngularJS text interpolation

---

1.6. Checks and fixups 29
Format C

**Summary**  Vargu në format C nuk përputhet me burimin

**Fokus**  translated strings

**Check class**  weblate.checks.format.CFormatCheck

**Flag to enable**  c-format

**Flag to ignore**  ignore-c-format

*Simple format string example*  Ka %d mollë

*Position format string example*  Your balance is %1$d %2$s

Shihni edhe:

**Vargje të formatuar**,  C format strings, C printf format

Format C#

**Summary**  Vargu në format C# nuk përputhet me burimin

**Fokus**  translated strings

**Check class**  weblate.checks.format.CSharpFormatCheck

**Flag to enable**  c-sharp-format

**Flag to ignore**  ignore-c-sharp-format

*Position format string example*  Ka {0} mollë

Shihni edhe:

**Vargje të formatuar**,  C# String Format

ECMAScript template literals

**Summary**  ECMAScript template literals do not match source

**Fokus**  translated strings

**Check class**  weblate.checks.format.ESTemplateLiteralsCheck

**Flag to enable**  es-format

**Flag to ignore**  ignore-es-format

*Interpolation example*  Ka ${number} mollë

Shihni edhe:

**Vargje të formatuar**,  Template literals
Ndërshtënie i18next

Të reja në versionin 4.0.

**Summary**  Ndërshtënia i18next nuk përputhen me burimin

**Fokus**  translated strings

**Check class**  `weblate.checks.format.I18NextInterpolationCheck`

**Flag to enable**  `i18next-interpolation`

**Flag to ignore**  `ignore-i18next-interpolation`

**Interpolation example**  Ka `{number}` mollë

**Nesting example**  Ka `$(number)` mollë

Shihni edhe:

**Vargje të formatuar, i18next interpolation**

### Format Java

**Summary**  Vargu në format Java nuk përputhet me burimin

**Fokus**  translated strings

**Check class**  `weblate.checks.format.JavaFormatCheck`

**Flag to enable**  `java-format`

**Flag to ignore**  `ignore-java-format`

**Simple format string example**  Ka `%d` mollë

**Position format string example**  Your balance is `%1$d %2$s`

Shihni edhe:

**Vargje të formatuar, Java Format Strings**

### Java MessageFormat

**Summary**  Vargu në format Java MessageFormat nuk përputhet me burimin

**Fokus**  translated strings

**Check class**  `weblate.checks.format.JavaMessageFormatCheck`

**Flag to enable uncoditionally**  `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**  Ka `{0}` mollë

Shihni edhe:

**Vargje të formatuar, Java MessageFormat**

1.6. Checks and fixups
**Format JavaScript**

**Summary** Vargu në format JavaScript nuk përputhet me burimin

**Fokus** translated strings

**Check class** `weblate.checks.format.JavaScriptFormatCheck`

**Flag to enable** `javascript-format`

**Flag to ignore** `ignore-javascript-format`

**Simple format string example** Ka %d mollë

Shihni edhe:

*Vargje të formatuar, JavaScript formatting strings*

**Format Lua**

**Summary** Vargu në format Lua s’përputhet me burimin

**Fokus** translated strings

**Check class** `weblate.checks.format.LuaFormatCheck`

**Flag to enable** `lua-format`

**Flag to ignore** `ignore-lua-format`

**Simple format string example** Ka %d mollë

Shihni edhe:

*Vargje të formatuar, Lua formatting strings*

**Object Pascal format**

**Summary** Vargu në format objekti Pascal nuk përputhet me burimin

**Fokus** translated strings

**Check class** `weblate.checks.format.ObjectPascalFormatCheck`

**Flag to enable** `object-pascal-format`

**Flag to ignore** `ignore-object-pascal-format`

**Simple format string example** Ka %d mollë

Shihni edhe:

*Vargje të formatuar, Object Pascal formatting strings Free Pascal formatting strings Delphi formatting strings*

**Vendmbajtëse përqindje**

Të reja në versionin 4.0.

**Summary** Vendmbajtëset përqindje s’përputhen me burimin

**Fokus** translated strings

**Check class** `weblate.checks.format.PercentPlaceholdersCheck`

**Flag to enable** `percent-placeholders`

**Flag to ignore** `ignore-percent-placeholders`
Simple format string example  Ka %number% mollë

Shihni edhe:
Vargje të formatuar,

Format Perl

Summary  Vargu në format Perl nuk përuphet me burimin
Fokus  translated strings
Check class  weblate.checks.format.PerlFormatCheck
Flag to enable  perl-format
Flag to ignore  ignore-perl-format
Simple format string example  Ka %d mollë
Position format string example  Your balance is %d %s

Shihni edhe:
Vargje të formatuar, Perl sprintf, Perl Format Strings

Format PHP

Summary  Vargu në format PHP nuk përuphet me burimin
Fokus  translated strings
Check class  weblate.checks.format.PHPFormatCheck
Flag to enable  php-format
Flag to ignore  ignore-php-format
Simple format string example  Ka %d mollë
Position format string example  Your balance is %d %s

Shihni edhe:
Vargje të formatuar, PHP sprintf documentation, PHP Format Strings

Format kllapash Python

Summary  Vargu në format kllapash Python nuk përuphet me burimin
Fokus  translated strings
Check class  weblate.checks.format.PythonBraceFormatCheck
Flag to enable  python-brace-format
Flag to ignore  ignore-python-brace-format
Simple format string  ""Ka {} mollë
Named format string example  Your balance is {amount} {currency}

Shihni edhe:
Vargje të formatuar, Python brace format, Python Format Strings
Format Python

Summary  Vargu në format Python nuk përputhet me burimin
Fokus  translated strings
Check class  weblate.checks.format.PythonFormatCheck
Flag to enable  python-format
Flag to ignore  ignore-python-format
Simple format string  Ka %d mollë
Named format string example  Your balance is %amount$d %currency$s

Shihni edhe:
Vargje të formatuar, Python string formatting, Python Format Strings

Format Qt

Summary  Vargu në format Qt nuk përputhet me burimin
Fokus  translated strings
Check class  weblate.checks.qt.QtFormatCheck
Flag to enable  qt-format
Flag to ignore  ignore-qt-format
Position format string example  Ka %1 mollë

Shihni edhe:
Vargje të formatuar, Qt QString::arg

Format shumësi Qt

Summary  Vargu në format shumësi Qt nuk përputhet me burimin
Fokus  translated strings
Check class  weblate.checks.qt.QtPluralCheck
Flag to enable  qt-plural-format
Flag to ignore  ignore-qt-plural-format
Plural format string example  Ka %Ln kumbull(a)

Shihni edhe:
Vargje të formatuar, Qt i18n guide
**Format Ruby**

**Summary**  Varg në format Ruby nuk përputhet me burimin

**Fokus**  translated strings

**Check class**  weblate.checks.ruby.RubyFormatCheck

**Flag to enable**  ruby-format

**Flag to ignore**  ignore-ruby-format

**Simple format string example**  Ka %d mollë

**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}

Shihni edhe:

*Vargje të formatuar, Ruby Kernel#sprintf*

---

**Format Schema**

**Summary**  Vargu në format Schema nuk përputhet me burimin

**Fokus**  translated strings

**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

Shihni edhe:

*Vargje të formatuar, Srfi 28, Chicken Scheme format, Guile Scheme formatted output*

---

**Formatimi i18n Vue**

**Summary**  Formatimi i18n Vue nuk përputhet me burimin

**Fokus**  translated strings

**Check class**  weblate.checks.format.VueFormattingCheck

**Flag to enable**  vue-format

**Flag to ignore**  ignore-vue-format

**Named formatting**  Ka {count} mollë

**Formatimi i18n Rails**  Ka %{count} mollë

**Linked locale messages**  @:message.dio @:message.the_world!

Shihni edhe:

*Vargje të formatuar, Vue I18n Formatting, Vue I18n Linked locale messages*
**Éshtë përkthyer**

**Summary** Ky varg éshtë përkthyer në të kaluarën

**Fokus** all strings

**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.

**Jo i njëtrajtshëm**

**Summary** Ky varg ka më shumë se një përkthim në këtë projekt ose s'éshët i përkthyer në disa për tërës.

**Fokus** all strings

**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.

---

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

---

**Shënim:** 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 not translated in some components just by clicking on the Use this translation button displayed on each line in the Other occurrences tab.

You can use *Përkthim i automatizuar* addon to automate translating of newly added strings which are already translated in another component.

---

**Shihni edhe:**

*Keeping translations same across components*

**Éshtë përdorur shkronjë Kashida**

Të reja në versionin 3.5.

**Summary** S’duhen përdorur shkronjë zbukuruese kashida

**Fokus** translated strings

**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.

**Shihni edhe:**

Kashida on Wikipedia
**Lidhje Markdown**

Të reja në versionin 3.5.

**Summary** Lidhje Markdown nuk përputhen me burimin

**Fokus** translated strings

**Check class** `weblate.checks.markup.MarkdownLinkCheck`

**Flag to enable** `md-text`

**Flag to ignore** `ignore-md-link`

Lidhje Markdown s’përputhen me burimin.

Shihni edhe:

Markdown links

**Referencë Markdown**

Të reja në versionin 3.5.

**Summary** Referencë lidhjesh Markdown nuk përputhen me burimin

**Fokus** translated strings

**Check class** `weblate.checks.markup.MarkdownRefLinkCheck`

**Flag to enable** `md-text`

**Flag to ignore** `ignore-md-reflink`

Referencë lidhjesh Markdown s’përputhen me burimin.

Shihni edhe:

Markdown links

**Sintaksë Markdown**

Të reja në versionin 3.5.

**Summary** Sintaksë Markdown nuk përputhet me burimin

**Fokus** translated strings

**Check class** `weblate.checks.markup.MarkdownSyntaxCheck`

**Flag to enable** `md-text`

**Flag to ignore** `ignore-md-syntax`

Sintaksë Markdown nuk përputhet me burimin

Shihni edhe:

Markdown span elements
The Webate Manual, Hedhje Në Qarkullim 4.9

Gjatësi maksimum e përkthimit

**Summary**  Përkthimi s’duhet të tejkalojë gjatësinë e dhënë

**Fokus**  translated strings

**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.

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

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

Madhësi maksimum e përkthimit

**Summary**  Teksti i shfaqur për përkthimin s’duhet të kalojë madhësinë e dhënë

**Fokus**  translated strings

**Check class**  weblate.checks.render.MaxSizeCheck

**Flag to enable**  max-size

**Flag to ignore**  ignore-max-size

Të reja në versionin 3.7.

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:

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

Ndihmëz: 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.

Shihni edhe:

*Administrim shkronjash, Customizing behavior using flags, Gjatësi maksimum e përkthimit*
The Weblate Manual, Hedhje Në Qarkullim 4.9

Mospërputhje n

**Summary**  Number of `\n` in translation does not match source

**Fokus**  translated strings

**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.

Mospërputhje dy pikash

**Summary**  Origjinali dhe përrikthimi nuk mbarojnë që të dy me dy pika

**Fokus**  translated strings

**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).

**Shihni edhe:**

Colon on Wikipedia

Mospërputhje tre pikash

**Summary**  Origjinali dhe përrikthimi nuk mbarojnë që të dy me shumëpikësh

**Fokus**  translated strings

**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.

**Shihni edhe:**

Ellipsis on Wikipedia

Mospërputhje pikëçuditjesh

**Summary**  Origjinali dhe përrikthimi nuk mbarojnë që të dy me pikëçuditje

**Fokus**  translated strings

**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).

**Shihni edhe:**

Exclamation mark on Wikipedia
**Mospërputhje pikë**

**Summary**  Origjinali dhe përkthimi nuk mbarojnë që të dy me pikë

**Fokus**  translated strings

**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).

**Shihni edhe:**

Full stop on Wikipedia

---

**Mospërputhje pikëpyetjesh**

**Summary**  Origjinali dhe përkthimi nuk mbarojnë që të dy me pikëpyetjë

**Fokus**  translated strings

**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).

**Shihni edhe:**

Question mark on Wikipedia

---

**Mospërputhje pikëpresjesh**

**Summary**  Origjinali dhe përkthimi nuk mbarojnë që të dy me pikëpresje

**Fokus**  translated strings

**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. This can be useful to keep formatting of entries such as desktop files.

**Shihni edhe:**

Semicolon on Wikipedia

---

**Mospërputhje ndërprerjeshe rreshtë**

**Summary**  Numri i shenjave për rresht të ri te përkthimi nuk përputhet me atë të burimit

**Fokus**  translated strings

**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.
**Mungon shumës**

**Summary**  S’janë përkthyther disa forma shumësi

**Fokus**  translated strings

**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.

**Vendmbajtëse**

Të reja në versionin 3.9.

**Summary**  Përktimit i mungojnë disa vendmbajtëse

**Fokus**  translated strings

**Check class**  `weblate.checks.placeholders.PlaceholderCheck`

**Flag to enable**  `placeholders`

**Flag to ignore**  `ignore-placeholders`

Ndryshuar në versionin 4.3: Mund të përdorni shprehje të rregullt si vendmbajtëse.

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:

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

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

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

Shihni edhe:

*Customizing behavior using flags*

**Hapësirë pikësimi**

Të reja në versionin 3.9.

**Summary**  Mungon shenjë hapësire të pandërprerë përpara shenjë të dyfishtë pikësimi

**Fokus**  translated strings

**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.

Shihni edhe:

*French and English spacing on Wikipedia*
Shprehje e rregullt

Të reja në versionin 3.9.

**Summary** Përkthimi s’përputhet me shprehje të rregullt:

**Fokus** translated strings

**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$
```

Shumës i njëjtë

**Summary** Disa forma shumës janë përkteher në të njëjtën mënyrë

**Fokus** translated strings

**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.

Shenjë rresht i ri në fillim

**Summary** Origjinali dhe përkteheri nuk fillojnë që të dy me simbol rreshti të ri

**Fokus** translated strings

**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.

Shihni edhe:

*Shenjë rresht i ri në fund*

Hapësira fillimi

**Summary** Burimi dhe përkteheri nuk fillojnë që të dy me të njëjtën numër hapësirash

**Fokus** translated strings

**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.
Shenjë rrresht i ri në fund

**Summary**  Origjinali dhe përkrthimi nuk mbarojnë që të dy me simbol rrresht të ri

**Fokus**  translated strings

**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.

Shihni edhe:

_Shenjë rrresht i ri në fillim_

Hapësirë në fund

**Summary**  Origjinali dhe përkrthimi nuk mbarojnë që të dy me hapësirë

**Fokus**  translated strings

**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.

Përkrthim i pandryshuar

**Summary**  Burimi dhe përkrthimi janë identikë

**Fokus**  translated strings

**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.

Shihni edhe:

_Component configuration, Customizing behavior using flags_
**HTML jo e parrezik**

Të reja në versionin 3.9.

*Summary*  Përkthimi përdor elementë HTML jo të parrezik

*Fokus*  translated strings

*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.

**Shihni edhe:**

The HTML check is performed by the Bleach library developed by Mozilla.

**URL**

Të reja në versionin 3.5.

*Summary*  Përkthimi s’përmban URL

*Fokus*  translated strings

*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.

**Markup XML**

*Summary*  Etiketa XML te përkthimi nuk përputhen me burimin

*Fokus*  translated strings

*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.

**Shënim:**  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.
Sintaksë XML

Të reja në versionin 2.8.

**Summary** Përkrimiti sështë XML e vlefsme

**Fokus** translated strings

**Check class** weblate.checks.markup.XMLValidityCheck

**Flag to ignore** ignore-xml-invalid

Ka elementë të pavlefshtëm shkrimi XML.

---

**Shënim:** 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.

---

Hapësirë me gjërësi zero

**Summary** Përkrimiti përmban shënje ekstra hapësire me gjërësi zero

**Fokus** translated strings

**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.

**Shihni edhe:**

Zero width space on Wikipedia

---

1.6.4 Kontrolle burimi

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

**Tre pika**

**Summary** Vargu përdor tre pika (...) në vend të shenjës shumëpikësh (…)

**Fokus** Vargje të burimit

**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.

**Shihni edhe:**

Ellipsis on Wikipedia
I papërkrthyer prej kohësh

Të reja në versionin 4.1.

**Summary** Ky varg sëshëtë përkthyer prej kohësh

**Fokus** Vargje të burimit

**Check class** `weblate.checks.source.LongUntranslatedCheck`

**Flag to ignore** `ignore-long-untranslated`

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

Dështime të shumta kontrolesh

**Summary** Përkthimet në disa nga gjërët kanë dështime kontrolesh

**Fokus** Vargje të burimit

**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.

Ndryshore të shumta të paemërtuara

Të reja në versionin 4.1.

**Summary** Te vargu ka ndryshore të shumta të paemërtuara, duke e bërë të pamundur renditjen e tyre nga përkkhyesit

**Fokus** Vargje të burimit

**Check class** `weblate.checks.format.MultipleUnnamedFormatsCheck`

**Flag to ignore** `ignore-unnamed-format`

Te vargu ka ndryshore të shumta të paemërtuara, duke e bërë të pamundur renditjen e tyre nga përkkhyesit.

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

Pa forma shumësi

**Summary** Vargu është përdorur si shumësi, por nuk përdoren forma shumësi

**Fokus** Vargje të burimit

**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.

Për shembull, me Gettext në Python do të ishte:

```python
from gettext import ngettext

print ngettext("Selected %d file", "Selected %d files", files) % files
```
1.7 Kërkim

Të reja në versionin 3.9.

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, Translate and Context fields.

<table>
<thead>
<tr>
<th>Query examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Review strings changed by other users</td>
</tr>
<tr>
<td>Translated strings</td>
</tr>
<tr>
<td>Strings with comments</td>
</tr>
<tr>
<td>Strings with any failing checks</td>
</tr>
<tr>
<td>Strings with suggestions from others</td>
</tr>
<tr>
<td>Approved strings with suggestions</td>
</tr>
<tr>
<td>All untranslated strings added the past month</td>
</tr>
<tr>
<td>Translated strings in a certain language</td>
</tr>
</tbody>
</table>
1.7.1 Kërkim i thjeshtë

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 Fusha

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 Comment string case insensitive search.
location:TEXT Location string case insensitive search.
priority:NUMBER Përparësivargu.
added:DATETIME Timestamp for when the string was added to Weblate.
state:TEXT State search (approved, translated, needs-editing, empty, read-only), supports Operatorë fushash.
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.
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 Emër përbërësi.
project:TEXT Identifikues projekt, shihni Identifikues URL-je.
changed_by:TEXT String was changed by author with given username.
changed:DATETIME String content was changed on date, supports Operatorë fushash.
change_time:DATETIME String was changed on date, supports Operatorë fushash, 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 Searching for changes for examples.
check:TEXT Për vargun ka kontroll të dështuar.
dismissed_check:TEXT String has dismissed check.
comment:TEXT Kërko te komente përdoruesi.
comment_author:TEXT Filtrojnë sipas autori komentesh.
suggestion:TEXT Kërko te sugjerime.
suggestion_author:TEXT Filtrojnë sipas autori sugjerimesh.
explanation:TEXT Search in explanations.
1.7.3 Operatorë buelanë

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 Operatorë fushash

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

- **state:**>=translated State is translated or better (approved).
- **ndryshuar më:** 2019 Ndryshuar në vitin 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 Searching for changes

Të reja në versionin 4.4.

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 Shprehje të rregullta

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:
The Weblate Manual, Hedhje Në Qarkullim 4.9

Translation

English

Singular

Word

Plural

Word

Czech, One

Word

Czech, Few

Word

Czech, Other

Word

Plural formula: \[(n=1) \, ? \, 0 \, : \, (n=2 \, \&\, n=4) \, ? \, 1 \, : \, 2\]

Not translated strings • \textit{state:empty}
Strings needing action • \textit{state:<translated}
Translated strings • \textit{state:>translated}
Strings marked for edit • \textit{state:needs-editing}
Strings with suggestions • \textit{has:suggestion}
Strings with variants • \textit{has:variant}
Strings with labels • \textit{has:label}
Strings with context • \textit{has:context}
Strings needing action without suggestions • \textit{state:translated AND NOT has:suggestion}
Strings with comments • \textit{has:comment}
Strings with any failing checks • \textit{has:check}
Approved strings • \textit{state:approved}
Strings waiting for review • \textit{state:translated}

Save "Suggest" "Skip"

Nearby strings: 20

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

Explanation

No explanation currently provided.

Labels

No labels currently set.

Flags

python-format

Source string location

webinterface/templates/translation.html

String age

13 seconds ago

Source string age

14 seconds ago

Translation file

webinterface/local/cz/C(TestCase string)

Powered by Weblate 4.8

About Weblate
Legal
Contact
Documentation
Donate to Weblate

50 Kapitulli 1. Dokumentime përdoruesi
1.7.9 Renditje përfundimesh

There are many options to order the strings according to your needs:
1.8  Rrjedha pune përkrhimi

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  Hyrje për përkrthim

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:

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

Lyp përpunim 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).

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

I miratuar 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.

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

The states are represented in the translation files when possible.

Ndihmëz: In case file format you use does not support storing states, you might want to use Vëru shenjë përkrthimeve të pandryshuara si “Lyp përditësim” addon to flag unchanged strings as needing editing.

Shihni edhe:

Translation types capabilities, Rrjedha pune përkrhimi

1.8.3  Përkthim i drejtpërdrejtë

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

• Përkrhime mund të përpuñojë cilido përdorues.

• Suggestions are optional ways to suggest changes, when translators are not sure about the change.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Vlerë</th>
<th>Shënim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aktivizoni sugjerime</td>
<td>off</td>
<td>Formësuar në shkalë projekt.</td>
</tr>
<tr>
<td>Enable suggestions</td>
<td>on</td>
<td>It is useful for users to be able to suggest when they are not sure.</td>
</tr>
<tr>
<td>Votim sugjerime</td>
<td>off</td>
<td></td>
</tr>
<tr>
<td>Vetëprano sugjerime</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Grup përkrthyesish</td>
<td>Përdorues</td>
<td>Or Translate with per-project access control.</td>
</tr>
<tr>
<td>Grup shqyrtesis</td>
<td>N/A</td>
<td>Not used.</td>
</tr>
</tbody>
</table>
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.

- Sugjerime mund të shtojë cilido përdorues.
- Për sugjerime mund të votojë cilido përdorues.
- Suggestions become translations when given a predetermined number of votes.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Vlerë</th>
<th>Shënim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aktivizoni sugjerime</td>
<td>off</td>
<td>Formësuar në shkallë projekt.</td>
</tr>
<tr>
<td>Enable suggestions</td>
<td>on</td>
<td></td>
</tr>
<tr>
<td>Votim sugjerimesh</td>
<td>off</td>
<td></td>
</tr>
<tr>
<td>Vetëprano sugjerime</td>
<td>1</td>
<td>You can set higher value to require more peer reviews.</td>
</tr>
<tr>
<td>Grup përkthyesish</td>
<td>Përdorues</td>
<td>Or Translate with per-project access control.</td>
</tr>
<tr>
<td>Grup shqyrtuesish</td>
<td>N/A</td>
<td>Not used, all translators review.</td>
</tr>
</tbody>
</table>

1.8.5 Shqyrtues enkas

Të reja në versionin 2.18: 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.
- Shqyrtuesi mund të miratojë / shmiratojë vargje.
- Reviewer can edit all translations (including approved ones).
- Suggestions can also be used to suggest changes for approved strings.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Vlerë</th>
<th>Shënim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aktivizoni shqyrtme</td>
<td>on</td>
<td>Formësuar në shkallë projekt.</td>
</tr>
<tr>
<td>Enable suggestions</td>
<td>off</td>
<td>It is useful for users to be able to suggest when they are not sure.</td>
</tr>
<tr>
<td>Votim sugjerimesh</td>
<td>off</td>
<td></td>
</tr>
<tr>
<td>Vetëprano sugjerime</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Grup përkthyesish</td>
<td>Përdorues</td>
<td>Or Translate with per-project access control.</td>
</tr>
<tr>
<td>Grup shqyrtuesish</td>
<td>Shqyrtues</td>
<td>Or Review with per-project access control.</td>
</tr>
</tbody>
</table>

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):
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 Kartelë gjuhe ndërmjetëse, 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.
1.8.8 Shqyrtim vargjesh burim

With Aktivizo shqyrtme burimi 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 Shqyrtues enkas - 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.

1.9 Pyetje të Bëra Rëndom

1.9.1 Formësim

Si të krijohet një rrjedhë pune e automatizuar?

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 Hook-e njofimesh 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 Push on commit on your Component configuration in Weblate, this will make Weblate push changes to your repository whenever they happen at Weblate.

How to access repositories over SSH?

Please see Hyrje në depo 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.
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/

# 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/
```

(vazhdon ne faqen pasuese)
# 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

git push

Shihni edhe:

*How to export the Git repository that Weblate uses?*, *Përthimin në vazhdimësi*, *Shëmangje përplasjesh përzierje, Klient Weblate*

**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
```

Shihni edhe:

*Keeping translations same across components*

**How to translate multi-platform projects?**

Weblate supports a wide range of file formats (see *Formate të mbuluar kartelash*) 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.

Shihni edhe:

*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 \texttt{DATA\_DIR} directory and is named \texttt{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 \texttt{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 \textit{Si të krijohet një rrjedhë pune e automatizuar?}).
- 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 \texttt{git submodule} for separating translations from source code while still having them under version control.

1. Krijoni një depo me kartelat tuaja të përkthimit.
2. Shtonite kodu juaj këtë si një nënmodul:

   \begin{verbatim}
   git submodule add git@example.com:project-translations.git path/to/translations
   \end{verbatim}

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:

   \begin{verbatim}
   git submodule update --remote path/to/translations
   \end{verbatim}

Please consult the \texttt{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 \textit{Performance report} link in the admin interface, or open the \texttt{/manage/performance/} URL directly.

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

This is the default committer name, configured when you create a translation component. You can change it in the administration at any time.

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

Shihni edhe:

\textit{Component configuration}
1.9.2 Usage

Si t’i shqyrtoj përkrhimet e të tjërëve?

- There are several review based workflows available in Weblate, see **Rrjedha pune përkrhimi**.
- You can subscribe to any changes made in **Njoftime** 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.

**Shihni edhe:**

**Rrjedha pune përkrhimi**

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.

**Shihni edhe:**

report-source, **Komente**

How can I use existing translations while translating?

- All translations within Weblate can be used thanks to shared translation memory.
- You can import existing translation memory files into Weblate.
- 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.

**Shihni edhe:**

**Përkrhim nga makina, Sugjerime të automatizuar, Kujtesë Përkrhimesh**

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 reformattting the file. If you want to keep the file formatted your way, please use a pre-commit hook for that.

**Shihni edhe:**

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.

Shihni edhe:

*Language definitions*

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.

Shënim: 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 addon *Përditësoni kartela PO, që të përputhen me POT* (`msgmerge`).

Shihni edhe:

updating-target-files

1.9.3 Diagnostikim

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.

Rruga më e lehtë për ta bërë këtë është të xhirohet:

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

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.

- Në rast se dëshirohet për kthimi në gjuhën burim, ju lutemi, ndryshoni `Gjuhë burimi` te rregullimet e përbërësve.
- Në rast se s’dëshirohet për kthimi në gjuhën burim, ju lutemi, hiqeni prej depos.
- In case the translation file for the source language is needed, but should be ignored by Weblate, please adjust the `Filtër gjuhësh` to exclude it.

Ndihmëz: 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 (`ja` and `jp` for Japanese) or including both country specific and generic codes (`fr` and `fr_FR`). See Parsing language codes for more details.

1.9.4 Veçori

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`, `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 Kartela vendore.

Shënim: 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.

Integrim kontrolli versioni
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.

Shihni edhe:

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.

Ndryshuar në versionin 4.2: Translators can translate all the components of a project into a specific language as a whole.

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.

Shënim:  Weblate defaults to POSIX style language codes with underscore, see Language definitions for more details.

Shihni edhe:

Language definitions, Stil kodi gjuhe, Shtim përktimesh të reja

1.10 Formate të mbuluar kartelash

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.

Shihni edhe:

Translation Related File Formats

Shënimi:  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 Formate dygjuhësh dhe njëgjuhësh

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 Vargje Apple iOS). 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 Burime vargjesh Android). 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 Kartelë njëgjuhëshe gjuhe bazë within Weblate, though the naming might vary in your paradigm.

Additionally this workflow can be extended by utilizing Kartelë gjuhe ndërmjetëse to include strings provided by developers, but not to be used as is in the final strings.

1.10.2 Pikasje e vetvetishme

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:

<table>
<thead>
<tr>
<th>Format</th>
<th>Linguality</th>
<th>Shumësa</th>
<th>Komente</th>
<th>Kontekst</th>
<th>Vendndodhje</th>
<th>Flags</th>
<th>Additional states</th>
</tr>
</thead>
<tbody>
<tr>
<td>GNU gettext</td>
<td>dygjuhëshe</td>
<td>po</td>
<td>po</td>
<td>po</td>
<td>po</td>
<td>po^7</td>
<td>lyphërputni</td>
</tr>
<tr>
<td>Monolingual gettext</td>
<td>mono</td>
<td>po</td>
<td>po</td>
<td>po</td>
<td>po</td>
<td>po^7</td>
<td>lyphërputni</td>
</tr>
<tr>
<td>XLIFF</td>
<td>të dyja</td>
<td>po</td>
<td>po</td>
<td>po</td>
<td>po</td>
<td>po^7</td>
<td>lyphërputni, i miratur</td>
</tr>
<tr>
<td>Veti Java</td>
<td>të dyja</td>
<td>no</td>
<td>po</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>m18n lang files</td>
<td>mono</td>
<td>no</td>
<td>po</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Veti GWT</td>
<td>mono</td>
<td>po</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Përkthime Joomla</td>
<td>mono</td>
<td>no</td>
<td>po</td>
<td>no</td>
<td>po</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Qt Linguist .ts</td>
<td>të dyja</td>
<td>po</td>
<td>po</td>
<td>no</td>
<td>po</td>
<td>po^7</td>
<td>lyphërputni</td>
</tr>
<tr>
<td>Burime vargjesh Android</td>
<td>mono</td>
<td>po</td>
<td>po^7</td>
<td>no</td>
<td>no</td>
<td>po^7</td>
<td></td>
</tr>
<tr>
<td>Vargje Apple iOS</td>
<td>dygjuhëshe</td>
<td>no</td>
<td>po</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Vargje PHP</td>
<td>mono</td>
<td>no^1</td>
<td>po</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Kartela JSON</td>
<td>mono</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Kartela JSON i18next</td>
<td>mono</td>
<td>po</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>go-i18n JSON files</td>
<td>mono</td>
<td>po</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>

vazhdon në faqen pasuese
Tabela 1 – vazhduar nga faqja e mëparshme

<table>
<thead>
<tr>
<th>Format</th>
<th>Linguality</th>
<th>Shumësa</th>
<th>Komente</th>
<th>Kontekst</th>
<th>Vendndodhje</th>
<th>Flags</th>
<th>Additional states</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kartelë ARB</td>
<td>mono</td>
<td>po</td>
<td>po</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>po</td>
</tr>
<tr>
<td>JSON WebExtension</td>
<td>mono</td>
<td>po</td>
<td>po</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>Kartela burimesh .XML</td>
<td>mono</td>
<td>no</td>
<td>po</td>
<td>no</td>
<td>no</td>
<td>po</td>
<td></td>
</tr>
<tr>
<td>Kartela CSV</td>
<td>të dyja</td>
<td>no</td>
<td>po</td>
<td>po</td>
<td>po</td>
<td>no</td>
<td>lyp përpunim</td>
</tr>
<tr>
<td>Kartela YAML</td>
<td>mono</td>
<td>no</td>
<td>po</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>Kartela Ruby YAML</td>
<td>mono</td>
<td>po</td>
<td>po</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>Kartela DTD</td>
<td>mono</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>Kartelë e sheshtë XML</td>
<td>mono</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>po</td>
</tr>
<tr>
<td>Kartela Windows RC</td>
<td>mono</td>
<td>no</td>
<td>po</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>Excel Open XML</td>
<td>mono</td>
<td>no</td>
<td>po</td>
<td>po</td>
<td>po</td>
<td>no</td>
<td>lyp përpunim</td>
</tr>
<tr>
<td>Kartela tejëdhenash shitoreje aplikacionesh</td>
<td>mono</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>Kartela titrash</td>
<td>mono</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>po</td>
<td>no</td>
</tr>
<tr>
<td>Kartela HTML</td>
<td>mono</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>Format OpenDocument</td>
<td>mono</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>Format IDML</td>
<td>mono</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>Përkthime INI</td>
<td>mono</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>Inno Setup INI translations</td>
<td>mono</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>TermBase eXchange format</td>
<td>dygjuhëshe</td>
<td>no</td>
<td>po</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>po</td>
</tr>
<tr>
<td>Kartela tekst</td>
<td>mono</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>Stringsdict format</td>
<td>mono</td>
<td>po</td>
<td>po</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>Fluent format</td>
<td>mono</td>
<td>no</td>
<td>po</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td></td>
</tr>
</tbody>
</table>

64 Kapitulli 1. Dokumentime përdoruesi
Vargje vetëm për lexim

Të reja në versionin 3.10.

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 Burime vargjesh Android), but can be emulated in others by adding a read-only flag, see Customizing behavior using flags.

1.10.4 GNU gettext

Most widely used format for translating libre software.

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:

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

#: 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 Component configuration

- Maskë karte (po/*.po)
- Kartelë njërgjuhëshe gjëre bazë (Empty)
- Gjedhe për përkthime të reja (po/messages.pot)
- Format kartelash (Gettext PO file)

Shihni edhe:

devel/gettext, devel/sphinx, Gettext on Wikipedia, PO Files, Pérditësoni ndryshoren ALL_LINGUAS te kartela «configure», Pérshtarit përfundimin e gettext-it, Pérditëso kartelën LINGUAS, Prodho kartela MO, Pérditësoni kartela PO, që të përputhen me POT (msgmerge)

1. Shihni Formate dyguhesh dhe njërgjuhesh
2. Plurals are necessary to properly localize strings with variable count.
3. Comments 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.
6. Shihni Customizing behavior using flags
7. Additional states supported by the file format in addition to «Not translated» and «Translated».
8. The gettext type comments are used as flags.
9. 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.
10. XML comment placed before the `<string>` element, parsed as a developer comment.
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.
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:

```plaintext
#: 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:

```plaintext
#: 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"
```

<table>
<thead>
<tr>
<th>Typical Weblate Component configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë karte</td>
</tr>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
</tr>
<tr>
<td>Gjedhe për përkthime të reja</td>
</tr>
<tr>
<td>Format kartelash</td>
</tr>
</tbody>
</table>

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.

Shihni edhe:

XML Localization Interchange File Format (XLIFF) specification
Translation states

Ndryshuar në versionin 3.3: 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.

Shihni edhe:

Shqyrtues enkas

Hapësira të zbrazëta dhe rreshta të rinj në 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.

Për shembull:

```
<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 Administrim shkronjash).
String keys

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.

<table>
<thead>
<tr>
<th>Typical Weblate Component configuration for bilingual XLIFF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
</tr>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
</tr>
<tr>
<td>Gjedhe për përkthime të reja</td>
</tr>
<tr>
<td>Format kartelash</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Typical Weblate Component configuration for monolingual XLIFF</th>
</tr>
</thead>
<tbody>
<tr>
<td>File mask</td>
</tr>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
</tr>
<tr>
<td>Gjedhe për përkthime të reja</td>
</tr>
<tr>
<td>Format kartelash</td>
</tr>
</tbody>
</table>

Shihni edhe:

XLIFF on Wikipedia, XLIFF, font attribute in XLIFF 1.2, maxwidth attribute in XLIFF 1.2

1.10.6 Veti Java

Format i brendshëm Java për përkthimet.

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.

Shënim: Loading escape sequences works in UTF-8 mode as well, so please be careful choosing the correct encoding set to match your application needs.

<table>
<thead>
<tr>
<th>Typical Weblate Component configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
</tr>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
</tr>
<tr>
<td>Gjedhe për përkthime të reja</td>
</tr>
<tr>
<td>Format kartelash</td>
</tr>
</tbody>
</table>

Shihni edhe:

Java properties on Wikipedia, Mozilla and Java properties files, mi18n lang files, Veti GWT, updating-target-files, Formaton kartelën e vetive Java, Spastro kartela përkthimi
1.10.7 **mi18n lang files**

Të reja në versionin 4.7.

File format used for JavaScript localization by mi18n. Syntactically it matches Veti Java.

<table>
<thead>
<tr>
<th>Typical Weblate Component configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
</tr>
<tr>
<td>Kartelë njëjuhëshe gjëhu bazë</td>
</tr>
<tr>
<td>Gjëdhë për përkthime të reja</td>
</tr>
<tr>
<td>Format kartelash</td>
</tr>
</tbody>
</table>

Shihni edhe:

mi18n Mozilla and Java properties files, Veti Java, updating-target-files, Formaton kartelën e vetive Java, Spastro kartela përkthimi

1.10.8 **Veti GWT**

Format i brendshëm GWT për përkthimet.

GWT properties are usually used as monolingual translations.

<table>
<thead>
<tr>
<th>Typical Weblate Component configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
</tr>
<tr>
<td>Kartelë njëjuhëshe gjëhu bazë</td>
</tr>
<tr>
<td>Gjëdhë për përkthime të reja</td>
</tr>
<tr>
<td>Format kartelash</td>
</tr>
</tbody>
</table>

Shihni edhe:

GWT localization guide, GWT Internationalization Tutorial, Mozilla and Java properties files, updating-target-files, Formaton kartelën e vetive Java, Spastro kartela përkthimi

1.10.9 **Përkthime INI**

Të reja në versionin 4.1.

Format kartele INI për përkthime.

INI translations are usually used as monolingual translations.

<table>
<thead>
<tr>
<th>Typical Weblate Component configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
</tr>
<tr>
<td>Kartelë njëjuhëshe gjëhu bazë</td>
</tr>
<tr>
<td>Gjëdhë për përkthime të reja</td>
</tr>
<tr>
<td>Format kartelash</td>
</tr>
</tbody>
</table>

**Shënim:** Weblate only extracts keys from sections within an INI file. In case your INI file lacks sections, you might want to use Përkthime Joomla or Veti Java instead.

Shihni edhe:

INI Files, Veti Java, Përkthime Joomla, Inno Setup INI translations

---

1.10. **Formate të mbular kartelash** 69
1.10.10 Inno Setup INI translations

Të reja në versionin 4.1.
Inno Setup INI file format for translations.
Inno Setup INI translations are usually used as monolingual translations.

**Shënìm:** The only notable difference to **Përkthime INI** is in supporting \%n and \%t placeholders for line break and tab.

<table>
<thead>
<tr>
<th>Typical Weblate Component configuration</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
<td>language/*.islu</td>
</tr>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
<td>language/en.islu</td>
</tr>
<tr>
<td>Gjedhe për përkthime të reja</td>
<td>Empty</td>
</tr>
<tr>
<td>Format kartelash</td>
<td><strong>Inno Setup INI File</strong></td>
</tr>
</tbody>
</table>

**Shënìm:** Only Unicode files (.islu) are currently supported, ANSI variant (.isl) is currently not supported.

**Shihni edhe:**
INI Files, **Përkthime Joomla, Përkthime INI**

1.10.11 Përkthime Joomla

Të reja në versionin 2.12.
Format i brendshëm Joomla për përkthime.
Joomla translations are usually used as monolingual translations.

<table>
<thead>
<tr>
<th>Typical Weblate Component configuration</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
<td>language/*/com_foobar.ini</td>
</tr>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
<td>language/en-GB/com_foobar.ini</td>
</tr>
<tr>
<td>Gjedhe për përkthime të reja</td>
<td>Empty</td>
</tr>
<tr>
<td>Format kartelash</td>
<td><strong>Kartelë Përkthimi Joomla</strong></td>
</tr>
</tbody>
</table>

**Shihni edhe:**
Specification of Joomla language files, Mozilla and Java properties files, **Përkthime INI, Inno Setup INI translations**

1.10.12 Qt Linguist .ts

Translation format used in Qt based applications.
Qt Linguist files are used as both bilingual and monolingual translations.

<table>
<thead>
<tr>
<th>Typical Weblate Component configuration when using as bilingual</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
<td>i18n/app.*.ts</td>
</tr>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
<td>Empty</td>
</tr>
<tr>
<td>Gjedhe për përkthime të reja</td>
<td>i18n/app.de.ts</td>
</tr>
<tr>
<td>Format kartelash</td>
<td><strong>Kartelë Përkthimi Qt Linguist</strong></td>
</tr>
</tbody>
</table>
Typical Weblate Component configuration when using as monolingual

<table>
<thead>
<tr>
<th>Component</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
<td><code>i18n/app.*.ts</code></td>
</tr>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
<td><code>i18n/app.en.ts</code></td>
</tr>
<tr>
<td>Gjedhe për përkthime të reja</td>
<td><code>i18n/app.en.ts</code></td>
</tr>
<tr>
<td>Format kartelash</td>
<td><strong>Kartelë Përkthimi Qt Linguist</strong></td>
</tr>
</tbody>
</table>

Shihni edhe:

Qt Linguist manual, `Qt.ts`, *Format dygjuhësh dhe njëgjuhësh*

### 1.10.13 Burime vargjesh Android

Android specific file format for translating applications.

Android string resources are monolingual, the *Kartelë njëgjuhëshe gjuhe bazë* is stored in a different location from the others `res/values/strings.xml`.

Typical Weblate Component configuration

<table>
<thead>
<tr>
<th>Component</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
<td><code>res/values-*/strings.xml</code></td>
</tr>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
<td><code>res/values/strings.xml</code></td>
</tr>
<tr>
<td>Gjedhe për përkthime të reja</td>
<td><strong>Empty</strong></td>
</tr>
<tr>
<td>Format kartelash</td>
<td><strong>Burim Vargu Android</strong></td>
</tr>
</tbody>
</table>

Shihni edhe:

Android string resources documentation, Android string resources

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

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

become:

```xml
<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](https://gist.github.com/paour/11291062)
1.10.14 Vargje Apple iOS

Apple specific file format for translating applications, used for both iOS and iPhone/iPad application translations. Apple iOS strings are usually used as bilingual translations.

<table>
<thead>
<tr>
<th>Typical Weblate Component configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
</tr>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
</tr>
<tr>
<td>Gjedhe për përkrthime të reja</td>
</tr>
<tr>
<td>Format kartelash</td>
</tr>
</tbody>
</table>

Shihni edhe:

Stringsdict format, Apple «strings files» documentation, Mac OSX strings

1.10.15 Vargje PHP

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

Shembull karteve:

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

<table>
<thead>
<tr>
<th>Typical Weblate Component configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
</tr>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
</tr>
<tr>
<td>Gjedhe për përkrthime të reja</td>
</tr>
<tr>
<td>Format kartelash</td>
</tr>
</tbody>
</table>

Vargje PHP Lavarel

Ndryshuar në versionin 4.1.

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

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

Shihni edhe:

PHP, Localization in Laravel
1.10.16 Kartela JSON

Të reja në versionin 2.0.

Ndryshuar në versionin 2.16: Since Weblate 2.16 and with translate-toolkit at-least 2.2.4, nested structure JSON files are supported as well.

Ndryshuar në versionin 4.3: 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.
- Kartela JSON i18next
- go-i18n JSON files
- JSON WebExtension
- Kartelë ARB

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

Shembull kartele:

```
{
    "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": ""
    }
}
```

Ndihmëz: 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 Component configuration

<table>
<thead>
<tr>
<th>Component</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
<td>langs/translation-*.json</td>
</tr>
<tr>
<td>Kartelë njëgjubëshë gjyhe bazë</td>
<td>langs/translation-en.json</td>
</tr>
<tr>
<td>Gjedhe për përkthime të reja</td>
<td>Empty</td>
</tr>
<tr>
<td>Format kartelash</td>
<td>JSON nested structure file</td>
</tr>
</tbody>
</table>

Shihni edhe:

JSON, updating-target-files, Përshtatni përfundim JSON-i, Spastro kartela përkthimi,

1.10.17 Kartela JSON i18next

Ndryshuar në versionin 2.17: 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.

Shënim: Weblate supports the i18next JSON v3 format. The v2 and v1 variants are mostly compatible, with exception of how plurals are handled.

Shembull kartele:

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

Typical Weblate Component configuration

<table>
<thead>
<tr>
<th>Component</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
<td>langs/*.json</td>
</tr>
<tr>
<td>Kartelë njëgjubëshë gjyhe bazë</td>
<td>langs/en.json</td>
</tr>
<tr>
<td>Gjedhe për përkthime të reja</td>
<td>Empty</td>
</tr>
<tr>
<td>Format kartelash</td>
<td>kartelë JSON i18next</td>
</tr>
</tbody>
</table>

Shihni edhe:

JSON, i18next JSON Format, updating-target-files, Përshtatni përfundim JSON-i, Spastro kartela përkthimi

1.10.18 go-i18n JSON files

Të reja në versionin 4.1.

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

Shënim: Weblate supports the go-i18n JSON v1 format, for flat JSON formats please use Kartela JSON. The v2 format with hash is currently not supported.
Typical Weblate Component configuration

<table>
<thead>
<tr>
<th>Type</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
<td>langs/*.json</td>
</tr>
<tr>
<td>Kartelë njëgjëshëste gjuhe bazë</td>
<td>langs/en.json</td>
</tr>
<tr>
<td>Gjedhe për përkthime të reja</td>
<td>Empty</td>
</tr>
<tr>
<td>Format kartelash</td>
<td>go-i18n JSON file</td>
</tr>
</tbody>
</table>

Shihni edhe:

JSON, go-i18n, updating-target-files, Përshatnë për perfundim JSON-i, Spastro kartela përkthimi.

1.10.19 Kartelë ARB

Të reja në versionin 4.1.

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

Typical Weblate Component configuration

<table>
<thead>
<tr>
<th>Type</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
<td>lib/l10n/intl_*.arb</td>
</tr>
<tr>
<td>Kartelë njëgjëshëste gjuhe bazë</td>
<td>lib/l10n/intl_en.arb</td>
</tr>
<tr>
<td>Gjedhe për përkthime të reja</td>
<td>Empty</td>
</tr>
<tr>
<td>Format kartelash</td>
<td>Kartelë ARB</td>
</tr>
</tbody>
</table>

Shihni edhe:

JSON, Application Resource Bundle Specification, Internationalizing Flutter apps, updating-target-files, Përshatnë për perfundim JSON-i, Spastro kartela përkthimi

1.10.20 JSON WebExtension

Të reja në versionin 2.16: 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.

Shënëm: 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.

Shembull kartele:

```json
{
  "hello": {
    "message": "Ahoj světe\n",
    "description": "Description",
    "placeholders": {
      "url": {
        "content": "$1",
        "example": "https://developer.mozilla.org"
      }
    },
  },
  "orangutan": {
    "message": "",
    "description": "Description"
  },
  "try": {
    "message": "",
  }
}
```

(vazhdon ne faqen pasuese)

1.10. Formate të mbuluar kartelash
Typical Weblate Component configuration

<table>
<thead>
<tr>
<th>Maskë kartele</th>
<th>_locales/*/messages.json</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
<td>_locales/en/messages.json</td>
</tr>
<tr>
<td>Gjedhe për përkthime të reja</td>
<td>Empty</td>
</tr>
<tr>
<td>Format kartelash</td>
<td>Kartelë WebExtension JSON</td>
</tr>
</tbody>
</table>

1.10.21 Kartela burimesh .XML

Të reja në versionin 2.3.

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.

<table>
<thead>
<tr>
<th>Maskë kartele</th>
<th>Resources/Language.*.resx</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
<td>Resources/Language.resx</td>
</tr>
<tr>
<td>Gjedhe për përkthime të reja</td>
<td>Empty</td>
</tr>
<tr>
<td>Format kartelash</td>
<td>.NET resource file</td>
</tr>
</tbody>
</table>

Shihni edhe:

JSON, Google chrome.i18n, Mozilla Extensions Internationalization

1.10.22 Kartela CSV

Të reja në versionin 2.4.

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 Kartelë njëgjuhëshe gjuhe bazë when your files are monolingual (see Formate dygjuhësh dhe njëgjuhësh).

Sinjalizim: 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.
Shembull kartele:

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

### Typical Weblate Component configuration for bilingual CSV

<table>
<thead>
<tr>
<th>Maskë kartele</th>
<th>locale/*.csv</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
<td>Empty</td>
</tr>
<tr>
<td>Gjedhe për përkkthime të reja</td>
<td>locale/en.csv</td>
</tr>
<tr>
<td>Format kartelash</td>
<td>Kartelë CSV</td>
</tr>
</tbody>
</table>

### Typical Weblate Component configuration for monolingual CSV

<table>
<thead>
<tr>
<th>Maskë kartele</th>
<th>locale/*.csv</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
<td>locale/en.csv</td>
</tr>
<tr>
<td>Gjedhe për përkkthime të reja</td>
<td>locale/en.csv</td>
</tr>
<tr>
<td>Format kartelash</td>
<td>Simple CSV file</td>
</tr>
</tbody>
</table>

Shihni edhe:

CSV

#### 1.10.23 Kartela YAML

Të reja nё versionin 2.9.

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

Shembull kartele YAML:

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

Shihni edhe:

YAML, Kartela Ruby YAML

#### 1.10.24 Kartela Ruby YAML

Të reja nё versionin 2.9.

Ruby i18n YAML files with language as root node.

Shembull kartele Ruby i18n YAML:

```ruby
cs:
  weblate:
    hello: ""
    orangutan: ""
```

(vazhdon ne faqen pasuese)
try: ""
thanks: ""

Typical Weblate Component configuration

| Maskë kartele | translations/messages.*.yml |
| Kartelë njëgjuhëshe gjuhe bazë | translations/messages.en.yml |
| Gjedhe për përkrithme të reja | Empty |
| Format kartelash | Kartelë Ruby YAML |

Shihni edhe:

YAML, Kartela YAML

1.10.25 Kartela DTD

Të reja në versionin 2.18.

Shembull kartele DTD:

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

Typical Weblate Component configuration

| Maskë kartele | locale/*.dtd |
| Kartelë njëgjuhëshe gjuhe bazë | locale/en.dtd |
| Gjedhe për përkrithme të reja | Empty |
| Format kartelash | Kartelë DTD |

Shihni edhe:

Mozilla DTD format

1.10.26 Kartelë XML e sheshtë

Të reja në versionin 3.9.

Shembull kartele XML të sheshtë:

```xml
<?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 Component configuration

| Maskë kartele | locale/*.xml |
| Kartelë njëgjuhëshe gjuhe bazë | locale/en.xml |
| Gjedhe për përkrithme të reja | Empty |
| Format kartelash | Kartelë XML e sheshtë |

Shihni edhe:

Flat XML
1.10.27 Kartela Windows RC

Ndryshuar në versionin 4.1: Support for Windows RC files has been rewritten.

Shënim: Support for this format is currently in beta, feedback from testing is welcome.

Shembull kartele Windows RC:

<table>
<thead>
<tr>
<th>LANGUAGE</th>
<th>LANG_CZECH, SUBLANG_DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>STRINGTABLE</td>
<td>BEGIN</td>
</tr>
<tr>
<td>IDS_MSG1</td>
<td>&quot;Hello, world!\n&quot;</td>
</tr>
<tr>
<td>IDS_MSG2</td>
<td>&quot;Orangutan has %d banana.\n&quot;</td>
</tr>
<tr>
<td>IDS_MSG3</td>
<td>&quot;Try Weblate at <a href="http://demo.weblate.org/!%5Cn">http://demo.weblate.org/!\n</a>&quot;</td>
</tr>
<tr>
<td>IDS_MSG4</td>
<td>&quot;Thank you for using Weblate.&quot;</td>
</tr>
<tr>
<td>END</td>
<td></td>
</tr>
</tbody>
</table>

Typical Weblate Component configuration

| Maskë kartele | lang/*.rc |
| Kartelë njëgjuhëshe gjue bazë | lang/en-US.rc |
| Gjedhe për përkhimë të reja | lang/en-US.rc |
| Format kartelash | Kartela RC |

Shihni edhe:

Windows RC files

1.10.28 Kartela tejëdhënash shitoreje aplikacionesh

Të reja në versionin 3.5.

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.

<table>
<thead>
<tr>
<th>Typical Weblate Component configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
</tr>
<tr>
<td>Kartelë njëgjuhëshe gjue bazë</td>
</tr>
<tr>
<td>Gjedhe për përkhimë të reja</td>
</tr>
<tr>
<td>Format kartelash</td>
</tr>
</tbody>
</table>

Ndihmëz: 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 Përpunim në masë.
1.10.29 Kartela titrash

Të reja në versionin 3.7.

Weblate can translate various subtitle files:

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

<table>
<thead>
<tr>
<th>Typical Weblate Component configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
</tr>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
</tr>
<tr>
<td>Gjedhe për përkthime të reja</td>
</tr>
<tr>
<td>Format kartelash</td>
</tr>
</tbody>
</table>

Shihni edhe:

Subtitles

1.10.30 Excel Open XML

Të reja në versionin 3.2.

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.

1.10.31 Kartela HTML

Të reja në versionin 4.1.

Shënim: 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.

Shihni edhe:

HTML

1.10.32 Kartela tekst

Të reja në versionin 4.6.

Shënim: 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.

There are three flavors of this format:
• Kartelë tekst i thjeshtë
• Kartelë tekst DokuWiki
• Kartelë tekst MediaWiki

Shihni edhe:

Simple Text Documents

1.10.33 Format OpenDocument

Të reja në versionin 4.1.

Shënim: 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.

Shihni edhe:

OpenDocument Format

1.10.34 Format IDML

Të reja në versionin 4.1.

Shënim: 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.

1.10.35 TermBase eXchange format

Të reja në versionin 4.5.

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

<table>
<thead>
<tr>
<th>Typical Weblate Component configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maskë kartele</td>
</tr>
<tr>
<td>Kartelë njëgjuhëshe gjuhë bazë</td>
</tr>
<tr>
<td>Gjedhe për përkhime të reja</td>
</tr>
<tr>
<td>Format kartelash</td>
</tr>
</tbody>
</table>

Shihni edhe:

TBX on Wikipedia, TBX, Fjalorth
1.10.36 Stringsdict format

Të reja në versionin 4.8.

**Shënim:** 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 Component configuration**

<table>
<thead>
<tr>
<th>Maskë kartele</th>
<th>Resources/*.lproj/Localizable.stringsdict</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
<td>Resources/en.lproj/Localizable.stringsdict or Resources/Base.lproj/Localizable.stringsdict</td>
</tr>
<tr>
<td>Gjedhe përkthime të reja</td>
<td>Empty</td>
</tr>
<tr>
<td>Format kartelash</td>
<td>Stringsdict file</td>
</tr>
</tbody>
</table>

**Shihni edhe:**

*Vargje Apple iOS, Stringsdict File Format,*

1.10.37 Fluent format

Të reja në versionin 4.8.

**Shënim:** 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 Component configuration**

<table>
<thead>
<tr>
<th>Maskë kartele</th>
<th>locales/*/messages.ftl</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kartelë njëgjuhëshe gjuhe bazë</td>
<td>locales/en/messages.ftl</td>
</tr>
<tr>
<td>Gjedhe përkthime të reja</td>
<td>Empty</td>
</tr>
<tr>
<td>Format kartelash</td>
<td>Fluent file</td>
</tr>
</tbody>
</table>

**Shihni edhe:**

*Project Fluent website*

1.10.38 Supporting other formats

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.

**Shihni edhe:**

*Translation Related File Formats*
1.11 Integrim kontrolli versioni

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

1.11.1 Hyrje në depo

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 Depo kodi burim and URL push-i për depon using the SSH protocol (for example git@github.com:WeblateOrg/weblate.git).

Depo SSH

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

Sinjalizim: On GitHub, each key can only be used once, see Depo GitHub 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:
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.

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

**Ndihmëz:** 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:
Depo GitHub

Access via SSH is possible (see Depo SSH), 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 Degë push 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 Kyç SSH Weblate-i) 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.

Shihni edhe:

Accessing repositories from Hosted Weblate
**URL të brendshme Weblate**

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.

**Sinjalizim:** 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.

**Arsye për të përdorur këtë:**
- 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 addons can operate on multiple components sharing one repository, for example *Kryej «squash» për depozitëme Git*.

**Depo HTTPS**

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.

**Shënim:** If your username or password contains special characters, those have to be URL encoded, for example https://user%40example.com:%24password%23@bitbucket.org/…

**Përdotrim ndërmjetësi**

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
```

**Shënim:** The proxy configuration needs to be done under user running Weblate (see also *Leje sistemi kartelash*) and with `HOME=$DATA_DIR/home` (see *DATA_DIR*), otherwise Git executed by Weblate will not use it.

**Shihni edhe:**

The *cURL manpage*, *Git config documentation*
1.11.2 Git

Ndihmëz: Weblate needs Git 2.12 or newer.

Shihni edhe: See Hyrje në depo for info on how to access different kinds of repositories.

Git with force push

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.

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

Përshtatje formësimi Git

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
```

Sinjalizim: 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

Të reja në versionin 2.3.

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 creates pull requests. The latter is not needed for merely accessing Git repositories.

Shihni edhe: Pushing changes from Weblate
Pushing changes to GitHub as pull requests

If not wanting to push translations to a GitHub repository, they can be sent as either one or many pull requests instead. You need to configure API credentials to make this work.

Shihni edhe:

GITHUB_USERNAME, GITHUB_TOKEN, GITHUB_CREDENTIALS

1.11.4 GitLab

Të reja në versionin 3.9.

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 creates merge request.

Shihni edhe:

Pushing changes from Weblate

Pushing changes to GitLab as merge requests

If not wanting to push translations to a GitLab repository, they can be sent as either one or many merge requests instead. You need to configure API credentials to make this work.

Shihni edhe:

GITLAB_USERNAME, GITLAB_TOKEN, GITLAB_CREDENTIALS

1.11.5 Pagure

Të reja në versionin 4.3.2.

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 creates merge request.

Shihni edhe:

Pushing changes from Weblate

Pushing changes to Pagure as merge requests

If not wanting to push translations to a Pagure repository, they can be sent as either one or many merge requests instead. You need to configure API credentials to make this work.

Shihni edhe:

PAGURE_USERNAME, PAGURE_TOKEN, PAGURE_CREDENTIALS
1.11.6 Gerrit

Të reja në versionin 2.2.

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.

1.11.7 Mercurial

Të reja në versionin 2.1.

Mercurial is another VCS you can use directly in Weblate.

Shënim: It should work with any Mercurial version, but there are sometimes incompatible changes to the command-line interface which breaks Weblate integration.

Shihni edhe:

See Hyrje në depo for info on how to access different kinds of repositories.

1.11.8 Subversion

Të reja në versionin 2.8.

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.

Shënim: 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.

Ndryshuar në versionin 2.19: Before this, only repositories using the standard layout were supported.

Kredenciale Subversion-i

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
```

Shihni edhe:

DATA_DIR
1.11.9 Kartela vendore

1.11.10 Git

Ndihmëz: Underneath, this uses Git. It requires Git installed and allows you to switch to using Git natively with full history of your translations.

Të reja në versionin 3.8.

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

Të reja në versionin 2.6: 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 Klient Weblate.

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 examples**

**Example request:**

```bash
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/"
}
```

**CURL example:**

```bash
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.

The status of rate limiting is reported in following headers:

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-RateLimit-Limit</td>
<td>Rate limiting limit of requests to perform</td>
</tr>
<tr>
<td>X-RateLimit-Remaining</td>
<td>Remaining limit of requests</td>
</tr>
<tr>
<td>X-RateLimit-Reset</td>
<td>Number of seconds until ratelimit window resets</td>
</tr>
</tbody>
</table>

Ndryshuar në versionin 4.1: Added ratelimiting status headers.

Shihni edhe:

Rate limiting, Rate limiting
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 Përdorues

Të reja në versionin 4.0.

**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.

Shihni edhe:

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

**POST /api/users/**

Creates a new user.

Parameters

- **username**(string) – Emër përdoruesi
- **full_name**(string) – User full name
- **email**(string) – User email
- **is_superuser**(boolean) – Is user superuser? (optional)
- **is_active**(boolean) – Is user active? (optional)

**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
- **date_joined** *(string)* – date the user is created
- **groups** *(array)* – link to associated groups; see **GET /api/groups/(int:id)/**

Example JSON data:

```json
{
    "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,
    "date_joined": "2020-01-29T18:42:42.617681Z",
    "url": "http://example.com/api/users/exampleusername/",
    "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
- **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
- **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) – Number of translations by user
   • suggested(int) – Number of suggestions by user
   • uploaded(int) – Number of uploads by user
   • commented(int) – Number of comments by user
   • languages(int) – Number of languages user can translate

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) – ID of notification registered

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)** – ID of notification registered

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)** – ID of notification registered

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 (int)**

1.12.4 Grupe

Të reja në versionin 4.0.

**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.

Shihni edhe:

Group object attributes are documented at **GET /api/groups/(int:id)/**.

**POST /api/groups/**

Creates a new group.

Parameters

• **name (string)** – Emër grupi
• **project_selection (int)** – Group of project selection from given options
• **language_selection (int)** – Group of languages selected from given options

**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)/

• **componentlist** *(array)* – link to associated componentlist; see GET /api/component-lists/(str:slug)/

Example JSON data:

```json
{
  "name": "Guests",
  "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/**

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/**

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/**

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 Role

GET /api/roles/
Returns a list of all roles associated with user. If user is superuser, then list of all existing roles is returned.

Shihni edhe:
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:

```json
{
  "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
The Weblate Manual, Hëndje Në Qarkullim 4.9

- 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 Gjuhë

**GET /api/languages/**
Returns a list of all languages.

Shihni edhe:
Language object attributes are documented at **GET /api/languages/(string:language)/**.

**POST /api/languages/**
Creates a new language.

**Parameters**
- code (string) – Emër gjuhe
- name (string) – Emër gjuhe
- direction (string) – Drejtim teksti
- plural (object) – Language plural formula and number

**GET /api/languages/(string: language)**
Returns information about a language.

**Parameters**
- language (string) – Kod gjuhe

**Response JSON Object**
- code (string) – Kod gjuhe
- direction (string) – Drejtim teksti
- plural (object) – Object of language plural information
- aliases (array) – Array of aliases for language

**Example JSON data:**

```json
{
    "code": "en",
    "direction": "ltr",
    "name": "English",
    "plural": {
        ...
    }
}
```
1.12. Weblate’s REST API

PUT /api/languages/(string: language) /
Changes the language parameters.

Parameters

- language (string) – Language’s code

Request JSON Object

- name (string) – Emër gjuhe
- direction (string) – Drejtim teksti
- 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) – Emër gjuhe
- direction (string) – Drejtim teksti
- plural (object) – Language plural details

DELETE /api/languages/(string: language) /
Deletes the language.

Parameters

- language (string) – Language’s code

GET /api/languages/(string: language)/statistics/
Returns statistics for a language.

Parameters

- language (string) – Kod gjuhe

Response JSON Object

- total (int) – total number of strings
- total_words (int) – total number of words
- last_change (timestamp) – last changes in the language
• **recentchanges** *(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)* – number of fuzzy (marked for edit) strings
• **fuzzy_percent** *(int)* – percentage of fuzzy (marked for edit) strings
• **failing** *(int)* – number of failing strings
• **failing_percent** *(int)* – percentage of failing strings

### 1.12.7 Projekte

**GET /api/projects/**
Returns a list of all projects.

Shihni edhe:
Project object attributes are documented at **GET /api/projects/(string:project)/**.

**POST /api/projects/**
Të rëna në versionin 3.9.
Creates a new project.

**Parameters**

• **name** *(string)* – Emër projekti
• **slug** *(string)* – Project slug
• **web** *(string)* – Sajt projekti

**GET /api/projects/(string: project)/**
Returns information about a project.

**Parameters**

• **project** *(string)* – Identifikues URL-je projekti

**Response JSON Object**

• **name** *(string)* – Emër projekti
• **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)* – Aktivizoni shqyrime
• `source_review(boolean)` – Aktivizo shqyrtime burimi
• `set_language_team(boolean)` – Caktoni krye «Language-Team»
• `enable_hooks(boolean)` – Aktivizoni hook-e
• `instructions(string)` – Udhëzime përkthimi
• `language_aliases(string)` – Aliase gjuhe

Example JSON data:

```json
{
    "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)/**

Të reja në versionin 4.3.

Edit a project by a PATCH request.

**Parameters**

• `project(string)` – Identifikues URL-je projekti

**PUT /api/projects/(string: project)/**

Të reja në versionin 4.3.

Edit a project by a PUT request.

**Parameters**

• `project(string)` – Identifikues URL-je projekti

**DELETE /api/projects/(string: project)/**

Të reja në versionin 3.9.

Deletes a project.

**Parameters**

• `project(string)` – Identifikues URL-je projekti

**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)` – Identifikues URL-je projekti

**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)` – Identifikues URL-je projekti

**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:**

```json
{
    "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) – Identifikues URL-je projekti

**Request JSON Object**

• **operation** (string) – Operation to perform: one of push, pull, commit, reset, 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:**

```bash
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) – Identifikues URL-je projekti
Response JSON Object

- **results** (array) – array of component objects; see GET /api/components/(string:project)/(string:component)/

**POST /api/projects/(string: project)/components/**
Të reja në versionin 3.9.

Ndryshuar në versionin 4.3: The zipfile and docfile parameters are now accepted for VCS-less components, see Kartela vendore.

Ndryshuar në versionin 4.6: The cloned repositories are now automatically shared within a project using URL të brendshme Weblate. Use disable_autoshare to turn off this. Creates translation components in the given project.

**Ndihmëz:** Use URL të brendshme Weblate when creating multiple components from a single VCS repository.

**Shënim:** 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) – Identifikues URL-je projektit

**Form Parameters**

- **file zipfile** – ZIP file to upload into Weblate for translations initialization
- **file docfile** – Dokument për përkthim
- **boolean disable_autoshare** – Disables automatic repository sharing via URL të brendshme Weblate.

**Response JSON Object**

- **result** (object) – Created component object; see GET /api/components/(string:project)/(string:component)/

JSON cannot 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", \"git_export": ", \"license": ",
```
```json
{
    "license_url": ",
    "name": "Weblate",
    "slug": "weblate",
    "repo": "file:///home/nijel/work/weblate-hello",
    "template": "",
    "new_base": "",
    "vcs": "git"
}
```

### JSON request example:

```bash
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",
    "git_export": "",
    "license": "",
    "license_url": "",
    "name": "Weblate",
    "slug": "weblate",
    "repo": "file:///home/nijel/work/weblate-hello",
    "template": "",
    "new_base": "",
    "vcs": "git"
}
```

### JSON response example:

```json
{
    "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",
```
GET /api/projects/(string: project)/languages/

Returns paginated statistics for all languages within a project.

Të reja në versionin 3.8.

Parameters

- **project (string)** – Identifikues URL-je projektit

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.

Të reja në versionin 3.8.

Parameters

- **project (string)** – Identifikues URL-je projektit

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 Përbërës

**GET /api/components/**

Returns a list of translation components.

Shihni edhe:

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) – Identifikues URL-je projekti
- *component*(string) – Identifikues URL-je përbërësi

**Response JSON Object**

- *project*(object) – the translation project; see **GET /api/projects/**
- *name*(string) – Emër përbërësi
- *slug*(string) – Component slug
- *vcs*(string) – Sistem kontrolli versionesh
- *repo*(string) – Depo kodi burim
- *git_export*(string) – URL depoje të eksportuar
- *branch*(string) – Degë depoje
- *push_branch*(string) – Degë push
- *filemask*(string) – File mask
- *template*(string) – Kartelë njëgjuhëshe gjuhe bazë
- *edit_template*(string) – Përpunoni kartelën bazë
- *intermediate*(string) – Kartelë gjuhe ndërmjetëse
- *new_base*(string) – Gjedhe për përkthime të reja
- *file_format*(string) – Format kartelash
- *license*(string) – Licencë përkthimi
- *agreement*(string) – Marrëveshje kontribuesi
- *new_lang*(string) – Shtim përkthimi të ri
- *language_code_style*(string) – Stil kodi gjuhe
- *source_language*(object) – source language object; see **GET /api/languages/(string:language)/**
- *push*(string) – URL push-i për depon
- *check_flags*(string) – Flamuj përkthimi
- *priority*(string) – Përparësi
- *enforced_checks*(string) – Kontrolle me detyrim
- *restricted*(string) – Restricted access
- *repoweb*(string) – Shfletues depoje
- *report_source_bugs*(string) – Adresë njofëmi të metash në vargje burim
• **merge_style** (string) – Stil përzierje
• **commit_message** (string) – Commit, add, delete, merge and addon messages
• **add_message** (string) – Commit, add, delete, merge and addon messages
• **delete_message** (string) – Commit, add, delete, merge and addon messages
• **merge_message** (string) – Commit, add, delete, merge and addon messages
• **addon_message** (string) – Commit, add, delete, merge and addon messages
• **allow_translation_propagation** (string) – Lejo përhapje përkthimesh
• **enable_suggestions** (string) – Enable suggestions
• **suggestion_voting** (string) – Votim sugjerimesh
• **suggestion_autoaccept** (string) – Vetëprano sugjerime
• **push_on_commit** (string) – Push on commit
• **commit_pending_age** (string) – Moshë ndryshimesh për depozitim
• **auto_lock_error** (string) – Kyçe, në rast gabimesh
• **language_regex** (string) – Filtër gjuhësh
• **variant_regex** (string) – Shprehje e rregullt variantesh
• **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:

```json
{
  "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",
      "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/"
  }
}
```
The Weblate Manual, Hedhje Në Qarkullim 4.9

```
{  
    "web_url": "http://example.com/projects/hello/"  
),  
"source_language": {  
    "code": "en",  
    "direction": "ltr",  
    "name": "English",  
    "url": "http://example.com/api/languages/en/",  
    "web_url": "http://example.com/example.com/languages/en/"  
},  
"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/"  
}
```

PATCH /api/components/(string: project)/

**string: component** Edit a component by a PATCH request.

**Parameters**

- **project (string)** – Identifikues URL-je projektit
- **component (string)** – Identifikues URL-je përbërësit
- **source_language (string)** – Kod gjuhë burimi projektit (opsionale)

**Request JSON Object**

- **name (string)** – emër përbërësit
- **slug (string)** – identifikues përbërësit
- **repo (string)** – URL depoje VCS

**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

```json
{
    "branch": "main",
    "file_format": "po",
    "filemask": "po/*.po",
    "git_export": "",
    "license": "",
    "license_url": "",
    "name": "new name",
    "slug": "weblate",
    "project": {
        "name": "Hello",
        "slug": "hello",
        "source_language": {
            "code": "en",
            "direction": "ltr",
            "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)** – Identifikues URL-je projekti
- **component (string)** – Identifikues URL-je përberësi

**Request JSON Object**

- **branch (string)** – VCS repository branch
- **file_format (string)** – format kartele përthimesh
- **filemask (string)** – mask of translation files in the repository
- **name (string)** – emër përberësi
- **slug (string)** – identifikues përberësi
- **repo (string)** – URL depoje VCS
- **template (string)** – base file for monolingual translations
- **new_base (string)** – base file for adding new translations
- **vcs (string)** – sistem kontrolli versionesh

**DELETE /api/components/**

**string:** *project* / **string:** *component* 
Të reja në versionin 3.9.

1.12. Weblate’s REST API
Fshin një përberësi.

Parameters

- project (string) – Identifikues URL-je projekti
- component (string) – Identifikues URL-je përberësi

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) – Identifikues URL-je projekti
- component (string) – Identifikues URL-je përberësi

Response JSON Object

- results (array) – array of component objects; see GET /api/changes/(int:id)/

GET /api/components/(string: project)/
string: component/screenshots/ Returns a list of component screenshots.

Parameters

- project (string) – Identifikues URL-je projekti
- component (string) – Identifikues URL-je përberësi

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) – Identifikues URL-je projekti
- component (string) – Identifikues URL-je përberësi

Response JSON Object

- locked (boolean) – whether component is locked for updates

Example JSON data:

```json
{
  "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) – Identifikues URL-je projekti
- component (string) – Identifikues URL-je përberësi

Request JSON Object

- lock – Boolean whether to lock or not.

CURL example:
```bash
curl \
  -d "lock=true" \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/components/hello/weblate/repository/
```

**JSON request example:**

```plaintext
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:**

```plaintext
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 the same as for **GET /api/projects/(string:project)/repository/**.

**Parameters**

- **project (string)** – Identifikues URL-je projektit
- **component (string)** – Identifikues URL-je përbernës

**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)** – Identifikues URL-je projektit
- **component (string)** – Identifikues URL-je përbernës

**Request JSON Object**

- **operation (string)** – Operation to perform: one of push, pull, commit, reset, cleanup
Response JSON Object

- **result** (boolean) – result of the operation

CURL example:

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

JSON request example:

```json
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:

```plaintext
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) – Identifikues URL-je projekti
- **component** (string) – Identifikues URL-je përbehësi

GET /api/components/(string: project)/string: component/new_template/ Downloads template file for new translations.

Parameters

- **project** (string) – Identifikues URL-je projekti
- **component** (string) – Identifikues URL-je përbehësi

GET /api/components/(string: project)/string: component/translations/ Returns a list of translation objects in the given component.

Parameters

- **project** (string) – Identifikues URL-je projekti
- **component** (string) – Identifikues URL-je përbehësi

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*) – Identifikues URL-je projektit
- **component** (*string*) – Identifikues URL-je përbërësit

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:

```
```

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
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",
      "name": "Czech",
      "url": "http://example.com/api/languages/cs/",
      "web_url": "http://example.com/languages/cs/"
   },
   "language_code": "cs",
   "id": 125,
}
```
GET /api/components/(string: project) /
string: component/statistics/ Returns paginated statistics for all translations within component.

Të reja në versionin 2.7.

Parameters

- project (string) – Identifikues URL-je projektia
- component (string) – Identifikues URL-je përberësi

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/ Returns projects linked with a component.

Të reja në versionin 4.5.

Parameters

- project (string) – Identifikues URL-je projektia
- component (string) – Identifikues URL-je përberësi

Response JSON Object

- projects (array) – associated projects; see GET /api/projects/(string:project)/

POST /api/components/(string: project) /
string: component/links/ Associate project with a component.

Të reja në versionin 4.5.

Parameters

- project (string) – Identifikues URL-je projektia
- component (string) – Identifikues URL-je përberësi

Form Parameters

- string project_slug – Project slug

DELETE /api/components/(string: project) /
string: component/links/string: project_slug/ Remove association of a project with a component.

Të reja në versionin 4.5.

Parameters

- project (string) – Identifikues URL-je projektia
1.12.9 Përkrithme

GET /api/translations/
Returns a list of translations.

Shihni edhe:
Translation object attributes are documented at GET /api/translations/ (string:project)/
(string:component)/(string:language)/.

GET /api/translations/ (string:project)/
string: component/string: language/ Sjell informacion rrëth një përkrithimi

Parameters
• project (string) – Identifikues URL-je projektit
• component (string) – Identifikues URL-je përkrërësi
• language (string) – Kod gjuhe përkrithimi

Response JSON Object
• component (object) – component object; see GET /api/components/
(string:project)/(string:component)/
• failing_checks (int) – numër vargjesh që s’i kalonë kontrolllet
• failing_checks_percent (float) – përqindje vargjesh që s’i kalonë kontrolllet
• failing_checks_words (int) – numër fjalësh që s’i kalonë kontrolllet
• filename (string) – emër kartele përkrithimi
• fuzzy (int) – number of fuzzy (marked for edit) strings
• 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) – numër vargjesh me koment
• have_suggestion (int) – numër vargjesh me sugjerim
• is_template (boolean) – whether the translation has a monolingual base
• 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) – emër autori të fundit
• last_change (timestamp) – vullë kohore e ndryshimit të fundit
• 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 për përkrithim
• 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:

```json
{
    "component": {
        "branch": "main",
        "file_format": "po",
        "filemask": "po/*.po",
        "git_export": "",
        "license": "",
        "license_url": "",
        "name": "Weblate",
        "new_base": "",
        "project": {
            "name": "Hello",
            "slug": "hello",
            "source_language": {
                "code": "en",
                "direction": "ltr",
                "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": {
```
DELETE /api/translations/ (string: project) / 
  string: component/string: language/ Të reja në versionin 3.9.

Fshin një përkthim.

Parameters

- **project** (string) – Identifikues URL-je projektit
- **component** (string) – Identifikues URL-je përbërësit
- **language** (string) – Kod gjuhë përkthimi

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) – Identifikues URL-je projektit
- **component** (string) – Identifikues URL-je përbërësit
- **language** (string) – Kod gjuhë përkthimi

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) – Identifikues URL-je projektit
- **component** (string) – Identifikues URL-je përbërësit
- **language** (string) – Kod gjuhë përkthimi
- **q** (string) – Search query string Kërkim (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 monolingual unit.

Parameters

• project (string) – Identifikues URL-je projekt
• component (string) – Identifikues URL-je përberësi
• language (string) – Kod gjuhe përkrthimi

Request JSON Object

• key (string) – Name of translation unit
• value (array) – The translation unit value

Shihni edhe:

Administroni vargje, adding-new-strings

POST /api/translations/(string: project) /

Parameters

• project (string) – Identifikues URL-je projekt
• component (string) – Identifikues URL-je përberësi
• language (string) – Kod gjuhe përkrthimi

Request JSON Object

• mode (string) – Mënyrë përkrthim i automatizuar
• filter_type (string) – Automatic translation filter type
• auto_source (string) – Automatic translation source - mt or others
• component (string) – Aktivizoni kontribut te kujtesë e përbashkët përkrthimesh për projektin, që të mund të hyni te përberës të tjerë.
• engines (array) – Motorë përkrthimi nga makina
• threshold (string) – Prag vlerësimi

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 Shkarkim përkrthimesh).

Shënim: 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) – Identifikues URL-je projekt
• component (string) – Identifikues URL-je përberësi
• language (string) – Kod gjuhe përkrthimi
POST /api/translations/(string: project)/

Parameters
• project (string) – Identifikues URL-je projekti
• component (string) – Identifikues URL-je përmbërsi
• language (string) – Kod gjuhë përkshtimi

Form Parameters
• string conflict – How to deal with conflicts (ignore, replace-translated or replace-approved)
• file file – Uploaded file
• string email – Email autori
• string author – Emër autori
• 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) – Identifikues URL-je projekti
• component (string) – Identifikues URL-je përmbërsi
• language (string) – Kod gjuhë përkshtimi

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) – Identifikues URL-je projekti
• component (string) – Identifikues URL-je përmbërsi
• language (string) – Kod gjuhë përkshtimi

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)/

Të reja në versionin 2.7.

Parameters

- **project**(string) – Identifikues URL-je projekti
- **component**(string) – Identifikues URL-je përbërësi
- **language**(string) – Kod gjuhe përkthimi

Response JSON Object

- **code**(string) – language code
- **failing**(int) – numër kontrollosh të dështuar
- **failing_percent**(float) – përqi ndje kontrollosh të dështuar
- **fuzzy**(int) – number of fuzzy (marked for edit) strings
- **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) – emër autori të fundit
- **last_change**(timestamp) – datë e ndryshimit të fundit
- **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 Njësi

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.

Të reja në versionin 2.10.

GET /api/units/ 
Returns list of translation units.

Shihni edhe:

Unit object attributes are documented at GET /api/units/(int:id)/.

GET /api/units/(int: id)/ 
Ndryshuar në versionin 4.3: The target and source are now arrays to properly handle plural strings.

Returns information about translation unit.

Parameters

- **id**(int) – ID Njësie

Response JSON Object

- **translation**(string) – URL of a related translation object
- **source**(array) – varg burim
• **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 - not translated, 10 - needs editing, 20 - translated, 30 - approved, 100 - read only

• **fuzzy** *(boolean)* – whether the unit is fuzzy or marked for review

• **translated** *(boolean)* – whether the unit is translated

• **approved** *(boolean)* – whether the translation is approved

• **position** *(int)* – unit position in translation file

• **has_suggestion** *(boolean)* – whether the unit has suggestions

• **has_comment** *(boolean)* – whether the unit has comments

• **has_failing_check** *(boolean)* – whether the unit has failing checks

• **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)/

**PATCH /api/units/(int: id)/**

Të reja në versionin 4.3.

Performs partial update on translation unit.

**Parameters**

• **id** *(int)* – ID Njësie

**Request JSON Object**

• **state** *(int)* – unit state, 0 - not translated, 10 - needs editing, 20 - translated, 30 - approved (need review workflow enabled, see Shqyrues enkas)

• **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)/**

Të reja në versionin 4.3.

Performs full update on translation unit.
Parameters

- **id** (*int*) – ID Njësie

Request JSON Object

- **state** (*int*) – unit state, 0 - not translated, 10 - needs editing, 20 - translated, 30 - approved (need review workflow enabled, see *Shqyrtues enkas*)
- **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)/

Të reja në versionin 4.3.

Deletes a translation unit.

Parameters

- **id** (*int*) – ID Njësie

1.12.11 Ndryshime

Të reja në versionin 2.10.

GET /api/changes/ /

Ndryshuar në versionin 4.1: Filtering of changes was introduced in the 4.1 release.

Returns a list of translation changes.

Shihni edhe:

Change object attributes are documented at GET /api/changes/(int:id)/.

Query Parameters

- **user** (*string*) – Username of user to filter
- **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*) – vëlë kohëre e aktit
• **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 Foto ekrani

Të reja në versionin 2.14.

**GET /api/screenshots/**

Returns a list of screenshot string information.

Shihni edhe:

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:**

```bash
curl -X POST \
-F image=@image.png \n-H "Authorization: Token TOKEN" \nhttp://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** – ID Njësie

**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** Remove source string association with screenshot.

**Parameters**

• **id (int)** – Screenshot ID

• **unit_id** – Source string unit ID

**POST /api/screenshots/**

Creates a new screenshot.

**Form Parameters**

• **file image** – Uploaded file

• **string name** – Emërfotoje ekran

• **string project_slug** – Project slug

• **string component_slug** – Component slug

• **string language_code** – Kod gjuhe

**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)/**

Edit partial information about screenshot.

**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)/**

Edit full information about screenshot.

**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)/**
Delete screenshot.

**Parameters**

• **id** (*int*) – Screenshot ID

## 1.12.13 Shtesa

Të reja në versionin 4.4.1.

**GET /api/addons/**
Returns a list of addons.

Shihni edhe:

Add-on object attributes are documented at `GET /api/addons/(int:id)/`.

**GET /api/addons/(int: id)/**
Returns information about addon information.

**Parameters**

• **id** (*int*) – Add-on ID

**Response JSON Object**

• **name** (*string*) – name of an addon
• **component** (*string*) – URL of a related component object
• **configuration** (*object*) – Optional addon configuration

Shihni edhe:

**Shtesa**

**POST /api/components/(string: project)/string: component/addons/** Creates a new addon.

**Parameters**

• **project_slug** (*string*) – Project slug
• **component_slug** (*string*) – Component slug

**Request JSON Object**

• **name** (*string*) – name of an addon
• **configuration** (*object*) – Optional addon configuration

**PATCH /api/addons/(int: id)/**
Edit partial information about addon.

**Parameters**

• **id** (*int*) – Add-on ID
Response JSON Object

- `configuration (object)` – Optional addon configuration

**PUT /api/addons/(int: id)/**
Edit full information about addon.

Parameters

- `id (int)` – Add-on ID

Response JSON Object

- `configuration (object)` – Optional addon configuration

**DELETE /api/addons/(int: id)/**
Delete addon.

Parameters

- `id (int)` – Add-on ID

### 1.12.14 Lista përërrësish

Të reja në versionin 4.0.

**GET /api/component-lists/**
Returns a list of component lists.

Shihni edhe:
Component list object attributes are documented at [GET /api/component-lists/(str:slug)/](#).

**GET /api/component-lists/(str: slug)/**
Returns information about component list.

Parameters

- `slug (string)` – Identifikues liste përërrësish

Response JSON Object

- `name (string)` – Emër i një liste përërrësish
- `slug (string)` – Identifikues i një liste përërrësish
- `show_dashboard (boolean)` – Të shfaqet apo jo në një pульт
- `components (array)` – Link to associated components; see GET /api/components/(string:project)/(string:component)/
- `auto_assign (array)` – Automatic assignment rules

**PUT /api/component-lists/(str: slug)/**
Ndryshon parametrat e listës së përërrësive.

Parameters

- `slug (string)` – Identifikues liste përërrësish

Request JSON Object

- `name (string)` – Emër i një liste përërrësish
- `slug (string)` – Identifikues i një liste përërrësish
- `show_dashboard (boolean)` – Të shfaqet apo jo në një pульт

**PATCH /api/component-lists/(str: slug)/**
Ndryshon parametrat e listës së përërrësve.

Parameters

- `slug (string)` – Identifikues liste përërrësish
• **slug**(string) – Identifikues liste përmbërisht

Request JSON Object

• **name**(string) – emër i një liste përmbërisht
• **slug**(string) – identifikues i një liste përmbërisht
• **show_dashboard**(boolean) – të shfaqet apo jo në një pult

DELETE /api/component-lists/(str: slug)/
Fshin listën e përmbërësve.

Parameters

• **slug**(string) – Identifikues liste përmbërisht

POST /api/component-lists/(str: slug)/components/
Përshoqëron përmbërësi një listë përmbërisht.

Parameters

• **slug**(string) – Identifikues liste përmbërisht

Form Parameters

• **string component_id** – ID Përbërësi

DELETE /api/component-lists/(str: slug)/components/(str: component_slug)
Disassociate a component from the component list.

Parameters

• **slug**(string) – Identifikues liste përmbërisht
• **component_slug**(string) – Component slug

1.12.15 Fjalorth

Ndryshuar në versionin 4.5: Glossaries are now stored as regular components, translations and strings, please use respective API instead.

1.12.16 Tasks

Të reja në versionin 4.4.

GET /api/tasks/
Listing of the tasks is currently not available.

GET /api/tasks/(str: uuid)/
Returns information about a task

Parameters

• **uuid**(string) – Task UUID

Response JSON Object

• **completed**(boolean) – Whether the task has completed
• **progress**(int) – Task progress in percent
• **result**(object) – Task result or progress details
• **log**(string) – Task log
## 1.12.17 Metrics

GET /api/metrics/

Returns server metrics.

**Response JSON Object**

- **units** *(int)* – Number of units
- **units_translated** *(int)* – Number of translated units
- **users** *(int)* – Number of users
- **changes** *(int)* – Number of changes
- **projects** *(int)* – Number of projects
- **components** *(int)* – Number of components
- **translations** *(int)* – Number of translations
- **languages** *(int)* – Number of used languages
- **checks** *(int)* – Number of triggered quality checks
- **configuration_errors** *(int)* – Number of configuration errors
- **suggestions** *(int)* – Number of pending suggestions
- **celery_queues** *(object)* – Lengths of Celery queues, see *Background tasks using Celery*
- **name** *(string)* – Configured server name

## 1.12.18 Hook-e njoftimesh

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)/

Triggers update of a component (pulling from VCS and scanning for translation changes).

GET /hooks/update/(string: project)/

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.

**Shënim:** 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.

**Shihni edhe:**

*Automatically receiving changes from GitHub* For instruction on setting up GitHub integration

**ENABLE_HOOKS** For enabling hooks for whole Weblate

**POST /hooks/gitlab/**
Special hook for handling GitLab notifications and automatically updating matching components.

Shihni edhe:

* Automatically receiving changes from GitLab For instruction on setting up GitLab integration
* [https://docs.gitlab.com/ee/user/project/integrations/webhooks.html](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.

Shihni edhe:

* Automatically receiving changes from Bitbucket For instruction on setting up Bitbucket integration
* [https://support.atlassian.com/bitbucket-cloud/docs/manage-webhooks/](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/**
Të reja në versionin 3.3.
Special hook for handling Pagure notifications and automatically updating matching components.

Shihni edhe:

* Automatically receiving changes from Pagure For instruction on setting up Pagure integration
* [https://docs.pagure.org/pagure/usage/using_webhooks.html](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/**
Të reja në versionin 3.8.
Special hook for handling Azure Repos notifications and automatically updating matching components.

Shihni edhe:

* Automatically receiving changes from Azure Repos For instruction on setting up Azure integration

**ENABLE_HOOKS** For enabling hooks for whole Weblate

**POST /hooks/gitea/**
Të reja në versionin 3.9.
Special hook for handling Gitea Webhook notifications and automatically updating matching components.

Shihni edhe:

* Automatically receiving changes from Gitea Repos For instruction on setting up Gitea integration
* [https://docs.gitea.io/en-us/webhooks/](https://docs.gitea.io/en-us/webhooks/) Generic information about Gitea Webhooks

**ENABLE_HOOKS** For enabling hooks for whole Weblate

### 1.12. Weblate’s REST API
POST /hooks/gitee/
Të reja në versionin 3.9.

Special hook for handling Gitee Webhook notifications and automatically updating matching components.

Shihni edhe:

*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 Eksporte

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

Nxjerrë nga përdorimi që me versionin 2.6: 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.

Shembull kërkesa:

GET /exports/stats/weblate/main/ HTTP/1.1
Host: example.com
Accept: application/json, text/javascript

Shembull përgjigjeje:

HTTP/1.1 200 OK
Vary: Accept
Content-Type: application/json

```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": 0.0,
    "fuzzy": 0,
    "fuzzy_percent": 0.0,
    "last_author": "Michal Čihař",
    "last_change": "2012-03-28T15:07:38+00:00",
    "name": "Dutch",
    "total": 383,
    "total_words": 12124,
    "translated": 307,
    "translated_percent": 79.9,
    "translated_words": 9661,
    "url": "http://hosted.weblate.org/engage/weblate/nl/",
    "url_translate": "http://hosted.weblate.org/projects/weblate/main/nl/
  }
]```

(vazhdon në faqen pasuese)
### 1.12.20 Prurje RSS

Changes in translations are exported in RSS feeds.

**GET /exports/rss/**

- **GET /exports/rss/** *(string: `project`)/** *(string: `component`)/** *(string: `language`)/**
  
  Retrieves RSS feed with recent changes for a translation.

- **GET /exports/rss/** *(string: `project`)*/** *(string: `component`)*
  
  Retrieves RSS feed with recent changes for a component.

- **GET /exports/rss/** *(string: `project`)*
  
  Retrieves RSS feed with recent changes for a project.

- **GET /exports/rss/** *(string: `language`)*
  
  Retrieves RSS feed with recent changes for a language.

- **GET /exports/rss/**
  
  Retrieves RSS feed with recent changes for Weblate instance.

**Shihni edhe:**

RSS on wikipedia

```json
{"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/"
}
```

1.12. Weblate’s REST API
1.13 Klient Weblate

Të reja në versionin 2.7: 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 Instalim

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 Përdorim Docker-i

The Weblate Client is also available as a Docker image.

The image is published on Docker Hub: https://hub.docker.com/r/weblate/wlc

Installing:

```
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]
```

Shembull:

```
docker run --rm weblate/wlc --url https://hosted.weblate.org/api/ list-projects
```

You might want to pass your Kartela Formësimi 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 Si t’ia fillohet

The wlc configuration is stored in `~/.config/weblate` (see Kartela Formësimi 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
```

Shihni edhe:

Kartela Formësimi
1.13.4 Synopsis

```
wlc [arguments] <command> [options]
```

Commands actually indicate which operation should be performed.

1.13.5 Përshkrim

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`.

Argumente

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}
   Përcaktioni formatin e përfundimeve.

--url URL
   Specify the API URL. Overrides any value found in the configuration file, see Kartela Formësimi. 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 Kartela Formësimi. You can find your key in your profile on Weblate.

--config PATH
   Overrides the configuration file path, see Kartela Formësimi.

--config-section SECTION
   Overrides configuration file section in use, see Kartela Formësimi.

Urdhra

Janë gati urdhrat vijues:

version
   Shtyp versionin e tanishëm.

list-languages
   Paraqet gjuhët e përdorura në Weblate.

list-projects
   Paraqet projekte në Weblate.

list-components
   Paraqet përberës në Weblate.

list-translations
   Paraqet përktihme në Weblate.

show
   Shows Weblate object (translation, component or project).

ls
   Lists Weblate object (translation, component or project).

commit
   Commits changes made in a Weblate object (translation, component or project).
pull
Pulls remote repository changes into Weblate object (translation, component or project).

push
Pushes Weblate object changes into remote repository (translation, component or project).

reset
Të reja në versionin 0.7: Supported since wlc 0.7.
Resets changes in Weblate object to match remote repository (translation, component or project).

cleanup
Të reja në versionin 0.9: Supported since wlc 0.9.
Removes any untracked changes in a Weblate object to match the remote repository (translation, component or project).

repo
Displays repository status for a given Weblate object (translation, component or project).

statistics
Displays detailed statistics for a given Weblate object (translation, component or project).

lock-status
Të reja në versionin 0.5: Supported since wlc 0.5.
Shfaq gjendje kyçjeje.

lock
Të reja në versionin 0.5: Supported since wlc 0.5.
Locks component from further translation in Weblate.

unlock
Të reja në versionin 0.5: Supported since wlc 0.5.
Bën shkycjën e përkrimit të përberësit Weblate.

changes
Të reja në versionin 0.7: Supported since wlc 0.7 and Weblate 2.10.
Shfaq ndryshime për një objekt të dhënë.

download
Të reja në versionin 0.7: Supported since wlc 0.7.
Shkarkon një kartelë përkrithimi.

--convert
Converts file format, if unspecified no conversion happens on the server and the file is downloaded as is to the repository.

--output
Specifies file to save output in, if left unspecified it is printed to stdout.

upload
Të reja në versionin 0.9: Supported since wlc 0.9.
Ngarkon një kartelë përkrithimi.

--overwrite
Mbishkruaj përkrithime ekzistuese gjatë ngarkimit.

--input
File from which content is read, if left unspecified it is read from stdin.

Ndihmëz: You can get more detailed information on invoking individual commands by passing --help, for example: wlc ls --help.
1.13.6 Kartela Formësimi

`.weblate`, `.weblate.ini`, `weblate.ini` Ndryshuar në versionin 1.6: The files with `.ini` extension are accepted as well.

Per project configuration file

`C:\Users\NAME\AppData\weblate.ini` Të reja në versionin 1.6.

Kartelë formësimi përdoruesi në Windows.

`~/.config/weblate` Kartelë formësimi përdoruesi

`/etc/xdg/weblate` Kartelë formësimi për mbarë sistemin

The program follows the XDG specification, so you can adjust placement of config files by environment variables XDG_CONFIG_HOME or XDG_CONFIG_DIRS. On Windows APPDATA directory is preferred location for the configuration file.

Following settings can be configured in the `[weblate]` section (you can customize this by `--config-section`):

**key**
KYÇ API për hyrjenë Weblate.

**url**
API server URL, defaults to `http://127.0.0.1:8000/api/`.

**translation**
Path to the default translation - component or project.

Kartela e formësimit, është një kartelë INI, për shembull:

```
[weblate]
url = https://hosted.weblate.org/api/
key = APIKEY
translation = weblate/application
```

Additionally API keys can be stored in the `[keys]` section:

```
[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 Shembuj

Shtyp të dhëna versioni aktual të programit:

```
$ wlc version
version: 0.1
```

Paraqit krejt projektet:

```
$ 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/
```

You can also designate what project wlc should work on:
$ 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/

With this setup it is easy to commit pending changes in the current project:

$ wlc commit

### 1.14 API Python e Weblate-it

#### 1.14.1 Instalim

The Python API is shipped separately, you need to install the *Klient Weblate* (`wlc`) to have it.

```bash
pip install wlc
```

#### 1.14.2 wlc

*WeblateException*

```python
exception wlc.WeblateException
    Klasë bazë për krejt përjashtimet.
```

*Weblate*

```python
class wlc.Weblate *(key='', url=None, config=None)*
```

**Parametra**

- **key** *(str)* – Kyç përdoruesi
- **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)

Parametra path (str) – Shteg kërkeze
Lloj kthimi object
Performa a single API GET call.

post (path, **kwargs)

Parametra path (str) – Shteg kërkeze
Lloj kthimi object
Performa a single API GET call.

1.14.3 wlc.config

WeblateConfig

class wlc.config.WeblateConfig (section='wlc')

Parametra section (str) – Ndërformësimi për të përdorur
Configuration file parser following XDG specification.
load (path=None)

Parametra 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)

Parametra

• 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 Udhëzime formësimi

2.1.1 Instalim i Weblate-it

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.

Domosdoshmëri hardware

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 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.

Shënim: Actual requirements for your installation of Weblate vary heavily based on the size of the translations managed in it.
Instalim

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 `Ndryshore mjedisi Docker` for full list of environment variables.

   ```yaml
   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
   ```

   **Shënim:** 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.

**Ndryshuar në versionin 2.15-2:** 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.

**Ndryshuar në versionin 3.7.1-6:** 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.

**Shihni edhe:**

*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.

<table>
<thead>
<tr>
<th>Tag name</th>
<th>Përshkrim</th>
<th>Use case</th>
</tr>
</thead>
<tbody>
<tr>
<td>latest</td>
<td>Weblate stable release, matches latest tagged release</td>
<td>Rolling updates in a production environment</td>
</tr>
<tr>
<td>&lt;VERSION&gt;-&lt;PATCH&gt;</td>
<td>Weblate stable release</td>
<td>Well defined deploy in a production environment</td>
</tr>
<tr>
<td>edge</td>
<td>Weblate stable release with development changes in the Docker container</td>
<td>Rolling updates in a staging environment</td>
</tr>
<tr>
<td>edge-&lt;DATE&gt;-&lt;SHA&gt;</td>
<td>Weblate stable release with development changes in the Docker container</td>
<td>Well defined deploy in a staging environment</td>
</tr>
<tr>
<td>bleeding</td>
<td>Development version Weblate from Git</td>
<td>Rolling updates to test upcoming Weblate features</td>
</tr>
<tr>
<td>bleeding-&lt;DATE&gt;-&lt;SHA&gt;</td>
<td>Development version Weblate from Git</td>
<td>Well defined deploy to test upcoming Weblate features</td>
</tr>
</tbody>
</table>

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 Instalim for generic deployment instructions, this section only mentions differences compared to it.

Using own SSL certificates

Të reja në versionin 3.8-3.
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:

```yaml
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:

```bash
server {
  listen 443;
  listen [::]:443;
}
```

(vazhdon ne faqen pasuese)
server_name <SITE_URL>;
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 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.

You can do this by sticking with the existing docker-compose and just pull the latest images and then restart:

```
docker-compose stop
docker-compose pull
docker-compose up
```

The Weblate database should be automatically migrated on first startup, and there should be no need for additional manual actions.
Shënim: Upgrades across 3.0 are not supported by Weblate. If you are on 2.x series and want to upgrade to 3.x, first upgrade to the latest 3.0.1-x (at time of writing this it is the 3.0.1-7) image, 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. Please beware of PostgreSQL version changes in this case as it’s not straightforward to upgrade the database, see GitHub issue for more info.

Admin sign in

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.

Shihni edhe:

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:
    UWSGI_WORKERS: 4
    CELERY_MAIN_OPTIONS: --concurrency 2
    CELERY_NOTIFY_OPTIONS: --concurrency 1
    CELERY_TRANSLATE_OPTIONS: --concurrency 1
```

Shihni edhe:

WEBLATE_WORKERS, CELERY_MAIN_OPTIONS, CELERY_NOTIFY_OPTIONS, CELERY_MEMORY_OPTIONS, CELERY_TRANSLATE_OPTIONS, CELERY_BACKUP_OPTIONS, CELERY_BEAT_OPTIONS, UWSGI_WORKERS

Scaling horizontally

Të reja në versionin 4.6.

Sinjalizim: This feature is a technology preview.

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.

### Ndryshore mjedisi Docker

Many of Weblate’s Formësim can be set in the Docker container using environment variables:

#### Rregullime të përgjithshme

**WEBLATE_DEBUG**

Configures Django debug mode using DEBUG.

**Example:**

```yaml
environment:
  WEBLATE_DEBUG: 1
```

**Shihni edhe:**

*Disable debug mode*

**WEBLATE_LOGLEVEL**

Configures the logging verbosity.

**WEBLATE_SITE_TITLE**

Changes the site-title shown in the header of all pages.

**WEBLATE_SITE_DOMAIN**

Configures the site domain. This parameter is required.

**Shihni edhe:**

*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).

**Example:**

```yaml
environment:
  WEBLATE_ADMIN_NAME: Weblate admin
  WEBLATE_ADMIN_EMAIL: noreply@example.com
```

**Shihni edhe:**

*Admin sign in, 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.

**Sinjalizim:** 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.
The Weblate Manual, Hedhje Në Qarkullim 4.9

Shihni edhe:

*Admin sign in*,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.

Shihni edhe:

**WEBLATE_SERVER_EMAIL**

**WEBLATE_DEFAULT_FROM_EMAIL**
Configures the address for outgoing e-mails.

Shihni edhe:

*Formësoni dërgimi email-esh*

**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.

Example:

```
environment:
  WEBLATE_ALLOWED_HOSTS: weblate.example.com,example.com
```

Shihni edhe:

ALLOWED_HOSTS, Allowed hosts setup, *Set correct site domain*

**WEBLATE_REGISTRATION_OPEN**
Configures whether registrations are open by toggling REGISTRATION_OPEN.

Example:

```
environment:
  WEBLATE_REGISTRATION_OPEN: 0
```

**WEBLATE_REGISTRATION_ALLOW_BACKENDS**
Configure which authentication methods can be used to create new account via
REGISTRATION_ALLOW_BACKENDS.

Example:

```
environment:
  WEBLATE_REGISTRATION_OPEN: 0
  WEBLATE_REGISTRATION_ALLOW_BACKENDS: azuread-oauth2,azuread-tenant-
    _oath2
```

**WEBLATE_TIME_ZONE**
Configures the used time zone in Weblate, see TIME_ZONE.

Shënim: To change the time zone of the Docker container itself, use the TZ environment variable.

Example:

```
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.

Ndihmëz: Please see ENABLE_HTTPS documentation for possible caveats.

Shënim: 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.

Example:

```environment:
WEBLATE_ENABLE_HTTPS: 1
```

Shihni edhe:
ENABLE_HTTPS Set correct site domain, WEBLATE_SECURE_PROXY_SSL_HEADER

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.

Shënim: 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.

Example:

```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.

Example:

```environment:
WEBLATE_SECURE_PROXY_SSL_HEADER: HTTP_X_FORWARDED_PROTO,https
```

Shihni edhe:
SECURE_PROXY_SSL_HEADER

WEBLATE_REQUIRE_LOGIN
Enables REQUIRE_LOGIN to enforce authentication on whole Weblate.

Example:

```environment:
WEBLATE_REQUIRE_LOGIN: 1
```

WEBLATE_LOGIN_REQUIRED_URLS_EXCEPTIONS
**WEBLATE_ADD_LOGIN_REQUIRED_URLS_EXCEPTIONS**

Add 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_REMOVE_LOGIN_REQUIRED_URLS_EXCEPTIONS**

**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`.

Shihni edhe: GitHub

**WEBLATE_GITHUB_TOKEN**

Të rëja në versionin 4.3.

Configures GitHub personal access token for GitHub pull-requests via API by changing `GITHUB_TOKEN`.

Shihni edhe: GitHub

**WEBLATE_GITLAB_USERNAME**

Configures GitLab username for GitLab merge-requests by changing `GITLAB_USERNAME`

Shihni edhe: GitLab

**WEBLATE_GITLAB_TOKEN**

Configures GitLab personal access token for GitLab merge-requests via API by changing `GITLAB_TOKEN`.

Shihni edhe: GitLab

**WEBLATE_PAGURE_USERNAME**

Configures Pagure username for Pagure merge-requests by changing `PAGURE_USERNAME`.

Shihni edhe: Pagure

**WEBLATE_PAGURE_TOKEN**

Configures Pagure personal access token for Pagure merge-requests via API by changing `PAGURE_TOKEN`.

Shihni edhe: Pagure

**WEBLATE_SIMPLIFY_LANGUAGES**

Configures the language simplification policy, see `SIMPLIFY_LANGUAGES`.

**WEBLATE_DEFAULT_ACCESS_CONTROL**

Configures the default Kontroll hyrjesh 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 Lejo përhapje përkkthimesh for new components, see `DEFAULT_TRANSLATION_PROPAGATION`.

**WEBLATE_DEFAULT_COMMITER_EMAIL**

Configures `DEFAULT_COMMITER_EMAIL`.

2.1. Udhëzime formësimi

149
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.

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
AllowstocustomizeContent-Security-PolicyHTTPheader.

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
Tërejanënëversionin4.6.
Configuresratelimiter.

Ndihmëz: You can set configuration for any rate limiter scopes. To do that add WEBLATE_ prefix to any of setting described in Rate limiting.
Shihni edhe:

*Rate limiting, RATELIMIT_ATTEMPTS, RATELIMIT_WINDOW, RATELIMIT_LOCKOUT*

**WEBLATE_ENABLE_AVATARS**
Të reja në versionin 4.6.1.

Configures *ENABLE_AVATARS*.

Rregullime mbi përkthim nga makina

Ndihmëz: Configuring API key for a service automatically configures it in *MT_SERVICES*.

**WEBLATE_MT_APERTIUM_API**
Enables *Apertium* machine translation and sets *MT_APERTIUM_API*

**WEBLATE_MT_AWS_REGION**

**WEBLATE_MT_AWS_ACCESS_KEY_ID**

**WEBLATE_MT_AWS_SECRET_ACCESS_KEY**
Configures *AWS* machine translation.

<table>
<thead>
<tr>
<th>Environment</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>WEBLATE_MT_AWS_REGION:</td>
<td>us-east-1</td>
</tr>
<tr>
<td>WEBLATE_MT_AWS_ACCESS_KEY_ID:</td>
<td>AKIAIOSFODNN7EXAMPLE</td>
</tr>
<tr>
<td>WEBLATE_MT_AWS_SECRET_ACCESS_KEY:</td>
<td>wJalrXUtNcS1FEMI/K7MDENG/bPxRfiCYEXAMPLEKEY</td>
</tr>
</tbody>
</table>

**WEBLATE_MT_DEEPL_KEY**
Enables *DeepL* machine translation and sets *MT_DEEPL_KEY*

**WEBLATE_MT_DEEPL_API_URL**
Configures *DeepL* API version to use, see *MT_DEEPL_API_URL*.

**WEBLATE_MT_LIBRETRANSLATE_KEY**
Enables *LibreTranslate* machine translation and sets *MT_LIBRETRANSLATE_KEY*

**WEBLATE_MT_LIBRETRANSLATE_API_URL**
Configures *LibreTranslate* API instance to use, see *MT_LIBRETRANSLATE_API_URL*.

**WEBLATE_MT_GOOGLE_KEY**
Enables *Google Translate* and sets *MT_GOOGLE_KEY*

**WEBLATE_MT_MICROSOFT_COGNITIVE_KEY**
Enables *Microsoft Cognitive Services Translator* and sets *MT_MICROSOFT_COGNITIVE_KEY*

**WEBLATE_MT_MICROSOFT_ENDPOINT_URL**
Sets *MT_MICROSOFT_ENDPOINT_URL*, please note this is supposed to contain domain name only.

**WEBLATE_MT_MICROSOFT_REGION**
Sets *MT_MICROSOFT_REGION*

**WEBLATE_MT_MICROSOFT_BASE_URL**
Sets *MT_MICROSOFT_BASE_URL*

**WEBLATE_MT_MODERNMT_KEY**
Enables *ModernMT* and sets *MT_MODERNMT_KEY*.

**WEBLATE_MT_MYMEMORY_ENABLED**
Enables *MyMemory* machine translation and sets *MT_MYMEMORY_EMAIL* to *WEBLATE_ADMIN_EMAIL*.

Example:
**WEBLATE_MT_GLOSBE_ENABLED**

Enables *Glosbe* machine translation.

```
environment:
    WEBLATE_MT_GLOSBE_ENABLED: 1
```

**WEBLATE_MT_MICROSOFT_TERMINOLOGY_ENABLED**

Enables *Microsoft Terminology Service* machine translation.

```
environment:
    WEBLATE_MT_MICROSOFT_TERMINOLOGY_ENABLED: 1
```

**WEBLATE_MT_SAP_BASE_URL**

**WEBLATE_MT_SAP_SANDBOX_APIKEY**

**WEBLATE_MT_SAP_USERNAME**

**WEBLATE_MT_SAP_PASSWORD**

**WEBLATE_MT_SAP_USE_MT**

Configures *SAP Translation Hub* machine translation.

```
environment:
   WEBLATE_MT_SAP_BASE_URL: "https://example.hana.ondemand.com/translationhub/api/v1/"
    WEBLATE_MT_SAP_USERNAME: "user"
    WEBLATE_MT_SAP_PASSWORD: "password"
    WEBLATE_MT_SAP_USE_MT: 1
```

**Rregullime për mirëfilltësim**

**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_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.
```

(vazhdon në faqen pasuese)
# 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:

```text
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:

```text
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: ou=users,dc=example,
  ou=otherusers,dc=example,dc=com
```

Example with search and bind against Active Directory:

```text
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_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)
```

Shihni edhe:

Mirëfillësim LDAP

GitHub

WEBLATE_SOCIAL_AUTH_GITHUB_KEY
WEBLATE_SOCIAL_AUTH_GITHUB_SECRET
Enables Mirëfillësim GitHub.

Bitbucket

WEBLATE_SOCIAL_AUTH_BITBUCKET_KEY
WEBLATE_SOCIAL_AUTH_BITBUCKET_SECRET
Enables Mirëfillësim Bitbucket.
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
   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.

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
   SAML Identity Provider settings, see Mirëfillësim SAML.

Other authentication settings

WEBLATE_NO_EMAIL_AUTH
   Disables e-mail authentication when set to any value.

Ujdisje baze të dhënash PostgreSQL

The database is created by docker-compose.yml, so these settings affect both Weblate and PostgreSQL containers.

Ujdisje baze të dhënash për Weblate

POSTGRES_PASSWORD
   Fjalëkalim PostgreSQL.

POSTGRES_PASSWORD_FILE
   Path to the file containing the PostgreSQL password. Use as an alternative to POSTGRES_PASSWORD.

POSTGRES_USER
   Emërpërdoruesi PostgreSQL.

POSTGRES_DATABASE
   EmërbazetëdhënashPostgreSQL.

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 *Formësim i Weblate-it që të përdorë PostgreSQL-in*.

**POSTGRES_CONN_MAX_AGE**
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.

**Sinjalizim:** This is currently experimental, not widely tested and not supported by the Weblate team.

**Database backup settings**

**Shihni edhe:**
*Dumped data for backups*

**WEBLATE_DATABASE_BACKUP**
Configure 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.

**Shihni edhe:**
*Aktivizon përdorim fshehtine*

**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_TLS**
Enables using SSL for Redis connection.

**REDIS_VERIFY_SSL**
Can be used to disable SSL certificate verification for Redis connection.
Ujdisje shërbyesi email-esh

To make outgoing e-mail work, you need to provide a mail server.

Shembull formësimi TLS-je:

```
environment:
  WEBLATE_EMAIL_HOST: smtp.example.com
  WEBLATE_EMAIL_HOST_USER: user
  WEBLATE_EMAIL_HOST_PASSWORD: pass
```

Shembull formësimi SSL-je:

```
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
```

Shihni edhe:

*Formësim i e-mail-eve që dërgohen*

**WEBLATE_EMAIL_HOST**
Mail server hostname or IP address.

Shihni edhe:

```
WEBLATE_EMAIL_PORT, WEBLATE_EMAIL_USE_SSL, WEBLATE_EMAIL_USE_TLS,
EMAIL_HOST
```

**WEBLATE_EMAIL_PORT**
Mail server port, defaults to 25.

Shihni edhe:

```
EMAIL_PORT
```

**WEBLATE_EMAIL_HOST_USER**
E-mail authentication user.

Shihni edhe:

```
EMAIL_HOST_USER
```

**WEBLATE_EMAIL_HOST_PASSWORD**
E-mail authentication password.

Shihni edhe:

```
EMAIL_HOST_PASSWORD
```

**WEBLATE_EMAIL_HOST_PASSWORD_FILE**
Path to the file containing the e-mail authentication password.

Shihni edhe:

```
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**.

Shihni edhe:

```
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`.

Shihni edhe:
`WEBLATE_EMAIL_PORT, WEBLATE_EMAIL_USE_SSL, EMAIL_USE_TLS`

**WEBLATE_EMAIL_BACKEND**
Configures Django back-end to use for sending e-mails.

Shihni edhe:
Formësoni dërgimi email-esh, EMAIL_BACKEND

**Site integration**

**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).

**CDN përkhimesh**

**WEBLATE_LOCALIZE_CDN_URL**
**WEBLATE_LOCALIZE_CDN_PATH**
Të reja në versionin 4.2.1.

Configuration for *CDN përkhimesh JavaScript*.

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

**Note:** You are responsible for setting up serving of the files generated by Weblate, it only does stores the files in configured location.

**Note:**

- `weblate-cdn`, `LOCALIZE_CDN_URL`, `LOCALIZE_CDN_PATH`

### Changing enabled apps, checks, addons or autofixes

Të reja në versionin 3.8.5.

The built-in configuration of enabled checks, addons 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`

Example:

```plaintext
environment:
  WEBLATE_ADD_AUTOFIX: weblate.trans.autofixes.whitespace.
  WEBLATE_REMOVE_AUTOFIX: weblate.trans.autofixes.whitespace.
  WEBLATE_ADD_ADDONS: customize.addons.MyAddon,customize.addons.OtherAddon
```

**Note:**

- `CHECK_LIST`, `AUTOFIX_LIST`, `WEBLATE_ADDONS`, `INSTALLED_APPS`

### Container settings

**WEBLATE_WORKERS**

Të reja në versionin 4.6.1.

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 `UWSGI_WORKERS`. You can use these settings to fine-tune.

**CELERY_MAIN_OPTIONS**

**CELERY_NOTIFY_OPTIONS**

**CELERY_MEMORY_OPTIONS**

**CELERY_TRANSLATE_OPTIONS**

---

2.1. Udhezime formesimi
**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`.

Example:

```plaintext
environment:
  CELERY_MAIN_OPTIONS: --concurrency 16
```

Shihni edhe:

Celery worker options, **Background tasks using Celery**

**UWSGI_WORKERS**

Configure how many uWSGI workers should be executed.

It defaults to `WEBLATE_WORKERS`.

Example:

```plaintext
environment:
  UWSGI_WORKERS: 32
```

**WEBLATE_SERVICE**

Defines which services should be executed inside the container. Use this for **Scaling horizontally**.

Following services are defined:

- **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** Translation memory Celery worker.
- **celery-notify** Notifications Celery worker.
- **celery-translate** Automatic translation Celery worker.
- **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_weblata-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.

Shihni edhe:

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

Të reja në versionin 3.8-5.

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.

Ndihmëz: 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.

Alternatively you can also include own module (see Përshatje e Weblate-it) 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

Të reja në versionin 3.8-5.

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.

Shihni edhe:
Përshatje e Weblate-it

Installing on Debian and Ubuntu

Domosdoshmëri hardware

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 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.

Shënim: Actual requirements for your installation of Weblate vary heavily based on the size of the translations managed in it.

Instalim

System requirements

Install the dependencies needed to build the Python modules (see Domosdoshmëri software):

```bash
apt install libxml2-dev libxslt-dev libfreetype6-dev libjpeg-dev libz-dev libyaml-dev
libcairo-dev gir1.2-pango-1.0 libgirepository1.0-dev libacl1-dev libssl-dev
build-essential python3-gdbm python3-dev python3-pip python3-virtualenv
```

Install wanted optional dependencies depending on features you intend to use (see Varësi opsionale):

```bash
apt install tesseract-ocr libtesseract-dev libleptonica-dev
```

Optionally install software for running production server, see Running server, Ujdisje baze të dhënash për 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:

```bash
# 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

Ndihmëz: 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 dependencies:

   ```
   pip install Weblate
   ```

4. Install database driver:

   ```
   pip install psycop2-binary
   ```

5. Install wanted optional dependencies depending on features you intend to use (some might require additional system libraries, check Varësi opsionale):

   ```
   pip install ruamel.yaml aeidon boto3 zeep chardet tesserocr
   ```

Configuring Weblate

Shënim: 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.


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 Përshtatje formësimi.

3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check Ujdisje baze të dhënas për 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):

2.1. Udhëzime formësimi
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.

- Login 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/](http://localhost:8000/manage/performance/) URL or using `weblate check --deploy`, see Production setup.

**Adding translation**

1. Open the admin interface ([http://localhost:8000/create/project/](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: Component name, VCS repository address and mask for finding translatable files. Weblate supports a wide range of formats including gettext PO files, Android resource strings, iOS string properties, Java properties or Qt Linguist files, see Formate të mbuluar kartelash 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**

**Domosdoshmëri hardware**

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 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.

**Shënim:** Actual requirements for your installation of Weblate vary heavily based on the size of the translations managed in it.

## Instalim

### System requirements

Install the dependencies needed to build the Python modules (see *Domosdoshmëri software*):

```bash
zypper install \
    libxslt-devel libxml2-devel freetype-devel libjpeg-devel zlib-devel libyaml-devel \
    cairo-devel typelib-1_0-Pango-1_0 gobject-introspection-devel libacl-devel \
    python3-pip python3-virtualenv python3-devel git
```

Install wanted optional dependencies depending on features you intend to use (see *Varësi opsionale*):

```bash
zypper install tesseract-ocr tesseract-devel leptonica-devel
```

Optionally install software for running production server, see *Running server, Ujdisje baze të dhënasht për 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:

```bash
# 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

**Ndihmëz:** 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:

   ```bash
   virtualenv --python=python3 ~/weblate-env
   ```

2. Activate the virtualenv for Weblate:
3. Install Weblate including all dependencies:

```
pip install Weblate
```

4. Install database driver:

```
pip install psycopg2-binary
```

5. Install wanted optional dependencies depending on features you intend to use (some might require additional system libraries, check Varësi opsionale):

```
pip install ruamel.yaml aeidon boto3 zeep chardet tesserocr
```

**Configuring Weblate**

**Shënim:** 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.


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 Përshtatje formësimi.

3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check Ujdisje baze të dhënash për 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.

- Login 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 or using `weblate check --deploy`, see `Production setup`.

Adding translation

1. Open the admin interface ([http://localhost:8000/create/project/](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: Component name, VCS repository address and mask for finding translatable files. Weblate supports a wide range of formats including gettext PO files, Android resource strings, iOS string properties, Java properties or Qt Linguist files, see `Formatë të mbuluar kartelash` 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

Domosdoshmëri hardware

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 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.

**Shënim:** Actual requirements for your installation of Weblate vary heavily based on the size of the translations managed in it.
Instalim

System requirements

Install the dependencies needed to build the Python modules (see Domosdoshmëri software):

```
dnf install \
  libxslt-devel libxml2-devel freetype-devel libjpeg-devel zlib-devel libyaml-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 Varësi opsionale):

```
dnf install tesseract-langpack-eng tesseract-devel leptonica-devel
```

Optionally install software for running production server, see Running server, Ujdisje baze të dhënash për 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

Ndihmëz: 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 dependencies:

   ```
   pip install Weblate
   ```

4. Install database driver:

   ```
   pip install psycopg2-binary
   ```
5. Install wanted optional dependencies depending on features you intend to use (some might require additional system libraries, check Varësi opsionale):

```
pip install ruamel.yaml aeidon boto3 zeep chardet tesserocr
```

Configuring Weblate

**Shënim:** 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.


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 Përshtatje formësimi.

3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check Ujdisje baze të dhënash për 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/.
- Login 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 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: Component name, VCS repository address and mask for finding translatable files. Weblate supports a wide range of formats including gettext PO files, Android resource strings, iOS string properties, Java properties or Qt Linguist files, see Formate të mbuluar kartelash 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

Domosdoshmëri hardware

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 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.

**Shënim:** Actual requirements for your installation of Weblate vary heavily based on the size of the translations managed in it.

Instalim

System requirements

Install the dependencies needed to build the Python modules (see Domosdoshmëri software):

```
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 Varësi opsionale):
brew install tesseract

Optionally install software for running production server, see Running server, Ujdisje baze të dhënashe për 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:

```bash
# 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

**Ndihmëz:** 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:
   ```bash
   virtualenv --python=python3 ~/weblate-env
   ```

2. Activate the virtualenv for Weblate:
   ```bash
   . ~/weblate-env/bin/activate
   ```

3. Install Weblate including all dependencies:
   ```bash
   pip install Weblate
   ```

4. Install database driver:
   ```bash
   pip install psycopg2-binary
   ```

5. Install wanted optional dependencies depending on features you intend to use (some might require additional system libraries, check Varësi opsionale):
   ```bash
   pip install ruamel.yaml aeidon boto3 zeep chardet tesserocr
   ```

Configuring Weblate

**Shënim:** 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.

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 Përshtatje formësimi.

3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check Ujdisje baze të dhënash për Weblate for production ready setup):

```bash
weblate migrate
```

4. Create the administrator user account and copy the password it outputs to the clipboard, and also save it for later use:

```bash
weblate createadmin
```

5. Collect static files for web server (see Running server and Serving static files):

```bash
weblate collectstatic
```

6. Compress JavaScript and CSS files (optional, see Compressing client assets):

```bash
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:

```bash
~/weblate-env/lib/python3.7/site-packages/weblate/examples/celery start
```

8. Start the development server (see Running server for production setup):

```bash
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/.
- Login 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 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: Component name, VCS repository address and mask for finding translatable files. Weblate supports a wide range of formats including gettext PO files, Android resource strings, iOS string properties, Java properties or Qt Linguist files, see Formate të mbuluar kartelash 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:
   - 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 Përshtatje formësimi.

6. Create the database used by Weblate, see Ujdisje baze të dhënash për Weblate.

7. Build Django tables, static files and initial data (see Mbushje e bazës së të dhënave and Serving static files):
   ```
   weblate migrate
   weblate collectstatic
   weblate compress
   weblate compilemessages
   ```

   Shënim: 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/>.

Instalim

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:

```
-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:

```
# If the template is already uploaded
$ oc process --parameters -n <PROJECT> weblate
```

Provisioning

You can also use the CLI to process templates and use the configuration that is generated to create objects immediately.

```
-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.
Eliminate

```
$ 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>
```

Formësim

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 `Ndryshore mjedisi Docker` for full list of environment variables.

Installing on Kubernetes

Shënim: 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>`.

Instalim

```
helm repo add weblate https://helm.weblate.org
helm install my-release weblate/weblate
```

Formësim

For further configuration options, see `Ndryshore mjedisi Docker` 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**
2.1.2 Domosdoshmëri software

Sistem operativ

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.

Shërbimetëtjera

Weblate is using other services for its operation. You will need at least following services running:

- PostgreSQL database server, see Ujdisje baze të dhënash për Weblate.
- Redis server for cache and tasks queue, see Background tasks using Celery.
- SMTP server for outgoing e-mail, see Formësim i email-eve që dërgohen.

Varësi Python

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.

Varësitëmë të rëndësishme:

Django https://www.djangoproject.com/
Celery https://docs.celeryproject.org/
Translate Toolkit https://toolkit.translatehouse.org/
translation-finder https://github.com/WeblakeOrg/translation-finder
Python Social Auth https://python-social-auth.readthedocs.io/
Django REST Framework https://www.django-rest-framework.org/

Varësi opsionale

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 support) https://github.com/viraptor/phply
tesserocr (optional for screenshots OCR) https://github.com/sirfz/tesserocr
akismet (optional for suggestion spam protection) https://github.com/ubernostrum/akismet
ruamel.yaml (optional for Kartela YAML) https://pypi.org/project/ruamel.yaml/
Zeep (optional for Microsoft Terminology Service) https://docs.python-zeep.org/
aeidon (optional for Kartela titrash) https://pypi.org/project/aeidon/
Database backend dependencies

Weblate supports PostgreSQL, MySQL and MariaDB, see Ujdisje baze të dhënash për Weblate and backends documentation for more details.

Domosdoshmëri të tjera sistemi

Varësitë vijuese duhet të instalohen te sistemi:

Git https://git-scm.com/

Pango, Cairo and related header files and gir introspection data https://cairographics.org/, https://pango.gnome.org/, see Pango dhe 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 Varësi Python 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 dhe Cairo

Ndryshuar në versionin 3.7.

Weblate uses Pango and Cairo for rendering bitmap widgets (see promotion) and rendering checks (see Administrim shkronjash). 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>.

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: 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:

• **Përdorni wkd për të shkarkuar kyçin:**

  ```sh
  $ 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]
  sub rsa4096 2009-06-17 [E]
  sub rsa4096 2015-09-09 [S]
  ```

• **Download the keyring from Michal's server, then import it with:**

  ```sh
  $ gpg --import wmxth3chu9jfxdxywj1skpmhsj311mzm
  ```

• **Download and import the key from one of the key servers:**

  ```sh
  $ 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:

```sh
$ 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
  using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
  Good signature from "Michal Čihař <michal@cihar.com>" [ultimate]
  aka "Michal Čihař <nijel@debian.org>" [ultimate]
  aka "[jpeg image of size 8848]" [ultimate]
  aka "Michal Čihař (Braiins) <michal.cihar@braiins.cz>" _
  →[ultimate]
  WARNING: This key is not certified with a trusted signature!
  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 though signatures of others, who have met the developer in person.

Once the key is trusted, the warning will not occur:

```sh
$ 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
  using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
  Good signature from "Michal Čihař <michal@cihar.com>" [ultimate]
  aka "Michal Čihař <nijel@debian.org>" [ultimate]
  aka "[jpeg image of size 8848]" [ultimate]
  →[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:
2.1.4 Leje sistemi kartelash

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 run 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 weblate user inside the container (UID 1000).

Shihni edhe:

Serving static files

2.1.5 Ujdisje baze të dhënash për Weblate

It is recommended to run Weblate with a PostgreSQL database server.

Shihni edhe:

Përdorni një mekanizëm të fuqishëm baze të dhënash, 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.

Shënim: Weblate uses trigram extension which has to be installed separately in some cases. Look for `postgresql-contrib` or a similarly named package.

Shihni edhe:

PostgreSQL notes

Krijim i një baze të dhënash në 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
```
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;
```

**Formësim i Weblate-it që të përdorë PostgreSQL-in**

The settings.py snippet for PostgreSQL:

```python
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 dhe MariaDB**

**Ndihmëz:** 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.

```sql
[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
```

**Ndihmëz:** 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.

**Ndihmëz:** In case you are getting #2006 - MySQL server has gone away error, configuring *CONN_MAX_AGE* might help.

### Configuring Weblate to use MySQL/MariaDB

The *settings.py* snippet for MySQL and MariaDB:

```python
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 Formësime të tjera

Formësim i email-eveqëdërgohen

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.

Ndihmëz: 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.

Shihni edhe:
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 Auditim regjistri.

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.

Shihni edhe:
Spam protection, Rate limiting, Auditim regjistri, IP_BEHIND_REVERSE_PROXY, IP_PROXY_HEADER, IP_PROXY_OFFSET, SECURE_PROXY_SSL_HEADER

Ndërmjetës HTTP

Weblate does execute VCS commands and those accept proxy configuration from environment. The recommended approach is to define proxy settings in settings.py:

```python
import os
os.environ["http_proxy"] = "http://proxy.example.com:8080"
oS.environ["HTTPS_PROXY"] = "http://proxy.example.com:8080"
```

Shihni edhe:
Proxy Environment Variables
2.1.7 Përshtatje formësimi

Shihni edhe:

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.

**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"]
```

Ndryshe, mund të përdorni shenjë të gjithëpุtëtshëm:

```
ALLOWED_HOSTS = ["*"]
```

**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 *Aktivizoni përdorim fshehtine*) it is recommended to use it for sessions as well:

```
SESSION_ENGINE = "django.contrib.sessions.backends.cache"
```

**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.

**SECRET_KEY**
Key used by Django to sign some info in cookies, see Django secret key for more info.

Shihni edhe:

SECRET_KEY

SERVER_EMAIL

E-mail used as sender address for sending e-mails to the administrator, for example notifications on failed merges.

Shihni edhe:

SERVER_EMAIL

2.1.8 Mbushje e bazës së të dhënave

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.

Shihni edhe:

Formësim, Listë privilegjes

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 Ndërfaqe adminstrimi.

Shihni edhe:

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.

Shihni edhe:

Përshtatje formësimi

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"),
```

Shihni edhe:

Përshtatje formësimi

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.

Ndryshuar në versionin 4.2: Prior to the 4.2 release the Django sites framework was used instead, please see The “sites” framework.

Shihni edhe:

```
Allowed hosts setup, Formësoni si duhet HTTPS-në SITE_DOMAIN, WEBLATE_SITE_DOMAIN, ENABLE_HTTPS
```

Formësoni si duhet HTTPS-në

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
```

Ndihmëz: You might want to set up HSTS as well, see SSL/HTTPS for more details.

Shihni edhe:

```
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.

| Sinjalizim: Setting this incorrectly can irreversibly (for some time) break your site. Read the HTTP Strict Transport Security documentation first. |

Përdorni një mekanizëm të fuqishëm baze të dhënash

- Please use PostgreSQL for a production environment, see Ujdisje baze të dhënash për 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.

Shihni edhe:

| Ujdisje baze të dhënash për Weblate, Migrating from other databases to PostgreSQL, Përshtatje formësimi, Databases |

Aktivizoni përdorim fshehtine

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",
    },
  },
} |

| Ndihmëz: In case you change Redis settings for the cache, you might need to adjust them for Celery as well, see Background tasks using Celery. |

Shihni edhe:

| Avatar caching, Django’s cache framework |
Avatar caching

In addition to caching of Django, Weblate performs caching of avatars. It is recommended to use a separate, file-backed cache for this purpose:

```python
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,
        },
    },
}
```

Shihni edhe:

`ENABLE_AVATARS`, `AVATAR_URL_PREFIX`, `Avatars`, `Aktivizoni përdorim fshehtine`, Django’s cache framework

Formësoni dërgimi email-esh

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"
```

Shënëm: 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.

Shihni edhe:

`Përshtatje formësimi`, `Formësim i email-eve që dërgohen`, `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`.

Ndihmëz: On Docker container, this is available as `WEBLATE_ALLOWED_HOSTS`. 

2.1. Udhëzime formësimi
Shihni edhe:

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.

Shihni edhe:

SECRET_KEY

Home directory

Ndryshuar ne versionin 2.1: 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:

```python
os.environ["HOME"] = os.path.join(BASE_DIR, "configuration")
```

Shënim: 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.

Shihni edhe:

Hyrje në depo

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):

```python
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": [
```
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.

Ndryshuarnë versionin 3.2: 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.

Vendore dhe kodim sistemi

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:

```bash
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):

```bash
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):

```python
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
```

Ndihmëz: The official Docker image has this feature already enabled.

Shihni edhe:
Common Deployment Scenarios, Serving static files

2.1.10 Running server

Ndihmëz: 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 Ujdisje baze të dhënash për Weblate)
- Cache server (see Aktivizoni përdorim fshehtine)
- 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)
Shënim: 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 heavy loaded, you can split up the services. The only limitation on this is that Celery and Wsgi servers need access to $DATA_DIR$.

Shënim: 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 Leje sistemi kartelash 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 webservers below).

For testing purposes, you can use the built-in web server in Django:

```
weblate runserver
```

Sinjalizim: 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.

Ndihmëz: 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

Ndryshuar në versionin 2.4: 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`.

Shihni edhe:

Sample configuration for NGINX and uWSGI, Sample configuration for Apache, Sample configuration for Apache and Gunicorn, Compressing client assets, Deploying Django, Deploying 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.

Shihni edhe:

- `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`):

```bash
# 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

# 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

Shihni edhe:

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):

```apache
# 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...
#
<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
    WSGIProcessGroup weblate
    WSGIApplicationGroup %{GLOBAL}
    WSGIScriptAlias / /home/weblate/weblate-env/lib/python3.7/site-packages/weblate/wsgi.py process-group=weblate request-timeout=600
    WSGIPassAuthorization On
    <Directory /home/weblate/weblate-env/lib/python3.7/site-packages/weblate/>
        <Files wsgi.py>
            Require all granted
        </Files>
    </Directory>
</VirtualHost>
```

Shënim: 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.

Shihni edhe: HowtouseDjangowithApacheandmod_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>
```

Shihni edhe:

How to use Django with Gunicorn

Running Weblate under path

Të reja në versionin 1.3.

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
#
```

(izindon ne fajen pasuese)
# 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.

```sh
<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
    WSGIProcessGroup weblate
    WSGIApplicationGroup %{GLOBAL}

    WSGIScriptAlias /weblate /home/weblate/weblate-env/lib/python3.7/site-packages/weblate/wsgi.py
    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

Të reja në versionin 3.2.

Weblate uses Celery to process background tasks. 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
```

**Shihni edhe:**

Redis broker configuration in Celery

For development, you might want to use eager configuration, which does process all tasks in place, but this will have performance impact on Weblate:
CELERY_TASK_ALWAYS_EAGER = True
CELERY_BROKER_URL = "memory://"
CELERY_TASK_EAGER_PROPAGATES = True

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
```

Shënim: 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 Leje sistemi kartelash and Running server.

**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`

```ini
[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} \
-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} \
-A ${CELERY_APP} --pidfile=${CELERYD_PID_FILE} \
--logfile=${CELERYD_LOG_FILE} --loglevel=${CELERYD_LOG_LEVEL} $(CELERYD_OPTS)"
ExecReload=/bin/sh -c '"${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"
```

(vazhdon ne faqen pasuese)
# 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

```
CELERYD_PID_FILE="/run/celery/weblate-%n.pid"
CELERYD_LOG_FILE="/var/log/celery/weblate-%n%I.log"
CELERYD_LOG_LEVEL="INFO"
```

# Internal Weblate variable to indicate we're running inside Celery

```
CELERY_WORKER_RUNNING="1"
```

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 use `celery_queues` to see current length of Celery task queues. In case the queue will get too long, you will also get configuration error in the admin interface.

**Sinjalizim:** 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.

**Shihni edhe:**

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.

Shihni edhe:
Webulate 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",  
]

# 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.
Migrim baze të dhënash

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.

Shënimë të Tjera

Don't forget to move other services Weblate might have been using like Redis, Cron jobs or custom authentication backends.

2.2 Sendësrtme Weblate

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

Shënim: 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 Package

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 në 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 commandline 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 upgrade steps integrated. There are no manual steps besides pulling latest version.

2.3.2 Generic upgrade instructions

Before upgrading, please check the current Domosdoshmëri software 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.

Shënim: 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 Kopjeruajtje dhe zhvendosje e 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
   ```

   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

3. Upgrade configuration file, refer to settings_example.py or Version specific instructions for needed steps.

4. Upgrade database structure:

   weblate migrate --noinput

5. Collect updated static files (see Running server and Serving static files):

   weblate collectstatic --noinput

6. Compress JavaScript and CSS files (optional, see Compressing client assets):

   weblate compress

7. 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

8. Verify that your setup is sane (see also Production setup):

   weblate check --deploy

9. 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.

Shihni edhe:
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.

Shihni edhe:
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.

Shihni edhe:

*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.

Shihni edhe:

*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 Klient Weblate 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.

Ndryshuar në versionin 4.3.1:

- The Celery configuration was changed to add memory queue. Please adjust your startup scripts and CELERY_TASK_ROUTES setting.

Ndryshuar në versionin 4.3.2:

- The post_update method of addons now takes extra skip_push parameter.

Shihni edhe:

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 dhe MariaDB.

Ndryshuar në versionin 4.4.1:

- 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.

Shihni edhe:

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.
- Glossaries are now stored as regular components.
- The glossary API is removed, use regular translation API to access glossaries.
- There is a change in `INSTALLED_APPS - weblate.metrics` should be added.

Ndryshuar në versionin 4.5.1:

- There is a new dependency on the `pyahcorasick` module.

**Shihni edhe:**

*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 URL të brendshme Weblate, see POST `/api/projects/(string:project)/components/`.
- There is a change in dependencies and `PASSWORD_HASHERS` to prefer Argon2 for passwords hashing.

**Shihni edhe:**

*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.

**Shihni edhe:**

*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.

Shihni edhe:
*Generic upgrade instructions*

2.3.4 Upgrading from Python 2 to Python 3

Weblate no longer supports Python older than 3.5. 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 dabatase 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.

Krijimin jë baze të dhënash nè PostgreSQL

It is usually a good idea to run Weblate in a separate database, and separate user account:

```bash
# 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*:

```python
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
    }
```

The Weblate Manual, Hedhje Në Qarkullim 4.9

"OPTIONS": {
    # If in case of using an older MySQL server, which has MyISAM as 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": "",
  },
}

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
```

2.3. Upgrading Weblate 207
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 Kopjeruajtje dhe zhvendosje e Weblate

2.4.1 Kopjeruajtje e automatizuar duke përdorur BorgBackup

Të reja në versionin 3.9.

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.

Ndryshuar në versionin 4.4.1: 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:

- Daily backups for 14 days back
- Weekly backups for 8 weeks back
- Monthly backups for 6 months back
Kyç fshehtëzimesh Borg

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.

Shihni edhe:

borg init

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. Backup your Borg credentials to be able to restore the backups, see Kyç fshehtëzimesh Borg.

Ndihmëz: 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.

Shihni edhe:

General in the Borg documentation

Sistem kartelash vendor

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 Leje sistemi kartelash). If it doesn’t exist, Weblate attempts to create it but needs the appropriate permissions to do so.

Ndihmëz: 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.

Kopjeruajtje të largëta

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 Kyç SSH Weblate-i).
6. Configure the backup location in Weblate as user@host:/path/to/backups.

Ndihmëz:  Weblate provisioned backup storage provides you automated remote backups without any effort.

Shihni edhe:

Kyç SSH Weblate-i

2.4.4 Rikthim prej 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 Përshtatje formësimi.
   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 Leje sistemi kartelash.

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...
$ borg extract /tmp/xxx::2019-09-26T14:56:08
Enter passphrase for key /tmp/xxx:
```

Shihni edhe:

borg list, borg extract

2.4. Kopjeruajtje dhe zhvendosje e Weblate
2.4.5 Kopjeruajtje dorazi

Depending on what you want to save, back up the type of data Weblate stores in each respective place.

Ndihmëz:  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")
```

Bazë të dhënash

The actual storage location depends on your database setup.

Ndihmëz:  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()
```

Kartela

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

Ndryshuar në versionin 4.7: 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
```

Depo nën kontroll versionesh

Stored in `DATA_DIR/vcs`.

The version control repositories contain a copy of your upstream repositories with Weblate changes. If you have `Push on commit` 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.

Kyçë SSH dhe GPG

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.

Kartela të ngarkuara nga përdoruesi

Stored in `DATA_DIR/media`.

You should back up all user uploaded files (e.g. `Visual context for strings`).

Akte Celery

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.

Shihni edhe:

`Background tasks using Celery`
Rresht urdhrash për kopjeruajtje dorazi

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 \texttt{XZ_OPT} allows you to choose your xz options, for instance the amount of memory used for compression; see \url{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 Rikthim kopjeruajtje të bërë dorazi

1. Restore all data you have backed up.
2. Update all repositories using \texttt{updatelogit}.

```
weblate updatelogit --all
```

2.4.7 Zhvendosje e një instalimi Weblate

Relocate your installation to a different system by following the backing up and restoration instructions above.

\textbf{Shihni edhe:}

\textit{Upgrading from Python 2 to Python 3, Migrating from other databases to PostgreSQL}

2.5 Mirëfilltësim

2.5.1 Regjistrim përdoruesi

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 \texttt{REGISTRATION_OPEN}.

The authentication attempts are subject to \textit{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 \textit{Migrating from Pootle}).

Django can additionally be set up to authenticate against other means too.

\textbf{Shihni edhe:}

\textit{Rregullime për mirëfilltësim} describes how to configure authentication in the official Docker image.
2.5.3 Mirëfilltësim me rrjetë shoqërore

Thanks to Welcome to Python Social Auth's documentation!, Webate support authentication using many third party services such as GitLab, Ubuntu, Fedora, etc.

Please check their documentation for generic configuration instructions in Django Framework.

**Shënëm:** By default, Webate 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 Webate side by configuring `FORCE_EMAIL_VALIDATION` for them. For example:

```python
SOCIAL_AUTH_OPENSUSE_FORCE_EMAIL_VALIDATION = True
```

**Shihni edhe:**

**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 Webate will not be able to properly credit contributions users make.

**Ndihmëz:** Most of the authentication backends require HTTPS. Once HTTPS is enabled in your web server please configure Webate to report it properly using `ENABLE_HTTPS`, or by `WEBLATE_ENABLE_HTTPS` in the Docker container.

**Shihni edhe:**

**Python Social Auth backend**

**Mirëfilltësim OpenID**

For OpenID-based services it's usually just a matter of enabling them. The following section enables OpenID authentication for OpenSUSE, Fedora and Ubuntu:

```python
# 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",
)
```

**Shihni edhe:**

OpenID
Mirëfilltësim GitHub

You need to register an OAuth application on GitHub and then tell Weblate all its secrets:

```python
# 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/.

Shënim: 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.

Shihni edhe:

GitHub

Mirëfilltësim Bitbucket

You need to register an application on Bitbucket and then tell Weblate all its secrets:

```python
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.bitbucket.BitbucketOAuth",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_BITBUCKET_KEY = "Bitbucket Client ID"
SOCIAL_AUTH_BITBUCKET_SECRET = "Bitbucket Client Secret"
SOCIAL_AUTH_BITBUCKET_VERIFIED_EMAILS_ONLY = True
```

Shënim: 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.

Shihni edhe:

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/

```python
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"
```

Shënim: 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.

Shihni edhe:

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/

```python
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"]
```

Shënim: 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.

Shihni edhe:

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.

```python
# 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/'
```

Shënim: 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.

Shihni edhe: GitLab

Microsoft Azure Active Directory

Weblate can be configured to use common or specific tenants for authentication.


```python
# 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
(vazhdon ne faqen pasuese)
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_KEY = ""
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_SECRET = ""

# Tenant ID
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_TENANT_ID = ""

Shën: 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.

Shihni edhe:

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 = ""

Shën: 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.

Shihni edhe:

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"
Çaktivizim mirëfilltësimesh me fjalëkalim

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.

Ndihmëz: You can still use password authentication for the admin interface, for users you manually create there. Just navigate to `/admin/`.

For example, authentication using only the openSUSE Open ID provider can be achieved using the following:

```python
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.suse.OpenSUSEOpenId",
    "weblate.accounts.auth.WeblateUserBackend",
)
```

2.5.4 Mirëfilltësim me fjalëkalim

The default `settings.py` comes with a reasonable set of `AUTH_PASSWORD_VALIDATORS`:

- Fjalëkalimet s’mund të jetë shumë të ngjashëm me të tjerë të dhëna personale tuajat.
- Fjalëkalimet duhet të përmbajnë të paktën 10 shenja.
- Fjalëkalimet s’mund të jenë fjalëkalim i përdorur rëndom.
- Fjalëkalimet s’mund të jenë tërësisht numerikë.
- Fjalëkalimet s’mund të përmben nga një shenjë e vetme ose nga vetëm një hapësirë.
- Fjalëkalimet s’mund të përputhen me një fjalëkalim që keni përdorur në të kaluarën.

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 Mirëfilltësim SAML

Të reja në versionin 4.1.1.

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`

Formësim shembull:

```python
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.email.EmailAuth",
    "social_core.backends.saml.SAMLAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_SAML_SP_ENTITY_ID = f'https://{SITE_DOMAIN}/accounts/metadata/saml/'
```
The default configuration extracts user details from following attributes, configure your IDP to provide them:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>SAML URI reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Emër i plotë</td>
<td>urn:oid:2.5.4.3</td>
</tr>
<tr>
<td>First name</td>
<td>urn:oid:2.5.4.42</td>
</tr>
<tr>
<td>Last name</td>
<td>urn:oid:2.5.4.4</td>
</tr>
<tr>
<td>Email</td>
<td>urn:oid:0.9.2342.19200300.100.1.3</td>
</tr>
<tr>
<td>Emër përdoruesi</td>
<td>urn:oid:0.9.2342.19200300.100.1.1</td>
</tr>
</tbody>
</table>

Ndihmëz: The example above and the Docker image define an IDP labelled `weblate`. You might need to configure this string as `Relay` in your IDP.

Shihni edhe:

*Configuring SAML in Docker, SAML*

### 2.5.6 Mirëfilltësim LDAP

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
```

Ndihmëz: This package is included in the Docker container, see *Installing using Docker*. 

---

2.5. Mirëfilltësim
Shënimi: 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:

```python
# 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
```

Shënimi: 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 cannot use direct bind for authentication, you will need to use search, and provide a user to bind for the search. For example:

```python
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)"
)
```
Integrim me Active Directory

```python
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": "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
```

Shihni edhe:

Django Authentication Using LDAP, Authentication

2.5.7 Mirëfilltësim CAS

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:

```python
# 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)

```python
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()
```

Shihni edhe:
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.

Shihni edhe:

*Mirëfilltësim LDAP, Mirëfilltësim CAS*

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`:

```python
AUTHENTICATION_BACKENDS = (
    # Add authentication backend here
    "weblate.accounts.auth.WeblateUserBackend",
)
INSTALLED_APPS += (
    # Install authentication app here
)
```

### 2.6 Kontroll hyrjesh

Weblate comes with a fine-grained privilege system to assign user permissions for the whole instance, or in a limited scope.

Ndryshuar në versionin 3.0: Before Weblate 3.0, the privilege system was based on Django privilege system only, but is specifically built for Weblate now. If using anything older, please consult the documentation for the specific version you are using.
2.6.1 Simple access control

If you are not administrating the whole Weblate installation and just have access to manage certain projects (like on Hosted Weblate), your access control management options are limited to following settings. If you don’t need any complex setup, those are sufficient for you.

**Kontroll hyrjesh te projekti**

**Shënim:** This feature is unavailable for the projects running Libre plan on Hosted Weblate.

You can limit user’s access to individual projects by selecting a different *Access control* setting. Available options are:

- **Publik** Publicly visible, translatable for all logged-in users.
- **I mbrojtur** Publicly visible, but translatable only for selected users.
- **Privat** Visible and translatable only for selected users.

**Vetjake** *User management* features will be disabled; by default all users are forbidden to perform any actions on the project. You will have to set up all the permissions using *Custom access control*.

*Access control* can be changed in the *Access* tab of the configuration (Manage ↓ Settings) of each respective project.

![Weblate Access Control](image)

The default value can be changed by `DEFAULT_ACCESS_CONTROL`.

**Shënim:** Even for *Private* projects, some info about your project will be exposed: statistics and language summary for the whole instance will include counts for all projects despite the access control setting. Your project name and other information can’t be revealed through this.

**Shënim:** The actual set of permissions available for users by default in *Public*, *Protected*, and *Private* projects can be redefined by Weblate instance administrator using *custom settings*.

**Sinjalizim:** By turning on *Custom* access control, Weblate will remove all *special groups* it has created for a selected project. If you are doing this without admin permission for the whole Weblate instance, you will instantly lose your access to manage the project.
Shihni edhe:

Kontroll hyrjesh

Managing per-project access control

Users with the Manage project access privilege (see Listë privilegjesh) can manage users in projects with non-Custom access control. They can assign users to one of the following groups.

For Public, Protected and Private projects:

- **Administrim** Includes all permissions available for the project.
- **Review** (only if review workflow is turned on) Can approve translations during review.

For Protected and Private projects only:

- **Përktheni** Can translate the project and upload translations made offline.
- **Burime** Can edit source strings (if allowed in the project settings) and source string info.
- **Gjuhë** Can manage translated languages (add or remove translations).
- **Fjalorth** Can manage glossary (add or remove entries, also upload).
- **Kujtesë** Can manage translation memory.
- **Foto ekrani** Can manage screenshots (add or remove them, and associate them to source strings).
- **VCS** Can manage VCS and access the exported repository.
- **Faturim** Can access billing info and settings (see Faturim).

Unfortunately, it's not possible to change this predefined set of groups for now. Also this way it's not possible to give just some additional permissions to all users.

Shënim: For non-Custom access control an instance of each group described above is actually defined for each project. The actual name of those groups will be Project@Group, also displayed in the Django admin interface this way. Although they can't be edited from Weblate user-interface.
These features are available on the Access control page, which can be accessed from the project’s menu Manage → Users.

New user invitation

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 group) would be sufficient.

Ndhëmëz: 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.

Të reja në versionin 3.11: 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.
Blocking users

Të reja në versionin 4.7.

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.

Per-project permission management

You can set your projects to Protected or Private, and manage users per-project in the Weblate user interface.

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 group of the project.

2.6.2 Custom access control

Shënim: This feature is unavailable for the projects running Libre plan on Hosted Weblate.

The permission system is based on groups and roles, where roles define a set of permissions, and groups link them to users and translations, see Users, roles, groups, and permissions for more details.

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 Simple access control section.

Ujdisje të rëndomta

This section contains an overview of some common configurations you may be interested in.

Site-wide permission management

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 Kontroll hyrjesh te projektit), 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).
Custom permissions for languages, components or projects

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.

Example:

If you want (for whatever reason) to allow translation to a specific language (let’s 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.

   Tabela 1: Group Users

<table>
<thead>
<tr>
<th>Gjuhë</th>
<th>As defined</th>
</tr>
</thead>
<tbody>
<tr>
<td>Përzgjedhje gjuhe</td>
<td></td>
</tr>
<tr>
<td>Czuhi</td>
<td>All but Czech</td>
</tr>
</tbody>
</table>

2. Add a dedicated group for Czech translators.

   Tabela 2: Group Czech translators

<table>
<thead>
<tr>
<th>Role</th>
<th>Power users</th>
</tr>
</thead>
<tbody>
<tr>
<td>Përzgjedhje projekti</td>
<td>All public projects</td>
</tr>
<tr>
<td>Përzgjedhje gjuhe</td>
<td>As defined</td>
</tr>
<tr>
<td>Gjuhë</td>
<td>Czech</td>
</tr>
</tbody>
</table>

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.

Users, roles, groups, and permissions

The authentication models consist of several objects:

Leje Individual permission defined by Weblate. Permissions cannot be assigned to users. This can only be done through assignment of roles.

Rol A role defines a set of permissions. This allows reuse of these sets in several places, making the administration easier.

Përdorues User can belong to several groups.

Grup Group connect roles, users, and authentication objects (projects, languages, and component lists).
**Shënim:** A group can have no roles assigned to it, in that case access to browse the project by anyone is assumed (see below).

**Access for browse to a project**

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 Listë grupesh).

**Access for browse to a component**

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).

**Scope of groups**

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 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.
Ndihmëz: Use Language selection or Project selection to automate inclusion of all languages or projects.

Example:

Let’s say there is a project foo with the components: foo/bar and foo/baz and the following group:

<table>
<thead>
<tr>
<th>Role</th>
<th>Review Strings, Manage repository</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perbëres</td>
<td>foo/bar</td>
</tr>
<tr>
<td>Gjuhë</td>
<td>Spanish</td>
</tr>
</tbody>
</table>

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.

Caktimi automatizuar grupesh

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.

Shënim: 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).

Shënim: 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.

Role dhe grupe parazgjedhje

After installation, a default set of groups is created (see Listë grupesh).

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.
Listë privilegjesh

Faturim (shihni Faturim)  View billing info [Administration, Billing]

Ndryshime  Download changes [Administration]

Komentë  Post comment [Administration, Edit source, Power user, Review strings, Translate]

   Të fshijë komentin [Administrim]

Përmbërës  Të përpunojë rregullime përbërës [Administrim]

   Lock component, preventing translations [Administration]

Fjalorth  Add glossary entry [Administration, Manage glossary, Power user]

   Edit glossary entry [Administration, Manage glossary, Power user]

   Delete glossary entry [Administration, Manage glossary, Power user]

   Upload glossary entries [Administration, Manage glossary, Power user]

Sugjerime të automatizuara  Use automatic suggestions [Administration, Edit source, Power user, Review strings, Translate]

Kujtetë përktimesh  Edit translation memory [Administration, Manage translation memory]

   Delete translation memory [Administration, Manage translation memory]

Projekte  Të përpunojë rregullime projekti [Administrim]

   Të administrojë hyrje te projekti [Administrim]

Raportime  Të shkarkojë raporte [Administrim]

Foto ekranë  Add screenshot [Administration, Manage screenshots]

   Edit screenshot [Administration, Manage screenshots]

   Delete screenshot [Administration, Manage screenshots]

Vargjet e burimit  Edit additional string info [Administration, Edit source]

Vargje  Add new string [Administration]

   Remove a string [Administration]

   Dismiss failing check [Administration, Edit source, Power user, Review strings, Translate]

   Edit strings [Administration, Edit source, Power user, Review strings, Translate]

   Review strings [Administration, Review strings]

   Edit string when suggestions are enforced [Administration, Review strings]

   Edit source strings [Administration, Edit source, Power user]

Sugjerime  Accept suggestion [Administration, Edit source, Power user, Review strings, Translate]

   Add suggestion [Administration, Edit source, Add suggestion, Power user, Review strings, Translate]

   Delete suggestion [Administration, Power user]

   Vote on suggestion [Administration, Edit source, Power user, Review strings, Translate]

Përktihime  Add language for translation [Administration, Power user, Manage languages]

   Perform automatic translation [Administration, Manage languages]

   Delete existing translation [Administration, Manage languages]

   Add several languages for translation [Administration, Manage languages]
Ngarkime  Define author of uploaded translation [Administration]
  Overwrite existing strings with upload [Administration, Edit source, Power user, Review strings, Translate]
  Upload translations [Administration, Edit source, Power user, Review strings, Translate]

VCS  Access the internal repository [Administration, Access repository, Power user, Manage repository]
  Commit changes to the internal repository [Administration, Manage repository]
  Push change from the internal repository [Administration, Manage repository]
  Reset changes in the internal repository [Administration, Manage repository]
  View upstream repository location [Administration, Access repository, Power user, Manage repository]
  Update the internal repository [Administration, Manage repository]

Privilegje në mbarë sajtin  Të përdorë ndërfaqe administrimi
  Të shtojë projekte të rinj
  Të shtojë përkufizime gjuhësh
  Të administrojë përkufizime gjuhësh
  Të administrojë grupe
  Të administrojë përdorues
  Të administrojë role
  Të administrojë lajmërime
  Të administrojë kujtesë përkthimesh
  Të administrojë listë përbërësh

Shënimi: Site-wide privileges are not granted to any default role. These are powerful and quite close to superuser status. Most of them affect all projects in your Weblate installation.

Listë grupesh

The following groups are created upon installation (or after executing setupgroups) and you are free to modify them. The migration will, however, re-create them if you delete or rename them.

Vizitorë  Defines permissions for non-authenticated users.
  This group only contains anonymous users (see ANONYMOUS_USER_NAME).
  You can remove roles from this group to limit permissions for non-authenticated users.
  Default roles: Add suggestion, Access repository

Parës  This role ensures visibility of public projects for all users. By default, all users are members of this group.
  By default, automatic group assignment makes all new accounts members of this group when they join.
  Role parazgjedhje: asnjë

Përdorues  Grup parazgjedhje për krejt përdoruesit.
  By default, automatic group assignment makes all new accounts members of this group when they join.
  Role parazgjedhje: Përdorues i regjull

Shqyrtes  Group for reviewers (see Rrjedha pune përkthimi).
  Role parazgjedhje: Të shqyrtojë vargje
**Managers** Grup për përgjegjësit.

Role parazgjedhje: *Administr*

**Sinjalizim:** Never remove the predefined Weblate groups and users as this can lead to unexpected problems! If you have no use for them, you can removing all their privileges instead.

### 2.6.3 Additional access restrictions

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 logging-in to access all the site pages. This is basically the way to lock your Weblate installation.

**Ndihmëz:** You can use built-in *invitations* to add new users.

### 2.7 Projekte përktimi

#### 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 *File mask* 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 *Formate të mbuluar kartelash*.

**Shënshm:** You can share cloned VCS repositories using *URL të brendshme Weblate*. 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

Ndryshuar në versionin 3.2: An interface for adding projects and components is included, and you no longer have to use `Ndërfaqe përgjegjësi Django`.

Ndryshuar në versionin 3.4: 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/](https://hosted.weblate.org/) see *Faturim*), 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:
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:
After creating the project, you are taken directly to the project page:

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:

- **Prej kontrolli versionesh**  Creates component from remote version control repository.
- **Prej përbërësi ekzistues**  Creates additional component to existing one by choosing different files.
- **Degë shtesë**  Creates additional component to existing one, just for different branch.
- **Ngarkoni kartela përkthimesh**  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.
- **Përktheni dokument**  Upload single document or translation file and translate that.
- **Nisjani nga hiçi**  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:
On the next page, you are presented with a list of discovered translatable resources:

As a last step, you review the translation component info and fill in optional details:
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 Kujtesë Përkthimesh.

Shihni edhe:
/devel/integration

These basic attributes set up and inform translators of a project:

**Emër projekti**

Verbose project name, used to display the project name.

**Identifikues URL-je**

Project name suitable for URLs.

**Sajt projekti**

URL where translators can find more info about the project.

This is a required parameter unless turned off by `WEBSITE_REQUIRED`.

**Udhëzime përkthimi**

Text describing localization process in the project, and any other information useful for translators. Markdown can be used for text formatting or inserting links.

**Caktoni krye «Language-Team»**

Whether Weblate should manage the `Language-Team` header (this is a GNU `gettext` only feature right now).

**Përdor kujtesë përkthimesh tê përbashkët**

Whether to use shared translation memory, see Kujtesë përkthimesh e përbashkët for more details.

Default value is determined by `DEFAULT_SHARED_TM`. 

Shihni edhe:
Ndërfaqe përgjegjësi Django, Project configuration, Component configuration
**Kontribuoni te kujtesë përkthimesh të përbashkët**

Whether to contribute to shared translation memory, see *Kujtesë përkthimesh e përbashkët* for more details. Default value is determined by `DEFAULT_SHARED_TM`.

**Kontroll hyrjesh**

Configure per project access control, see *Kontroll hyrjesh te projektit* for more details. Default value can be changed by `DEFAULT_ACCESS_CONTROL`.

**Aktivizoni shqyrtime**

Enable review workflow for translations, see *Shqyrtues enkas*.

**Aktivizo shqyrtme burimi**

Enable review workflow for source strings, see *Shqyrtim vargjesh burimi*.

**Shihni edhe:**

- `report-source`, *Komente*

**Aktivizoni hook-e**

Whether unauthenticated *Hook-e njofimesh* are to be used for this repository.

**Shihni edhe:**

- *Kartelë gjite ndërmjetëse, Quality gateway for the source strings, Formate dygjuhësh dhe njëgjuhësh, Language definitions*

**Aliase gjuhe**

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`

**Ndihmëz:** 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.

**Shihni edhe:**

*Parsing language codes*
### 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.

**Shihni edhe:**
/devel/integration

You can find some examples of typical configurations in the *Format e mbuluar kartela**.

**Shënim:** 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 addons, 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:

**Emër përmbërësi**
Verbosc component name, used to display the component name.

**Component slug**
Component name suitable for URLs.

**Component project**

*Project configuration* where the component belongs.

**Sistem kontrolli versionesh**

VCS to use, see *Integrim kontrolli versioni* for details.

**Shihni edhe:**

*Pushing changes from Weblate*

**Depo kodi burim**

VCS repository used to pull changes.

**Shihni edhe:**

See *Hyrje në depo* for more details on specifying URLs.

**Ndihmëz:** This can either be a real VCS URL or *weblate://project/component* indicating that the repository should be shared with another component. See *URL i brendshme Weblate* for more details.
URL push-i për depon

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.

Shihni edhe:
See Hyrje në depo for more details on how to specify a repository URL and Pushing changes from Weblate for more details on pushing changes from Weblate.

Shfletues depoje

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}}

URL depoje të eksportuar

URL where changes made by Weblate are exported. This is important when Përkthim në vazhdimësi is not used, or when there is a need to manually merge changes. You can use Git exporter to automate this for Git repositories.

Degë depoje

Which branch to checkout from the VCS, and where to look for translations.

Degë push

Branch for pushing changes, leave empty to use Degë depoje.

Shënimi: This is currently only supported for Git, GitLab and GitHub, it is ignored for other VCS integrations.

Shihni edhe:
Pushing changes from Weblate

File mask

Mask of files to translate, including path. It should include one «*» replacing language code (see Language definitions 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 [ ]].

Shihni edhe:
Formate dygjuhësh dhe njëgjuhësh, What does mean «There are more files for the single language (en)»?
Kartelë njëgjuhëshe gjuhe bazë

Base file containing string definitions for Përbërës njëgjuhësh.

Shihni edhe:
*Format dhe njëgjuhësh, What does mean “There are more files for the single language (en)”?

Përpunoni kartelën bazë

Whether to allow editing the base file for Përbërës njëgjuhësh.

Kartelë gjuhe ndërmjetëse

Intermediate language file for Përbërës njëgjuhësh. 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 Kartelë njëgjuhëshe gjuhe bazë. 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.

Shihni edhe:
*Quality gateway for the source strings, Formate dygjuhësh dhe njëgjuhësh, What does mean «There are more files for the single language (en)”?

Gjedhe për përkthime të reja

Base file used to generate new translations, e.g. *.pot file with gettext.

Ndihmëz: In many monolingual formats Weblate starts with blank file by default. Use this in case you want to have all strings present with empty value when creating new translation.

Shihni edhe:
*adding-translation, Shtim për kthimesh të reja, Shtim për kthimi të ri, Formate dygjuhësh dhe njëgjuhësh, What does mean «There are more files for the single language (en)”?

Format kartelash

Translation file format, see also Formate të mbuluar kartelash.

Adresë njoftimi të metash në vargje burim

Email address used for reporting upstream bugs. This address will also receive notification about any source string comments made in Weblate.
Lejo përhapje përkkthimesh

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`.

Shihni edhe:

`Keeping translations same across components`

Enable suggestions

Whether translation suggestions are accepted for this component.

Votim sugjerimesh

Turns on vote casting for suggestions, see `Votim sugjerimesh`.

Vetëprano sugjerime

Automatically accept voted suggestions, see `Votim sugjerimesh`.

Flamuj përkkthimi

Customization of quality checks and other Weblate behavior, see `Customizing behavior using flags`.

Kontrolle me detyrim

List of checks which cannot be ignored, see `Enforcing checks`.

Shënimi: Enforcing the check does not automatically enable it, you still should enabled it using `Customizing behavior using flags` in `Flamuj përkkthimi` or `Additional info on source strings`.

Licencë përkkthimi

License of the translation (does not need to be the same as the source code license).

Marrëveshje kontribuesi

Marrëveshje përdoruesi që duhet miratuar përpara se një përdorues të mund të përktjejë këtë përbërës.
Shtim përkthimi të ri

How to handle requests for creation of new languages. Available options:

**Lidhuni me mirëmbajtësit** 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.

**Shpjer te URL udhëzimesh përkthimi** 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).

**Krijoni kartelë gjuhe të re** User can select language and Weblate automatically creates the file for it and translation can begin.

**Çaktivizo shtim përkthimesh të reja** There will be no option for user to start new translation.

**Ndihmëz:** The project admins can add new translations even if it is disabled here when it is possible (either **Gjedhe për përkthime të reja** or the file format supports starting from an empty file).

Shihni edhe: **adding-translation, Shtim përkthimesh të reja**

**Administroni vargje**

Të reja në versionin 4.5.

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.

Shihni edhe: **Formate dygjuhësh dhe njëgjuhësh, adding-new-strings, POST /api/translations/(string:project)/(string:component)/(string:language)/units/**

**Stilk kod gjuhe**

Përshtatni kod gjuhe të përdorur për të prodhuar emrin e kartelës për përkthime të krijuar nga Weblate.

Shihni edhe: **Shtim përkthimesh të reja, Kod gjuhe, Parsing language codes**

---

2.7. Projekte përkthimi
Stil përzierjeje

You can configure how updates from the upstream repository are handled. This might not be supported for some VCSs. See Merge or rebase for more details.

Default value can be changed by DEFAULT_MERGE_STYLE.

Commit, add, delete, merge and addon 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.

Push on commit

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.

Moshë ndryshimesh për depozitim

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.

Ndihmëz: There are other situations where pending changes might be committed, see Lazy commits.

Kyçe, në rast gabimesh

Locks the component (and linked components, see URL të brendshme Weblate) 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.

Gjuhë burimi

Language used for source strings. Change this if you are translating from something else than English.

Ndihmëz: 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 Kartelë gjuhe ndërmjetëse.
Filtër gjuhësh

Regular expression used to filter the translation when scanning for filemask. It can be used to limit the list of languages managed by Weblate.

Shënimi: You need to list language codes as they appear in the filename.

Some examples of filtering:

<table>
<thead>
<tr>
<th>Filter description</th>
<th>Shprehje e rregullt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selected languages only</td>
<td>^(cs</td>
</tr>
<tr>
<td>Exclude languages</td>
<td>^(?!([it</td>
</tr>
<tr>
<td>Filter two letter codes only</td>
<td>^..$</td>
</tr>
<tr>
<td>Exclude non language files</td>
<td>^(?!(blank)$)+$</td>
</tr>
<tr>
<td>Include all files (default)</td>
<td>^^[^.]+$</td>
</tr>
</tbody>
</table>

Shprehje e rregullt variantesh

Regular expression used to determine the variants of a string, see variants.

Shënimi: Most of the fields can be edited by project owners or managers, in the Weblate interface.

Shihni edhe:

Does Weblate support other VCSes than Git and Mercurial? alerts

Përparësi

Përberësitë më të lartë u ofrohen të parat përkthyesve.

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.

Ndihmëzë: This applies to project admins as well — please make sure you will not loose access to the component after toggling the status.
**Ndaje te projekte**

You can choose additional projects where the component will be visible. Useful for shared libraries which you use in several projects.

**Shënim:** 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.

**Përdore si fjalorth**

Të reja në versionin 4.5.

Allows using this component as a glossary. You can configure how it will be listed using *Ngjyrë fjalorthi*.

The glossary will be accessible in all projects defined by *Ndaje te projekte*.

It is recommended to enable *Administroni vargje* on glossaries in order to allow adding new words to them.

**Shihni edhe:**

*Fjalorth*

**Ngjyrë fjalorthi**

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 addons
  - Pikasje përmbërësisht
  - Prodhues statistikash
  - Executing scripts from add-on

There following variables are available in the component templates:

- `{{ language_code }}` Kod gjuhe
- `{{ language_name }}` Emër gjuhe
- `{{ component_name }}` Emër përmbërësi
- `{{ component_slug }}` Component slug
- `{{ project_name }}` Emër projekti
- `{{ project_slug }}` Project slug
- `{{ url }}` Translation URL
- `{{ filename }}` Emër kartele përkrithimi
- `{{ stats }}` Translation stats, this has further attributes, examples below.
- `{{ stats.all }}` Total strings count
- `{{ stats.fuzzy }}` Count of strings needing review
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|stripext }}`
- 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*)
- **Aktivizoni përdorim fshehtine**
- **Përdorni një mekanizëm të fuqishëm baze të dhëna**
- 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 Pikasje përbërësish addon.

To use the addon, 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 addon 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.

Shihni edhe:

Management commands, Pikasje përbërësish

2.8 Language definitions

To present different translations properly, info about language name, text direction, plural definitions and language code is needed.

2.8.1 Parsing language codes

While parsing translations, Weblate attempts to map language code (usually the ISO 639-1 one) to any existing language object.

You can further adjust this mapping at project level by Aliase gjuhe.

If no exact match can be found, an attempt will be made to best fit it into an existing language. Following steps are tried:

- Case insensitive lookups.
- Normalizing underscores and dashes.
- Looking up built-in language aliases.
- Looking up by language name.
- Ignoring the default country code for a given language—choosing cs instead of cs_CZ.
Should that also fail, a new language definition will be created using the defaults (left to right text direction, one plural). The automatically created language with code xx_XX will be named as xx_XX (generated). You might want to change this in the admin interface later, (see Changing language definitions) and report it to the issue tracker (see Dhënie ndihmese te Weblate), so that the proper definition can be added to the upcoming Weblate release.

**Ndihmëz:** In case you see something unwanted as a language, you might want to adjust **Filët gjuhësh** to ignore such file when parsing translations.

**Shihni edhe:**

*Kod gjuhe, Shtim përkthimesh të reja*

### 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 Built-in language definitions

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 disable using **UPDATE_LANGUAGES**. You can also enforce updating the database to match Weblate built-in data using **setuplang**.

**Shihni edhe:**

*Extending built-in language definitions*

### 2.8.4 Ambiguous language codes and macrolanguages

In many cases it is not a good idea to use macro language 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 macro languages.

**Shihni edhe:**

*Macrolanguages definition, List of macrolanguages*

### 2.8.5 Language definitions

Each language consists of following fields:
Kod gjuhe

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.

Shihni edhe:
Parsing language codes, Shtrim përkthimesh të reja

Emër gjuhe

Visible name of the language. The language names included in Weblate are also being localized depending on user interface language.

Drejtim teksti

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.

Formulë shumësi

Gettext compatible plural formula used to determine which plural form is used for given count.

Shihni edhe:
Shumësa, GNU gettext utilities: Plural forms, Language Plural Rules by the Unicode Consortium

2.8.6 Shtim përkthimesh të reja

Ndryshuar në versionin 2.18: 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 Burime vargjesh Android), while others expect to have all keys present (for example GNU gettext). 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 Kartela JSON).

When you specify Gjedhe për përkthime të reja in Component configuration, Weblate will use this file to start new translations. Any existing translations will be removed from the file when doing so.

When Gjedhe për përkthime të reja is empty and the file format supports it, an empty file is created where new strings will be added once they are translated.

The Stil kodi gjuhe allows you to customize language code used in generated filenames:

Parazgjedhje e bazuar te formati i kartelave Dependent on file format, for most of them POSIX is used.

Në stil POSIX, me përdorim të nënvijës së ndarës Typically used by gettext and related tools, produces language codes like pt_BR.

Në stil POSIX, me përdorim të nënvijës së ndarës, përftshi kod vendi POSIX style language code including the country code even when not necessary (for example cs_CZ).

Në stil BCP, me përdorim të vijës në mes së ndarës Typically used on web platforms, produces language codes like pt-BR.
The Weblate Manual, Hedhje Në Qarkullim 4.9

Në stil BCP, me përdorim të vijës në mes si ndarës, përfshi kod vendi  BCP style language code including the country code even when not necessary (for example cs-CZ).

Në stil Android  Only used in Android apps, produces language codes like pt-br.

Në stil Java  Used by Java—mostly BCP with legacy codes for Chinese.

Additionally, any mappings defined in Aliase gjuhe are applied in reverse.

Shënim:  Weblate recognizes any of these when parsing translation files, the above settings only influences how new files are created.

Shihni edhe:

Kod gjuhe, Parsing language codes

2.9 Përkthim në vazhdimësi

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.

Shihni edhe:

/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 Përditësim deposh.
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 Përditësim deposh

You should set up some way of updating backend repositories from their source.

- Use *Hook-e njofimesh* 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 *Klient Weblate*

- Enable *AUTO_UPDATE* to automatically update all components on your Weblate instance

- Execute *updatedgit* (with selection of project or *--all* to update all)

Whenever Weblate updates the repository, the post-update addons will be triggered, see *Shtesa.*
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
# Push changes to upstream repository
# 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
```

**Shënim:** The example uses Klient Weblate, 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.

**Shihni edhe:**

Klient Weblate

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:
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).

Shihni edhe:

**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/).
The Weblate Manual, Hedhje Në Qarkullim 4.9

Shihni edhe:

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/).

Shihni edhe:

POST /hooks/gitlab/, Accessing repositories from Hosted Weblate

Automatically receiving changes from Pagure

Të reja në versionin 3.3.

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:
Shihni edhe:

POST /hooks/pagure/, Accessing repositories from Hosted Weblate

Automatically receiving changes from Azure Repos

Të reja në version 3.8.

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.

Shihni edhe:

Web hooks in Azure DevOps manual, POST /hooks/azure/, Accessing repositories from Hosted Weblate
Automatically receiving changes from Gitea Repos

Të reja në versionin 3.9.

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.

Shihni edhe:
Webhooks in Gitea manual, POST /hooks/gitea/, Accessing repositories from Hosted Weblate

Automatically receiving changes from Gitee Repos

Të reja në versionin 3.9.

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.

Shihni edhe:
Webhooks in Gitee manual, POST /hooks/gitee/, Accessing repositories from Hosted Weblate

Përditësim i automatizuar i përnatshëm i depove

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 URL push-i për depon), 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 Push on commit). 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 Integrin kontrolli versioni used, more details are found in that chapter.

In case you do not want direct pushes by Weblate, there is support for GitHub, GitLab, Pagure pull requests or Gerrit reviews, you can activate these by choosing GitHub, GitLab, Gerrit or Pagure as Sistem kontrolli versionesh in Component configuration.

Overall, following options are available with Git, GitHub and GitLab:

<table>
<thead>
<tr>
<th>Ujdisje e dëshiruar</th>
<th>Sistem kontrolli versionesh</th>
<th>URL push-i për depon</th>
<th>Degë push</th>
</tr>
</thead>
<tbody>
<tr>
<td>No push</td>
<td>Git</td>
<td>empty</td>
<td>empty</td>
</tr>
<tr>
<td>Push directly</td>
<td>Git</td>
<td>URL SSH</td>
<td>empty</td>
</tr>
<tr>
<td>Push to separate branch</td>
<td>Git</td>
<td>empty</td>
<td>empty</td>
</tr>
<tr>
<td>GitHub pull request from fork</td>
<td>GitLab</td>
<td>URL SSH</td>
<td>Emër dege</td>
</tr>
<tr>
<td>GitHub pull request from branch</td>
<td>GitHub</td>
<td>empty</td>
<td>empty</td>
</tr>
<tr>
<td>GitLab merge request from fork</td>
<td>GitLab</td>
<td>empty</td>
<td>empty</td>
</tr>
<tr>
<td>GitLab merge request from branch</td>
<td>GitLab</td>
<td>URL SSH¹</td>
<td>Emër dege</td>
</tr>
<tr>
<td>Pagure merge request from fork</td>
<td>Pagure</td>
<td>empty</td>
<td>empty</td>
</tr>
<tr>
<td>Pagure merge request from branch</td>
<td>Pagure</td>
<td>URL SSH¹</td>
<td>Emër dege</td>
</tr>
</tbody>
</table>

¹ Can be empty in case Depo kodi burimi supports pushing.
You can also enable automatic pushing of changes after Weblate commits, this can be done in *Push on commit*.

**Shënim:** See *Hyrjenë depo* for setting up SSH keys, and *Lazy commits* for info about when Weblate decides to commit changes.

**Degë të mbrojtura**

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:

![GitHub repository configuration](image)

2.9.3 **Merge or rebase**

By default, Weblate merges the upstream repository into its own. This is the safest way in case you also access the underlying repository by other means. In case you don’t need this, you can enable rebasing of changes on upstream, which will produce a history with fewer merge commits.

**Shënim:** Rebasing can cause you trouble in case of complicated merges, so carefully consider whether or not you want to enable them.
2.9.4 Ndërveprim me të tjerët

Weblate makes it easy to interact with others using its API.

Shihni edhe:

Weblate’s REST API

2.9.5 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 Listë privilegjesh).

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 Moshë ndryshimesh për depozitim on Component configuration.

Ndihmëz: Commits are created for every component. So in case you have many components you will still see lot of commits. You might utilize Kryej «squash» për depozitime Git addon 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:

```python
CELERY_BEAT_SCHEDULE = {
    "commit": {
        "task": "weblate.trans.tasks.commit_pending",
        # Unconditionally commit all changes every 2 minutes
        "hours": 0,
        # How frequently to execute the job in seconds
        "schedule": 120,
    }
}
```

2.9.6 Processing repository with scripts

The way to customize how Weblate interacts with the repository is Shtesa. Consult Executing scripts from add-on for info on how to execute external scripts through addons.

2.9.7 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.
Përhapje përkthimesh

With Lejo përhapje përkthimesh 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.

Shënëm: 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 Jo i njërrajshëm check fires whenever the strings are different. You can utilize this to review such differences manually and choose the right translation.

Përkthim i automatizuar

Automatic translation based on different components can be way to synchronize the translations across components. You can either trigger it manually (see Përkthim i automatizuar) or make it run automatically on repository update using addon (see Përkthim i automatizuar).

2.10 Licencim përkthimesh

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 Të dhëna licence

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 Marrëveshje kontribuesi

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:
The entered text is formatted into paragraphs and external links can be included. HTML markup can not be used.

2.10.3 Licenca përdoruesi

Any user can review all translation licenses of all public projects on the instance from their profile:

2.10. Licencim përkrithëshëm
2.11 Procesi i përkthimit

2.11.1 Votim sugjerimesh

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).

Shënim: 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.
Strings prioritization

Të reja në version 2.0.

String priority can be changed to offer higher priority strings for translation earlier by using the `priority` flag.

**Ndihmëz**: This can be used to order the flow of translation in a logical manner.

**Shihni edhe:**

*Kontrolle cilësie*

Flamuj përkrthimi

Të reja në version 2.4.

Ndryshuar në versionin 3.3: Previously called *Quality checks flags*, it no longer configures only checks.

The default set of translation flags is determined by the translation *Component configuration* and the translation file. However, you might want to use it to customize this per source string.

**Shihni edhe:**

*Kontrolle cilësie, Customizing behavior using flags*

Shpjegim

Ndryshuar në versionin 4.1: 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

Të reja në versionin 2.9.

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:

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:
2.11. Procesi i përkthimit
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 `AUTOFIX_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`:

```python
# Copyright © 2012 - 2021 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, addons and auto-fixes.

2.12.2 Customizing behavior using flags

You can fine-tune the behavior of Weblate (mostly checks) for each source string (in source strings review, see Additional info on source strings) or in the Component configuration (Flamuj përkthimi). Some file formats also allow to specify flags directly in the format (see Formate të mbuluar kartelash).

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:.*
```

Here is a list of flags currently accepted:
rst-text  Treat a text as an reStructuredText document, affects Përkthim i pandryshuar.
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 Vargje vetëm për lexim.
priority:N  Priority of the string. Higher priority strings are presented first for translation. The default priority is 100, the higher the 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 Gjatësi maksimum e përkthimit.
xml-text  Treat text as XML document, affects Sintaksë XML and Markup XML.
font-family:NAME  Define font-family for rendering checks, see Administrim shkronjash.
font-weight:WEIGHT  Define font-weight for rendering checks, see Administrim shkronjash.
font-size:SIZE  Define font-size for rendering checks, see Administrim shkronjash.
font-spacing:SPACING  Define letter spacing for rendering checks, see Administrim shkronjash.
placeholders:NAME:NAME2:...  Placeholder strings expected in translation, see Vendmbajëse.
replacements:FROM:TO:FROM2:TO2...  Replacements to perform when checking resulting text parameters (for example in Madhësi maksimum e përkthimit or Gjatësi maksimum e përkthimit). 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  Shpërrijë e rregullt kundrejt të cilës duhen kërkuar përputhje te kartela përkthimi, shihni Shprehje e rregullt.
forbidden  Indicates forbidden translation in a glossary, see Forbidden translations.
strict-same  Make «Unchanged translation» avoid using built-in words blacklist, see Përkthim i pandryshuar.
check-glossary  Enable the S'ndjek fjalorthin quality check.
angularjs-format  Enable the Varg ndërshtënje AngularJS quality check.
c-format  Enable the Format C quality check.
c-sharp-format  Enable the Format C# quality check.
es-format  Enable the ECMA Script template literals quality check.
i18next-interpolation  Enable the Ndërshtënje i18nexa quality check.
java-format  Enable the Format Java quality check.
java-messageformat  Enable the Java MessageFormat quality check.
javascript-format  Enable the Format JavaScript quality check.
lua-format  Enable the Format Lua quality check.
object-pascal-format  Enable the Object Pascal format quality check.
percent-placeholders  Enable the Vendmbajëse përqindjeje quality check.
perl-format  Enable the Format Perl quality check.
php-format  Enable the Format PHP quality check.
python-brace-format  Enable the Format kllapash Python quality check.
python-format  Enable the Format Python quality check.
qt-format  Enable the Format Qt quality check.
qt-plural-format  Enable the Format shumësi Qt quality check.
ruby-format  Enable the Format Ruby quality check.
scheme-format  Enable the Format Schema quality check.
**vue-format** Enable the *Formatin i18n Vue* quality check.

**md-text** Treat text as a Markdown document. Enable *Lidhje Markdown*, *Referenca Markdown*, and *Sintaksë Markdown* quality checks.

**safe-html** Enable the *HTML jo e parrezik* quality check.

**url** The string should consist of only a URL. Enable the *URL* quality check.

**ignore-bbcode** Skip the *Markup BBCode* quality check.

**ignore-duplicate** Skip the *Fjalë të njëpasnjëshme të përsëdytura* quality check.

**ignore-check-glossary** Skip the *S'ndjek fjalortë* quality check.

**ignore-double-space** Skip the *Hapësirë dyshe* quality check.

**ignore-angularjs-format** Skip the *Varg ndërshtënie AngularJS* quality check.

**ignore-c-format** Skip the *Format C* quality check.

**ignore-c-sharp-format** Skip the *Format C#* quality check.

**ignore-es-format** Skip the *ECMAScript template literals* quality check.

**ignore-i18next-interpolation** Skip the *Ndërshtënie i18next* quality check.

**ignore-java-format** Skip the *Format Java* quality check.

**ignore-java-messageformat** Skip the *Java MessageFormat* quality check.

**ignore-javascript-format** Skip the *Format JavaScript* quality check.

**ignore-lua-format** Skip the *Format Lua* quality check.

**ignore-object-pascal-format** Skip the *Object Pascal format* quality check.

**ignore-percent-placeholders** Skip the *Vendmbajtëse përqindjeje* quality check.

**ignore-perl-format** Skip the *Format Perl* quality check.

**ignore-php-format** Skip the *Format PHP* quality check.

**ignore-python-brace-format** Skip the *Format kllapash Python* quality check.

**ignore-python-format** Skip the *Format Python* quality check.

**ignore-qt-format** Skip the *Format Qt* quality check.

**ignore-qt-plural-format** Skip the *Format shumësi Qt* quality check.

**ignore-ruby-format** Skip the *Format Ruby* quality check.

**ignore-scheme-format** Skip the *Format Schema* quality check.

**ignore-vue-format** Skip the *Formatim i18n Vue* quality check.

**ignore-translated** Skip the *Është përkthyer* quality check.

**ignore-inconsistent** Skip the *Jo i njëtrajtshëm* quality check.

**ignore-kashida** Skip the *Është përdorur shkronjë Kashida* quality check.

**ignore-md-link** Skip the *Lidhje Markdown* quality check.

**ignore-md-reflink** Skip the *Referenca Markdown* quality check.

**ignore-md-syntax** Skip the *Sintaksë Markdown* quality check.

**ignore-max-length** Skip the *Gjatësi maksimum e përkthimit* quality check.

**ignore-max-size** Skip the *Madhësi maksimum e përkthimit* quality check.

**ignore-escaped-newline** Skip the *Mospërputhje n* quality check.

**ignore-end-colon** Skip the *Mospërputhje dy pikash* quality check.
ignore-end-ellipsis  Skip the Mospërputhje tre pikash quality check.
ignore-end-exclamation  Skip the Mospërputhje pikëçuditjesh quality check.
ignore-end-stop  Skip the Mospërputhje pike quality check.
ignore-end-question  Skip the Mospërputhje pikëpyetjesh quality check.
ignore-end-semicolon  Skip the Mospërputhje pikëpresjesh quality check.
ignore-newline-count  Skip the Mospërputhje ndërprerjesh rreshti quality check.
ignore-plurals  Skip the Mungon shumës quality check.
ignore-placeholders  Skip the Vendmbajtëse quality check.
ignore-punctuation-spacing  Skip the Hapësirë pikësimi quality check.
ignore-regex  Skip the Shprehje e rregullt quality check.
ignore-same-plurals  Skip the Shumës i njëjtë quality check.
ignore-begin-newline  Skip the Shenjë rresht i ri në fillim quality check.
ignore-begin-space  Skip the Hapësira fillimi quality check.
ignore-end-newline  Skip the Shenjë rresht i ri në fund quality check.
ignore-end-space  Skip the Hapërshëm i ndërprerjesh rreshti quality check.
ignore-same  Skip the Përkthim i pandryshuar quality check.
ignore-safe-html  Skip the HTML jo e parrezik quality check.
ignore-url  Skip the URL quality check.
ignore-xml-tags  Skip the Markup XML quality check.
ignore-xml-invalid  Skip the Sintaksë XML quality check.
ignore-zero-width-space  Skip the Hapësirë me gjërsi zero quality check.
ignore-ellipsis  Skip the Tre pika quality check.
ignore-long-untranslated  Skip the I papërketëprej kohësh quality check.
ignore-multiple-failures  Skip the Dështime të shumta kontrollesh quality check.
ignore-unnamed-format  Skip the Ndryshore të shumta të paemërtuara quality check.
ignore-optional-plural  Skip the Pa forma shumës quality check.

Shënim:  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

Të reja në versionin 3.11.

You can configure a list of checks which can not be ignored by setting Kontrolle me detyrim in Component configuration. Each listed check can not be ignored in the user interface and any string failing this check is marked as Needs editing (see Translation states).
2.12.4 Administrim shkronjash

Të reja në versionin 3.7.

Ndihmëz: Fonts uploaded into Weblate are used purely for purposes of the Madhësi maksimum e përkthimit check, they do not have an effect in Weblate user interface.

The Madhësi maksimum e përkthimit 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 font management tool](image)

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:
Font-family and style is automatically recognized after uploading them:

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` Përcakton gjerësinë maksimum.
- `font-family:ubuntu` Defines font group to use by specifying its identifier.
- `font-size:22` Përcakton madhësi shkronjashe.

### 2.12.5 Si të shkruani kontrollojet tuaja

A wide range of quality checks are built-in, (see Kontrole cilësie), 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. Caktoni ndoca atribut.
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).

Disa shembuj:

To install custom checks, provide a fully-qualified path to the Python class in the `CHECK_LIST`, see Custom quality checks, addons 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».

```python
# Copyright © 2012 - 2021 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.
```
Checking that Czech translation text plurals differ

Check using language info to verify the two plural forms in Czech language are not same.
```python
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 Përkrithim nga makina

Built-in support for several machine translation services and can be turned on by the administrator using `MT_SERVICES` for each one. They come subject to their terms of use, so ensure you are allowed to use them how you want.

The source language can be configured at `Project configuration`.

#### 2.13.1 amaGama

Special installation of `tmserver` run by the authors of Virtaal.

Turn on this service by adding `weblate.machinery.tmserver.AmagamaTranslation` to `MT_SERVICES`.

Shihni edhe:

Installing amaGama, Amagama, amaGama Translation Memory

#### 2.13.2 Apertium

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.

Turn on this service by adding `weblate.machinery.apertium.ApertiumAPYTranslation` to `MT_SERVICES` and set `MT_APERTIUM_APY`.

Shihni edhe:

`MT_APERTIUM_APY`, Apertium website, Apertium APy documentation
2.13.3 AWS

Të reja në versionin 3.1.

Amazon Translate is a neural machine translation service for translating text to and from English across a breadth of supported languages.

1. Turn on this service by adding `weblate.machinery.aws.AWSTranslation` to `MT_SERVICES`.
2. Install the `boto3` module.
3. Configure Weblate.

Shihni edhe:

`MT_AWS_REGION, MT_AWS_ACCESS_KEY_ID, MT_AWS_SECRET_ACCESS_KEY` Amazon Translate Documentation

2.13.4 Baidu API machine translation

Të reja në versionin 3.2.

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.

Turn on this service by adding `weblate.machinery.baidu.BaiduTranslation` to `MT_SERVICES` and set `MT_BAIDU_ID` and `MT_BAIDU_SECRET`.

Shihni edhe:

`MT_BAIDU_ID, MT_BAIDU_SECRET` Baidu Translate API

2.13.5 DeepL

Të reja në versionin 2.20.

DeepL is a 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.

Turn on this service by adding `weblate.machinery.deepl.DeepLTranslation` to `MT_SERVICES` and set `MT_DEEPL_KEY`.

Ndihmëz: In case you have subscription for CAT tools, you are supposed to use «v1 API» instead of default «v2» used by Weblate (it is not really an API version in this case). In case you are on a free instead of a paid plan, you have to use `https://api-free.deepl.com/` instead of `https://api.deepl.com/` You can adjust both parameters by `MT_DEEPL_API_URL`.

Shihni edhe:

`MT_DEEPL_KEY, MT_DEEPL_API_URL`, DeepL website, DeepL pricing, DeepL API documentation
2.13.6 LibreTranslate

Të reja në versionin 4.7.1.

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.

Turn on this service by adding `weblate.machinery.libretranslate.LibreTranslateTranslation` to `MT_SERVICES` and set `MT_LIBRETRANSLATE_API_URL`. If your instance requires an API key, you must also set `MT_LIBRETRANSLATE_KEY`.

Shihni edhe:

- `MT_LIBRETRANSLATE_KEY`, `MT_LIBRETRANSLATE_API_URL`, LibreTranslate website, LibreTranslate repository, LibreTranslate mirrors

2.13.7 Glosbe

Free dictionary and translation memory for almost every living language.

The API is gratis to use, but subject to the used data source license. There is a limit of calls that may be done from one IP in a set period of time, to prevent abuse.

Turn on this service by adding `weblate.machinery.glosbe.GlosbeTranslation` to `MT_SERVICES`.

Shihni edhe:

- Glosbe website

2.13.8 Google Translate

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.

To turn on this service, add `weblate.machinery.google.GoogleTranslation` to `MT_SERVICES` and set `MT_GOOGLE_KEY`.

Shihni edhe:

- `MT_GOOGLE_KEY`, Google translate documentation

2.13.9 Google Translate API V3 (Advanced)

Machine translation service provided by Google Cloud services.

This service differs from the former one in how it authenticates. To enable service, add `weblate.machinery.googlev3.GoogleV3Translation` to `MT_SERVICES` and set

- `MT_GOOGLE_CREDENTIALS`
- `MT_GOOGLE_PROJECT`

If `location` fails, you may also need to specify `MT_GOOGLE_LOCATION`.

Shihni edhe:

- `MT_GOOGLE_CREDENTIALS`, `MT_GOOGLE_PROJECT`, `MT_GOOGLE_LOCATION` Google translate documentation
2.13.10 Microsoft Cognitive Services Translator

Të reja në versionin 2.10.

Machine translation service provided by Microsoft in Azure portal as a one of Cognitive Services.

Weblate implements Translator API V3.

To enable this service, add `weblate.machinery.microsoft.MicrosoftCognitiveTranslation` to `MT_SERVICES` and set `MT_MICROSOFT_COGNITIVE_KEY`.

**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 `MT_MICROSOFT_REGION` to locale of your service.

Shihni edhe:

`MT_MICROSOFT_COGNITIVE_KEY, MT_MICROSOFT_REGION`, Cognitive Services - Text Translation API, Microsoft Azure Portal

2.13.11 Microsoft Terminology Service

Të reja në versionin 2.19.

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.

Turn this service on by adding `weblate.machinery.microsoftterminology`. MicrosoftTerminologyService to `MT_SERVICES`.

Shihni edhe:

Microsoft Terminology Service API

2.13.12 ModernMT

Të reja në versionin 4.2.

Turn this service on by adding `weblate.machinery.modernmt.ModernMTTranslation` to `MT_SERVICES` and configure `MT_MODERNMT_KEY`.

Shihni edhe:

ModernMT API, `MT_MODERNMT_KEY, MT_MODERNMT_URL`
2.13.13 MyMemory

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 `MT_MYMEMORY_EMAIL`. You can also ask them for more.

Turn on this service by adding `weblate.machinery.mymemory.MyMemoryTranslation` to `MT_SERVICES` and set `MT_MYMEMORY_EMAIL`.

Shihni edhe:

`MT_MYMEMORY_EMAIL, MT_MYMEMORY_USER, MT_MYMEMORY_KEY`, MyMemory website

2.13.14 NetEase Sight API machine translation

Tërejanë versionin 3.3.

Machine translation service provided by NetEase.

This service uses an API, and you need to obtain key and secret from NetEase.

Turn on this service by adding `weblate.machinery.youdao.NeteaseSightTranslation` to `MT_SERVICES` and set `MT_NETEASE_KEY` and `MT_NETEASE_SECRET`.

Shihni edhe:

`MT_NETEASE_KEY, MT_NETEASE_SECRET`, NetEase Sight Translation Platform

2.13.15 tmserver

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:

2. Turn on this service by adding `weblate.machinery.tmserver.TMServerTranslation` to `MT_SERVICES`.

```bash
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
```

3. Start tmserver to listen to your requests:

```bash
tmserver -d /var/lib/tm/db
```

4. Configure Weblate to talk to it:

```bash
MT_TMSERVER = "http://localhost:8888/tmserver/
```

Shihni edhe:

`MT_TMSERVER`, tmserver Installing amaGama, Amagama, Amagama Translation Memory
2.13.16 Yandex Translate

Machine translation service provided by Yandex.

This service uses a Translation API, and you need to obtain an API key from Yandex.

Turn on this service by adding weblate.machinery.yandex.YandexTranslation to \texttt{MT\_SERVICES}, and set \texttt{MT\_YANDEX\_KEY}.

Shihni edhe:

\texttt{MT\_YANDEX\_KEY}, Yandex Translate API, Powered by Yandex.Translate

2.13.17 Youdao Zhiyun API machine translation

Të reja në versionin 3.2.

Machine translation service provided by Youdao.

This service uses an API, and you need to obtain an ID and an API key from Youdao.

Turn on this service by adding weblate.machinery.youdao.YoudaoTranslation to \texttt{MT\_SERVICES} and set \texttt{MT\_YOUDAO\_ID} and \texttt{MT\_YOUDAO\_SECRET}.

Shihni edhe:

\texttt{MT\_YOUDAO\_ID, MT\_YOUDAO\_SECRET} Youdao Zhiyun Natural Language Translation Service

2.13.18 Weblate

Weblate can be the source of machine translations as well. It is based on the Woosh fulltext engine, and provides both exact and inexact matches.

Turn on these services by adding weblate.machinery.weblatetm.WeblateTranslation to \texttt{MT\_SERVICES}.

2.13.19 Weblate Translation Memory

Të reja në versionin 2.20.

The Keshtje Përkthimesh can be used as a source for machine translation suggestions as well.

Turn on these services by adding weblate.memory.machine.WeblateMemory to the \texttt{MT\_SERVICES}. This service is turned on by default.

2.13.20 SAP Translation Hub

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.

Turn on this service by adding weblate.machinery.saptransla\-tionhub.SAPTranslationHub to \texttt{MT\_SERVICES} and set the appropriate access to either the sandbox or the production API.

Shënim: To access the Sandbox API, you need to set \texttt{MT\_SAP\_BASE\_URL} and \texttt{MT\_SAP\_SANDBOX\_APIKEY}.

To access the productive API, you need to set \texttt{MT\_SAP\_BASE\_URL}, \texttt{MT\_SAP\_USERNAME} and \texttt{MT\_SAP\_PASSWORD}. 
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:

```python
# Copyright © 2012 - 2021 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, unit, user, search, threshold=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 `MT_SERVICES` and Weblate will start using that.
2.14 Shtesa

Të reja në versionin 2.19.

Add-ons provide ways to customize and automate the translation workflow. Admins can add and manage add-ons from the Manage ↓ Add-ons menu of each respective translation component.

Ndihmëz: You can also configure add-ons using API, DEFAULT_ADDONS, or install_addon.
## Kapitulli 2. Dokumentime përgjegjësi

<table>
<thead>
<tr>
<th>Add-on Name</th>
<th>Description</th>
<th>Installation Type</th>
<th>Install</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automatic translation</td>
<td>Automatically translates strings using machine translation or other components.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add missing languages</td>
<td>Ensures a consistent set of languages is used for all components within a project.</td>
<td>project wide</td>
<td></td>
</tr>
<tr>
<td>Component discovery</td>
<td>Automatically adds or removes project components based on file changes in the version control system.</td>
<td>repository wide</td>
<td></td>
</tr>
<tr>
<td>Bulk edit</td>
<td>Bulk edit flags, labels, or states of strings.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Statistics generator</td>
<td>Generates a file containing detailed info about the translation status.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pseudolocale generation</td>
<td>Generates a translation by adding prefix and suffix to source strings automatically.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contributors in comment</td>
<td>Updates the comment part of the PO file header to include contributor names and years of contributions.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Customize gettext output</td>
<td>Allows customization of gettext output behavior, for example line wrapping.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Generate MO files</td>
<td>Automatically generates a MO file for every changed PO file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Update PO files to match POT (msgmerge)</td>
<td>Updates all PO files (as configured by “Filenmask”) to match the POT file (as configured by “Template for new translations”) using msgmerge.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Squash Git commits</td>
<td>Squash Git commits prior to pushing changes.</td>
<td>repository wide</td>
<td></td>
</tr>
<tr>
<td>Stale comment removal</td>
<td>Set a timeframe for removal of comments.</td>
<td>project wide</td>
<td></td>
</tr>
<tr>
<td>Stale suggestion removal</td>
<td>Set a timeframe for removal of suggestions.</td>
<td>project wide</td>
<td></td>
</tr>
</tbody>
</table>

Some add-ons will ask for additional configuration during installation.
2.14.1 Built-in add-ons

Përkthim i automatizuar

Të reja në versionin 3.9.

**Add-on ID** weblate.autotranslate.autotranslate

**Formësim**

<table>
<thead>
<tr>
<th>mode</th>
<th>Mënyrë përkthim i automatizuar</th>
<th>Available choices:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>suggest – Add as suggestion</td>
</tr>
<tr>
<td></td>
<td></td>
<td>translate – Add as translation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fuzzy – Add as needing edit</td>
</tr>
</tbody>
</table>

| filter_type | Filtër kërkimi | Available choices: |
|            |                | all – All strings |
|            |                | nottranslated – Not translated strings |
|            |                | todo – Strings needing action |
|            |                | fuzzy – Strings marked for edit |
|            | check:inconsistent – Failed check: Inconsistent |

| auto_source | Burim përkthimesh të automatizuarara | Available choices: |
|            |                                  | others – Other translation components |
|            |                                  | mt – Machine translation |

| component engines | Përbërës Motorë përkthimi nga makina |
|                   | Aktivizoni kontribut te kujtesë e përbashkët përkthimesh për projektin, që të mund të hyni te përbërës të tjerë. |

<table>
<thead>
<tr>
<th>threshold</th>
<th>Prag përkthimi vlerësimi</th>
</tr>
</thead>
</table>

**Triggers** component update, daily

Përkthen automatikisht vargje duke përdorur përkthim nga makina ose nga përbërës të tjerë.

It is triggered:

- When new strings appear in a component.
- Once in a month for every component, this can be configured using `BACKGROUND_TASKS`.

**Shihni edhe:**

*Përkthim i automatizuar, Keeping translations same across components*

**CDN përkthimesh JavaScript**

Të reja në versionin 4.2.

**Add-on ID** weblate.cdn.cdnjs

**Formësim**

<table>
<thead>
<tr>
<th>threshold</th>
<th>Prag përkthimi</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Prag për përfshirje përkthimesh.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>css_selector</th>
<th>Përzgjedhës CSS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Përzgjedhës CSS për të pikasur elementë të përkthyeshm.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>cookie_name</th>
<th>Emër për cookie-n e gjuhës</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Emër për cookie-n e cila depoziton parapëlqim gjuhe.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>files</th>
<th>Përftovarje prej kartelash HTML</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Listë e emrave të kartelave në depon e tanishme ose URL-ra të largëta që duhen kontrolluar për vargje të përkthyeshm.</td>
</tr>
</tbody>
</table>
Triggers daily, repository post-commit, repository post-update

Beton përkrthime te një rrjet shpjëndarjeje lënde për përdorim në përkrthime JavaScript-i ose HTML-je.

Can be used to localize static HTML pages, or to load localization in the JavaScript code.

Generates a unique URL for your component you can include in HTML pages to localize them. See weblate-cdn for more details.

Shihni edhe:

cdn-addon-config, weblate-cdn, cdn-addon-extract, cdn-addon-html

Hiqni vargje të zbrazët

Të reja në versionin 4.4.

Add-on ID weblate.cleanup.blank

Formësim This add-on has no configuration.

Triggers repository post-commit, repository post-update

Heq nga kartelat e përkrthimit vargje pa përkrthim.

Use this to not have any empty strings in translation files (for example if your localization library displays them as missing instead of falling back to the source string).

Shihni edhe:

Does Weblate update translation files besides translations?

Spastro kartela përkrthimi

Add-on ID weblate.cleanup.generic

Formësim This add-on has no configuration.

Triggers repository pre-commit, repository post-update

Përditësoni krejt kartelat e përkrthimit që të përputhen me kartelën bazë njëgjuhëshe. Për shumicën e formateve të kartelave, kjo do të thotë heqje zërash të ndenjur përkrthimi, që s’janë më të tranishëm te kartela bazë.

Shihni edhe:

Does Weblate update translation files besides translations?

Add missing languages

Add-on ID weblate.consistency.languages

Formësim This add-on has no configuration.

Triggers daily, repository post-add

Siguron përdorimin e një grupi të qëndrueshëm gjuhë për krejt përbërësit brenda një projekti.

Missing languages are checked once every 24 hours, and when new languages are added in Weblate.

Unlike most others, this add-on affects the whole project.

Ndihmëz: Auto-translate the newly added strings with Përkrthim i automatizuar.
**Pikasje përxbërësisht**

**Add-on ID** weblate.discovery.discovery

**Formësim**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>match</td>
<td>Shprehje e rregullt kundrejt të cilës duhen kërkuar përputhje te kartela përkhimi</td>
</tr>
<tr>
<td>file_format</td>
<td>Format kartelash</td>
</tr>
<tr>
<td>name_template</td>
<td>Përshtatni emrin e përxbërësit</td>
</tr>
<tr>
<td>base_file</td>
<td>Përcaktimi emër kartelë bazë njëjëshës</td>
</tr>
<tr>
<td>new_base</td>
<td>Përcaktimi kartelën bazë për përkrthime të reja</td>
</tr>
<tr>
<td>language_regex</td>
<td>Filtër gjësh</td>
</tr>
<tr>
<td>language_regex</td>
<td>Shprehje e rregullt kundrejt së cilës të filtrohen kartela përkhimi, kur skanohet për filemask.</td>
</tr>
<tr>
<td>copy_addons</td>
<td>Klono shtesa prej përxbërësit kryesor te të reja të sapokrijuara</td>
</tr>
<tr>
<td>remove</td>
<td>Hiq përxbërës për kartela që s’ekzistonjë</td>
</tr>
<tr>
<td>confirm</td>
<td>Ripohoj se përputhjet më sipër duken të sakta</td>
</tr>
</tbody>
</table>

**Triggers** repository post-update

Shtonose heq automatikisht përxbërës projektqi, bazuar te ndryshime kartelash nën sistem kontrolli versionesh.

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:
Kapitulli 2. Dokumentime përgjegjësi
Component discovery add-on uses URL të brendshme Weblate. 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 `Depo kodi burim` field (in `Manage ↓ Settings ↓ Version control system`) of each respective component. This saves time with configuration and system resources too.

Shihni edhe:

Template markup

Përpunim në masë

Të reja në versionin 3.11.

Add-on ID `weblate.flags.bulk`

Formësim

<table>
<thead>
<tr>
<th>q</th>
<th>Kërkesë</th>
<th>Available choices:</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>Gjendje për t'u caktuar</td>
<td>-1 – Do not change</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10 – Needs editing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20 – Translated</td>
</tr>
<tr>
<td></td>
<td></td>
<td>30 – Approved</td>
</tr>
<tr>
<td>add_flags</td>
<td>Flamurka për kthimi për shtim</td>
<td></td>
</tr>
<tr>
<td>remove_flags</td>
<td>Flamurka për kthimi për heqje</td>
<td></td>
</tr>
<tr>
<td>add_labels</td>
<td>Etiket për shtim</td>
<td></td>
</tr>
<tr>
<td>remove_labels</td>
<td>Etiket për heqje</td>
<td></td>
</tr>
</tbody>
</table>

Triggers component update

Përputim në masë vargjesh shenjesh, etiketash ose gjendjesh.

Automate labeling by starting out with the search query `NOT has:label` and add labels till all strings have all required labels. Other automated operations for Weblate metadata can also be done.

Shembuj:

<table>
<thead>
<tr>
<th>Search query</th>
<th>NOT has:label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Etiket për shtim</td>
<td>recent</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search query</th>
<th>language:en AND key:changelogs/</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flamurka për kthimi për shtim</td>
<td>read-only</td>
</tr>
</tbody>
</table>

Shihni edhe:

_Përpunim në masë, Customizing behavior using flags, labels_
Vëru shenjë përkthimeve të pandryshuara si “Lyp përditësim”

Të reja në versionin 3.1.

Add-on ID  weblate.flags.same_edit
Formësim  This add-on has no configuration.
Triggers  unit post-create

Kurdo që importohet një varg i ri i përkthyeshëm nga VCS-ja dhe përputhet me vargun burim, në Weblate i vihet shenjësi që ka nevojë të përpunohet. Kjo është veçanërisht e dobishme për formate kartelash që përfshijnë vargje burim për vargjet e papërkthyer.

Ndihmëz: You might also want to tighten the Përkthimi i pandryshuar check by adding strict-same flag to Flamuj përkthimi.

Vëru shenjë vargjeve të reja burim si “Lyp përditësim”

Add-on ID  weblate.flags.source_edit
Formësim  This add-on has no configuration.
Triggers  unit post-create

Kurdo që importohet një varg i ri burim nga VCS-ja, në Weblate i vihet shenjësi “Lyp përpunim”. Në këtë mënyrë mund të filtroni lehtësisht dhe përpunoni vargje burim shkruar nga zhvilluesit.

Shihni edhe:
Translation states

Vëru shenjë përkthimeve të reja si “Lyp përditësim”

Add-on ID  weblate.flags.target_edit
Formësim  This add-on has no configuration.
Triggers  unit post-create

Kurdo që importohet një varg i ri i përkthyeshëm nga VCS-ja, në Weblate i vihet shenjësi “Lyp përpunim”. Në këtë mënyrë mund të filtroni lehtësisht dhe përpunoni vargje të përkthyeshëm krijuar nga zhvilluesit.

Shihni edhe:
Translation states

Prodhues statistikash

Add-on ID  weblate.generate.generate
Formësim

```
<table>
<thead>
<tr>
<th>filename</th>
<th>Emër i kartelës së prodhuar</th>
</tr>
</thead>
<tbody>
<tr>
<td>template</td>
<td>Lëndë e kartelës së prodhuar</td>
</tr>
</tbody>
</table>
```

Triggers repository pre-commit
Prodhon një kartelë që përmban të dhëna të hollësishme rreth gjendjes së përkthimit. You can use a Django template in both filename and content, see Template markup for a detailed markup description.

For example generating a summary file for each translation:

Emër i kartelës së prodhuar `locale/{{ language_code }}.json`

Lëndë

```
{
    "language": "{{ language_code }}",
    "strings": "{{ stats.all }}",
    "translated": "{{ stats.translated }}",
    "last_changed": "{{ stats.last_changed }}",
    "last_author": "{{ stats.last_author }}",
}
```

Shihni edhe:

Template markup

**Prodhim pseudovendoreje**

Add-on ID `weblate.generate.pseudolocale`

Formësim

<table>
<thead>
<tr>
<th>source</th>
<th>Vargjet e burimit</th>
</tr>
</thead>
<tbody>
<tr>
<td>target</td>
<td>Përkthimi i synuar</td>
</tr>
<tr>
<td>prefix</td>
<td>Parashtesë vargjesh</td>
</tr>
<tr>
<td>suffix</td>
<td>Prapashtesë vargjesh</td>
</tr>
</tbody>
</table>

**Triggers** component update, daily

Prodhon një përkthimin duke shtruar automatikisht parashtesë dhe prapashtesë te vargjet burim.

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.

**Ndihmëz** You can use real languages for testing, but there are dedicated pseudolocales available in Weblate - `en_XA` and `ar_XB`.

**Ndihmëz** 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.
Kontribues në koment

Add-on ID  weblate.gettext.authors

Formësim  This add-on has no configuration.

Triggers  repository pre-commit

Përditëson pjesën e komentit te kryet e kartelës PO, për të përfshirë emrat e kontribuesve dhe vite kontributi.

The PO file header will look like this:

```
# Pavel Borecki <pavel@example.com>, 2018, 2019.
# Filip Hron <filip@example.com>, 2018, 2019.
# anonymous <noreply@weblate.org>, 2019.
```

Përditëson ndryshoren ALL_LINGUAS te kartela «configure»

Add-on ID  weblate.gettext.configure

Formësim  This add-on has no configuration.

Triggers  repository post-add, daily

Updates the ALL_LINGUAS variable in configure, configure.in or any configure.ac files, when a new translation is added.

Përshtatni përfundimin e gettext-it

Add-on ID  weblate.gettext.customize

Formësim

<table>
<thead>
<tr>
<th>width</th>
<th>Mbështetje rreshtash të gjatë</th>
</tr>
</thead>
<tbody>
<tr>
<td>77</td>
<td>Mbështetje rreshtat kur mbërritëhet në 77 shenja dhe rreshta të rinj. Me parametrin --no-wrap, mbështetja kryhet vetëm në rreshta të rinj. Available choices: 77 – Wrap lines at 77 characters and at newlines 65535 – Only wrap lines at newlines -1 – No line wrapping</td>
</tr>
</tbody>
</table>

Triggers  storage post-load

Lejon përshtatje të flogj të përfundimeve nga gettext-i, për shembull, mbështetje rreshtash.

Ofron mundësitë vijuese:

- Mbështëtje rreshtat pas 77 shenjash dhe në rreshta të rinj
- Mbështëtje rreshtat vetëm në rreshta të rinj
- Pa mbështetje rreshtash

Shënim:  By default gettext wraps lines at 77 characters and at newlines. With the --no-wrap parameter, wrapping is only done at newlines.
Përditëso kartelën LINGUAS

**Add-on ID** weblate.gettext.linguas

**Formësim** *This add-on has no configuration.*

**Triggers** repository post-add, daily

Përditëson kartelën LINGUAS kur shtohet një përkthim i ri.

**Prodho kartela MO**

**Add-on ID** weblate.gettext.mo

**Formësim**

| path | Shteg për te kartela MO e prodhuar | Nëse s’është përcaktuar, do të përdoret vendndodhja e kartelës PO. |

| path | Shteg për te kartela MO e prodhuar | Nëse s’është përcaktuar, do të përdoret vendndodhja e kartelës PO. |

**Triggers** repository pre-commit

Prodhon automatikisht një kartelë MO për çdo kartelë PO të ndryshuar. The location of the generated MO file can be customized and the field for it uses Template markup.

Përditësoni kartela PO, që të përputhen me POT (msgmerge)

**Add-on ID** weblate.gettext.msgmerge

**Formësim**

| previous | Mba msgid t’ëmëparshëm të vargjeve t’ë përkthyera |
| no_location | Hiq vendndodhje vargjesh të përkthyer |
| fuzzy | Përdor përputhje të turbullt |

**Triggers** repository post-update

Updates all PO files (as configured by File mask) to match the POT file (as configured by Gjedhe për përkthime të reja) using msgmerge.

Triggered whenever new changes are pulled from the upstream repository. Most msgmerge command-line options can be set up through the add-on configuration.

**Shihni edhe:**

Does Weblate update translation files besides translations?

Kryej «squash» për depozit TIME Git

**Add-on ID** weblate.git.squash

**Formësim**
**Available choices:**
- `all` – All commits into one language
- `language` – Per language
- `file` – Per file
- `author` – Per author

**Available choices:**
- `append_trailers` – Append trailers to squashed commit message

**Commit message**
This commit message will be used instead of the combined commit messages from the squashed commits.

---

**Triggers** repository post-commit

Kryej «squash» për depozitimet Git, përpara kryerjes së push-it për ndryshimet.

Git commits can be squashed prior to pushing changes in one of the following modes:

Të reja në versionin 3.4.
- Krejt depozitimet në një të tilli
- Për gjuhë
- Për kartelë

Të reja në versionin 3.5.
- Për autor

Mesazhet originale të depozitimeve mbahen, por autorësia humb, veç në qoftë për zgjidhur *Për autor*, ose mesazhi i depozitimit është përshtatuar për ta përmbytjet atë.

Të reja në versionin 4.1.

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.

---

**Përshtatni përfundim JSON-i**

**Add-on ID** weblate.json.customize

**Formësim**

<table>
<thead>
<tr>
<th>sort_keys</th>
<th>Rendit kyçe JSON</th>
</tr>
</thead>
<tbody>
<tr>
<td>indent</td>
<td>Hapësirë kryeradhe JSON</td>
</tr>
<tr>
<td>style</td>
<td>Stil hapësirash kryeradhe JSON</td>
</tr>
<tr>
<td></td>
<td>Available choices:</td>
</tr>
<tr>
<td></td>
<td><code>spaces</code> – Spaces</td>
</tr>
<tr>
<td></td>
<td><code>tabs</code> – Tabs</td>
</tr>
</tbody>
</table>

**Triggers** storage post-load

Lejon përshtatje të sjelljes së JSON-it për përfundimet, për shembull, lidhur me shmangje kryeradhe ose renditje.
Formaton kartelën e vetive Java

Add-on ID  weblate.properties.sort
Formësim  This add-on has no configuration.
Triggers  repository pre-commit
Rendit kartelën e vetive Java.

Heqje komentesh të vjetruar

Të reja në versionin 3.7.

Add-on ID  weblate.removal.comments
Formësim

<table>
<thead>
<tr>
<th>age</th>
<th>Ditë për t’u mbajtur</th>
</tr>
</thead>
</table>

Triggers  daily
Caktoni kohë për heqje komentesh.
This can be useful to remove old comments which might have become outdated. Use with care as comments getting old does not mean they have lost their importance.

Heqje sugjerimesh të vjetruar

Të reja në versionin 3.7.

Add-on ID  weblate.removal.suggestions
Formësim

<table>
<thead>
<tr>
<th>age</th>
<th>Ditë për t’u mbajtur</th>
</tr>
</thead>
<tbody>
<tr>
<td>votes</td>
<td>Prag votimi</td>
</tr>
<tr>
<td></td>
<td>Prag për heqje. Kjo fushë s’do të ketë efekt kur votimi është i çaktivizuar.</td>
</tr>
</tbody>
</table>

Triggers  daily
Caktoni kohë për heqje sugjerimesh.
Can be very useful in connection with suggestion voting (see Peer review) to remove suggestions which don’t receive enough positive votes in a given timeframe.

Përditësoni kartela RESX

Të reja në versionin 3.9.

Add-on ID  weblate.resx.update
Formësim  This add-on has no configuration.
Triggers  repository post-update
Përditëso krejt kartelat e përkrimit të t’i përputhur me kartelën bazë njëgjushëse të upstream-it. Kartelat e papërdorura hiqen, dhe të reja shtohen si kopje të vargjeve burim.

Ndihmëz:  Use Spastro kartela përkthimi if you only want to remove stale translation keys.
Shihni edhe:

Does Weblate update translation files besides translations?

Përshtatnë për fundim YAML

Të reja në versionin 3.10.2.

**Add-on ID** weblate.yaml.customize

**Formësim**

<table>
<thead>
<tr>
<th>indent</th>
<th>Hapësirë kryeradhe YAML</th>
</tr>
</thead>
<tbody>
<tr>
<td>width</td>
<td>Mbetëtjullje rreshtash të gjatë</td>
</tr>
</tbody>
</table>
|        | Available choices: 80 – Wrap lines at 80 chars  
|        | 100 – Wrap lines at 100 chars  
|        | 120 – Wrap lines at 120 chars  
|        | 180 – Wrap lines at 180 chars  
|        | 65535 – No line wrapping |
| line_break | Ndërprerje rreshtash |
|        | Available choices: dos – DOS (\r\n)  
|        | unix – UNIX (\n)  
|        | mac – MAC (\r) |

**Triggers** storage post-load

Lejon përshatjetë të sjelljes së YAML-it për për fundimet, për shembull, lidhur me gjatësi rreshtit ose rreshtat të rinj.

2.14.2 Customizing list of add-ons

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 Writing add-on

You can write your own add-ons too, create a subclass of `weblate.addons.base.BaseAddon` to define the addon metadata, and then implement a callback to do the processing.

Shihni edhe:

*Developing addons*

2.14.4 Executing scripts from add-on

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.
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 addon 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, addons 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**
Sistem kontrolli versionesh i përdorur.

**WL_REPO**
Upstream repository URL.

**WL_PATH**
Shteg absolut për te depo VCS.

**WL_BRANCH**
Të reja në versionin 2.11.

Repository branch configured in the current component.

**WL_FILEMASK**
Filemask for current component.

**WL_TEMPLATE**
Filename of template for monolingual translations (can be empty).

**WL_NEW_BASE**
Të reja në versionin 2.14.

Filename of the file used for creating new translations (can be empty).
The Weblate Manual, Hedhje Në Qarkullim 4.9

WL_FILE_FORMAT
File format used in current component.

WL_LANGUAGE
Language of currently processed translation (not available for component-level hooks).

WL_PREVIOUS_HEAD
Previous HEAD after update (only available after running the post-update hook).

WL_COMPONENT_SLUG
Të reja në versionin 3.9.
Identifikues përbërësi i përdorur për të ndërtuar URL-në.

WL_PROJECT_SLUG
Të reja në versionin 3.9.
Identifikues projekt i përdorur për të ndërtuar URL-në.

WL_COMPONENT_NAME
Të reja në versionin 3.9.
Emër përbërësi.

WL_PROJECT_NAME
Të reja në versionin 3.9.
Emër projekt.

WL_COMPONENT_URL
Të reja në versionin 3.9.
URL përbërësi.

WL_ENGAGE_URL
Të reja në versionin 3.9.
Project engage URL.

Shihni edhe:

Component configuration

Post-update repository processing

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:

```sh
#!/bin/sh
gulp --gulpfile gulp-i18n-extract.js
git commit -m 'Update source strings' src/languages/en.lang.json
```

Pre-commit processing of translations

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 Kujtesë Përkthimesh

Të reja në versionin 2.20.

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, Sugjerime të automatizuara view while translating.
• Automatically, by translating strings using Përkhim i automatizuar, or Përkhim i automatizuar addon.

For installation tips, see Weblate Translation Memory, which is turned on by default.

2.15.1 Translation memory scopes

Të reja në versionin 3.2: 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.

Kujtesë përkthimesh e përbashkët

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

Të reja në versionin 3.2.

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.

Ndihmëz: Translation memory in JSON can be imported into Weblate, TMX is provided for interoperability with other tools.

Shihni edhe:

Weblate Translation Memory Schema

<table>
<thead>
<tr>
<th>Translation memory status</th>
<th>Number of your entries</th>
<th>Download as JSON</th>
<th>Download as TMX</th>
<th>Delete</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Total number of entries   | 0                      |

| Import translation memory | File                     |                      |
|----------------------------|--------------------------|
|                            | Choose File: No file chosen |
|                            | You can upload a TMX or JSON file. |

| Upload                     |                         |

Ndërfaqe administrimi

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 Formësim

All settings are stored in `settings.py` (as is usual for Django).

Shënim: 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.

Shihni edhe:

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

Emër përdoruesi për përdorues që nuk kanë bërë hyrjen.

Shihni edhe:

Kontroll hyrjesh

2.16.3 AUDITLOG_EXPIRY

Të reja në versionin 3.6.

How many days Weblate should keep audit logs, which contain info about account activity.

180 ditë, si parazgjedhje..

2.16.4 AUTH_LOCK_ATTEMPTS

Të reja në versionin 2.14.

Maximum number of failed authentication attempts before rate limiting is applied.

This is currently applied in the following locations:

- Logins. Deletes the account password, preventing the user from signing in without requesting a new password.
- Password resets. Prevents new e-mails from being sent, avoiding spamming users with too many password reset attempts.

10, si parazgjedhje.

Shihni edhe:

Rate limiting

2.16.5 AUTO_UPDATE

Të reja në versionin 3.2.

Ndryshuar në versionin 3.11: The original on/off option was changed to differentiate which strings are accepted.

Updates all repositories on a daily basis.

Ndihmëz: Useful if you are not using Hook-e njofimesh to update Weblate repositories automatically.

Shënim: On/off options exist in addition to string selection for backward compatibility.

Mundësitë janë:

"none" Pa përditësime të përditshme.

"remote" also False Vetëm përditësime të largëta.

"full" also True Update remotes and merge working copy.
Shënim: 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/](https://gravatar.com/)
  
  ```
  AVATAR_URL_PREFIX = 'https://www.gravatar.com/
  ```

- **Libravatar, sipas** [https://www.libravatar.org/](https://www.libravatar.org/)
  
  ```
  AVATAR_URL_PREFIX = 'https://www.libravatar.org/
  ```

Shihni edhe:

*Avatar caching, ENABLE_AVATARS, Avatars*

### 2.16.7 AUTH_TOKEN_VALID

Të reja në versionin 2.14.

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

Të reja në versionin 2.15.

How many days using the same password should be allowed.

Shënim: Password changes made prior to Weblate 2.15 will not be accounted for in this policy.

180 ditë, si parazgjedhje..

### 2.16.9 AUTOFIX_LIST

List of automatic fixes to apply when saving a string.

Shënim: Provide a fully-qualified path to the Python class that implementing the autofixer interface.

Ndreqje të mundshme:

- **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 *HTML jo e parrezik*).
You can select which ones to use:

```python
AUTOFIX_LIST = {
    "weblate.trans.autofixes.whitespace.SameBookendingWhitespace",
    "weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis",
}
```

Shihni edhe:

*Automatic fixups, Custom automatic fixups*

### 2.16.10 BACKGROUND_TASKS

Të reja në versionin 4.5.2.

Defines how often lengthy maintenance tasks should be triggered for a component.

Right now this controls:

- *Përktëm i automatizuar* addon
- *Checks and fixups* recalculation

Possible choices:

- *monthly* (this is the default)
- *weekly*
- *daily*
- *never*

**Shënim:** 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

Të reja në versionin 4.4.

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.

**Shënim:** This does not define new languages for Weblate, it only filters existing ones in the database.

Example:

```
BASIC_LANGUAGES = {"cs", "it", "ja", "en"}
```
2.16.13 **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.

**Example:**

```
# Enable Cloudflare Javascript optimizations
CSP_SCRIPT_SRC = ['ajax.cloudflare.com']
```

Shihni edhe:

*Content security policy, Content Security Policy (CSP)*

2.16.14 **CHECK_LIST**

List of quality checks to perform on a translation.

**Shënim:** 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 Kontrolle cilësise 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.

Mund të çaktivizoni krejt kontrolet:

```
CHECK_LIST = ()
```

Mund të aktivizoni vetë, pak prej tyre:

```
CHECK_LIST = (  
    "weblate.checks.chars.BeginNewlineCheck",  
    "weblate.checks.chars.EndNewlineCheck",  
    "weblate.checks.chars.MaxLengthCheck",  
)
```

**Shënim:** 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`.

Shihni edhe:

*Kontrolle cilësise, Customizing behavior using flags*
2.16.15 COMMENT_CLEANUP_DAYS
Të reja në versionin 3.6.
Delete comments after a given number of days. Defaults to None, meaning no deletion at all.

2.16.16 COMMIT_PENDING_HOURS
Të reja në versionin 2.10.
Number of hours between committing pending changes by way of the background task.
Shihni edhe:
Component configuration, Moshë ndryshimesh për depozitën, Xhirim aktesh mirëmbajtjeje, commit_pending

2.16.17 CONTACT_FORM
Të reja në versionin 4.6.
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.

2.16.18 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.
Zakonisht ekzistojnë njëndjëtoritë vijuese:
home Home directory used for invoking scripts.
ssh Kyç dhe formësim SSH-je.
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 Version control repositories for translations.
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 Administrim shkronjash.

Shënim: 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
```

$BASE_DIR/data, si parazgjedhje.
Shihni edhe:

BASE_DIR, Leje sistemi kartelash, Kopjeruajtje dhe zhvendosje e Weblate

## 2.16.19 DATABASE_BACKUP

Të reja në versionin 3.1.

Whether the database backups should be stored as plain text, compressed or skipped. The authorized values are:

- "plain"
- "compressed"
- "none"

Shihni edhe:

Kopjeruajtje dhe zhvendosje e Weblate

## 2.16.20 DEFAULT_ACCESS_CONTROL

Të reja në versionin 3.3.

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.

Shihni edhe:

Kontroll hyrjesh te projekt, Kontroll hyrjesh

## 2.16.21 DEFAULT_AUTO_WATCH

Të reja në versionin 4.5.

Configures whether Automatically watch projects on contribution should be turned on for new users. Defaults to True.

Shihni edhe:

Njoftime

## 2.16.22 DEFAULT_RESTRICTED_COMPONENT

Të reja në versionin 4.1.

Vlera parazgjedhje për kufizim përbërësish.

Shihni edhe:

Restricted access, Scope of groups
2.16.23 DEFAULT_ADD_MESSAGE, DEFAULT_COMMIT_MESSAGE, DEFAULT_MERGE_MESSAGE

Default commit messages for different operations, please check Component configuration for details.

Shihni edhe:
Template markup, Component configuration, Commit, add, delete, merge and addon messages

2.16.24 DEFAULT_ADDONS

Default addons to install on every created component.

Shënim: This setting affects only newly created components.

Shembull:

```json
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",
    },
}
```

Shihni edhe:
install_addon, Shtesa, WEBLATE_ADDONS

2.16.25 DEFAULT_COMMITER_EMAIL

Të reja në versionin 2.4.
Committer e-mail address defaulting to noreply@weblate.org.

Shihni edhe:
DEFAULT_COMMITER_NAME

2.16.26 DEFAULT_COMMITER_NAME

Të reja në versionin 2.4.
Committer name defaulting to Weblate.

Shihni edhe:
DEFAULT_COMMITER_EMAIL
2.16.27 DEFAULT_LANGUAGE

Të reja në versionin 4.3.2.
Default source language to use for example in Gjuhë burimi.
Defaults to \textit{en}. The matching language object needs to exist in the database.

Shihni edhe:
\textit{Language definitions, Gjuhë burimi}

2.16.28 DEFAULT_MERGE_STYLE

Të reja në versionin 3.4.
Merge style for any new components.
\begin{itemize}
  \item \textit{rebase} - default
  \item \textit{merge}
\end{itemize}

Shihni edhe:
\textit{Component configuration, Stil përzierje}

2.16.29 DEFAULT_SHARED_TM

Të reja në versionin 3.2.
Configures default value of \textit{Përdor kujtesë përkthimesh të përbashkët} and \textit{Kontribuoni te kujtesë përkthimesh të përbashkët}.

2.16.30 DEFAULT_TRANSLATION_PROPAGATION

Të reja në versionin 2.5.
Default setting for translation propagation, defaults to \textit{True}.

Shihni edhe:
\textit{Component configuration, Lejo përhapje përkthimesh}

2.16.31 DEFAULT_PULL_MESSAGE

Title for new pull requests, defaulting to \textquote{Update from Weblate}'.

2.16.32 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.

Shihni edhe:
\textit{Avatar caching, AVATAR\_URL\_PREFIX, Avatars}
2.16.33 ENABLE_HOOKS

Whether to enable anonymous remote hooks.

Shihni edhe:

Hook-e njofimesh

2.16.34 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.

Shihni edhe:

SESSION_COOKIE_SECURE, CSRF_COOKIE_SECURE, SECURE_SSL_REDIRECT, SECURE_PROXY_SSL_HEADER Set correct site domain

2.16.35 ENABLE_SHARING

Turn on/off the Share menu so users can share translation progress on social networks.

2.16.36 GET_HELP_URL

Të reja në versionin 4.5.2.

URL where support for your Weblate instance can be found.

2.16.37 GITLAB_CREDENTIALS

Të reja në versionin 4.3.

List for credentials for GitLab servers.

Ndihmëz: Use this in case you want Weblate to interact with more of them, for single GitLab endpoint stick with GITLAB_USERNAME and GITLAB_TOKEN.

```python
GITLAB_CREDENTIALS = {
    "gitlab.com": {
        "username": "weblate",
        "token": "your-api-token",
    },
    "gitlab.example.com": {
        "username": "weblate",
        "token": "another-api-token",
    },
}
```
2.16.38 GITLAB_USERNAME

GitLab username used to send merge requests for translation updates.

Shihni edhe:

GITLAB_CREDENTIALS, GitLab

2.16.39 GITLAB_TOKEN

Të reja në versionin 4.3.

GitLab personal access token used to make API calls for translation updates.

Shihni edhe:

GITLAB_CREDENTIALS, GitLab, GitLab: Personal access token

2.16.40 GITHUB_CREDENTIALS

Të reja në versionin 4.3.

List for credentials for GitHub servers.

Ndihmëz: 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",
    },
}
```

2.16.41 GITHUB_USERNAME

GitHub username used to send pull requests for translation updates.

Shihni edhe:

GITHUB_CREDENTIALS, GitHub

2.16.42 GITHUB_TOKEN

Të reja në versionin 4.3.

GitHub personal access token used to make API calls to send pull requests for translation updates.

Shihni edhe:

GITHUB_CREDENTIALS, GitHub, Creating a GitHub personal access token
2.16.43 GOOGLE_ANALYTICS_ID

Google Analytics ID to turn on monitoring of Weblate using Google Analytics.

2.16.44 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.

Shënëm: Kjo është e aktivizuar, si parazgjedhje.

2.16.45 HIDE_VERSION

Të reja në versionin 4.3.1.

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.

Shënëm: Kjo është e çaktivizuar, si parazgjedhje.

2.16.46 IP_BEHIND_REVERSE_PROXY

Të reja në versionin 2.14.

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`.

Sinjalizim: Ensure you are actually using a reverse proxy and that it sets this header, otherwise users will be able to fake the IP address.

Shënëm: Kjo sështë e aktivizuar, si parazgjedhje.

Shihni edhe:

Running behind reverse proxy, Rate limiting, `IP_PROXY_HEADER`, `IP_PROXY_OFFSET`
2.16.47 IP_PROXY_HEADER

Të reja në versionin 2.14.
Indicates which header Weblate should obtain the IP address from when IP_BEHIND_REVERSE_PROXY is turned on.
Defaults to HTTP_X_FORWARDED_FOR.
Shihni edhe:
Running behind reverse proxy, Rate limiting, SECURE_PROXY_SSL_HEADER, IP_BEHIND_REVERSE_PROXY, IP_PROXY_OFFSET

2.16.48 IP_PROXY_OFFSET

Të reja në versionin 2.14.
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.

Sinjalizim: Setting this affects the security of your installation, you should only configure it to use trusted proxies for determining IP address.

0, si parazgjedhje.
Shihni edhe:
Running behind reverse proxy, Rate limiting, SECURE_PROXY_SSL_HEADER, IP_BEHIND_REVERSE_PROXY, IP_PROXY_HEADER

2.16.49 LEGAL_URL

Të reja në versionin 3.5.
URL where your Weblate instance shows its legal documents.

Ndihmëz: Useful if you host your legal documents outside Weblate for embedding them inside Weblate, please check Ligjore for details.

Shembull:

LEGAL_URL = "https://weblate.org/terms/*"

Shihni edhe:
PRIVACY_URL
2.16.50 LICENSE_EXTRA

Additional licenses to include in the license choices.

**Shënìm:** Each license definition should be tuple of its short name, a long name and an URL.

Për shembull:

```python
LICENSE_EXTRA = [
    {
        "AGPL-3.0",
        "GNU Affero General Public License v3.0",
        "https://www.gnu.org/licenses/agpl-3.0-standalone.html",
    },
]
```

2.16.51 LICENSE_FILTER

Ndryshuar në versionin 4.3: 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.

**Shënìm:** This filter uses the short license names.

Për shembull:

```python
LICENSE_FILTER = {"AGPL-3.0", "GPL-3.0-or-later"}
```

Following disables the license alert:

```python
LICENSE_FILTER = set()
```

**Shihni edhe:**

alerts

2.16.52 LICENSE_REQUIRED

Defines whether the license attribute in *Component configuration* is required.

**Shënìm:** This is off by default.

2.16.53 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.

**Ndihmëz:** Set this to `False` to allow longer translations (up to 10.000 characters) irrespective of source string length.

**Shënìm:** `True`, siparazgjedhje.
2.16.54 LOCALIZE_CDN_URL and LOCALIZE_CDN_PATH

These settings configure the CDN përkthimesh JavaScript addon. 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.


Shihni edhe:
CDN përkthimesh JavaScript

2.16.55 LOGIN_REQUIRED_URLS

A list of URLs you want to require logging into. (Besides the standard rules built into Weblate).

Ndihmëz: This allows you to password protect a whole installation using:

```python
LOGIN_REQUIRED_URLS = (r"/(.*$)"),
REST_FRAMEWORK["DEFAULT_PERMISSION_CLASSES"] = [
    "rest_framework.permissions.IsAuthenticated"
]
```

Ndihmëz: It is desirable to lock down API access as well, as shown in the above example.

Shihni edhe:
REQUIRE_LOGIN

2.16.56 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:

```python
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
)
```
2.16.57 MATOMO_SITE_ID

ID of a site in Matomo (formerly Piwik) you want to track.

_Shenim:_ This integration does not support the Matomo Tag Manager.

Shihni edhe:  
MATOMO_URL

2.16.58 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.

_Ndihmëz:_ This integration does not support the Matomo Tag Manager.

_Për shembull:_

```
MATOMO_SITE_ID = 1
MATOMO_URL = "https://example.matomo.cloud/
```

Shihni edhe:  
MATOMO_SITE_ID

2.16.59 MT_SERVICES

Ndryshuar në versionin 3.0: The setting was renamed from _MACHINE_TRANSLATION_SERVICES_ to _MT_SERVICES_ to be consistent with other machine translation settings.

List of enabled machine translation services to use.

_Shënim:_ Many of the services need additional configuration like API keys, please check their documentation _Përkthimin nga makina_ for more details.

_Ndihmëz:_ When using Docker container, this configuration is automatically generated based on provided API keys, see _Rregullime mbi përkthimin nga makina_.

```
MT_SERVICES = {
    "weblate.machinery.apertium.ApertiumAPYTranslation",
    "weblate.machinery.deepl.DeepLTranslation",
    "weblate.machinery.glosbe.GlosbeTranslation",
    "weblate.machinery.google.GoogleTranslation",
    "weblate.machinery.libretranslate.LibreTranslateTranslation",
    "weblate.machinery.microsoft.MicrosoftCognitiveTranslation",
    "weblate.machinery.microsoftterminology.MicrosoftTerminologyService",
    "weblate.machinery.mymemory.MyMemoryTranslation",
    "weblate.machinery.tmserver.AmagamaTranslation",
    "weblate.machinery.tmserver.TMServerTranslation",
    "weblate.machinery.yandex.YandexTranslation",
    "weblate.machinery.weblatetm.WeblateTranslation",
    "weblate.machinery.saptranslationalhub.SAPTranslationHub",
}
```

(vazhdon ne faqe pasuese)
Shihni edhe:

Përkthim nga makina, Sugjerime të automatizuara

### 2.16.60 MT_APERTIUM_APY


Shihni edhe:

*Apertium, Përkthim nga makina, Sugjerime të automatizuara*

### 2.16.61 MT_AWS_ACCESS_KEY_ID

Access key ID for Amazon Translate.

Shihni edhe:

*AWS, Përkthim nga makina, Sugjerime të automatizuara*

### 2.16.62 MT_AWS_SECRET_ACCESS_KEY

Kyç API ifshehtë për në Amazon Translate.

Shihni edhe:

*AWS, Përkthim nga makina, Sugjerime të automatizuara*

### 2.16.63 MT_AWS_REGION

Emër rajoni për t’u përdorur për Amazon Translate.

Shihni edhe:

*AWS, Përkthim nga makina, Sugjerime të automatizuara*

### 2.16.64 MT_BAIDU_ID

Client ID for the Baidu Zhiyun API, you can register at [https://api.fanyi.baidu.com/api/trans/product/index](https://api.fanyi.baidu.com/api/trans/product/index)

Shihni edhe:

*Baidu API machine translation, Përkthim nga makina, Sugjerime të automatizuara*
2.16.65 MT_BAIDU_SECRET

Client secret for the Baidu Zhiyun API, you can register at https://api.fanyi.baidu.com/api/trans/product/index

Shihni edhe:

Baidu API machine translation, Përkthim nga makina, Sugjerime të automatizuara

2.16.66 MT_DEEPL_API_URL

Ndryshuar në versionin 4.7: The full API URL is now configured to allow using the free plan. Before, it was only possible to configure the API version using MT_DEEPL_API_VERSION.

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:


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.

Shihni edhe:

DeepL, Përkthim nga makina, Sugjerime të automatizuara

2.16.67 MT_DEEPL_KEY

API key for the DeepL API, you can register at https://www.deepl.com/pro.html

Shihni edhe:

DeepL, Përkthim nga makina, Sugjerime të automatizuara

2.16.68 MT_LIBRETRANSLATE_API_URL

Të reja në versionin 4.7.1.

API URL for the LibreTranslate instance to use.

https://libretranslate.com/ (official public instance) Requires an API key to use outside of the website.

Mirrors are documented on the LibreTranslate GitHub repository, some of which can be used without authentication:

https://github.com/LibreTranslate/LibreTranslate#user-content-mirrors

Shihni edhe:

LibreTranslate, Përkthim nga makina, Sugjerime të automatizuara
2.16.69 MT_LIBRETRANSLATE_KEY

Të reja në versionin 4.7.1.

API key for the LibreTranslate instance specified in MT_LIBRETRANSLATE_API_URL.

Shihni edhe:
LibreTranslate, Përkthim nga makina, Sugjerime të automatizuara

2.16.70 MT GOOGLE_KEY

API key for Google Translate API v2, you can register at https://cloud.google.com/translate/docs

Shihni edhe:
Google Translate, Përkthim nga makina, Sugjerime të automatizuara

2.16.71 MT GOOGLE CREDENTIALS

API v3 JSON credentials file obtained in the Google cloud console. Please provide a full OS path. Credentials are per service-account affiliated with certain project. Please check https://cloud.google.com/docs/authentication/getting-started for more details.

2.16.72 MT GOOGLE PROJECT

Google Cloud API v3 project id with activated translation service and billing activated. Please check https://cloud.google.com/appengine/docs/standard/nodejs/building-app/creating-project for more details

2.16.73 MT GOOGLE LOCATION

API v3 Google Cloud App Engine may be specific to a location. Change accordingly if the default global fallback does not work for you.

Please check https://cloud.google.com/appengine/docs/locations for more details

Shihni edhe:
Google Translate API V3 (Advanced)

2.16.74 MT MICROSOFT_BASE_URL

Region base URL domain as defined in the «Base URLs» section.
Defaults to api.cognitive.microsofttranslator.com for Azure Global.
For Azure China, please use api.translator.azure.cn.
2.16.75 **MT_MICROSOFT_COGNITIVE_KEY**

Client key for the Microsoft Cognitive Services Translator API.

**Shihni edhe:**

*Microsoft Cognitive Services Translator, Përkkhim nga makina, Sugjerime të automatizuara, Cognitive Services - Text Translation API, Microsoft Azure Portal*

2.16.76 **MT_MICROSOFT_REGION**

Region prefix as defined in the «Authenticating with a Multi-service resource» section.

2.16.77 **MT_MICROSOFT_ENDPOINT_URL**

Region endpoint URL domain for access token as defined in the «Authenticating with an access token» section.

Defaults to api.cognitive.microsoft.com for Azure Global.

For Azure China, please use your endpoint from the Azure Portal.

2.16.78 **MT_MODERNMT_KEY**

Kyç API për motorin e përkthimit nga makina ModernMT.

**Shihni edhe:**

*ModernMT MT_MODERNMT_URL*

2.16.79 **MT_MODERNMT_URL**

URL of ModernMT. It defaults to https://api.modernmt.com/ for the cloud service.

**Shihni edhe:**

*ModernMT MT_MODERNMT_KEY*

2.16.80 **MT_MYMEMORY_EMAIL**

MyMemory identification e-mail address. It permits 1000 requests per day.

**Shihni edhe:**

*MyMemory, Përkkhim nga makina, Sugjerime të automatizuara, MyMemory: API technical specifications*

2.16.81 **MT_MYMEMORY_KEY**

MyMemory access key for private translation memory, use it with **MT_MYMEMORY_USER**.

**Shihni edhe:**

*MyMemory, Përkkhim nga makina, Sugjerime të automatizuara, MyMemory: API key generator*
2.16.82 MT_MYMEMORY_USER

MyMemory user ID for private translation memory, use it with $MT_MYMEMORY_KEY$.

Shihni edhe:
MyMemory, Përkkthim nga makina, Sugjerime të automatizuarra, MyMemory: API key generator

2.16.83 MT_NETEASE_KEY

App key for NetEase Sight API, you can register at https://sight.youdao.com/

Shihni edhe:
NetEase Sight API machine translation, Përkkthim nga makina, Sugjerime të automatizuarra

2.16.84 MT_NETEASE_SECRET

App secret for the NetEase Sight API, you can register at https://sight.youdao.com/

Shihni edhe:
NetEase Sight API machine translation, Përkkthim nga makina, Sugjerime të automatizuarra

2.16.85 MT_TMSERVER

URL-ja ku xhiron shërbyesi i kujtesës së përkthimeve.

Shihni edhe:
tmserver, Përkkthim nga makina, Sugjerime të automatizuarra, tmserver

2.16.86 MT_YANDEX_KEY

API key for the Yandex Translate API, you can register at https://yandex.com/dev/translate/

Shihni edhe:
Yandex Translate, Përkkthim nga makina, Sugjerime të automatizuarra

2.16.87 MT_YOUDAO_ID


Shihni edhe:
Youdao Zhiyun API machine translation, Përkkthim nga makina, Sugjerime të automatizuarra

2.16.88 MT_YOUDAO_SECRET


Shihni edhe:
Youdao Zhiyun API machine translation, Përkkthim nga makina, Sugjerime të automatizuarra
2.16.89 **MT_SAP_BASE_URL**

API URL to the SAP Translation Hub service.

Shihni edhe:
*SAP Translation Hub, Përkthim nga makina, Sugjerime të automatizuara*

2.16.90 **MT_SAP_SANDBOX_APIKEY**

API key for sandbox API usage

Shihni edhe:
*SAP Translation Hub, Përkthim nga makina, Sugjerime të automatizuara*

2.16.91 **MT_SAP_USERNAME**

Emri juaj i përdoruesit SAP

Shihni edhe:
*SAP Translation Hub, Përkthim nga makina, Sugjerime të automatizuara*

2.16.92 **MT_SAP_PASSWORD**

Fjalëkalimi juaj SAP

Shihni edhe:
*SAP Translation Hub, Përkthim nga makina, Sugjerime të automatizuara*

2.16.93 **MT_SAP_USE_MT**

Whether to also use machine translation services, in addition to the term database. Possible values: True or False

Shihni edhe:
*SAP Translation Hub, Përkthim nga makina, Sugjerime të automatizuara*

2.16.94 **NEARBY_MESSAGES**

How many strings to show around the currently translated string. This is just a default value, users can adjust this in *Profil përdoruesi.*

2.16.95 **DEFAULT_PAGE_LIMIT**

Të reja në versionin 4.7.

Default number of elements to display when pagination is active.
2.16.96 PAGURE_CREDENTIALS

Të rea në versionin 4.3.2.
List for credentials for Pagure servers.

Ndihmëz: 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",
    },
}
```

2.16.97 PAGURE_USERNAME

Të rea në versionin 4.3.2.
Pagure username used to send merge requests for translation updates.

Shihni edhe:
PAGURE_CREDENTIALS, Pagure

2.16.98 PAGURE_TOKEN

Të rea në versionin 4.3.2.
Pagure personal access token used to make API calls for translation updates.

Shihni edhe:
PAGURE_CREDENTIALS, Pagure, Pagure API

2.16.99 PRIVACY_URL

Të rea në versionin 4.8.1.
URL where your Weblate instance shows its privacy policy.

Ndihmëz: Useful if you host your legal documents outside Weblate for embedding them inside Weblate, please check Ligjore for details.

Shembull:

```
PRIVACY_URL = "https://weblate.org/terms/"
```

Shihni edhe:
LEGAL_URL
2.16.100 RATELIMIT_ATTEMPTS

Të reja në versionin 3.2.
Maximum number of authentication attempts before rate limiting is applied.
5, si parazgjedhje.
Shihni edhe:
Rate limiting, RATELIMIT_WINDOW, RATELIMIT_LOCKOUT

2.16.101 RATELIMIT_WINDOW

Të reja në versionin 3.2.
How long authentication is accepted after rate limiting applies.
An amount of seconds defaulting to 300 (5 minutes).
Shihni edhe:
Rate limiting, RATELIMIT_ATTEMPTS, RATELIMIT_LOCKOUT

2.16.102 RATELIMIT_LOCKOUT

Të reja në versionin 3.2.
How long authentication is locked after rate limiting applies.
An amount of seconds defaulting to 600 (10 minutes).
Shihni edhe:
Rate limiting, RATELIMIT_ATTEMPTS, RATELIMIT_WINDOW

2.16.103 REGISTRATION_ALLOW_BACKENDS

Të reja në versionin 4.1.
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.
Shembull:

REGISTRATION_ALLOW_BACKENDS = ["azuread-oauth2", "azuread-tenant-oauth2"]

Ndihmëz: The backend names match names used in URL for authentication.

Shihni edhe:
REGISTRATION_OPEN, Mirëfillësint
2.16.104 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 user enters their e-mail address:

- Regjistrim llogari të re.
- Rimarrje fjalëkalimesh.
- Shtim email-i te një llogari.
- Formular kontaktesh për përdorues që s’kanë bërë hyrjen.

2.16.105 REGISTRATION_EMAIL_MATCH

Të reja në versionin 2.17.

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$"
```

2.16.106 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).

Shënim: If using third-party authentication methods such as Mirëfilltësim LDAP, it just hides the registration form, but new users might still be able to sign in and create accounts.

Shihni edhe: 
REGISTRATION_ALLOW_BACKENDS, REGISTRATION_EMAIL_MATCH, Mirëfilltësim

2.16.107 REPOSITORY_ALERT_THRESHOLD

Të reja në versionin 4.0.2.

Threshold for triggering an alert for outdated repositories, or ones that contain too many changes. Defaults to 25.

Shihni edhe:
alerts
2.16.108 REQUIRE_LOGIN

Të reja në versionin 4.1.

This enables LOGIN_REQUIRED_URLS and configures REST framework to require authentication for all API endpoints.

Shënë: This is implemented in the Sample configuration. For Docker, use WEBLATE_REQUIRE_LOGIN.

2.16.109 SENTRY_DSN

Të reja në versionin 3.9.

Sentry DSN to use for Collecting error reports.

Shihni edhe:
Django integration for Sentry

2.16.110 SESSION_COOKIE_AGE_AUTHENTICATED

Të reja në versionin 4.3.

Set session expiry for authenticated users. This complements SESSION_COOKIE_AGE which is used for unauthenticated users.

Shihni edhe:
SESSION_COOKIE_AGE

2.16.111 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.

2.16.112 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.

Shembuj:

```
# Production site with domain name
SITE_DOMAIN = "weblate.example.com"

# Local development with IP address and port
SITE_DOMAIN = "127.0.0.1:8000"
```

Shënë: 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.
The Weblate Manual, Hedhje Në Qarkullim 4.9

Ndihmëz: On a Docker container, the site domain is configured through `WEBLATE_ALLOWED_HOSTS`.

Shihni edhe:

Set correct site domain, Allowed hosts setup, Formësoni si duhet HTTPS-në `WEBLATE_SITE_DOMAIN`, `ENABLE_HTTPS`

2.16.113 SITE_TITLE

Site title to be used for the website and sent e-mails.

2.16.114 SPECIAL_CHARS

Additional characters to include in the visual keyboard, *Tastierë pamore*.

Vleraparazgjedhjeëshëtë:

```
SPECIAL_CHARS = ("\t", "\n", "\u00a0", "_.")
```

2.16.115 SINGLE_PROJECT

Të reja në versionin 3.8.

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.

Ndryshuar në versionin 3.11: The setting now also accepts a project slug, to force displaying that single project.

Shembull:

```
SINGLE_PROJECT = "test"
```

2.16.116 STATUS_URL

The URL where your Weblate instance reports its status.

2.16.117 SUGGESTION_CLEANUP_DAYS

Të reja në versionin 3.2.1.

Automatically deletes suggestions after a given number of days. Defaults to `None`, meaning no deletions.

2.16.118 UPDATE_LANGUAGES

Të reja në versionin 4.3.2.

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`.

Shihni edhe:

*Built-in language definitions*
2.16.119 URL_PREFIX

This setting allows you to run Weblate under some path (otherwise it relies on being run from the webserver root).

Shënim: 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`.

Ndihmëz: Prefiksi duhet të fillojë me një `/`.

Shembull:

```python
URL_PREFIX = "/translations"
```

Shënim: This setting does not work with Django’s built-in server, you would have to adjust `urls.py` to contain this prefix.

2.16.120 VCS_BACKENDS

Configuration of available VCS backends.

Shënim: Weblate tries to use all supported back-ends you have the tools for.

Ndihmëz: You can limit choices or add custom VCS back-ends by using this.

```python
VCS_BACKENDS = ("weblate.vcs.git.GitRepository",)
```

Shihni edhe:

Integrim kontrolli versioni

2.16.121 VCS_CLONE_DEPTH

Të reja në versionin 3.10.2.

Configures how deep cloning of repositories Weblate should do.

Shënim: 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 `Shtesa`), you might want to increase the depth or turn off shallow clones completely by setting this to 0.

Ndihmëz: 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:

```python
VCS_CLONE_DEPTH = 0
```
2.16.122 WEBLATE_ADDONS

List of addons available for use. To use them, they have to be enabled for a given translation component. By default this includes all built-in addons, when extending the list you will probably want to keep existing ones enabled, for example:

```
WEBLATE_ADDONS = (  
    # Built-in 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",  
    "weblate.addons.cleanup.CleanupAddon",  
    "weblate.addons.consistency.LanguageConsistencyAddon",  
    "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",  
)
```

Shënim: Removing the addon from the list does not uninstall it from the components. Weblate will crash in that case. Please uninstall addon from all components prior to removing it from this list.

Shihni edhe:

```
# Shihes,
DEFAULT_ADDONS
```

2.16.123 WEBLATE_EXPORTERS

Të reja në versionin 4.2.

List of available exporters offering downloading translations or glossaries in various file formats.

Shihni edhe:

```
# Formate të mbështetë kartelash
```
2.16.124 WEBLATE_FORMATS

Të reja në versionin 3.0.
Listë formatesh kartelash që mund të përdoren.

**Shënim:** The default list already has the common formats.

**Shihni edhe:**

*Formate të mbuluar kartelash*

2.16.125 WEBLATE_GPG_IDENTITY

Të reja në versionin 3.1.
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.

**Shihni edhe:**

*Signing Git commits with GnuPG*

2.16.126 WEBSITE_REQUIRED

Defines whether *Sajt projekt* 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:

```python
# Copyright © 2012 - 2021 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
```

(vazhdon në faqen pasuese)
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",
        # 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,
    }
}

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"),
    ("bg", "Български"),
    ("br", "Brezhoneg"),
    ("ca", "Català"),
    ("cs", "Čeština"),
    ("da", "Dansk"),
    ("de", "Deutsch"),
    ("en", "English"),
    ("el", "ΕΛΛΗΝΙΚΑ"),
    ("en-gb", "English (United Kingdom)"),
    ("es", "Español"),
    ("fi", "Suomi"),
    ("fr", "Français"),
    ("gl", "Galego"),
    ("he", "עברית"),
    ("hu", "Magyar"),
    ("hr", "Hrvatski"),
    ("id", "Indonesia"),
    ("is", "íslenska"),
    ("it", "Italiano"),
    ("ja", "日本"),
    ("ka", "ქართული"),
    ("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", "ภาษาไทย"),
    ("tr", "Türkçe"),
    ("uk", "Українська"),
    ("zh-hans", "中文"),
    ("zh-hant", "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.
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 = ""

_TEMPLATE_LOADERS = [  
    "django.template.loaders.filesystem.Loader",  
    "django.template.loaders.app_directories.Loader",  
]

if not DEBUG:  
    _TEMPLATE_LOADERS = [("django.template.loaders.cached.Loader", _TEMPLATE_ 
Loads)]

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",
  ],
  "loaders": _TEMPLATE_LOADERS,
}
]

# 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.BitbucketOAuth",
  # "social_core.backends.suse.OpenSUSEOpenId",
  # "social_core.backends.ubuntu.UbuntuOpenId",
  # "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_BITBUCKET_KEY = ""
SOCIAL_AUTH_BITBUCKET_SECRET = ""
SOCIAL_AUTH_BITBUCKET_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
The Weblate Manual, Hedhje Në Qarkullim 4.9

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.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",
    "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},
}
The Weblate Manual, Hedhje Në Qarkullim 4.9

(vazhduar nga faqja e mëparshme)

{ "NAME": "django.contrib.auth.password_validation.CommonPasswordValidator" },
{ "NAME": "django.contrib.auth.password_validation.NumericPasswordValidator" },
{ "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
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",
    "weblate.middleware.MessageMiddleware",
    "weblate.middleware.clickjacking.XFrameOptionsMiddleware",
    "social_django.middleware.SocialAuthExceptionMiddleware",
    "weblate.accounts.middleware.RequireLoginMiddleware",
    "weblate.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",
"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.

```json
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",
        },
        "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,
        },
    }
}
```

(vazhdon nefaqen pasuese)
# 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.deepi.DeepLTranslation",
    # "weblate.machinery.glosbe.GlosbeTranslation",
    # "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.netease.NeteaseSightTranslation",
    # "weblate.machinery.tmserver.AmagamaTranslation",
    # "weblate.machinery.tmserver.TMServerTranslation",
    # "weblate.machinery.yandex.YandexTranslation",
    # "weblate.machinery.saptranslacionhub.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
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:
# 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 referrer 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 cookie settings to language cookie as well
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

# 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",  
#     "weblate.checks.format.PerlFormatCheck",  
#     "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.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.MaxLengthCheck",  
#     "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",  
# )
# 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",
#    "weblate.addons.cleanup.CleanupAddon",
#    "weblate.addons.cleanup.RemoveBlankAddon",
#    "weblate.addons.consistency.LanguageConsistencyAddon",
#    "weblate.addons.discovery.DiscoveryAddon",
#    "weblate.addons.autotranslate.AutoTranslateAddon",
#    "weblate.addons.flags.SourceEditAddon",
#    "weblate.addons.flags.TargetEditAddon",
#    "weblate.addons.flags.SameEditAddon",
#    "weblate.addons.generate.GenerateFileAddon",
#    "weblate.addons.generate.PseudolocaleAddon",
#    "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"

# 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

```python
if REQUIRE_LOGIN:
    LOGIN_REQUIRED_URLS = (r"/(.*)$",)
```

# In such case you will want to include some of the exceptions

```python
# 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
# )
```

# Silence some of the Django system checks

```python
SILENCED_SYSTEM_CHECKS = [
    # We have modified django.contrib.auth.middleware.AuthenticationMiddleware
    # as weblate.accounts.middleware.AuthenticationMiddleware
    "admin.E408"
]
```

# Celery worker configuration for testing

```python
# 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

```python
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_emails": {"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

```python
DATABASE_BACKUP = "plain"
```

# Enable auto updating

```python
AUTO_UPDATE = False
```

# PGP commits signing

```python
WEBLATE_GPG_IDENTITY = None
```

# Third party services integration
2.18 Management commands

Shënim: Running management commands under a different user than the one running your webserver can result in files getting wrong permissions, please check Leje sistemi kartelash 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 importusers /tmp/users.json
```

Shihni edhe:

Installing using Docker, Installing on Debian and Ubuntu, Installing on SUSE and openSUSE, Installing on RedHat, Fedora and CentOS, Installing from sources

2.18. Management commands
2.18.2 add_suggestions

`weblate add_suggestions <project> <component> <language> <file>`

Të reja në versionin 2.5.

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).

Shembull:
```
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>`

Të reja në versionin 2.5.

Ndryshuar në versionin 4.6: Added parameter for translation mode.

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 Jo i njëraftshëm).

`--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.

Shembull:
```
weblate auto_translate --user nijel --inconsistent --source weblate/application ←weblate website cs
```

Shihni edhe:

_ Përktihim i automatizuar_
2.18.4 celery_queues

**weblate celery_queues**

Të reja në versionin 3.7.
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.

**Shënim:**  
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.

**Shihni edhe:**

*Xhirim aktesh mirëmbajtje, 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.

**Shihni edhe:**

*Xhirim aktesh mirëmbajtje*
## 2.18.9 createadmin

```plaintext
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).

Ndryshuar në versionin 2.9: Added parameters `--username`, `--email`, `--name` and `--update`.

## 2.18.10 dump_memory

```plaintext
weblate dump_memory
```

Të reja në versionin 2.20.

Export a JSON file containing Weblate Translation Memory content.

**Shihni edhe:**

*Kujtesë Përkthimesh, Weblate Translation Memory Schema*

## 2.18.11 dumpuserdata

```plaintext
weblate dumpuserdata <file.json>
```

Dumps userdata to a file for later use by `importuserdata`

*Ndihmëz:* This comes in handy when migrating or merging Weblate instances.

## 2.18.12 import_demo

```plaintext
weblate import_demo
```

Të reja në versionin 4.1.

Creates a demo project with components based on `<https://github.com/WeblateOrg/demo>`.

This can be useful when developing Weblate.
2.18.13 import_json

**weblate import_json <json-file>**

Të reja në versionin 2.7.

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.
```

Ndryshuar në versionin 2.9: 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",
    "template": "android/values/strings.xml",
    "repo": "weblate://test/test",
    "file_format": "aresource"
  }
]
```

Shihni edhe:

**import_memory**

2.18.14 import_memory

**weblate import_memory <file>**

Të reja në versionin 2.20.

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.
2.18.15 import_project

`weblate import_project <project> <gitrepo> <branch> <filemask>`

Ndryshuar në versionin 3.0: The import_project command is now based on the Pikasje përbërësisht add-on, leading to some changes in behavior and what parameters are accepted.

Batch imports components into project based on filemask.

**<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.

Për shembull: {{ component }}/res/values/string.xml

**--new-base-template** TEMPLATE

Customize the base file for addition of new translations.

Për shembull: {{ component }}/ts/en.ts

**--file-format** FORMAT

You can also specify the file format to use (see Formate të mbuluar kartelash), 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'
```

Shihni edhe:

More detailed examples can be found in the starting chapter, alternatively you might want to use `import_json`.  

2.18. Management commands
2.18.16 importuserdata

weblate importuserdata <file.json>

Imports user data from a file created by dumpuserdata

2.18.17 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

2.18.18 install_addon

Të reja në versionin 3.2.

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
JSON encoded configuration of an add-on.

--update
Update the existing add-on configuration.

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 Përshatni përfundimin e gettext-it for all components:

weblate install_addon --addon weblate.gettext.customize --config '{"width": -1}' --update --all

Shihni edhe:
Shtesa

2.18.19 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.l10n.cz/Slovn%C3%ADk_s_n%C3%A1zvy_jazyk%C5%AF>. 
2.18.20 list_translators

weblate list_translators <project|project/component>

Lists translators by contributed language for the given project:

<table>
<thead>
<tr>
<th>Language</th>
<th>Translator</th>
<th>Email</th>
</tr>
</thead>
<tbody>
<tr>
<td>[French]</td>
<td>Jean Dupont</td>
<td><a href="mailto:jean.dupont@example.com">jean.dupont@example.com</a></td>
</tr>
<tr>
<td></td>
<td>John Doe</td>
<td><a href="mailto:jd@example.com">jd@example.com</a></td>
</tr>
</tbody>
</table>

--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.

2.18.21 list_versions

weblate list_versions

Lists all Weblate dependencies and their versions.

2.18.22 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.

Shënimi: 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.

2.18.23 lock_translation

weblate lock_translation <project|project/component>

Prevents further translation of a component.

Ndihmëz: 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.

Shihni edhe:
unlock_translation
2.18.24 move_language

weblate move_language source target

Të reja në versionin 3.0.

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.

Shembull:

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.

2.18.25 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.

Shënëm: Weblate pushes changes automatically if Push on commit in Component configuration is turned on, which is the default.

2.18.26 unlock_translation

weblate unlock_translation <project|project/component>

Unlocks a given component, making it available for translation.

Ndihmëz: 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.

Shihni edhe:

lock_translation
2.18.27 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 *Kontroll hyrjesh te projekti*.

**Shihni edhe:**

*Listë privilegjesh*

2.18.28 setuplang

**weblate setuplang**

Updates list of defined languages in Weblate.

---no-update
   Turns off automatic updates of existing languages (only adds new ones).

2.18.29 updatechecks

**weblate updatechecks <project|project/component>**

Updates all checks for all strings.

**Ndihmëz:** 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.

2.18.30 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.

**Shënim:** Usually it is better to configure hooks in the repository to trigger *Hook-e njoftimesh*, instead of regular polling by *updategit*.

---

2.18. Management commands 357
2.19 Lajmërime

Ndryshuar në versionin 4.0: 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 posted on each level in the Manage menu, using Post announcement:

It can be also added using the admin interface:
The announcements are then shown based on their specified context:

S’është specifikuar kontekst

Shfaqur te pulti (faqja ardhjesh).

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 Lista Përbërësisht

Specify multiple lists of components to appear as options on the user dashboard, from which users can pick one as their default view. See Pult to learn more.

Ndryshua në versionin 2.20: 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.

Ndryshua në versionin 2.13: Change dashboard settings for anonymous users from the admin interface, altering what dashboard is presented to unauthenticated users.

2.20.1 Automatic component lists

Të reja në versionin 2.13.

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.

Ndihmëz: 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:
2.21 Optional Weblate modules

Several optional modules are available for your setup.
2.21.1 Git exporter

Të reja në versionin 2.10.
Provides you read-only access to the underlying Git repository using HTTP(S).

**Instalim**

1. Add `weblate.gitexport` to installed apps in `settings.py`:

```python
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/
```

**Ndihmëz**: 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 Faturim

Të reja në versionin 2.4.
This is used on Hosted Weblate to define billing plans, track invoices and usage limits.

**Instalim**

1. Add `weblate.billing` to installed apps in `settings.py`:

```python
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 Profil përdoruesit.

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 Ligjore

Të reja në versionin 2.15.

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

Shënim: 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.

Instalim

1. Add weblate.legal to installed apps in settings.py:

```python
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:

```bash
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

Shihni edhe:

*Avatar caching, 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:
- Suggestions from unauthenticated users
- Project and component descriptions and links

Shënëm: This (among other things) relies on IP address of the client, please see *Running behind reverse proxy* for properly configuring that.

Shihni edhe:

*Running behind reverse proxy, AKISMET_API_KEY, WEBLATE_AKISMET_API_KEY*

2.21.6 Signing Git commits with GnuPG

Të reja në versionin 3.1.

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:
2. Alternatively you can also import existing keys into Weblate, just set `HOME=$DATA_DIR/home` when invoking gpg.

Shihni edhe:

`WEBLATE_GPG_IDENTITY`

### 2.21.7 Rate limiting

Ndryshuar në versionin 3.2: The rate limiting now accepts more fine-grained configuration.

Ndryshuar në versionin 4.6: 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:

<table>
<thead>
<tr>
<th>Emër</th>
<th>Fokus</th>
<th>Allowed attempts</th>
<th>Rate limit window</th>
<th>Lockout period</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regjistrim</td>
<td>REGISTRATION</td>
<td>5</td>
<td>300</td>
<td>600</td>
</tr>
<tr>
<td>Sending message to admins</td>
<td>MESSAGE</td>
<td>5</td>
<td>300</td>
<td>600</td>
</tr>
<tr>
<td>Password authentication on sign in</td>
<td>LOGIN</td>
<td>5</td>
<td>300</td>
<td>600</td>
</tr>
<tr>
<td>Sitewide search</td>
<td>SEARCH</td>
<td>6</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>Translating</td>
<td>TRANSLATE</td>
<td>30</td>
<td>60</td>
<td>600</td>
</tr>
<tr>
<td>Adding to glossary</td>
<td>GLOSSARY</td>
<td>30</td>
<td>60</td>
<td>600</td>
</tr>
<tr>
<td>Starting translation into a new language</td>
<td>LANGUAGE</td>
<td>2</td>
<td>300</td>
<td>600</td>
</tr>
<tr>
<td>Creating new project</td>
<td>PROJECT</td>
<td>5</td>
<td>600</td>
<td>600</td>
</tr>
</tbody>
</table>
If a user fails to log 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 `RATERLIMIT_ATTEMPTS` becomes `WEBLATE_RATELIMIT_ATTEMPTS`.

The API has separate rate limiting settings, see *API rate limiting*.

**Shihni edhe:**

*Rate limiting, Running behind reverse proxy, API rate limiting*

### 2.21.8 Fedora Messaging integration

Fedora Messaging is AMQP-based publisher for all changes happening in Weblate. You can hook additional services on changes happening in Weblate using this.

The Fedora Messaging integration is available as a separate Python module `weblate-fedora-messaging`. Please see [https://github.com/WeblateOrg/fedora_messaging/](https://github.com/WeblateOrg/fedora_messaging/) for setup instructions.

### 2.22 Përshtatje e Weblate-it

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.

**Sinjalizim:** 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.

**Shihni edhe:**

*Dhënie ndihmese te Weblate*

#### 2.22.1 Krijim i një moduli Python

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:

   ```python
   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",
   )
   ```

   (vazhdon ne laqen pasuese)
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 Ndryshimi i stemës

1. Create a simple Django app containing the static files you want to overwrite (see Krijim i një moduli Python).
   Branding appears in the following files:
   - `icons/weblate.svg` Stemë e shfaqur te shtylla e lëvizjeve.
   - `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` Epërdorur në email-e njoftimesh.
2. Add it to `INSTALLED_APPS`:

   ```python
   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.

**Shihni edhe:**

Managing static files (e.g. images, JavaScript, CSS), Serving static files
2.22.3 Custom quality checks, addons and auto-fixes

To install your code for Custom automatic fixes, Së të shkruani kontrollet tuaja or Writing add-on in Weblate:

1. Place the files into your Python module containing the Weblate customization (see Krijim i një moduli Python).
2. Add its fully-qualified path to the Python class in the dedicated settings (WEBLATE_ADDONS, CHECK_LIST or AUTOFIX_LIST):

```python
# Checks
CHECK_LIST += ("weblate_customization.checks.FooCheck",)

# Autofixes
AUTOFIX_LIST += ("weblate_customization.autofix.FooFixer",)

# Add-ons
WEBLATE_ADDONS += ("weblate_customization.addons.ExamplePreAddon",)
```

Shihni edhe:

Custom automatic fixes, Së të shkruani kontrollet tuaja, Writing add-on, Executing scripts from add-on

2.23 Ndërfaqe administrimi

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:

It includes basic overview of your Weblate:

- Support status, see Getting support for Weblate
- Backups, see Kopjeraujtje dhe zhvendosje e Weblate
- Shared translation memory, see Kujtesë Përkthimesh
- Performance report to review Weblate health and length of Celery queues
• SSH keys management, see Depo SSH
• Alerts overview for all components, see alerts

2.23.1 Ndërfaqe përgjegjësi Django

Sinjalizim: Will be removed in the future, as its use is discouraged—most features can be managed directly in Weblate.

Here you can manage objects stored in the database, such as users, translations and other settings:
### Weblate administration

#### Site administration

- Reports
- Weblate support status
- Status of repositories
- SSO keys
- Performance report
- Translation-memory

#### Accounts

- Audit log entries
- User profiles
- Verified e-mails

#### AUTH TOKEN

- Tokens

#### AUTHENTICATION

- Groups
- Roles
- Users

#### BILLING

- Billing plans
- Customer billings
- Invoices

#### CORE

- First groups
- Fasts

#### LEGAL

- TOS agreements

#### PYTHON SOCIAL AUTH

- Associations
- Nonces
- User social auths

#### SCREENSHOTS

- Screenshots

#### TRANSLATION MEMORY

- Translation-memory entries

#### WEBLATE CONFIGURATION

- Settings

#### WEBLATE LANGUAGES

- Languages

#### WEBLATE TRANSLATORS

- Announcements
- Component bits
- Components
- Contributor agreements
- Projects

---

<table>
<thead>
<tr>
<th>Recent actions</th>
<th>My actions</th>
<th>None available</th>
</tr>
</thead>
</table>

---

Kapitulli 2. Dokumentime përgjegjësi
In the *Reports* section, you can check the status of your site, tweak it for *Production setup*, or manage SSH keys used to access *Hyrje në depo*.

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 *Language definitions*.

**Shtimi i një projekti**

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
Home: Weblate translations | Projects | Add Project

Add Project

Required fields are marked in bold.

**Project name:**
WeblateOrg
Display name: Webtranslate

**URL slug:**
weblateorg
Name used in URLs and filenames.

**Project website:**
https://weblate.org/
Main website of translated project.

**Translation instructions:**
https://weblate.org/contributed/
You can use Markdown and mention users by @username.

- Set "Language-Team" header
  until Weblate adds the "Language-Team" file header of your project.

- Use shared translation memory
  Use the pool of shared translations between projects.

- Contribute to shared translation memory
  Contributes to the pool of shared translations between projects.

**Access control**

- **Protected**
  How to restrict access to the project is detailed in the documentation.

- **Enable reviewers**
  Requires dedicated reviewers to approve translations.

- **Enable source reviewers**
  Requires dedicated reviewers to approve source strings.

- **Enable hooks**
  Whether to allow updating this repository by remote hooks.

**Language aliases:**

Contra-separated list of language code mappings, for example: es,de,es,vn,de

[Save and add another] [Save and continue editing] [SAVE]

---

**2.23. Ndëŕfaqe adminstrimi**
Shihni edhe:

*Project configuration*

**Përbërës dygjuhësh**

Once you have added a project, translation components can be added to it. (See *Component configuration* for info regarding individual parameters):
Shihni edhe:

*Component configuration, Formate dygjuhësh dhe njëgjuhësh*

**Përxbërës njëgjuhësh**

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):
2.23. Ndërfaqe administrimi
Shihni edhe:

Component configuration, Formate dyqanësh dhe njëqanësh

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

Të reja në versionin 3.8.

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.

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 Zbuloni Weblate-in 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
- Zbuloni Weblate-in

Ndihmëz: Purchased support packages are already activated upon purchase, and can be used without integrating them.

2.24.4 Zbuloni Weblate-in

Të reja në versionin 4.5.2.

Shënim: Kjo veçori gjendet aktualisht në version të hershëm beta.

Discover Weblate is an opt-in service that makes it easier for users to find Weblate servers and communities. Users can browse registered services on <https://weblate.org/discover/>, and find there projects to contribute.

Gettung listed

Ndihmëz: Participating in Discover Weblate makes Weblate submit some information about your server, please see Data submitted to the Weblate.

To list your server with an active support subscription (see Integrating support) in Discover Weblate all you need to do is turn this on in the management panel:
Listing your server without a support subscription in Discover Weblate:

1. Register yourself at <https://weblate.org/user/>
2. Register your Weblate server in the discovery database at <https://weblate.org/subscription/discovery/>
3. Confirm the service activation in your Weblate and turn on the discovery listing in your Weblate management page using *Enable discovery* button:
Customizing listing

You can customize the listing by providing a text and image (570 x 260 pixels) at <https://weblate.org/user/>.

2.25 Dokumente ligjore

**Shënim:** 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 dhe kontrole të tjera eksporti

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 Kontrolle fshehtëzimi të ShBA-ee

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** Përdorur nga Git

**OpenSSL** Përdorur nga Python dhe 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)

Shihni edhe:

Export Controls (EAR) on Open Source Software
3.1 Dhënie ndihmese te Weblate

There are dozens of ways to improve Weblate. You can choose the one you feel comfortable with, be it coding, graphics design, documentation, sponsorship, or an idea:

- Njoftim problemesh në Weblate
- Starting contributing code to Weblate
- Përkthim i Weblate-it
- Contribute to Weblate documentation
- Weblate discussions
- Financim i zhvillimit të Weblate-it

3.1.1 Përkthim i Weblate-it

Weblate is continually being translated using Weblate itself. Feel free to take your part in the effort of making Weblate available in as many human languages as possible. It brings Weblate closer to its users!

If you find a possible mistake in the source string, you can mark it with a comment in the Weblate editor. This way, it can be discussed and corrected. If you’re certain, you can also click on the link in the Source string location section and submit a PR with your correction.

3.1.2 Contribute to Weblate documentation

You are welcome to improve the documentation page of your choice. Do it easily by clicking the Edit on GitHub button in the top-right corner of the page.

Please respect these guidelines while writing:

1. Don’t remove part of the documentation if it’s valid.
2. Use clear and easily-understandable language. You are writing tech docs, not a poem. Not all docs readers are native speakers, be thoughtful.
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. Verify your changes by performing described actions while following the docs.

5. Send PR with changes in small chunks to make it easier and quicker to review and merge.

6. If you want to rewrite and change the structure of a big article, do it in two steps:
   1. Rewrite
   2. Once the rewrite is reviewed, polished, and merged, change the structure of the paragraphs in another PR.

Ndihmëz: You can translate the docs.

### 3.1.3 Extending built-in language definitions

The language definitions are in the weblate-language-data repository. You are welcome to add missing language definitions to languages.csv, other files are generated from that file.

### 3.1.4 Weblate discussions

If you have an idea and not sure if it's suitable for an issue, don’t worry. You can join the community in GitHub discussions.

### 3.1.5 Financim i zhvillimit të Weblate-it

You can boost Weblate’s development on the donate page. Funds collected there are used to enable gratis hosting for libre software projects and further development of Weblate. Please check the donate page for options, such as funding goals and the rewards you get as a proud funder.

**Backers who have funded Weblate**

Listë e Përkrabëseve të Weblate-it:
- 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 Starting contributing code to Weblate

Understand the Weblate source code by going through Kod burim i Weblate-it, Pjesa e dukshme e Weblate-it and Gjërat e brendshme të Weblate-it.

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 Xhirim lokalisht i Weblate-it

The most comfortable approach to get started with Weblate development is to follow Installing from sources. It will get you a virtualenv with editable Weblate sources.

1. Clone the Weblate source code:

```bash
git clone https://github.com/WeblateOrg/weblate.git
cd weblate
```

2. Create a virtualenv:

```bash
virtualenv .venv
.venv/bin/activate
```

3. Install Weblate (for this you need some system dependencies, see Installing from sources):

```bash
pip install -e .
```

3. Install all dependencies useful for development:

```bash
pip install -r requirements-dev.txt
```

4. Start a development server:

```bash
weblate runserver
```

5. Depending on your configuration, you might also want to start Celery workers:

```bash
./weblate/examples/celery start
```

6. To run a test (see Testim vendor for more details):

```bash
./scripts/test-database
./manage.py test
```

Shihni edhe:

Installing from sources
3.2.3 Running Weblate locally in Docker

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
```

Shënim: 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.

Shënim: 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:
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 Kod burim i Weblate-it

Weblate is developed on GitHub. You are welcome to fork the code and open pull requests. Patches in any other form are welcome too.

Shihni edhe:

Check out *Gjërat e brendshme të Weblate-it* to see how Weblate looks from inside.
3.3.1 Security by Design Principles

Any code for Weblate should be written with Security by Design Principles in mind.

3.3.2 Standarde kodimi

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. Weblate 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 Diagnostikim Weblate-i

Bugs can behave as application crashes or as misbehavior. You are welcome to collect info on any such issue and submit it to the issue tracker.

3.4.1 Mënyra diagnostikim

Turning on debug mode will make the exceptions show in the browser. This is useful to debug issues in the web interface, but not suitable for production environment as it has performance consequences and might leak private data.

Shihni edhe:

Disable debug mode

3.4.2 Regjistra Weblate-i

Weblate can produce detailed logs of what is going 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 own logs as well. The example system-wide setups log to several files under /var/log/celery/.

Docker containers log to their output (as usual in the Docker world), so you can look at the logs using docker-compose logs.

Shihni edhe:

Sample configuration contains LOGGING configuration.
3.4.3 Not processing background tasks

Lot of things happen in background Celery workers. In case things like sending out e-mails or component removal does not work, there might be some issue with it.

Things to check in that case:

- Check Celery process is running, see [Background tasks using Celery](#)
- Check Celery queue status either in [Ndërfaqe administrimi](#) or using `celery_queues`
- Look into Celery logs for errors (see [Regjistra Weblate-i](#))

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 using [Ndërfaqe administrimi](#) under the Tools tab.

These send e-mail directly, so this verifies that your SMTP configuration is correct (see [Formësim i email-eve që dërgohen](#)). 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. The easiest way to achieve this is 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 Probleme funksionimi

In case Weblate performs badly in some situation, 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 `doggslow` along with [Collecting error reports](#) and get pinpointed and detailed tracebacks in the error collection tool.

3.5 Gjërat e brendshme të Weblate-it

**Shënim:** 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 Strukturë drejtorish

Quick overview of directory structure of Weblate main repository:

**dokumentime** Source code for this documentation, which can be built using Sphinx.

**dev-docker** Docker code to run development server, see *Running Weblate locally in Docker.*

**weblate** Source code of Weblate as a Django application, see *Gjërat e brendshme të Weblate-it.*

**weblate/static** Client files (CSS, Javascript and images), see *Pjesa e dukshme e Weblate-it.*

3.5.2 Module

Weblate consists of several Django applications (some optional, see *Optional Weblate modules*):

- **llogari**
  
  Llogari, profile dhe njoftime përdoruesi.

- **shtesa**
  
  Add-ons to tweak Weblate behavior, see *Shtesa.*

- **api**
  
  API based on Django REST framework.

- **auth**
  
  Mirëfilltësim dhe Leje.

- **faturim**
  
  The optional *Faturim* module.

- **kontrolle**
  
  Translation string *Kontrolle cilësie* module.

- **shkronja**
  
  Font rendering checks module.

- **formate**
  
  File format abstraction layer based on translate-toolkit.

- **gitexport**
  
  The optional *Git exporter* module.

- **gjuhë**
  
  Module defining language and plural models.

- **ligjore**
  
  The optional *Ligjore* module.

- **makineri**
  
  Integrim shërbumesh përkthimi nga makina.

- **kujtesë**
  
  Built-in translation memory, see *Kujtesë Përkthimesh.*

- **foto ekrani**
  
  Screenshots management and OCR module.

- **trans**
Main module handling translations.

```
utils
Various helper utilities.
vcs
Version control system abstraction.
wladmin
Përshtatje ndërfaqejë përgjegjësi Django.
```

## 3.6 Developing addons

_Shtesa_ are way to customize localization workflow in Weblate.

```python
class weblate.addons.base.BaseAddon (storage=None)

    @classmethod
can_install (component, user)
    Check whether add-on is compatible with given component.

    configure (settings)
    Save configuration.

    daily (component)
    Hook triggered daily.

    @classmethod
get_add_form (user, component, **kwargs)
    Return configuration form for adding new add-on.

    get_settings_form (user, **kwargs)
    Return configuration form for this add-on.

    post_add (translation)
    Hook triggered after new translation is added.

    post_commit (component)
    Hook triggered after changes are committed to the repository.

    post_push (component)
    Hook triggered after repository is pushed upstream.

    post_update (component, previous_head: str, skip_push: bool)
    Hook triggered after repository is updated from upstream.

    Parametra

    - previous_head (str) – HEAD of the repository prior to update, can be blank on initial clone.

    - skip_push (bool) – Whether the add-on operation should skip pushing changes upstream. Usually you can pass this to underlying methods as commit_and_push or commit_pending.

    pre_commit (translation, author)
    Hook triggered before changes are committed to the repository.

    pre_push (component)
    Hook triggered before repository is pushed upstream.

    pre_update (component)
    Hook triggered before repository is updated from upstream.

    save_state ()
    Save add-on state information.
```
The Weblate Manual, Hedhje Në Qarkullim 4.9

```

``` # Copyright © 2012 - 2021 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,)
    # 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 Pjesa e dukshme e Weblate-it

The frontend is currently built using Bootstrap, jQuery and few third party libraries.

3.7.1 Supported browsers

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 Administrim varësish

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 party 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:

```bash
# 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 Stil kodimi

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 Përkthim

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:

```javascript
document.write(gettext('this is to be translated'));

var object_count = 1; // or 0, or 2, or 3, ...
s = ngettext("literal for the singular case", "literal for the plural case", object_count);

fmts = ngettext('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'
```

Shihni edhe:

Translation topic in the Django documentation
3.7.5 Ikona

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 Njoftim problemesh në Weblate

Weblate issue tracker is hosted at GitHub.

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 Probleme sigurie section below.

If you are not sure about your bug report or feature request, you can try Weblate discussions.

3.8.1 Probleme sigurie

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.

Shënim: 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.

Disa nga këto janë:
- 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:

```bash
# 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:

```bash
./scripts/test-database
./ci/run-migrate
./ci/run-test
./ci/run-docs
```

3.9.2 Testim vendor

To run a testsuite locally, use:

```bash
DJANGO_SETTINGS_MODULE=weblate.settings_test ./manage.py test
```

**Ndihmëz:** 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 Ujdisje baze të dhënash për 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:

```bash
# 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
```

(vazhdon nefaqen pasuese)
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

Ndihmëz: The tests can also be executed inside developer docker container, see Running Weblate locally in Docker.

Shihni edhe: See Testing in Django for more info on running and writing tests for Django.

3.10 Skema të dhënash

Weblate uses JSON Schema to define layout of external JSON files.

3.10.1 Weblate Translation Memory Schema

<table>
<thead>
<tr>
<th>Lloj</th>
<th>array</th>
</tr>
</thead>
<tbody>
<tr>
<td>objekte</td>
<td>The Translation Memory Item</td>
</tr>
<tr>
<td>lloj</td>
<td>objekt</td>
</tr>
<tr>
<td>veti</td>
<td></td>
</tr>
<tr>
<td>• kategori</td>
<td>The String Category</td>
</tr>
<tr>
<td>lloj</td>
<td>integer</td>
</tr>
<tr>
<td>shembuj</td>
<td>1</td>
</tr>
<tr>
<td>minimum</td>
<td>0</td>
</tr>
<tr>
<td>parazgjedhje</td>
<td>1</td>
</tr>
<tr>
<td>• origjinë</td>
<td>The String Origin</td>
</tr>
<tr>
<td>lloj</td>
<td>varg</td>
</tr>
<tr>
<td>shembuj</td>
<td>test.tmx</td>
</tr>
<tr>
<td>project/component</td>
<td>parazgjedhje</td>
</tr>
<tr>
<td>• burim</td>
<td>The Source String</td>
</tr>
<tr>
<td>lloj</td>
<td>varg</td>
</tr>
<tr>
<td>shembuj</td>
<td>Tungjatjeta</td>
</tr>
<tr>
<td>minLength</td>
<td>1</td>
</tr>
<tr>
<td>parazgjedhje</td>
<td></td>
</tr>
<tr>
<td>• source_language</td>
<td>The Source Language</td>
</tr>
<tr>
<td>lloj</td>
<td>varg</td>
</tr>
<tr>
<td>shembuj</td>
<td>en</td>
</tr>
</tbody>
</table>

vazhdon në faqen pasuese
Tabela 1 – vazhduar nga faqja e mëparshme

<table>
<thead>
<tr>
<th></th>
<th>rregullsi</th>
<th>^[^ ]+$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>parazgjedhje</td>
<td></td>
</tr>
</tbody>
</table>

**• target**

<table>
<thead>
<tr>
<th>lloj</th>
<th>varg</th>
</tr>
</thead>
<tbody>
<tr>
<td>shembuj</td>
<td>Ahoj</td>
</tr>
<tr>
<td>minLength</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**• target_language**

<table>
<thead>
<tr>
<th>lloj</th>
<th>varg</th>
</tr>
</thead>
<tbody>
<tr>
<td>shembuj</td>
<td>cs</td>
</tr>
<tr>
<td>rregullsi</td>
<td>^[^ ]+$</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

përkufizime

Shihni edhe:

Kujtesë Përkthimesh, dump_memory, import_memory

3.10.2 Eksportim të dhënash përdoruesi në Weblate

https://weblate.org/schemas/weblate-userdata.schema.json

<table>
<thead>
<tr>
<th>lloj</th>
<th>objekt</th>
</tr>
</thead>
<tbody>
<tr>
<td>veti</td>
<td></td>
</tr>
<tr>
<td><strong>• basic</strong></td>
<td></td>
</tr>
<tr>
<td>lloj</td>
<td>objekt</td>
</tr>
<tr>
<td>veti</td>
<td></td>
</tr>
<tr>
<td><strong>• emër përdoruesi</strong></td>
<td></td>
</tr>
<tr>
<td>lloj</td>
<td>varg</td>
</tr>
<tr>
<td>shembuj</td>
<td>përgjegjës</td>
</tr>
<tr>
<td>parazgjedhje</td>
<td></td>
</tr>
<tr>
<td><strong>• full_name</strong></td>
<td></td>
</tr>
<tr>
<td>lloj</td>
<td>varg</td>
</tr>
<tr>
<td>shembuj</td>
<td>Përgjegjës Weblate</td>
</tr>
<tr>
<td>parazgjedhje</td>
<td></td>
</tr>
<tr>
<td><strong>• email</strong></td>
<td></td>
</tr>
<tr>
<td>lloj</td>
<td>varg</td>
</tr>
<tr>
<td>shembuj</td>
<td><a href="mailto:noreply@example.com">noreply@example.com</a></td>
</tr>
<tr>
<td>parazgjedhje</td>
<td></td>
</tr>
<tr>
<td><strong>• date Joined</strong></td>
<td></td>
</tr>
<tr>
<td>lloj</td>
<td>varg</td>
</tr>
<tr>
<td>shembuj</td>
<td>2019-11-18T18:53:54.862Z</td>
</tr>
<tr>
<td>parazgjedhje</td>
<td></td>
</tr>
<tr>
<td><strong>• profil</strong></td>
<td></td>
</tr>
<tr>
<td>lloj</td>
<td>objekt</td>
</tr>
<tr>
<td>veti</td>
<td></td>
</tr>
<tr>
<td><strong>• gjuhë</strong></td>
<td></td>
</tr>
<tr>
<td>lloj</td>
<td>varg</td>
</tr>
<tr>
<td>shembuj</td>
<td>cs</td>
</tr>
<tr>
<td>rregullsi</td>
<td>^[^ ]+$</td>
</tr>
<tr>
<td>parazgjedhje</td>
<td></td>
</tr>
<tr>
<td><strong>• suggested</strong></td>
<td></td>
</tr>
<tr>
<td>lloj</td>
<td>integer</td>
</tr>
<tr>
<td>shembuj</td>
<td>1</td>
</tr>
</tbody>
</table>

vazhdon në faqen pasuese

3.10. Skema të dhënash 397
Tabela 2 – vazhduar nga faqja e mëparshme

<table>
<thead>
<tr>
<th>Parazgjedhje</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>translated</strong></td>
<td>Numer vargjesh të përkthyer</td>
</tr>
<tr>
<td>Lloj</td>
<td>integer</td>
</tr>
<tr>
<td>Shembuj</td>
<td>24</td>
</tr>
<tr>
<td>Parazgjedhje</td>
<td>0</td>
</tr>
<tr>
<td><strong>uploaded</strong></td>
<td>Numer fotosh ekrani të ngarkuara</td>
</tr>
<tr>
<td>Lloj</td>
<td>integer</td>
</tr>
<tr>
<td>Shembuj</td>
<td>1</td>
</tr>
<tr>
<td>Parazgjedhje</td>
<td>0</td>
</tr>
<tr>
<td><strong>hide_completed</strong></td>
<td>Hide completed translations on the dashboard</td>
</tr>
<tr>
<td>Lloj</td>
<td>boolean</td>
</tr>
<tr>
<td>Shembuj</td>
<td>False</td>
</tr>
<tr>
<td>Parazgjedhje</td>
<td>True</td>
</tr>
<tr>
<td><strong>secondary_in_zen</strong></td>
<td>Show secondary translations in the Zen mode</td>
</tr>
<tr>
<td>Lloj</td>
<td>boolean</td>
</tr>
<tr>
<td>Shembuj</td>
<td>True</td>
</tr>
<tr>
<td>Parazgjedhje</td>
<td>True</td>
</tr>
<tr>
<td><strong>hide_source_secondary</strong></td>
<td>Fshihe burimin nëse ekziston një përkthim dytësor</td>
</tr>
<tr>
<td>Lloj</td>
<td>boolean</td>
</tr>
<tr>
<td>Shembuj</td>
<td>False</td>
</tr>
<tr>
<td>Parazgjedhje</td>
<td>True</td>
</tr>
<tr>
<td><strong>editor_link</strong></td>
<td>Lidhje përpunuesi</td>
</tr>
<tr>
<td>Lloj</td>
<td>varg</td>
</tr>
<tr>
<td>Shembuj</td>
<td>^.*$</td>
</tr>
<tr>
<td>Parazgjedhje</td>
<td></td>
</tr>
<tr>
<td><strong>translate_mode</strong></td>
<td>Mënyrë përpunuesi përkthimesh</td>
</tr>
<tr>
<td>Lloj</td>
<td>integer</td>
</tr>
<tr>
<td>Shembuj</td>
<td>0</td>
</tr>
<tr>
<td>Parazgjedhje</td>
<td>0</td>
</tr>
<tr>
<td><strong>zen_mode</strong></td>
<td>Mënyra përpunuesi Zen</td>
</tr>
<tr>
<td>Lloj</td>
<td>integer</td>
</tr>
<tr>
<td>Shembuj</td>
<td>0</td>
</tr>
<tr>
<td>Parazgjedhje</td>
<td>0</td>
</tr>
<tr>
<td><strong>special_chars</strong></td>
<td>Shenja speciale</td>
</tr>
<tr>
<td>Lloj</td>
<td>varg</td>
</tr>
<tr>
<td>Shembuj</td>
<td>^.*$</td>
</tr>
<tr>
<td>Parazgjedhje</td>
<td></td>
</tr>
<tr>
<td><strong>dashboard_view</strong></td>
<td>Panje parazgjedhje e puliti</td>
</tr>
<tr>
<td>Lloj</td>
<td>integer</td>
</tr>
<tr>
<td>Shembuj</td>
<td>1</td>
</tr>
<tr>
<td>Parazgjedhje</td>
<td>0</td>
</tr>
<tr>
<td><strong>dashboard_component_list</strong></td>
<td>Listë përbërësish parazgjedhje</td>
</tr>
<tr>
<td>Lloj</td>
<td>integer</td>
</tr>
<tr>
<td>Shembuj</td>
<td>1</td>
</tr>
<tr>
<td>Parazgjedhje</td>
<td>0</td>
</tr>
<tr>
<td><strong>gjuhë</strong></td>
<td>Gjuhë të përkthyesa</td>
</tr>
<tr>
<td>Lloj</td>
<td>array</td>
</tr>
<tr>
<td>Shembuj</td>
<td></td>
</tr>
<tr>
<td>Parazgjedhje</td>
<td></td>
</tr>
<tr>
<td><strong>secondary_languages</strong></td>
<td>Gjuhë dytësore</td>
</tr>
</tbody>
</table>

Vazhdon në faqen pasuese
<table>
<thead>
<tr>
<th>lloj</th>
<th>array</th>
</tr>
</thead>
<tbody>
<tr>
<td>parazgjedhje</td>
<td>Kod gjyhe</td>
</tr>
<tr>
<td>objekte</td>
<td>lloj</td>
</tr>
<tr>
<td></td>
<td>varg</td>
</tr>
<tr>
<td></td>
<td>shembuj</td>
</tr>
<tr>
<td></td>
<td>rregullsi</td>
</tr>
<tr>
<td></td>
<td>^.*$</td>
</tr>
<tr>
<td></td>
<td>parazgjedhje</td>
</tr>
</tbody>
</table>

**watched**

<table>
<thead>
<tr>
<th>lloj</th>
<th>array</th>
</tr>
</thead>
<tbody>
<tr>
<td>parazgjedhje</td>
<td>Projekte të mbikëqyrur</td>
</tr>
<tr>
<td>objekte</td>
<td>Identifikues projektit</td>
</tr>
<tr>
<td></td>
<td>lloj</td>
</tr>
<tr>
<td></td>
<td>varg</td>
</tr>
<tr>
<td></td>
<td>shembuj</td>
</tr>
<tr>
<td></td>
<td>rregullsi</td>
</tr>
<tr>
<td></td>
<td>^.*$</td>
</tr>
<tr>
<td></td>
<td>parazgjedhje</td>
</tr>
</tbody>
</table>

**auditlog**

<table>
<thead>
<tr>
<th>lloj</th>
<th>array</th>
</tr>
</thead>
<tbody>
<tr>
<td>parazgjedhje</td>
<td>Audit log</td>
</tr>
<tr>
<td>objekte</td>
<td>Items</td>
</tr>
<tr>
<td></td>
<td>lloj</td>
</tr>
<tr>
<td></td>
<td>objekt</td>
</tr>
<tr>
<td>veti</td>
<td>IP address</td>
</tr>
<tr>
<td></td>
<td>lloj</td>
</tr>
<tr>
<td></td>
<td>varg</td>
</tr>
<tr>
<td></td>
<td>shembuj</td>
</tr>
<tr>
<td></td>
<td>rregullsi</td>
</tr>
<tr>
<td></td>
<td>^.*$</td>
</tr>
<tr>
<td></td>
<td>parazgjedhje</td>
</tr>
</tbody>
</table>

**adressë**

<table>
<thead>
<tr>
<th>lloj</th>
<th>varg</th>
</tr>
</thead>
<tbody>
<tr>
<td>parazgjedhje</td>
<td>IP address</td>
</tr>
<tr>
<td></td>
<td>lloj</td>
</tr>
<tr>
<td></td>
<td>varg</td>
</tr>
<tr>
<td></td>
<td>shembuj</td>
</tr>
<tr>
<td></td>
<td>rregullsi</td>
</tr>
<tr>
<td></td>
<td>^.*$</td>
</tr>
<tr>
<td></td>
<td>parazgjedhje</td>
</tr>
</tbody>
</table>

**user_agent**

<table>
<thead>
<tr>
<th>lloj</th>
<th>varg</th>
</tr>
</thead>
<tbody>
<tr>
<td>parazgjedhje</td>
<td>Agjent përdoruesi</td>
</tr>
<tr>
<td></td>
<td>lloj</td>
</tr>
<tr>
<td></td>
<td>varg</td>
</tr>
<tr>
<td></td>
<td>shembuj</td>
</tr>
<tr>
<td></td>
<td>rregullsi</td>
</tr>
<tr>
<td></td>
<td>^.*$</td>
</tr>
<tr>
<td></td>
<td>parazgjedhje</td>
</tr>
</tbody>
</table>

**vulë kohore**

<table>
<thead>
<tr>
<th>lloj</th>
<th>varg</th>
</tr>
</thead>
<tbody>
<tr>
<td>parazgjedhje</td>
<td>Vulë kohore</td>
</tr>
<tr>
<td></td>
<td>lloj</td>
</tr>
<tr>
<td></td>
<td>varg</td>
</tr>
<tr>
<td></td>
<td>shembuj</td>
</tr>
<tr>
<td></td>
<td>rregullsi</td>
</tr>
<tr>
<td></td>
<td>^.*$</td>
</tr>
<tr>
<td></td>
<td>parazgjedhje</td>
</tr>
</tbody>
</table>

**veprimtari**

<table>
<thead>
<tr>
<th>lloj</th>
<th>varg</th>
</tr>
</thead>
<tbody>
<tr>
<td>parazgjedhje</td>
<td>Veprimtari</td>
</tr>
<tr>
<td></td>
<td>lloj</td>
</tr>
<tr>
<td></td>
<td>varg</td>
</tr>
<tr>
<td></td>
<td>shembuj</td>
</tr>
<tr>
<td></td>
<td>rregullsi</td>
</tr>
<tr>
<td></td>
<td>^.*$</td>
</tr>
<tr>
<td></td>
<td>parazgjedhje</td>
</tr>
</tbody>
</table>

përkuftime

Shihni edhe:

**Profil përdoruesi, 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.

Shihni edhe:

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 -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 Security and privacy

Development of Weblate adheres to the Best Practices of the Linux Foundation’s Core Infrastructure Initiative.

**Probleme sigurie**

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.

**Ndihmëz:** 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 scanned using Anchore and Trivy. 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).

**Shihni edhe:**

Continuous integration

3.13 Rreth Weblate-it

3.13.1 Objektiva projekti

Web-based continuous localization tool with tight Integrim kontrolli versioni supporting a wide range of Formate të mbuluar kartelash, making it easy for translators to contribute.

3.13.2 Emër projekti

«Weblate» is a portmanteau of the words «web» and «translate».
3.13.3 Sajt projekti

The landing page is https://weblate.org and there is a cloud-hosted service at https://hosted.weblate.org. The documentation can be read at https://docs.weblate.org.

3.13.4 Stema projekti

The project logos and other graphics are available in https://github.com/WeblateOrg/graphics.

3.13.5 Drejtues

This project is maintained by Michal Čihař, who can be reached at michal@cihar.com.

3.13.6 Autorë

Weblate was started by Michal Čihař. Since its inception in 2012, thousands of people have contributed.

3.14 Licencë

Copyright (C) 2012 - 2021 Michal Čihař <michal@cihar.com>

Ky program është software i lirë; mund ta rishpërndani dhe/ose ndryshoni sipas kushteve të licencës GNU General Public License, siç botohet nga Free Software Foundation; ose versioni 3 i Licencës, ose (në dëshirën tuaj) çfarëdo versioni të mëvonshëm.

Ky program shpërndahet me shpresën se do të jetë i dobishëm, por PA ASNJË GARANCI; madje pa as edhe garancinë e nënkuptuar të TREGTUESHMËRISË ose PËRSHTATSHMËRISË PËR NJË QËLLIM TË VEÇANTË. Për më tepër hollësi, shihni licencën GNU General Public License.

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 4.9

Not yet released.

4.2 Weblate 4.8.1

Released on September 10th 2021.

- Fixed user removal in Django admin interface.
- Document add-on parameters in more detail.
- Fixed JavaScript error on glossary.
- Add limit to number of matches in consistency check.
- Improve placeholders handling in machine translations.
- Fixed creating add-ons using API.
- Added PRIVACY_URL setting to add privacy policy link to the footer.
- Hide member e-mail addresses from project admins.
- Improved gettext PO merging in case of conflicts.
- Improved glossary highlighting.
- Improved safe-html flag behavior with XML checks.
- Fixed commit messages on linked components.

All changes in detail.
4.3 Weblate 4.8

Released on August 21th 2021.

- Added support for Apple stringsdict format.
- The exact search operator is now case-sensitive on PostgreSQL.
- Fixed saving glossary explanations in some cases.
- Documentation improvements.
- Përmirësime funksionimi.
- Improved squash add-on compatibility with Gerrit.
- Fixed adding strings to monolingual glossary components.
- Improved performance in handling variants.
- Fixed squash add-on sometimes skipping parsing upstream changes.
- Preserve file extension for downloads.
- Added support for the Fluent format.
- Added support for using tabs to indent JSON formats.

All changes in detail.

4.4 Weblate 4.7.2

Released on July 15th 2021.

- Support more language aliases to be configured on a project.
- Fixed search string validation in API.
- Fixed Git exporter URLs after a domain change.
- Fixed cleanup add-on for Windows RC files.
- Fixed possible crash in XLIFF updating.

All changes in detail.

4.5 Weblate 4.7.1

Released on June 30th 2021.

- Improved popup for adding terms to glossary.
- Added support for LibreTranslate machine translation service.
- Added rate limiting on creating new projects.
- Improved performance of file updates.

All changes in detail.
4.6 Weblate 4.7

Released on June 17th 2021.

- Improved configuration health check.
- Added support for `object-pascal-format` used in gettext PO, see Object Pascal format.
- Renamed Nearby keys to Similar keys to better describe the purpose.
- Added support for `mi18n lang files`.
- Improved SAML authentication integration.
- Fixed Gerrit integration to better handle corner cases.
- Weblate now requires Django 3.2.
- Fixed inviting users when e-mail authentication is disabled.
- Improved language definitions.
- Added support for blocking users from contributing to a project.
- Fixed automatic creation of glossary languages.
- Extended documentation about add-ons.
- Performance improvements for components with linked repositories.
- Added support for free DeepL API.
- The user management no longer needs Django admin interface.

All changes in detail.

4.7 Weblate 4.6.2

Released on May 8th 2021.

- Fixed crash after moving shared component between projects.
- Fixed adding new strings to empty properties files.
- Fixed copy icon alignment in RTL languages.
- Extended string statistics on the Info tab.
- Fixed handling of translation files ignored in Git.
- Improved metrics performance.
- Fixed possible bug in saving glossaries.
- Fixed consistency check behavior on languages with different plural rules.

All changes in detail.
4.8 Weblate 4.6.1

Released on May 2nd 2021.

• Remove obsolete spam protection code.
• Improve source plural check accuracy.
• Update list of user interface languages in Docker.
• Improved error messages when creating pull requests.
• Fixed creating pull requests on Pagure.
• Fixed triggering automatically installed add-ons.
• Fixed possible caching issues on upgrade.
• Fixed adding new units to monolingual translations using upload.

All changes in detail.

4.9 Weblate 4.6

Released on April 19th 2021.

• The auto_translate management command has now a parameter for specifying translation mode.
• Added support for Kartela tekst.
• Added trends and metrics for all objects.
• Added support for direct copying text from secondary languages.
• Added date filtering when browsing changes.
• Improved activity charts.
• Sender for contact form e-mails can now be configured.
• Improved parameters validation in component creation API.
• The rate limiting no longer applies to superusers.
• Improved automatic translation add-on performance and reliability.
• The rate limiting now can be customized in the Docker container.
• API for creating components now automatically uses URL të brendshme Weblate.
• Simplified state indication while listing strings.
• Password hashing now uses Argon2 by default.
• Simplified progress bars indicating translation status.
• Renamed Add missing languages to clarify the purpose.
• Fixed saving string state to XLIFF.
• Added language-wide search.
• Initial support for Scaling horizontally the Docker deployment.

All changes in detail.
4.10 Weblate 4.5.3

Released on April 1st 2021.

• Fixed metrics collection.
• Fixed possible crash when adding strings.
• Improved search query examples.
• Fixed possible loss of newly added strings on replace upload.

4.11 Weblate 4.5.2

Released on March 26th 2021.

• Configurable schedule for automatic translation.
• Added Lua format check.
• Ignore format strings in the Fjalë të njëpasnjëshme të përsëdytura check.
• Allow uploading screenshot from a translate page.
• Added forced file synchronization to the repository maintenance.
• Fixed automatic suggestions for languages with a longer code.
• Improved performance when adding new strings.
• Several bug fixes in quality checks.
• Several performance improvements.
• Added integration with Zbuloni Weblate-in.
• Fixed checks behavior with read-only strings.

All changes in detail.

4.12 Weblate 4.5.1

Released on March 5th 2021.

• Fixed editing of glossary flags in some corner cases.
• Extend metrics usage to improve performance of several pages.
• Store correct source language in TMX files.
• Better handling for uploads of monolingual PO using API.
• Improved alerts behavior on glossary components.
• Improved Markdown link checks.
• Indicate glossary and source language in breadcrumbs.
• Paginated component listing of huge projects.
• Improved performance of translation, component or project removal.
• Improved bulk edit performance.
• Fixed preserving «Needs editing» and «Approved» states for ODF files.
• Improved interface for customizing translation-file downloads.
All changes in detail.

## 4.13 Weblate 4.5

Released on February 19th 2021.

- Added support for **lua-format** used in gettext PO.
- Added support for sharing a component between projects.
- Fixed multiple unnamed variables check behavior with multiple format flags.
- Dropped mailing list field on the project in favor of generic instructions for translators.
- Added pseudolocale generation add-on.
- Added support for TermBase eXchange files.
- Added support for manually defining string variants using a flag.
- Improved performance of consistency checks.
- Improved performance of translation memory for long strings.
- Added support for searching in explanations.
- Strings can now be added and removed in bilingual formats as well.
- Extend list of supported languages in Amazon Translate machine translation.
- Automatically enable Java MessageFormat checks for Java Properties.
- Added a new upload method to add new strings to a translation.
- Added a simple interface to browse translation.
- Glossaries are now stored as regular components.
- Dropped specific API for glossaries as component API is used now.
- Added simplified interface to toggle some of the flags.
- Added support for non-translatable or forbidden terms in the glossary.
- Added support for defining terminology in a glossary.
- Moved text direction toggle to get more space for the visual keyboard.
- Added option to automatically watch projects user-contributed to.
- Added check whether translation matches the glossary.
- Added support for customizing navigation text color.

All changes in detail.

## 4.14 Weblate 4.4.2

Released on January 14th 2021.

- Fixed corruption of one distributed MO file.
4.15 Weblate 4.4.1

Released on January 13th 2021.

• Fixed reverting plural changes.
• Fixed displaying help for project settings.
• Improved administration of users.
• Improved handling of context in monolingual PO files.
• Fixed cleanup add-on behavior with HTML, ODF, IDML and Windows RC formats.
• Fixed parsing of location from CSV files.
• Use content compression for file downloads.
• Improved user experience on importing from ZIP file.
• Improved detection of file format for uploads.
• Avoid duplicate pull requests on Pagure.
• Improved performance when displaying ghost translations.
• Reimplemented translation editor to use native browser text area.
• Fixed cleanup add-on breaking adding new strings.
• Added API for add-ons.

All changes in detail.

4.16 Weblate 4.4

Released on December 15th 2020.

• Improved validation when creating a component.
• Weblate now requires Django 3.1.
• Added support for appearance customization in the management interface.
• Fixed read-only state handling in bulk edit.
• Improved CodeMirror integration.
• Added add-on to remove blank strings from translation files.
• The CodeMirror editor is now used for translations.
• Syntax highlighting in translation editor for XML, HTML, Markdown and reStructuredText.
• Highlight placeables in translation editor.
• Improved support for non-standard language codes.
• Added alert when using ambiguous language codes.
• The user is now presented with a filtered list of languages when adding a new translation.
• Extended search capabilities for changes in history.
• Improved billing detail pages and libre hosting workflow.
• Extended translation statistics API.
• Improved «other translations» tab while translating.
• Added tasks API.
• Improved performance of file upload.
• Improved display of user defined special characters.
• Improved performance of auto-translation.
• Several minor improvements in the user interface.
• Improved naming of ZIP downloads.
• Added option for getting notifications on unwatched projects.

All changes in detail.

4.17 Weblate 4.3.2

Released on November 4th 2020.

• Fixed crash on certain component filemasks.
• Improved accuracy of the consecutive duplicated words check.
• Added support for Pagure pull requests.
• Improved error messages for failed registrations.
• Reverted rendering developer comments as Markdown.
• Simplified setup of Git repositories with different default branch than «master».
• Newly created internal repositories now use main as the default branch.
• Reduced false positives rate of unchanged translation while translating reStructuredText.
• Fixed CodeMirror display issues in some situations.
• Renamed Template group to «Sources» to clarify its meaning.
• Fixed GitLab pull requests on repositories with longer paths.

All changes in detail.

4.18 Weblate 4.3.1

Hedhunë qarkullim m 21 tetor, 2020.

• Improved auto-translation performance.
• Fixed session expiry for authenticated users.
• Shtim mbulimi për fshihje të dhënash versioni.
• Improve hooks compatibility with Bitbucket Server.
• Improved performance of translation memory updates.
• Reduced memory usage.
• Improved performance of Matrix view.
• Added confirmation before removing a user from a project.

All changes in detail.
4.19 Weblate 4.3

Released on October 15th 2020.

- Include user stats in the API.
- Fixed component ordering on paginated pages.
- Define source language for a glossary.
- Rewritten support for GitHub and GitLab pull requests.
- Fixed stats counts after removing suggestion.
- Extended public user profile.
- Fixed configuration of enforced checks.
- Improve documentation about built-in backups.
- Moved source language attribute from project to a component.
- Add Vue i18n formatting check.
- Generic placeholders check now supports regular expressions.
- Improved look of Matrix mode.
- Machinery is now called automatic suggestions.
- Added support for interacting with multiple GitLab or GitHub instances.
- Extended API to cover project updates, unit updates and removals and glossaries.
- Unit API now properly handles plural strings.
- Component creation can now handle ZIP file or document upload.
- Consolidated API response status codes.
- Support Markdown in contributor agreement.
- Improved source strings tracking.
- Improved JSON, YAML and CSV formats compatibility.
- Added support for removing strings.
- Improved performance of file downloads.
- Improved repository management view.
- Automatically enable java-format for Android.
- Added support for localized screenshots.
- Added support for Python 3.9.
- Fixed translating HTML files under certain conditions.

All changes in detail.
4.20 Weblate 4.2.2

Released on September 2nd 2020.

- Fixed matching of source strings for JSON formats.
- Fixed login redirect for some authentication configurations.
- Fixed LDAP authentication with group sync.
- Fixed crash in reporting automatic translation progress.
- Fixed Git commit squashing with trailers enabled.
- Fixed creating local VCS components using API.

4.21 Weblate 4.2.1

Released on August 21st 2020.

- Fixed saving plurals for some locales in Android resources.
- Fixed crash in the cleanup add-on for some XLIFF files.
- Allow setting up localization CDN in Docker image.

4.22 Weblate 4.2

Released on August 18th 2020.

- Improved user pages and added listing of users.
- Dropped support for migrating from 3.x releases, migrate through 4.1 or 4.0.
- Added exports into several monolingual formats.
- Improved activity charts.
- Number of displayed nearby strings can be configured.
- Added support for locking components experiencing repository errors.
- Simplified main navigation (replaced buttons with icons).
- Improved language code handling in Google Translate integration.
- The Git squash add-on can generate Co-authored-by: trailers.
- Improved query search parser.
- Improved user feedback from format strings checks.
- Improved performance of bulk state changes.
- Added compatibility redirects after project or component renaming.
- Added notifications for strings approval, component locking and license change.
- Added support for ModernMT.
- Allow to avoid overwriting approved translations on file upload.
- Dropped support for some compatibility URL redirects.
- Added check for ECMAScript template literals.
- Added option to watch a component.
• Removed leading dot from JSON unit keys.
• Removed separate Celery queue for translation memory.
• Allow translating all components a language at once.
• Allow to configure Content-Security-Policy HTTP headers.
• Added support for aliasing languages at project level.
• New add-on to help with HTML or JavaScript localization, see CDN përkthimesh JavaScript.
• The Weblate domain is now configured in the settings, see SITE_DOMAIN.
• Add support for searching by component and project.

4.23 Weblate 4.1.1

Released on June 19th 2020.
• Fixed changing autofix or add-ons configuration in Docker.
• Fixed possible crash in «About» page.
• Improved installation of byte-compiled locale files.
• Fixed adding words to glossary.
• Fixed keyboard shortcuts for machinery.
• Removed debugging output causing discarding log events in some setups.
• Fixed lock indication on project listing.
• Fixed listing GPG keys in some setups.
• Added option for which DeepL API version to use.
• Added support for acting as SAML Service Provider, see Mirëfillësim SAML.

4.24 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: Vargje PHP Lavarel, Kartela HTML, Format OpenDocument, Format IDML, Kartela Windows RC, Përkhime INI, Inno Setup INI translations, Veti GWT, go-i18n JSON files, Kartelë ARB.
• Consistently use dismissed as state of dismissed checks.
• Add support for configuring default add-ons to enable.
• 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: Ndryshore të shumta të paemërtuara, I papërkthyer prej kohësh, Fjalë të njëpasnjëshme të përsëdytura.
• 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.

4.25 Weblate 4.0.4

Released on May 7th 2020.

• 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.
### 4.26 Weblate 4.0.3

Released on May 2nd 2020.

- 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.

### 4.27 Weblate 4.0.2

Released on April 27th 2020.

- Improved performance of translation stats.
- Improved performance of changing labels.
- Improved bulk edit performance.
- 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.

### 4.28 Weblate 4.0.1

Released on April 16th 2020.

- Fixed package installation from PyPI.
4.29 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 i18next interpolation and nesting.
- Added check and highlighter for percent placeholders.
- Display suggestions failing checks.
- Record source string changes in history.
- Upgraded Microsoft Translator to version 3 API.
- Reimplemented translation memory backend.
- Added support for several \texttt{i18n} lookups in \texttt{Kërkim}.
- Allow to make \texttt{Përkthim i pandryshuar} 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 \texttt{Kërkim}.
- Extended API to cover screenshots, users, groups, component lists 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.

4.30 Weblate 3.x series

4.30.1 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.
• Fixed documentation links for add-ons.
• 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.

4.30.2 Weblate 3.11.2

Released on February 22nd 2020.

• Fixed rendering of suggestions.
• Fixed some strings wrongly reported as having no words.

4.30.3 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.

4.30.4 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.
• Added support for grouping string shapings.
• 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.

4.30. Weblate 3.x series
• Automatically unshallow local copies if needed.
• Fixed download of strings needing action.
• New alert to warn about using the same filemask 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.
• Added bulk edit add-on.
• Added option for *Enforcing checks*.
• Increased default validity of confirmation links.
• Improved Matomo integration.
• Fixed `Është përkthyer` to correctly handle source string change.
• Extended automatic updates configuration by `AUTO_UPDATE`.
• LINGUAS add-ons now do full sync of translations in Weblate.

### 4.30.5 Weblate 3.10.3

Released on January 18th 2020.

• Support for translate-toolkit 2.5.0.

### 4.30.6 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.
4.30.7 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.
- Improved data clean-up performance.
- 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.

4.30.8 Weblate 3.10

Released on December 20th 2019.

- Improved application user interface.
- Added double space check.
- Fixed creating new languages.
- Avoid sending auditlog notifications to deleted e-mails.
- Added support for read only strings.
- 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.
4.30.9 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 fix up now honor source string markup.
- Avoid sending notifications to invited and deleted users.
- Fix SSL connection to redis in Celery in Docker container.

4.30.10 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.
- Added automatic translation add-on to bootstrap translations.
- 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.

4.30.11 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.
• Improve performance of repository scoped add-ons.
• 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.
• Add support for cloning add-ons to discovered components.
• 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.

4.30.12 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.

4.30.13 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.
• Add support for configuring msgmerge add-on.
• Delay opening SMTP connection when sending notifications.
• Improved error logging.
• Allow custom location in MO generating add-on.
• 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.

4.30.14 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.

4.30.15 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.

4.30. Weblate 3.x series 423
4.30.16  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.

4.30.17  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.
- Add-on commit message now can be configured.
- 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.
4.30.18 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.
- New add-on to squash commits prior to push.
- 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.

4.30.19 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 resname as context when available.
- Added support for XLIFF states.
- Added check for non writable files in DATA_DIR.
- Improved CSV export for changes.

4.30.20 Weblate 3.2.2

Released on October 20th 2018.

- Remove no longer needed Babel dependency.
- Updated language definitions.
- Improve documentation for add-ons, LDAP and Celery.
- Fixed enabling new dos-coll 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.
4.30.21 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 giteexport 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.

4.30.22 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.
- Built-in translation memory now automatically stores translations done.
- 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.
4.30.23 Weblate 3.1.1

Released on July 27th 2018.

- Fix testsuite failure on some setups.

4.30.24 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 libravatar service.
- New add-on to mark unchanged translations as needing edit.
- 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.

4.30.25 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.

4.30.26 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.
• New add-on to ensure all components in a project have same translations.
• Support for more variables in commit message templates.
• Add support for providing additional textual context.

4.31 Weblate 2.x series

4.31.1 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.
• Added built-in translation memory.
• Added component lists overview to dashboard and per component list overview pages.
• Added support for DeepL machine translation service.
• Machine translation results are now cached inside Weblate.
• Added support for reordering committed changes.

4.31.2 Weblate 2.19.1

Released on February 20th 2018.

• Fixed migration issue on upgrade from 2.18.
• Improved file upload API validation.

4.31.3 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.
• Improved built-in machine translation service.
• 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.
• Added add-ons to customize translation workflows.

4.31.4 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.

4.31.5 Weblate 2.17.1

Released on October 13th 2017.
• Fixed running testsuite in some specific situations.
• Locales updates.
4.31.6 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.

4.31.7 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.

4.31.8 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.

4.31.9 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.

4.31.10 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.
4.31.11  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.

4.31.12  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 ALCs are now implemented using Group ACL.
• Added more fine grained privileges control.
• Various minor UI improvements.

4.31.13  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.
4.31.14 **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.

4.31.15 **Weblate 2.10.1**

Released on Jan 20th 2017.

- Do not leak account existence on password reset form (CVE-2017-5537).

4.31.16 **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.
4.31.17 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.

4.31.18 Weblate 2.8

Hedhur në qarkullim më 31 gusht, 2016.

- Documentation improvements.
- 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.
4.31.19 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.

4.31.20 Weblate 2.6

Hedhur në qarkullim më 28 prill, 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.

4.31.21 Weblate 2.5

Hedhur në qarkullim më 10 mars, 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.
- Documentation improvements.
- 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.
- Fixed site wide search for failing checks.
• 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 not translated strings.
• Mbulim për 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.
• Documentation improvements.
• Added support for configurable automatic group assignments.
• Improved adding of new translations.

4.31.22 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.

4.31.23 Weblate 2.3

Hedhur në qarkullim më 22 maj, 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.
4.31.24 Webate 2.2

Hedhur në qarkullim më 19 shkurt, 2015.

- Përmirësime funksionimi.
- Fulltext search on location and comments fields.
- New SVG/javascript based activity charts.
- Mbulim për Django 1.8.
- Mbulim për fshirje komentesh.
- Added own SVG badge.
- U shtua mbulim për Google Analytics.
- U përirmësua trajtimi i emrave të kartelave të përkthimit.
- Added support for monolingual JSON translations.
- Record component locking in a history.
- Support for editing source (template) language for monolingual translations.
- U shtua mbulim i thjeshtë për Gerrit.

4.31.25 Webate 2.1

Released on Dec 5th 2014.

- U shtua mbulim për depo Mercurial.
- U zëvendësuan shkronjat Glyphicon me ato Awesome.
- U shtuan ikona për shërbime mirëfillësimi përmes rrjetesh shoqërore.
- Better consistency of button colors and icons.
- Documentation improvements.
- Ndreqje të metash të ndryshme.
- Automatic hiding of columns in translation listing for small screens.
- U ndryshua formësimi për shtigje sistemi kartelash.
- Improved SSH keys handling and storage.
- U përmirësua kyçje deposh.
- Customizable quality checks per source string.
- Allow to hide completed translations from dashboard.

4.31.26 Webate 2.0

Released on Nov 6th 2014.

- New responsive UI using Bootstrap.
- Rewritten VCS backend.
- Documentation improvements.
- Added whiteboard for site wide messages.
- Configurable strings priority.
- U shtua mbulim për format JSON kartelash.
• Fixed generating mo files in certain cases.
• U shtua mbulim për njoftime GitLab.
• U shtua mbulim për çaktivizim sugjerimesh përkthimi.
• Mbulim për Django 1.7.
• ACL projects now have user management.
• U zgjeruan mundësitë e kërkimit.
• Give more hints to translators about plurals.
• U ndreq kycje deposh Git.
• Përputhshmëri me versione të vjetër Git-i.
• U përmirësua mbulimi për ACL.
• Added buttons for per language quotes and other special characters.
• Mbulim për eksportim statistikash si JSONP.

4.32 Seritë Weblate 1.x

4.32.1 Weblate 1.9

Hedhur në qarkullim më 6 maj, 2014.
• Përputhshmëri me Django 1.6.
• No longer maintained compatibility with Django 1.4.
• Management commands for locking/unlocking translations.
• Improved support for Qt TS files.
• Përdoruesit tani mund të fshijnë llogarinë e tyre.
• Avatarët mund të çaktivizohen.
• Merged first and last name attributes.
• Avatars are now fetched and cached server side.
• U shtua mbulim për stemë shields.io.

4.32.2 Weblate 1.8

Hedhur në qarkullim më 7 nëntor, 2013.
• Ju lutemi, shihni doracakun për udhëzime përmirësimi.
• 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.
• Documentation improvements.
• U përmirësua shqyrtim vargjesh burim.
• Kërkim nëpër krejt vargjet.
4.32.3 Weblate 1.7

Hedhur në qarkullim më 7 tetor, 2013.

- Ndëkkje më e mirë e vargjeve burim.
- Mbrojtje CAPTCHA për regjistrimet.
- MbrojtjeCAPTCHA për regjistrimet.

- MbrojtjeCAPTCHA për regjistrimet.

- 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.
- MbrojtjeCAPTCHA për regjistrimet.
- MbrojtjeCAPTCHA për regjistrimet.
- MbrojtjeCAPTCHA për regjistrimet.

- MbrojtjeCAPTCHA për regjistrimet.
- MbrojtjeCAPTCHA për regjistrimet.
- MbrojtjeCAPTCHA për regjistrimet.
- MbrojtjeCAPTCHA për regjistrimet.

- MbrojtjeCAPTCHA për regjistrimet.
- MbrojtjeCAPTCHA për regjistrimet.
4.32.5 Weblate 1.5

Hedhur në qarkullim më 6 prill, 2013.

• Ju lutemi, shihni doracakun për udhëzime përmirësimi.
• U shtuan faqe publike përdoruesi.
• Emërtim më i mirë i formave të shumësit.
• Added support for TBX export of glossary.
• U shtua mbulim për njoftime Bitbucket.
• Activity charts are now available for each translation, language or user.
• Extended options of import_project admin command.
• I përputhshëm me Django 1.5.
• Avatars are now shown using libravatar.
• 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.
• Ndërfaqe e re për përkthime nga makina.
• Added support for monolingual po files.
• Extend amount of cached metadata to improve speed of various searches.
• Now shows word counts as well.

4.32.6 Weblate 1.4

Hedhur në qarkullim më 23 janar, 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 tktit loader.
• Added sitemaps to allow easier access by crawlers.
• Provide direct links to string in notification e-mails or feeds.
• Përmirësime të ndryshme të ndërfaqes së përgjegjësit.
• Provide hints for production setup in admin interface.
• Added per language widgets and engage page.
• U përmirësa trajtimi i kycjes së përkthimeve.
• 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.
• Përmirësime të ndryshme kontrollesh cilësise.
• Support for per project ACL.
• Basic code coverage by unit tests.

4.32.7 Weblate 1.3

Released on November 16th 2012.

• Compatibility with PostgreSQL database backend.
• Removes languages removed in upstream git repository.
• U përminësua kryerja e kontolleve të cilësisë.
• Added new checks (BB code, 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.
• Mbulim më i mirë për burime Android.
• Support for generating SSH key from web interface.
• More visible data exports.
• New buttons to enter some special characters.
• Mbulim për eksportim fjalori.
• 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.
• U përminësua mbulimi për gjuhë DNM.

4.32.8 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.
• Mundësi për çaktivizim të disa përkrthimeve.
• Mundësi të formësueshme për shtim përkrthimesh të reja.
• Configuration of git commits per project.
- Mbrojtje e thjeshtë antispam.
- 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.
- Caching counts of strings with failing checks.
- Checking of requirements during setup.
- Documentation improvements.

### 4.32.9 Weblate 1.1

Released on July 4th 2012.

- U përmirësuan disa përkthime.
- 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.

### 4.32.10 Weblate 1.0

Hedhur në qarkullim më 10 maj, 2012.

- Improved validation while adding/saving component.
- Experimental support for Android component files (needs patched ttkit).
- Updates from hooks are run in background.
- U përmirësuan udhëzime instalimi.
- U përmirësua lëvizja në fjalar.

### 4.33 Weblate 0.x series

#### 4.33.1 Weblate 0.9

Released on April 18th 2012.

- U ndreq importimi i gjuhëve të panjohura.
- Improved listing of nearby messages.
- U përmirësuan disa kontolle.
- Documentation updates.
• Added definition for several more languages.
• Pastrimë të ndryshme kodi.
• Documentation improvements.
• 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 not translated strings ignores format string only messages.

4.33.2 Weblate 0.8

Hedhur në qarkullim më 3 prill, 2012.
• Replaced own full text search with Whoosh.
• Ndreqje dhe përtrimë të ndryshme të kontrolet.
• New command updatechecks.
• Plot përtrimë të përkatënesh.
• 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.
• Përputhshëmëri me Django 1.4.
• Ability to push changes to remote repo from web interface.
• Added review of translations done by others.

4.33.3 Weblate 0.7

Released on February 16th 2012.
• Mbulim i drejtpërdrejtë për njoftime GitHub-u.
• Added support for cleaning up orphaned checks and translations.
• Displays nearby strings while translating.
• Displays similar strings while translating.
• U përtrimësua kërkimë për varg.
4.33.4 Weblate 0.6

Released on February 14th 2012.

- U shtuan kontolle të ndryshme për mesazhe të përkthyer.
- 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.

4.33.5 Weblate 0.5

Released on February 12th 2012.

- Support for machine translation using following online services:
  - Apertium
  - Microsoft Translator
  - MyMemory
- Disa përkthime të reja.
- 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).

4.33.6 Weblate 0.4

Released on February 8th 2012.

- U shtua udhëzues përdorimi te dokumentimi.
- Fixed API hooks not to require CSRF protection.

4.33.7 Weblate 0.3

Released on February 8th 2012.

- Better display of source for plural translations.
- Dokumentim i ri, në formatin Sphinx.
- Displays secondary languages while translating.
- Improved error page to give list of existing projects.
- New per language stats.
4.33.8 Weblate 0.2

Hedhur në qarkullim më 7 shkurt, 2012.

- U përmirësua vlerësimi për disa formularë.
- Sinjalizoj vuqë përdoruesve përmirësim profili.
- Remember URL for login.
- Naming of text areas while entering plural forms.
- Automatic expanding of translation area.

4.33.9 Weblate 0.1

Released on February 6th 2012.

- Hedhja fillestare në qarkullim.
W
wic, 138
wic.config, 139
wic.main, 139
## HTTP Routing Table

<table>
<thead>
<tr>
<th>Method</th>
<th>Route</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANY</td>
<td>/</td>
<td>90</td>
</tr>
<tr>
<td>GET</td>
<td>/api</td>
<td>93</td>
</tr>
<tr>
<td>GET</td>
<td>/api/addons</td>
<td>127</td>
</tr>
<tr>
<td>PUT</td>
<td>/api/addons/(int:id)</td>
<td>128</td>
</tr>
<tr>
<td>DELETE</td>
<td>/api/addons/(int:id)</td>
<td>128</td>
</tr>
<tr>
<td>PATCH</td>
<td>/api/addons/(int:id)</td>
<td>127</td>
</tr>
<tr>
<td>GET</td>
<td>/api/addons/(int:id)</td>
<td>127</td>
</tr>
<tr>
<td>GET</td>
<td>/api/addons/(int:id)</td>
<td>128</td>
</tr>
<tr>
<td>PUT</td>
<td>/api/addons/(int:id)</td>
<td>128</td>
</tr>
<tr>
<td>DELETE</td>
<td>/api/addons/(int:id)</td>
<td>128</td>
</tr>
<tr>
<td>PATCH</td>
<td>/api/addons/(int:id)</td>
<td>127</td>
</tr>
<tr>
<td>GET</td>
<td>/api/changes</td>
<td>124</td>
</tr>
<tr>
<td>GET</td>
<td>/api/changes/(int:id)</td>
<td>124</td>
</tr>
<tr>
<td>GET</td>
<td>/api/component-lists</td>
<td>128</td>
</tr>
<tr>
<td>POST</td>
<td>/api/component-lists</td>
<td>128</td>
</tr>
<tr>
<td>PUT</td>
<td>/api/component-lists</td>
<td>128</td>
</tr>
<tr>
<td>DELETE</td>
<td>/api/component-lists</td>
<td>129</td>
</tr>
<tr>
<td>DELETE</td>
<td>/api/component-lists</td>
<td>129</td>
</tr>
<tr>
<td>PATCH</td>
<td>/api/component-lists</td>
<td>128</td>
</tr>
<tr>
<td>GET</td>
<td>/api/components</td>
<td>108</td>
</tr>
<tr>
<td>GET</td>
<td>/api/components/(string:project)/(string:component)</td>
<td>108</td>
</tr>
<tr>
<td>POST</td>
<td>/api/components/(string:project)/(string:component)</td>
<td>108</td>
</tr>
<tr>
<td>POST</td>
<td>/api/components/(string:project)/(string:component)/addons</td>
<td>127</td>
</tr>
<tr>
<td>POST</td>
<td>/api/components/(string:project)/(string:component)/links</td>
<td>116</td>
</tr>
<tr>
<td>POST</td>
<td>/api/components/(string:project)/(string:component)/languages</td>
<td>114</td>
</tr>
<tr>
<td>POST</td>
<td>/api/components/(string:project)/(string:component)/lock</td>
<td>112</td>
</tr>
<tr>
<td>POST</td>
<td>/api/components/(string:project)/(string:component)/monolingual_base</td>
<td>114</td>
</tr>
<tr>
<td>POST</td>
<td>/api/components/(string:project)/(string:component)/new_template</td>
<td>114</td>
</tr>
<tr>
<td>POST</td>
<td>/api/components/(string:project)/(string:component)/repository</td>
<td>113</td>
</tr>
<tr>
<td>POST</td>
<td>/api/components/(string:project)/(string:component)/screenshots</td>
<td>112</td>
</tr>
<tr>
<td>POST</td>
<td>/api/components/(string:project)/(string:component)/statistics</td>
<td>116</td>
</tr>
<tr>
<td>POST</td>
<td>/api/components/(string:project)/(string:component)/translations</td>
<td>114</td>
</tr>
<tr>
<td>PUT</td>
<td>/api/components/(string:project)/(string:component)</td>
<td>111</td>
</tr>
<tr>
<td>DELETE</td>
<td>/api/components/(string:project)/(string:project_slug)</td>
<td>111</td>
</tr>
<tr>
<td>PATCH</td>
<td>/api/components/(string:project)/(string:component)</td>
<td>110</td>
</tr>
<tr>
<td>GET</td>
<td>/api/groups</td>
<td>96</td>
</tr>
<tr>
<td>GET</td>
<td>/api/groups/(int:id)</td>
<td>96</td>
</tr>
<tr>
<td>POST</td>
<td>/api/groups</td>
<td>96</td>
</tr>
<tr>
<td>POST</td>
<td>/api/groups/(int:id)/componentlists</td>
<td>98</td>
</tr>
<tr>
<td>POST</td>
<td>/api/groups/(int:id)/components</td>
<td>98</td>
</tr>
<tr>
<td>POST</td>
<td>/api/groups/(int:id)/languages</td>
<td>98</td>
</tr>
<tr>
<td>POST</td>
<td>/api/groups/(int:id)/links</td>
<td>98</td>
</tr>
</tbody>
</table>
POST /api/groups/(int:id)/projects/, 98
POST /api/groups/(int:id)/roles/, 98
PUT /api/groups/(int:id)/, 97
DELETE /api/groups/(int:id)/, 97
DELETE /api/groups/(int:id)/componentlists/(int:component_list_id), 99
DELETE /api/groups/(int:id)/components/(int:component_id), 98
DELETE /api/groups/(int:id)/languages/(string:language_code), 98
DELETE /api/groups/(int:id)/projects/(int:project_id), 98
PATCH /api/groups/(int:id)/, 97

GET /api/languages/, 100
GET /api/languages/(string:language)/, 100
GET /api/languages/(string:language)/statistics/, 101
POST /api/languages/, 100
PUT /api/languages/(string:language)/, 101
DELETE /api/languages/(string:language)/, 101
PATCH /api/languages/(string:language)/, 101

GET /api/translations/, 117
GET /api/translations/(string:project)/(string:component)/(string:language)/, 117
GET /api/translations/(string:project)/(string:component)/(string:language)/changes/, 119
GET /api/translations/(string:project)/(string:component)/(string:language)/file/, 120
GET /api/translations/(string:project)/(string:component)/(string:language)/repository/, 121
GET /api/translations/(string:project)/(string:component)/(string:language)/statistics/, 121
GET /api/translations/(string:project)/(string:component)/(string:language)/units/, 119
POST /api/translations/(string:project)/(string:component)/(string:language)/autotranslate/, 120
POST /api/translations/(string:project)/(string:component)/(string:language)/file/, 120
POST /api/translations/(string:project)/(string:component)/(string:language)/repository/, 121
POST /api/translations/(string:project)/(string:component)/(string:language)/units/, 119
DELETE /api/translations/(string:project)/(string:component)/(string:language)/, 119

GET /api/units/, 122
GET /api/units/(int:id)/, 122
PUT /api/units/(int:id)/, 123

GET /api/roles/, 99
GET /api/roles/(int:id)/, 99
POST /api/roles/, 99
PUT /api/roles/(int:id)/, 99
DELETE /api/roles/(int:id)/, 100
PATCH /api/roles/(int:id)/, 100

GET /api/screenshots/(string:language_code)/, 125
GET /api/screenshots/(int:id)/, 125
GET /api/screenshots/(int:id)/file/, 125
POST /api/screenshots/, 126
POST /api/screenshots/(int:id)/file/, 125
POST /api/screenshots/(int:id)/units/, 126
PUT /api/screenshots/(int:id)/, 126
DELETE /api/screenshots/(int:id)/units/(int:unit_id), 126
PATCH /api/screenshots/(int:id)/, 126

GET /api/tasks/, 129
GET /api/tasks/(str:uuid)/, 129

GET /api/metrics/, 130

GET /api/projects/, 102
GET /api/projects/(string:project)/, 102
GET /api/projects/(string:project)/changes/, 103
GET /api/projects/(string:project)/components/, 104
GET /api/projects/(string:project)/languages/, 107
GET /api/projects/(string:project)/repository/, 103
GET /api/projects/(string:project)/statistics/, 107
POST /api/projects/, 102
POST /api/projects/(string:project)/components/, 105
POST /api/projects/(string:project)/repository/, 104
PUT /api/projects/(string:project)/, 103
DELETE /api/projects/(string:project)/, 103
PATCH /api/projects/(string:project)/, 103

HTTP Routing Table

449
### DELETE
```
DELETE /api/units/(int:id)/, 124
```

### PATCH
```
PATCH /api/units/(int:id)/, 123
```

### /api/users
```
GET /api/users/, 93
GET /api/users/(str:username)/, 93
GET /api/users/(str:username)/notifications/, 95
GET /api/users/(str:username)/notifications/(int:subscription_id)/, 95
GET /api/users/(str:username)/statistics/, 95
POST /api/users/, 93
POST /api/users/(str:username)/groups/, 95
POST /api/users/(str:username)/notifications/, 95
PUT /api/users/(str:username)/, 94
PUT /api/users/(str:username)/notifications/(int:subscription_id)/, 95
DELETE /api/users/(str:username)/, 95
DELETE /api/users/(str:username)/notifications/(int:subscription_id)/, 96
PATCH /api/users/(str:username)/, 94
PATCH /api/users/(str:username)/notifications/(int:subscription_id)/, 96
```

### /exports
```
GET /exports/rss/, 133
GET /exports/rss/(string:project)/, 133
GET /exports/rss/(string:project)/(string:component)/, 133
GET /exports/rss/(string:project)/(string:component)/(string:language)/, 133
GET /exports/rss/language/(string:language)/, 133
GET /exports/stats/(string:project)/(string:component)/, 133
```

### /hooks
```
GET /hooks/update/(string:project)/, 130
GET /hooks/update/(string:project)/(string:component)/, 130
POST /hooks/azure/, 131
POST /hooks/bitbucket/, 131
POST /hooks/gitea/, 131
POST /hooks/gitee/, 131
POST /hooks/github/, 130
POST /hooks/gitlab/, 131
POST /hooks/pagure/, 131
```

### HTTP Routing Table
Simbole

.XML resource file
  file format, 76
  --add
    Mundësi për rresht urdhrash
      auto_translate, 348
  --addon ADDON
    Mundësi për rresht urdhrash
      install_addon, 354
  --age HOURS
    Mundësi për rresht urdhrash
      commit_pending, 349
  --author USER@EXAMPLE.COM
    Mundësi për rresht urdhrash
      add_suggestions, 348
  --base-file-template TEMPLATE
    Mundësi për rresht urdhrash
      import_project, 352
  --check
    Mundësi për rresht urdhrash
      importusers, 354
  --config PATH
    Mundësi për rresht urdhrash
      wlc, 135
  --config-section SECTION
    Mundësi për rresht urdhrash
      wlc, 135
  --configuration CONFIG
    Mundësi për rresht urdhrash
      install_addon, 354
  --convert
    Mundësi për rresht urdhrash
      wlc, 136
  --email USER@EXAMPLE.COM
    Mundësi për rresht urdhrash
      createadmin, 350
  --file-format FORMAT
    Mundësi për rresht urdhrash
      import_project, 352
  --force
    Mundësi për rresht urdhrash
      loadpo, 355
  --force-commit
    Mundësi për rresht urdhrash
      pushgit, 356
  --format {csv, json, text, html}
cleanup
  Mundësi për rresht urdhrash wlc, 136
cleanuptrans
  weblate admin command, 349
Comma separated values
  file format, 76
Command (klasë te wlc.main), 140
COMMENT_CLEANUP_DAYS
  setting, 306
commit
  Mundësi për rresht urdhrash wlc, 135
commit_pending
  weblate admin command, 349
COMMIT_PENDING_HOURS
  setting, 307
commitgit
  weblate admin command, 349
configure()
  (metodë weblate.addons.base.BaseAddon), 391
CONTACT_FORM
  setting, 307
createadmin
  weblate admin command, 350
CSP_CONNECT_SRC
  setting, 306
CSP_FONT_SRC
  setting, 306
CSP_IMG_SRC
  setting, 306
CSP_SCRIPT_SRC
  setting, 306
CSP_STYLE_SRC
  setting, 306
CSV
  file format, 76
D
daily()
  (metodë weblate.addons.base.BaseAddon), 391
DATA_DIR
  setting, 307
DATABASE_BACKUP
  setting, 308
DATABASES
  setting, 183
DEBUG
  setting, 183
DEFAULT_ACCESS_CONTROL
  setting, 308
DEFAULT_ADD_MESSAGE
  setting, 308
DEFAULT_ADDON_MESSAGE
  setting, 308
DEFAULT_ADDONS
  setting, 309
DEFAULT_AUTO_WATCH
  setting, 308
DEFAULT_COMMIT_MESSAGE
  setting, 308
DEFAULT_COMMITTER_EMAIL
  setting, 309
DEFAULT_COMMITTER_NAME
  setting, 309
DEFAULT_DELETE_MESSAGE
  setting, 308
DEFAULT_FROM_EMAIL
  setting, 183
DEFAULT_LANGUAGE
  setting, 309
DEFAULT_MERGE_MESSAGE
  setting, 308
DEFAULT_MERGE_STYLE
  setting, 310
DEFAULT_PAGE_LIMIT
  setting, 323
DEFAULT_PULL_MESSAGE
  setting, 310
DEFAULT_RESTRICTED_COMPONENT
  setting, 308
DEFAULT_SHARED_TM
  setting, 310
DEFAULT_TRANSLATION_PROPAGATION
  setting, 310
download
  Mundësi për rresht urdhrash wlc, 136
DTD
  file format, 78
dump_memory
  weblate admin command, 350
dumpuserdata
  weblate admin command, 350
E
ENABLE_AVATARS
  setting, 310
ENABLE_HOOKS
  setting, 310
ENABLE_HTTPS
  setting, 311
ENABLE_SHARING
  setting, 311
F
file format
  .XML resource file, 76
  Android, 71
  Apple strings, 72
  ARB, 75
  Comma separated values, 76
  CSV, 76
  DTD, 78
gettext
  65
go-i18n
  74
GWT properties
  69
  70
i18next
  74
INI translations
  69
  70
Java properties, 68
Joomla translations, 70
JSON, 73
mi18n.lang, 69
PHP strings, 72
PO, 65
Qt, 70
RC, 79
RESX, 76
Ruby YAML, 77
Ruby YAML Ain’t Markup Language, 77
string resources, 71
TS, 70
XLIFF, 66
XML, 78
YAML, 77
YAML Ain’t Markup Language, 77

G
get() (metodë wlc.Weblate), 138
get_add_form() (metodë klase
weblate.addons.base.BaseAddon), 391
GET_HELP_URL
setting, 311
get_settings_form() (metodë
weblate.addons.base.BaseAddon), 391
gettext
file format, 65
GITHUB_CREDENTIALS
setting, 312
GITHUB_TOKEN
setting, 312
GITHUB_USERNAME
setting, 312
GITLAB_CREDENTIALS
setting, 311
GITLAB_TOKEN
setting, 312
GITLAB_USERNAME
setting, 311
go-i18n
file format, 74
GOOGLE_ANALYTICS_ID
setting, 312
GWT properties
file format, 69

H
HIDE_REPO_CREDENTIALS
setting, 313
HIDE_VERSION
setting, 313

I
i18next
file format, 74
import_demo
weblate admin command, 350
import_json
weblate admin command, 351
import_memory
weblate admin command, 351
import_project
weblate admin command, 352
importuserdata
weblate admin command, 354
importusers
weblate admin command, 354
INI translations
file format, 69, 70
install_addon
weblate admin command, 354
IP_BEHIND.Reverse_PROXY
setting, 313
IP_PROXY_HEADER
setting, 313
IP_PROXY_OFFSET
setting, 314
iPad
translation, 72
iPhone
translation, 72

J
Java properties
file format, 68
Joomla translations
file format, 70
JSON
file format, 73

L
LEGAL_URL
setting, 314
LICENSE_EXTRA
setting, 314
LICENSE_FILTER
setting, 315
LICENSE_REQUIRED
setting, 315
LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH
setting, 315
list_languages
weblate admin command, 354
list_translators
weblate admin command, 355
listVersions
weblate admin command, 355
list-components
Mundësi për rresht urdhrash wlc, 135
list-languages
Mundësi për rresht urdhrash wlc, 135
list-projects
Mundësi për rresht urdhrash wlc, 135
list-translations
Mundësi për rresht urdhrash wlc, 135
The Weblate Manual, Hedhje Në Qarkullim 4.9

load() (metodë wlc.config.WeblateConfig), 139
loadpo
  weblate admin command, 355
LOCALIZE_CDN_PATH
  setting, 315
LOCALIZE_CDN_URL
  setting, 315
lock
  Mundësi për rresht urdhrash wlc, 136
lock_translation
  weblate admin command, 355
lock-status
  Mundësi për rresht urdhrash wlc, 136
LOGIN_REQUIRED_URLS
  setting, 316
LOGIN_REQUIRED_URLS_EXCEPTIONS
  setting, 316
ls
  Mundësi për rresht urdhrash wlc, 135

M
MACHINE_TRANSLATION_SERVICES
  setting, 317
main() (te moduli wlc.main), 139
MATOMO_SITE_ID
  setting, 316
MATOMO_URL
  setting, 317
mi18n lang
  file format, 69
modul
  wlc, 138
  wlc.config, 139
  wlc.main, 139
monolingual
  translation, 63
move_language
  weblate admin command, 356
MT_APERTIUM_API
  setting, 318
MT_AWS_ACCESS_KEY_ID
  setting, 318
MT_AWS_REGION
  setting, 318
MT_AWS_SECRET_ACCESS_KEY
  setting, 318
MT_BAIDU_ID
  setting, 318
MT_BAIDU_SECRET
  setting, 318
MT_DEEPL_API_URL
  setting, 319
MT_DEEPL_KEY
  setting, 319
MT_GOOGLE_CREDENTIALS
  setting, 320
MT_GOOGLE_KEY
  setting, 320
MT_GOOGLE_LOCATION
  setting, 320
MT_GOOGLE_PROJECT
  setting, 320
MT_LIBRETRANSLATE_API_URL
  setting, 319
MT_LIBRETRANSLATE_KEY
  setting, 319
MT_MICROSOFT_BASE_URL
  setting, 320
MT_MICROSOFT_COGNITIVE_KEY
  setting, 320
MT_MICROSOFT_ENDPOINT_URL
  setting, 321
MT_MICROSOFT_REGION
  setting, 321
MT_MODERNMT_KEY
  setting, 321
MT_MODERNMT_URL
  setting, 321
MT_MYMEMORY_EMAIL
  setting, 321
MT_MYMEMORY_KEY
  setting, 321
MT_MYMEMORY_USER
  setting, 321
MT_NETEASE_KEY
  setting, 322
MT_NETEASE_SECRET
  setting, 322
MT_SAP_BASE_URL
  setting, 322
MT_SAP_PASSWORD
  setting, 323
MT_SAP_SANDBOX_APIKEY
  setting, 323
MT_SAP_USE_MT
  setting, 323
MT_SAP_USERNAME
  setting, 323
MT_SERVICES
  setting, 317
MT_TMSERVER
  setting, 322
MT_YANDEX_KEY
  setting, 322
MT_YOUDAO_ID
  setting, 322
MT_YOUDAO_SECRET
  setting, 322
Mundësi për rresht urdhrash
  add_suggestions
    --add, 348
    --inconsistent, 348
    --mode MODE, 348
WEBLATE_REMOVE_ADDONS, 159
WEBLATE_REMOVE_APPS, 159
WEBLATE_REMOVE_AUTOFIX, 159
WEBLATE_REMOVE_CHECK, 159
WEBLATE_REMOVE_LOGIN_REQUIRED_URLS_EXCEPTIONS, 149
WEBLATE_REQUIRE_LOGIN, 148, 327
WEBLATE_SAML_IDP_ENTITY_ID, 155
WEBLATE_SAML_IDP_URL, 155
WEBLATE_SAML_IDP_X509CERT, 155
WEBLATE_SECURE_PROXY_SSL_HEADER, 148
WEBLATE_SERVER_EMAIL, 147
WEBLATE_SERVICE, 145, 160
WEBLATE_SILENCED_SYSTEM_CHECKS, 150, 212
WEBLATE_SIMPLIFY_LANGUAGES, 149
WEBLATE_SITE_DOMAIN, 146, 185, 203, 328
WEBLATE_SITE_TITLE, 146
WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_KEY, 154
WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_SECRET, 154
WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_KEY, 154
WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_SECRET, 154
WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_TENANT_ID, 154
WEBLATE_SOCIAL_AUTH_BITBUCKET_KEY, 153
WEBLATE_SOCIAL_AUTH_BITBUCKET_SECRET, 153
WEBLATE_SOCIAL_AUTH_FACEBOOK_KEY, 154
WEBLATE_SOCIAL_AUTH_FACEBOOK_SECRET, 154
WEBLATE_SOCIAL_AUTH_FEDORA, 155
WEBLATE_SOCIAL_AUTH_GITHUB_KEY, 153
WEBLATE_SOCIAL_AUTH_GITHUB_SECRET, 153
WEBLATE_SOCIAL_AUTH_GITLAB_API_URL, 154
WEBLATE_SOCIAL_AUTH_GITLAB_KEY, 154
WEBLATE_SOCIAL_AUTH_GITLAB_SECRET, 154
WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_KEY, 154
WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET, 154
WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_WHITELISTED_DOMAINS, 154
WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_WHITELISTED_EMAILS, 154
WEBLATE_SOCIAL_AUTH_KEYCLOAK_ACCESS_TOKEN_URL, 154
WEBLATE_SOCIAL_AUTH_KEYCLOAK_ALGORITHM, 154
WEBLATE_SOCIAL_AUTH_KEYCLOAK_AUTHORIZATION_URL, 154
WEBLATE_SOCIAL_AUTH_KEYCLOAK_KEY, 154
WEBLATE_SOCIAL_AUTH_KEYCLOAK_PUBLIC_KEY, 154
WEBLATE_SOCIAL_AUTH_KEYCLOAK_SECRET, 154
WEBLATE_SOCIAL_AUTH_OPENSUSE, 155
WEBLATE_SOCIAL_AUTH_SLACK_KEY, 155
WEBLATE_SOCIAL_AUTH𝑄 UBE NTU, 155
WEBLATE_STATUS_URL, 158
WEBLATE_TIME_ZONE, 147
WEBLATE_URL_PREFIX, 150
WEBLATE_WEBSITE_REQUIRED, 150
WEBLATE_WORKERS, 145, 159, 160
WL_BRANCH, 299
WL_COMPONENT_NAME, 300
WL_COMPONENT_SLUG, 300
WL_FILE_FORMAT, 299
WL_FILEMASK, 299
WL_LANGUAGE, 300
WL_NEW_BASE, 299
WL_PREVIOUS_HEAD, 300
WL_PROJECT_NAME, 300
WL_PROJECT_SLUG, 300
WL_REPO, 299
WL_TEMPLATE, 299
WL_VCS, 299
NEARBY_MESSAGES, 323
PIWIK_SITE_ID, 316
PIWIK_URL, 317
PIWIK_SITE_ID, 316
PIWIK_URL, 317
PHP strings, 72
post() (metodë wlc.Weblate), 139
post_add() (metodë weblate.addons.base.BaseAddon), 391
post_commit() (metodë weblate.addons.base.BaseAddon), 391
post_push() (metodë weblate.addons.base.BaseAddon), 391
post_update() (metodë weblate.addons.base.BaseAddon), 391
pre_commit() (metodë weblate.addons.base.BaseAddon), 391
pre_push() (metodë weblate.addons.base.BaseAddon), 391
pre_update() (metodë weblate.addons.base.BaseAddon), 391

PRIVACY_URL setting, 324
pull Mundësi për rresht urdhrash wlc, 135
push Mundësi për rresht urdhrash wlc, 136
pushgit weblate admin command, 356
Python, 138

Q
Qt file format, 70

R
RATELIMIT_ATTEMPTS setting, 324
RATELIMIT_LOCKOUT setting, 325
RATELIMIT_WINDOW setting, 325
RC file format, 79
register_command() (te moduli wlc.main), 139
REGISTRATION_ALLOW_BACKENDS setting, 325
REGISTRATION_CAPTCHA setting, 325
REGISTRATION_EMAIL_MATCH setting, 326
REGISTRATION_OPEN setting, 326
repo Mundësi për rresht urdhrash wlc, 136
REPOSITORY_ALERT_THRESHOLD setting, 326
REQUIRE_LOGIN setting, 326
reset Mundësi për rresht urdhrash wlc, 136
REST, 90
RESX file format, 76
RFC RFC 5646, 62
Ruby YAML file format, 77
Ruby YAML Ain't Markup Language file format, 77
The Weblate Manual, Hedhje Në Qarkullim 4.9

DEFAULT_TRANSLATION_PROPAGATION, 310
ENABLE_AVATARS, 310
ENABLE_HOOKS, 310
ENABLE_HTTPS, 311
ENABLE_SHARING, 311
GET_HELP_URL, 311
GITHUB_CREDENTIALS, 312
GITHUB_TOKEN, 312
GITHUB_USERNAME, 312
GITLAB_CREDENTIALS, 311
GITLAB_TOKEN, 312
GITLAB_USERNAME, 311
GOOGLE_ANALYTICS_ID, 312
HIDE_REPO_CREDENTIALS, 313
HIDE_VERSION, 313
IP_BEHIND_REVERSE_PROXY, 313
IP_PROXY_HEADER, 313
IP_PROXY_OFFSET, 314
LEGAL_URL, 314
LICENSE_EXTRA, 314
LICENSE_FILTER, 315
LICENSE_REQUIRED, 315
LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH, 315
LOCALIZE_CDN_PATH, 315
LOCALIZE_CDN_URL, 315
LOGIN_REQUIRED_URLS, 316
LOGIN_REQUIRED_URLS_EXCEPTIONS, 316
MACHINE_TRANSLATION_SERVICES, 317
MATOMO_SITE_ID, 316
MATOMO_URL, 317
MT_APERTIUM_API, 318
MT_AWS_ACCESS_KEY_ID, 318
MT_AWS_REGION, 318
MT_AWS_SECRET_ACCESS_KEY, 318
MT_BAIDU_ID, 318
MT_BAIDU_SECRET, 318
MT_DEEPL_API_URL, 319
MT_DEEPL_KEY, 319
MT_GOOGLE_CREDENTIALS, 320
MT_GOOGLE_KEY, 320
MT_GOOGLE_LOCATION, 320
MT_GOOGLE_PROJECT, 320
MT_LIBRETRANSLATE_API_URL, 319
MT_LIBRETRANSLATE_KEY, 319
MT_MICROSOFT_BASE_URL, 320
MT_MICROSOFT_COGNITIVE_KEY, 320
MT_MICROSOFT_ENDPOIINT_URL, 321
MT_MICROSOFT_REGION, 321
MT_MODERNMT_KEY, 321
MT_MODERNMT_URL, 321
MT_MYMEMORY_EMAIL, 321
MT_MYMEMORY_KEY, 321
MT_MYMEMORY_USER, 321
MT_NETEASE_KEY, 322
MT_NETEASE_SECRET, 322
MT_SAP_BASE_URL, 322
MT_SAP_PASSWORD, 323
MT_SAP_SANDBOX_APIKEY, 323
MT_SAP_USE_MT, 323
MT_SAP_USERNAME, 323
MT_SERVICES, 317
MT_TMSERVER, 322
MT_YANDEX_KEY, 322
MT_YOUDAO_ID, 322
MT_YOUDAO_SECRET, 322
NEARBY_MESSAGES, 323
PAGURE_CREDENTIALS, 323
PAGURE_TOKEN, 324
PAGURE_USERNAME, 324
PIWIK_SITE_ID, 316
PIWIK_URL, 317
PRIVATE_URL, 324
RATELIMIT_ATTEMPTS, 324
RATELIMIT_LOCKOUT, 325
RATELIMIT_WINDOW, 325
REGISTRATION_ALLOW_BACKENDS, 325
REGISTRATION_CAPTCHA, 325
REGISTRATION_EMAIL_MATCH, 326
REGISTRATION_OPEN, 326
REPOSITORY_ALERT_THRESHOLD, 326
REQUIRE_LOGIN, 326
SECRET_KEY, 183
SENTRY_DSN, 327
SERVER_EMAIL, 184
SESSION_COOKIE_AGE_AUTHENTICATED, 327
SESSION_ENGINE, 183
SIMPLIFY_LANGUAGES, 327
SINGLE_PROJECT, 328
SITE_DOMAIN, 327
SITE_TITLE, 328
SPECIAL_CHARS, 328
STATUS_URL, 328
SUGGESTION_CLEANUP_DAYS, 328
UPDATE_LANGUAGES, 328
URL_PREFIX, 328
VCS_BACKENDS, 329
VCS_CLONE_DEPTH, 329
WEBLATE_ADDONS, 329
WEBLATE_EXPORTERS, 330
WEBLATE_FORMATS, 330
WEBSITE_REQUIRED, 331
setupgroups
weblate admin command, 357
setulang
weblate admin command, 357
show
Mundësi për rresht urdhrash wlc, 135
SIMPLIFY_LANGUAGES
setting, 327
SINGLE_PROJECT
setting, 328
SITE_DOMAIN

460  Tregues
weblate.addons.base.BaseAddon
weblate.addons.base.BaseAddon
weblate.addons.base.BaseAddon
weblate.addons.base.BaseAddon
weblate.addons.base.BaseAddon

setting, 331
wlc, 133
  modul, 138
wlc.config
  modul, 139
wlc.main
  modul, 139

X
XLIFF
  file format, 66
XML
  file format, 78

Y
YAML
  file format, 77
YAML Ain't Markup Language
  file format, 77