The Weblate Manual

Release 4.14

Michal Čihař

22.08.2022

User docs

KAPITEL 1

User docs

1.1 Weblate basics

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 *Android string resources*). The projects are there to help you organize component into logical sets (for example to group all translations used within one application).

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

Siehe auch:

../devel/integration

1.2 Registration and user profile

1.2.1 Registrierung

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

You can register by following a few simple steps:

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

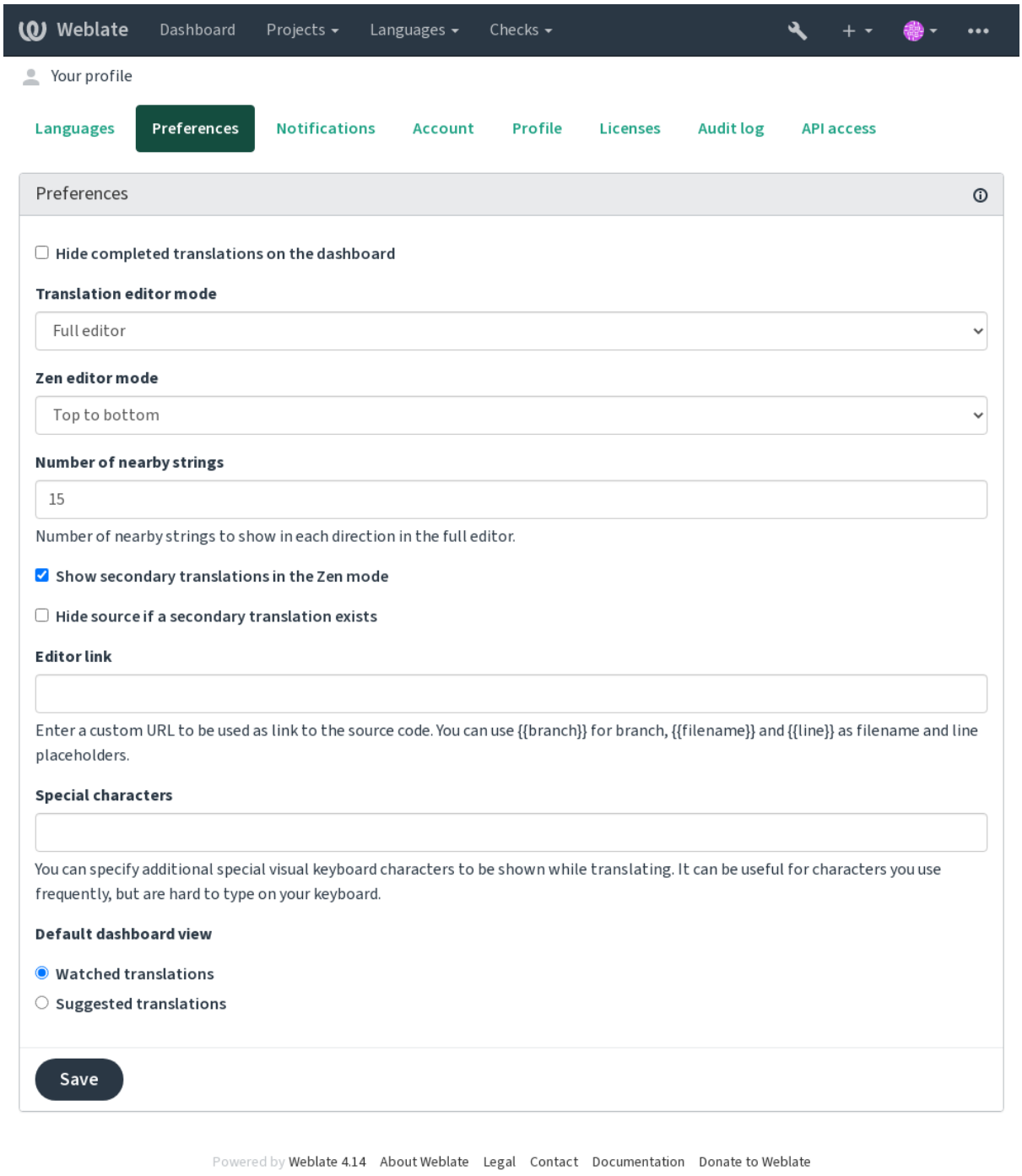
1.2.2 Übersicht

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

Neu in Version 2.5.

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

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



The menu has these options:

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

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

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

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

1.2.3 Benutzerprofil

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.

Bemerkung: All language selections only offer currently translated languages.

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

Sprachen

1.2.4 Interface language

Choose the language you want to display the UI in.

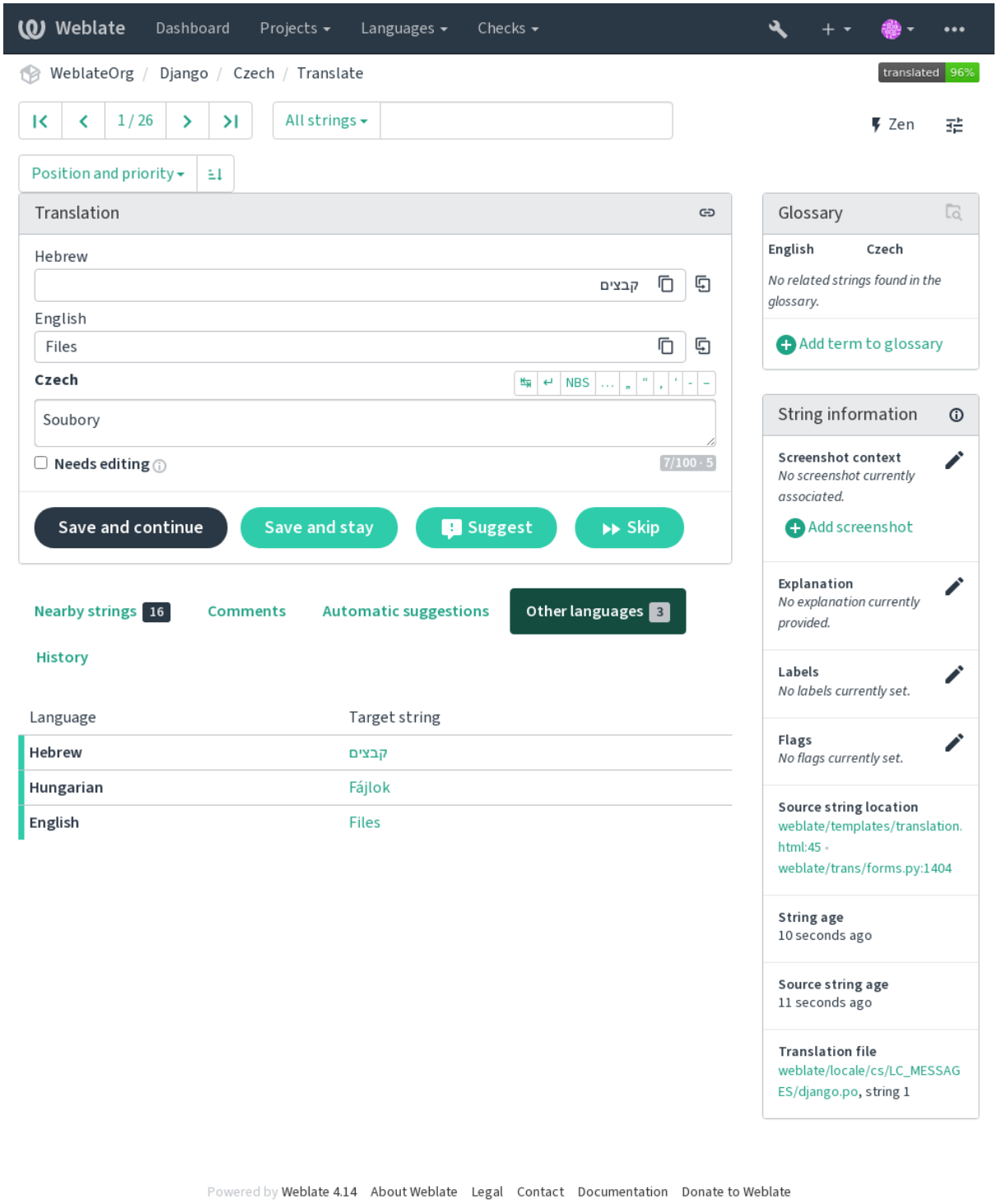
Übersetzte Sprachen

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.


Weblate
Dashboard
Projects
Languages
Checks
Dashboard
Watched translations 13
Suggested translations 5
Insights
Search
Translation
Translated
Unfinished
Unfinished words
Checks
Suggestions
WeblateOrg/Android — Czech
MIT
76%
3
3
WeblateOrg/Django — Hebrew
GPL-3.0
92%
2
15
WeblateOrg/Django — Czech
GPL-3.0
96%
1
12
4
WeblateOrg/Django — Hungarian
GPL-3.0
69%
8
109
1
WeblateOrg/Djangojs — Hebrew
GPL-3.0
WeblateOrg/Djangojs — Hungarian
GPL-3.0
96%
2
6
WeblateOrg/Djangojs — Czech
GPL-3.0
WeblateOrg/Language names — Czech
GPL-3.0
WeblateOrg/Language names — Hebrew
GPL-3.0
WeblateOrg/Language names — Hungarian
GPL-3.0
81%
4
5
WeblateOrg/WeblateOrg — Hebrew
GPL-3.0
WeblateOrg/WeblateOrg — Czech
GPL-3.0
WeblateOrg/WeblateOrg — Hungarian
GPL-3.0
Powered by Weblate 4.14
About Weblate
Legal
Contact
Documentation
Donate to Weblate

Zweitsprachen

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:



1.2.5 Einstellungen

Standardansicht der Übersichtsseite

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.

Siehe auch:

Component Lists

Editor-URL

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

Hinweis: 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.

Siehe auch:

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

Sonderzeichen

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

1.2.6 Benachrichtigungen

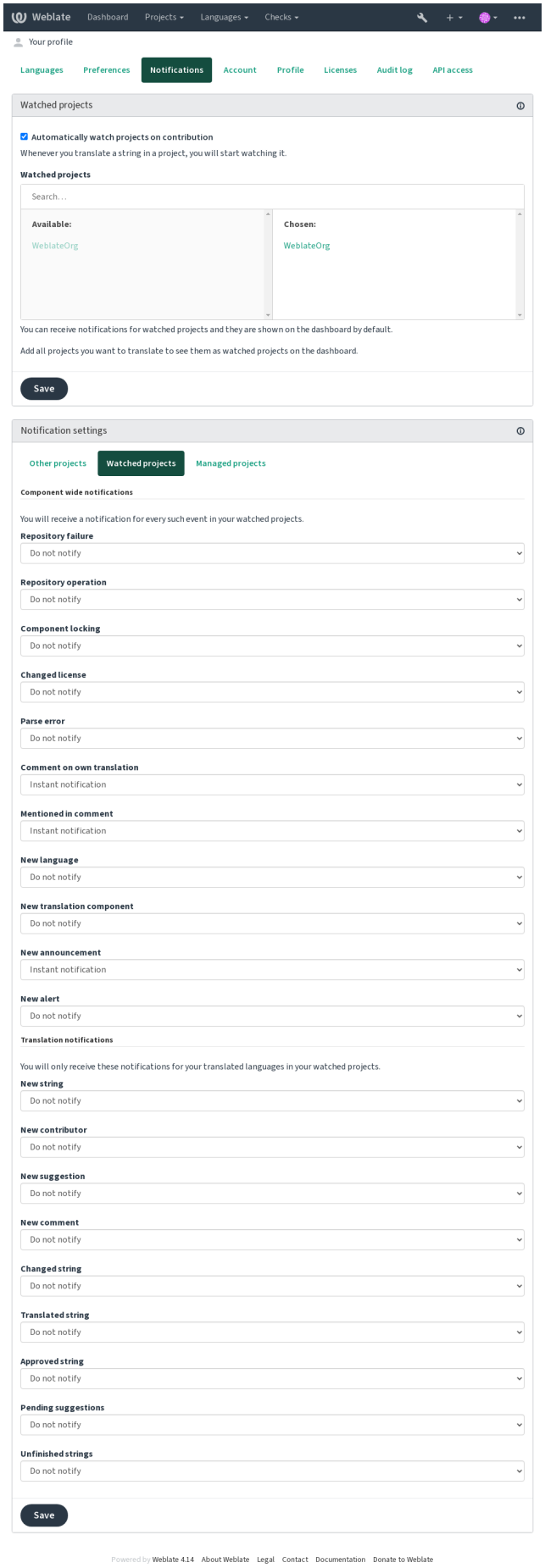
Subscribe to various notifications from the *Notifications* tab. Notifications for selected events on watched or administered projects will be sent to you per e-mail.

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`.

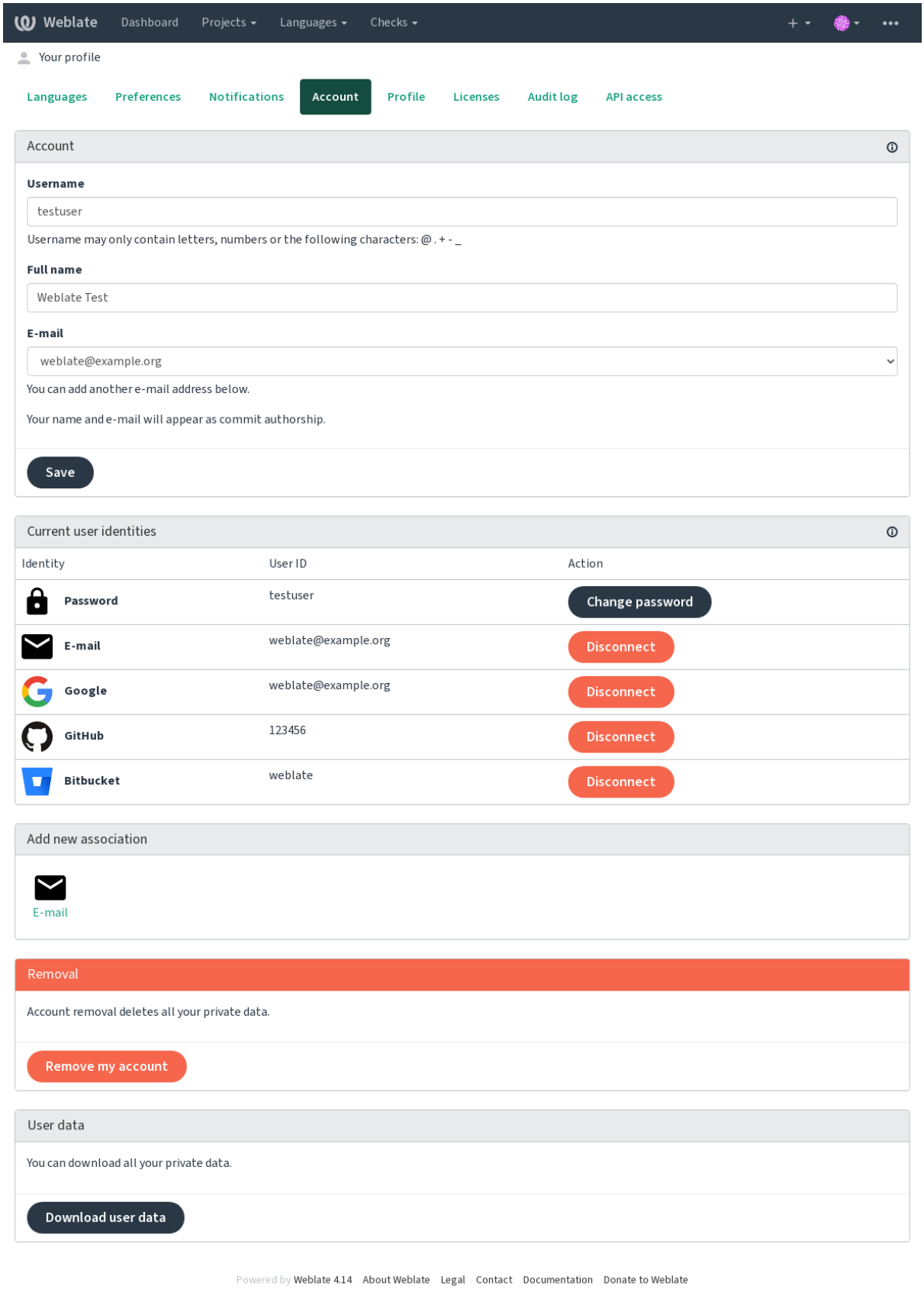
Bemerkung: You will not receive notifications for your own actions.


Weblate
Dashboard
Projects
Languages
Checks
Your profile
Languages
Preferences
Notifications
Account
Profile
Licenses
Audit log
API access
Watched projects
Automatically watch projects on contribution
Whenever you translate a string in a project, you will start watching it.
Watched projects
Search...
Available:
WeblateOrg
Chosen:
WeblateOrg
You can receive notifications for watched projects and they are shown on the dashboard by default.
Add all projects you want to translate to see them as watched projects on the dashboard.
Save
Notification settings
Other projects
Watched projects
Managed projects
Component wide notifications
You will receive a notification for every such event in your watched projects.
Repository failure
Do not notify
Repository operation
Do not notify
Component locking
Do not notify
Changed license
Do not notify
Parse error
Do not notify
Comment on own translation
Instant notification
Mentioned in comment
Instant notification
New language
Do not notify
New translation component
Do not notify
New announcement
Instant notification
New alert
Do not notify
Translation notifications
You will only receive these notifications for your translated languages in your watched projects.
New string
Do not notify
New contributor
Do not notify
New suggestion
Do not notify
New comment
Do not notify
Changed string
Do not notify
Translated string
Do not notify
Approved string
Do not notify
Pending suggestions
Do not notify
Unfinished strings
Do not notify
Save
Powered by Weblate 4.14
About Weblate
Legal
Contact
Documentation
Donate to Weblate

1.2.7 Benutzerkonto

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

Bemerkung: The list of services depends on your Weblate configuration, but can be made to include popular sites such as GitLab, GitHub, Google, Facebook, or Bitbucket or other OAuth 2.0 providers.


Weblate
Dashboard
Projects
Languages
Checks
Your profile
Languages
Preferences
Notifications
Account
Profile
Licenses
Audit log
API access
Account
Username
testuser
Username may only contain letters, numbers or the following characters: @ . + - _
Full name
Weblate Test
E-mail
weblate@example.org
You can add another e-mail address below.
Your name and e-mail will appear as commit authorship.
Save
Current user identities
Identity
User ID
Action
Password
testuser
Change password
E-mail
weblate@example.org
Disconnect
Google
weblate@example.org
Disconnect
GitHub
123456
Disconnect
Bitbucket
weblate
Disconnect
Add new association
E-mail
Removal
Account removal deletes all your private data.
Remove my account
User data
You can download all your private data.
Download user data
Powered by Weblate 4.14
About Weblate
Legal
Contact
Documentation
Donate to Weblate

1.2.8 Benutzerprofil

Alle Felder auf dieser Seite sind optional und können jederzeit gelöscht werden. Indem Sie diese Felder ausfüllen, geben Sie uns Ihr Einverständnis, diese Daten überall dort weiterzugeben, wo Ihr Benutzerprofil erscheint.

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

1.2.9 Lizenzen

1.2.10 API-Zugang

You can get or reset your API access token here.

1.2.11 Audit-Protokoll

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.

Siehe auch:

Running behind reverse proxy

1.3 Translating using Weblate

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

Overall, there are two modes of translation:

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

Please see *Übersetzungs-Workflows* for more info on translation workflow.

Options for translation project visibility:

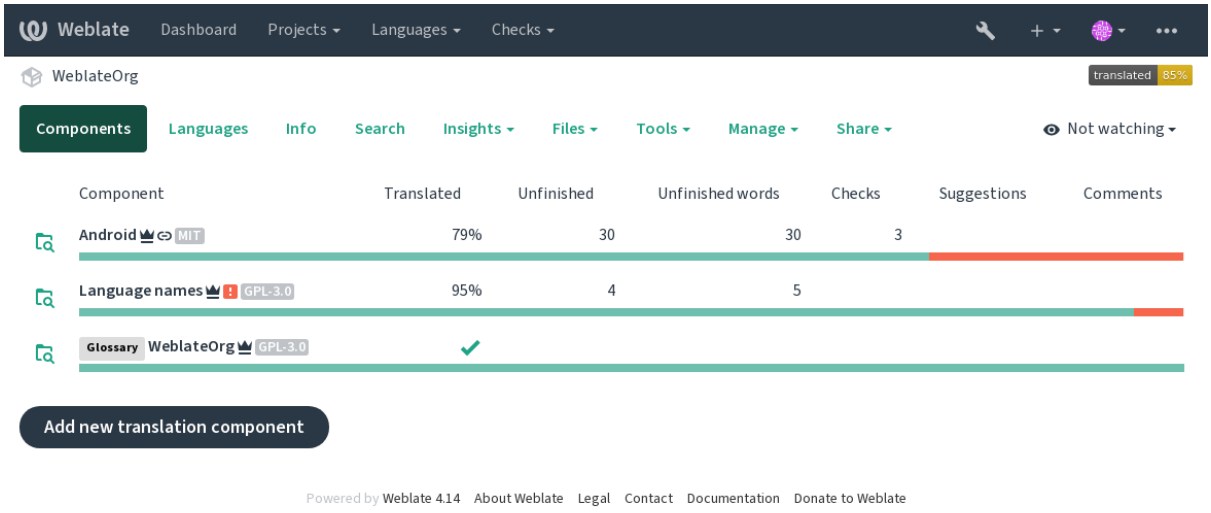
- Publicly visible
- Visible only to a certain group of translators

Siehe auch:

Zugriffssteuerung, *Übersetzungs-Workflows*

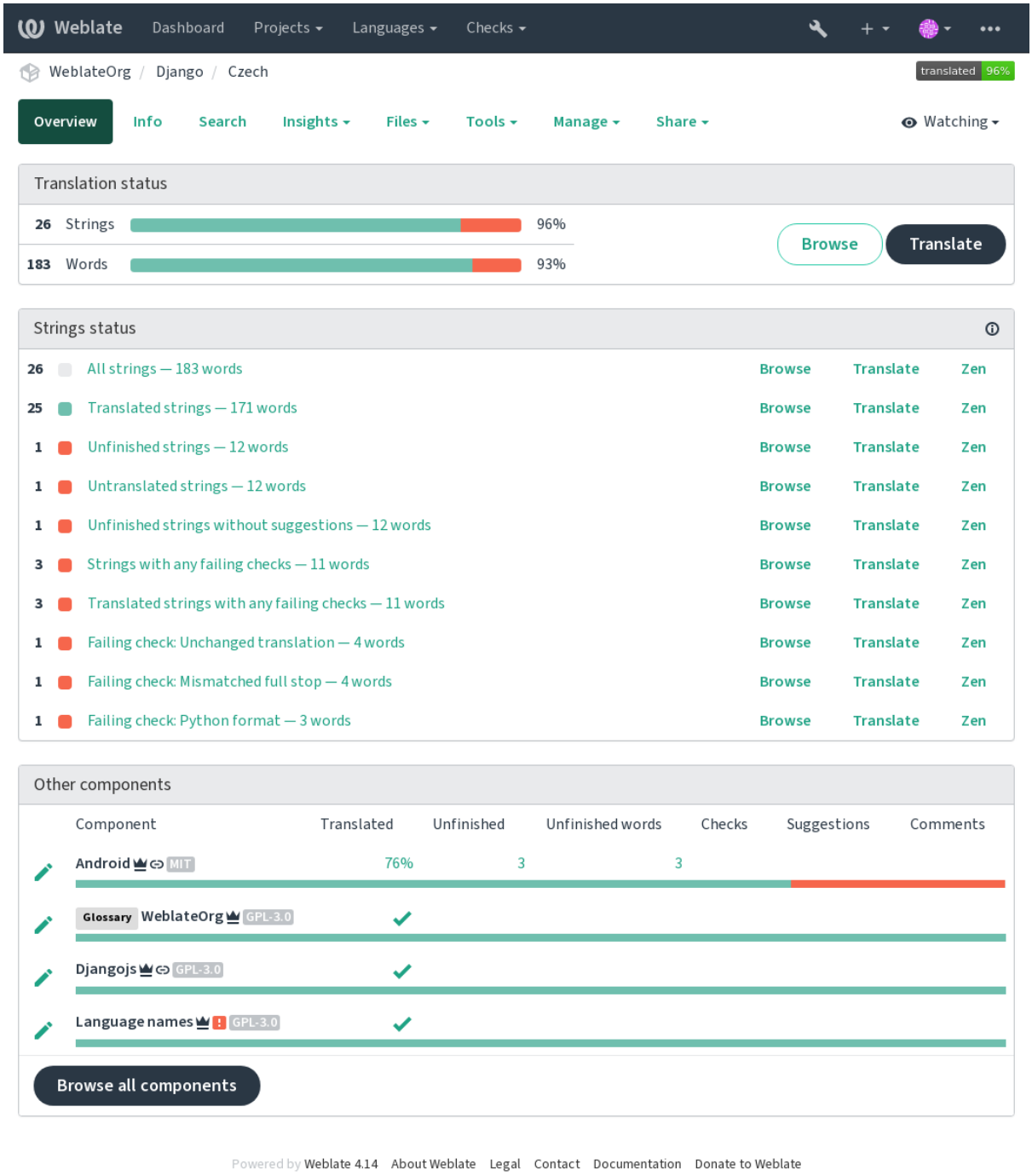
1.3.1 Übersetzungsprojekte

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



1.3.2 Translation links

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



1.3.3 Vorschläge

Bemerkung: 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 Kommentare

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

Siehe auch:

report-source, *Überprüfungen der Ausgangszeichenketten*, *Quellenüberprüfung aktivieren*

1.3.5 Varianten

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.

Siehe auch:

variants, *Varianten*

1.3.6 Labels

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

Following labels are used by Weblate:

Automatisch übersetzt
: String was translated using *Automatische Übersetzung*.

Quelle muss überprüft werden
: Zeichenkette wurde mittels *Überprüfungen der Ausgangszeichenketten* zur Überprüfung markiert.

Siehe auch:

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.

Auf dieser Seite können verschiedene zusätzliche Informationen angezeigt werden, von denen die meisten aus dem Quellcode des Projekts stammen (wie Kontext, Kommentare oder wo die Nachricht verwendet wird). Übersetzungsfelder für Zweitsprachen, die der Übersetzer in den Einstellungen ausgewählt hat, werden über der Ausgangszeichenkette angezeigt (siehe *Zweitsprachen*).

Unterhalb der Übersetzung finden die Übersetzer Vorschläge von anderen, die sie annehmen (✓), mit Änderungen annehmen (✏) oder löschen (⍰) können.

Plural

Wörter, die aufgrund ihrer numerischen Bezeichnung ihre Form ändern, werden Pluralformen genannt. Jede Sprache hat ihre eigene Definition von Pluralformen. Im Englischen zum Beispiel gibt es eine Form. In der Singular-Definition von z.B. „car“ ist implizit ein Auto gemeint, in der Plural-Definition „cars“ sind zwei oder mehr Autos gemeint (oder das Konzept von Autos als Substantiv). Sprachen wie z. B. Tschechisch oder Arabisch haben mehr Pluralformen und auch ihre Regeln für Pluralformen sind anders.

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

Siehe auch:

Pluralformel

Alternative Übersetzungen

Neu in Version 4.13.

Bemerkung: This is currently only supported with *Multivalue CSV file*.

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

Weblate Dashboard Projects Languages Checks

WeblateOrg / Django / Czech / Translate translated 96%

1 / 1 Custom search '%(count)s word' Zen

Position and priority

Translation

English

Singular
%(count)s word

Plural
%(count)s words

Czech, One NBS ... „ “ , ‘ - –
%(count)s slovo
15/140 · 14

Czech, Few NBS ... „ “ , ‘ - –
%(count)s slova
15/140 · 15

Czech, Other NBS ... „ “ , ‘ - –
%(count)s slov
14/140 · 15

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

Needs editing

Save and continue | Save and stay | Suggest | Skip

Nearby strings 20 | Comments | Automatic suggestions | Other languages 3 | History

New comment

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

Scope

Translation comment, discussions with other translators

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

New comment

You can use Markdown and mention users by @username.

Save

Glossary

English Czech

No related strings found in the glossary.

Add term to glossary

String information

Screenshot context
No screenshot currently associated.
Add screenshot

Explanation
No explanation currently provided.

Labels
No labels currently set.

Flags
python-format

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

String age
6 seconds ago

Source string age
6 seconds ago

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

Powered by Weblate 4.14 About Weblate Legal Contact Documentation Donate to Weblate

Tastaturkürzel

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

The following keyboard shortcuts can be utilized during translation:

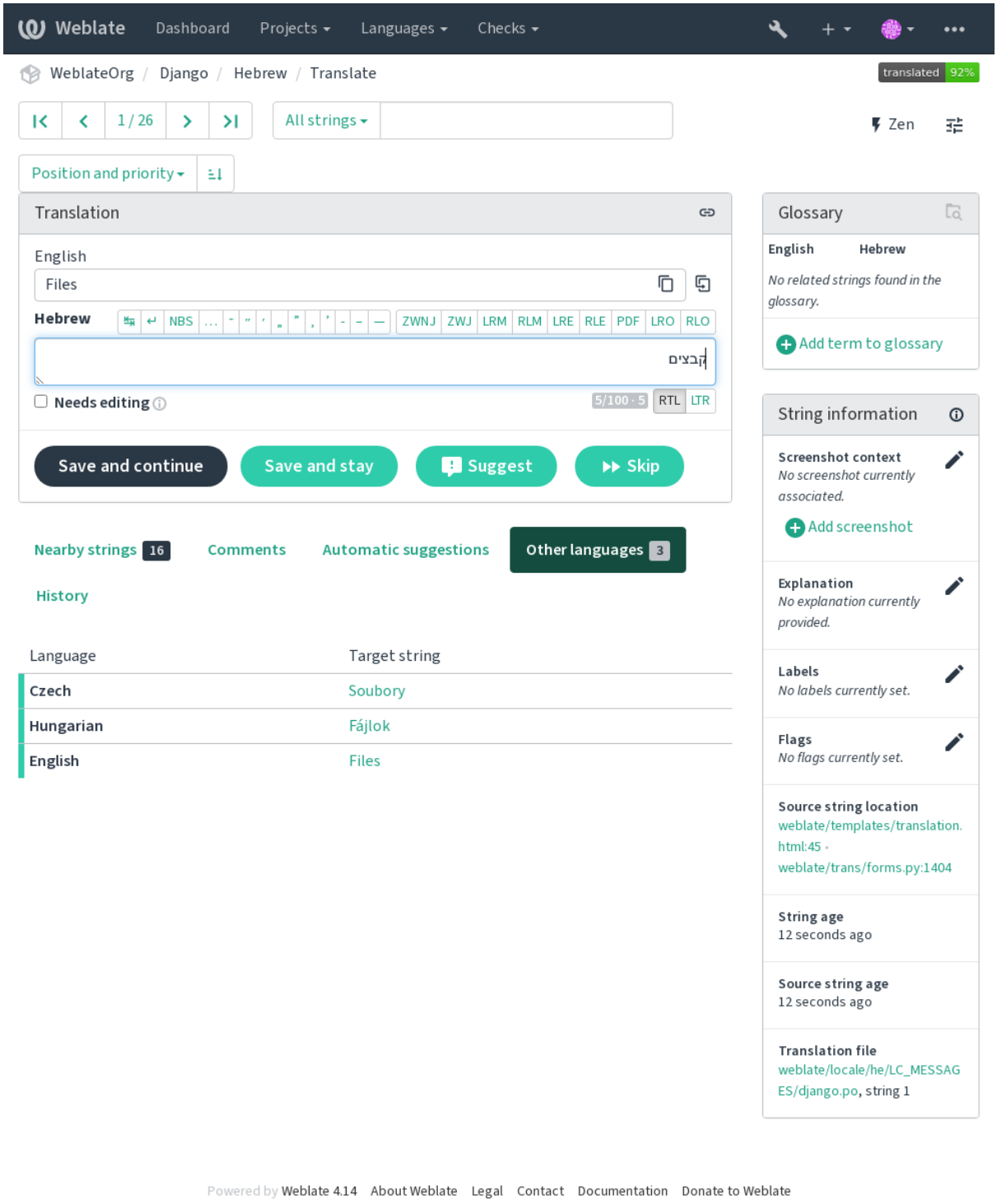
Tastaturkürzel	Beschreibung
`Alt+Home`	Navigate to first translation in current search.
`Alt+End`	Navigate to last translation in current search.
`Alt+PageUp` or `Ctrl+↑` or `Alt+↑` or `Cmd+↑`	Navigate to previous translation in current search.
`Alt+PageDown` or `Ctrl+↓` or `Alt+↓` or `Cmd+↓`	Navigate to next translation in current search.
`Alt+Enter` or `Ctrl+Enter` or `Cmd+Enter`	Submit current form; this is same as pressing *Save and continue* while editing translation.
`Ctrl+Shift+Enter` or `Cmd+Shift+Enter`	Bearbeitungsmarkierung der Übersetzung aufheben und absenden.
`Ctrl+E` or `Cmd+E`	Focus translation editor.
`Ctrl+U` or `Cmd+U`	Focus comment editor.
`Ctrl+M` or `Cmd+M`	Shows *Automatic suggestions* tab, see *Automatische Vorschläge*.
`Ctrl+1` to `Ctrl+9` or `Cmd+1` to `Cmd+9`	Copies placeable of given number from source string.
`Ctrl+M+1` to `9` or `Cmd+M+1` to `9`	Copy the machine translation of given number to current translation.
`Ctrl+I+1` to `9` or `Cmd+I+1` to `9`	Ignorieren Sie ein Element in der Liste der fehlgeschlagenen Qualitätsprüfungen.
`Ctrl+J` or `Cmd+J`	Zeigt die Registerkarte *Benachbarte Zeichenketten* an.
`Ctrl+S` or `Cmd+S`	Focus search field.
`Ctrl+O` or `Cmd+O`	Ausgangszeichenkette kopieren.
`Ctrl+Y` or `Cmd+Y`	Toggle the *Needs editing* checkbox.

Visual keyboard

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

The shown symbols factor into three categories:

- User configured *Sonderzeichen* defined in the *Benutzerprofil*
- Per-language characters provided by Weblate (e.g. quotes or RTL specific characters)
- Characters configured using `SPECIAL_CHARS`


Weblate
Dashboard
Projects
Languages
Checks
WeblateOrg / Django / Hebrew / Translate
translated 92%
1 / 26
All strings
Zen
Position and priority
Translation
English
Files
Hebrew
NBS ... ZWNJ ZWJ LRM RLM LRE RLE PDF LRO RLO
קבצים
Needs editing
5/100 · 5
RTL LTR
Save and continue
Save and stay
Suggest
Skip
Nearby strings 16
Comments
Automatic suggestions
Other languages 3
History
Language
Target string
Czech
Soubory
Hungarian
Fájlok
English
Files
Glossary
English
Hebrew
No related strings found in the glossary.
Add term to glossary
String information
Screenshot context
No screenshot currently associated.
Add screenshot
Explanation
No explanation currently provided.
Labels
No labels currently set.
Flags
No flags currently set.
Source string location
weblate/templates/translation.html:45 · weblate/trans/forms.py:1404
String age
12 seconds ago
Source string age
12 seconds ago
Translation file
weblate/locale/he/LC_MESSAGES/django.po, string 1
Powered by Weblate 4.14
About Weblate
Legal
Contact
Documentation
Donate to Weblate

Translation context

This contextual description provides related info about the current string.

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

Bildschirmfotos
Bildschirmfotos können auf Weblate hochgeladen werden, um Übersetzer besser darüber zu informieren, wo und wie die Zeichenkette verwendet wird, siehe Bildschirmfotos.

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

Andere Vorkommnisse
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 *Inkonsistent*). You can choose which one to use.

Übersetzungsspeicher
Look at similar strings translated in past, see *Übersetzungsspeicher*.

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

Letzte Änderungen
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.

Translation history

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

Translated string length

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

- The default limitation for translation is ten times longer than the source string. This can be turned off by `LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH`. In case you are hitting this, it might be also caused by a monolingual translation erroneously set up as bilingual one, making Weblate mistaking the translation key for the actual source string. See *Bilingual and monolingual formats* for more info.
- Maximal length in characters defined by translation file or flag, see *Maximale Länge der Übersetzung*.
- Maximal rendered size in pixels defined by flags, see *Maximaler Umfang der Übersetzung*.

1.3.8 Automatische Vorschläge

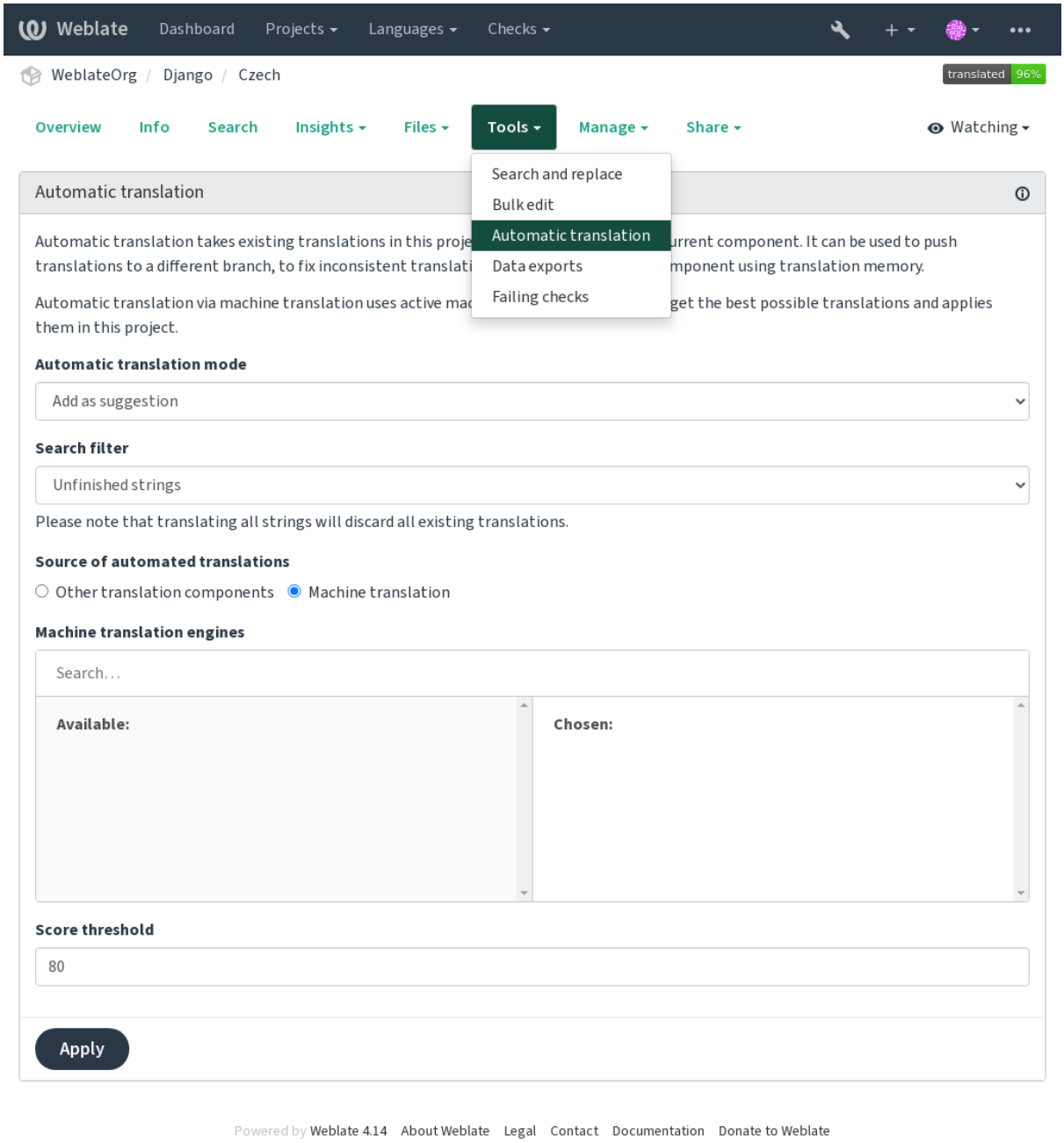
Based on configuration and your translated language, Weblate provides suggestions from several machine translation tools and *Übersetzungsspeicher*. All machine translations are available in a single tab of each translation page.

Siehe auch:

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

1.3.9 Automatische Übersetzung

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.

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

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

The automatically translated strings are labelled *Automatically translated*.

Siehe auch:

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 Suchen und Ersetzen

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

Hinweis: Machen Sie sich keine Sorgen, dass Sie die Zeichenketten durcheinander bringen. Dies ist ein zweistufiger Prozess, der eine Vorschau der bearbeiteten Zeichenketten anzeigt, bevor die tatsächliche Änderung bestätigt wird.

1.3.12 Massenbearbeitung

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)

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

Siehe auch:

Bulk edit add-on

1.3.13 Matrix View

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

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

1.3.14 Zen Mode

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

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

1.4 Downloading and uploading translations

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

Bemerkung: Available options might be limited by *access control* settings.

1.4.1 Downloading translations

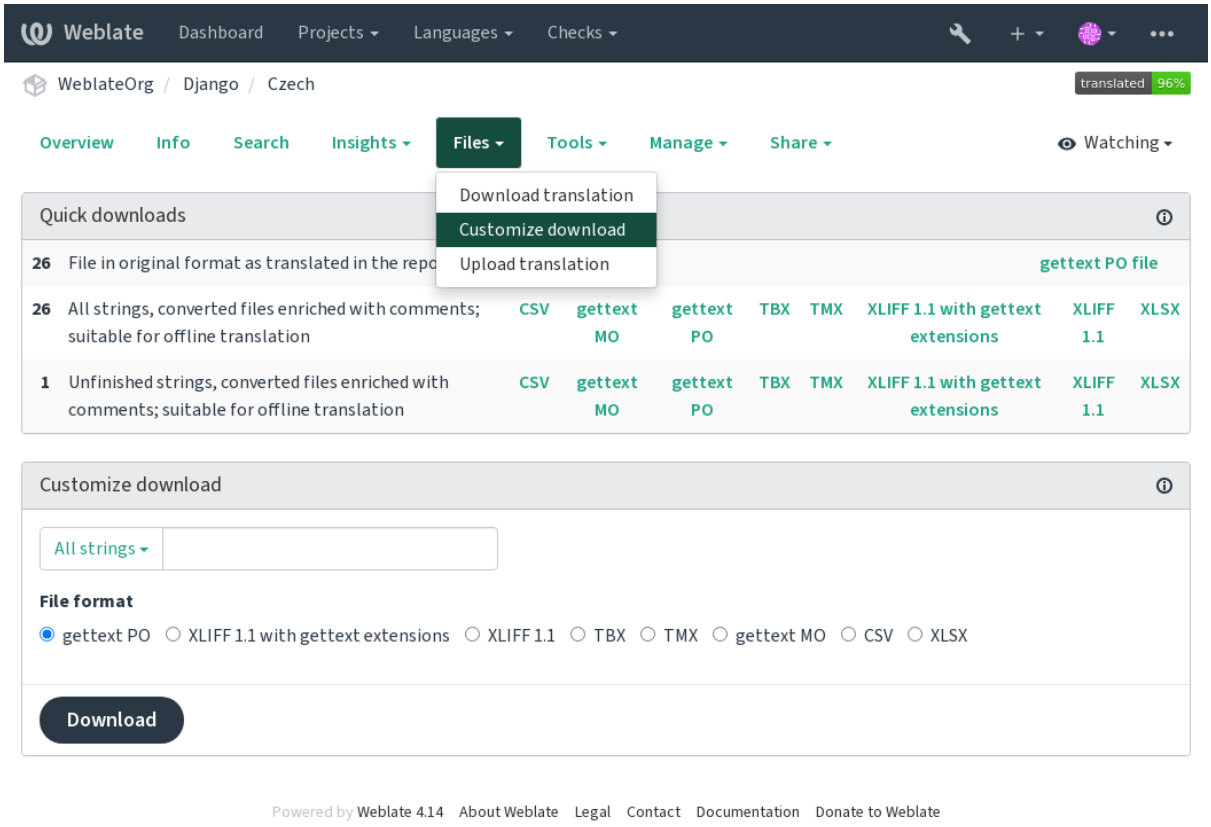
From the project or component dashboard, translatable files can be downloaded in the *Files* menu.

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

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

- gettext PO
- XLIFF mit Gettext-Erweiterungen
- XLIFF 1.1
- TermBase eXchange
- Translation Memory eXchange
- gettext MO (nur verfügbar, wenn die Übersetzung mit gettext PO erfolgt)
- CSV
- Excel Open XML
- JSON (nur für einsprachige Übersetzungen verfügbar)
- Android String Resource (nur für einsprachige Übersetzungen verfügbar)
- iOS-Zeichenketten (nur für einsprachige Übersetzungen verfügbar)

Hinweis: Der in den konvertierten Dateien verfügbare Inhalt unterscheidet sich aufgrund von Dateiformatmerkmalen, eine Übersicht finden Sie in *Translation types capabilities*.

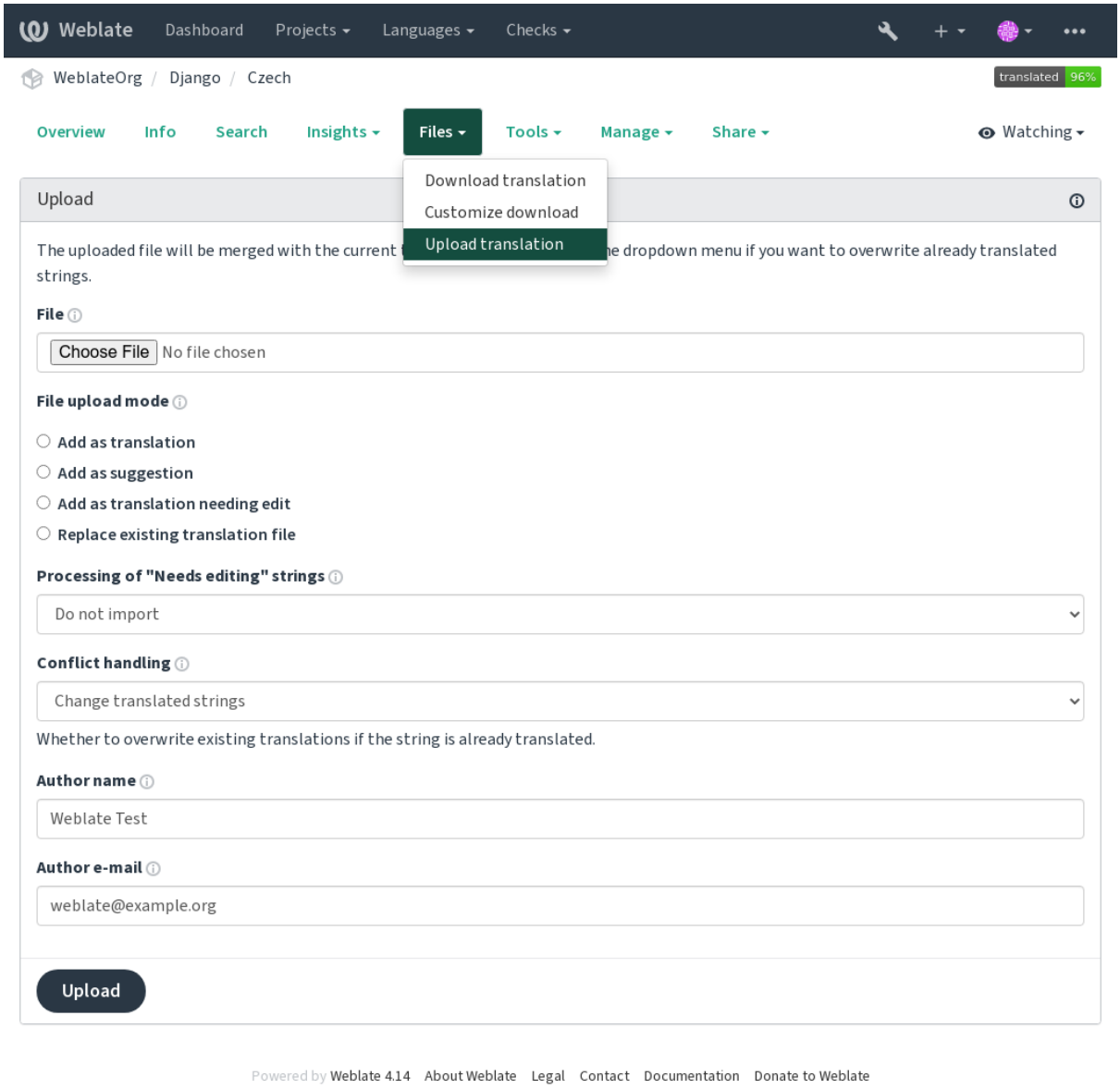


Siehe auch:

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

1.4.2 Uploading translations

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



Supported file formats

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

Siehe auch:

Supported file formats, *Downloading and uploading translations*

Import methods

These are the choices presented when uploading translation files:

Add as translation (`translate`)
Imported strings are added as translations to existing strings. This is the most common usecase, and the default behavior.

Es werden nur Übersetzungen aus der hochgeladenen Datei verwendet und keine zusätzlichen Inhalte.

Add as suggestion (`suggest`)
Imported strings are added as suggestions, do this when you want to have your uploaded strings reviewed.

Es werden nur Übersetzungen aus der hochgeladenen Datei verwendet und keine zusätzlichen Inhalte.

Add as translation needing edit (`fuzzy`)
Imported strings are added as translations needing edit. This can be useful when you want translations to be used, but also reviewed.

Es werden nur Übersetzungen aus der hochgeladenen Datei verwendet und keine zusätzlichen Inhalte.

Replace existing translation file (`replace`)
Existing file is replaced with new content. This can lead to loss of existing translations, use with caution.

Update source strings (`source`)
Updates source strings in bilingual translation file. This is similar to what *PO-Dateien auf POT aktualisieren (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 *Zeichenketten verwalten* turned on.

Es werden nur Ausgangszeichenkette, Übersetzung und Schlüssel (Kontext) aus der hochgeladenen Datei verwendet.

Siehe auch:

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

Conflicts handling

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

Strings needing edit

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

Overriding authorship

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

1.5 Glossar

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

Ein Glossar für jede Sprache kann separat verwaltet werden, aber sie werden zusammen als eine einzige Komponente gespeichert, was Projektadministratoren und mehrsprachigen Übersetzern hilft, eine gewisse sprachübergreifende Konsistenz zu wahren. Begriffe aus dem Glossar, die Wörter aus dem aktuell übersetzten Text enthalten, werden in der Seitenleiste des Übersetzungseditors angezeigt.

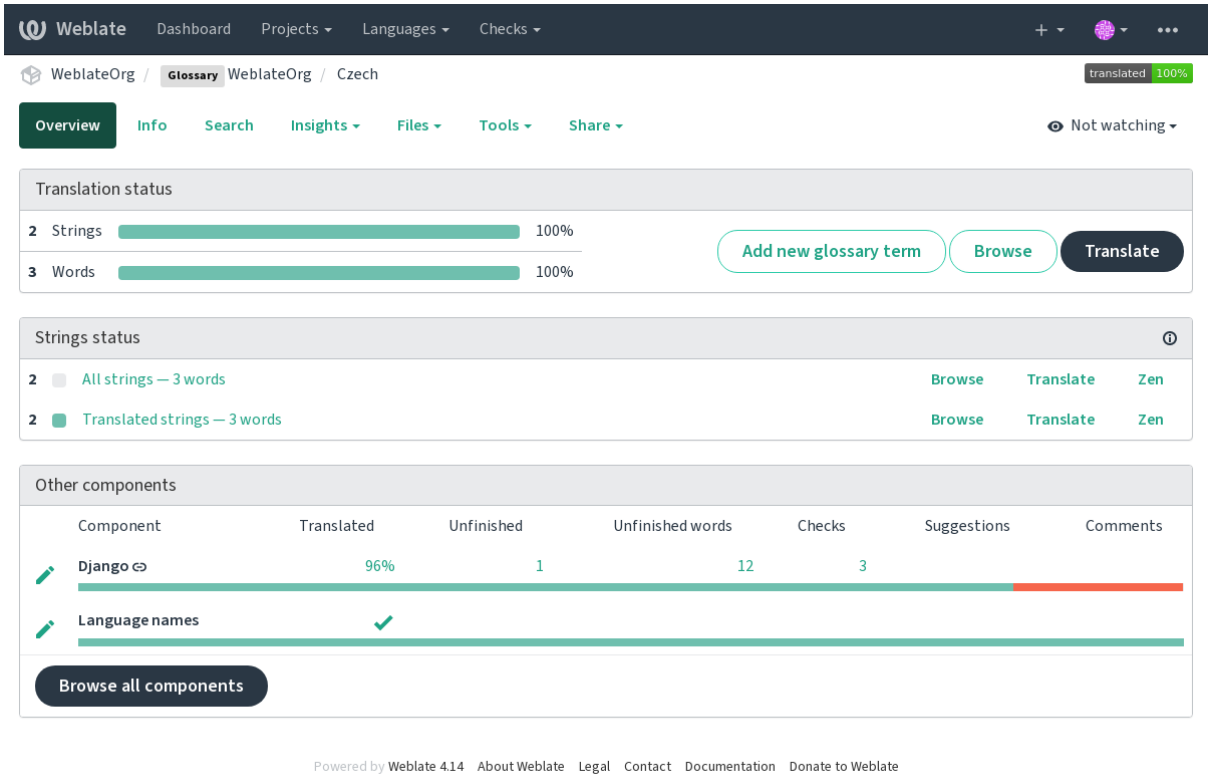
1.5.1 Managing glossaries

Geändert in Version 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.

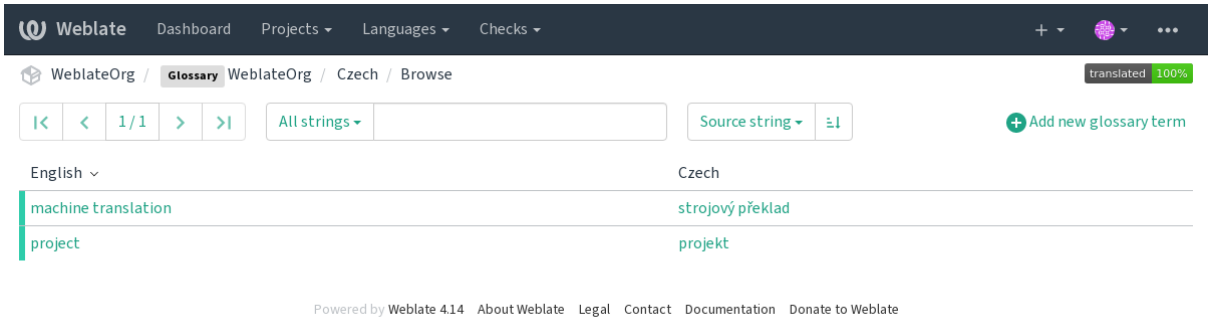
Use any component as a glossary by turning on *Verwendung als Glossar*. You can create multiple glossaries for one project.

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 *In Projekten teilen* from the respective glossary component.

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



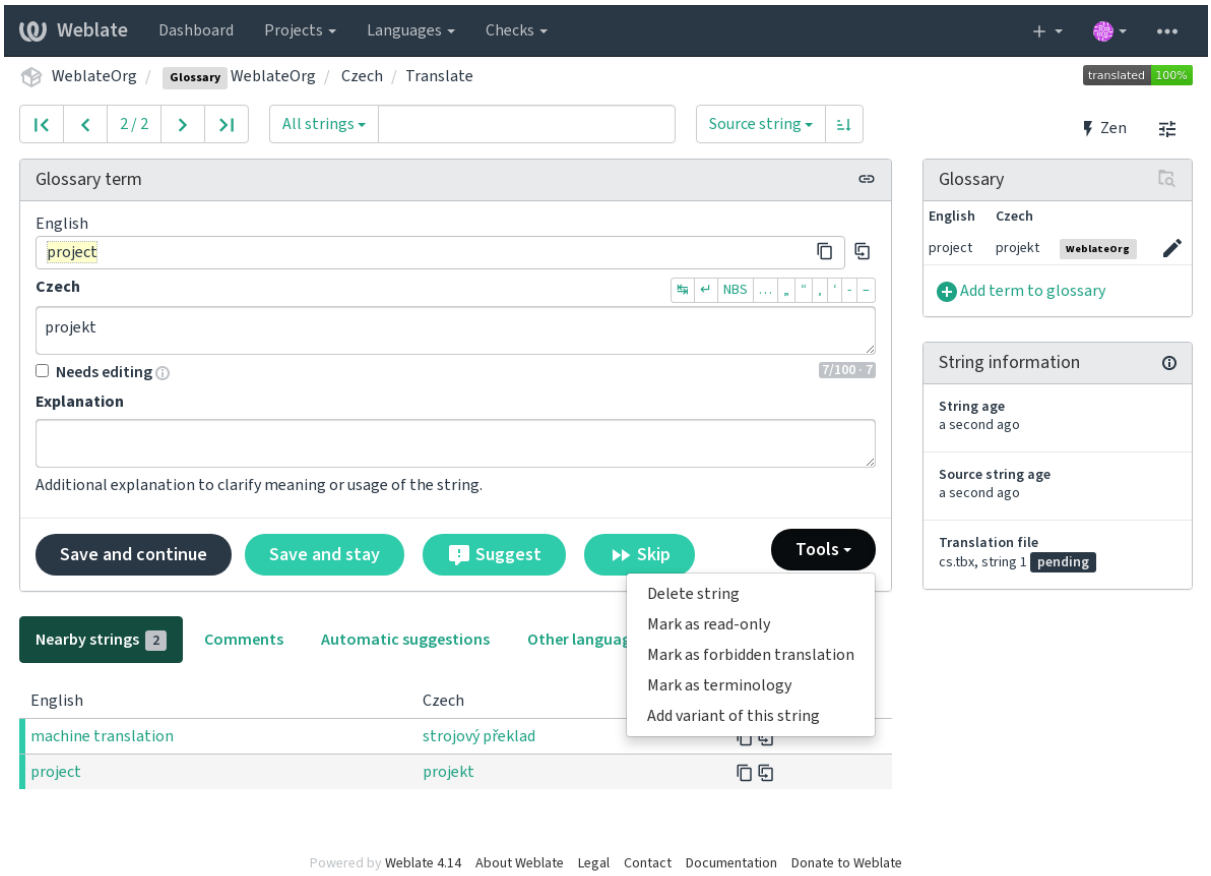
Sie können alle Begriffe des Glossars durchsuchen:



oder sie als beliebige Übersetzungen bearbeiten.

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.



Unübersetzbare Begriffe

Neu in Version 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.

Siehe auch:

Customizing behavior using flags

Forbidden translations

Neu in Version 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.

Siehe auch:

Customizing behavior using flags

Terminologie

Neu in Version 4.5.

Das Markieren bestimmter Glossarbegriffe als `Terminologie` durch Massenbearbeitung, Eingabe der Markierung oder durch Verwendung von *Tools ↓ Als Terminologie markieren* fügt Einträge für diese Begriffe in allen Sprachen des Glossars hinzu. Verwenden Sie dies für wichtige Begriffe, die gut durchdacht sein sollten und in allen Sprachen eine einheitliche Bedeutung behalten.

Siehe auch:

Customizing behavior using flags

Varianten

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

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

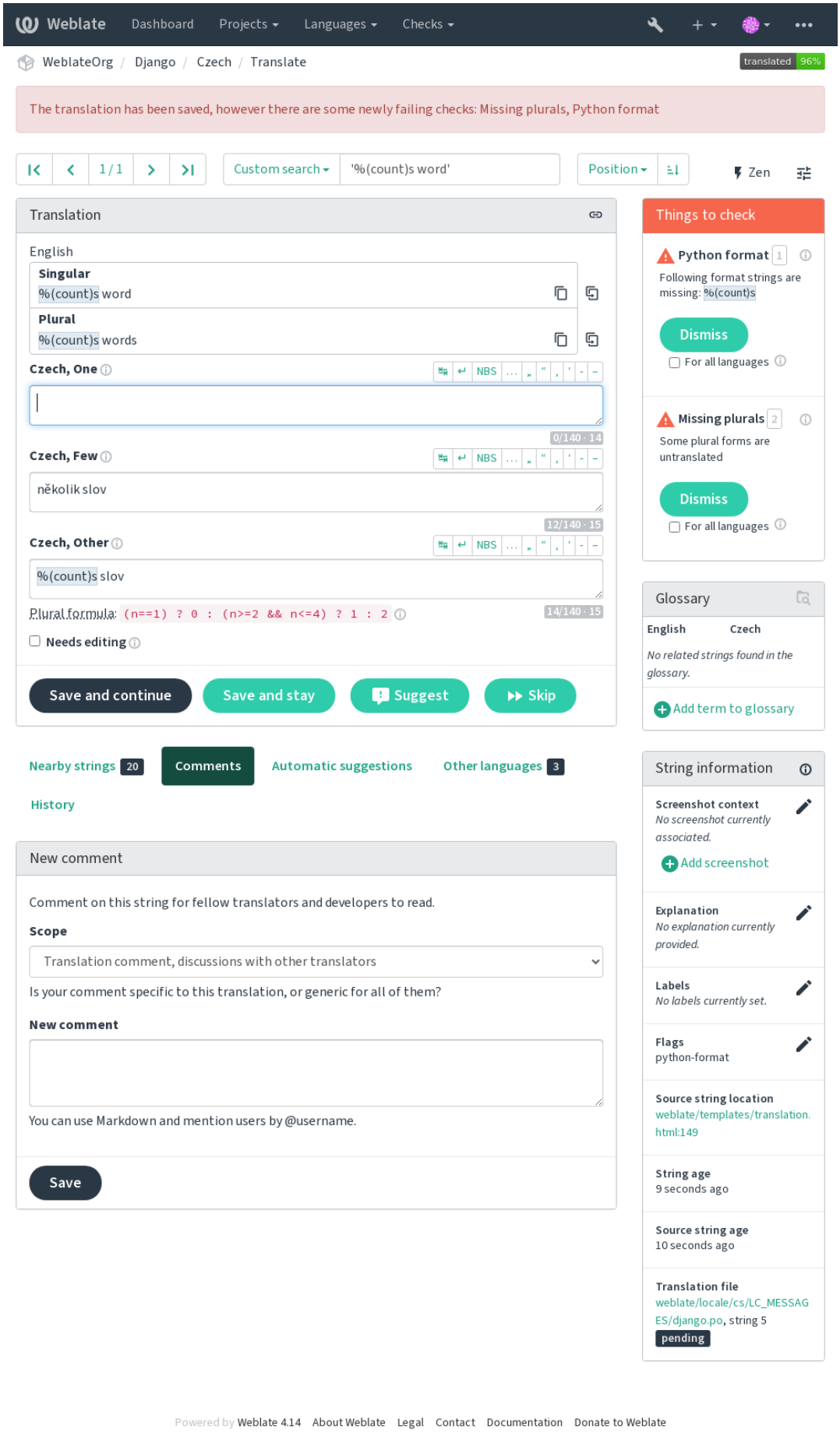
Siehe auch:

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:


Weblate
Dashboard
Projects
Languages
Checks
WeblateOrg / Django / Czech / Translate
translated 96%
The translation has been saved, however there are some newly failing checks: Missing plurals, Python format
1 / 1
Custom search
'%(count)s word'
Position
Zen
Translation
English
Singular
%(count)s word
Plural
%(count)s words
Czech, One
0/140 · 14
Czech, Few
několik slov
12/140 · 15
Czech, Other
%(count)s slov
14/140 · 15
Plural formula: (n==1) ? 0 : (n>=2 && n<=4) ? 1 : 2
Needs editing
Save and continue
Save and stay
Suggest
Skip
Nearby strings 20
Comments
Automatic suggestions
Other languages 3
History
New comment
Comment on this string for fellow translators and developers to read.
Scope
Translation comment, discussions with other translators
Is your comment specific to this translation, or generic for all of them?
New comment
You can use Markdown and mention users by @username.
Save
Things to check
Python format 1
Following format strings are missing: %(count)s
Dismiss
For all languages
Missing plurals 2
Some plural forms are untranslated
Dismiss
For all languages
Glossary
English
Czech
No related strings found in the glossary.
Add term to glossary
String information
Screenshot context
No screenshot currently associated.
Add screenshot
Explanation
No explanation currently provided.
Labels
No labels currently set.
Flags
python-format
Source string location
weblate/templates/translation.html:149
String age
9 seconds ago
Source string age
10 seconds ago
Translation file
weblate/locale/cs/LC_MESSAGES/django.po, string 5
pending
Powered by Weblate 4.14
About Weblate
Legal
Contact
Documentation
Donate to Weblate

1.6.1 Automatische Korrekturen

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

Siehe auch:

`AUTOFIX_LIST`

1.6.2 Qualitätsprüfungen

Weblate wendet eine Vielzahl von Qualitätsprüfungen für Zeichenketten an. Der folgende Abschnitt beschreibt sie alle im Detail. Es gibt auch sprachspezifische Prüfungen. Bitte melden Sie einen Fehler, wenn etwas falsch gemeldet wird.

Siehe auch:

`CHECK_LIST`, *Customizing behavior using flags*

1.6.3 Übersetzungsprüfungen

Wird bei jeder Übersetzungsänderung ausgeführt und hilft den Übersetzern, die Qualität der Übersetzungen aufrechtzuerhalten.

BBCode-Markup

Zusammenfassung
BBCode in der Übersetzung passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.markup.BBCodeCheck`

Check identifier
`bbcode`

Zu ignorierende Markierung
`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.

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

Aufeinanderfolgende doppelte Wörter

Neu in Version 4.1.

Zusammenfassung
Text enthält zweimal hintereinander dasselbe Wort:

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.duplicate.DuplicateCheck`

Check identifier
`duplicate`

Zu ignorierende Markierung
`ignore-duplicate`

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

Hinweis: This check includes language specific rules to avoid false positives. In case it triggers falsely in your case, let us know. See *Melden von Problemen in Weblate*.

Folgt nicht dem Glossar

Neu in Version 4.5.

Zusammenfassung
Die Übersetzung folgt nicht den definierten Begriffen im Glossar.

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.glossary.GlossaryCheck`

Check identifier
`check_glossary`

Flag to enable
`check-glossary`

Zu ignorierende Markierung
`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.

Siehe auch:

Glossar, *Customizing behavior using flags*, *Übersetzungsmarkierungen*

Doppeltes Leerzeichen

Zusammenfassung
Übersetzung enthält doppelte Leerzeichen

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.DoubleSpaceCheck`

Check identifier
`double_space`

Zu ignorierende Markierung
`ignore-double-space`

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

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

Formatted strings

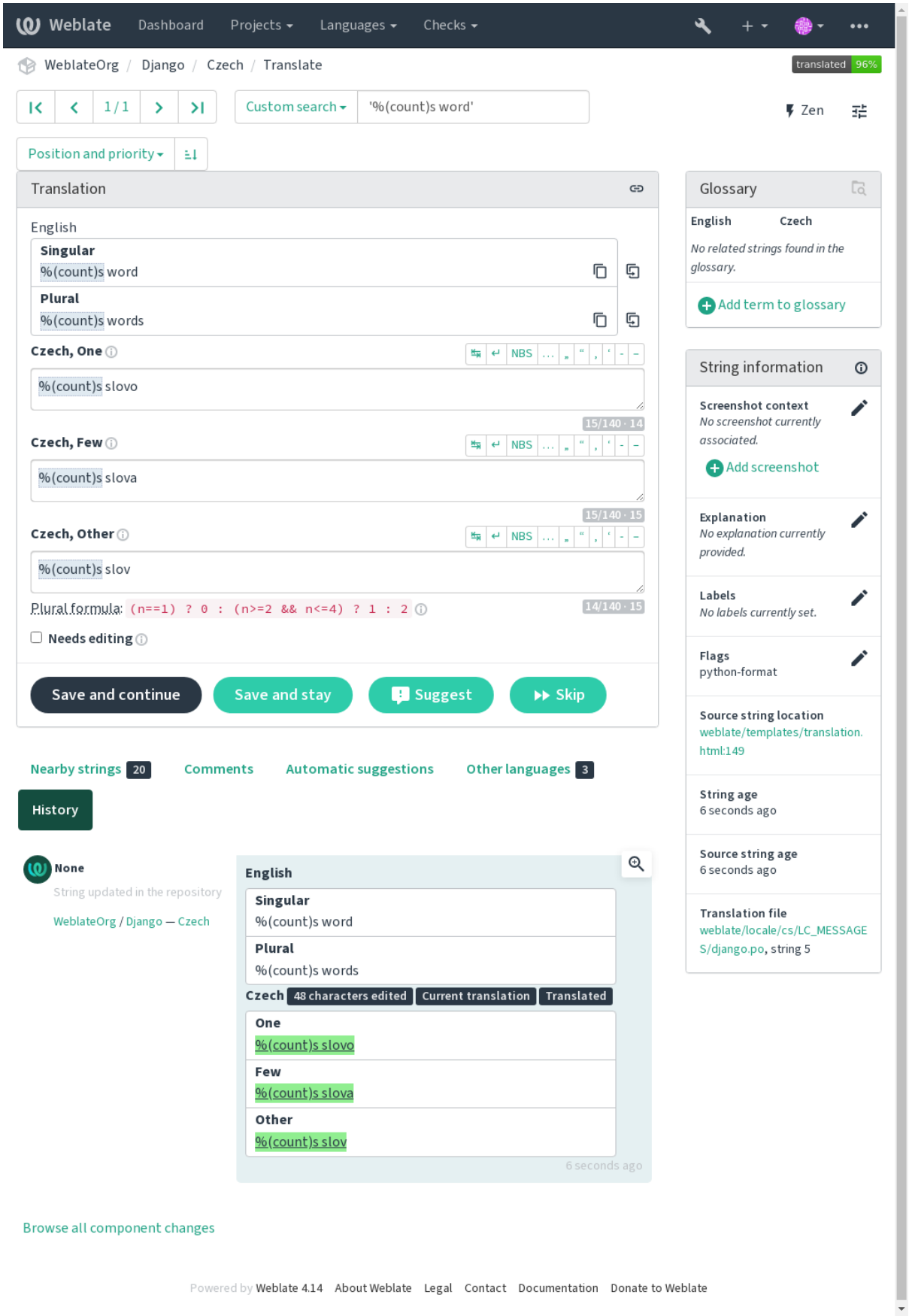
Checks that formatting in strings are replicated between both source and translation. Omitting format strings in translation usually causes severe problems, so the formatting in strings should usually match the source.

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

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

Hinweis: In case specific format check is not available in Weblate, you can use generic *Platzhalter*.

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


Weblate
Dashboard
Projects
Languages
Checks
WeblateOrg / Django / Czech / Translate
translated 96%
1 / 1
Custom search
'%(count)s word'
Zen
Position and priority
Translation
English
Singular
%(count)s word
Plural
%(count)s words
Czech, One
%(count)s slovo
15/140 · 14
Czech, Few
%(count)s slova
15/140 · 15
Czech, Other
%(count)s slov
14/140 · 15
Plural formula: (n==1) ? 0 : (n>=2 && n<=4) ? 1 : 2
Needs editing
Save and continue
Save and stay
Suggest
Skip
Glossary
English
Czech
No related strings found in the glossary.
Add term to glossary
String information
Screenshot context
No screenshot currently associated.
Add screenshot
Explanation
No explanation currently provided.
Labels
No labels currently set.
Flags
python-format
Source string location
weblate/templates/translation.html:149
String age
6 seconds ago
Source string age
6 seconds ago
Translation file
weblate/locale/cs/LC_MESSAGES/django.po, string 5
Nearby strings 20
Comments
Automatic suggestions
Other languages 3
History
None
String updated in the repository
WeblateOrg / Django — Czech
English
Singular
%(count)s word
Plural
%(count)s words
Czech
48 characters edited
Current translation
Translated
One
%(count)s slovo
Few
%(count)s slova
Other
%(count)s slov
6 seconds ago
Browse all component changes
Powered by Weblate 4.14
About Weblate
Legal
Contact
Documentation
Donate to Weblate

AngularJS-Interpolations-Zeichenkette

Zusammenfassung
AngularJS-Interpolations-Format-Zeichenketten passen nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.angularjs.AngularJSInterpolationCheck`

Check identifier
`angularjs_format`

Flag to enable
`angularjs-format`

Zu ignorierende Markierung
`ignore-angularjs-format`

Beispiel für eine benannte Formatzeichenfolge
`Your balance is {{amount}} {{ currency }}`

Siehe auch:

Formatted strings, AngularJS text interpolation

C-Format

Zusammenfassung
C-Format-Zeichenkette passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.format.CFormatCheck`

Flag to enable
`c-format`

Zu ignorierende Markierung
`ignore-c-format`

Einfaches Beispiel für eine formatierte Zeichenkette
`There are %d apples`

Beispiel für eine Zeichenkette im Positionsformat
`Your balance is %1$d %2$s`

Siehe auch:

***Formatted strings*,**
C format strings, C printf format

C#-Format

Zusammenfassung
C#-Format-Zeichenkette passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.format.CSharpFormatCheck`

Check identifier
`c_sharp_format`

Flag to enable
`c-sharp-format`

Zu ignorierende Markierung
`ignore-c-sharp-format`

Beispiel für eine Zeichenkette im Positionsformat
`There are {0} apples`

Siehe auch:

Formatted strings, C# String Format

ECMAScript-Buchstabenvorlagen

Zusammenfassung
ECMAScript-Buchstabenvorlagen passen nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.format.ESTemplateLiteralsCheck`

Check identifier
`es_format`

Flag to enable
`es-format`

Zu ignorierende Markierung
`ignore-es-format`

Beispiel für Interpolation
`There are ${number} apples`

Siehe auch:

Formatted strings, Template literals

i18next Interpolation

Neu in Version 4.0.

Zusammenfassung
Die i18next Interpolation stimmt nicht mit der Quelle überein

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.format.I18NextInterpolationCheck`

Check identifier
`i18next_interpolation`

Flag to enable
`i18next-interpolation`

Zu ignorierende Markierung
`ignore-i18next-interpolation`

Beispiel für Interpolation
`There are {{number}} apples`

Beispiel für Verschachtelung
`There are $t(number) apples`

Siehe auch:

Formatted strings, i18next interpolation

ICU MessageFormat

Neu in Version 4.9.

Zusammenfassung
Syntaxfehler und/oder nicht übereinstimmende Platzhalter in ICU MessageFormat-Zeichenketten.

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.icu.ICUMessageFormatCheck`

Check identifier
`icu_message_format`

Flag to enable
`icu-message-format`

Zu ignorierende Markierung
`ignore-icu-message-format`

Beispiel für Interpolation
`There {number, plural, one {is one apple} other {are # apples}}.`

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

`xml`	Enable support for simple XML tags. By default, XML tags are parsed loosely. Stray < characters are ignored if they are not reasonably part of a tag.
`strict-xml`	Enable support for strict XML tags. All < characters must be escaped if they are not part of a tag.
`-highlight`	Disable highlighting placeholders in the editor.
`-require_other`	Disable requiring sub-messages to have an `other` selector.
`-submessage_selector`	Skip checking that sub-message selectors match the source.
`-types`	Die Prüfung überspringen, ob Platzhaltertypen mit der Ausgangszeichenkette übereinstimmen.
`-extra`	Die Prüfung überspringen, ob Platzhalter vorhanden sind, die in der Ausgangszeichenkette nicht vorhanden waren.
`-missing`	Die Prüfung überspringen, ob keine Platzhalter fehlen, die in der Ausgangszeichenkette vorhanden waren.

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

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

Siehe auch:

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

Java-Format

Zusammenfassung
Java-Format-Zeichenkette passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.format.JavaFormatCheck`

Check identifier
`java_printf_format`

Flag to enable
`java-printf-format`

Zu ignorierende Markierung
`ignore-java-printf-format`

Einfaches Beispiel für eine formatierte Zeichenkette
`There are %d apples`

Beispiel für eine Zeichenkette im Positionsformat
`Your balance is %1$d %2$s`

Geändert in Version 4.14: This used to be toggled by `java-format` flag, it was changed for consistency with GNU gettext.

Siehe auch:

Formatted strings, Java Format Strings

Java-MessageFormat

Zusammenfassung
Java-MessageFormat-Zeichenkette passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.format.JavaMessageFormatCheck`

Check identifier
`java_format`

Markierung zur uncodierten Aktivierung
`java-format`

Markierung zur Aktivierung der automatischen Erkennung
`auto-java-messageformat` aktiviert die Prüfung nur, wenn eine Formatzeichenkette in der Ausgangszeichenkette vorhanden ist

Zu ignorierende Markierung
`ignore-java-format`

Beispiel für eine Zeichenkette im Positionsformat
`There are {0} apples`

Geändert in Version 4.14: This used to be toggled by `java-messageformat` flag, it was changed for consistency with GNU gettext.

Siehe auch:

Formatted strings, Java MessageFormat

JavaScript-Format

Zusammenfassung
JavaScript-Format-Zeichenkette passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.format.JavaScriptFormatCheck`

Check identifier
`javascript_format`

Flag to enable
`javascript-format`

Zu ignorierende Markierung
`ignore-javascript-format`

Einfaches Beispiel für eine formatierte Zeichenkette
`There are %d apples`

Siehe auch:

Formatted strings, JavaScript formatting strings

Lua-Format

Zusammenfassung
Lua-Format-Zeichenkette passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.format.LuaFormatCheck`

Check identifier
`lua_format`

Flag to enable
`lua-format`

Zu ignorierende Markierung
`ignore-lua-format`

Einfaches Beispiel für eine formatierte Zeichenkette
`There are %d apples`

Siehe auch:

Formatted strings, Lua formatting strings

Object Pascal-Format

Zusammenfassung
Object Pascal-Format-Zeichenkette passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.format.ObjectPascalFormatCheck`

Check identifier
`object_pascal_format`

Flag to enable
`object-pascal-format`

Zu ignorierende Markierung
`ignore-object-pascal-format`

Einfaches Beispiel für eine formatierte Zeichenkette
`There are %d apples`

Siehe auch:

Formatted strings, Object Pascal formatting strings, Free Pascal formatting strings Delphi formatting strings

Platzhalter in Prozent

Neu in Version 4.0.

Zusammenfassung
Die prozentualen Platzhalter stimmen nicht mit der Quelle überein

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.format.PercentPlaceholdersCheck`

Check identifier
`percent_placeholders`

Flag to enable
`percent-placeholders`

Zu ignorierende Markierung
`ignore-percent-placeholders`

Einfaches Beispiel für eine formatierte Zeichenkette
`There are %number% apples`

Siehe auch:

Formatted strings,

Perl-Format

Zusammenfassung
Perl-Format-Zeichenkette passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.format.PerlFormatCheck`

Check identifier
`perl_format`

Flag to enable
`perl-format`

Zu ignorierende Markierung
`ignore-perl-format`

Einfaches Beispiel für eine formatierte Zeichenkette
`There are %d apples`

Beispiel für eine Zeichenkette im Positionsformat
`Your balance is %1$d %2$s`

Siehe auch:

Formatted strings, Perl sprintf, Perl Format Strings

PHP-Format

Zusammenfassung
PHP-Format-Zeichenkette passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.format.PHPFormatCheck`

Check identifier
`php_format`

Flag to enable
`php-format`

Zu ignorierende Markierung
`ignore-php-format`

Einfaches Beispiel für eine formatierte Zeichenkette
`There are %d apples`

Beispiel für eine Zeichenkette im Positionsformat
`Your balance is %1$d %2$s`

Siehe auch:

Formatted strings, PHP sprintf documentation, PHP Format Strings

Python-Brace-Format

Zusammenfassung
Python-Format-Zeichenkette (geschweifte Klammern) passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.format.PythonBraceFormatCheck`

Check identifier
`python_brace_format`

Flag to enable
`python-brace-format`

Zu ignorierende Markierung
`ignore-python-brace-format`

Simple format string
`There are {} apples`

Beispiel für eine benannte Formatzeichenfolge
`Your balance is {amount} {currency}`

Siehe auch:

Formatted strings, Python brace format, Python Format Strings

Python-Format

Zusammenfassung
Python-Format-Zeichenkette passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.format.PythonFormatCheck`

Check identifier
`python_format`

Flag to enable
`python-format`

Zu ignorierende Markierung
`ignore-python-format`

Simple format string
`There are %d apples`

Beispiel für eine benannte Formatzeichenfolge
`Ihr Saldo beträgt %(amount)d %(currency)s`

Siehe auch:

Formatted strings, Python string formatting, Python Format Strings

Qt-Format

Zusammenfassung
Qt-Format-Zeichenkette passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.qt.QtFormatCheck`

Check identifier
`qt_format`

Flag to enable
`qt-format`

Zu ignorierende Markierung
`ignore-qt-format`

Beispiel für eine Zeichenkette im Positionsformat
`There are %1 apples`

Siehe auch:

Formatted strings, Qt QString::arg()

Qt-Plural-Format

Zusammenfassung
Qt-Plural-Format-Zeichenkette passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.qt.QtPluralCheck`

Check identifier
`qt_plural_format`

Flag to enable
`qt-plural-format`

Zu ignorierende Markierung
`ignore-qt-plural-format`

Beispiel für eine Zeichenkette im Pluralformat
`There are %Ln apple(s)`

Siehe auch:

Formatted strings, Qt i18n guide

Ruby-Format

Zusammenfassung
Ruby-Format-Zeichenkette passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.ruby.RubyFormatCheck`

Check identifier
`ruby_format`

Flag to enable
`ruby-format`

Zu ignorierende Markierung
`ignore-ruby-format`

Einfaches Beispiel für eine formatierte Zeichenkette
`There are %d apples`

Beispiel für eine Zeichenkette im Positionsformat
`Your balance is %1$f %2$s`

Beispiel für eine benannte Formatzeichenfolge
`Your balance is %+.2<amount>f %<currency>s`

Named template string
`Your balance is %{amount} %{currency}`

Siehe auch:

Formatted strings, Ruby Kernel#sprintf

Scheme-Format

Zusammenfassung
Scheme-Formatzeichenkette stimmt nicht mit Quelle überein

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.format.SchemeFormatCheck`

Check identifier
`scheme_format`

Flag to enable
`scheme-format`

Zu ignorierende Markierung
`ignore-scheme-format`

Einfaches Beispiel für eine formatierte Zeichenkette
`There are ~d apples`

Siehe auch:

Formatted strings, Srfi 28, Chicken Scheme format, Guile Scheme formatted output

Vue I18n-Formatierung

Zusammenfassung
Die Vue I18n-Formatierung passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.format.VueFormattingCheck`

Check identifier
`vue_format`

Flag to enable
`vue-format`

Zu ignorierende Markierung
`ignore-vue-format`

Named formatting
`There are {count} apples`

Rails i18n formatting
`There are %{count} apples`

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

Siehe auch:

Formatted strings, Vue I18n Formatting, Vue I18n Linked locale messages

Ist übersetzt worden

Zusammenfassung
Diese Zeichenkette wurde in der Vergangenheit übersetzt

Zielgruppe
Alle Zeichenketten

Klasse prüfen
`weblate.checks.consistency.TranslatedCheck`

Check identifier
`translated`

Zu ignorierende Markierung
`ignore-translated`

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

Inkonsistent

Zusammenfassung
Diese Zeichenkette hat in diesem Projekt mehr als eine Übersetzung oder ist in einigen Komponenten nicht übersetzt.

Zielgruppe
Alle Zeichenketten

Klasse prüfen
`weblate.checks.consistency.ConsistencyCheck`

Check identifier
`inconsistent`

Zu ignorierende Markierung
`ignore-inconsistent`

Weblate prüft Übersetzungen derselben Zeichenkette für alle Übersetzungen innerhalb eines Projekts, um Ihnen zu helfen, konsistente Übersetzungen zu erhalten.

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

This check applies to all components in a project that have *Verbreitung von Übersetzungen erlauben* turned on.

Hinweis: Aus Leistungsgründen findet die Prüfung möglicherweise nicht alle Inkonsistenzen, so dass die Anzahl der Übereinstimmungen begrenzt ist.

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

You can use *Automatische Übersetzung* add-on to automate translating of newly added strings which are already translated in another component.

Siehe auch:

Keeping translations same across components

Kashida-Buchstabe verwendet

Neu in Version 3.5.

Zusammenfassung
Dekorative Kashida-Schriftzüge sollten nicht verwendet werden

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.KashidaCheck`

Check identifier
`kashida`

Zu ignorierende Markierung
`ignore-kashida`

Die dekorativen Kashida-Buchstaben sollten in der Übersetzung nicht verwendet werden. Diese sind auch als Tatweel bekannt.

Siehe auch:

Kashida auf Wikipedia

Markdown-Links

Neu in Version 3.5.

Zusammenfassung
Markdown-Links in der Übersetzung passen nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.markup.MarkdownLinkCheck`

Check identifier
`md-link`

Flag to enable
`md-text`

Zu ignorierende Markierung
`ignore-md-link`

Markdown links do not match source.

Siehe auch:

Markdown links

Markdown-Referenzen

Neu in Version 3.5.

Zusammenfassung
Markdown-Link-Referenzen passen nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.markup.MarkdownRefLinkCheck`

Check identifier
`md-reflink`

Flag to enable
`md-text`

Zu ignorierende Markierung
`ignore-md-reflink`

Markdown link references do not match source.

Siehe auch:

Markdown links

Markdown-Syntax

Neu in Version 3.5.

Zusammenfassung
Markdown-Syntax passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.markup.MarkdownSyntaxCheck`

Check identifier
`md-syntax`

Flag to enable
`md-text`

Zu ignorierende Markierung
`ignore-md-syntax`

Markdown-Syntax passt nicht zur Ausgangszeichenkette

Siehe auch:

Markdown span elements

Maximale Länge der Übersetzung

Zusammenfassung
Die Übersetzung sollte die angegebene Länge nicht überschreiten

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.MaxLengthCheck`

Check identifier
`max-length`

Flag to enable
`max-length`

Zu ignorierende Markierung
`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`.

Hinweis: This check looks at number of chars, what might not be the best metric when using proportional fonts to render the text. The *Maximaler Umfang der Übersetzung* check does check actual rendering of the text.

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

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

Maximaler Umfang der Übersetzung

Zusammenfassung
Der übersetzte, gerenderte Text sollte die vorgegebene Größe nicht überschreiten

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.render.MaxSizeCheck`

Check identifier
`max-size`

Flag to enable
`max-size`

Zu ignorierende Markierung
`ignore-max-size`

Neu in Version 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:

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

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

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

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

Siehe auch:

Managing fonts, *Customizing behavior using flags*, *Maximale Länge der Übersetzung*

Mismatched \n

Zusammenfassung
Die Menge an \n in der Übersetzung passt nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.EscapedNewlineCountingCheck`

Check identifier
`escaped_newline`

Zu ignorierende Markierung
`ignore-escaped-newline`

Normalerweise sind maskierte Zeilenumbrüche wichtig für die Formatierung der Programmausgabe. Die Prüfung schlägt fehl, wenn die Anzahl der `\n`-Literale in der Übersetzung nicht mit der Quelle übereinstimmt.

Nicht übereinstimmender Doppelpunkt

Zusammenfassung
Ausgangs- und übersetzte Zeichenkette enden nicht beide mit einem Doppelpunkt

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.EndColonCheck`

Check identifier
`end_colon`

Zu ignorierende Markierung
`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).

Siehe auch:

Doppelpunkt auf Wikipedia

Nicht übereinstimmende Auslassungspunkte

Zusammenfassung
Ausgangs- und übersetzte Zeichenkette enden nicht beide mit Auslassungspunkten (…)

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.EndEllipsisCheck`

Check identifier
`end_ellipsis`

Zu ignorierende Markierung
`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.

Siehe auch:

Auslassungspunkte auf Wikipedia

Nicht übereinstimmendes Ausrufezeichen

Zusammenfassung
Ausgangs- und übersetzte Zeichenkette enden nicht beide mit einem Ausrufezeichen

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.EndExclamationCheck`

Check identifier
`end_exclamation`

Zu ignorierende Markierung
`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).

Siehe auch:

Ausrufezeichen auf Wikipedia

Nicht übereinstimmender Punkt

Zusammenfassung
Ausgangs- und übersetzte Zeichenkette enden nicht beide mit einem Punkt

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.EndStopCheck`

Check identifier
`end_stop`

Zu ignorierende Markierung
`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).

Siehe auch:

Punkt auf Wikipedia

Nicht übereinstimmendes Fragezeichen

Zusammenfassung
Ausgangs- und übersetzte Zeichenkette enden nicht beide mit einem Fragezeichen

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.EndQuestionCheck`

Check identifier
`end_question`

Zu ignorierende Markierung
`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).

Siehe auch:

Fragezeichen auf Wikipedia

Nicht übereinstimmendes Semikolon

Zusammenfassung
Ausgangs- und übersetzte Zeichenkette enden nicht beide mit einem Semikolon

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.EndSemicolonCheck`

Check identifier
`end_semicolon`

Zu ignorierende Markierung
`ignore-end-semicolon`

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

Siehe auch:

Semikolon auf Wikipedia

Nicht übereinstimmende Zeilenumbrüche

Zusammenfassung
Anzahl der neuen Zeilen in der Übersetzung stimmt nicht mit der Quelle überein

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.NewLineCountCheck`

Check identifier
`newline-count`

Zu ignorierende Markierung
`ignore-newline-count`

Normalerweise sind Zeilenumbrüche wichtig für die Formatierung der Programmausgabe. Check schlägt fehl, wenn die Anzahl der `\n`-Literale in der Übersetzung nicht mit der Quelle übereinstimmt.

Fehlende Pluralformen

Zusammenfassung
Einige Pluralformen sind nicht übersetzt

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.consistency.PluralsCheck`

Check identifier
`plurals`

Zu ignorierende Markierung
`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.

Platzhalter

Neu in Version 3.9.

Zusammenfassung
Der Übersetzung fehlen einige Platzhalter

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.placeholders.PlaceholderCheck`

Check identifier
`placeholders`

Flag to enable
`placeholders`

Zu ignorierende Markierung
`ignore-placeholders`

Geändert in Version 4.3: Sie können einen regulären Ausdruck als Platzhalter verwenden.

Geändert in Version 4.13: With the `case-insensitive` flag, the placeholders are not case-sensitive.

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

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

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

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

You can also have case insensitive placeholders:

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

Siehe auch:

Customizing behavior using flags

Satzzeichenabstand

Neu in Version 3.9.

Zusammenfassung
Es fehlt ein untrennbares Leerzeichen vor doppelten Satzzeichen

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.PunctuationSpacingCheck`

Check identifier
`punctuation_spacing`

Zu ignorierende Markierung
`ignore-punctuation-spacing`

Prüft, ob vor doppelten Interpunktionszeichen (Ausrufezeichen, Fragezeichen, Semikolon und Doppelpunkt) ein nicht abbrechbares Leerzeichen steht. Diese Regel wird nur in einigen ausgewählten Sprachen wie Französisch oder Bretonisch verwendet, wo das Leerzeichen vor doppelten Satzzeichen eine typografische Regel ist.

Siehe auch:

Französische und englische Abstände auf Wikipedia

Regulärer Ausdruck

Neu in Version 3.9.

Zusammenfassung
Übersetzung stimmt nicht mit regulärem Ausdruck überein

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.placeholders.RegexCheck`

Check identifier
`regex`

Flag to enable
`regex`

Zu ignorierende Markierung
`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$
```

Identische Pluralformen

Zusammenfassung
Einige Pluralformen sind identisch übersetzt

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.consistency.SamePluralsCheck`

Check identifier
`same-plurals`

Zu ignorierende Markierung
`ignore-same-plurals`

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

Zeilenumbruch am Anfang

Zusammenfassung
Ausgangs- und übersetzte Zeichenkette beginnen nicht beide mit einem Zeilenumbruch

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.BeginNewlineCheck`

Check identifier
`begin_newline`

Zu ignorierende Markierung
`ignore-begin-newline`

Zeilenumbrüche erscheinen in der Regel aus gutem Grund in den Quelltexten, denn fehlende oder hinzugefügte Zeilenumbrüche können zu Formatierungsproblemen führen, wenn der übersetzte Text verwendet wird.

Siehe auch:

Zeilenumbruch am Ende

Leerzeichen am Anfang

Zusammenfassung
Ausgangs- und übersetzte Zeichenkette beginnen mit unterschiedlich vielen Leerzeichen

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.BeginSpaceCheck`

Check identifier
`begin_space`

Zu ignorierende Markierung
`ignore-begin-space`

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

Zeilenumbruch am Ende

Zusammenfassung
Quelle und Übersetzung enden nicht beide mit einem Zeilenumbruch

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.EndNewlineCheck`

Check identifier
`end_newline`

Zu ignorierende Markierung
`ignore-end-newline`

Zeilenumbrüche erscheinen in der Regel aus gutem Grund in den Quelltexten, denn fehlende oder hinzugefügte Zeilenumbrüche können zu Formatierungsproblemen führen, wenn der übersetzte Text verwendet wird.

Siehe auch:

Zeilenumbruch am Anfang

Leerzeichen am Ende

Zusammenfassung
Ausgangs- und übersetzte Zeichenkette enden nicht beide mit einem Leerzeichen

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.EndSpaceCheck`

Check identifier
`end_space`

Zu ignorierende Markierung
`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.

Unveränderte Übersetzung

Zusammenfassung
Ausgangs- und übersetzte Zeichenkette sind identisch

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.same.SameCheck`

Check identifier
`same`

Zu ignorierende Markierung
`ignore-same`

Tritt ein, wenn der Ausgangstext und die entsprechende Übersetzung identisch sind, bis hin zu mindestens einer der Pluralformen. Einige Zeichenfolgen, die in allen Sprachen vorkommen, werden ignoriert, und verschiedene Auszeichnungselemente werden entfernt. Dadurch wird die Anzahl der falsch-positiven Ergebnisse reduziert.

This check can help find strings mistakenly untranslated.

Das Standardverhalten dieser Prüfung ist, dass Wörter aus der eingebauten schwarzen Liste von der Prüfung ausgeschlossen werden. Dies sind Wörter, die häufig nicht übersetzt werden. Dies ist nützlich, um Fehlalarme bei kurzen Zeichenketten zu vermeiden, die nur aus einem einzigen Wort bestehen, das in mehreren Sprachen gleich ist. Diese schwarze Liste kann durch Hinzufügen der Markierung``strict-same`` zur Zeichenkette oder Komponente deaktiviert werden.

Siehe auch:

Component configuration, *Customizing behavior using flags*

Unsicheres HTML

Neu in Version 3.9.

Zusammenfassung
Die Übersetzung verwendet unsichere HTML-Markierungen

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.markup.SafeHTMLCheck`

Check identifier
`safe-html`

Flag to enable
`safe-html`

Zu ignorierende Markierung
`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.

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

Siehe auch:

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

URL

Neu in Version 3.5.

Zusammenfassung
Die Übersetzung enthält keine URL

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.markup.URLCheck`

Check identifier
`url`

Flag to enable
`url`

Zu ignorierende Markierung
`ignore-url`

Die Übersetzung enthält keine URL. Dies wird nur ausgelöst, wenn die Einheit als URL-haltig markiert ist. In diesem Fall muss die Übersetzung eine gültige URL sein.

XML-Auszeichnung

Zusammenfassung
XML-Tags in der Übersetzung passen nicht zur Ausgangszeichenkette

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.markup.XMLTagsCheck`

Check identifier
`xml-tags`

Zu ignorierende Markierung
`ignore-xml-tags`

Das bedeutet in der Regel, dass die Ausgabe anders aussieht. In den meisten Fällen ist dies kein erwünschtes Ergebnis einer Änderung der Übersetzung, aber gelegentlich schon.

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

Bemerkung: Diese Prüfung wird durch das `safe-html`-Flag deaktiviert, da die von ihr durchgeführte HTML-Bereinigung HTML-Markup erzeugen kann, das kein gültiges XML ist.

XML-Syntax

Neu in Version 2.8.

Zusammenfassung
Die Übersetzung ist kein gültiges XML

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.markup.XMLValidityCheck`

Check identifier
`xml-invalid`

Zu ignorierende Markierung
`ignore-xml-invalid`

The XML markup is not valid.

Bemerkung: Diese Prüfung wird durch das `safe-html`-Flag deaktiviert, da die von ihr durchgeführte HTML-Bereinigung HTML-Markup erzeugen kann, das kein gültiges XML ist.

Leerzeichen ohne Breite

Zusammenfassung
Übersetzung enthält zusätzliche Leerzeichen ohne Breite

Zielgruppe
Übersetzte Zeichenketten

Klasse prüfen
`weblate.checks.chars.ZeroWidthSpaceCheck`

Check identifier
`zero-width-space`

Zu ignorierende Markierung
`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.

Siehe auch:

Leerzeichen ohne Breite auf Wikipedia

1.6.4 Source checks

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

Auslassungspunkte

Zusammenfassung
Die Zeichenkette verwendet drei Punkte (…) anstelle der Auslassungspunkte (…)

Zielgruppe
Ausgangszeichenketten

Klasse prüfen
`weblate.checks.source.EllipsisCheck`

Check identifier
`ellipsis`

Zu ignorierende Markierung
`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.

Siehe auch:

Auslassungspunkte auf Wikipedia

ICU MessageFormat-Syntax

Neu in Version 4.9.

Zusammenfassung
Syntaxfehler in ICU MessageFormat-Zeichenketten.

Zielgruppe
Ausgangszeichenketten

Klasse prüfen
`weblate.checks.icu.ICUSourceCheck`

Check identifier
`icu_message_format_syntax`

Flag to enable
`icu-message-format`

Zu ignorierende Markierung
`ignore-icu-message-format`

Siehe auch:

ICU MessageFormat

Lange nicht übersetzt

Neu in Version 4.1.

Zusammenfassung
Die Zeichenkette wurde lange Zeit nicht übersetzt

Zielgruppe
Ausgangszeichenketten

Klasse prüfen
`weblate.checks.source.LongUntranslatedCheck`

Check identifier
`long_untranslated`

Zu ignorierende Markierung
`ignore-long-untranslated`

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

Mehrfach fehlgeschlagene Überpüfungen

Zusammenfassung
In mehreren Sprachen liegen bei den Übersetzungen fehlgeschlagene Qualitätsprüfungen vor

Zielgruppe
Ausgangszeichenketten

Klasse prüfen
`weblate.checks.source.MultipleFailingCheck`

Check identifier
`multiple_failures`

Zu ignorierende Markierung
`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.

Mehrere unbenannte Variablen

Neu in Version 4.1.

Zusammenfassung
Es gibt mehrere unbenannte Variablen in der Zeichenkette, was es für Übersetzer unmöglich macht, sie neu anzuordnen

Zielgruppe
Ausgangszeichenketten

Klasse prüfen
`weblate.checks.format.MultipleUnnamedFormatsCheck`

Check identifier
`unnamed_format`

Zu ignorierende Markierung
`ignore-unnamed-format`

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

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

Ohne Pluralformen

Zusammenfassung
Die Zeichenkette wird im Plural verwendet, hat aber keine Pluralformen

Zielgruppe
Ausgangszeichenketten

Klasse prüfen
`weblate.checks.source.OptionalPluralCheck`

Check identifier
`optional_plural`

Zu ignorierende Markierung
`ignore-optional-plural`

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

For example with Gettext in Python it could be:

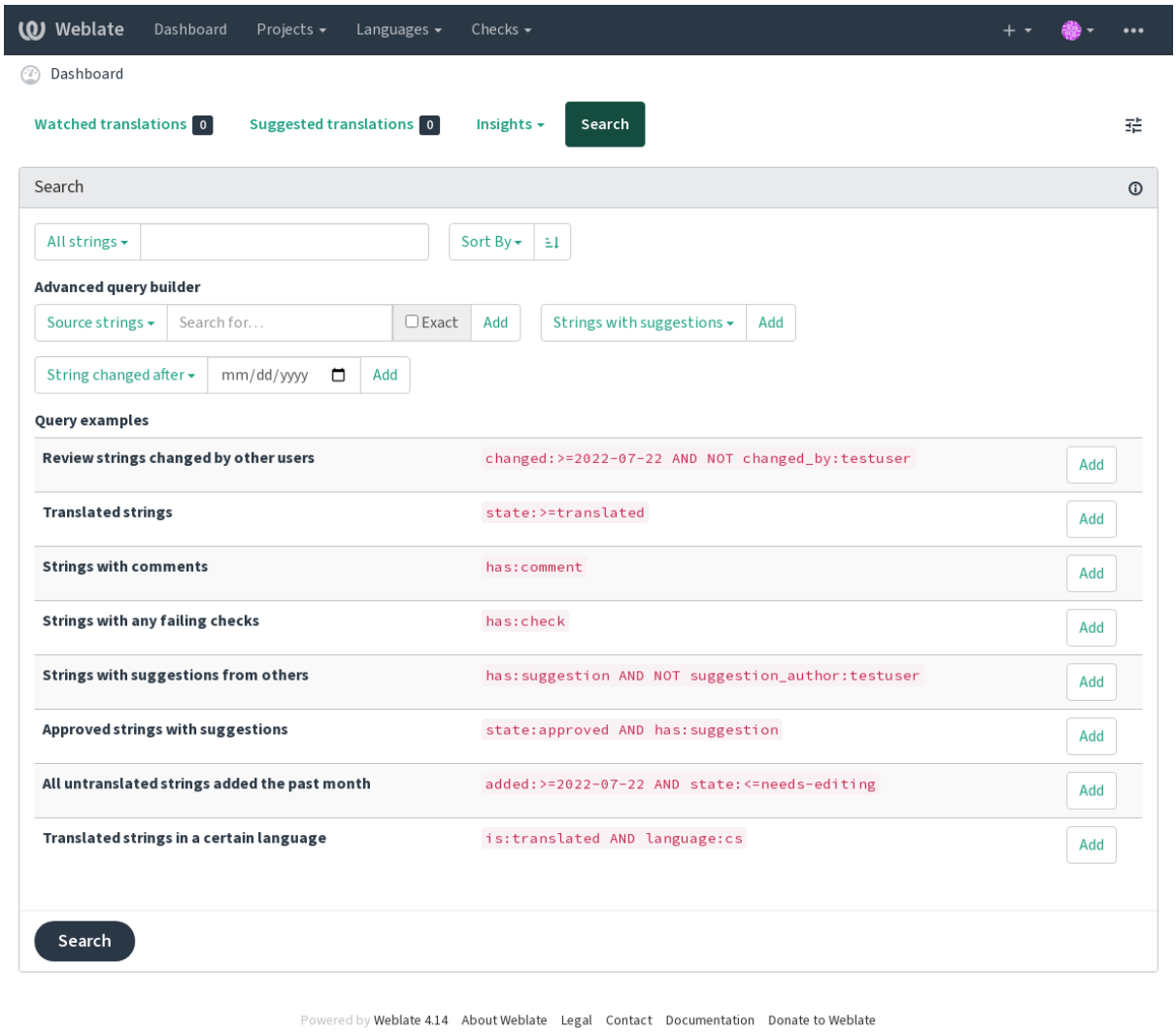
```
from gettext import ngettext

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

1.7 Searching

Neu in Version 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, target, and context strings.



1.7.1 Simple search

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

1.7.2 Fields

`source:TEXT`
Suche in Ausgangszeichenkette ohne Berücksichtigung der Groß-/Kleinschreibung.

`target:TEXT`
Target string case-insensitive search.

`context:TEXT`
Context string case-insensitive search.

`key:TEXT`
Key string case-insensitive search.

`note:TEXT`
Suche in Beschreibung der Ausgangszeichenkette ohne Berücksichtigung der Groß-/Kleinschreibung.

`location:TEXT`
Location string case-insensitive search.

`priority:NUMBER`
String priority.

`added:DATETIME`
Timestamp for when the string was added to Weblate.

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

`pending:BOOLEAN`
String pending for flushing to VCS.

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

`is:TEXT`
Search for string states (`pending`, `translated`, `untranslated`).

`language:TEXT`
String target language.

`component:TEXT`
Component slug or name case-insensitive search, see *Component slug* and *Name der Komponente*.

`project:TEXT`
Project slug, see *URL-Kurzbegriff*.

`changed_by:TEXT`
String was changed by author with given username.

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

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

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

check:TEXT
String has failing check, see *Checks and fixups* for check identifiers.

dismissed_check:TEXT
String has dismissed check, see *Checks and fixups* for check identifiers.

comment:TEXT
Search in user comments.

resolved_comment:TEXT
Suche in geklärten Kommentaren.

comment_author:TEXT
Filter by comment author.

suggestion:TEXT
Search in suggestions.

suggestion_author:TEXT
Filter by suggestion author.

explanation:TEXT
Suche in Erklärungen.

label:TEXT
Search in labels.

screenshot:TEXT
Suche in Bildschirmfotos.

1.7.3 Boolean operators

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

1.7.4 Field operators

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

state:>=translated
State is `translated` or better (`approved`).

changed:2019
Changed in year 2019.

changed:[2019-03-01 to 2019-04-01]
Changed between two given dates.

1.7.5 Exact operators

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

1.7.6 Searching for changes

Neu in Version 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"`.

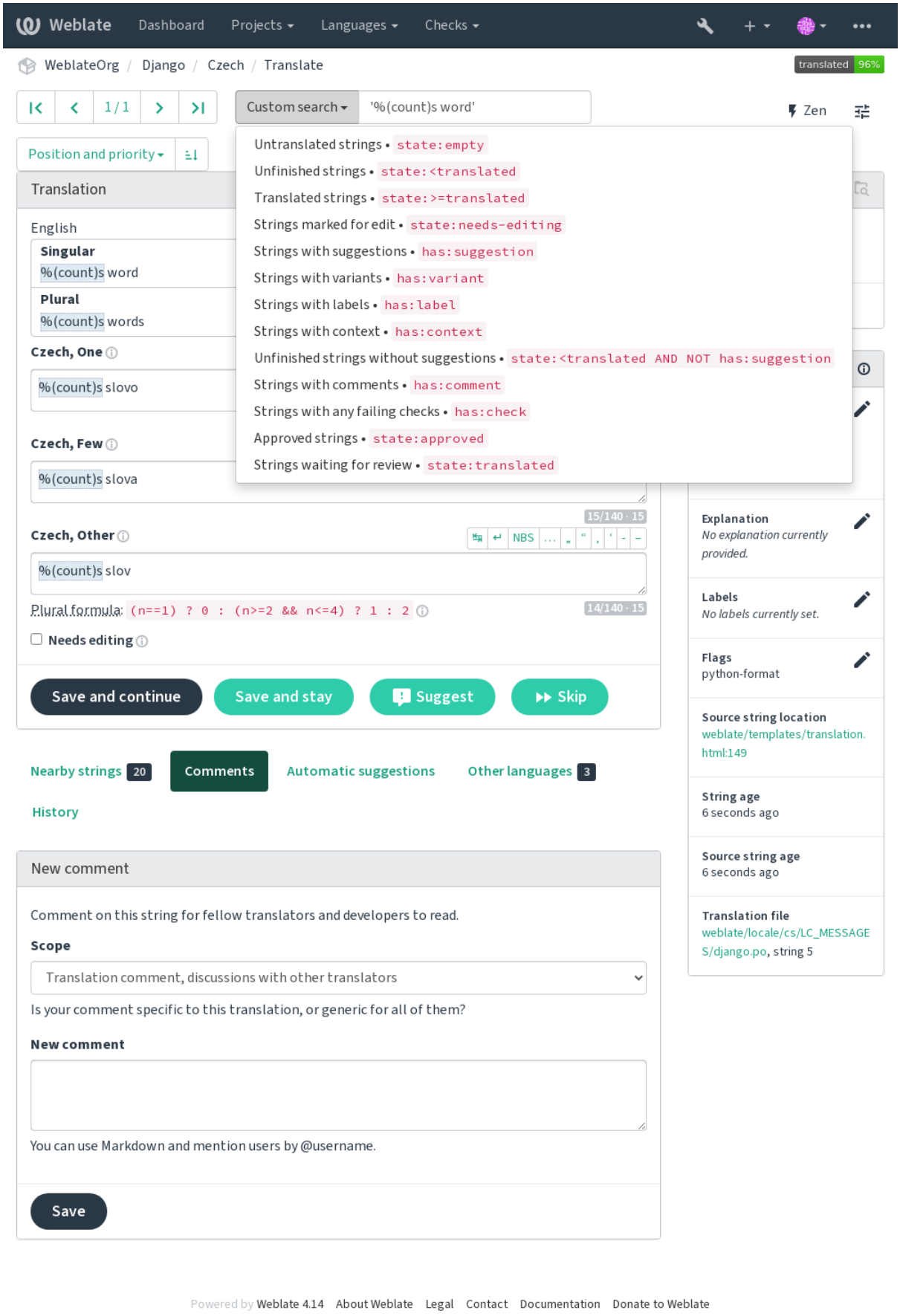
1.7.7 Reguläre Ausdrücke

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

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

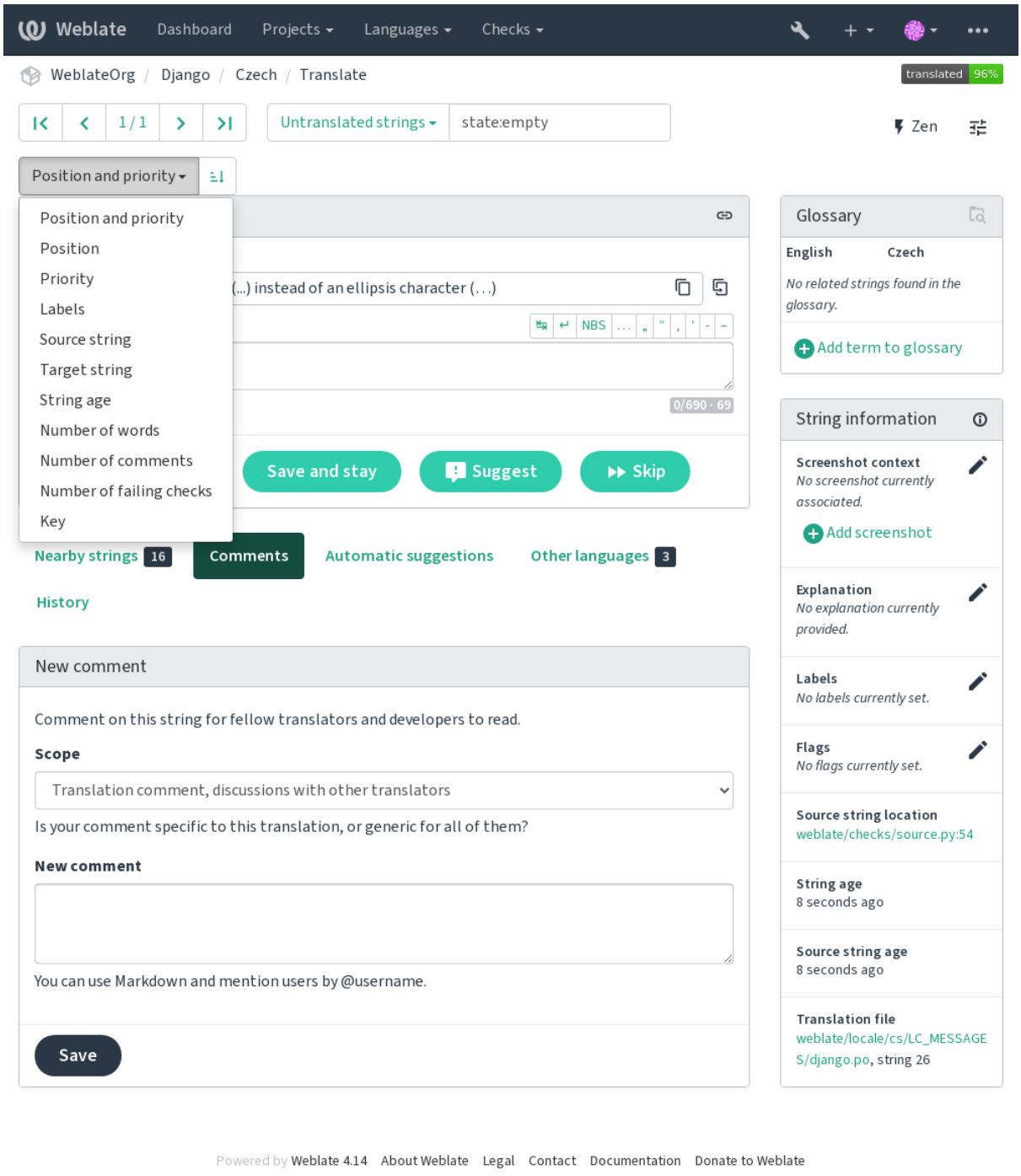
1.7.8 Predefined queries

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


Weblate
Dashboard
Projects
Languages
Checks
WeblateOrg / Django / Czech / Translate
translated 96%
1 / 1
Custom search
'%(count)s word'
Zen
Position and priority
Untranslated strings • state:empty
Unfinished strings • state:<translated
Translated strings • state:>=translated
Strings marked for edit • state:needs-editing
Strings with suggestions • has:suggestion
Strings with variants • has:variant
Strings with labels • has:label
Strings with context • has:context
Unfinished strings without suggestions • state:<translated AND NOT has:suggestion
Strings with comments • has:comment
Strings with any failing checks • has:check
Approved strings • state:approved
Strings waiting for review • state:translated
Translation
English
Singular
%(count)s word
Plural
%(count)s words
Czech, One
%(count)s slovo
Czech, Few
%(count)s slova
15/140 · 15
Czech, Other
%(count)s slov
Plural formula: (n==1) ? 0 : (n>=2 && n<=4) ? 1 : 2
14/140 · 15
Needs editing
Save and continue
Save and stay
Suggest
Skip
Nearby strings 20
Comments
Automatic suggestions
Other languages 3
History
New comment
Comment on this string for fellow translators and developers to read.
Scope
Translation comment, discussions with other translators
Is your comment specific to this translation, or generic for all of them?
New comment
You can use Markdown and mention users by @username.
Save
Explanation
No explanation currently provided.
Labels
No labels currently set.
Flags
python-format
Source string location
weblate/templates/translation.html:149
String age
6 seconds ago
Source string age
6 seconds ago
Translation file
weblate/locale/cs/LC_MESSAGES/django.po, string 5
Powered by Weblate 4.14 About Weblate Legal Contact Documentation Donate to Weblate

1.7.9 Ordering the results

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



1.8 Übersetzungs-Workflows

Der Einsatz von Weblate bedeutet mehr Nähe zwischen Ihnen und Ihren Benutzern und damit auch mehr Nähe zwischen Ihnen und Ihren Übersetzern. Es liegt an Ihnen zu entscheiden, wie viele der Funktionen Sie nutzen möchten.

Die folgende Liste ist keine vollständige Auflistung der Konfigurationsmöglichkeiten von Weblate. Sie können weitere Workflows auf der Grundlage der hier aufgelisteten gängigen Beispiele erstellen.

1.8.1 Zugriff auf die Übersetzung

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 den folgenden Kapiteln bedeutet *jeder Benutzer* einen Benutzer, der Zugriff auf die Übersetzung hat. Das kann jeder authentifizierte Benutzer sein, wenn das Projekt öffentlich ist, oder ein Benutzer, der eine *Translate*-Berechtigung für das Projekt hat.

1.8.2 Translation states

Jede übersetzte Zeichenkette kann sich in einem der folgenden Zustände befinden:

Nicht übersetzt
Die Übersetzung ist leer, je nach Dateiformat kann sie in der Datei gespeichert werden oder nicht.

Bearbeitung erforderlich
Die Übersetzung muss bearbeitet werden. Dies ist in der Regel das Ergebnis einer Änderung der Ausgangszeichenkette, eines Fuzzy Matchings oder einer Aktion des Übersetzers. Die Übersetzung wird in der Datei gespeichert. Je nach Dateiformat kann sie als bearbeitungsbedürftig gekennzeichnet sein (z.B. durch ein `fuzzy`-Flag in der Gettext-Datei).

Ausstehende Überprüfungen
Die Übersetzung wird vorgenommen, aber nicht überprüft. Sie wird in der Datei als eine gültige Übersetzung gespeichert.

Genehmigt
Die Übersetzung wurde in der Überprüfung genehmigt. Sie kann nicht mehr von Übersetzern geändert werden, sondern nur noch von Prüfern. Übersetzer können nur noch Vorschläge einreichen.

This state is only available when reviews are enabled.

Vorschläge
Vorschläge werden nur in Weblate und nicht in der Übersetzungsdatei gespeichert.

Die Zustände werden soweit möglich in den Übersetzungsdateien dargestellt.

Hinweis: Falls das von Ihnen verwendete Dateiformat das Speichern von Zuständen nicht unterstützt, sollten Sie die Erweiterung *Unveränderte Übersetzungen als „bearbeitungsbedürftig“ markieren* verwenden, um unveränderte Zeichenketten als zu bearbeiten zu kennzeichnen.

Siehe auch:

Translation types capabilities, *Übersetzungs-Workflows*

1.8.3 Direkte Übersetzung

Dies ist die übliche Einstellung für kleinere Teams, in denen jeder direkt übersetzen kann. Dies ist auch die Standardeinstellung in Weblate.

- *Jeder Benutzer* kann Übersetzungen bearbeiten.
- Vorschläge sind eine optionale Möglichkeit, Änderungen vorzuschlagen, wenn sich die Übersetzer über die Änderung unsicher sind.

Einstellung	Value	Anmerkung
Begutachtung aktivieren	aus	Auf Projektebene konfiguriert.
Vorschläge aktivieren	an	Es ist nützlich für die Benutzer, Vorschläge machen zu können, wenn sie unsicher sind.
Abstimmen über Vorschläge	aus	
Vorschläge automatisch annehmen	0	
Übersetzergruppe	*Benutzer*	Oder *Übersetzen* mit *Projekt-Zugriffssteuerung*.
Prüfergruppe	N/A	Nicht verwendet.

1.8.4 Peer-Review

In diesem Workflow kann jeder Übersetzungsvorschläge einbringen, die von weiteren Mitgliedern genehmigt werden müssen, bevor sie als Übersetzung akzeptiert werden.

- *Jeder Benutzer* kann Übersetzungsvorschläge hinzufügen.
- *Jeder Benutzer* kann für Übersetzungsvorschläge abstimmen.
- Vorschläge werden zu Übersetzungen, wenn sie eine bestimmte Anzahl von Stimmen erhalten haben.

Einstellung	Value	Anmerkung
Begutachtung aktivieren	aus	Auf Projektebene konfiguriert.
Vorschläge aktivieren	an	
Abstimmen über Vorschläge	aus	
Vorschläge automatisch annehmen	1	Sie können einen höheren Wert einstellen, um mehr Peer-Reviews anzufordern.
Übersetzergruppe	*Benutzer*	Oder *Übersetzen* mit *Projekt-Zugriffssteuerung*.
Prüfergruppe	N/A	Nicht verwendet, alle Übersetzer überprüfen.

1.8.5 Zugehörige Prüfer

Neu in Version 2.18: The proper review workflow is supported since Weblate 2.18.

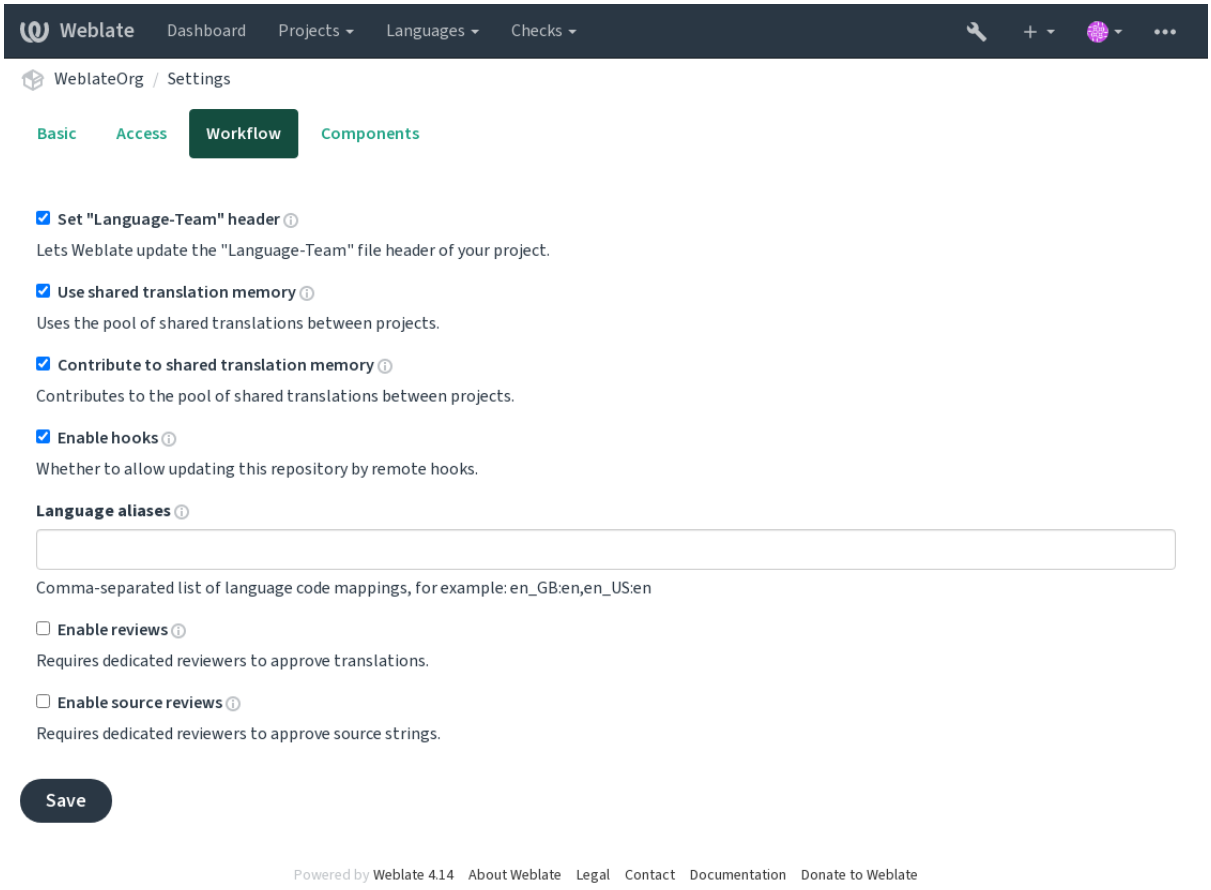
Mit speziellen Prüfern haben Sie zwei Benutzergruppen, von denen eine Übersetzungen einreichen kann und die andere sie überprüft, um sicherzustellen, dass die Übersetzungen konsistent sind und die Qualität stimmt.

- *Jeder Benutzer* kann nicht genehmigte Übersetzungen bearbeiten.
- *Prüfer* kann Zeichenketten genehmigen / nicht genehmigen.
- *Prüfer* können alle Übersetzungen (einschließlich genehmigten) bearbeiten.
- Vorschläge können auch verwendet werden, um Änderungen für genehmigte Zeichenketten vorzuschlagen.

Einstellung	Value	Anmerkung
Begutachtung aktivieren	an	Auf Projektebene konfiguriert.
Vorschläge aktivieren	aus	Es ist nützlich für die Benutzer, Vorschläge machen zu können, wenn sie unsicher sind.
Abstimmen über Vorschläge	aus	
Vorschläge automatisch annehmen	0	
Übersetzergruppe	*Benutzer*	Oder *Übersetzen* mit *Projekt-Zugriffssteuerung*.
Prüfergruppe	*Prüfer*	Or *Review* with *per-project access control*.

1.8.6 Überprüfungen einschalten

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

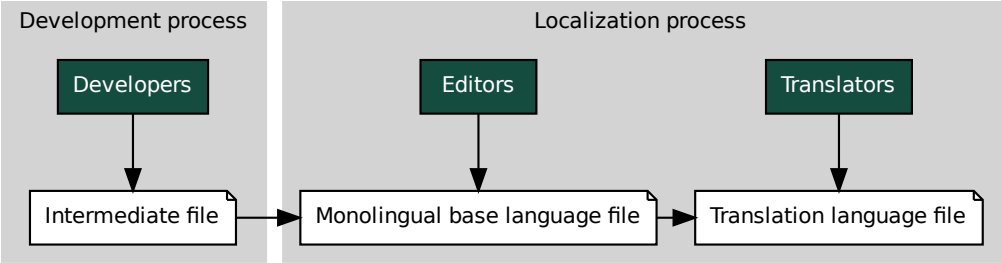


Bemerkung: Je nach Weblate-Konfiguration steht Ihnen diese Einstellung möglicherweise nicht zur Verfügung. Bei Hosted Weblate zum Beispiel ist diese Einstellung für kostenlos gehostete Projekte nicht verfügbar.

1.8.7 Qualitäts-Gateway für die Ausgangszeichenketten

In vielen Fällen stammen die ursprünglichen Zeichenketten der Ausgangssprache von den Entwicklern, da sie den Code schreiben und die ursprünglichen Zeichenketten bereitstellen. Die Entwickler sind jedoch oft keine Muttersprachler in der Ausgangssprache und bieten nicht die gewünschte Qualität der Quelltexte. Die Zwischenübersetzung kann Ihnen dabei helfen, dies zu beheben - es gibt eine zusätzliche Qualitätskontrolle für die Texte zwischen Entwicklern, Übersetzern und Benutzern.

Wenn Sie *Zwischensprachedatei* einstellen, wird diese Datei als Quelle für die Zeichenketten verwendet, aber sie wird in der Ausgangssprache bearbeitet, um sie zu optimieren. Sobald die Zeichenkette in der Ausgangssprache fertig ist, steht sie auch für Übersetzer zur Verfügung, die sie in weitere Sprachen übersetzen.



Siehe auch:

Zwischensprachedatei, *Einsprachige Basis-Sprachdatei*, *Bilingual and monolingual formats*

1.8.8 Überprüfungen der Ausgangszeichenketten

Wenn *Quellenüberprüfung aktivieren* aktiviert ist, kann der Überprüfungsprozess auf die Quelltexte angewendet werden. Sobald er aktiviert ist, können Benutzer Probleme in den Quelltexten melden. Der tatsächliche Prozess hängt davon ab, ob Sie ein- oder zweisprachige Formate verwenden.

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

Die zweisprachigen Formate erlauben keine direkte Bearbeitung der Ausgangszeichenketten (diese werden normalerweise direkt aus dem Quellcode extrahiert). In diesem Fall wird das Label *Quelltext muss überprüft werden* an Strings angehängt, die von Übersetzern gemeldet werden. Sie sollten solche Zeichenfolgen überprüfen und sie entweder im Quellcode bearbeiten oder die Kennzeichnung entfernen.

Siehe auch:

Bilingual and monolingual formats, *Zugehörige Prüfer*, labels, *Kommentare*

1.9 Frequently Asked Questions

1.9.1 Konfiguration

How to create an automated workflow?

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

1. Set up your Git repository to tell Weblate when there is any change, see *Benachrichtigungs-Hooks* 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 *Bei Commit gleichzeitig Pushen* on your *Component configuration* in Weblate, this will make Weblate push changes to your repository whenever they happen at Weblate.

Siehe auch:

Kontinuierliche Lokalisierung, *Avoiding merge conflicts*

How to access repositories over SSH?

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

How to fix merge conflicts in translations?

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

Wenn Sie bereits auf einen Merge-Konflikt gestoßen sind, ist es am einfachsten, alle Konflikte lokal auf Ihrem Rechner zu lösen, indem Sie Weblate als Remote-Repository hinzufügen, es mit Upstream zusammenführen und alle Konflikte beheben. Sobald Sie die Änderungen pushen, kann Weblate die zusammengeführte Version ohne weitere besondere Maßnahmen verwenden.

Bemerkung: Depending on your setup, access to the Weblate repository might require authentication. When using the built-in *Git exporter* in Weblate, you authenticate with your username and the API key.

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

# Update weblate remote:
git remote update weblate

# Merge Weblate changes:
git merge weblate/main

# Resolve conflicts:
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
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 weblates-second/main
... # Resolve conflicts
git commit

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

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

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

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

# Update Weblate remote:
git remote update weblate

# Merge Weblate changes:
git merge weblate/main

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

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

Siehe auch:

How to export the Git repository that Weblate uses?, *Kontinuierliche Lokalisierung*, *Avoiding merge conflicts*, *Weblate Client*

How do I translate several branches at once?

Weblate unterstützt das Pushen von Übersetzungsänderungen innerhalb eines Projekts. Für jede Komponente, die dies aktiviert hat (Standardeinstellung), wird die vorgenommene Änderung automatisch an andere weitergegeben. Auf diese Weise werden die Übersetzungen synchron gehalten, auch wenn die Zweige selbst schon sehr weit auseinander liegen und es nicht möglich ist, Übersetzungsänderungen einfach zwischen ihnen zusammenzuführen.

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
```

Siehe auch:

Keeping translations same across components

How to translate multi-platform projects?

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

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

Siehe auch:

Keeping translations same across components

How to export the Git repository that Weblate uses?

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

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

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

What are the options for pushing changes back upstream?

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

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

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

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

You can use git submodule for separating translations from source code while still having them under version control.

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

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

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

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

Please consult the git submodule documentation for more details.

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

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

Siehe auch:

Monitoring Weblate, *Monitoring Celery status*

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

This is the default committer name, configured by `DEFAULT_COMMITER_EMAIL` and `DEFAULT_COMMITER_NAME`.

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

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

Siehe auch:

Component configuration

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

Um den Verlauf, die Kommentare oder Bildschirmfotos, die mit Zeichenketten verknüpft sind, nach einer Änderung des Dateispeicherorts beizubehalten, müssen Sie sicherstellen, dass diese Zeichenketten in Weblate niemals gelöscht werden. Dies kann passieren, wenn das Weblate-Repository aktualisiert wird, die Komponentenkonfiguration aber noch auf die alten Dateien verweist. Dies führt dazu, dass Weblate davon ausgeht, dass es alle Übersetzungen löschen sollte.

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

1. Sperren Sie die betroffene Komponente in Weblate.
2. Commiten Sie alle ausstehenden Änderungen und führen Sie sie in das Upstream-Repository ein.
3. Disable receiving webhooks the *Project configuration*; this prevents Weblate from immediately seeing changes in the repository.
4. Do any needed changes in the repo (for example using **`git mv`**), push them to the upstream repository.
5. Change the *Component configuration* to match the new setup; upon changing configuration, Weblate will fetch the updated repository and notice the changed locations while keeping existing strings.

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

1.9.2 Usage

How do I review the translations of others?

- There are several review based workflows available in Weblate, see *Übersetzungs-Workflows*.
- Sie können alle Änderungen, die in Abonnements gemacht werden, abonnieren und dann andere Beiträge, die per E-Mail eintreffen, überprüfen.
- 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.

Siehe auch:

Übersetzungs-Workflows

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.

Siehe auch:

report-source, *Kommentare*

How can I use existing translations while translating?

- Dank des gemeinsamen Übersetzungsspeichers können alle Übersetzungen innerhalb von Weblate verwendet werden.
- Sie können vorhandene Übersetzungsspeicherdateien in Weblate importieren.
- 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.

Siehe auch:

Configuring automatic suggestions, *Automatische Vorschläge*, *Übersetzungsspeicher*

Does Weblate update translation files besides translations?

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

Siehe auch:

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.

Siehe auch:

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.

Bemerkung: It is usually a good idea to merge changes done in Weblate before updating translation files, as otherwise you will usually end up with some conflicts to merge.

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

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

In case you want to do the update automatically, you can install add-on *PO-Dateien auf POT aktualisieren (msgmerge)*.

Siehe auch:

updating-target-files

1.9.3 Troubleshooting

Requests sometimes fail with „too many open files“ error

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

The easiest way to do this is to run:

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

Siehe auch:

DATA_DIR

When accessing the site I get a „Bad Request (400)“ error

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

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

Siehe auch:

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.

- In case the translation to the source language is desired, please change the *Ausgangssprache* in the component settings.
- Falls die Übersetzungsdatei für die Ausgangssprache nicht benötigt wird, entfernen Sie diese bitte aus dem Repository.
- In case the translation file for the source language is needed, but should be ignored by Weblate, please adjust the *Sprachen-Filter* to exclude it.

Hinweis: 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 Funktionen

Does Weblate support other VCSes than Git and Mercurial?

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

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

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

Bemerkung: 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.

Siehe auch:

Integration der Versionsverwaltung

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.

Siehe auch:

`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.

Geändert in Version 4.2: Übersetzer können alle Komponenten eines Projekts als Ganzes in eine bestimmte Sprache übersetzen.

Why does Weblate use language codes such sr_Latn or zh_Hant?

Dies sind Sprachcodes, die durch **RFC 5646** definiert sind, um besser zu zeigen, dass es sich wirklich um verschiedene Sprachen handelt, anstatt der bisher fälschlicherweise verwendeten Modifikatoren (für `@latin` Varianten) oder Ländercodes (für Chinesisch).

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`.

Bemerkung: Weblate verwendet standardmäßig Sprachcodes im POSIX-Stil mit Unterstrich, siehe *Language definitions* für weitere Details.

Siehe auch:

Language definitions, *Stil des Sprachcodes*, *Adding new translations*

1.10 Supported file formats

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

Siehe auch:

Translation Related File Formats

Bemerkung: When choosing a file format for your application, it's better to stick some well established format in the toolkit/platform you use. This way your translators can additionally use whatever tools they are used to, and will more likely contribute to your project.

1.10.1 Bilingual and monolingual formats

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

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

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

1.10.2 Automatische Detektion

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

1.10.3 Translation types capabilities

Capabilities of all supported formats:

Format	Linguality[Seite 80, 1]	Plurals[Seite 80, 2]	Descriptions[Seite 80, 3]	Context[Seite 80, 4]	Location[Seite 80, 5]	Flags[Seite 80, 8]	Additional states[Seite 80, 6]
GNU gettext	bilingual	yes	yes	yes	yes	yes[9]	needs editing
Monolingual gettext	mono	yes	yes	yes	yes	yes[Seite 80, 9]	needs editing
XLIFF	both	yes	yes	yes	yes	yes[10]	needs editing, approved
Java properties	both	no	yes	no	no	no	
mi18n lang Dateien	mono	no	yes	no	no	no	
GWT-Eigenschaften	mono	yes	yes	no	no	no	
Joomla translations	mono	no	yes	no	yes	no	
Qt Linguist .ts	both	yes	yes	no	yes	yes[Seite 80, 10]	needs editing
Android string resources	mono	yes	yes[7]	no	no	yes[Seite 80, 10]	
Apple iOS strings	both	no	yes	no	no	no	
PHP-Zeichenketten	mono	no[11]	yes	no	no	no	
JSON files	mono	no	no	no	no	no	

Fortsetzung auf der nächsten Seite

Tab. 1 – Fortsetzung der vorherigen Seite

Format	Linguality[Seite 80, 1]	Plurals[Seite 80, 2]	Descriptions[Seite 80, 3]	Context[Seite 80, 4]	Location[Seite 80, 5]	Flags[Seite 80, 8]	Additional states[Seite 80, 6]
JSON i18next files	mono	yes	no	no	no	no	
go-i18n JSON files	mono	yes	no	no	no	no	
ARB File	mono	yes	yes	no	no	no	
WebExtension JSON	mono	yes	yes	no	no	no	
.XML resource files	mono	no	yes	no	no	yes[Seite 80, 10]	
ResourceDictionary files	mono	no	no	no	no	yes[Seite 80, 10]	
CSV files	both	no	yes	yes	yes	no	needs editing
YAML files	mono	no	yes	no	no	no	
Ruby YAML files	mono	yes	yes	no	no	no	
DTD files	mono	no	no	no	no	no	
Flat XML files	mono	no	no	no	no	yes[Seite 80, 10]	
Windows RC files	mono	no	yes	no	no	no	
Excel Open XML	mono	no	yes	yes	yes	no	needs editing
App-Store Metadatendateien	mono	no	no	no	no	no	
Subtitle files	mono	no	no	no	yes	no	
HTML files	mono	no	no	no	no	no	
OpenDocument Format	mono	no	no	no	no	no	
IDML Format	mono	no	no	no	no	no	
INI translations	mono	no	no	no	no	no	
Inno Setup INI translations	mono	no	no	no	no	no	
TermBase eXchange format	bilingual	no	yes	no	no	yes[Seite 80, 10]	
Textdateien	mono	no	no	no	no	no	
Stringsdict-Format	mono	yes	yes	no	no	no	
Fluent-Format	mono	no[12]	yes	no	no	no	

Schreibgeschützte Zeichenketten

Neu in Version 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 *Android string resources*), but can be emulated in others by adding a `read-only` flag, see *Customizing behavior using flags*.

1.10.4 GNU gettext

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:

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

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

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

Typical Weblate *Component configuration*	
Dateimaske	`po/*.po`
Einsprachige Basis-Sprachdatei	*Empty*
Vorlage für neue Übersetzungen	`po/messages.pot`
Dateiformat	*Gettext PO file*

Siehe auch:

devel/gettext, devel/sphinx, Gettext on Wikipedia, PO Files, *Aktualisieren Sie die Variable ALL_LINGUAS in der „configure“-Datei*, *Ausgabe von Gettext anpassen*, *LINGUAS-Datei aktualisieren*, *MO-Dateien erzeugen*, *PO-Dateien auf POT aktualisieren (msgmerge)*

[1] See *Bilingual and monolingual formats*
[2] Plurals are necessary to properly localize strings with variable count.
[3] Source string descriptions can be used to pass additional info about the string to translate.
[4] Context is used to differentiate identical strings used in different scopes (for example *Sun* can be used as an abbreviated name of the day „Sunday“ or as the name of our closest star).
[5] Location of a string in source code might help proficient translators figure out how the string is used.
[8] See *Customizing behavior using flags*
[6] Additional states supported by the file format in addition to „Untranslated“ and „Translated“.
[9] The gettext type comments are used as flags.
[10] The flags are extracted from the non-standard attribute `weblate-flags` for all XML based formats. Additionally `max-length:N` is supported through the `maxwidth` attribute as defined in the XLIFF standard, see *Specifying translation flags*.
[7] XML comment placed before the `<string>` element, parsed as a source string description.
[11] The plurals are supported only for Laravel which uses in string syntax to define them, see Localization in Laravel.
[12] Plurale werden in der Syntax der Zeichenketten behandelt und nicht als Plural in Weblate angezeigt.

Monolingual gettext

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

The monolingual gettext PO file typically looks like this:

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

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

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

While the base language file will be:

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

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

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

Typical Weblate *Component configuration*	
Dateimaske	`po/*.po`
Einsprachige Basis-Sprachdatei	`po/en.po`
Vorlage für neue Übersetzungen	`po/messages.pot`
Dateiformat	*Gettext PO file (monolingual)*

1.10.5 XLIFF

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

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

Weblate supports XLIFF in several variants:

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

XLIFF with placeables support
: Standard XLIFF supporting placeables and other XML elements.

XLIFF mit Gettext-Erweiterungen
: XLIFF enriched by XLIFF 1.2 Representation Guide for Gettext PO to support plurals.

Siehe auch:

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

Translation states

Geändert in Version 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*.

Siehe auch:

Zugehörige Prüfer

Leerzeichen und Zeilenumbrüche in XLIFF

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

For example:

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

Specifying translation flags

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

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

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

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.

Typical Weblate *Component configuration* for bilingual XLIFF	
Dateimaske	`localizations/*.xliff`
Einsprachige Basis-Sprachdatei	*Empty*
Vorlage für neue Übersetzungen	`localizations/en-US.xliff`
Dateiformat	*XLIFF Translation File*

Typical Weblate *Component configuration* for monolingual XLIFF	
Dateimaske	`localizations/*.xliff`
Einsprachige Basis-Sprachdatei	`localizations/en-US.xliff`
Vorlage für neue Übersetzungen	`localizations/en-US.xliff`
Dateiformat	*XLIFF Translation File*

Siehe auch:

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

1.10.6 Java properties

Native Java format for translations.

Java properties are usually used as monolingual translations.

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

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

Typical Weblate *Component configuration*	
Dateimaske	`src/app/Bundle_*.properties`
Einsprachige Basis-Sprachdatei	`src/app/Bundle.properties`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*Java Properties (ISO-8859-1)*

Siehe auch:

Java-Eigenschaften auf Wikipedia, Mozilla and Java properties files, *mi18n lang Dateien*, *GWT-Eigenschaften*, updating-target-files, *Die Übersetzung der Java-Eigenschaften-Datei formatieren*, *Übersetzungsdateien bereinigen*

1.10.7 mi18n lang Dateien

Neu in Version 4.7.

Dateiformat, das für die JavaScript-Lokalisierung von mi18n verwendet wird. Syntaktisch entspricht es *Java properties*.

Typical Weblate *Component configuration*	
Dateimaske	`*.lang`
Einsprachige Basis-Sprachdatei	`en-US.lang`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*mi18n lang file*

Siehe auch:

mi18n, Mozilla and Java properties files, *Java properties*, updating-target-files, *Die Übersetzung der Java-Eigenschaften-Datei formatieren*, *Übersetzungsdateien bereinigen*

1.10.8 GWT-Eigenschaften

Native GWT format for translations.

GWT properties are usually used as monolingual translations.

Typical Weblate *Component configuration*	
Dateimaske	`src/app/Bundle_*.properties`
Einsprachige Basis-Sprachdatei	`src/app/Bundle.properties`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*GWT Properties*

Siehe auch:

GWT localization guide, GWT Internationalization Tutorial, Mozilla and Java properties files, updating-target-files, *Die Übersetzung der Java-Eigenschaften-Datei formatieren*, *Übersetzungsdateien bereinigen*

1.10.9 INI translations

Neu in Version 4.1.

INI file format for translations.

INI translations are usually used as monolingual translations.

Typical Weblate *Component configuration*	
Dateimaske	`language/*.ini`
Einsprachige Basis-Sprachdatei	`language/en.ini`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*INI File*

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

Siehe auch:

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

1.10.10 Inno Setup INI translations

Neu in Version 4.1.

Inno Setup INI file format for translations.

Inno Setup INI translations are usually used as monolingual translations.

Bemerkung: The only notable difference to *INI translations* is in supporting `%n` and `%t` placeholders for line break and tab.

Typical Weblate *Component configuration*	
Dateimaske	`language/*.islu`
Einsprachige Basis-Sprachdatei	`language/en.islu`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*Inno Setup INI File*

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

Siehe auch:

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

1.10.11 Joomla translations

Neu in Version 2.12.

Native Joomla format for translations.

Joomla translations are usually used as monolingual translations.

Typical Weblate *Component configuration*	
Dateimaske	`language/*/com_foobar.ini`
Einsprachige Basis-Sprachdatei	`language/en-GB/com_foobar.ini`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*Joomla Language File*

Siehe auch:

Mozilla and Java properties files, *INI translations*, *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.

Typical Weblate *Component configuration* when using as bilingual	
Dateimaske	`i18n/app.*.ts`
Einsprachige Basis-Sprachdatei	*Empty*
Vorlage für neue Übersetzungen	`i18n/app.de.ts`
Dateiformat	*Qt Linguist Translation File*

Typical Weblate *Component configuration* when using as monolingual	
Dateimaske	`i18n/app.*.ts`
Einsprachige Basis-Sprachdatei	`i18n/app.en.ts`
Vorlage für neue Übersetzungen	`i18n/app.en.ts`
Dateiformat	*Qt Linguist Translation File*

Siehe auch:

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

1.10.13 Android string resources

Android specific file format for translating applications.

Android string resources are monolingual, the *Einsprachige Basis-Sprachdatei* is stored in a different location from the other files – `res/values/strings.xml`.

Typical Weblate *Component configuration*	
Dateimaske	`res/values-*/strings.xml`
Einsprachige Basis-Sprachdatei	`res/values/strings.xml`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*Android String Resource*

Siehe auch:

Android string resources documentation, Android string resources

---

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

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

become:

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

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

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

1.10.14 Apple iOS strings

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

Apple iOS strings are usually used as monolingual.

Typical Weblate *Component configuration*	
Dateimaske	`Resources/*.lproj/Localizable.strings`
Einsprachige Basis-Sprachdatei	`Resources/en.lproj/Localizable.strings` or `Resources/Base.lproj/Localizable.strings`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*iOS Strings (UTF-8)*

Siehe auch:

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

1.10.15 PHP-Zeichenketten

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

Example file:

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

Typical Weblate *Component configuration*	
Dateimaske	`lang/*/texts.php`
Einsprachige Basis-Sprachdatei	`lang/en/texts.php`
Vorlage für neue Übersetzungen	`lang/en/texts.php`
Dateiformat	*PHP strings*

Laravel PHP-Zeichenketten

Geändert in Version 4.1.

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

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

Siehe auch:

PHP, Localization in Laravel

1.10.16 JSON files

Neu in Version 2.0.

Geändert in Version 2.16: Since Weblate 2.16 and with translate-toolkit at-least 2.2.4, nested structure JSON files are supported as well.

Geändert in Version 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.
- *JSON i18next files*
- *go-i18n JSON files*
- *WebExtension JSON*
- *ARB File*

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

Example file:

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

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

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

Hinweis: 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*	
Dateimaske	`langs/translation-*.json`
Einsprachige Basis-Sprachdatei	`langs/translation-en.json`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*JSON nested structure file*

Siehe auch:

JSON, updating-target-files, *Ausgabe von JSON anpassen*, *Übersetzungsdateien bereinigen*,

1.10.17 JSON i18next files

Geändert in Version 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.

Bemerkung: Weblate unterstützt das i18next JSON-v3-Format. Die Varianten v2 und v1 sind größtenteils kompatibel, mit Ausnahme der Behandlung von Pluralformen.

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

Example file:

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

Typical Weblate *Component configuration*	
Dateimaske	`langs/*.json`
Einsprachige Basis-Sprachdatei	`langs/en.json`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*i18next JSON file*

Siehe auch:

JSON, i18next JSON-Format, updating-target-files, *Ausgabe von JSON anpassen*, *Übersetzungsdateien bereinigen*

1.10.18 go-i18n JSON files

Neu in Version 4.1.

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

Bemerkung: Weblate supports the go-i18n JSON v1 format, for flat JSON formats please use *JSON files*. The v2 format with hash is currently not supported.

Typical Weblate *Component configuration*	
Dateimaske	`langs/*.json`
Einsprachige Basis-Sprachdatei	`langs/en.json`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*go-i18n JSON file*

Siehe auch:

JSON, go-i18n, updating-target-files, *Ausgabe von JSON anpassen*, *Übersetzungsdateien bereinigen*,

1.10.19 ARB File

Neu in Version 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*	
Dateimaske	`lib/l10n/intl_*.arb`
Einsprachige Basis-Sprachdatei	`lib/l10n/intl_en.arb`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*ARB file*

Siehe auch:

JSON, Application Resource Bundle Specification, Internationalizing Flutter apps, updating-target-files, *Ausgabe von JSON anpassen*, *Übersetzungsdateien bereinigen*

1.10.20 WebExtension JSON

Neu in Version 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.

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

Example file:

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

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
        "example": "5"
      }
    }
  },
  "try": {
    "message": "",
    "description": "Description"
  },
  "thanks": {
    "message": "",
    "description": "Description"
  }
}
```

Typical Weblate *Component configuration*	
Dateimaske	`_locales/*/messages.json`
Einsprachige Basis-Sprachdatei	`_locales/en/messages.json`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*WebExtension JSON file*

Siehe auch:

JSON, Google chrome.i18n, Mozilla Extensions Internationalization

1.10.21 .XML resource files

Neu in Version 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.

Typical Weblate *Component configuration*	
Dateimaske	`Resources/Language.*.resx`
Einsprachige Basis-Sprachdatei	`Resources/Language.resx`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*.NET resource file*

Siehe auch:

.NET Resource files (.resx), updating-target-files, *Übersetzungsdateien bereinigen*

1.10.22 ResourceDictionary files

Neu in Version 4.13.

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

Typical Weblate *Component configuration*	
Dateimaske	`Languages/*.xaml`
Einsprachige Basis-Sprachdatei	`Language/en.xaml`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*ResourceDictionary file*

Siehe auch:

Flat XML, *Flat XML files*, updating-target-files, *Übersetzungsdateien bereinigen*

1.10.23 CSV files

Neu in Version 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 *Einsprachige Basis-Sprachdatei* when your files are monolingual (see *Bilingual and monolingual formats*).

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

Warnung: Das CSV-Format erkennt derzeit automatisch den Dialekt der CSV-Datei. In einigen Fällen kann die automatische Erkennung fehlschlagen und Sie erhalten gemischte Ergebnisse. Dies gilt insbesondere für CSV-Dateien mit Zeilenumbrüchen in den Werten. Als Abhilfe empfiehlt es sich, die Anführungszeichen wegzulassen.

Example file:

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

Typical Weblate *Component configuration* for bilingual CSV	
Dateimaske	`locale/*.csv`
Einsprachige Basis-Sprachdatei	*Empty*
Vorlage für neue Übersetzungen	`locale/en.csv`
Dateiformat	*CSV file*

Typical Weblate *Component configuration* for monolingual CSV	
Dateimaske	`locale/*.csv`
Einsprachige Basis-Sprachdatei	`locale/en.csv`
Vorlage für neue Übersetzungen	`locale/en.csv`
Dateiformat	*Simple CSV file*

Multivalue CSV file

Neu in Version 4.13.

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

Siehe auch:

CSV

1.10.24 YAML files

Neu in Version 2.9.

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

Example of a YAML file:

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

Typical Weblate *Component configuration*	
Dateimaske	`translations/messages.*.yml`
Einsprachige Basis-Sprachdatei	`translations/messages.en.yml`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*YAML file*

Siehe auch:

YAML, *Ruby YAML files*

1.10.25 Ruby YAML files

Neu in Version 2.9.

Ruby i18n YAML files with language as root node.

Example Ruby i18n YAML file:

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

Typical Weblate *Component configuration*	
Dateimaske	`translations/messages.*.yml`
Einsprachige Basis-Sprachdatei	`translations/messages.en.yml`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*Ruby YAML file*

Siehe auch:

YAML, *YAML files*

1.10.26 DTD files

Neu in Version 2.18.

Example DTD file:

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

Typical Weblate *Component configuration*	
Dateimaske	`locale/*.dtd`
Einsprachige Basis-Sprachdatei	`locale/en.dtd`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*DTD file*

Siehe auch:

Mozilla DTD format

1.10.27 Flat XML files

Neu in Version 3.9.

Example of a flat XML file:

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

Typical Weblate *Component configuration*	
Dateimaske	`locale/*.xml`
Einsprachige Basis-Sprachdatei	`locale/en.xml`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*Flat XML file*

Siehe auch:

Flat XML

1.10.28 Windows RC files

Geändert in Version 4.1: Support for Windows RC files has been rewritten.

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

Example Windows RC file:

```
LANGUAGE LANG_CZECH, SUBLANG_DEFAULT

STRINGTABLE
BEGIN
    IDS_MSG1                "Hello, world!\n"
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

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

Typical Weblate *Component configuration*	
Dateimaske	`lang/*.rc`
Einsprachige Basis-Sprachdatei	`lang/en-US.rc`
Vorlage für neue Übersetzungen	`lang/en-US.rc`
Dateiformat	*RC file*

Siehe auch:

Windows RC files

1.10.29 App-Store Metadatendateien

Neu in Version 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.

Typical Weblate *Component configuration*	
Dateimaske	`fastlane/android/metadata/*`
Einsprachige Basis-Sprachdatei	`fastlane/android/metadata/en-US`
Vorlage für neue Übersetzungen	`fastlane/android/metadata/en-US`
Dateiformat	*App store metadata files*

Hinweis: 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 *Massenbearbeitung*.

1.10.30 Subtitle files

Neu in Version 3.7.

Weblate kann verschiedene Untertiteldateien übersetzen:

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

Typical Weblate *Component configuration*	
Dateimaske	`path/*.srt`
Einsprachige Basis-Sprachdatei	`path/en.srt`
Vorlage für neue Übersetzungen	`path/en.srt`
Dateiformat	*SubRip subtitle file*

Siehe auch:

Subtitles

1.10.31 Excel Open XML

Neu in Version 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.

1.10.32 HTML files

Neu in Version 4.1.

Bemerkung: 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.

Siehe auch:

HTML

1.10.33 Textdateien

Neu in Version 4.6.

Bemerkung: 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.

Dieses Format gibt es in drei Varianten:

- Klartextdatei
- DokuWiki-Textdatei
- MediaWiki-Textdatei

Siehe auch:

Simple Text Documents

1.10.34 OpenDocument Format

Neu in Version 4.1.

Bemerkung: 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.

Siehe auch:

OpenDocument Format

1.10.35 IDML Format

Neu in Version 4.1.

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

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

1.10.36 TermBase eXchange format

Neu in Version 4.5.

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

Typical Weblate *Component configuration*	
Dateimaske	`tbx/*.tbx`
Einsprachige Basis-Sprachdatei	*Empty*
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*TermBase eXchange file*

Siehe auch:

TBX auf Wikipedia, TBX, *Glossar*

1.10.37 Stringsdict-Format

Neu in Version 4.8.

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

Ein von Apple verwendetes XML-basiertes Format, das mehrere Pluralformen einer Zeichenkette speichern kann.

Typical Weblate *Component configuration*	
Dateimaske	`Resources/*.lproj/Localizable.stringsdict`
Einsprachige Basis-Sprachdatei	`Resources/en.lproj/Localizable.stringsdict` oder `Resources/Base.lproj/Localizable.stringsdict`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*Stringsdict-Datei*

Siehe auch:

Apple iOS strings, Stringsdict File Format

1.10.38 Fluent-Format

Neu in Version 4.8.

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

Fluent ist ein einsprachiges Textformat, das sich auf asymmetrische Lokalisierung konzentriert: Eine einfache Zeichenkette in einer Sprache kann in einer anderen Sprache eine komplexe Übersetzung mit mehreren Varianten ergeben.

Typical Weblate *Component configuration*	
Dateimaske	`locales/*/messages.ftl`
Einsprachige Basis-Sprachdatei	`locales/en/messages.ftl`
Vorlage für neue Übersetzungen	*Empty*
Dateiformat	*Fluent-Datei*

Siehe auch:

Projekt Fluent Website

1.10.39 Unterstützung anderer Formate

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.

Siehe auch:

Translation Related File Formats

1.11 Integration der Versionsverwaltung

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

1.11.1 Accessing repositories

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

Accessing repositories from Hosted Weblate

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

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

Once the *weblate* user is added, you can configure *Quellcode-Repository* and *Push-URL für Repository* using the SSH protocol (for example `git@github.com:WeblateOrg/weblate.git`).

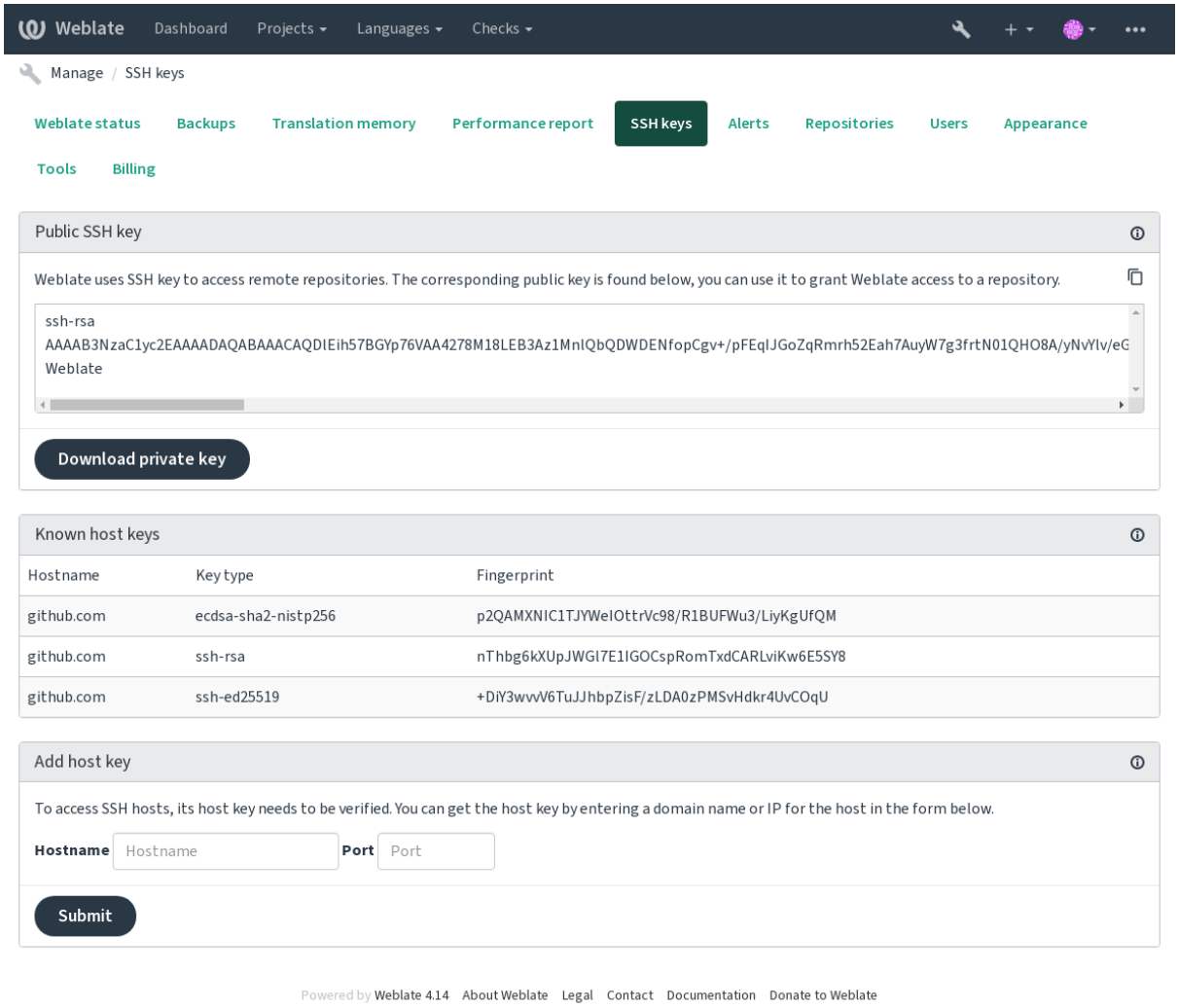
SSH repositories

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

> **Warnung:** On GitHub, each key can only be used once, see *GitHub repositories* and *Accessing repositories from Hosted Weblate*.

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

Falls eine Anpassung erforderlich ist, nehmen Sie diese über die Weblate-Adminoberfläche vor:



Weblate-SSH-Schlüssel

Der öffentliche Schlüssel von Weblate ist für alle Benutzer sichtbar, welche die Seite *Über Weblate* besuchen.

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

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

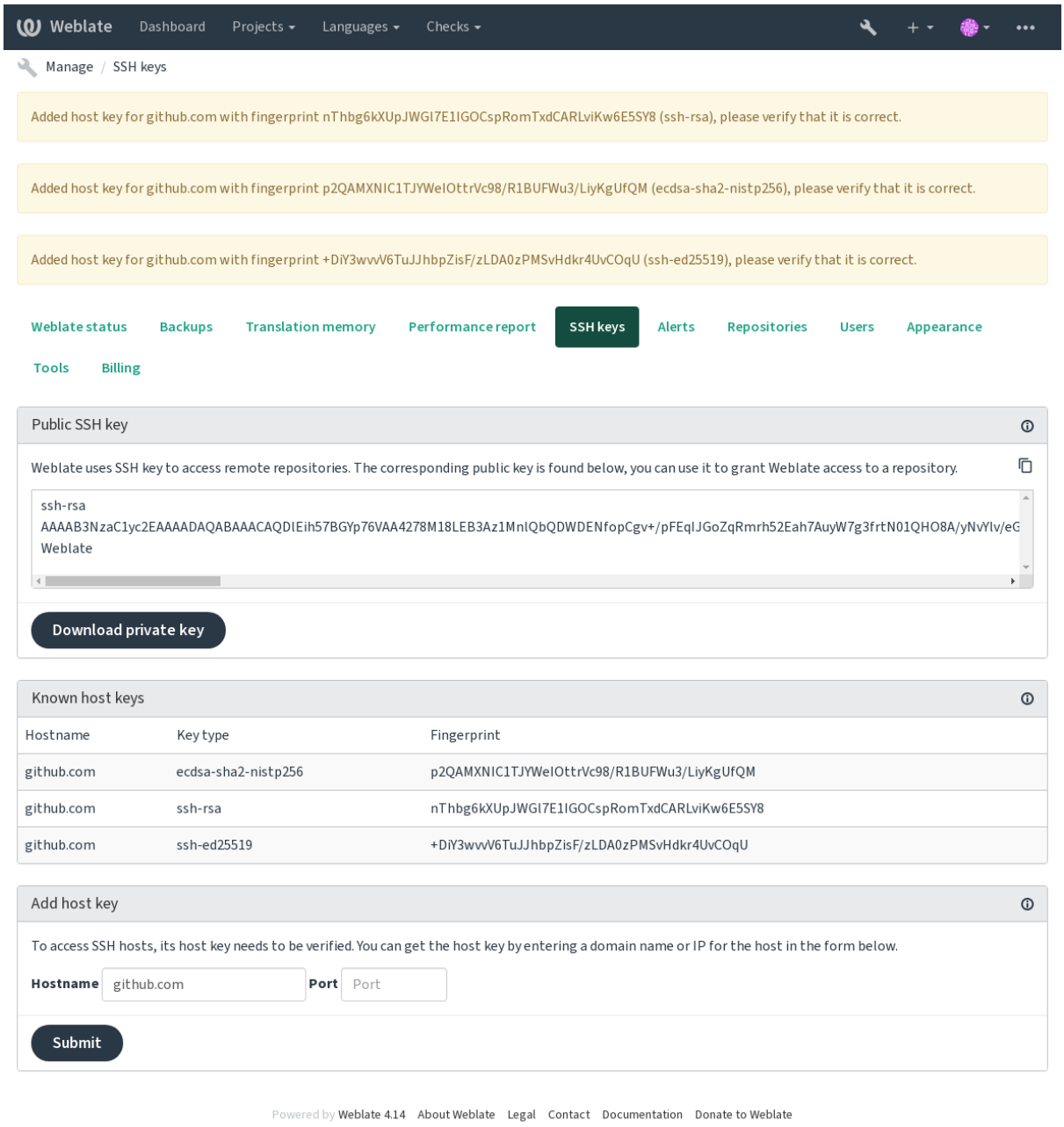
Hinweis: Make a backup of the generated private Weblate SSH key.

Verifying SSH host keys

Weblate speichert die SSH-Hostschlüssel beim ersten Zugriff automatisch und merkt sie sich für die weitere Verwendung.

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:



GitHub repositories

Der Zugriff über SSH ist möglich (siehe *SSH repositories*), aber falls Sie auf mehr als ein Repository zugreifen müssen, stoßen Sie auf eine GitHub-Beschränkung für die Verwendung von SSH-Schlüsseln (da jeder Schlüssel nur einmal verwendet werden kann).

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

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

For bigger setups, it is usually better to create a dedicated user for Weblate, assign it the public SSH key generated in Weblate (see *Weblate-SSH-Schlüssel*) 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.

Siehe auch:

Accessing repositories from Hosted Weblate

Weblate internal URLs

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

Warnung: Removing main component also removes linked components.

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

Reasons to use this:

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

HTTPS repositories

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

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

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

Using proxy

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

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

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

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

Siehe auch:

The cURL manpage, Git config documentation

1.11.2 Git

Hinweis: Weblate benötigt Git 2.12 oder neuer.

Siehe auch:

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

Git Push erzwingen

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.

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

Customizing Git configuration

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

Git remote helpers

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

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

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

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

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

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

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

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

1.11.3 GitHub-Pull-Anfragen

Neu in Version 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-Pull-Anfragen* creates pull requests. The latter is not needed for merely accessing Git repositories.

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

Siehe auch:

Pushing changes from Weblate, `GITHUB_USERNAME`, `GITHUB_TOKEN`, `GITHUB_CREDENTIALS`

1.11.4 GitLab Merge Requests

Neu in Version 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 Merge Requests* creates merge request.

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

Siehe auch:

Pushing changes from Weblate, `GITLAB_USERNAME`, `GITLAB_TOKEN`, `GITLAB_CREDENTIALS`

1.11.5 Gitea pull requests

Neu in Version 4.12.

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

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

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

Siehe auch:

Pushing changes from Weblate, `GITEA_USERNAME`, `GITEA_TOKEN`, `GITEA_CREDENTIALS`

1.11.6 Pagure Merge Requests

Neu in Version 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 Merge Requests* creates merge request.

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

Siehe auch:

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

1.11.7 Gerrit

Neu in Version 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.

1.11.8 Mercurial

Neu in Version 2.1.

Mercurial is another VCS you can use directly in Weblate.

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

Siehe auch:

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

1.11.9 Subversion

Neu in Version 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.

Bemerkung: 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.

Geändert in Version 2.19: Before this, only repositories using the standard layout were supported.

Subversion credentials

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

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

Siehe auch:

`DATA_DIR`

1.11.10 Local files

1.11.11 Git

Hinweis: Darunter verwendet es *Git*. Es erfordert die Installation von Git und ermöglicht es Ihnen, Git nativ mit einer vollständigen Historie Ihrer Übersetzungen zu verwenden.

Neu in Version 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

Neu in Version 2.6: Die REST-API ist seit Weblate 2.6 verfügbar.

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

1.12.1 Authentication and generic parameters

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

ANY /

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

Query Parameters

- **format** – Response format (overrides Accept). Possible values depends on REST framework setup, by default `json` and `api` are supported. The latter provides web browser interface for API.
- **page** – Gibt eine Seite mit paginierten Ergebnissen zurück (verwenden Sie die Felder *next* und *previous* in der Antwort, um die Navigation zu automatisieren).

Request Headers

- Accept – the response content type depends on Accept header

- Authorization – optionales Token zur Authentifizierung als `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

Statuscodes

- 200 OK – wenn die Anfrage korrekt bearbeitet wurde
- 201 Created – wenn ein neues Objekt erfolgreich erstellt wurde
- 204 No Content – wenn ein Objekt erfolgreich gelöscht wurde
- 400 Bad Request – wenn Formularparameter fehlen
- 403 Forbidden – wenn der Zugriff verweigert wird
- 429 Too Many Requests – wenn die Drosselung in Kraft ist

Authentifizierungstoken

Geändert in Version 4.10: Projektspezifische Token wurden in der Version 4.10 eingeführt.

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

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

Beispiele für die Authentifizierung

Beispielanfrage:

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

Beispielantwort:

```
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
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
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-Beispiel:

```
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*).

Formularanfrage-Beispiel:

```
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-Anfrage-Beispiel:

```
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-Beispiel:

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

CURL-JSON-Beispiel:

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

API rate limiting

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

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

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

The status of rate limiting is reported in following headers:

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

Geändert in Version 4.1: Added ratelimiting status headers.

Siehe auch:

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

1.12.2 API Entry Point

GET /api/

The API root entry point.

Beispielanfrage:

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

Beispielantwort:

```
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 Benutzer

Neu in Version 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.

Siehe auch:

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

POST /api/users/

Erstellt einen neuen Benutzer.

Parameter

- **username** (*string*) – Benutzername
- **full_name** (*string*) – Vollständiger Name des Benutzers
- **email** (*string*) – E-Mail-Adresse des Benutzers
- **is_superuser** (*boolean*) – Ist der Benutzer Superuser? (Optional)
- **is_active** (*boolean*) – Ist der Benutzer aktiv? (Optional)
- **is_bot** (*boolean*) – Is user bot? (optional) (used for project scoped tokens)

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

Gibt Informationen über Benutzer zurück.

Parameter

- **username** (*string*) – Name des Benutzers

Response JSON Object

- **username** (*string*) – Benutzernamen eines Benutzers
- **full_name** (*string*) – Vollständiger Name eines Benutzers
- **email** (*string*) – E-Mail-Adresse eines Benutzers
- **is_superuser** (*boolean*) – Ob der Benutzer ein Superuser ist
- **is_active** (*boolean*) – Ob der Benutzer aktiv ist
- **is_bot** (*boolean*) – whether the user is bot (used for project scoped tokens)
- **date_joined** (*string*) – Datum der Erstellung des Benutzers
- **groups** (*array*) – link to associated groups; see *GET /api/groups/(int:id)/*

Example JSON data:

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

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
↪"
}
```

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

Ändert die Benutzerparameter.

Parameter

- **username** (*string*) – Name des Benutzers

Response JSON Object

- **username** (*string*) – Benutzernamen eines Benutzers
- **full_name** (*string*) – Vollständiger Name eines Benutzers
- **email** (*string*) – E-Mail-Adresse eines Benutzers
- **is_superuser** (*boolean*) – Ob der Benutzer ein Superuser ist
- **is_active** (*boolean*) – Ob der Benutzer aktiv ist
- **is_bot** (*boolean*) – whether the user is bot (used for project scoped tokens)
- **date_joined** (*string*) – Datum der Erstellung des Benutzers

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

Ändert die Benutzerparameter.

Parameter

- **username** (*string*) – Name des Benutzers

Response JSON Object

- **username** (*string*) – Benutzernamen eines Benutzers
- **full_name** (*string*) – Vollständiger Name eines Benutzers
- **email** (*string*) – E-Mail-Adresse eines Benutzers
- **is_superuser** (*boolean*) – Ob der Benutzer ein Superuser ist
- **is_active** (*boolean*) – Ob der Benutzer aktiv ist
- **is_bot** (*boolean*) – whether the user is bot (used for project scoped tokens)
- **date_joined** (*string*) – Datum der Erstellung des Benutzers

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

Löscht alle Benutzerinformationen und markiert den Benutzer als inaktiv.

Parameter

- **username** (*string*) – Name des Benutzers

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

Zuordnen von Gruppen zu einem Benutzer.

Parameter

- **username** (*string*) – Name des Benutzers

Form Parameters

- **string group_id** – Die eindeutige Gruppen-ID

DELETE /api/users/(**str:** *username*)**/groups/**

Neu in Version 4.13.1.

Remove user from a group.

Parameter

- **username** (*string*) – Name des Benutzers

Form Parameters

- **string group_id** – Die eindeutige Gruppen-ID

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

Statistik eines Benutzers auflisten.

Parameter

- **username** (*string*) – Name des Benutzers

Response JSON Object

- **translated** (*int*) – Anzahl der Übersetzungen des Benutzers
- **suggested** (*int*) – Anzahl der Vorschläge des Benutzers
- **uploaded** (*int*) – Anzahl der Uploads des Benutzers
- **commented** (*int*) – Anzahl der Kommentare des Benutzers
- **languages** (*int*) – Anzahl der Sprachen, die der Benutzer übersetzen kann

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

Liste der Abonnements eines Benutzers.

Parameter

- **username** (*string*) – Name des Benutzers

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

Verknüpfen Sie Abonnements mit einem Benutzer.

Parameter

- **username** (*string*) – Name des Benutzers

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.

Parameter

- **username** (*string*) – Name des Benutzers
- **subscription_id** (*int*) – ID der registrierten Benachrichtigung

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

Edit a subscription associated with a user.

Parameter

- **username** (*string*) – Name des Benutzers
- **subscription_id** (*int*) – ID der registrierten Benachrichtigung

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.

Parameter

- **username** (*string*) – Name des Benutzers
- **subscription_id** (*int*) – ID der registrierten Benachrichtigung

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.

Parameter

- **username** (*string*) – Name des Benutzers
- **subscription_id** – Name of notification registered
- **subscription_id** – int

1.12.4 Gruppen

Neu in Version 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.

Siehe auch:

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

POST /api/groups/

Erstellt eine neue Gruppe.

Parameter

- **name** (*string*) – Gruppenname
- **project_selection** (*int*) – Group of project selection from given options
- **language_selection** (*int*) – Group of languages selected from given options
- **defining_project** (*str*) – link to the defining project, used for *Verwaltung der Zugriffssteuerung nach Projekt*; see *GET /api/projects/(string:project)/*

GET /api/groups/(**int:** *id*)**/**

Returns information about group.

Parameter

- **id** (*int*) – Group's ID

Response JSON Object

- **name** (*string*) – Name einer Gruppe
- **project_selection** (*int*) – integer corresponding to group of projects
- **language_selection** (*int*) – integer corresponding to group of languages
- **roles** (*array*) – link to associated roles; see `GET /api/roles/(int:id)/`
- **projects** (*array*) – link to associated projects; see `GET /api/projects/(string:project)/`
- **components** (*array*) – link to associated components; see `GET /api/components/(string:project)/(string:component)/`
- **componentlists** (*array*) – link to associated componentlist; see `GET /api/component-lists/(str:slug)/`
- **defining_project** (*str*) – link to the defining project, used for *Verwaltung der Zugriffssteuerung nach Projekt*; see `GET /api/projects/(string:project)/`

Example JSON data:

```
{
    "name": "Guests",
    "defining_project": null,
    "project_selection": 3,
    "language_selection": 1,
    "url": "http://example.com/api/groups/1/",
    "roles": [
        "http://example.com/api/roles/1/",
        "http://example.com/api/roles/2/"
    ],
    "languages": [
        "http://example.com/api/languages/en/",
        "http://example.com/api/languages/cs/",
    ],
    "projects": [
        "http://example.com/api/projects/demo1/",
        "http://example.com/api/projects/demo/"
    ],
    "componentlist": "http://example.com/api/component-lists/new/",
    "components": [
        "http://example.com/api/components/demo/weblate/"
    ]
}
```

PUT /api/groups/(int: *id*)/

Changes the group parameters.

Parameter

- **id** (*int*) – Group's ID

Response JSON Object

- **name** (*string*) – Name einer Gruppe
- **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.

Parameter

- **id** (*int*) – Group's ID

Response JSON Object

- **name** (*string*) – Name einer Gruppe
- **project_selection** (*int*) – integer corresponding to group of projects
- **language_selection** (*int*) – integer corresponding to group of languages

DELETE /api/groups/(int: *id*)/

Löscht die Gruppe.

Parameter

- **id** (*int*) – Group's ID

POST /api/groups/(int: *id*)/roles/

Zuordnen von Rollen zu einer Gruppe.

Parameter

- **id** (*int*) – Group's ID

Form Parameters

- **string role_id** – Die eindeutige Rollen-ID

POST /api/groups/(int: *id*)/components/

Associate components with a group.

Parameter

- **id** (*int*) – Group's ID

Form Parameters

- **string component_id** – Die eindeutige Komponenten-ID

DELETE /api/groups/(int: *id*)/components/
int: *component_id*

Komponente aus einer Gruppe löschen.

Parameter

- **id** (*int*) – Group's ID
- **component_id** (*int*) – Die eindeutige Komponenten-ID

POST /api/groups/(int: *id*)/projects/

Associate projects with a group.

Parameter

- **id** (*int*) – Group's ID

Form Parameters

- **string project_id** – Die eindeutige Projekt-ID

DELETE /api/groups/(int: *id*)/projects/
int: *project_id*

Projekt aus einer Gruppe löschen.

Parameter

- **id** (*int*) – Group's ID
- **project_id** (*int*) – Die eindeutige Projekt-ID

POST /api/groups/(int: *id*)/languages/

Associate languages with a group.

Parameter

- **id** (*int*) – Group's ID

Form Parameters

- **string language_code** – Der eindeutige Sprachcode

DELETE /api/groups/(int: *id*)/languages/
string: *language_code*

Sprache aus einer Gruppe löschen.

Parameter

- **id** (*int*) – Group's ID
- **language_code** (*string*) – Der eindeutige Sprachcode

POST /api/groups/(int: *id*)/componentlists/

Associate componentlists with a group.

Parameter

- **id** (*int*) – Group's ID

Form Parameters

- **string component_list_id** – Die eindeutige Komponentenlisten-ID

DELETE /api/groups/(int: *id*)/componentlists/
int: *component_list_id*

Komponentenliste aus einer Gruppe löschen.

Parameter

- **id** (*int*) – Group's ID
- **component_list_id** (*int*) – Die eindeutige Komponentenlisten-ID

1.12.5 Rollen

GET /api/roles/

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

Siehe auch:

Roles object attributes are documented at *GET /api/roles/(int:id)/*.

POST /api/roles/

Creates a new role.

Parameter

- **name** (*string*) – Role name
- **permissions** (*array*) – List of codenames of permissions

GET /api/roles/(int: *id*)/

Returns information about a role.

Parameter

- **id** (*int*) – Role ID

Response JSON Object

- **name** (*string*) – Role name
- **permissions** (*array*) – list of codenames of permissions

Example JSON data:

```
{
    "name": "Access repository",
    "permissions": [
        "vcs.access",
        "vcs.view"
    ],
    "url": "http://example.com/api/roles/1/",
}
```

PUT /api/roles/(int: *id*)/

Changes the role parameters.

Parameter

- **id** (*int*) – Role's ID

Response JSON Object

- **name** (*string*) – Role name
- **permissions** (*array*) – list of codenames of permissions

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

Changes the role parameters.

Parameter

- **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.

Parameter

- **id** (*int*) – Role's ID

1.12.6 Sprachen

GET /api/languages/

Returns a list of all languages.

Siehe auch:

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

POST /api/languages/

Creates a new language.

Parameter

- **code** (*string*) – Sprachenname
- **name** (*string*) – Sprachenname
- **direction** (*string*) – Leserichtung
- **population** (*int*) – Anzahl der Sprecher

- **plural** (*object*) – Language plural formula and number

GET /api/languages/(string: *language*)/

Returns information about a language.

Parameter

- **language** (*string*) – Sprachkürzel

Response JSON Object

- **code** (*string*) – Sprachkürzel
- **direction** (*string*) – Leserichtung
- **plural** (*object*) – Object of language plural information
- **aliases** (*array*) – Array of aliases for language

Request JSON Object

- **population** (*int*) – Anzahl der Sprecher

Example JSON data:

```
{
    "code": "en",
    "direction": "ltr",
    "name": "English",
    "population": 159034349015,
    "plural": {
        "id": 75,
        "source": 0,
        "number": 2,
        "formula": "n != 1",
        "type": 1
    },
    "aliases": [
        "english",
        "en_en",
        "base",
        "source",
        "eng"
    ],
    "url": "http://example.com/api/languages/en/",
    "web_url": "http://example.com/languages/en/",
    "statistics_url": "http://example.com/api/languages/en/statistics/"
}
```

PUT /api/languages/(string: *language*)/

Changes the language parameters.

Parameter

- **language** (*string*) – Language's code

Request JSON Object

- **name** (*string*) – Sprachenname
- **direction** (*string*) – Leserichtung
- **population** (*int*) – Anzahl der Sprecher
- **plural** (*object*) – Language plural details

PATCH /api/languages/(string: *language*)/

Changes the language parameters.

Parameter

- **language** (*string*) – Language's code

Request JSON Object

- **name** (*string*) – Sprachenname
- **direction** (*string*) – Leserichtung
- **population** (*int*) – Anzahl der Sprecher
- **plural** (*object*) – Language plural details

DELETE /api/languages/(string: *language*)/

Löscht die Sprache.

Parameter

- **language** (*string*) – Language's code

GET /api/languages/(string: *language*)/statistics/

Returns statistics for a language.

Parameter

- **language** (*string*) – Sprachkürzel

Response JSON Object

- **total** (*int*) – total number of strings
- **total_words** (*int*) – total number of words
- **last_change** (*timestamp*) – last changes in the language
- **recent_changes** (*int*) – total number of changes
- **translated** (*int*) – number of translated strings
- **translated_percent** (*float*) – percentage of translated strings
- **translated_words** (*int*) – number of translated words
- **translated_words_percent** (*int*) – percentage of translated words
- **translated_chars** (*int*) – number of translated characters
- **translated_chars_percent** (*int*) – percentage of translated characters
- **total_chars** (*int*) – number of total characters
- **fuzzy** (*int*) – Anzahl der fragwürdigen (zur Bearbeitung markierten) Zeichenfolgen
- **fuzzy_percent** (*int*) – percentage of fuzzy (marked for edit) strings
- **failing** (*int*) – number of failing strings
- **failing** – percentage of failing strings

1.12.7 Projekte

GET /api/projects/

Returns a list of all projects.

Siehe auch:

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

POST /api/projects/

Neu in Version 3.9.

Creates a new project.

Parameter

- **name** (*string*) – Projektname
- **slug** (*string*) – Project slug
- **web** (*string*) – Projektseite

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

Returns information about a project.

Parameter

- **project** (*string*) – Kurzer Projekt-URL

Response JSON Object

- **name** (*string*) – Projektname
- **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*) – *Begutachtung aktivieren*
- **source_review** (*boolean*) – *Quellenüberprüfung aktivieren*
- **set_language_team** (*boolean*) – *Kopfzeile „Language-Team“ setzen*
- **enable_hooks** (*boolean*) – *Hooks aktivieren*
- **instructions** (*string*) – *Übersetzungsanweisungen*
- **language_aliases** (*string*) – *Sprachaliasnamen*

Example JSON data:

```
{
    "name": "Hello",
    "slug": "hello",
    "url": "http://example.com/api/projects/hello/",
    "web": "https://weblate.org/",
    "web_url": "http://example.com/projects/hello/"
}
```

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

Neu in Version 4.3.

Edit a project by a PATCH request.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

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

Neu in Version 4.3.

Edit a project by a PUT request.

Parameter

- **project** (*string*) – Kurzer Projekt-URL

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

Neu in Version 3.9.

Deletes a project.

Parameter

- **project** (*string*) – Kurzer Projekt-URL

GET /api/projects/(string: *project*)/changes/

Returns a list of project changes. This is essentially a project scoped `GET /api/changes/` accepting same params.

Parameter

- **project** (*string*) – Kurzer Projekt-URL

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/`.

Parameter

- **project** (*string*) – Kurzer Projekt-URL

Response JSON Object

- **needs_commit** (*boolean*) – whether there are any pending changes to commit
- **needs_merge** (*boolean*) – whether there are any upstream changes to merge
- **needs_push** (*boolean*) – whether there are any local changes to push

Example JSON data:

```
{
    "needs_commit": true,
    "needs_merge": false,
    "needs_push": true
}
```

POST /api/projects/(string: *project*)/repository/

Performs given operation on the VCS repository.

Parameter

- **project** (*string*) – Kurzer Projekt-URL

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-Beispiel:

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

JSON-Anfrage-Beispiel:

```
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.

Parameter

- **project** (*string*) – Kurzer Projekt-URL

Response JSON Object

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

POST /api/projects/(string: *project*)/components/

Neu in Version 3.9.

Geändert in Version 4.3: The `zipfile` and `docfile` parameters are now accepted for VCS-less components, see *Local files*.

Geändert in Version 4.6: The cloned repositories are now automatically shared within a project using *Weblate internal URLs*. Use `disable_autoshare` to turn off this.

Creates translation components in the given project.

Hinweis: Use *Weblate internal URLs* when creating multiple components from a single VCS repository.

Bemerkung: Most of the component creation happens in the background. Check the `task_url` attribute of created component and follow the progress there.

Parameter

- **project** (*string*) – Kurzer Projekt-URL

Form Parameters

- **file zipfile** – ZIP file to upload into Weblate for translations initialization
- **file docfile** – Dokument zum Übersetzen
- **boolean disable_autoshare** – Disables automatic repository sharing via *Weblate internal URLs*.

Request JSON Object

- **object** – Component parameters, see *GET /api/components/(string:project)/(string:component)/*

Response JSON Object

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

JSON can not be used when uploading the files using the `zipfile` and `docfile` parameters. The data has to be uploaded as *multipart/form-data*.

CURL form request example:

```
curl \
    --form docfile=@strings.html \
    --form name=Weblate \
    --form slug=weblate \
    --form file_format=html \
    --form new_lang=add \
    -H "Authorization: Token TOKEN" \
    http://example.com/api/projects/hello/components/
```

CURL JSON request example:

```
curl \
    --data-binary '{
        "branch": "main",
        "file_format": "po",
        "filemask": "po/*.po",
        "name": "Weblate",
        "slug": "weblate",
        "repo": "https://github.com/WeblateOrg/hello.git",
        "template": "",
        "new_base": "po/hello.pot",
        "vcs": "git"
    }' \
    -H "Content-Type: application/json" \
    -H "Authorization: Token TOKEN" \
    http://example.com/api/projects/hello/components/
```

JSON request to create a new component from Git:

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

{
    "branch": "main",
    "file_format": "po",
    "filemask": "po/*.po",
    "name": "Weblate",
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    "slug": "weblate",
    "repo": "https://github.com/WeblateOrg/hello.git",
    "template": "",
    "new_base": "po/hello.pot",
    "vcs": "git"
}
```

JSON request to create a new component from another one:

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

{
    "file_format": "po",
    "filemask": "po/*.po",
    "name": "Weblate",
    "slug": "weblate",
    "repo": "weblate://weblate/hello",
    "template": "",
    "new_base": "po/hello.pot",
    "vcs": "git"
}
```

JSON response example:

```
HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, POST, HEAD, OPTIONS

{
    "branch": "main",
    "file_format": "po",
    "filemask": "po/*.po",
    "git_export": "",
    "license": "",
    "license_url": "",
    "name": "Weblate",
    "slug": "weblate",
    "project": {
        "name": "Hello",
        "slug": "hello",
        "source_language": {
            "code": "en",
            "direction": "ltr",
             "population": 159034349015,
            "name": "English",
            "url": "http://example.com/api/languages/en/",
            "web_url": "http://example.com/languages/en/"
        },
        "url": "http://example.com/api/projects/hello/",
        "web": "https://weblate.org/",
        "web_url": "http://example.com/projects/hello/"
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    },
    "repo": "file:///home/nijel/work/weblate-hello",
    "template": "",
    "new_base": "",
    "url": "http://example.com/api/components/hello/weblate/",
    "vcs": "git",
    "web_url": "http://example.com/projects/hello/weblate/"
}
```

GET /api/projects/(string: *project*)/languages/

Returns paginated statistics for all languages within a project.

Neu in Version 3.8.

Parameter

- **project** (*string*) – Kurzer Projekt-URL

Response JSON Object

- **results** (*array*) – array of translation statistics objects
- **language** (*string*) – language name
- **code** (*string*) – Sprachcode
- **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.

Neu in Version 3.8.

Parameter

- **project** (*string*) – Kurzer Projekt-URL

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 Komponenten

Hinweis: Use `POST /api/projects/(string:project)/components/` to create new components.

GET /api/components/

Returns a list of translation components.

Siehe auch:

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.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

Response JSON Object

- **project** (*object*) – the translation project; see `GET /api/projects/(string:project)/`
- **name** (*string*) – *Name der Komponente*
- **slug** (*string*) – *Component slug*
- **vcs** (*string*) – *Versionsverwaltung*
- **repo** (*string*) – *Quellcode-Repository*
- **git_export** (*string*) – *Exportierte Paketquelladresse*
- **branch** (*string*) – *Repository-Zweig*
- **push_branch** (*string*) – *Push Branch*
- **filemask** (*string*) – *Dateimaske*
- **template** (*string*) – *Einsprachige Basis-Sprachdatei*
- **edit_template** (*string*) – *Basisdatei bearbeiten*
- **intermediate** (*string*) – *Zwischensprachedatei*
- **new_base** (*string*) – *Vorlage für neue Übersetzungen*
- **file_format** (*string*) – *Dateiformat*
- **license** (*string*) – *Lizenzierung der Übersetzung*
- **agreement** (*string*) – *Mitwirkungsvereinbarung*
- **new_lang** (*string*) – *Neue Übersetzung hinzufügen*
- **language_code_style** (*string*) – *Stil des Sprachcodes*
- **source_language** (*object*) – source language object; see `GET /api/languages/(string:language)/`
- **push** (*string*) – *Push-URL für Repository*
- **check_flags** (*string*) – *Übersetzungsmarkierungen*
- **priority** (*string*) – *Priorität*
- **enforced_checks** (*string*) – *Erzwungene Qualitätsprüfungen*

- **restricted** (*string*) – *Restricted access*
- **repoweb** (*string*) – *Paketquellnavigator*
- **report_source_bugs** (*string*) – *Adresse für Fehlerberichte bei Ausgangszeichenketten*
- **merge_style** (*string*) – *Git-Strategie*
- **commit_message** (*string*) – *Commit, add, delete, merge, add-on, and merge request messages*
- **add_message** (*string*) – *Commit, add, delete, merge, add-on, and merge request messages*
- **delete_message** (*string*) – *Commit, add, delete, merge, add-on, and merge request messages*
- **merge_message** (*string*) – *Commit, add, delete, merge, add-on, and merge request messages*
- **addon_message** (*string*) – *Commit, add, delete, merge, add-on, and merge request messages*
- **pull_message** (*string*) – *Commit, add, delete, merge, add-on, and merge request messages*
- **allow_translation_propagation** (*string*) – *Verbreitung von Übersetzungen erlauben*
- **enable_suggestions** (*string*) – *Vorschläge aktivieren*
- **suggestion_voting** (*string*) – *Abstimmen über Vorschläge*
- **suggestion_autoaccept** (*string*) – *Vorschläge automatisch annehmen*
- **push_on_commit** (*string*) – *Bei Commit gleichzeitig Pushen*
- **commit_pending_age** (*string*) – *Alter der Änderungen, das erreicht sein muss, bevor ein Commit erfolgt*
- **auto_lock_error** (*string*) – *Sperre bei Fehler*
- **language_regex** (*string*) – *Sprachen-Filter*
- **variant_regex** (*string*) – *Varianten regulärer Ausdruck*
- **repository_url** (*string*) – URL to repository status; see `GET /api/components/(string:project)/(string:component)/repository/`
- **translations_url** (*string*) – URL to translations list; see `GET /api/components/(string:project)/(string:component)/translations/`
- **lock_url** (*string*) – URL to lock status; see `GET /api/components/(string:project)/(string:component)/lock/`
- **changes_list_url** (*string*) – URL to changes list; see `GET /api/components/(string:project)/(string:component)/changes/`
- **task_url** (*string*) – URL to a background task (if any); see `GET /api/tasks/(str:uuid)/`

Example JSON data:

```
{
    "branch": "main",
    "file_format": "po",
    "filemask": "po/*.po",
    "git_export": "",
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    "license": "",
    "license_url": "",
    "name": "Weblate",
    "slug": "weblate",
    "project": {
        "name": "Hello",
        "slug": "hello",
        "source_language": {
            "code": "en",
            "direction": "ltr",
             "population": 159034349015,
            "name": "English",
            "url": "http://example.com/api/languages/en/",
            "web_url": "http://example.com/languages/en/"
        },
        "url": "http://example.com/api/projects/hello/",
        "web": "https://weblate.org/",
        "web_url": "http://example.com/projects/hello/"
    },
    "source_language": {
        "code": "en",
        "direction": "ltr",
        "population": 159034349015,
        "name": "English",
        "url": "http://example.com/api/languages/en/",
        "web_url": "http://example.com/languages/en/"
    },
    "repo": "file:///home/nijel/work/weblate-hello",
    "template": "",
    "new_base": "",
    "url": "http://example.com/api/components/hello/weblate/",
    "vcs": "git",
    "web_url": "http://example.com/projects/hello/weblate/"
}
```

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

Edit a component by a PATCH request.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL
- **source_language** (*string*) – Project source language code (optional)

Request JSON Object

- **name** (*string*) – name of component
- **slug** (*string*) – slug of component
- **repo** (*string*) – VCS repository URL

CURL-Beispiel:

```
curl \
    --data-binary '{"name": "new name"}' \
    -H "Content-Type: application/json" \
    -H "Authorization: Token TOKEN" \
    PATCH http://example.com/api/projects/hello/components/
```

JSON-Anfrage-Beispiel:

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

{
    "name": "new name"
}
```

JSON response example:

```
HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, POST, HEAD, OPTIONS

{
    "branch": "main",
    "file_format": "po",
    "filemask": "po/*.po",
    "git_export": "",
    "license": "",
    "license_url": "",
    "name": "new name",
    "slug": "weblate",
    "project": {
        "name": "Hello",
        "slug": "hello",
        "source_language": {
            "code": "en",
            "direction": "ltr",
            "population": 159034349015,
            "name": "English",
            "url": "http://example.com/api/languages/en/",
            "web_url": "http://example.com/languages/en/"
        },
        "url": "http://example.com/api/projects/hello/",
        "web": "https://weblate.org/",
        "web_url": "http://example.com/projects/hello/"
    },
    "repo": "file:///home/nijel/work/weblate-hello",
    "template": "",
    "new_base": "",
    "url": "http://example.com/api/components/hello/weblate/",
    "vcs": "git",
    "web_url": "http://example.com/projects/hello/weblate/"
}
```

PUT /api/components/(string: *project*)/string: *component*/

Edit a component by a PUT request.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

Request JSON Object

- **branch** (*string*) – VCS repository branch
- **file_format** (*string*) – file format of translations
- **filemask** (*string*) – mask of translation files in the repository
- **name** (*string*) – name of component
- **slug** (*string*) – slug of component
- **repo** (*string*) – VCS repository URL
- **template** (*string*) – base file for monolingual translations
- **new_base** (*string*) – base file for adding new translations
- **vcs** (*string*) – Versionsverwaltung

DELETE /api/components/(string: *project*)/string: *component*/

Neu in Version 3.9.

Deletes a component.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

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.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

Response JSON Object

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

GET /api/components/(string: *project*)/string: *component*/file/

Neu in Version 4.9.

Downloads all available translations associated with the component as an archive file using the requested format.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

Query Parameters

- **format** (*string*) – The archive format to use; If not specified, defaults to `zip`; Supported formats: `zip`

GET /api/components/(string: *project*)/string: *component*/screenshots/

Gibt eine Liste von Komponenten-Bildschirmfotos zurück.

Parameter

- **project** (*string*) – Kurzer Projekt-URL

- **component** (*string*) – Kurzer Komponenten-URL

Response JSON Object

- **results** (*array*) – Reihe von Komponenten-Bildschirmfotos; siehe *GET /api/screenshots/(int:id)/*

GET /api/components/(string: *project*)/string: *component*/lock/

Returns component lock status.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

Response JSON Object

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

Example JSON data:

```
{
    "locked": false
}
```

POST /api/components/(string: *project*)/string: *component*/lock/

Sets component lock status.

Response is same as *GET /api/components/(string:project)/(string:component)/lock/*.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

Request JSON Object

- **lock** – Boolean whether to lock or not.

CURL-Beispiel:

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

JSON-Anfrage-Beispiel:

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

{"lock": true}
```

JSON response example:

```
HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, POST, HEAD, OPTIONS

{"locked":true}
```

GET /api/components/(string: *project*)/(string: *component*)/repository/

Returns information about VCS repository status.

The response is same as for *GET /api/projects/(string:project)/repository/*.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

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.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

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-Beispiel:

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

JSON-Anfrage-Beispiel:

```
POST /api/components/hello/weblate/repository/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
Authorization: Token TOKEN
Content-Length: 20

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

JSON response example:

```
HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, POST, HEAD, OPTIONS

{"result":true}
```

GET /api/components/(string: *project*)/string: *component*/monolingual_base/

Downloads base file for monolingual translations.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

GET /api/components/(string: *project*)/string: *component*/new_template/

Downloads template file for new translations.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

GET /api/components/(string: *project*)/string: *component*/translations/

Returns a list of translation objects in the given component.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

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.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

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-Beispiel:

```
curl \
    -d language_code=cs \
    -H "Authorization: Token TOKEN" \
    http://example.com/api/projects/hello/components/
```

JSON-Anfrage-Beispiel:

```
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",
        "population": 1303174280
        "name": "Czech",
        "url": "http://example.com/api/languages/cs/",
        "web_url": "http://example.com/languages/cs/"
    },
    "language_code": "cs",
    "id": 125,
    "last_author": null,
    "last_change": null,
    "share_url": "http://example.com/engage/hello/cs/",
    "total": 4,
    "total_words": 15,
    "translate_url": "http://example.com/translate/hello/weblate/cs/",
    "translated": 0,
    "translated_percent": 0.0,
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    "translated_words": 0,
    "url": "http://example.com/api/translations/hello/weblate/cs/",
    "web_url": "http://example.com/projects/hello/weblate/cs/"
}
```

GET /api/components/(string: *project*)/string: *component*/statistics/

Returns paginated statistics for all translations within component.

Neu in Version 2.7.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

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/

Gibt Projekte zurück, die mit einer Komponente verknüpft sind.

Neu in Version 4.5.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

Response JSON Object

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

POST /api/components/(string: *project*)/string: *component*/links/

Projekt mit einer Komponente verknüpfen.

Neu in Version 4.5.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL

Form Parameters

- **string project_slug** – Project slug

DELETE /api/components/(string: *project*)/string: *component*/links/string: *project_slug*/

Aufhebung der Zuordnung eines Projekts zu einer Komponente.

Neu in Version 4.5.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL
- **project_slug** (*string*) – Slug des zu entfernenden Projekts

1.12.9 Übersetzungen

GET /api/translations/

Returns a list of translations.

Siehe auch:

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

GET /api/translations/(string: *project***)/string:** *component***/string:** *language***/**

Returns information about a translation.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL
- **language** (*string*) – Translation language code

Response JSON Object

- **component** (*object*) – component object; see *GET /api/components/(string:project)/(string:component)/*
- **failing_checks** (*int*) – Anzahl der Zeichenketten, die nicht geprüft wurden
- **failing_checks_percent** (*float*) – Prozentsatz der Zeichenketten, welche die Qualitätsprüfungen nicht bestanden haben
- **failing_checks_words** (*int*) – Anzahl der Wörter mit nicht bestandener Qualitätsprüfungen
- **filename** (*string*) – translation filename
- **fuzzy** (*int*) – Anzahl der fragwürdigen (zur Bearbeitung markierten) Zeichenfolgen
- **fuzzy_percent** (*float*) – percentage of fuzzy (marked for edit) strings
- **fuzzy_words** (*int*) – number of words in fuzzy (marked for edit) strings
- **have_comment** (*int*) – number of strings with comment
- **have_suggestion** (*int*) – number of strings with suggestion
- **is_template** (*boolean*) – ob die Übersetzung eine einsprachige Basis hat
- **language** (*object*) – source language object; see *GET /api/languages/(string:language)/*
- **language_code** (*string*) – language code used in the repository; this can be different from language code in the language object
- **last_author** (*string*) – name of last author
- **last_change** (*timestamp*) – last change timestamp
- **revision** (*string*) – revision hash for the file
- **share_url** (*string*) – URL for sharing leading to engagement page
- **total** (*int*) – total number of strings
- **total_words** (*int*) – total number of words
- **translate_url** (*string*) – URL for translating
- **translated** (*int*) – number of translated strings
- **translated_percent** (*float*) – percentage of translated strings

- **translated_words** (*int*) – number of translated words
- **repository_url** (*string*) – URL to repository status; see *GET /api/translations/(string:project)/(string:component)/(string:language)/repository/*
- **file_url** (*string*) – URL to file object; see *GET /api/translations/(string:project)/(string:component)/(string:language)/file/*
- **changes_list_url** (*string*) – URL to changes list; see *GET /api/translations/(string:project)/(string:component)/(string:language)/changes/*
- **units_list_url** (*string*) – URL to strings list; see *GET /api/translations/(string:project)/(string:component)/(string:language)/units/*

Example JSON data:

```
{
    "component": {
        "branch": "main",
        "file_format": "po",
        "filemask": "po/*.po",
        "git_export": "",
        "license": "",
        "license_url": "",
        "name": "Weblate",
        "new_base": "",
        "project": {
            "name": "Hello",
            "slug": "hello",
            "source_language": {
                "code": "en",
                "direction": "ltr",
                "population": 159034349015,
                "name": "English",
                "url": "http://example.com/api/languages/en/",
                "web_url": "http://example.com/languages/en/"
            },
            "url": "http://example.com/api/projects/hello/",
            "web": "https://weblate.org/",
            "web_url": "http://example.com/projects/hello/"
        },
        "repo": "file:///home/nijel/work/weblate-hello",
        "slug": "weblate",
        "template": "",
        "url": "http://example.com/api/components/hello/weblate/",
        "vcs": "git",
        "web_url": "http://example.com/projects/hello/weblate/"
    },
    "failing_checks": 3,
    "failing_checks_percent": 75.0,
    "failing_checks_words": 11,
    "filename": "po/cs.po",
    "fuzzy": 0,
    "fuzzy_percent": 0.0,
    "fuzzy_words": 0,
    "have_comment": 0,
    "have_suggestion": 0,
    "is_template": false,
    "language": {
        "code": "cs",
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
        "direction": "ltr",
        "population": 1303174280
        "name": "Czech",
        "url": "http://example.com/api/languages/cs/",
        "web_url": "http://example.com/languages/cs/"
    },
    "language_code": "cs",
    "last_author": "Weblate Admin",
    "last_change": "2016-03-07T10:20:05.499",
    "revision": "7ddfafe6daaf57fc8654cc852ea6be212b015792",
    "share_url": "http://example.com/engage/hello/cs/",
    "total": 4,
    "total_words": 15,
    "translate_url": "http://example.com/translate/hello/weblate/cs/",
    "translated": 4,
    "translated_percent": 100.0,
    "translated_words": 15,
    "url": "http://example.com/api/translations/hello/weblate/cs/",
    "web_url": "http://example.com/projects/hello/weblate/cs/"
}
```

DELETE /api/translations/(string: *project*)/(string: *component*)/(string: *language*)/

Neu in Version 3.9.

Deletes a translation.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL
- **language** (*string*) – Translation language code

GET /api/translations/(string: *project*)/(string: *component*)/(string: *language*)/changes/

Returns a list of translation changes. This is essentially a translations-scoped *GET /api/changes/* accepting the same parameters.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL
- **language** (*string*) – Translation language code

Response JSON Object

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

GET /api/translations/(string: *project*)/(string: *component*)/(string: *language*)/units/

Returns a list of translation units.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL
- **language** (*string*) – Translation language code
- **q** (*string*) – Search query string *Searching* (optional)

Response JSON Object

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

POST /api/translations/(string: *project*)/string: *component*/string: *language*/units/

Add new unit.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL
- **language** (*string*) – Translation language code

Request JSON Object

- **key** (*string*) – Name of translation unit (used as key or context)
- **value** (*array*) – Source strings (use single string if not creating plural)

Response JSON Object

- **unit** (*object*) – newly created unit; see *GET /api/units/(int:id)/*

Siehe auch:

Zeichenketten verwalten, adding-new-strings

POST /api/translations/(string: *project*)/string: *component*/string: *language*/autotranslate/

Trigger automatic translation.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL
- **language** (*string*) – Translation language code

Request JSON Object

- **mode** (*string*) – Automatischer Übersetzungsmodus
- **filter_type** (*string*) – Automatic translation filter type
- **auto_source** (*string*) – Automatische Übersetzung - `mt` oder `others`
- **component** (*string*) – Aktivieren Sie den Beitrag zum gemeinsamen Übersetzungsspeicher für das Projekt, um Zugriff auf zusätzliche Komponenten zu erhalten.
- **engines** (*array*) – Vorschläge aus automatischer Übersetzung
- **threshold** (*string*) – Scoreschwellwert

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

Download current translation file as it is stored in the VCS (without the `format` parameter) or converted to another format (see *Downloading translations*).

Bemerkung: 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`
- **q** (*string*) – Filter downloaded strings, see search.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL
- **language** (*string*) – Translation language code

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

Upload new file with translations.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL
- **language** (*string*) – Translation language code

Form Parameters

- **string conflict** – How to deal with conflicts (`ignore`, `replace-translated` or `replace-approved`)
- **file file** – Uploaded file
- **string email** – Autor-E-Mail
- **string author** – Autor-Name
- **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-Beispiel:

```
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/*.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL
- **language** (*string*) – Translation language code

POST /api/translations/(string: *project*)/(string: *component*)/(string: *language*)/repository/

Performs given operation on the VCS repository.

See *POST /api/projects/(string:project)/repository/* for documentation.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL
- **language** (*string*) – Translation language code

Request JSON Object

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

Response JSON Object

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

GET /api/translations/(string: *project*)/string: *component*/string: *language*/statistics/

Returns detailed translation statistics.

Neu in Version 2.7.

Parameter

- **project** (*string*) – Kurzer Projekt-URL
- **component** (*string*) – Kurzer Komponenten-URL
- **language** (*string*) – Translation language code

Response JSON Object

- **code** (*string*) – Sprachcode
- **failing** (*int*) – number of failing checks
- **failing_percent** (*float*) – percentage of failing checks
- **fuzzy** (*int*) – Anzahl der fragwürdigen (zur Bearbeitung markierten) Zeichenfolgen
- **fuzzy_percent** (*float*) – percentage of fuzzy (marked for edit) strings
- **total_words** (*int*) – total number of words
- **translated_words** (*int*) – number of translated words
- **last_author** (*string*) – name of last author
- **last_change** (*timestamp*) – date of last change
- **name** (*string*) – language name
- **total** (*int*) – total number of strings
- **translated** (*int*) – number of translated strings
- **translated_percent** (*float*) – percentage of translated strings
- **url** (*string*) – URL to access the translation (engagement URL)
- **url_translate** (*string*) – URL to access the translation (real translation URL)

1.12.10 Speicher

Neu in Version 4.14.

GET /api/memory/

Returns a list of memory results.

DELETE /api/memory/(int: *memory_object_id*)/

Deletes a memory object

Parameter

- **memory_object_id** – Memory Object ID

1.12.11 Units

A *unit* is a single piece of a translation which pairs a source string with a corresponding translated string and also contains some related metadata. The term is derived from the Translate Toolkit and XLIFF.

Neu in Version 2.10.

GET /api/units/

Returns list of translation units.

Siehe auch:

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

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

Geändert in Version 4.3: The `target` and `source` are now arrays to properly handle plural strings.

Returns information about translation unit.

Parameter

- **id** (*int*) – Unit ID

Response JSON Object

- **translation** (*string*) – URL of a related translation object
- **source** (*array*) – Ausgangszeichenkette
- **previous_source** (*string*) – previous source string used for fuzzy matching
- **target** (*array*) – target string
- **id_hash** (*string*) – unique identifier of the unit
- **content_hash** (*string*) – unique identifier of the source string
- **location** (*string*) – location of the unit in source code
- **context** (*string*) – translation unit context
- **note** (*string*) – translation unit note
- **flags** (*string*) – translation unit flags
- **state** (*int*) – unit state, 0 - untranslated, 10 - needs editing, 20 - translated, 30 - approved, 100 - read only
- **fuzzy** (*boolean*) – ob die Einheit fragwürdig oder zur Überprüfung markiert ist
- **translated** (*boolean*) – ob die Einheit übersetzt wird
- **approved** (*boolean*) – ob die Übersetzung genehmigt wird
- **position** (*int*) – Position der Einheit in der Übersetzungsdatei
- **has_suggestion** (*boolean*) – ob die Einheit Vorschläge hat

- **has_comment** (*boolean*) – ob die Einheit Kommentare hat
- **has_failing_check** (*boolean*) – ob die Einheit fehlerhafte Prüfungen aufweist
- **num_words** (*int*) – Anzahl der Ausgangswörter
- **priority** (*int*) – Übersetzungspriorität; 100 ist Standard
- **id** (*int*) – Einheitenkennung
- **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, unter der die Einheit bearbeitet werden kann
- **source_unit** (*string*) – Source unit link; see `GET /api/units/(int:id)/`
- **pending** (*boolean*) – whether the unit is pending for write
- **timestamp** (*timestamp*) – string age

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

Neu in Version 4.3.

Führt eine teilweise Aktualisierung der Übersetzungseinheit durch.

Parameter

- **id** (*int*) – Unit ID

Request JSON Object

- **state** (*int*) – unit state, 0 - untranslated, 10 - needs editing, 20 - translated, 30 - approved (need review workflow enabled, see *Zugehörige Prüfer*)
- **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*)/

Neu in Version 4.3.

Führt ein vollständiges Update der Übersetzungseinheit durch.

Parameter

- **id** (*int*) – Unit ID

Request JSON Object

- **state** (*int*) – unit state, 0 - untranslated, 10 - needs editing, 20 - translated, 30 - approved (need review workflow enabled, see *Zugehörige Prüfer*)
- **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*)/

Neu in Version 4.3.

Löscht eine Übersetzungseinheit.

Parameter

- **id** (*int*) – Unit ID

1.12.12 Änderungen

Neu in Version 2.10.

GET /api/changes/

Geändert in Version 4.1: Filtering of changes was introduced in the 4.1 release.

Returns a list of translation changes.

Siehe auch:

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

Query Parameters

- **user** (*string*) – Username of user to filters
- **action** (*int*) – Action to filter, can be used several times
- **timestamp_after** (*timestamp*) – ISO 8601 formatted timestamp to list changes after
- **timestamp_before** (*timestamp*) – ISO 8601 formatted timestamp to list changes before

GET /api/changes/(int: *id*)/

Returns information about translation change.

Parameter

- **id** (*int*) – Change ID

Response JSON Object

- **unit** (*string*) – URL eines zugehörigen Einheitenobjekts
- **translation** (*string*) – URL of a related translation object
- **component** (*string*) – URL eines zugehörigen Komponentenobjekts
- **user** (*string*) – URL eines zugehörigen Benutzerobjekts
- **author** (*string*) – URL eines zugehörigen Autorenobjekts
- **timestamp** (*timestamp*) – Zeitstempel des Ereignisses
- **action** (*int*) – numeric identification of action
- **action_name** (*string*) – text description of action
- **target** (*string*) – event changed text or detail
- **id** (*int*) – change identifier

1.12.13 Bildschirmfotos

Neu in Version 2.14.

GET /api/screenshots/

Returns a list of screenshot string information.

Siehe auch:

Die Attribute von Bildschirmfoto-Objekten sind unter *GET /api/screenshots/(int:id)/* dokumentiert.

GET /api/screenshots/(int: *id*)/

Gibt Informationen über Bildschirmfotos zurück.

Parameter

- **id** (*int*) – Bildschirmfoto-ID

Response JSON Object

- **name** (*string*) – Name eines Bildschirmfotos
- **component** (*string*) – URL eines zugehörigen Komponentenobjekts
- **file_url** (*string*) – URL zum Herunterladen einer Datei; siehe *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/

Laden Sie das Bildschirmfoto herunter.

Parameter

- **id** (*int*) – Bildschirmfoto-ID

POST /api/screenshots/(int: *id*)/file/

Ersetzen Sie das Bildschirmfoto.

Parameter

- **id** (*int*) – Bildschirmfoto-ID

Form Parameters

- **file image** – Uploaded file

CURL-Beispiel:

```
curl -X POST \
    -F image=@image.png \
    -H "Authorization: Token TOKEN" \
    http://example.com/api/screenshots/1/file/
```

POST /api/screenshots/(int: *id*)/units/

Verknüpfen Sie die Ausgangszeichenkette mit dem Bildschirmfoto.

Parameter

- **id** (*int*) – Bildschirmfoto-ID

Form Parameters

- **string unit_id** – Unit ID

Response JSON Object

- **name** (*string*) – Name eines Bildschirmfotos

- **translation** (*string*) – URL of a related translation object
- **file_url** (*string*) – URL zum Herunterladen einer Datei; siehe *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*

Entfernen Sie die Verknüpfung der Ausgangszeichenkette mit dem Bildschirmfoto.

Parameter

- **id** (*int*) – Bildschirmfoto-ID
- **unit_id** – Source string unit ID

POST /api/screenshots/

Erzeugt ein neues Bildschirmfoto.

Form Parameters

- **file image** – Uploaded file
- **string name** – Name des Bildschirmfotos
- **string project_slug** – Project slug
- **string component_slug** – Component slug
- **string language_code** – Sprachkürzel

Response JSON Object

- **name** (*string*) – Name eines Bildschirmfotos
- **component** (*string*) – URL eines zugehörigen Komponentenobjekts
- **file_url** (*string*) – URL zum Herunterladen einer Datei; siehe *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*)/

Teilinformationen zum Bildschirmfoto bearbeiten.

Parameter

- **id** (*int*) – Bildschirmfoto-ID

Response JSON Object

- **name** (*string*) – Name eines Bildschirmfotos
- **component** (*string*) – URL eines zugehörigen Komponentenobjekts
- **file_url** (*string*) – URL zum Herunterladen einer Datei; siehe *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*)/

Bearbeiten Sie alle Informationen zum Bildschirmfoto.

Parameter

- **id** (*int*) – Bildschirmfoto-ID

Response JSON Object

- **name** (*string*) – Name eines Bildschirmfotos
- **component** (*string*) – URL eines zugehörigen Komponentenobjekts
- **file_url** (*string*) – URL zum Herunterladen einer Datei; siehe `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*)/

Bildschirmfoto löschen.

Parameter

- **id** (*int*) – Bildschirmfoto-ID

1.12.14 Erweiterungen

Neu in Version 4.4.1.

GET /api/addons/

Gibt eine Liste der Erweiterungen aus.

Siehe auch:

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

GET /api/addons/(int: *id*)/

Returns information about add-on information.

Parameter

- **id** (*int*) – Erweiterungs-ID

Response JSON Object

- **name** (*string*) – name of an add-on
- **component** (*string*) – URL eines zugehörigen Komponentenobjekts
- **configuration** (*object*) – Optional add-on configuration

Siehe auch:

Erweiterungen

POST /api/components/(string: *project*)/ string: *component*/addons/

Creates a new add-on.

Parameter

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

Request JSON Object

- **name** (*string*) – name of an add-on
- **configuration** (*object*) – Optional add-on configuration

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

Edit partial information about add-on.

Parameter

- **id** (*int*) – Erweiterungs-ID

Response JSON Object

- **configuration** (*object*) – Optional add-on configuration

PUT /api/addons/(int: *id*)/

Edit full information about add-on.

Parameter

- **id** (*int*) – Erweiterungs-ID

Response JSON Object

- **configuration** (*object*) – Optional add-on configuration

DELETE /api/addons/(int: *id*)/

Delete add-on.

Parameter

- **id** (*int*) – Erweiterungs-ID

1.12.15 Komponentenlisten

Neu in Version 4.0.

GET /api/component-lists/

Returns a list of component lists.

Siehe auch:

Component list object attributes are documented at *GET /api/component-lists/(str:slug)/*.

GET /api/component-lists/(str: *slug*)/

Returns information about component list.

Parameter

- **slug** (*string*) – Component list slug

Response JSON Object

- **name** (*string*) – name of a component list
- **slug** (*string*) – slug of a component list
- **show_dashboard** (*boolean*) – whether to show it on a dashboard
- **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*)/

Changes the component list parameters.

Parameter

- **slug** (*string*) – Component list slug

Request JSON Object

- **name** (*string*) – name of a component list
- **slug** (*string*) – slug of a component list
- **show_dashboard** (*boolean*) – whether to show it on a dashboard

PATCH /api/component-lists/(**str:** *slug*)**/**

Changes the component list parameters.

Parameter

- **slug** (*string*) – Component list slug

Request JSON Object

- **name** (*string*) – name of a component list
- **slug** (*string*) – slug of a component list
- **show_dashboard** (*boolean*) – whether to show it on a dashboard

DELETE /api/component-lists/(**str:** *slug*)**/**

Deletes the component list.

Parameter

- **slug** (*string*) – Component list slug

POST /api/component-lists/(**str:** *slug*)**/components/**

Associate component with a component list.

Parameter

- **slug** (*string*) – Component list slug

Form Parameters

- **string component_id** – Component ID

DELETE /api/component-lists/(**str:** *slug*)**/components/**
str: *component_slug*

Disassociate a component from the component list.

Parameter

- **slug** (*string*) – Component list slug
- **component_slug** (*string*) – Component slug

1.12.16 Glossar

Geändert in Version 4.5: Glossaries are now stored as regular components, translations and strings, please use respective API instead.

1.12.17 Tasks

Neu in Version 4.4.

GET /api/tasks/

Eine Auflistung der Aufgaben ist derzeit nicht verfügbar.

GET /api/tasks/(**str:** *uuid*)**/**

Gibt Informationen über eine Aufgabe zurück

Parameter

- **uuid** (*string*) – Task UUID

Response JSON Object

- **completed** (*boolean*) – ob die Aufgabe abgeschlossen ist
- **progress** (*int*) – Task progress in percent

- **result** (*object*) – Task result or progress details
- **log** (*string*) – Aufgabenprotokoll

1.12.18 Metrics

GET /api/metrics/

Returns server metrics.

Response JSON Object

- **units** (*int*) – Anzahl der Einheiten
- **units_translated** (*int*) – Anzahl der übersetzten Einheiten
- **users** (*int*) – Anzahl der Benutzer
- **changes** (*int*) – Anzahl der Änderungen
- **projects** (*int*) – Anzahl der Projekte
- **components** (*int*) – Gleiche Pluralformen
- **translations** (*int*) – Anzahl der Übersetzungen
- **languages** (*int*) – Anzahl der verwendeten Sprachen
- **checks** (*int*) – Anzahl der ausgelösten Qualitätsprüfungen
- **configuration_errors** (*int*) – Anzahl der Konfigurationsfehler
- **suggestions** (*int*) – Anzahl der ausstehenden Übersetzungsvorschläge
- **celery_queues** (*object*) – Lengths of Celery queues, see *Background tasks using Celery*
- **name** (*string*) – Konfigurierter Servername

1.12.19 Benachrichtigungs-Hooks

Notification hooks allow external applications to notify Weblate that the VCS repository has been updated.

You can use repository endpoints for projects, components and translations to update individual repositories; see `POST /api/projects/(string:project)/repository/` for documentation.

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

Veraltet ab Version 2.6: Please use `POST /api/components/(string:project)/(string:component)/repository/` instead which works properly with authentication for ACL limited projects.

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

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

Veraltet ab Version 2.6: Please use `POST /api/projects/(string:project)/repository/` instead which works properly with authentication for ACL limited projects.

Triggers update of all components in a project (pulling from VCS and scanning for translation changes).

POST /hooks/github/

Special hook for handling GitHub notifications and automatically updating matching components.

Bemerkung: 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.

Siehe auch:

Automatically receiving changes from GitHub
Für Anweisungen zur Einrichtung der GitHub-Integration

https://docs.github.com/en/get-started/customizing-your-github-workflow/exploring-integrations/about-webhooks
Allgemeine Informationen zu GitHub-Webhooks

ENABLE_HOOKS
For enabling hooks for whole Weblate

POST /hooks/gitlab/

Special hook for handling GitLab notifications and automatically updating matching components.

Siehe auch:

Automatically receiving changes from GitLab
Für Anweisungen zur Einrichtung der GitLab-Integration

https://docs.gitlab.com/ee/user/project/integrations/webhooks.html
Allgemeine Informationen zu GitLab-Webhooks

ENABLE_HOOKS
For enabling hooks for whole Weblate

POST /hooks/bitbucket/

Special hook for handling Bitbucket notifications and automatically updating matching components.

Siehe auch:

Automatically receiving changes from Bitbucket
Für Anweisungen zur Einrichtung der Bitbucket-Integration

https://support.atlassian.com/bitbucket-cloud/docs/manage-webhooks/
Allgemeine Informationen zu Bitbucket-Webhooks

ENABLE_HOOKS
For enabling hooks for whole Weblate

POST /hooks/pagure/

Neu in Version 3.3.

Special hook for handling Pagure notifications and automatically updating matching components.

Siehe auch:

Automatically receiving changes from Pagure
Für Anweisungen zur Einrichtung der Pagure-Integration

https://docs.pagure.org/pagure/usage/using_webhooks.html
Allgemeine Informationen zu Pagure-Webhooks

ENABLE_HOOKS
For enabling hooks for whole Weblate

POST /hooks/azure/

Neu in Version 3.8.

Special hook for handling Azure DevOps notifications and automatically updating matching components.

Bemerkung: Please make sure that *Resource details to send* is set to *All*, otherwise Weblate will not be able to match your Azure repository.

Siehe auch:

Automatically receiving changes from Azure Repos
Für Anweisungen zur Einrichtung der Azure-Integration

https://docs.microsoft.com/de-de/azure/devops/service-hooks/services/webhooks?view=azure-devops
Generic information about Azure DevOps Web Hooks

ENABLE_HOOKS
For enabling hooks for whole Weblate

POST /hooks/gitea/

Neu in Version 3.9.

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

Siehe auch:

Automatically receiving changes from Gitea Repos
Für Anweisungen zur Einrichtung der Gitea-Integration

https://docs.gitea.io/en-us/webhooks/
Allgemeine Informationen zu Gitea-Webhooks

ENABLE_HOOKS
For enabling hooks for whole Weblate

POST /hooks/gitee/

Neu in Version 3.9.

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

Siehe auch:

Automatically receiving changes from Gitee Repos
Für Anweisungen zur Einrichtung der Gitee-Integration

https://gitee.com/help/categories/40
Allgemeine Informationen zu Gitee-Webhooks

ENABLE_HOOKS
For enabling hooks for whole Weblate

1.12.20 Exports

Weblate provides various exports to allow you to further process the data.

GET /exports/stats/(string: *project*)/(string: *component*)/

Query Parameters

- **format** (*string*) – Output format: either `json` or `csv`

Veraltet ab Version 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.

Beispielanfrage:

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

Beispielantwort:

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

[
    {
        "code": "cs",
        "failing": 0,
        "failing_percent": 0.0,
        "fuzzy": 0,
        "fuzzy_percent": 0.0,
        "last_author": "Michal Čihař",
        "last_change": "2012-03-28T15:07:38+00:00",
        "name": "Czech",
        "total": 436,
        "total_words": 15271,
        "translated": 436,
        "translated_percent": 100.0,
        "translated_words": 3201,
        "url": "http://hosted.weblate.org/engage/weblate/cs/",
        "url_translate": "http://hosted.weblate.org/projects/weblate/main/cs/"
    },
    {
        "code": "nl",
        "failing": 21,
        "failing_percent": 4.8,
        "fuzzy": 11,
        "fuzzy_percent": 2.5,
        "last_author": null,
        "last_change": null,
        "name": "Dutch",
        "total": 436,
        "total_words": 15271,
        "translated": 319,
        "translated_percent": 73.2,
        "translated_words": 3201,
        "url": "http://hosted.weblate.org/engage/weblate/nl/",
        "url_translate": "http://hosted.weblate.org/projects/weblate/main/nl/"
    },
    {
        "code": "el",
        "failing": 11,
        "failing_percent": 2.5,
        "fuzzy": 21,
        "fuzzy_percent": 4.8,
        "last_author": null,
        "last_change": null,
        "name": "Greek",
        "total": 436,
        "total_words": 15271,
        "translated": 312,
        "translated_percent": 71.6,
        "translated_words": 3201,
        "url": "http://hosted.weblate.org/engage/weblate/el/",
        "url_translate": "http://hosted.weblate.org/projects/weblate/main/el/"
    }
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
]
```

1.12.21 RSS-Feeds

Änderungen an Übersetzungen werden in RSS-Feeds exportiert.

GET /exports/rss/(**string:** *project*)**/**
string: *component***/string:** *language***/**

Ruft den RSS-Feed mit den letzten Änderungen für eine Übersetzung ab.

GET /exports/rss/(**string:** *project*)**/**
string: *component***/**

Ruft den RSS-Feed mit den letzten Änderungen für eine Komponente ab.

GET /exports/rss/(**string:** *project*)**/**

Ruft den RSS-Feed mit den letzten Änderungen für ein Projekt ab.

GET /exports/rss/language/(**string:** *language*)**/**

Ruft den RSS-Feed mit den letzten Änderungen für eine Sprache ab.

GET /exports/rss/

Ruft den RSS-Feed mit den letzten Änderungen für die Weblate-Instanz ab.

Siehe auch:

RSS auf Wikipedia

1.13 Weblate Client

Neu in Version 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 Installation

The Weblate Client is shipped separately and includes the Python module. To use the commands below, you need to install `wlc`:

```
pip install wlc
```

1.13.2 Docker usage

The Weblate Client is also available as a Docker image.

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

Installieren:

```
docker pull weblate/wlc
```

Der Docker-Container verwendet die Standardeinstellungen von Weblate und verbindet sich mit der auf localhost bereitgestellten API. Die API-URL und der API_KEY können über die von Weblate akzeptierten Argumente konfiguriert werden.

The command to launch the container uses the following syntax:

```
docker run --rm weblate/wlc [WLC_ARGS]
```

Example:

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

You might want to pass your *Configuration files* to the Docker container, the easiest approach is to add your current directory as `/home/weblate` volume:

```
docker run --volume $PWD:/home/weblate --rm weblate/wlc show
```

1.13.3 Getting started

The wlc configuration is stored in `~/.config/weblate` (see *Configuration files* 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
```

Siehe auch:

Configuration files

1.13.4 Synopsis

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

Commands actually indicate which operation should be performed.

1.13.5 Beschreibung

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

Das Programm akzeptiert die folgenden Argumente, die das Ausgabeformat oder die zu verwendende Weblate-Instanz definieren. Diese müssen vor jedem Befehl eingegeben werden.

`--format` `{csv,json,text,html}`

Specify the output format.

`--url` `URL`

Specify the API URL. Overrides any value found in the configuration file, see *Configuration files*. 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 *Configuration files*. You can find your key in your profile on Weblate.

--config PATH

Overrides the configuration file path, see *Configuration files*.

--config-section SECTION

Overrides configuration file section in use, see *Configuration files*.

Befehle

The following commands are available:

version

Prints the current version.

list-languages

Lists used languages in Weblate.

list-projects

Lists projects in Weblate.

list-components

Lists components in Weblate.

list-translations

Lists translations in 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

Neu in Version 0.7: Supported since wlc 0.7.

Resets changes in Weblate object to match remote repository (translation, component or project).

cleanup

Neu in Version 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

Neu in Version 0.5: Supported since wlc 0.5.

Displays lock status.

lock

Neu in Version 0.5: Supported since wlc 0.5.

Locks component from further translation in Weblate.

unlock

Neu in Version 0.5: Supported since wlc 0.5.

Unlocks translation of Weblate component.

changes

Neu in Version 0.7: Supported since wlc 0.7 and Weblate 2.10.

Displays changes for a given object.

download

Neu in Version 0.7: Supported since wlc 0.7.

Downloads a translation file.

--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

Neu in Version 0.9: Supported since wlc 0.9.

Uploads a translation file.

--overwrite

Overwrite existing translations upon uploading.

--input

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

--method

Upload method to use, see *Import methods*.

--fuzzy

Fuzzy (marked for edit) strings processing (*empty*, `process`, `approve`)

--author-name

Author name, to override currently authenticated user

--author-email

Author e-mail, to override currently authenticated user

Hinweis: You can get more detailed information on invoking individual commands by passing `--help`, for example: `wlc ls --help`.

1.13.6 Configuration files

.weblate, .weblate.ini, weblate.ini

Geändert in Version 1.6: The files with *.ini* extension are accepted as well.

Per project configuration file

C:\Users\NAME\AppData\weblate.ini

Neu in Version 1.6.

User configuration file on Windows.

~/.config/weblate

User configuration file

/etc/xdg/weblate

System wide configuration file

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

API KEY to access Weblate.

url

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

translation

Pfad zur Standardübersetzung - Komponente oder Projekt.

The configuration file is an INI file, for example:

```
[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 Beispiele

Aktuelle Programmversion drucken:

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

Alle Projekte auflisten:

```
$ 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/
```

Übersetzungsdatei hochladen:

```
$ wlc upload project/component/language --input /tmp/hello.po
```

Sie können auch angeben, an welchem Projekt wlc arbeiten soll:

```
$ 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/
```

Mit diesem Setup können anstehende Änderungen im aktuellen Projekt einfach übertragen werden:

```
$ wlc commit
```

1.14 Weblate's Python API

1.14.1 Installation

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

```
pip install wlc
```

1.14.2 `wlc`

`WeblateException`

exception wlc.**WeblateException**

Base class for all exceptions.

Weblate

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

Parameter

- **key** (*str*) – User key
- **url** (*str*) – API server URL, if not specified default is used
- **config** (wlc.config.WeblateConfig) – Configuration object, overrides any other parameters.

Access class to the API, define API key and optionally API URL.

get(*path*)

Parameter
path (*str*) – Request path

Rückgabetyp
object

Performs a single API GET call.

post(*path*, ***kwargs*)

Parameter
path (*str*) – Request path

Rückgabetyp
object

Performs a single API GET call.

1.14.3 wlc.config

WeblateConfig

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

Parameter
section (*str*) – Configuration section to use

Configuration file parser following XDG specification.

load(*path=None*)

Parameter
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*)

Parameter

- **settings** (*list*) – Settings to override as list of tuples
- **stdout** (*object*) – stdout file object for printing output, uses `sys.stdout` as default
- **args** (*list*) – Befehlszeilenargumente, die verarbeitet werden sollen, verwendet standardmäßig `sys.args`

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.

KAPITEL 2

Administrator docs

2.1 Configuration instructions

2.1.1 Installing Weblate

Installing using Docker

With dockerized Weblate deployment you can get your personal Weblate instance up and running in seconds. All of Weblate's dependencies are already included. PostgreSQL is set up as the default database.

Hardware requirements

Weblate should run on any contemporary hardware without problems, the following is the minimal configuration required to run Weblate on a single host (Weblate, database and webserver):

- 2 GB of RAM
- 2 CPU cores
- 1 GB of storage space

The more memory the better - it is used for caching on all levels (filesystem, database and Weblate).

Many concurrent users increases the amount of needed CPU cores. For hundreds of translation components at least 4 GB of RAM is recommended.

The typical database storage usage is around 300 MB per 1 million hosted words. Storage space needed for cloned repositories varies, but Weblate tries to keep their size minimal by doing shallow clones.

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

Installation

The following examples assume you have a working Docker environment, with `docker-compose` installed. Please check the Docker documentation for instructions.

1. Clone the weblate-docker repo:

```
git clone https://github.com/WeblateOrg/docker-compose.git weblate-docker
cd weblate-docker
```

2. Create a `docker-compose.override.yml` file with your settings. See *Docker-Umgebungsvariablen* for full list of environment variables.

```
version: '3'
services:
  weblate:
    ports:
      - 80:8080
    environment:
      WEBLATE_EMAIL_HOST: smtp.example.com
      WEBLATE_EMAIL_HOST_USER: user
      WEBLATE_EMAIL_HOST_PASSWORD: pass
      WEBLATE_SERVER_EMAIL: weblate@example.com
      WEBLATE_DEFAULT_FROM_EMAIL: weblate@example.com
      WEBLATE_SITE_DOMAIN: weblate.example.com
      WEBLATE_ADMIN_PASSWORD: password for the admin user
      WEBLATE_ADMIN_EMAIL: weblate.admin@example.com
```

Bemerkung: 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.

Geändert in Version 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.

Geändert in Version 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.

Siehe auch:

Invoking management commands

Auswählen des Docker-Hub-Tags

Sie können die folgenden Tags auf Docker Hub verwenden. Eine vollständige Liste der verfügbaren Tags finden Sie unter https://hub.docker.com/r/weblate/weblate/tags/.

Tag-Name	Beschreibung	Use case
`latest`	Stabile Version von Weblate, entspricht der neuesten getaggten Version	Fortlaufende Updates in einer Produktionsumgebung
`<VERSION>-<PATCH>`	Weblate stabile Version	Gut definierter Einsatz in einer Produktionsumgebung
`edge`	Stabiles Weblate-Release mit Entwicklungsänderungen im Docker-Container (z. B. aktualisierte Abhängigkeiten)	Fortlaufende Updates in einer Staging-Umgebung
`edge-<DATE>-<SHA>`	Stabiles Weblate-Release mit Entwicklungsänderungen im Docker-Container (z. B. aktualisierte Abhängigkeiten)	Gut definierter Einsatz in einer Staging-Umgebung
`bleeding`	Weblate-Entwicklungsversion von Git	Fortlaufende Updates zum Testen kommender Weblate-Funktionen
`bleeding-<DATE>-<SHA>`	Weblate-Entwicklungsversion von Git	Gut definierter Einsatz zum Testen kommender Weblate-Funktionen

Jedes Bild wird vor der Veröffentlichung von unserer CI getestet, so dass selbst die *bleeding*-Version sicher zu verwenden ist.

Docker-Container mit HTTPS-Unterstützung

Please see *Installation* for generic deployment instructions, this section only mentions differences compared to it.

Verwendung eigener SSL-Zertifikate

Neu in Version 3.8-3.

Wenn Sie ein eigenes SSL-Zertifikat haben, das Sie verwenden möchten, legen Sie die Dateien einfach in das Weblate-Datenvolumen (siehe *Docker-Container-Volumes*):

- `ssl/fullchain.pem` containing the certificate including any needed CA certificates
- `ssl/privkey.pem` containing the private key

Both of these files must be owned by the same user as the one starting the docker container and have file mask set to `600` (readable and writable only by the owning user).

Additionally, Weblate container will now accept SSL connections on port 4443, you will want to include the port forwarding for HTTPS in docker compose override:

```
version: '3'
services:
  weblate:
    ports:
      - 80:8080
      - 443:4443
```

If you already host other sites on the same server, it is likely ports `80` and `443` are used by a reverse proxy, such as NGINX. To pass the HTTPS connection from NGINX to the docker container, you can use the following configuration:

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

    server_name <SITE_URL>;
    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.

Automatische SSL-Zertifikate mit Let's Encrypt

In case you want to use Let's Encrypt automatically generated SSL certificates on public installation, you need to add a reverse HTTPS proxy an additional Docker container, https-portal will be used for that. This is made use of in the `docker-compose-https.yml` file. Then create a `docker-compose-https.override.yml` file with your settings:

```
version: '3'
services:
  weblate:
    environment:
      WEBLATE_EMAIL_HOST: smtp.example.com
      WEBLATE_EMAIL_HOST_USER: user
      WEBLATE_EMAIL_HOST_PASSWORD: pass
      WEBLATE_SITE_DOMAIN: weblate.example.com
      WEBLATE_ADMIN_PASSWORD: password for admin user
  https-portal:
    environment:
      DOMAINS: 'weblate.example.com -> http://weblate:8080'
```

Whenever invoking **`docker-compose`** you need to pass both files to it, and then do:

```
docker-compose -f docker-compose-https.yml -f docker-compose-https.override.yml␣
↪build
docker-compose -f docker-compose-https.yml -f docker-compose-https.override.yml up
```

Aktualisieren des Docker-Containers

Normalerweise ist es eine gute Idee, nur den Weblate-Container zu aktualisieren und den PostgreSQL-Container auf der vorhandenen Version zu belassen, da ein Upgrade von PostgreSQL ziemlich mühsam ist und in den meisten Fällen nicht viel bringt.

Geändert in Version 4.10-1: Since Weblate 4.10-1, the Docker container uses Django 4.0 what requires PostgreSQL 10 or newer, please upgrade it prior to upgrading Weblate. See *Upgrade von 4.0 auf 4.10* and *Aktualisieren des PostgreSQL-Containers*.

Sie können dies tun, indem Sie das bestehende Docker-Compose beibehalten und einfach die neuesten Images ziehen und dann neu starten:

```
# Fetch latest versions of the images
docker-compose pull
# Stop and destroy the containers
docker-compose down
# Spawn new containers in the background
docker-compose up -d
# Follow the logs during upgrade
docker-compose logs -f
```

Die Weblate-Datenbank sollte beim ersten Start automatisch migriert werden, und es sollten keine weiteren manuellen Maßnahmen erforderlich sein.

Bemerkung: Upgrades across major versions are not supported by Weblate. For example, if you are on 3.x series and want to upgrade to 4.x, first upgrade to the latest 4.0.x-y image (at time of writing this it is the `4.0.4-5`), which will do the migration and then continue upgrading to newer versions.

You might also want to update the `docker-compose` repository, though it's not needed in most case. See *Aktualisieren des PostgreSQL-Containers* for upgrading the PostgreSQL server.

Aktualisieren des PostgreSQL-Containers

PostgreSQL containers do not support automatic upgrading between version, you need to perform the upgrade manually. Following steps show one of the options of upgrading.

Siehe auch:

https://github.com/docker-library/postgres/issues/37

1. Weblate-Container stoppen:

```
docker-compose stop weblate cache
```

2. Backup the database:

```
docker-compose exec database pg_dumpall --clean --username weblate > backup.sql
```

3. Datenbank-Container sperren:

```
docker-compose stop database
```

4. Entfernen des PostgreSQL-Volumes:

```
docker-compose rm -v database
docker volume remove weblate_postgres-data
```

5. Adjust `docker-compose.yml` to use new PostgreSQL version.
6. Datenbank-Container öffnen:

```
docker-compose up -d database
```

7. Datenbank aus Sicherung wiederherstellen:

```
cat backup.sql | docker-compose exec -T database psql --username weblate --
↪dbname postgres
```

8. Alle verbleibenden Container öffnen:

```
docker-compose up -d
```

Administrator-Anmeldung

Nach der Container-Einrichtung können Sie sich als Benutzer *admin* mit dem in *WEBLATE_ADMIN_PASSWORD* bereitgestellten Passwort oder, falls es nicht festgelegt wurde, mit einem beim ersten Öffnen erzeugten Zufallspasswort anmelden.

Um das Passwort für *admin* zurückzusetzen, öffnen Sie den Container mit dem in *WEBLATE_ADMIN_PASSWORD* neu festgelegten Passwort nochmals.

Siehe auch:

WEBLATE_ADMIN_PASSWORD, *WEBLATE_ADMIN_NAME*, *WEBLATE_ADMIN_EMAIL*

Anzahl der Prozesse und Speicherverbrauch

Die Anzahl der Mitarbeitervorgänge wird sowohl für uWSGI als auch Celery automatisch auf Grundlage der Anzahl der CPUs bestimmt. Dies funktioniert für die meisten virtuellen Maschinen in der Cloud gut, da sie typischerweise wenig CPUs und große Speicherkapazitäten besitzen.

Für den Fall, dass Sie sehr viele CPU-Kerne haben und auf Speicherprobleme stoßen, versuchen Sie die Zahl der Arbeitskräfte zu reduzieren:

```
environment:
  WEBLATE_WORKERS: 2
```

Sie können auch individuelle Arbeitskräftekategorien feinabstimmen:

```
environment:
  WEB_WORKERS: 4
  CELERY_MAIN_OPTIONS: --concurrency 2
  CELERY_NOTIFY_OPTIONS: --concurrency 1
  CELERY_TRANSLATE_OPTIONS: --concurrency 1
```

Siehe auch:

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

Horizontale Skalierung

Neu in Version 4.6.

Sie können mehrere Weblate-Container ausführen, um den Dienst horizontal zu skalieren. Das Volume `/app/data` muss von allen Containern gemeinsam genutzt werden, es wird empfohlen, dafür ein Cluster-Dateisystem wie z. B. GlusterFS zu verwenden. Das Volume `/app/cache` sollte für jeden Container separat sein.

Jeder Weblate-Container hat eine definierte Rolle mit der Umgebungsvariablen *WEBLATE_SERVICE*. Bitte folgen Sie sorgfältig der Dokumentation, da einige der Dienste nur einmal im Cluster laufen sollen und auch die Reihenfolge der Dienste wichtig ist.

Ein Beispiel-Setup finden Sie im `docker-compose`-Repo als docker-compose-split.yml.

Docker-Umgebungsvariablen

Many of Weblate's *Konfiguration* can be set in the Docker container using environment variables:

Allgemeine Einstellungen

WEBLATE_DEBUG

Configures Django debug mode using `DEBUG`.

Beispiel:

```
environment:
  WEBLATE_DEBUG: 1
```

Siehe auch:

Disable debug mode

WEBLATE_LOGLEVEL

Configures the logging verbosity.

WEBLATE_LOGLEVEL_DATABASE

Configures the logging of the database queries verbosity.

WEBLATE_SITE_TITLE

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

WEBLATE_SITE_DOMAIN

Konfiguriert die Seitendomain. Dieser Parameter ist erforderlich.

Siehe auch:

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).

Beispiel:

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

Siehe auch:

Administrator-Anmeldung, *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`.

> **Warnung:** 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.

Siehe auch:

Administrator-Anmeldung, *WEBLATE_ADMIN_PASSWORD*, *WEBLATE_ADMIN_PASSWORD_FILE*, *WEBLATE_ADMIN_NAME*, *WEBLATE_ADMIN_EMAIL*

WEBLATE_ADMIN_PASSWORD_FILE

Legt den Pfad zu einer Datei fest, die das Passwort für den Benutzer *Administrator* enthält.

Siehe auch:

WEBLATE_ADMIN_PASSWORD

WEBLATE_SERVER_EMAIL

The email address that error messages are sent from.

Siehe auch:

SERVER_EMAIL, *Configure e-mail sending*

WEBLATE_DEFAULT_FROM_EMAIL

Legt die Adresse für ausgehende E-Mails fest.

Siehe auch:

DEFAULT_FROM_EMAIL, *Configure e-mail sending*

WEBLATE_CONTACT_FORM

Konfiguriert das Verhalten des Kontaktformulars, siehe *CONTACT_FORM*.

WEBLATE_ALLOWED_HOSTS

Configures allowed HTTP hostnames using *ALLOWED_HOSTS*.

Defaults to `*` which allows all hostnames.

Beispiel:

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

Siehe auch:

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

WEBLATE_REGISTRATION_OPEN

Configures whether registrations are open by toggling *REGISTRATION_OPEN*.

Beispiel:

```
environment:
  WEBLATE_REGISTRATION_OPEN: 0
```

WEBLATE_REGISTRATION_ALLOW_BACKENDS

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

Beispiel:

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

WEBLATE_TIME_ZONE

Configures the used time zone in Weblate, see TIME_ZONE.

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

Beispiel:

```
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.

Hinweis: Please see *ENABLE_HTTPS* documentation for possible caveats.

Bemerkung: This does not make the Weblate container accept HTTPS connections, you need to configure that as well, see *Docker-Container mit HTTPS-Unterstützung* for examples.

Beispiel:

```
environment:
  WEBLATE_ENABLE_HTTPS: 1
```

Siehe auch:

ENABLE_HTTPS Set correct site domain, *WEBLATE_SECURE_PROXY_SSL_HEADER*

WEBLATE_INTERLEDGER_PAYMENT_POINTERS

Neu in Version 4.12.1.

Lets Weblate set the *meta[name=monetization]* field in the head of the document. If multiple are specified, chooses one randomly.

Siehe auch:

INTERLEDGER_PAYMENT_POINTERS

WEBLATE_IP_PROXY_HEADER

Lets Weblate fetch the IP address from any given HTTP header. Use this when using a reverse proxy in front of the Weblate container.

Enables *IP_BEHIND_REVERSE_PROXY* and sets *IP_PROXY_HEADER*.

Bemerkung: 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`.

Beispiel:

```
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.

Beispiel:

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

Siehe auch:

SECURE_PROXY_SSL_HEADER

WEBLATE_REQUIRE_LOGIN

Enables *REQUIRE_LOGIN* to enforce authentication on whole Weblate.

Beispiel:

```
environment:
  WEBLATE_REQUIRE_LOGIN: 1
```

WEBLATE_LOGIN_REQUIRED_URLS_EXCEPTIONS

WEBLATE_ADD_LOGIN_REQUIRED_URLS_EXCEPTIONS

WEBLATE_REMOVE_LOGIN_REQUIRED_URLS_EXCEPTIONS

Adds URL exceptions for authentication required for the whole Weblate installation using *LOGIN_REQUIRED_URLS_EXCEPTIONS*.

You can either replace whole settings, or modify default value using `ADD` and `REMOVE` variables.

WEBLATE_GOOGLE_ANALYTICS_ID

Configures ID for Google Analytics by changing *GOOGLE_ANALYTICS_ID*.

WEBLATE_GITHUB_USERNAME

Configures GitHub username for GitHub pull-requests by changing *GITHUB_USERNAME*.

Siehe auch:

GitHub-Pull-Anfragen

WEBLATE_GITHUB_TOKEN

Neu in Version 4.3.

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

Siehe auch:

GitHub-Pull-Anfragen

WEBLATE_GITLAB_USERNAME

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

Siehe auch:

GitLab Merge Requests

WEBLATE_GITLAB_TOKEN

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

Siehe auch:

GitLab Merge Requests

WEBLATE_PAGURE_USERNAME

Configures Pagure username for Pagure merge-requests by changing *PAGURE_USERNAME*

Siehe auch:

Pagure Merge Requests

WEBLATE_PAGURE_TOKEN

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

Siehe auch:

Pagure Merge Requests

WEBLATE_DEFAULT_PULL_MESSAGE

Configures the default title and message for pull requests via API by changing *DEFAULT_PULL_MESSAGE*

Siehe auch:

DEFAULT_PULL_MESSAGE

WEBLATE_SIMPLIFY_LANGUAGES

Configures the language simplification policy, see *SIMPLIFY_LANGUAGES*.

WEBLATE_DEFAULT_ACCESS_CONTROL

Configures the default *Zugriffssteuerung* 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 *Verbreitung von Übersetzungen erlauben* for new components, see *DEFAULT_TRANSLATION_PROPAGATION*.

WEBLATE_DEFAULT_COMMITER_EMAIL

Configures *DEFAULT_COMMITER_EMAIL*.

WEBLATE_DEFAULT_COMMITER_NAME

Configures *DEFAULT_COMMITER_NAME*.

WEBLATE_DEFAULT_SHARED_TM

Configures *DEFAULT_SHARED_TM*.

WEBLATE_AKISMET_API_KEY

Configures the Akismet API key, see *AKISMET_API_KEY*.

WEBLATE_GPG_IDENTITY

Configures GPG signing of commits, see *WEBLATE_GPG_IDENTITY*.

Siehe auch:

Signing Git commits with GnuPG

WEBLATE_URL_PREFIX

Configures URL prefix where Weblate is running, see *URL_PREFIX*.

WEBLATE_SILENCED_SYSTEM_CHECKS

Configures checks which you do not want to be displayed, see SILENCED_SYSTEM_CHECKS.

WEBLATE_CSP_SCRIPT_SRC

WEBLATE_CSP_IMG_SRC

WEBLATE_CSP_CONNECT_SRC

WEBLATE_CSP_STYLE_SRC

WEBLATE_CSP_FONT_SRC

Allows to customize `Content-Security-Policy` HTTP header.

Siehe auch:

Content security policy, `CSP_SCRIPT_SRC`, `CSP_IMG_SRC`, `CSP_CONNECT_SRC`, `CSP_STYLE_SRC`, `CSP_FONT_SRC`

WEBLATE_LICENSE_FILTER

Configures `LICENSE_FILTER`.

WEBLATE_LICENSE_REQUIRED

Configures `LICENSE_REQUIRED`

WEBLATE_WEBSITE_REQUIRED

Configures `WEBSITE_REQUIRED`

WEBLATE_HIDE_VERSION

Configures `HIDE_VERSION`.

WEBLATE_BASIC_LANGUAGES

Configures `BASIC_LANGUAGES`.

WEBLATE_DEFAULT_AUTO_WATCH

Configures `DEFAULT_AUTO_WATCH`.

WEBLATE_RATELIMIT_ATTEMPTS

WEBLATE_RATELIMIT_LOCKOUT

WEBLATE_RATELIMIT_WINDOW

Neu in Version 4.6.

Konfiguriert den Ratelimiter.

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

Siehe auch:

Rate limiting, `RATELIMIT_ATTEMPTS`, `RATELIMIT_WINDOW`, `RATELIMIT_LOCKOUT`

WEBLATE_API_RATELIMIT_ANON

WEBLATE_API_RATELIMIT_USER

Neu in Version 4.11.

Configures API rate limiting. Defaults to `100/day` for anonymous and `5000/hour` for authenticated users.

Siehe auch:

API rate limiting

WEBLATE_ENABLE_HOOKS

Neu in Version 4.13.

Configures `ENABLE_HOOKS`.

WEBLATE_ENABLE_AVATARS

Neu in Version 4.6.1.

Configures `ENABLE_AVATARS`.

WEBLATE_LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH

Neu in Version 4.9.

Konfiguriert *LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH*.

WEBLATE_SSH_EXTRA_ARGS

Neu in Version 4.9.

Konfiguriert *SSH_EXTRA_ARGS*.

WEBLATE_BORG_EXTRA_ARGS

Neu in Version 4.9.

Konfiguriert *BORG_EXTRA_ARGS*.

Automatic suggestion settings

Geändert in Version 4.13: Automatic suggestion services are now configured in the user interface, see *Configuring automatic suggestions*.

The existing environment variables are imported during the migration to Weblate 4.13, but changing them will not have any further effect.

Authentifizierungseinstellungen

LDAP

WEBLATE_AUTH_LDAP_SERVER_URI

WEBLATE_AUTH_LDAP_USER_DN_TEMPLATE

WEBLATE_AUTH_LDAP_USER_ATTR_MAP

WEBLATE_AUTH_LDAP_BIND_DN

WEBLATE_AUTH_LDAP_BIND_PASSWORD

WEBLATE_AUTH_LDAP_BIND_PASSWORD_FILE

Path to the file containing the LDAP server bind password.

Siehe auch:

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

Konfiguration der LDAP-Authentifizierung.

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.
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
  # another example that can be used with OpenLDAP: 'full_name:cn,email:mail'
  WEBLATE_AUTH_LDAP_USER_ATTR_MAP: full_name:name,email:mail
```

Example for search and bind:

```
environment:
  WEBLATE_AUTH_LDAP_SERVER_URI: ldap://ldap.example.org
  WEBLATE_AUTH_LDAP_BIND_DN: CN=ldap,CN=Users,DC=example,DC=com
  WEBLATE_AUTH_LDAP_BIND_PASSWORD: password
  WEBLATE_AUTH_LDAP_USER_ATTR_MAP: full_name:name,email:mail
  WEBLATE_AUTH_LDAP_USER_SEARCH: CN=Users,DC=example,DC=com
```

Example for union search and bind:

```
environment:
  WEBLATE_AUTH_LDAP_SERVER_URI: ldap://ldap.example.org
  WEBLATE_AUTH_LDAP_BIND_DN: CN=ldap,CN=Users,DC=example,DC=com
  WEBLATE_AUTH_LDAP_BIND_PASSWORD: password
  WEBLATE_AUTH_LDAP_USER_ATTR_MAP: full_name:name,email:mail
  WEBLATE_AUTH_LDAP_USER_SEARCH_UNION: ou=users,dc=example,
↪dc=com|ou=otherusers,dc=example,dc=com
```

Example with search and bind against Active Directory:

```
environment:
  WEBLATE_AUTH_LDAP_BIND_DN: CN=ldap,CN=Users,DC=example,DC=com
  WEBLATE_AUTH_LDAP_BIND_PASSWORD: password
  WEBLATE_AUTH_LDAP_SERVER_URI: ldap://ldap.example.org
  WEBLATE_AUTH_LDAP_CONNECTION_OPTION_REFERRALS: 0
  WEBLATE_AUTH_LDAP_USER_ATTR_MAP: full_name:name,email:mail
  WEBLATE_AUTH_LDAP_USER_SEARCH: CN=Users,DC=example,DC=com
  WEBLATE_AUTH_LDAP_USER_SEARCH_FILTER: (sAMAccountName=%(user)s)
```

Siehe auch:

LDAP-Authentifizierung

GitHub

WEBLATE_SOCIAL_AUTH_GITHUB_KEY

WEBLATE_SOCIAL_AUTH_GITHUB_SECRET

WEBLATE_SOCIAL_AUTH_GITHUB_ORG_KEY

WEBLATE_SOCIAL_AUTH_GITHUB_ORG_SECRET

WEBLATE_SOCIAL_AUTH_GITHUB_ORG_NAME

WEBLATE_SOCIAL_AUTH_GITHUB_TEAM_KEY

WEBLATE_SOCIAL_AUTH_GITHUB_TEAM_SECRET

WEBLATE_SOCIAL_AUTH_GITHUB_TEAM_ID

Enables *GitHub-Authentifizierung*.

Bitbucket

WEBLATE_SOCIAL_AUTH_BITBUCKET_OAUTH2_KEY

WEBLATE_SOCIAL_AUTH_BITBUCKET_OAUTH2_SECRET

WEBLATE_SOCIAL_AUTH_BITBUCKET_KEY

WEBLATE_SOCIAL_AUTH_BITBUCKET_SECRET

Enables *Bitbucket-Authentifizierung*.

Facebook

WEBLATE_SOCIAL_AUTH_FACEBOOK_KEY

WEBLATE_SOCIAL_AUTH_FACEBOOK_SECRET

Enables *Facebook OAuth 2*.

Google

WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_KEY

WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET

WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_WHITELISTED_DOMAINS

WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_WHITELISTED_EMAILS

Enables *Google OAuth 2*.

GitLab

WEBLATE_SOCIAL_AUTH_GITLAB_KEY

WEBLATE_SOCIAL_AUTH_GITLAB_SECRET

WEBLATE_SOCIAL_AUTH_GITLAB_API_URL

Enables *GitLab OAuth 2*.

Azure Active Directory

WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_KEY

WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_SECRET

Enables Azure Active Directory authentication, see *Microsoft Azure Active Directory*.

Azure Active Directory with Tenant support

WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_KEY

WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_SECRET

WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_TENANT_ID

Enables Azure Active Directory authentication with Tenant support, see *Microsoft Azure Active Directory*.

Keycloak

WEBLATE_SOCIAL_AUTH_KEYCLOAK_KEY

WEBLATE_SOCIAL_AUTH_KEYCLOAK_SECRET

WEBLATE_SOCIAL_AUTH_KEYCLOAK_PUBLIC_KEY

WEBLATE_SOCIAL_AUTH_KEYCLOAK_ALGORITHM

WEBLATE_SOCIAL_AUTH_KEYCLOAK_AUTHORIZATION_URL

WEBLATE_SOCIAL_AUTH_KEYCLOAK_ACCESS_TOKEN_URL

WEBLATE_SOCIAL_AUTH_KEYCLOAK_TITLE

WEBLATE_SOCIAL_AUTH_KEYCLOAK_IMAGE

Enables Keycloak authentication, see documentation.

Linux vendors

You can enable authentication using Linux vendors authentication services by setting following variables to any value.

WEBLATE_SOCIAL_AUTH_FEDORA

WEBLATE_SOCIAL_AUTH_OPENSUSE

WEBLATE_SOCIAL_AUTH_UBUNTU

Slack

WEBLATE_SOCIAL_AUTH_SLACK_KEY

SOCIAL_AUTH_SLACK_SECRET

Enables Slack authentication, see *Slack*.

OpenID Connect

Neu in Version 4.13-1.

WEBLATE_SOCIAL_AUTH_OIDC_OIDC_ENDPOINT

WEBLATE_SOCIAL_AUTH_OIDC_KEY

WEBLATE_SOCIAL_AUTH_OIDC_SECRET

WEBLATE_SOCIAL_AUTH_OIDC_USERNAME_KEY

Configures generic OpenID Connect intergration.

Siehe auch:

OIDC (OpenID Connect)

SAML

Self-signed SAML keys are automatically generated on first container startup. In case you want to use own keys, place the certificate and private key in `/app/data/ssl/saml.crt` and `/app/data/ssl/saml.key`.

WEBLATE_SAML_IDP_ENTITY_ID

WEBLATE_SAML_IDP_URL

WEBLATE_SAML_IDP_X509CERT

WEBLATE_SAML_IDP_IMAGE

WEBLATE_SAML_IDP_TITLE

SAML Identity Provider settings, see *SAML-Authentifizierung*.

Other authentication settings

WEBLATE_NO_EMAIL_AUTH

Disables e-mail authentication when set to any value. See *Passwort-Authentifizierung deaktivieren*.

PostgreSQL database setup

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

Siehe auch:

Database setup for Weblate

POSTGRES_PASSWORD

PostgreSQL-Passwort.

POSTGRES_PASSWORD_FILE

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

POSTGRES_USER

PostgreSQL-Benutzername.

POSTGRES_DATABASE

PostgreSQL-Datenbankname.

POSTGRES_HOST

PostgreSQL server hostname or IP address. Defaults to `database`.

POSTGRES_PORT

PostgreSQL server port. Defaults to none (uses the default value).

POSTGRES_SSL_MODE

Configure how PostgreSQL handles SSL in connection to the server, for possible choices see SSL Mode Descriptions

POSTGRES_ALTER_ROLE

Configures name of role to alter during migrations, see *Configuring Weblate to use PostgreSQL*.

POSTGRES_CONN_MAX_AGE

Neu in Version 4.8.1.

Die Lebensdauer einer Datenbankverbindung als ganze Zahl von Sekunden. Verwenden Sie 0, um Datenbankverbindungen am Ende jeder Anfrage zu schließen (dies ist das Standardverhalten).

Die Aktivierung der Verbindungsaufrechterhaltung führt in der Regel zu mehr offenen Verbindungen zur Datenbank. Bitte passen Sie Ihre Datenbankkonfiguration vor der Aktivierung an.

Beispielkonfiguration:

```
environment:
    POSTGRES_CONN_MAX_AGE: 3600
```

Siehe auch:

`CONN_MAX_AGE`, Persistent connections

POSTGRES_DISABLE_SERVER_SIDE_CURSORS

Neu in Version 4.9.1.

Disable server side cursors in the database. This is necessary in some **pgbouncer** setups.

Beispielkonfiguration:

```
environment:
    POSTGRES_DISABLE_SERVER_SIDE_CURSORS: 1
```

Siehe auch:

`DISABLE_SERVER_SIDE_CURSORS`, Transaction pooling and server-side cursors

Database backup settings

Siehe auch:

Dumped data for backups

WEBLATE_DATABASE_BACKUP

Configures the daily database dump using *`DATABASE_BACKUP`*. Defaults to `plain`.

Caching server setup

Using Redis is strongly recommended by Weblate and you have to provide a Redis instance when running Weblate in Docker.

Siehe auch:

Enable caching

REDIS_HOST

The Redis server hostname or IP address. Defaults to `cache`.

REDIS_PORT

The Redis server port. Defaults to `6379`.

REDIS_DB

The Redis database number, defaults to `1`.

REDIS_PASSWORD

The Redis server password, not used by default.

REDIS_PASSWORD_FILE

Path to the file containing the Redis server password.

Siehe auch:

REDIS_PASSWORD

REDIS_TLS

Enables using SSL for Redis connection.

REDIS_VERIFY_SSL

Can be used to disable SSL certificate verification for Redis connection.

Einrichtung eines E-Mail-Servers

Damit ausgehende E-Mails funktionieren, müssen Sie einen Mailserver bereitstellen.

Beispiel für eine TLS-Konfiguration:

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

Beispiel für eine SSL-Konfiguration:

```
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
```

Siehe auch:

Configuring outgoing e-mail

WEBLATE_EMAIL_HOST

Mail server hostname or IP address.

Siehe auch:

WEBLATE_EMAIL_PORT, *WEBLATE_EMAIL_USE_SSL*, *WEBLATE_EMAIL_USE_TLS*, EMAIL_HOST

WEBLATE_EMAIL_PORT

Mail server port, defaults to 25.

Siehe auch:

EMAIL_PORT

WEBLATE_EMAIL_HOST_USER

Benutzer der E-Mail-Authentifizierung.

Siehe auch:

EMAIL_HOST_USER

WEBLATE_EMAIL_HOST_PASSWORD

Passwort für die E-Mail-Authentifizierung.

Siehe auch:

EMAIL_HOST_PASSWORD

WEBLATE_EMAIL_HOST_PASSWORD_FILE

Pfad zu der Datei, die das Kennwort für die E-Mail-Authentifizierung enthält.

Siehe auch:

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*.

Geändert in Version 4.11: The SSL/TLS support is automatically enabled based on the *WEBLATE_EMAIL_PORT*.

Siehe auch:

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*.

Geändert in Version 4.11: The SSL/TLS support is automatically enabled based on the *WEBLATE_EMAIL_PORT*.

Siehe auch:

WEBLATE_EMAIL_PORT, *WEBLATE_EMAIL_USE_SSL*, EMAIL_USE_TLS

WEBLATE_EMAIL_BACKEND

Konfiguriert das Django-Backend, das für den Versand von E-Mails verwendet werden soll.

Siehe auch:

Configure e-mail sending, EMAIL_BACKEND

WEBLATE_AUTO_UPDATE

Configures if and how Weblate should update repositories.

Siehe auch:

AUTO_UPDATE

Bemerkung: This is a Boolean setting (use `"true"` or `"false"`).

Integration der Website

WEBLATE_GET_HELP_URL

Configures *GET_HELP_URL*.

WEBLATE_STATUS_URL

Configures *STATUS_URL*.

WEBLATE_LEGAL_URL

Configures *LEGAL_URL*.

WEBLATE_PRIVACY_URL

Konfiguriert *PRIVACY_URL*.

Fehlerbericht

Es wird empfohlen, Fehler bei der Installation systematisch zu sammeln, siehe *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).

Lokalisierung CDN

WEBLATE_LOCALIZE_CDN_URL

WEBLATE_LOCALIZE_CDN_PATH

Neu in Version 4.2.1.

Configuration for *JavaScript-Lokalisierung CDN*.

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
```

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

Siehe auch:

weblate-cdn, *LOCALIZE_CDN_URL*, *LOCALIZE_CDN_PATH*

Changing enabled apps, checks, add-ons or autofixes

Neu in Version 3.8-5.

The built-in configuration of enabled checks, add-ons or autofixes can be adjusted by the following variables:

WEBLATE_ADD_APPS

WEBLATE_REMOVE_APPS

WEBLATE_ADD_CHECK

WEBLATE_REMOVE_CHECK

WEBLATE_ADD_AUTOFIX

WEBLATE_REMOVE_AUTOFIX

WEBLATE_ADD_ADDONS

WEBLATE_REMOVE_ADDONS

Beispiel:

```
environment:
  WEBLATE_REMOVE_AUTOFIX: weblate.trans.autofixes.whitespace.
↪SameBookendingWhitespace
  WEBLATE_ADD_ADDONS: customize.addons.MyAddon,customize.addons.OtherAddon
```

Siehe auch:

CHECK_LIST, *AUTOFIX_LIST*, *WEBLATE_ADDONS*, INSTALLED_APPS

Container-Einstellungen

WEBLATE_WORKERS

Neu in Version 4.6.1.

Basisanzahl der im Container laufenden Arbeitsprozesse. Wenn sie nicht festgelegt ist, wird sie automatisch beim Start des Containers anhand der Anzahl der verfügbaren CPU-Kerne ermittelt.

Wird zur Bestimmung von *CELERY_MAIN_OPTIONS*, *CELERY_NOTIFY_OPTIONS*, *CELERY_MEMORY_OPTIONS*, *CELERY_TRANSLATE_OPTIONS*, *CELERY_BACKUP_OPTIONS*, *CELERY_BEAT_OPTIONS* und *WEB_WORKERS* verwendet. Sie können diese Einstellungen zur Feinabstimmung nutzen.

CELERY_MAIN_OPTIONS

CELERY_NOTIFY_OPTIONS

CELERY_MEMORY_OPTIONS

CELERY_TRANSLATE_OPTIONS

CELERY_BACKUP_OPTIONS

CELERY_BEAT_OPTIONS

These variables allow you to adjust Celery worker options. It can be useful to adjust concurrency (`--concurrency 16`) or use different pool implementation (`--pool=gevent`).

By default, the number of concurrent workers is based on *WEBLATE_WORKERS*.

Beispiel:

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

Siehe auch:

Celery worker options, *Background tasks using Celery*

WEB_WORKERS

Configure how many uWSGI workers should be executed.

It defaults to `WEBLATE_WORKERS`.

Beispiel:

```
environment:
  WEB_WORKERS: 32
```

WEBLATE_SERVICE

Defines which services should be executed inside the container. Use this for *Horizontale Skalierung*.

Folgende Dienste sind definiert:

celery-beat
Celery-Aufgabenplaner, es sollte nur eine Instanz ausgeführt werden. Dieser Container ist auch für die Migrationen der Datenbankstruktur zuständig und sollte vor den anderen gestartet werden.

celery-backup
Celery-Worker für Backups, es sollte nur eine Instanz laufen.

celery-celery
Generischer Celery-Worker.

celery-memory
Übersetzungsspeicher Celery-Worker.

celery-notify
Benachrichtigungen Celery-Worker.

celery-translate
Automatische Übersetzung Celery-Worker.

web
Webserver.

Docker-Container-Volumes

Es gibt zwei Volumes (Daten und Cache), die vom Weblate-Container exportiert werden. Die anderen Service-Container (PostgreSQL oder Redis) verfügen ebenfalls über Datenvolumina, die jedoch in diesem Dokument nicht behandelt werden.

Das Datenvolumen wird verwendet, um persistente Weblate-Daten wie geklonte Repositorys zu speichern oder die Weblate-Installation anzupassen.

Die Platzierung des Docker-Volumes auf dem Host-System hängt von Ihrer Docker-Konfiguration ab, aber normalerweise wird es in `/var/lib/docker/volumes/weblate-docker_weblate-data/_data/` gespeichert (der Pfad besteht aus dem Namen Ihres Docker-Compose-Verzeichnisses, dem Container und den Volume-Namen). Im Container wird es als `/app/data` eingehängt.

Das Cache-Volume wird als `/app/cache` gemountet und dient der Speicherung statischer Dateien. Sein Inhalt wird beim Start des Containers neu erstellt und das Volume kann mit einem ephemeren Dateisystem wie *tmpfs* eingebunden werden.

Wenn Sie die Volumes manuell erstellen, sollten die Verzeichnisse der UID 1000 gehören, da dies der im Container verwendete Benutzer ist.

Siehe auch:

Dokumentation der Docker-Volumes

Weitere Anpassungen der Konfiguration

Sie können die Installation von Weblate im Datenvolumen weiter anpassen, siehe *Docker-Container-Volumes*.

Benutzerdefinierte Konfigurationsdateien

Sie können die Konfiguration zusätzlich in `/app/data/settings-override.py` überschreiben (siehe *Docker-Container-Volumes*). Dies wird am Ende der integrierten Einstellungen ausgeführt, nachdem alle Umgebungseinstellungen geladen wurden, und Sie können sie anpassen oder überschreiben.

Ersetzen des Logos und anderer statischer Dateien

Neu in Version 3.8-5.

Die mit Weblate gelieferten statischen Dateien können überschrieben werden, indem sie in `/app/data/python/customize/static` abgelegt werden (siehe *Docker-Container-Volumes*). Zum Beispiel wird durch die Erstellung von `/app/data/python/customize/static/favicon.ico` das Favicon ersetzt.

Hinweis: Die Dateien werden beim Start des Containers an den entsprechenden Ort kopiert, so dass ein Neustart von Weblate erforderlich ist, wenn der Inhalt des Volumes geändert wurde.

This approach can be also used to override Weblate templates. For example *Rechtliche Grundlagen* documents can be placed into `/app/data/python/customize/templates/legal/documents`.

Alternativ können Sie auch ein eigenes Modul einbinden (siehe *Customizing Weblate*) und es z.B. als separates Volume dem Docker-Container hinzufügen:

```
weblate:
  volumes:
    - weblate-data:/app/data
    - ./weblate_customization/weblate_customization:/app/data/python/weblate_
↪customization
  environment:
    WEBLATE_ADD_APPS: weblate_customization
```

Hinzufügen eigener Python-Module

Neu in Version 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 *Benutzerdefinierte Konfigurationsdateien*.

Siehe auch:

Customizing Weblate

Configuring PostgreSQL server

The PostgtreSQL container uses default PostgreSQL configuration and it won't effectively utilize your CPU cores or memory. It is recommended to customize the configuration to improve the performance.

The configuration can be adjusted as described in *Database Configuration* at https://hub.docker.com/_/postgres. The configuration matching your environment can be generated using https://pgtune.leopard.in.ua/.

Installing on Debian and Ubuntu

Hardware requirements

Weblate should run on any contemporary hardware without problems, the following is the minimal configuration required to run Weblate on a single host (Weblate, database and webserver):

- 2 GB of RAM
- 2 CPU cores
- 1 GB of storage space

The more memory the better - it is used for caching on all levels (filesystem, database and Weblate).

Many concurrent users increases the amount of needed CPU cores. For hundreds of translation components at least 4 GB of RAM is recommended.

The typical database storage usage is around 300 MB per 1 million hosted words. Storage space needed for cloned repositories varies, but Weblate tries to keep their size minimal by doing shallow clones.

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

Installation

Systemvoraussetzungen

Install the dependencies needed to build the Python modules (see *Software-Anforderungen*):

```
apt install -y \
   libxml2-dev libxslt-dev libfreetype6-dev libjpeg-dev libz-dev libyaml-dev \
   libffi-dev libcairo-dev gir1.2-pango-1.0 libgirepository1.0-dev \
   libacl1-dev libssl-dev libpq-dev libjpeg62-turbo-dev build-essential \
   python3-gdbm python3-dev python3-pip python3-virtualenv virtualenv git
```

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

```
apt install -y \
   tesseract-ocr libtesseract-dev libleptonica-dev \
   libldap2-dev libldap-common libsasl2-dev \
   libxmlsec1-dev
```

Optionally install software for running production server, see *Running server*, *Database setup for Weblate*, *Background tasks using Celery*. Depending on size of your installation you might want to run these components on dedicated servers.

The local installation instructions:

```
# Web server option 1: NGINX and uWSGI
apt install -y nginx uwsgi uwsgi-plugin-python3

# Web server option 2: Apache with ``mod_wsgi``
apt install -y apache2 libapache2-mod-wsgi-py3

# Caching backend: Redis
apt install -y redis-server

# Database server: PostgreSQL
apt install -y postgresql postgresql-contrib

# SMTP server
apt install -y exim4
```

Python modules

Hinweis: 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 ~/weblate-env
```

2. Activate the virtualenv for Weblate:

```
. ~/weblate-env/bin/activate
```

3. Install Weblate including all optional dependencies:

```
# Install Weblate with all optional dependencies
pip install "Weblate[all]"
```

Please check *Optional dependencies* for fine-tuning of optional dependencies.

Bemerkung: On some Linux distributions running Weblate fails with libffi error:

```
ffi_prep_closure(): bad user_data (it seems that the version of the libffi␣
↪library seen at runtime is different from the 'ffi.h' file seen at compile-
↪time)
```

This is caused by incompatibility of binary packages distributed via PyPI with the distribution. To address this, you need to rebuild the package on your system:

```
pip install --force-reinstall --no-binary :all: cffi
```

Configuring Weblate

Bemerkung: Following steps assume virtualenv used by Weblate is active (what can be done by `. ~/weblate-env/bin/activate`). In case this is not true, you will have to specify full path to **`weblate`** command as `~/weblate-env/bin/weblate`.

1. Copy the file `~/weblate-env/lib/python3.9/site-packages/weblate/settings_example.py` to `~/weblate-env/lib/python3.9/site-packages/weblate/settings.py`.
2. Passen Sie die Werte in der neuen Datei `settings.py` nach Ihren Wünschen an. Sie müssen zumindest die Datenbank-Zugangsdaten und den geheimen Django-Schlüssel angeben, aber Sie werden mehr Änderungen für die Produktionseinrichtung benötigen, siehe *Adjusting configuration*.
3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check *Database setup for Weblate* for production ready setup):

```
weblate migrate
```

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

```
weblate createadmin
```

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

```
weblate collectstatic
```

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

```
weblate compress
```

7. Start Celery workers. This is not necessary for development purposes, but strongly recommended otherwise. See *Background tasks using Celery* for more info:

```
~/weblate-env/lib/python3.9/site-packages/weblate/examples/celery start
```

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

```
weblate runserver
```

After installation

Congratulations, your Weblate server is now running and you can start using it.

- You can now access Weblate on `http://localhost:8000/`.
- Sign in with admin credentials obtained during installation or register with new users.
- You can now run Weblate commands using **`weblate`** command when Weblate virtualenv is active, see *Management commands*.
- You can stop the test server with Ctrl+C.
- Review potential issues with your installation either on `/manage/performance/` URL (see *Verwaltungsoberfläche*) 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: *Name der Komponente*, *Quellcode-Repository*, and *Dateimaske* for finding translatable files. Weblate supports a wide range of formats including *GNU gettext*, *Android string resources*, *Apple iOS strings*, *Java properties*, *Stringsdict-Format* or *Fluent-Format*, see *Supported file formats* for more details.

3. Once the above is completed (it can be lengthy process depending on the size of your VCS repository, and number of messages to translate), you can start translating.

Installing on SUSE and openSUSE

Hardware requirements

Weblate should run on any contemporary hardware without problems, the following is the minimal configuration required to run Weblate on a single host (Weblate, database and webserver):

- 2 GB of RAM
- 2 CPU cores
- 1 GB of storage space

The more memory the better - it is used for caching on all levels (filesystem, database and Weblate).

Many concurrent users increases the amount of needed CPU cores. For hundreds of translation components at least 4 GB of RAM is recommended.

The typical database storage usage is around 300 MB per 1 million hosted words. Storage space needed for cloned repositories varies, but Weblate tries to keep their size minimal by doing shallow clones.

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

Installation

Systemvoraussetzungen

Install the dependencies needed to build the Python modules (see *Software-Anforderungen*):

```
zypper install \
   libxslt-devel libxml2-devel freetype-devel libjpeg-devel zlib-devel \
   libyaml-devel libffi-devel cairo-devel pango-devel \
   gobject-introspection-devel libacl-devel python3-pip python3-virtualenv \
   python3-devel git
```

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

```
zypper install tesseract-ocr tesseract-devel leptonica-devel
zypper install libldap2-devel libsasl2-devel
zypper install libxmlsec1-devel
```

Optionally install software for running production server, see *Running server*, *Database setup for Weblate*, *Background tasks using Celery*. Depending on size of your installation you might want to run these components on dedicated servers.

The local installation instructions:

```
# Web server option 1: NGINX and uWSGI
zypper install nginx uwsgi uwsgi-plugin-python3

# Web server option 2: Apache with ``mod_wsgi``
zypper install apache2 apache2-mod_wsgi

# Caching backend: Redis
zypper install redis-server

# Database server: PostgreSQL
zypper install postgresql postgresql-contrib

# SMTP server
zypper install postfix
```

Python modules

Hinweis: 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 ~/weblate-env
   ```

2. Activate the virtualenv for Weblate:

   ```
   . ~/weblate-env/bin/activate
   ```

3. Install Weblate including all optional dependencies:

   ```
   # Install Weblate with all optional dependencies
   pip install "Weblate[all]"
   ```

   Please check *Optional dependencies* for fine-tuning of optional dependencies.

   Bemerkung: On some Linux distributions running Weblate fails with libffi error:

   ```
   ffi_prep_closure(): bad user_data (it seems that the version of the libffi␣
   ↪library seen at runtime is different from the 'ffi.h' file seen at compile-
   ↪time)
   ```

   This is caused by incompatibility of binary packages distributed via PyPI with the distribution. To address this, you need to rebuild the package on your system:

   ```
   pip install --force-reinstall --no-binary :all: cffi
   ```

Configuring Weblate

Bemerkung: Following steps assume virtualenv used by Weblate is active (what can be done by `. ~/weblate-env/bin/activate`). In case this is not true, you will have to specify full path to **`weblate`** command as `~/weblate-env/bin/weblate`.

1. Copy the file `~/weblate-env/lib/python3.9/site-packages/weblate/settings_example.py` to `~/weblate-env/lib/python3.9/site-packages/weblate/settings.py`.
2. Passen Sie die Werte in der neuen Datei `settings.py` nach Ihren Wünschen an. Sie müssen zumindest die Datenbank-Zugangsdaten und den geheimen Django-Schlüssel angeben, aber Sie werden mehr Änderungen für die Produktionseinrichtung benötigen, siehe *Adjusting configuration*.
3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check *Database setup for Weblate* for production ready setup):

```
weblate migrate
```

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

```
weblate createadmin
```

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

```
weblate collectstatic
```

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

```
weblate compress
```

7. Start Celery workers. This is not necessary for development purposes, but strongly recommended otherwise. See *Background tasks using Celery* for more info:

```
~/weblate-env/lib/python3.9/site-packages/weblate/examples/celery start
```

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

```
weblate runserver
```

After installation

Congratulations, your Weblate server is now running and you can start using it.

- You can now access Weblate on `http://localhost:8000/`.
- Sign in with admin credentials obtained during installation or register with new users.
- You can now run Weblate commands using **`weblate`** command when Weblate virtualenv is active, see *Management commands*.
- You can stop the test server with Ctrl+C.
- Review potential issues with your installation either on `/manage/performance/` URL (see *Verwaltungsoberfläche*) 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: *Name der Komponente*, *Quellcode-Repository*, and *Dateimaske* for finding translatable files. Weblate supports a wide range of formats including *GNU gettext*, *Android string resources*, *Apple iOS strings*, *Java properties*, *Stringsdict-Format* or *Fluent-Format*, see *Supported file formats* for more details.

3. Once the above is completed (it can be lengthy process depending on the size of your VCS repository, and number of messages to translate), you can start translating.

Installing on RedHat, Fedora and CentOS

Hardware requirements

Weblate should run on any contemporary hardware without problems, the following is the minimal configuration required to run Weblate on a single host (Weblate, database and webserver):

- 2 GB of RAM
- 2 CPU cores
- 1 GB of storage space

The more memory the better - it is used for caching on all levels (filesystem, database and Weblate).

Many concurrent users increases the amount of needed CPU cores. For hundreds of translation components at least 4 GB of RAM is recommended.

The typical database storage usage is around 300 MB per 1 million hosted words. Storage space needed for cloned repositories varies, but Weblate tries to keep their size minimal by doing shallow clones.

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

Installation

Systemvoraussetzungen

Install the dependencies needed to build the Python modules (see *Software-Anforderungen*):

```
dnf install \
   libxslt-devel libxml2-devel freetype-devel libjpeg-devel zlib-devel \
   libyaml-devel libffi-devel cairo-devel pango-devel \
   gobject-introspection-devel libacl-devel python3-pip python3-virtualenv \
   python3-devel git
```

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

```
dnf install tesseract-langpack-eng tesseract-devel leptonica-devel
dnf install libldap2-devel libsasl2-devel
dnf install libxmlsec1-devel
```

Optionally install software for running production server, see *Running server*, *Database setup for Weblate*, *Background tasks using Celery*. Depending on size of your installation you might want to run these components on dedicated servers.

The local installation instructions:

```
# Web server option 1: NGINX and uWSGI
dnf install nginx uwsgi uwsgi-plugin-python3

# Web server option 2: Apache with ``mod_wsgi``
dnf install apache2 apache2-mod_wsgi

# Caching backend: Redis
dnf install redis

# Database server: PostgreSQL
dnf install postgresql postgresql-contrib

# SMTP server
dnf install postfix
```

Python modules

Hinweis: 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 ~/weblate-env
   ```

2. Activate the virtualenv for Weblate:

   ```
   . ~/weblate-env/bin/activate
   ```

3. Install Weblate including all optional dependencies:

   ```
   # Install Weblate with all optional dependencies
   pip install "Weblate[all]"
   ```

   Please check *Optional dependencies* for fine-tuning of optional dependencies.

   Bemerkung: On some Linux distributions running Weblate fails with libffi error:

   ```
   ffi_prep_closure(): bad user_data (it seems that the version of the libffi␣
   ↪library seen at runtime is different from the 'ffi.h' file seen at compile-
   ↪time)
   ```

   This is caused by incompatibility of binary packages distributed via PyPI with the distribution. To address this, you need to rebuild the package on your system:

   ```
   pip install --force-reinstall --no-binary :all: cffi
   ```

Configuring Weblate

Bemerkung: Following steps assume virtualenv used by Weblate is active (what can be done by `. ~/weblate-env/bin/activate`). In case this is not true, you will have to specify full path to **`weblate`** command as `~/weblate-env/bin/weblate`.

1. Copy the file `~/weblate-env/lib/python3.9/site-packages/weblate/settings_example.py` to `~/weblate-env/lib/python3.9/site-packages/weblate/settings.py`.
2. Passen Sie die Werte in der neuen Datei `settings.py` nach Ihren Wünschen an. Sie müssen zumindest die Datenbank-Zugangsdaten und den geheimen Django-Schlüssel angeben, aber Sie werden mehr Änderungen für die Produktionseinrichtung benötigen, siehe *Adjusting configuration*.
3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check *Database setup for Weblate* for production ready setup):

```
weblate migrate
```

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

```
weblate createadmin
```

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

```
weblate collectstatic
```

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

```
weblate compress
```

7. Start Celery workers. This is not necessary for development purposes, but strongly recommended otherwise. See *Background tasks using Celery* for more info:

```
~/weblate-env/lib/python3.9/site-packages/weblate/examples/celery start
```

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

```
weblate runserver
```

After installation

Congratulations, your Weblate server is now running and you can start using it.

- You can now access Weblate on `http://localhost:8000/`.
- Sign in with admin credentials obtained during installation or register with new users.
- You can now run Weblate commands using **`weblate`** command when Weblate virtualenv is active, see *Management commands*.
- You can stop the test server with Ctrl+C.
- Review potential issues with your installation either on `/manage/performance/` URL (see *Verwaltungsoberfläche*) 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: *Name der Komponente*, *Quellcode-Repository*, and *Dateimaske* for finding translatable files. Weblate supports a wide range of formats including *GNU gettext*, *Android string resources*, *Apple iOS strings*, *Java properties*, *Stringsdict-Format* or *Fluent-Format*, see *Supported file formats* for more details.

3. Once the above is completed (it can be lengthy process depending on the size of your VCS repository, and number of messages to translate), you can start translating.

Installing on macOS

Hardware requirements

Weblate should run on any contemporary hardware without problems, the following is the minimal configuration required to run Weblate on a single host (Weblate, database and webserver):

- 2 GB of RAM
- 2 CPU cores
- 1 GB of storage space

The more memory the better - it is used for caching on all levels (filesystem, database and Weblate).

Many concurrent users increases the amount of needed CPU cores. For hundreds of translation components at least 4 GB of RAM is recommended.

The typical database storage usage is around 300 MB per 1 million hosted words. Storage space needed for cloned repositories varies, but Weblate tries to keep their size minimal by doing shallow clones.

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

Installation

Systemvoraussetzungen

Install the dependencies needed to build the Python modules (see *Software-Anforderungen*):

```
brew install python pango cairo gobject-introspection libffi glib libyaml
pip install virtualenv
```

Make sure pip will be able to find the `libffi` version provided by homebrew — this will be needed during the installation build step.

```
export PKG_CONFIG_PATH="/usr/local/opt/libffi/lib/pkgconfig"
```

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

```
brew install tesseract
```

Optionally install software for running production server, see *Running server*, *Database setup for Weblate*, *Background tasks using Celery*. Depending on size of your installation you might want to run these components on dedicated servers.

The local installation instructions:

```
# Web server option 1: NGINX and uWSGI
brew install nginx uwsgi

# Web server option 2: Apache with ``mod_wsgi``
brew install httpd

# Caching backend: Redis
brew install redis

# Database server: PostgreSQL
brew install postgresql
```

Python modules

Hinweis: 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 ~/weblate-env
   ```

2. Activate the virtualenv for Weblate:

   ```
   . ~/weblate-env/bin/activate
   ```

3. Install Weblate including all optional dependencies:

   ```
   # Install Weblate with all optional dependencies
   pip install "Weblate[all]"
   ```

   Please check *Optional dependencies* for fine-tuning of optional dependencies.

   Bemerkung: On some Linux distributions running Weblate fails with libffi error:

   ```
   ffi_prep_closure(): bad user_data (it seems that the version of the libffi␣
   ↪library seen at runtime is different from the 'ffi.h' file seen at compile-
   ↪time)
   ```

   This is caused by incompatibility of binary packages distributed via PyPI with the distribution. To address this, you need to rebuild the package on your system:

   ```
   pip install --force-reinstall --no-binary :all: cffi
   ```

Configuring Weblate

Bemerkung: Following steps assume virtualenv used by Weblate is active (what can be done by `. ~/weblate-env/bin/activate`). In case this is not true, you will have to specify full path to **`weblate`** command as `~/weblate-env/bin/weblate`.

1. Copy the file `~/weblate-env/lib/python3.9/site-packages/weblate/settings_example.py` to `~/weblate-env/lib/python3.9/site-packages/weblate/settings.py`.
2. Passen Sie die Werte in der neuen Datei `settings.py` nach Ihren Wünschen an. Sie müssen zumindest die Datenbank-Zugangsdaten und den geheimen Django-Schlüssel angeben, aber Sie werden mehr Änderungen für die Produktionseinrichtung benötigen, siehe *Adjusting configuration*.
3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check *Database setup for Weblate* for production ready setup):

```
weblate migrate
```

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

```
weblate createadmin
```

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

```
weblate collectstatic
```

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

```
weblate compress
```

7. Start Celery workers. This is not necessary for development purposes, but strongly recommended otherwise. See *Background tasks using Celery* for more info:

```
~/weblate-env/lib/python3.9/site-packages/weblate/examples/celery start
```

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

```
weblate runserver
```

After installation

Congratulations, your Weblate server is now running and you can start using it.

- You can now access Weblate on `http://localhost:8000/`.
- Sign in with admin credentials obtained during installation or register with new users.
- You can now run Weblate commands using **`weblate`** command when Weblate virtualenv is active, see *Management commands*.
- You can stop the test server with Ctrl+C.
- Review potential issues with your installation either on `/manage/performance/` URL (see *Verwaltungsoberfläche*) 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: *Name der Komponente*, *Quellcode-Repository*, and *Dateimaske* for finding translatable files. Weblate supports a wide range of formats including *GNU gettext*, *Android string resources*, *Apple iOS strings*, *Java properties*, *Stringsdict-Format* or *Fluent-Format*, see *Supported file formats* for more details.

3. Once the above is completed (it can be lengthy process depending on the size of your VCS repository, and number of messages to translate), you can start translating.

Installieren aus Quellen

1. Please follow the installation instructions for your system first up to installing Weblate:

   - *Installing on Debian and Ubuntu*
   - *Installing on SUSE and openSUSE*
   - *Installing on RedHat, Fedora and CentOS*

2. Grab the latest Weblate sources using Git (or download a tarball and unpack that):

```
git clone https://github.com/WeblateOrg/weblate.git weblate-src
```

   Alternatively you can use released archives. You can download them from our website <https://weblate.org/>. Those downloads are cryptographically signed, please see *Verifying release signatures*.

3. Install current Weblate code into the virtualenv:

```
. ~/weblate-env/bin/activate
pip install -e weblate-src
```

4. Copy `weblate/settings_example.py` to `weblate/settings.py`.

5. Passen Sie die Werte in der neuen Datei `settings.py` nach Ihren Wünschen an. Sie müssen zumindest die Datenbank-Zugangsdaten und den geheimen Django-Schlüssel angeben, aber Sie werden mehr Änderungen für die Produktionseinrichtung benötigen, siehe *Adjusting configuration*.

6. Create the database used by Weblate, see *Database setup for Weblate*.

7. Build Django tables, static files and initial data (see *Filling up the database* and *Serving static files*):

```
weblate migrate
weblate collectstatic
weblate compress
```

   Bemerkung: This step should be repeated whenever you update the repository.

Installing on OpenShift

With the OpenShift Weblate template you can get your personal Weblate instance up and running in seconds. All of Weblate's dependencies are already included. PostgreSQL is set up as the default database and persistent volume claims are used.

You can find the template at <https://github.com/WeblateOrg/openshift/>.

Installation

The following examples assume you have a working OpenShift v3.x environment, with `oc` client tool installed. Please check the OpenShift documentation for instructions.

The `template.yml` is suited for running all components in OpenShift. There is also `template-external-postgresql.yml` which does not start a PostgreSQL server and allows you to configure external PostgreSQL server.

Webkonsole

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

Um die Weblate-Vorlage in die Vorlagenbibliothek Ihres aktuellen Projekts hochzuladen, übergeben Sie die Datei `template.yml` mit dem folgenden Befehl:

```
$ oc create -f https://raw.githubusercontent.com/WeblateOrg/openshift/main/
↪template.yml \
   -n <PROJECT>
```

Die Vorlage kann nun über die Webkonsole oder das CLI ausgewählt werden.

Parameter

The parameters that you can override are listed in the parameters section of the template. You can list them with the CLI by using the following command and specifying the file to be used:

```
$ oc process --parameters -f https://raw.githubusercontent.com/WeblateOrg/
↪openshift/main/template.yml

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

Bereitstellung

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

```
$ oc process -f https://raw.githubusercontent.com/WeblateOrg/openshift/main/
↪template.yml \
    -p APPLICATION_NAME=weblate \
    -p WEBLATE_VERSION=4.3.1-1 \
    -p WEBLATE_SITE_DOMAIN=weblate.app-openshift.example.com \
    -p POSTGRESQL_IMAGE=docker-registry.default.svc:5000/openshift/postgresql:9.6 \
    -p REDIS_IMAGE=docker-registry.default.svc:5000/openshift/redis:3.2 \
    | oc create -f
```

The Weblate instance should be available after successful migration and deployment at the specified `WEBLATE_SITE_DOMAIN` parameter.

After container setup, you can sign in as *admin* user with password provided in `WEBLATE_ADMIN_PASSWORD`, or a random password generated on first start if that was not set.

To reset *admin* password, restart the container with `WEBLATE_ADMIN_PASSWORD` set to new password in the respective `Secret`.

Beseitigen

```
$ oc delete all -l app=<APPLICATION_NAME>
$ oc delete configmap -l app= <APPLICATION_NAME>
$ oc delete secret -l app=<APPLICATION_NAME>
# ATTTENTION! The following command is only optional and will permanently delete␣
↪all of your data.
$ oc delete pvc -l app=<APPLICATION_NAME>

$ oc delete all -l app=weblate \
    && oc delete secret -l app=weblate \
    && oc delete configmap -l app=weblate \
    && oc delete pvc -l app=weblate
```

Konfiguration

Durch die Verarbeitung des Templates wird eine entsprechende `ConfigMap` erstellt, die zur Anpassung des Weblate-Bildes verwendet werden kann. Die `ConfigMap` wird direkt als Umgebungsvariable eingebunden und löst bei jeder Änderung eine neue Bereitstellung aus. Für weitere Konfigurationsoptionen siehe *Docker-Umgebungsvariablen* für eine vollständige Liste der Umgebungsvariablen.

Installing on Kubernetes

Bemerkung: This guide is looking for contributors experienced with Kubernetes to cover the setup in more details.

With the Kubernetes Helm chart you can get your personal Weblate instance up and running in seconds. All of Weblate's dependencies are already included. PostgreSQL is set up as the default database and persistent volume claims are used.

You can find the chart at <https://github.com/WeblateOrg/helm/> and it can be displayed at <https://artifacthub.io/packages/helm/weblate/weblate>.

Installation

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

Konfiguration

Für weitere Konfigurationsoptionen siehe *Docker-Umgebungsvariablen* für eine vollständige Liste der Umgebungsvariablen.

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*
- *Installieren aus Quellen*, recommended for development.
- *Installing on OpenShift*
- *Installing on Kubernetes*

2.1.2 Software-Anforderungen

Betriebssystem

Weblate is known to work on Linux, FreeBSD and macOS. Other Unix like systems will most likely work too.

Weblate is not supported on Windows. But it may still work and patches are happily accepted.

Other services

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

- PostgreSQL database server, see *Database setup for Weblate*.
- Redis server for cache and tasks queue, see *Background tasks using Celery*.
- SMTP server for outgoing e-mail, see *Configuring outgoing e-mail*.

Python-Abhängigkeiten

Weblate is written in Python and supports Python 3.6 or newer. You can install dependencies using pip or from your distribution packages, full list is available in `requirements.txt`.

Most notable dependencies:

Django
https://www.djangoproject.com/

Celery
https://docs.celeryq.dev/

Translate Toolkit
https://toolkit.translatehouse.org/

translation-finder
https://github.com/WeblateOrg/translation-finder

Python Social Auth
https://python-social-auth.readthedocs.io/

Django REST Framework
https://www.django-rest-framework.org/

Optional dependencies

Following modules are necessary for some Weblate features. You can find all of them in `requirements-optional.txt`.

`Mercurial` (optional for *Mercurial* repositories support)
https://www.mercurial-scm.org/

`phply` (optional for *PHP-Zeichenketten*)
https://github.com/viraptor/phply

`tesserocr` (optional für OCR in Bildschirmfotos)
https://github.com/sirfz/tesserocr

`python-akismet` (optional for *Spam protection*)
https://github.com/Nekmo/python-akismet

`ruamel.yaml` (optional for *YAML files*)
https://pypi.org/project/ruamel.yaml/

`Zeep` (optional for *Microsoft Terminology*)
https://docs.python-zeep.org/

`aeidon` (optional for *Subtitle files*)
https://pypi.org/project/aeidon/

`fluent.syntax` (optional für *Fluent-Format*)
https://projectfluent.org/

Hinweis: When installing using pip, you can directly specify desired features when installing:

```
pip install "Weblate[PHP,Fluent]"
```

Or you can install Weblate with all optional features:

```
pip install "Weblate[all]"
```

Or you can install Weblate without any optional features:

```
pip install Weblate
```

Database backend dependencies

Weblate supports PostgreSQL, MySQL and MariaDB, see *Database setup for Weblate* and backends documentation for more details.

Other system requirements

The following dependencies have to be installed on the system:

`Git`
https://git-scm.com/

Pango, Cairo and related header files and GObject introspection data
https://cairographics.org/, https://pango.gnome.org/, see *Pango and Cairo*

`git-review` (optional for Gerrit support)
https://pypi.org/project/git-review/

`git-svn` (optional for Subversion support)
https://git-scm.com/docs/git-svn

`tesseract` und seine Daten (optional für Bildschirmfotos-OCR)
https://github.com/tesseract-ocr/tesseract

`licensee` (optional for detecting license when creating component)
https://github.com/licensee/licensee

Build-time dependencies

To build some of the *Python-Abhängigkeiten* you might need to install their dependencies. This depends on how you install them, so please consult individual packages for documentation. You won't need those if using prebuilt `Wheels` while installing using `pip` or when you use distribution packages.

Pango and Cairo

Geändert in Version 3.7.

Weblate uses Pango and Cairo for rendering bitmap widgets (see promotion) and rendering checks (see *Managing fonts*). To properly install Python bindings for those you need to install system libraries first - you need both Cairo and Pango, which in turn need GLib. All those should be installed with development files and GObject introspection data.

2.1.3 Verifying release signatures

Weblate release are cryptographically signed by the releasing developer. Currently this is Michal Čihař. Fingerprint of his PGP key is:

```
63CB 1DF1 EF12 CF2A C0EE 5A32 9C27 B313 42B7 511D
```

and you can get more identification information from <https://keybase.io/nijel>.

You should verify that the signature matches the archive you have downloaded. This way you can be sure that you are using the same code that was released. You should also verify the date of the signature to make sure that you downloaded the latest version.

Each archive is accompanied with `.asc` files which contain the PGP signature for it. Once you have both of them in the same folder, you can verify the signature:

```
$ gpg --verify Weblate-3.5.tar.xz.asc
gpg: assuming signed data in 'Weblate-3.5.tar.xz'
gpg: Signature made Ne 3. března 2019, 16:43:15 CET
gpg:                using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: Can't check signature: public key not found
```

As you can see GPG complains that it does not know the public key. At this point you should do one of the following steps:

- Use *wkd* to download the key:

```
$ gpg --auto-key-locate wkd --locate-keys michal@cihar.com
pub   rsa4096 2009-06-17 [SC]
      63CB1DF1EF12CF2AC0EE5A329C27B31342B7511D
uid           [ultimate] Michal Čihař <michal@cihar.com>
uid           [ultimate] Michal Čihař <nijel@debian.org>
uid           [ultimate] [jpeg image of size 8848]
uid           [ultimate] Michal Čihař (Braiins) <michal.cihar@braiins.cz>
sub   rsa4096 2009-06-17 [E]
sub   rsa4096 2015-09-09 [S]
```

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

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

- Download and import the key from one of the key servers:

```
$ gpg --keyserver hkp://pgp.mit.edu --recv-keys␣
↪87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: key 9C27B31342B7511D: "Michal Čihař <michal@cihar.com>" imported
gpg: Total number processed: 1
gpg:              unchanged: 1
```

This will improve the situation a bit - at this point you can verify that the signature from the given key is correct but you still can not trust the name used in the key:

```
$ gpg --verify Weblate-3.5.tar.xz.asc
gpg: assuming signed data in 'Weblate-3.5.tar.xz'
gpg: Signature made Ne 3. března 2019, 16:43:15 CET
gpg:                using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: Good signature from "Michal Čihař <michal@cihar.com>" [ultimate]
gpg:                 aka "Michal Čihař <nijel@debian.org>" [ultimate]
gpg:                 aka "[jpeg image of size 8848]" [ultimate]
gpg:                 aka "Michal Čihař (Braiins) <michal.cihar@braiins.cz>"␣
↪[ultimate]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 63CB 1DF1 EF12 CF2A C0EE  5A32 9C27 B313 42B7 511D
```

The problem here is that anybody could issue the key with this name. You need to ensure that the key is actually owned by the mentioned person. The GNU Privacy Handbook covers this topic in the chapter Validating other keys on your public keyring. The most reliable method is to meet the developer in person and exchange key fingerprints, however you can also rely on the web of trust. This way you can trust the key transitively through signatures of others, who have met the developer in person.

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

```
$ gpg --verify Weblate-3.5.tar.xz.asc
gpg: assuming signed data in 'Weblate-3.5.tar.xz'
gpg: Signature made Sun Mar  3 16:43:15 2019 CET
gpg:                using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: Good signature from "Michal Čihař <michal@cihar.com>" [ultimate]
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
gpg:                 aka "Michal Čihař <nijel@debian.org>" [ultimate]
gpg:                 aka "[jpeg image of size 8848]" [ultimate]
gpg:                 aka "Michal Čihař (Braiins) <michal.cihar@braiins.cz>"␣
↪[ultimate]
```

Should the signature be invalid (the archive has been changed), you would get a clear error regardless of the fact that the key is trusted or not:

```
$ gpg --verify Weblate-3.5.tar.xz.asc
gpg: Signature made Sun Mar  3 16:43:15 2019 CET
gpg:                using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: BAD signature from "Michal Čihař <michal@cihar.com>" [ultimate]
```

2.1.4 Filesystem permissions

The Weblate process needs to be able to read and write to the directory where it keeps data - `DATA_DIR`. All files within this directory should be owned and writable by the user running all Weblate processes (typically WSGI and Celery, see *Running server* and *Background tasks using Celery*).

The default configuration places them in the same tree as the Weblate sources, however you might prefer to move these to a better location such as: `/var/lib/weblate`.

Weblate tries to create these directories automatically, but it will fail when it does not have permissions to do so.

You should also take care when running *Management commands*, as they should be ran under the same user as Weblate itself is running, otherwise permissions on some files might be wrong.

In the Docker container, all files in the `/app/data` volume have to be owned by the `weblate` user inside the container (UID 1000).

Siehe auch:

Serving static files

2.1.5 Database setup for Weblate

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

Siehe auch:

Use a powerful database engine, Databases, *Migration von anderen Datenbanken zu PostgreSQL*

PostgreSQL

PostgreSQL is usually the best choice for Django-based sites. It's the reference database used for implementing Django database layer.

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

Siehe auch:

PostgreSQL notes

Creating a database in PostgreSQL

It is usually a good idea to run Weblate in a separate database, and separate user account:

```
# If PostgreSQL was not installed before, set the main password
sudo -u postgres psql postgres -c "\password postgres"

# Create a database user called "weblate"
sudo -u postgres createuser --superuser --pwprompt weblate

# Create the database "weblate" owned by "weblate"
sudo -u postgres createdb -E UTF8 -O weblate weblate
```

Hinweis: If you don't want to make the Weblate user a superuser in PostgreSQL, you can omit that. In that case you will have to perform some of the migration steps manually as a PostgreSQL superuser in schema Weblate will use:

```
CREATE EXTENSION IF NOT EXISTS pg_trgm WITH SCHEMA weblate;
```

Configuring Weblate to use PostgreSQL

The `settings.py` snippet for PostgreSQL:

```
DATABASES = {
    "default": {
        # Database engine
        "ENGINE": "django.db.backends.postgresql",
        # Database name
        "NAME": "weblate",
        # Database user
        "USER": "weblate",
        # Name of role to alter to set parameters in PostgreSQL,
        # use in case role name is different than user used for authentication.
        # "ALTER_ROLE": "weblate",
        # Database password
        "PASSWORD": "password",
        # Set to empty string for localhost
        "HOST": "database.example.com",
        # Set to empty string for default
        "PORT": "",
    }
}
```

The database migration performs ALTER ROLE on database role used by Weblate. In most cases the name of the role matches username. In more complex setups the role name is different than username and you will get error about non-existing role during the database migration (`psycopg2.errors.UndefinedObject: role "weblate@hostname" does not exist`). This is known to happen with Azure Database for PostgreSQL, but it's not limited to this environment. Please set `ALTER_ROLE` to change name of the role Weblate should alter during the database migration.

MySQL and MariaDB

Hinweis: Some Weblate features will perform better with *PostgreSQL*. This includes searching and translation memory, which both utilize full-text features in the database and PostgreSQL implementation is superior.

Weblate can be also used with MySQL or MariaDB, please see MySQL notes and MariaDB notes for caveats using Django with those. Because of the limitations it is recommended to use *PostgreSQL* for new installations.

Weblate requires MySQL at least 5.7.8 or MariaDB at least 10.2.7.

Following configuration is recommended for Weblate:

- Use the `utf8mb4` charset to allow representation of higher Unicode planes (for example emojis).
- Configure the server with `innodb_large_prefix` to allow longer indices on text fields.
- Set the isolation level to `READ COMMITTED`.
- The SQL mode should be set to `STRICT_TRANS_TABLES`.

MySQL 8.x, MariaDB 10.5.x or newer have reasonable default configuration so that no server tweaking should be necessary and all what is needed can be configured on the client side.

Below is an example `/etc/my.cnf.d/server.cnf` for a server with 8 GB of RAM. These settings should be sufficient for most installs. MySQL and MariaDB have tunables that will increase the performance of your server that are considered not necessary unless you are planning on having large numbers of concurrent users accessing the system. See the various vendors documentation on those details.

It is absolutely critical to reduce issues when installing that the setting `innodb_file_per_table` is set properly and MySQL/MariaDB restarted before you start your Weblate install.

```
[mysqld]
character-set-server = utf8mb4
character-set-client = utf8mb4
collation-server = utf8mb4_unicode_ci

datadir=/var/lib/mysql

log-error=/var/log/mariadb/mariadb.log

innodb_large_prefix=1
innodb_file_format=Barracuda
innodb_file_per_table=1
innodb_buffer_pool_size=2G
sql_mode=STRICT_TRANS_TABLES
```

Hinweis: 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.

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

Weblate für die Verwendung von MySQL/MariaDB konfigurieren

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

```
DATABASES = {
    "default": {
        # Database engine
        "ENGINE": "django.db.backends.mysql",
        # Database name
        "NAME": "weblate",
        # Database user
        "USER": "weblate",
        # Database password
        "PASSWORD": "password",
        # Set to empty string for localhost
        "HOST": "127.0.0.1",
        # Set to empty string for default
        "PORT": "3306",
        # In case you wish to use additional
        # connection options
        "OPTIONS": {},
    }
}
```

You should also create the `weblate` user account in MySQL or MariaDB before you begin the install. Use the commands below to achieve that:

```
GRANT ALL ON weblate.* to 'weblate'@'localhost' IDENTIFIED BY 'password';
FLUSH PRIVILEGES;
```

2.1.6 Other configurations

Configuring outgoing e-mail

Weblate sends out e-mails on various occasions - for account activation and on various notifications configured by users. For this it needs access to an SMTP server.

The mail server setup is configured using these settings: `EMAIL_HOST`, `EMAIL_HOST_PASSWORD`, `EMAIL_USE_TLS`, `EMAIL_USE_SSL`, `EMAIL_HOST_USER` and `EMAIL_PORT`. Their names are quite self-explanatory, but you can find more info in the Django documentation.

Hinweis: 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.

Siehe auch:

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 *Audit-Protokoll*.

In default configuration Weblate parses IP address from `REMOTE_ADDR` which is set by the WSGI handler.

In case you are running a reverse proxy, this field will most likely contain its address. You need to configure Weblate to trust additional HTTP headers and parse the IP address from these. This can not be enabled by default as it would allow IP address spoofing for installations not using a reverse proxy. Enabling *IP_BEHIND_REVERSE_PROXY* might be enough for the most usual setups, but you might need to adjust *IP_PROXY_HEADER* and *IP_PROXY_OFFSET* as well.

Another thing to take care of is the Host header. It should match to whatever is configured as *SITE_DOMAIN*. Additional configuration might be needed in your reverse proxy (for example use `ProxyPreserveHost On` for Apache or `proxy_set_header Host $host;` with nginx).

Siehe auch:

Spam protection, *Rate limiting*, *Audit-Protokoll*, *IP_BEHIND_REVERSE_PROXY*, *IP_PROXY_HEADER*, *IP_PROXY_OFFSET*, `SECURE_PROXY_SSL_HEADER`

HTTP proxy

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

```
import os

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

Siehe auch:

Proxy Environment Variables

2.1.7 Adjusting configuration

Siehe auch:

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.
>
> **Siehe auch:**
>
> `ADMINS`, *Properly configure admins*

`ALLOWED_HOSTS`

> You need to set this to list the hosts your site is supposed to serve. For example:
>
> ```
> ALLOWED_HOSTS = ["demo.weblate.org"]
> ```
>
> Alternatively you can include wildcard:
>
> ```
> ALLOWED_HOSTS = ["*"]
> ```
>
> **Siehe auch:**
>
> `ALLOWED_HOSTS`, *WEBLATE_ALLOWED_HOSTS*, *Allowed hosts setup*

SESSION_ENGINE

Configure how your sessions will be stored. In case you keep the default database backend engine, you should schedule: **weblate clearsessions** to remove stale session data from the database.

If you are using Redis as cache (see *Enable caching*) it is recommended to use it for sessions as well:

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

Siehe auch:

Configuring the session engine, SESSION_ENGINE

DATABASES

Connectivity to database server, please check Django's documentation for more details.

Siehe auch:

Database setup for Weblate, DATABASES, Databases

DEBUG

Disable this for any production server. With debug mode enabled, Django will show backtraces in case of error to users, when you disable it, errors will be sent per e-mail to ADMINS (see above).

Debug mode also slows down Weblate, as Django stores much more info internally in this case.

Siehe auch:

DEBUG, *Disable debug mode*

DEFAULT_FROM_EMAIL

E-mail sender address for outgoing e-mail, for example registration e-mails.

Siehe auch:

DEFAULT_FROM_EMAIL

SECRET_KEY

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

Siehe auch:

SECRET_KEY

SERVER_EMAIL

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

Siehe auch:

SERVER_EMAIL

2.1.8 Filling up the database

After your configuration is ready, you can run `weblate migrate` to create the database structure. Now you should be able to create translation projects using the admin interface.

In case you want to run an installation non interactively, you can use `weblate migrate --noinput`, and then create an admin user using `createadmin` command.

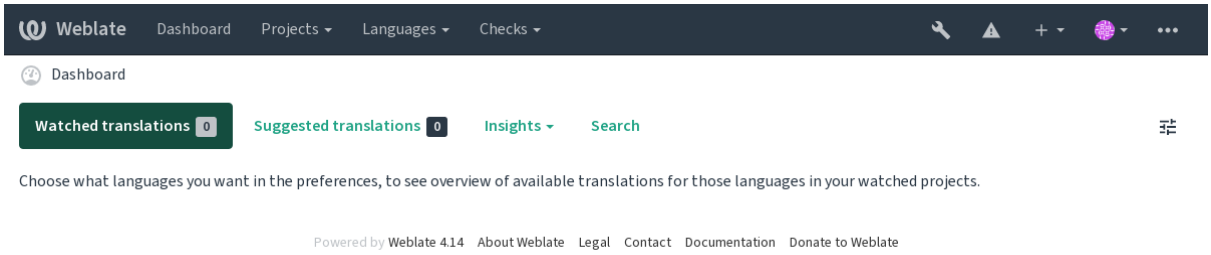
Once you are done, you should also check the *Performance report* in the admin interface, which will give you hints of potential non optimal configuration on your site.

Siehe auch:

Konfiguration, *Liste der Berechtigungen und integrierten Rollen*

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 *Verwaltungsoberfläche*.

Siehe auch:

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.

Siehe auch:

Adjusting configuration

Properly configure admins

Set the correct admin addresses to the `ADMINS` setting to defining who will receive e-mails in case something goes wrong on the server, for example:

```
ADMINS = (("Your Name", "your_email@example.com"),)
```

Siehe auch:

Adjusting configuration

Set correct site domain

Adjust site name and domain in the admin interface, otherwise links in RSS or registration e-mails will not work. This is configured using `SITE_DOMAIN` which should contain site domain name.

Geändert in Version 4.2: Prior to the 4.2 release the Django sites framework was used instead, please see The "sites" framework.

Siehe auch:

Allowed hosts setup, *Correctly configure HTTPS* `SITE_DOMAIN`, `WEBLATE_SITE_DOMAIN`, `ENABLE_HTTPS`

Correctly configure HTTPS

It is strongly recommended to run Weblate using the encrypted HTTPS protocol. After enabling it, you should set `ENABLE_HTTPS` in the settings:

```
ENABLE_HTTPS = True
```

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

Siehe auch:

`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.

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

Use a powerful database engine

- Please use PostgreSQL for a production environment, see *Database setup for Weblate* for more info.
- Verwenden Sie einen benachbarten Standort für den Betrieb des Datenbankservers, da sonst die Netzwerkleistung oder -zuverlässigkeit Ihr Weblate-Erlebnis beeinträchtigen könnte.
- Überprüfen Sie die Leistung des Datenbankservers oder passen Sie seine Konfiguration an, z. B. mit PGTune.

Siehe auch:

Database setup for Weblate, *Migration von anderen Datenbanken zu PostgreSQL*, *Adjusting configuration*, Databases

Enable caching

If possible, use Redis from Django by adjusting the `CACHES` configuration variable, for example:

```
CACHES = {
    "default": {
        "BACKEND": "django_redis.cache.RedisCache",
        "LOCATION": "redis://127.0.0.1:6379/0",
        # If redis is running on same host as Weblate, you might
        # want to use unix sockets instead:
        # 'LOCATION': 'unix:///var/run/redis/redis.sock?db=0',
        "OPTIONS": {
            "CLIENT_CLASS": "django_redis.client.DefaultClient",
            "PARSER_CLASS": "redis.connection.HiredisParser",
        },
    }
}
```

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

Siehe auch:

Avatar-Zwischenspeicherung, Django's cache framework

Avatar-Zwischenspeicherung

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

```
CACHES = {
    "default": {
        # Default caching backend setup, see above
        "BACKEND": "django_redis.cache.RedisCache",
        "LOCATION": "unix:///var/run/redis/redis.sock?db=0",
        "OPTIONS": {
            "CLIENT_CLASS": "django_redis.client.DefaultClient",
            "PARSER_CLASS": "redis.connection.HiredisParser",
        },
    },
    "avatar": {
        "BACKEND": "django.core.cache.backends.filebased.FileBasedCache",
        "LOCATION": os.path.join(DATA_DIR, "avatar-cache"),
        "TIMEOUT": 604800,
        "OPTIONS": {
            "MAX_ENTRIES": 1000,
        },
    },
}
```

Siehe auch:

`ENABLE_AVATARS`, `AVATAR_URL_PREFIX`, *Avatars*, *Enable caching*, Django's cache framework

Configure e-mail sending

Weblate needs to send out e-mails on several occasions, and these e-mails should have a correct sender address, please configure `SERVER_EMAIL` and `DEFAULT_FROM_EMAIL` to match your environment, for example:

```
SERVER_EMAIL = "admin@example.org"
DEFAULT_FROM_EMAIL = "weblate@example.org"
```

Bemerkung: 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.

Siehe auch:

Adjusting configuration, *Configuring outgoing e-mail*, `EMAIL_BACKEND`, `DEFAULT_FROM_EMAIL`, `SERVER_EMAIL`

Allowed hosts setup

Django requires *ALLOWED_HOSTS* to hold a list of domain names your site is allowed to serve, leaving it empty will block any requests.

In case this is not configured to match your HTTP server, you will get errors like `Invalid HTTP_HOST header: '1.1.1.1'. You may need to add '1.1.1.1' to ALLOWED_HOSTS.`

Hinweis: On Docker container, this is available as *WEBLATE_ALLOWED_HOSTS*.

Siehe auch:

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-generate-secret-key** shipped with Weblate.

Siehe auch:

SECRET_KEY

Template loading

It is recommended to use a cached template loader for Django. It caches parsed templates and avoids the need to do parsing with every single request. You can configure it using the following snippet (the `loaders` setting is important here):

```
TEMPLATES = [
    {
        "BACKEND": "django.template.backends.django.DjangoTemplates",
        "OPTIONS": {
            "context_processors": [
                "django.contrib.auth.context_processors.auth",
                "django.template.context_processors.debug",
                "django.template.context_processors.i18n",
                "django.template.context_processors.request",
                "django.template.context_processors.csrf",
                "django.contrib.messages.context_processors.messages",
                "weblate.trans.context_processors.weblate_context",
            ],
        },
        "APP_DIRS": True,
    }
]
```

Siehe auch:

`django.template.loaders.cached.Loader`

Running maintenance tasks

For optimal performance, it is good idea to run some maintenance tasks in the background. This is now automatically done by *Background tasks using Celery* and covers following tasks:

- Configuration health check (hourly).
- Committing pending changes (hourly), see *Lazy commits* and `commit_pending`.
- Updating component alerts (daily).
- Update remote branches (nightly), see `AUTO_UPDATE`.
- Translation memory backup to JSON (daily), see `dump_memory`.
- Fulltext and database maintenance tasks (daily and weekly tasks), see `cleanuptrans`.

Geändert in Version 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*.

System locales and encoding

The system locales should be configured to UTF-8 capable ones. On most Linux distributions this is the default setting. In case it is not the case on your system, please change locales to UTF-8 variant.

For example by editing `/etc/default/locale` and setting there `LANG="C.UTF-8"`.

In einigen Fällen haben die einzelnen Dienste eine separate Konfiguration für Gebietsschemata. Dies ist je nach Distribution und Webserver unterschiedlich, daher sollten Sie die Dokumentation Ihrer Webserver-Pakete daraufhin überprüfen.

Apache unter Ubuntu verwendet `/etc/apache2/envvars`:

```
export LANG='en_US.UTF-8'
export LC_ALL='en_US.UTF-8'
```

Apache unter CentOS verwendet `/etc/sysconfig/httpd` (oder `/opt/rh/httpd24/root/etc/sysconfig/httpd`):

```
LANG='en_US.UTF-8'
```

Using custom certificate authority

Weblate does verify SSL certificates during HTTP requests. In case you are using custom certificate authority which is not trusted in default bundles, you will have to add its certificate as trusted.

The preferred approach is to do this at system level, please check your distro documentation for more details (for example on debian this can be done by placing the CA certificate into `/usr/local/share/ca-certificates/` and running **`update-ca-certificates`**).

Once this is done, system tools will trust the certificate and this includes Git.

For Python code, you will need to configure requests to use system CA bundle instead of the one shipped with it. This can be achieved by placing following snippet to `settings.py` (the path is Debian specific):

```
import os

os.environ["REQUESTS_CA_BUNDLE"] = "/etc/ssl/certs/ca-certificates.crt"
```

Compressing client assets

Weblate comes with a bunch of JavaScript and CSS files. For performance reasons it is good to compress them before sending to a client. In default configuration this is done on the fly at cost of little overhead. On big installations, it is recommended to enable offline compression mode. This needs to be done in the configuration and the compression has to be triggered on every Weblate upgrade.

The configuration switch is simple by enabling `django.conf.settings.COMPRESS_OFFLINE` and configuring `django.conf.settings.COMPRESS_OFFLINE_CONTEXT` (the latter is already included in the example configuration):

```
COMPRESS_OFFLINE = True
```

On each deploy you need to compress the files to match current version:

```
weblate compress
```

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

Siehe auch:

Common Deployment Scenarios, *Serving static files*

2.1.10 Running server

Hinweis: In case you are not experienced with services described below, you might want to try *Installing using Docker*.

You will need several services to run Weblate, the recommended setup consists of:

- Database server (see *Database setup for Weblate*)
- Cache server (see *Enable caching*)
- Frontend web server for static files and SSL termination (see *Serving static files*)
- WSGI server for dynamic content (see *Sample configuration for NGINX and uWSGI*)
- Celery for executing background tasks (see *Background tasks using Celery*)

Bemerkung: 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`.

Bemerkung: The WSGI process has to be executed under the same user the Celery process, otherwise files in the `DATA_DIR` will be stored with mixed ownership, leading to runtime issues.

See also *Filesystem permissions* and *Background tasks using Celery*.

Running web server

Running Weblate is not different from running any other Django based program. Django is usually executed as uWSGI or fcgi (see examples for different webservers below).

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

```
weblate runserver
```

Warnung: 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`.

Hinweis: 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

Geändert in Version 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/
Wird für Medien-Uploads durch Benutzer (z. B. Bildschirmfotos) verwendet.

/favicon.ico
Should be rewritten to rewrite a rule to serve `/static/favicon.ico`.

Siehe auch:

Sample configuration for NGINX and uWSGI, *Sample configuration for Apache*, *Sample configuration for Apache and Gunicorn*, *Compressing client assets*, How to deploy Django, How to deploy static files

Content security policy

The default Weblate configuration enables `weblate.middleware.SecurityMiddleware` middleware which sets security related HTTP headers like `Content-Security-Policy` or `X-XSS-Protection`. These are by default set up to work with Weblate and its configuration, but this might need customization for your environment.

Siehe auch:

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/python3.9/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`):

```
#
# nginx configuration for Weblate
#
# You will want to change:
#
# - server_name
# - change /home/weblate/weblate-env to location where Weblate virtualenv is placed
# - change /home/weblate/data to match your DATA_DIR
# - change python3.9 to match your Python version
# - change weblate user to match your Weblate user
#
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`):

```
#
# uWSGI configuration for Weblate
#
# You will want to change:
#
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
# - change /home/weblate/weblate-env to location where Weblate virtualenv is placed
# - change /home/weblate/data to match your DATA_DIR
# - change python3.9 to match your Python version
# - change weblate user to match your Weblate user
#
[uwsgi]
plugins       = python3
master        = true
protocol      = uwsgi
socket        = 127.0.0.1:8080
wsgi-file     = /home/weblate/weblate-env/lib/python3.9/site-packages/weblate/wsgi.
↪py

# Add path to Weblate checkout if you did not install
# Weblate by pip
# python-path   = /path/to/weblate

# In case you're using virtualenv uncomment this:
virtualenv = /home/weblate/weblate-env

# Needed for OAuth/OpenID
buffer-size   = 8192

# Reload when consuming too much of memory
reload-on-rss = 250

# Increase number of workers for heavily loaded sites
workers       = 8

# Enable threads for Sentry error submission
enable-threads = true

# Child processes do not need file descriptors
close-on-exec = true

# Avoid default 0000 umask
umask = 0022

# Run as weblate user
uid = weblate
gid = weblate

# Enable harakiri mode (kill requests after some time)
# harakiri = 3600
# harakiri-verbose = true

# Enable uWSGI stats server
# stats = :1717
# stats-http = true

# Do not log some errors caused by client disconnects
ignore-sigpipe = true
ignore-write-errors = true
disable-write-exception = true
```

Siehe auch:

How to use Django with uWSGI

Sample configuration for Apache

It is recommended to use prefork MPM when using WSGI with Weblate.

The following configuration runs Weblate as WSGI, you need to have enabled `mod_wsgi` (available as `weblate/examples/apache.conf`):

```
#
# VirtualHost for Weblate
#
# You will want to change:
#
# - ServerAdmin and ServerName
# - change /home/weblate/weblate-env to location where Weblate virtualenv is placed
# - change /home/weblate/data to match your DATA_DIR
# - change python3.9 to match your Python version
# - change weblate user to match your Weblate user
#
<VirtualHost *:80>
    ServerAdmin admin@weblate.example.org
    ServerName weblate.example.org

    # DATA_DIR/static/favicon.ico
    Alias /favicon.ico /home/weblate/data/static/favicon.ico

    # DATA_DIR/static/
    Alias /static/ /home/weblate/data/static/
    <Directory /home/weblate/data/static/>
        Require all granted
    </Directory>

    # DATA_DIR/media/
    Alias /media/ /home/weblate/data/media/
    <Directory /home/weblate/data/media/>
        Require all granted
    </Directory>

    # Path to your Weblate virtualenv
    WSGIDaemonProcess weblate python-home=/home/weblate/weblate-env user=weblate␣
↪request-timeout=600
    WSGIProcessGroup weblate
    WSGIApplicationGroup %{GLOBAL}

    WSGIScriptAlias / /home/weblate/weblate-env/lib/python3.9/site-packages/
↪weblate/wsgi.py process-group=weblate
    WSGIPassAuthorization On

    <Directory /home/weblate/weblate-env/lib/python3.9/site-packages/weblate/>
        <Files wsgi.py>
        Require all granted
        </Files>
    </Directory>

</VirtualHost>
```

Bemerkung: 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`.

Siehe auch:

System locales and encoding, How to use Django with Apache and mod_wsgi

Sample configuration for Apache and Gunicorn

The following configuration runs Weblate in Gunicorn and Apache 2.4 (available as `weblate/examples/apache.gunicorn.conf`):

```
#
# VirtualHost for Weblate using gunicorn on localhost:8000
#
# You will want to change:
#
# - ServerAdmin and ServerName
# - change /home/weblate/weblate-env to location where Weblate virtualenv is placed
# - change /home/weblate/data to match your DATA_DIR
# - change python3.9 to match your Python version
# - change weblate user to match your Weblate user
#
<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>
```

Siehe auch:

How to use Django with Gunicorn

Running Weblate under path

Neu in Version 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
#
# You will want to change:
#
# - ServerAdmin and ServerName
# - change /home/weblate/weblate-env to location where Weblate virtualenv is placed
# - change /home/weblate/data to match your DATA_DIR
# - change python3.9 to match your Python version
# - change weblate user to match your Weblate user
#
<VirtualHost *:80>
    ServerAdmin admin@weblate.example.org
    ServerName weblate.example.org

    # DATA_DIR/static/favicon.ico
    Alias /weblate/favicon.ico /home/weblate/data/static/favicon.ico

    # DATA_DIR/static/
    Alias /weblate/static/ /home/weblate/data/static/
    <Directory /home/weblate/data/static/>
        Require all granted
    </Directory>

    # DATA_DIR/media/
    Alias /weblate/media/ /home/weblate/data/media/
    <Directory /home/weblate/data/media/>
        Require all granted
    </Directory>

    # Path to your Weblate virtualenv
    WSGIDaemonProcess weblate python-home=/home/weblate/weblate-env user=weblate␣
↪request-timeout=600
    WSGIProcessGroup weblate
    WSGIApplicationGroup %{GLOBAL}

    WSGIScriptAlias /weblate /home/weblate/weblate-env/lib/python3.9/site-packages/
↪weblate/wsgi.py process-group=weblate
    WSGIPassAuthorization On

    <Directory /home/weblate/weblate-env/lib/python3.9/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

Neu in Version 3.2.

Weblate uses Celery to execute regular and background tasks. You are supposed to run a Celery service that will execute these. For example, it is responsible for handling following operations (this list is not complete):

- Receiving webhooks from external services (see *Benachrichtigungs-Hooks*).
- Running regular maintenance tasks such as backups, cleanups, daily add-ons, or updates (see *Sichern und Verschieben von Weblate*, `BACKGROUND_TASKS`, *Erweiterungen*).
- Ausführung von *Automatische Übersetzung*.
- Zusammenfassungs-Benachrichtigungen senden.
- Offloading expensive operations from the wsgi process.
- Übergabe ausstehender Änderungen (siehe *Lazy commits*).

A typical setup using Redis as a backend looks like this:

```
CELERY_TASK_ALWAYS_EAGER = False
CELERY_BROKER_URL = "redis://localhost:6379"
CELERY_RESULT_BACKEND = CELERY_BROKER_URL
```

Siehe auch:

Redis broker configuration in Celery

You should also start the Celery worker to process the tasks and start scheduled tasks, this can be done directly on the command-line (which is mostly useful when debugging or developing):

```
./weblate/examples/celery start
./weblate/examples/celery stop
```

Bemerkung: The Celery process has to be executed under the same user as the WSGI process, otherwise files in the `DATA_DIR` will be stored with mixed ownership, leading to runtime issues.

See also *Filesystem permissions* and *Running server*.

Executing Celery tasks in the wsgi using eager mode

Bemerkung: This will have severe performance impact on the web interface, and will break features depending on regular trigger (for example committing pending changes, digest notifications, or backups).

For development, you might want to use eager configuration, which does process all tasks in place:

```
CELERY_TASK_ALWAYS_EAGER = True
CELERY_BROKER_URL = "memory://"
CELERY_TASK_EAGER_PROPAGATES = True
```

Running Celery as system service

Most likely you will want to run Celery as a daemon and that is covered by Daemonization. For the most common Linux setup using systemd, you can use the example files shipped in the `examples` folder listed below.

Systemd unit to be placed as `/etc/systemd/system/celery-weblate.service`:

```
[Unit]
Description=Celery Service (Weblate)
After=network.target

[Service]
Type=forking
User=weblate
Group=weblate
EnvironmentFile=/etc/default/celery-weblate
WorkingDirectory=/home/weblate
RuntimeDirectory=celery
RuntimeDirectoryPreserve=restart
LogsDirectory=celery
ExecStart=/bin/sh -c '${CELERY_BIN} multi start ${CELERYD_NODES} \
  -A ${CELERY_APP} --pidfile=${CELERYD_PID_FILE} \
  --logfile=${CELERYD_LOG_FILE} --loglevel=${CELERYD_LOG_LEVEL} ${CELERYD_OPTS}'
ExecStop=/bin/sh -c '${CELERY_BIN} multi stopwait ${CELERYD_NODES} \
  --pidfile=${CELERYD_PID_FILE}'
ExecReload=/bin/sh -c '${CELERY_BIN} multi restart ${CELERYD_NODES} \
  -A ${CELERY_APP} --pidfile=${CELERYD_PID_FILE} \
  --logfile=${CELERYD_LOG_FILE} --loglevel=${CELERYD_LOG_LEVEL} ${CELERYD_OPTS}'

[Install]
WantedBy=multi-user.target
```

Environment configuration to be placed as `/etc/default/celery-weblate`:

```
# Name of nodes to start
CELERYD_NODES="celery notify memory backup translate"

# Absolute or relative path to the 'celery' command:
CELERY_BIN="/home/weblate/weblate-env/bin/celery"

# App instance to use
# comment out this line if you don't use an app
CELERY_APP="weblate.utils"

# Extra command-line arguments to the worker,
# increase concurrency if you get weblate.E019
CELERYD_OPTS="--beat:celery --queues:celery=celery --prefetch-multiplier:celery=4 \
    --queues:notify=notify --prefetch-multiplier:notify=10 \
    --queues:memory=memory --prefetch-multiplier:memory=10 \
    --queues:translate=translate --prefetch-multiplier:translate=4 \
    --concurrency:backup=1 --queues:backup=backup  --prefetch-multiplier:backup=2"

# Logging configuration
# - %n will be replaced with the first part of the nodename.
# - %I will be replaced with the current child process index
#   and is important when using the prefork pool to avoid race conditions.
CELERYD_PID_FILE="/run/celery/weblate-%n.pid"
CELERYD_LOG_FILE="/var/log/celery/weblate-%n%I.log"
CELERYD_LOG_LEVEL="INFO"
```

Additional configuration to rotate Celery logs using **`logrotate`** to be placed as `/etc/logrotate.d/celery`:

```
/var/log/celery/*.log {
        weekly
        missingok
        rotate 12
        compress
        notifempty
}
```

Periodic tasks using Celery beat

Weblate comes with built-in setup for scheduled tasks. You can however define additional tasks in `settings.py`, for example see *Lazy commits*.

The tasks are supposed to be executed by Celery beats daemon. In case it is not working properly, it might not be running or its database was corrupted. Check the Celery startup logs in such case to figure out root cause.

Monitoring Celery status

You can find current length of the Celery task queues in the *Verwaltungsoberfläche* or you can use `celery_queues` on the command-line. In case the queue will get too long, you will also get configuration error in the admin interface.

Warnung: 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*.

Siehe auch:

Monitoring Weblate, *How can I check whether my Weblate is set up properly?*, Configuration and defaults, Workers Guide, Daemonization, Monitoring and Management Guide, `celery_queues`

2.1.12 Monitoring Weblate

Weblate provides the `/healthz/` URL to be used in simple health checks, for example using Kubernetes. The Docker container has built-in health check using this URL.

Zur Überwachung der Metriken von Weblate können Sie den `GET /api/metrics/` API-Endpunkt verwenden.

Siehe auch:

How can I check whether my Weblate is set up properly?, *Monitoring Celery status*, Weblate plugin for Munin

2.1.13 Collecting error reports

Weblate, as any other software, can fail. In order to collect useful failure states we recommend to use third party services to collect such information. This is especially useful in case of failing Celery tasks, which would otherwise only report error to the logs and you won't get notified on them. Weblate has support for the following services:

Sentry

Weblate has built-in support for Sentry. To use it, it's enough to set *SENTRY_DSN* in the `settings.py`:

```
SENTRY_DSN = "https://id@your.sentry.example.com/"
```

Rollbar

Weblate has built-in support for Rollbar. To use it, it's enough to follow instructions for Rollbar notifier for Python.

In short, you need to adjust `settings.py`:

```
# Add rollbar as last middleware:
MIDDLEWARE = [
    # … other middleware classes …
    "rollbar.contrib.django.middleware.RollbarNotifierMiddleware",
]

# Configure client access
ROLLBAR = {
    "access_token": "POST_SERVER_ITEM_ACCESS_TOKEN",
    "client_token": "POST_CLIENT_ITEM_ACCESS_TOKEN",
    "environment": "development" if DEBUG else "production",
    "branch": "main",
    "root": "/absolute/path/to/code/root",
}
```

Everything else is integrated automatically, you will now collect both server and client side errors.

2.1.14 Migrating Weblate to another server

Migrating Weblate to another server should be pretty easy, however it stores data in few locations which you should migrate carefully. The best approach is to stop Weblate for the migration.

Migrating database

Je nach Ihrem Datenbank-Backend haben Sie mehrere Möglichkeiten, die Datenbank zu migrieren. Die einfachste ist, die Datenbank auf einem Server zu löschen und sie auf dem neuen Server zu importieren. Alternativ können Sie auch die Replikation verwenden, sofern Ihre Datenbank dies unterstützt.

The best approach is to use database native tools, as they are usually the most effective (e.g. **mysqldump** or **pg_dump**). If you want to migrate between different databases, the only option might be to use Django management to dump and import the database:

```
# Export current data
weblate dumpdata > /tmp/weblate.dump
# Import dump
weblate loaddata /tmp/weblate.dump
```

Migrating VCS repositories

The VCS repositories stored under `DATA_DIR` need to be migrated as well. You can simply copy them or use **`rsync`** to do the migration more effectively.

Other notes

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

2.2 Weblate deployments

Weblate can be easily installed in your cloud. Please find detailed guide for your platform:

- *Installing using Docker*
- *Installing on OpenShift*
- *Installing on Kubernetes*

2.2.1 Third-party deployments for Weblate

Bemerkung: 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-Paket

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.

Cloudron install

Weblate in YunoHost

The self-hosting project YunoHost provides a package for Weblate. Once you have your YunoHost installation, you may install Weblate as any other application. It will provide you with a fully working stack with backup and restoration, but you may still have to edit your settings file for specific usages.

You may use your administration interface, or this button (it will bring you to your server):

Install with YUNOHOST

It also is possible to use the command-line interface:

```
yunohost app install https://github.com/YunoHost-Apps/weblate_ynh
```

2.3 Upgrading Weblate

2.3.1 Docker image upgrades

The official Docker image (see *Installing using Docker*) has all Weblate upgrade steps integrated. There are typically no manual steps needed besides pulling latest version.

Siehe auch:

Aktualisieren des Docker-Containers

2.3.2 Generic upgrade instructions

Before upgrading, please check the current *Software-Anforderungen* 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.

Bemerkung: Es wird empfohlen, vor dem Upgrade eine vollständige Datenbanksicherung durchzuführen, damit Sie die Datenbank zurücksetzen können, falls das Upgrade fehlschlägt, siehe *Sichern und Verschieben von Weblate*.

1. Stop wsgi and Celery processes. The upgrade can perform incompatible changes in the database, so it is always safer to avoid old processes running while upgrading.
2. Upgrade Weblate code.

   For pip installs it can be achieved by:

   ```
   pip install -U "Weblate[all]==version"
   ```

   Or, if you just want to get the latest released version:

   ```
   pip install -U "Weblate[all]"
   ```

   If you don't want to install all of the optional dependencies do:

   ```
   pip install -U Weblate
   ```

   With Git checkout you need to fetch new source code and update your installation:

   ```
   cd weblate-src
   git pull
   # Update Weblate inside your virtualenv
   . ~/weblate-env/bin/pip install -e .
   # Install dependencies directly when not using virtualenv
   pip install --upgrade -r requirements.txt
   # Install optional dependencies directly when not using virtualenv
   pip install --upgrade -r requirements-optional.txt
   ```

3. New Weblate release might have new *Optional dependencies*, please check if they cover features you want.

4. Upgrade configuration file, refer to `settings_example.py` or *Version specific instructions* for needed steps.
5. Upgrade database structure:

```
weblate migrate --noinput
```

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

```
weblate collectstatic --noinput --clear
```

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

```
weblate compress
```

8. If you are running version from Git, you should also regenerate locale files every time you are upgrading. You can do this by invoking:

```
weblate compilemessages
```

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

```
weblate check --deploy
```

10. Celery-Worker neu starten (siehe *Background tasks using Celery*).

2.3.3 Version specific instructions

Upgrade von 2.x

Wenn Sie von der Version 2.x upgraden, sollten Sie immer zuerst auf 3.0.1 upgraden und dann mit der 3.x-Serie fortfahren. Upgrades, die diesen Schritt überspringen, werden nicht unterstützt und führen zu Fehlern.

Siehe auch:

Upgrade von 2.20 auf 3.0 in der Weblate-3.0-Dokumentation

Upgrade von 3.x

Wenn Sie von der Version 3.x aktualisieren, führen Sie immer zuerst ein Upgrade auf 4.0.4 oder 4.1.1 durch und setzen Sie dann das Upgrade in der 4.x-Serie fort. Upgrades, die diesen Schritt überspringen, werden nicht unterstützt und führen zu Fehlern.

Siehe auch:

Upgrade von 3.11 auf 4.0 in der Weblate-4.0-Dokumentation <https://docs.weblate.org/en/weblate-4.0.4/admin/upgrade.html#upgrade-from-3-11-to-4-0>`_

Upgrade von 4.0 auf 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.

- Es gibt einige neue und aktualisierte Anforderungen.
- 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.

Siehe auch:

Generic upgrade instructions

Upgrade von 4.1 auf 4.2

Please follow *Generic upgrade instructions* in order to perform update.

Notable configuration or dependencies changes:

- Upgrades von 3.x Versionen werden nicht mehr unterstützt, bitte aktualisieren Sie zuerst auf 4.0 oder 4.1.
- Es gibt einige neue und aktualisierte Anforderungen.
- 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.

Siehe auch:

Generic upgrade instructions

Upgrade von 4.2 auf 4.3

Please follow *Generic upgrade instructions* in order to perform update.

Notable configuration or dependencies changes:

- There are some changes in quality checks, you might want to include them in case you modified the *`CHECK_LIST`*.
- The source language attribute was moved from project to a component what is exposed in the API. You will need to update *Weblate Client* in case you are using it.
- The database migration to 4.3 might take long depending on number of strings you are translating (expect around one hour of migration time per 100,000 source strings).
- There is a change in `INSTALLED_APPS`.
- There is a new setting *`SESSION_COOKIE_AGE_AUTHENTICATED`* which complements `SESSION_COOKIE_AGE`.
- In case you were using **`hub`** or **`lab`** to integrate with GitHub or GitLab, you will need to reconfigure this, see *`GITHUB_CREDENTIALS`* and *`GITLAB_CREDENTIALS`*.

Geändert in Version 4.3.1:

- The Celery configuration was changed to add `memory` queue. Please adjust your startup scripts and `CELERY_TASK_ROUTES` setting.

Geändert in Version 4.3.2:

- The `post_update` method of add-ons now takes extra `skip_push` parameter.

Siehe auch:

Generic upgrade instructions

Upgrade von 4.3 auf 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 ist jetzt erforderlich.
- In case you are using MySQL or MariaDB, the minimal required versions have increased, see *MySQL and MariaDB*.

Geändert in Version 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.

Siehe auch:

Generic upgrade instructions

Upgrade von 4.4 auf 4.5

Please follow *Generic upgrade instructions* in order to perform update.

Notable configuration or dependencies changes:

- Bei umfangreichen Glossaren kann die Migration viel Zeit in Anspruch nehmen.
- Glossare werden jetzt als reguläre Komponenten gespeichert.
- Die Glossar-API wurde entfernt. Verwenden Sie die reguläre Übersetzungs-API für den Zugriff auf Glossare.
- There is a change in `INSTALLED_APPS` - `weblate.metrics` should be added.

Geändert in Version 4.5.1:

- Es gibt eine neue Abhängigkeit vom Modul *pyahocorasick*.

Siehe auch:

Generic upgrade instructions

Upgrade von 4.5 auf 4.6

Please follow *Generic upgrade instructions* in order to perform update.

Notable configuration or dependencies changes:

- There are new file formats, you might want to include them in case you modified the `WEBLATE_FORMATS`.
- API for creating components now automatically uses *Weblate internal URLs*, see `POST /api/projects/(string:project)/components/`.
- Es gibt eine Änderung in den Abhängigkeiten und `PASSWORD_HASHERS` um Argon2 für das Hashing von Passwörtern zu bevorzugen.

Siehe auch:

Generic upgrade instructions

Upgrade von 4.6 auf 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 ist jetzt erforderlich.

Siehe auch:

Generic upgrade instructions

Upgrade von 4.7 auf 4.8

Please follow *Generic upgrade instructions* in order to perform update.

In dieser Version sind keine zusätzlichen Upgrade-Schritte erforderlich.

Siehe auch:

Generic upgrade instructions

Upgrade von 4.8 auf 4.9

Please follow *Generic upgrade instructions* in order to perform update.

- There is a change in storing metrics, the upgrade can take long time on larger sites.

Siehe auch:

Generic upgrade instructions

Upgrade von 4.0 auf 4.10

Please follow *Generic upgrade instructions* in order to perform update.

- There is a change in per-project groups, the upgrade can take long time on sites with thousands of projects.
- Django 4.0 has made some incompatible changes, see Backwards incompatible changes in 4.0. Weblate still supports Django 3.2 for now, in case any of these are problematic. Most notable changes which might affect Weblate:
  - Unterstützung für PostgreSQL 9.6 wurde eingestellt, Django 4.0 unterstützt PostgreSQL 10 und höher.
  - Format of `CSRF_TRUSTED_ORIGINS` was changed.
- Der Docker-Container verwendet jetzt Django 4.0, siehe oben für Änderungen.

Siehe auch:

Generic upgrade instructions

Upgrade von 4.10 auf 4.11

Please follow *Generic upgrade instructions* in order to perform update.

- Weblate erfordert jetzt Python 3.7 oder neuer.
- The implementation of *Verwaltung der Zugriffssteuerung nach Projekt* has changed, removing the project prefix from the group names. This affects API users.
- Weblate verwendet jetzt das Modul `charset-normalizer` anstelle von `chardet` für die Zeichensatzerkennung.
- **Changed in 4.11.1:** There is a change in `REST_FRAMEWORK` setting (removal of one of the backends in `DEFAULT_AUTHENTICATION_CLASSES`).

Siehe auch:

Generic upgrade instructions

Upgrade from 4.11 to 4.12

Please follow *Generic upgrade instructions* in order to perform update.

- There are no special steps required.

Siehe auch:

Generic upgrade instructions

Upgrade from 4.12 to 4.13

Please follow *Generic upgrade instructions* in order to perform update.

- The *Language definitions* are now automatically updated on upgrade, use `UPDATE_LANGUAGES` to disable that.
- Handling of context and location has been changed for *Windows RC files*, *HTML files*, *IDML Format*, and *Textdateien* file formats. In most cases the context is now shown as location.
- The machine translation services are now configured using the user interface, settings from the configuration file will be imported during the database migration.

Siehe auch:

Generic upgrade instructions

Upgrade from 4.13 to 4.14

Please follow *Generic upgrade instructions* in order to perform update.

- The Java formatting checks now match GNU gettext flags. The flags set in Weblate will be automatically migrated, but third-party scripts will need to use `java-printf-format` instead of `java-format` and `java-format` instead of `java-messageformat`.
- The *jellyfish* dependency has been replaced by *rapidfuzz*.

Siehe auch:

Generic upgrade instructions

2.3.4 Upgrade von Python 2 auf Python 3

Weblate no longer supports Python older than 3.6. In case you are still running on older version, please perform migration to Python 3 first on existing version and upgrade later. See Upgrading from Python 2 to Python 3 in the Weblate 3.11.1 documentation.

2.3.5 Migration von anderen Datenbanken zu PostgreSQL

Wenn Sie Weblate auf einer anderen Datenbank als PostgreSQL betreiben, sollten Sie eine Migration zu PostgreSQL in Betracht ziehen, da Weblate damit am besten funktioniert. Die folgenden Schritte werden Sie bei der Migration Ihrer Daten zwischen den Datenbanken anleiten. Bitte denken Sie daran, sowohl den Web- als auch den Celery-Server vor der Migration zu stoppen, da es sonst zu inkonsistenten Daten kommen kann.

Creating a database in PostgreSQL

It is usually a good idea to run Weblate in a separate database, and separate user account:

```
# If PostgreSQL was not installed before, set the main password
sudo -u postgres psql postgres -c "\password postgres"

# Create a database user called "weblate"
sudo -u postgres createuser -D -P weblate

# Create the database "weblate" owned by "weblate"
sudo -u postgres createdb -E UTF8 -O weblate weblate
```

Migration mit Django JSON-Dumps

The simplest approach for migration is to utilize Django JSON dumps. This works well for smaller installations. On bigger sites you might want to use pgloader instead, see *Migrating to PostgreSQL using pgloader*.

1. Add PostgreSQL as additional database connection to the `settings.py`:

```
DATABASES = {
    "default": {
        # Database engine
        "ENGINE": "django.db.backends.mysql",
        # Database name
        "NAME": "weblate",
        # Database user
        "USER": "weblate",
        # Database password
        "PASSWORD": "password",
        # Set to empty string for localhost
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
        "HOST": "database.example.com",
        # Set to empty string for default
        "PORT": "",
        # Additional database options
        "OPTIONS": {
            # In case of using an older MySQL server, which has MyISAM as a␣
↪default storage
            # 'init_command': 'SET storage_engine=INNODB',
            # Uncomment for MySQL older than 5.7:
            # 'init_command': "SET sql_mode='STRICT_TRANS_TABLES'",
            # If your server supports it, see the Unicode issues above
            "charset": "utf8mb4",
            # Change connection timeout in case you get MySQL gone away error:
            "connect_timeout": 28800,
        },
    },
    "postgresql": {
        # Database engine
        "ENGINE": "django.db.backends.postgresql",
        # Database name
        "NAME": "weblate",
        # Database user
        "USER": "weblate",
        # Database password
        "PASSWORD": "password",
        # Set to empty string for localhost
        "HOST": "database.example.com",
        # Set to empty string for default
        "PORT": "",
    },
}
```

2. Run migrations and drop any data inserted into the tables:

```
weblate migrate --database=postgresql
weblate sqlflush --database=postgresql | weblate dbshell --database=postgresql
```

3. Dump legacy database and import to PostgreSQL

```
weblate dumpdata --all --output weblate.json
weblate loaddata weblate.json --database=postgresql
```

4. Adjust `DATABASES` to use just PostgreSQL database as default, remove legacy connection.

Weblate should be now ready to run from the PostgreSQL database.

Migrating to PostgreSQL using pgloader

The pgloader is a generic migration tool to migrate data to PostgreSQL. You can use it to migrate Weblate database.

1. Adjust your `settings.py` to use PostgreSQL as a database.
2. Migrate the schema in the PostgreSQL database:

```
weblate migrate
weblate sqlflush | weblate dbshell
```

3. Run the pgloader to transfer the data. The following script can be used to migrate the database, but you might want to learn more about pgloader to understand what it does and tweak it to match your setup:

```
LOAD DATABASE
     FROM      mysql://weblate:password@localhost/weblate
     INTO postgresql://weblate:password@localhost/weblate

WITH include no drop, truncate, create no tables, create no indexes, no␣
↪foreign keys, disable triggers, reset sequences, data only

ALTER SCHEMA 'weblate' RENAME TO 'public'
;
```

2.3.6 Migrating from Pootle

As Weblate was originally written as replacement from Pootle, it is supported to migrate user accounts from Pootle. You can dump the users from Pootle and import them using `importusers`.

2.4 Sichern und Verschieben von Weblate

2.4.1 Project level backups

Neu in Version 4.14.

Warnung: Restoring backups is only supported when using PostgreSQL or MariaDB 10.5+ as a database.

The project backups all translation content from Weblate (project, components, translations, string comments, suggestions or checks). It is suitable for transferring a project to another Weblate instance.

You can perform a project backup in *Manage* ↓ *Backups*. The backup can be restored when creating a project (see *Adding translation projects and components*).

The backups currently do not include access control information and history.

The comments and suggestions are backed up with an username of user who did create them. Upon import it is assigned to a matching user. If there is no user with such username, it is assigned to anonymous user.

The generated backups are kept on the server as configured by `PROJECT_BACKUP_KEEP_DAYS` and `PROJECT_BACKUP_KEEP_COUNT` (it defaults to keep at most 3 backups for 30 days).

2.4.2 Automatisierte Datensicherung mit BorgBackup

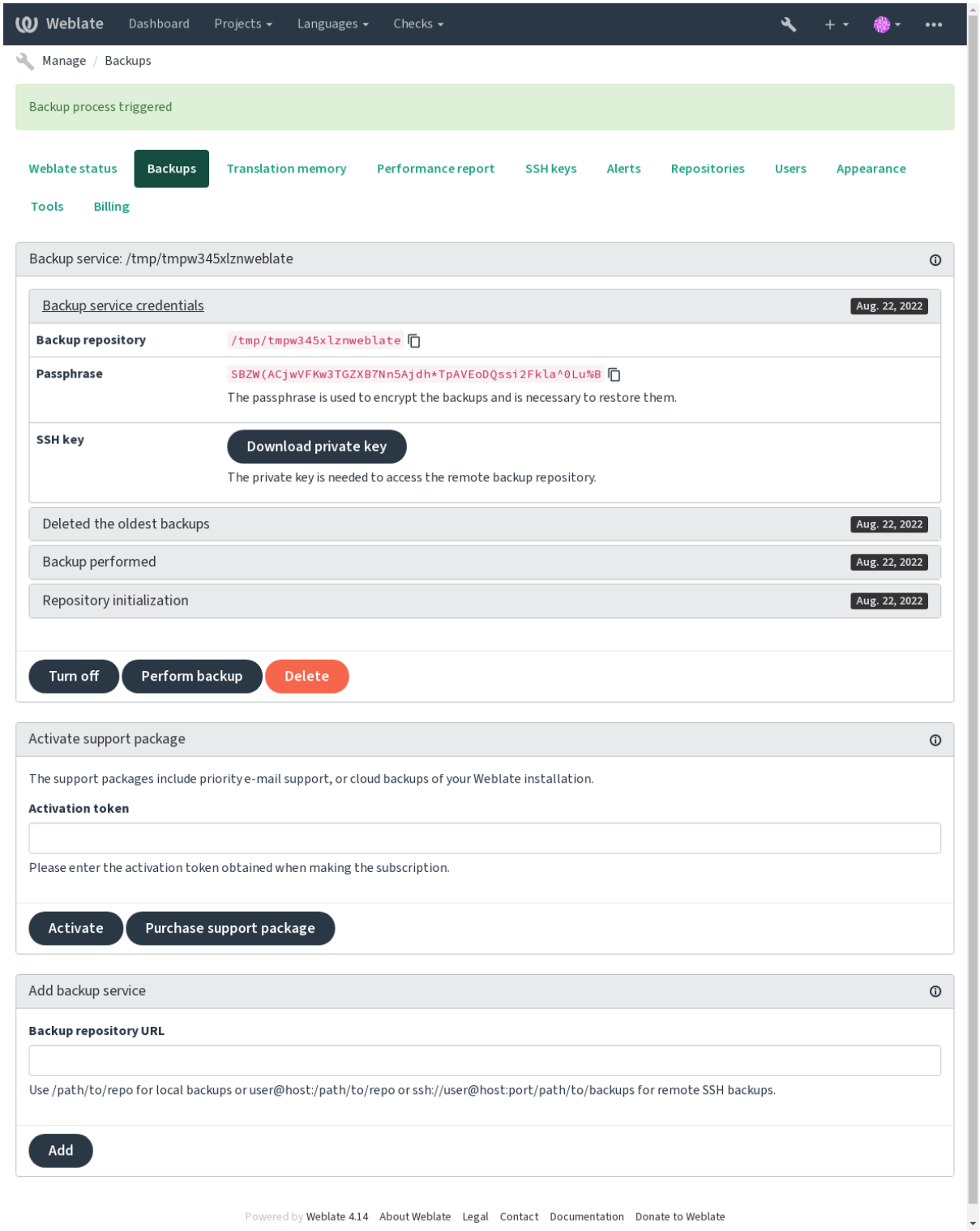
Neu in Version 3.9.

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.

Geändert in Version 4.4.1: Sowohl PostgreSQL- als auch MySQL/MariaDB-Datenbanken sind in den automatischen Backups enthalten.

Die Backups mit Borg sind inkrementell, und Weblate ist so konfiguriert, dass die folgenden Backups beibehalten werden:

- Tägliche Backups der letzten 14 Tage
- Wöchentliche Backups der letzten 8 Wochen
- Monatliche Backups der letzten 6 Monate


Weblate
Dashboard
Projects
Languages
Checks
Manage / Backups
Backup process triggered
Weblate status
Backups
Translation memory
Performance report
SSH keys
Alerts
Repositories
Users
Appearance
Tools
Billing
Backup service: /tmp/tmpw345xlznweblate
Backup service credentials
Aug. 22, 2022
Backup repository
/tmp/tmpw345xlznweblate
Passphrase
SBZW(ACjwVFKw3TGZXB7Nn5Ajdh*TpAVEoDQssi2Fkla^0Lu%B
The passphrase is used to encrypt the backups and is necessary to restore them.
SSH key
Download private key
The private key is needed to access the remote backup repository.
Deleted the oldest backups
Aug. 22, 2022
Backup performed
Aug. 22, 2022
Repository initialization
Aug. 22, 2022
Turn off
Perform backup
Delete
Activate support package
The support packages include priority e-mail support, or cloud backups of your Weblate installation.
Activation token
Please enter the activation token obtained when making the subscription.
Activate
Purchase support package
Add backup service
Backup repository URL
Use /path/to/repo for local backups or user@host:/path/to/repo or ssh://user@host:port/path/to/backups for remote SSH backups.
Add
Powered by Weblate 4.14
About Weblate
Legal
Contact
Documentation
Donate to Weblate

Borg encryption key

BorgBackup creates encrypted backups and you wouldn't be able to restore them without the passphrase. The passphrase is generated when adding a new backup service and you should copy it and keep it in a secure place.

If you are using *Von Weblate bereitgestellter Backup-Speicher*, please backup your private SSH key too, as it's used to access your backups.

Siehe auch:

borg init

Backup anpassen

- Das Datenbank-Backup kann über `DATABASE_BACKUP` konfiguriert werden.
- The backup creation can be customized using `BORG_EXTRA_ARGS`.

2.4.3 Von Weblate bereitgestellter Backup-Speicher

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. Erwerben Sie den *Backup-Dienst* auf https://weblate.org/support/#backup.
2. Geben Sie den erhaltenen Schlüssel in die Verwaltungsoberfläche ein, siehe *Integrating support*.
3. Weblate stellt eine Verbindung zum Cloud-Dienst her und erhält die Zugangsdaten für die Backups.
4. Aktivieren Sie die neue Backup-Konfiguration auf der Reiterkarte *Backups*.
5. Sichern Sie Ihre Borg-Anmeldedaten, um die Backups wiederherstellen zu können, siehe *Borg encryption key*.

Hinweis: Der manuelle Schritt, alles einzuschalten, dient Ihrer Sicherheit. Ohne Ihre Zustimmung werden keine Daten an den Backup-Speicher gesendet, den Sie durch den Registrierungsprozess erhalten haben.

2.4.4 Verwendung von eigenem Backup-Speicher

Sie können auch Ihren eigenen Speicher für die Backups verwenden. SSH kann verwendet werden, um Backups im entfernten Ziel zu speichern, der Zielserver muss BorgBackup installiert haben.

Siehe auch:

General in der Borg-Dokumentation

Lokales Dateisystem

Es wird empfohlen, den absoluten Pfad für das lokale Backup anzugeben, zum Beispiel */path/to/backup*. Das Verzeichnis muss für den Benutzer, unter dem Weblate läuft, beschreibbar sein (siehe *Filesystem permissions*). Existiert es nicht, versucht Weblate, es zu erstellen, benötigt dafür aber die entsprechenden Berechtigungen.

Hinweis: Wenn Sie Weblate in Docker ausführen, stellen Sie bitte sicher, dass der Speicherort des Backups vom Weblate-Container als Volume freigegeben wird. Andernfalls werden die Backups beim Neustart des Containers, in dem sie sich befinden, von Docker verworfen.

Eine Möglichkeit ist, Backups in ein bestehendes Volume zu legen, zum Beispiel `/app/data/borgbackup`. Dies ist ein vorhandenes Volume im Container.

Sie können auch einen neuen Container für die Backups in der Docker-Compose-Datei hinzufügen, indem Sie beispielsweise `/borgbackup` verwenden:

```
services:
  weblate:
    volumes:
      - /home/weblate/data:/app/data
      - /home/weblate/borgbackup:/borgbackup
```

Das Verzeichnis, in dem die Backups gespeichert werden, muss der UID 1000 gehören, ansonsten kann Weblate die Backups nicht dorthin schreiben.

Remote-Backups

Um Remote-Backups zu erstellen, müssen Sie BorgBackup auf einem anderen Server installieren, der für Ihre Weblate-Installation über SSH mit dem Weblate-SSH-Schlüssel erreichbar ist:

1. Bereiten Sie einen Server vor, auf dem Ihre Backups gespeichert werden sollen.
2. Installieren Sie den SSH-Server darauf (bei den meisten Linux-Distributionen erhalten Sie ihn standardmäßig).
3. Installieren Sie *BorgBackup* auf diesem Server; für die meisten Linux-Distributionen sind Pakete verfügbar (siehe Installation).
4. Wählen Sie einen vorhandenen Benutzer oder erstellen Sie einen neuen Benutzer, der für die Sicherung verwendet werden soll.
5. Fügen Sie dem Benutzer den SSH-Schlüssel von Weblate hinzu, damit Weblate ohne Passwort per SSH auf den Server zugreifen kann (siehe *Weblate-SSH-Schlüssel*).
6. Konfigurieren Sie den Backup-Speicherort in Weblate als `user@host:/path/to/backups` oder `ssh://user@host:port/path/to/backups`.

Hinweis: *Von Weblate bereitgestellter Backup-Speicher* bietet Ihnen automatisierte Remote-Backups ohne jeglichen Aufwand.

Siehe auch:

Weblate-SSH-Schlüssel, General

2.4.5 Restoring from BorgBackup

1. Restore access to your backup repository and prepare your backup passphrase.
2. List all the backups on the server using `borg list REPOSITORY`.
3. Restore the desired backup to the current directory using `borg extract REPOSITORY::ARCHIVE`.
4. Restore the database from the SQL dump placed in the `backup` directory in the Weblate data dir (see *Dumped data for backups*).
5. Copy the Weblate configuration (`backups/settings.py`, see *Dumped data for backups*) to the correct location, see *Adjusting configuration*.

   Bei der Verwendung von Docker-Containern ist die Einstellungsdatei bereits im Container enthalten und Sie sollten die ursprünglichen Umgebungsvariablen wiederherstellen. Die Datei `environment.yml` kann Ihnen dabei helfen (siehe *Dumped data for backups*).
6. Copy the whole restored data dir to the location configured by `DATA_DIR`.

   Bei der Verwendung von Docker-Containern legen Sie die Daten in das Datenvolumen, siehe *Docker-Container-Volumes*.

Bitte vergewissern Sie sich, dass die Dateien die korrekten Besitzverhältnisse und Berechtigungen haben, siehe *Filesystem permissions*.

The Borg session might look like this:

```
$ borg list /tmp/xxx
Enter passphrase for key /tmp/xxx:
2019-09-26T14:56:08                  Thu, 2019-09-26 14:56:08␣
↪[de0e0f13643635d5090e9896bdaceb92a023050749ad3f3350e788f1a65576a5]
$ borg extract /tmp/xxx::2019-09-26T14:56:08
Enter passphrase for key /tmp/xxx:
```

Siehe auch:

borg list, borg extract

2.4.6 Manual backup

Depending on what you want to save, back up the type of data Weblate stores in each respective place.

Hinweis: If you are doing the manual backups, you might want to silence Weblate's warning about a lack of backups by adding `weblate.I028` to `SILENCED_SYSTEM_CHECKS` in `settings.py` or *`WEBLATE_SILENCED_SYSTEM_CHECKS`* for Docker.

```
SILENCED_SYSTEM_CHECKS.append("weblate.I028")
```

Database

The actual storage location depends on your database setup.

Hinweis: 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()
```

Dateien

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

Geändert in Version 4.7: Der Umgebungsdump wurde als `environment.yml` hinzugefügt, um die Wiederherstellung in den Docker-Umgebungen zu erleichtern.

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`.
- Umgebungsdump als `environment.yml`.

The database backups are saved as plain text by default, but they can also be compressed or entirely skipped using *DATABASE_BACKUP*.

To restore the database backup load it using database tools, for example:

```
psql --file=database.sql weblate
```

Version control repositories

Stored in *DATA_DIR* `/vcs`.

Die Versionsverwaltung enthält eine Kopie Ihrer Upstream-Repositorys mit Weblate-Änderungen. Wenn Sie *Bei Commit gleichzeitig Pushen* für alle Ihre Übersetzungskomponenten aktiviert haben, werden alle Weblate-Änderungen Upstream aufgenommen. Es ist nicht notwendig, die Repositorys auf der Weblate-Seite zu sichern, da sie ohne Datenverlust von den Upstream-Speicherorten erneut geklont werden können.

SSH and GPG keys

Stored in *DATA_DIR* `/ssh` and *DATA_DIR* `/home`.

If you are using SSH or GPG keys generated by Weblate, you should back up these locations. Otherwise you will lose the private keys and you will have to regenerate new ones.

User uploaded files

Stored in `DATA_DIR` /media.

Sie sollten alle vom Benutzer hochgeladenen Dateien sichern (z. B. Bildschirmfotos).

Celery tasks

The Celery task queue might contain some info, but is usually not needed for a backup. At most you will lose updates not yet been processed to translation memory. It is recommended to perform the fulltext or repository update upon restoration anyhow, so there is no problem in losing these.

Siehe auch:

Background tasks using Celery

Command-line for manual backup

Using a cron job, you can set up a Bash command to be executed on a daily basis, for example:

```
$ XZ_OPT="-9" tar -Jcf ~/backup/weblate-backup-$(date -u +%Y-%m-%d_%H%M%S).xz␣
↪backups vcs ssh home media fonts secret
```

The string between the quotes after *XZ_OPT* allows you to choose your xz options, for instance the amount of memory used for compression; see https://linux.die.net/man/1/xz

You can adjust the list of folders and files to your needs. To avoid saving the translation memory (in backups folder), you can use:

```
$ XZ_OPT="-9" tar -Jcf ~/backup/weblate-backup-$(date -u +%Y-%m-%d_%H%M%S).xz␣
↪backups/database.sql backups/settings.py vcs ssh home media fonts secret
```

2.4.7 Restoring manual backup

1. Restore all data you have backed up.
2. Update all repositories using `updategit`.

   ```
   weblate updategit --all
   ```

2.4.8 Moving a Weblate installation

Relocate your installation to a different system by following the backing up and restoration instructions above.

Siehe auch:

Upgrade von Python 2 auf Python 3, *Migration von anderen Datenbanken zu PostgreSQL*

2.5 Legitimierung

2.5.1 Benutzerregistrierung

Die Standardeinstellung für Weblate ist die Verwendung von python-social-auth, einem Formular auf der Website zur Registrierung neuer Benutzer. Nach der Bestätigung ihrer E-Mail kann ein neuer Benutzer einen Beitrag leisten oder sich mit einem der Dienste von Drittanbietern authentifizieren.

Sie können die Registrierung neuer Benutzer auch mit `REGISTRATION_OPEN` abschalten.

Die Authentifizierungsversuche unterliegen dem *Rate limiting*.

2.5.2 Authentifizierungs-Backends

Die eingebaute Lösung von Django wird für die Authentifizierung verwendet, einschließlich verschiedener sozialer Optionen, um dies zu tun. Wenn Sie sie verwenden, können Sie die Benutzerdatenbank anderer Django-basierter Projekte importieren (siehe *Migrating from Pootle*).

Django can additionally be set up to authenticate against other means too.

Siehe auch:

Authentifizierungseinstellungen describes how to configure authentication in the official Docker image.

2.5.3 Social authentication

Thanks to Welcome to Python Social Auth's documentation!, Weblate support authentication using many third party services such as GitLab, Ubuntu, Fedora, etc.

Please check their documentation for generic configuration instructions in Django Framework.

Bemerkung: By default, Weblate relies on third-party authentication services to provide a validated e-mail address. If some of the services you want to use don't support this, please enforce e-mail validation on the Weblate side by configuring FORCE_EMAIL_VALIDATION for them. For example:

```
SOCIAL_AUTH_OPENSUSE_FORCE_EMAIL_VALIDATION = True
```

Siehe auch:

Pipeline

Enabling individual backends is quite easy, it's just a matter of adding an entry to the `AUTHENTICATION_BACKENDS` setting and possibly adding keys needed for a given authentication method. Please note that some backends do not provide user e-mail by default, you have to request it explicitly, otherwise Weblate will not be able to properly credit contributions users make.

Hinweis: Die meisten der Authentifizierungs-Backends erfordern HTTPS. Sobald HTTPS in Ihrem Webserver aktiviert ist, konfigurieren Sie Weblate bitte mit `ENABLE_HTTPS` oder durch `WEBLATE_ENABLE_HTTPS` im Docker-Container so, dass es korrekt gemeldet wird.

Siehe auch:

Python Social Auth Backend

OpenID-Authentifizierung

Für OpenID-basierte Dienste ist es normalerweise nur eine Frage der Aktivierung. Der folgende Abschnitt aktiviert die OpenID-Authentifizierung für OpenSUSE, Fedora und Ubuntu:

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.email.EmailAuth",
    "social_core.backends.suse.OpenSUSEOpenId",
    "social_core.backends.ubuntu.UbuntuOpenId",
    "social_core.backends.fedora.FedoraOpenId",
    "weblate.accounts.auth.WeblateUserBackend",
)
```

Siehe auch:

OpenID

GitHub-Authentifizierung

Sie müssen eine OAuth-Anwendung auf GitHub registrieren und dann Weblate alle ihre Geheimnisse mitteilen:

```
# 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"]
```

GitHub sollte so konfiguriert sein, dass die Callback-URL `https://example.com/accounts/complete/github/` lautet.

There are similar authentication backends for GitHub for Organizations and GitHub for Teams. Their settings are named `SOCIAL_AUTH_GITHUB_ORG_*` and `SOCIAL_AUTH_GITHUB_TEAM_*`, and they require additional setting of the scope - `SOCIAL_AUTH_GITHUB_ORG_NAME` or `SOCIAL_AUTH_GITHUB_TEAM_ID`. Their callback URLs are `https://example.com/accounts/complete/github-org/` and `https://example.com/accounts/complete/github-teams/`.

Bemerkung: Die von Weblate während der Authentifizierung bereitgestellte Callback-URL enthält die konfigurierte Domäne. Falls Sie Fehlermeldungen über eine nicht übereinstimmende URL erhalten, sollten Sie dies beheben, siehe *Set correct site domain*.

Siehe auch:

GitHub

Bitbucket-Authentifizierung

Sie müssen eine Anwendung bei Bitbucket registrieren und dann Weblate alle ihre Geheimnisse mitteilen:

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.bitbucket.BitbucketOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_BITBUCKET_OAUTH2_KEY = "Bitbucket Client ID"
SOCIAL_AUTH_BITBUCKET_OAUTH2_SECRET = "Bitbucket Client Secret"
SOCIAL_AUTH_BITBUCKET_OAUTH2_VERIFIED_EMAILS_ONLY = True
```

Bemerkung: Die von Weblate während der Authentifizierung bereitgestellte Callback-URL enthält die konfigurierte Domäne. Falls Sie Fehlermeldungen über eine nicht übereinstimmende URL erhalten, sollten Sie dies beheben, siehe *Set correct site domain*.

Siehe auch:

Bitbucket

Google OAuth 2

Um Google OAuth 2 zu verwenden, müssen Sie eine Anwendung auf <https://console.developers.google.com/> registrieren und die Google+ API aktivieren.

Die Weiterleitungs-URL lautet `https://WEBLATE SERVER/accounts/complete/google-oauth2/`

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.google.GoogleOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_GOOGLE_OAUTH2_KEY = "Client ID"
SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET = "Client secret"
```

Bemerkung: Die von Weblate während der Authentifizierung bereitgestellte Callback-URL enthält die konfigurierte Domäne. Falls Sie Fehlermeldungen über eine nicht übereinstimmende URL erhalten, sollten Sie dies beheben, siehe *Set correct site domain*.

Siehe auch:

Google

Facebook OAuth 2

Wie bei „OAuth 2“-Diensten üblich, müssen Sie Ihre Anwendung bei Facebook registrieren. Sobald dies geschehen ist, können Sie Weblate einrichten, um es zu nutzen:

Die Weiterleitungs-URL lautet `https://WEBLATE SERVER/accounts/complete/facebook/`

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.facebook.FacebookOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_FACEBOOK_KEY = "key"
SOCIAL_AUTH_FACEBOOK_SECRET = "secret"
SOCIAL_AUTH_FACEBOOK_SCOPE = ["email", "public_profile"]
```

Bemerkung: Die von Weblate während der Authentifizierung bereitgestellte Callback-URL enthält die konfigurierte Domäne. Falls Sie Fehlermeldungen über eine nicht übereinstimmende URL erhalten, sollten Sie dies beheben, siehe *Set correct site domain*.

Siehe auch:

Facebook

GitLab OAuth 2

Um GitLab OAuth 2 zu verwenden, müssen Sie eine Anwendung auf <https://gitlab.com/profile/applications> registrieren.

Die Weiterleitungs-URL lautet `https://WEBLATE SERVER/accounts/complete/gitlab/` und stellen Sie sicher, dass Sie den Bereich *read_user* markieren.

```
# 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/'
```

Bemerkung: Die von Weblate während der Authentifizierung bereitgestellte Callback-URL enthält die konfigurierte Domäne. Falls Sie Fehlermeldungen über eine nicht übereinstimmende URL erhalten, sollten Sie dies beheben, siehe *Set correct site domain*.

Siehe auch:

GitLab

Microsoft Azure Active Directory

Weblate kann so konfiguriert werden, dass allgemeine oder spezifische Mandanten für die Authentifizierung verwendet werden.

Die Weiterleitungs-URL lautet `https://WEBLATE SERVER/accounts/complete/azuread-oauth2/` für allgemeine und `https://WEBLATE SERVER/accounts/complete/azuread-tenant-oauth2/` für mandantenspezifische Authentifizierung.

```
# 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
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_KEY = ""
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_SECRET = ""
# Tenant ID
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_TENANT_ID = ""
```

Bemerkung: Die von Weblate während der Authentifizierung bereitgestellte Callback-URL enthält die konfigurierte Domäne. Falls Sie Fehlermeldungen über eine nicht übereinstimmende URL erhalten, sollten Sie dies beheben, siehe *Set correct site domain*.

Siehe auch:

Microsoft Azure Active Directory

Slack

Um Slack OAuth 2 zu nutzen, müssen Sie eine Anwendung auf <https://api.slack.com/apps> registrieren.

Die Weiterleitungs-URL lautet `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
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
SOCIAL_AUTH_SLACK_KEY = ""
SOCIAL_AUTH_SLACK_SECRET = ""
```

Bemerkung: Die von Weblate während der Authentifizierung bereitgestellte Callback-URL enthält die konfigurierte Domäne. Falls Sie Fehlermeldungen über eine nicht übereinstimmende URL erhalten, sollten Sie dies beheben, siehe *Set correct site domain*.

Siehe auch:

Slack

Überschreiben von Namen und Symbolen für Authentifizierungsmethoden

Sie können den Anzeigenamen und das Symbol der Authentifizierungsmethode überschreiben, indem Sie die Einstellungen `SOCIAL_AUTH_<NAME>_IMAGE` und `SOCIAL_AUTH_<NAME>_TITLE` verwenden. Zum Beispiel würde das Überschreiben der Benennung für Auth0 wie folgt aussehen:

```
SOCIAL_AUTH_AUTH0_IMAGE = "custom.svg"
SOCIAL_AUTH_AUTH0_TITLE = "Custom auth"
```

Passwort-Authentifizierung deaktivieren

E-Mail- und Passwort-Authentifizierung können ausgeschaltet werden, indem man `social_core.backends.email.EmailAuth` aus `AUTHENTICATION_BACKENDS` entfernt. Behalten Sie `weblate.accounts.auth.WeblateUserBackend` dort, es wird für die Kernfunktionalität von Weblate benötigt.

Disabling e-mail authentication will disable all e-mail related functionality – user invitation or password reset feature.

Tipp: Sie können weiterhin die Passwortauthentifizierung für die Adminoberfläche verwenden, für Benutzer, die Sie dort manuell anlegen. Navigieren Sie einfach zu `/admin/login/`.

Zum Beispiel kann die Authentifizierung nur mit dem openSUSE Open ID Provider wie folgt erreicht werden:

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.suse.OpenSUSEOpenId",
    "weblate.accounts.auth.WeblateUserBackend",
)
```

2.5.4 Passwort-Authentifizierung

The default `settings.py` comes with a reasonable set of `AUTH_PASSWORD_VALIDATORS`:

- Passwörter dürfen Ihren anderen persönlichen Daten nicht zu ähnlich sein.
- Passwörter müssen mindestens 10 Zeichen enthalten.
- Passwörter können kein häufig verwendetes Passwort sein.
- Passwörter dürfen nicht ausschließlich aus Zahlen bestehen.
- Passwörter dürfen nicht aus einem einzigen Zeichen oder nur aus Leerzeichen bestehen.
- Passwörter dürfen nicht mit einem Passwort übereinstimmen, das Sie in der Vergangenheit verwendet haben.

Sie können diese Einstellung an Ihre Passwortrichtlinien anpassen.

Additionally you can also install django-zxcvbn-password which gives quite realistic estimates of password difficulty and allows rejecting passwords below a certain threshold.

2.5.5 SAML-Authentifizierung

Neu in Version 4.1.1.

Bitte folgen Sie den Anweisungen von Python Social Auth für die Konfiguration. Bedeutende Unterschiede:

- Weblate supports single IDP which has to be called `weblate` in `SOCIAL_AUTH_SAML_ENABLED_IDPS`.
- Die URL der SAML-XML-Metadaten lautet `/accounts/metadata/saml/`.
- Die folgenden Einstellungen werden automatisch ausgefüllt: `SOCIAL_AUTH_SAML_SP_ENTITY_ID`, `SOCIAL_AUTH_SAML_TECHNICAL_CONTACT`, `SOCIAL_AUTH_SAML_SUPPORT_CONTACT`

Beispielkonfiguration:

```
# 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/"
SOCIAL_AUTH_SAML_SP_PUBLIC_CERT = "-----BEGIN CERTIFICATE-----"
SOCIAL_AUTH_SAML_SP_PRIVATE_KEY = "-----BEGIN PRIVATE KEY-----"
SOCIAL_AUTH_SAML_ENABLED_IDPS = {
    "weblate": {
        "entity_id": "https://idp.testshib.org/idp/shibboleth",
        "url": "https://idp.testshib.org/idp/profile/SAML2/Redirect/SSO",
        "x509cert": "MIIEDjCCAvagAwIBAgIBADA ... 8Bbnl+ev0peYzxFyF5sQA==",
        "attr_name": "full_name",
        "attr_username": "username",
        "attr_email": "email",
    }
}
SOCIAL_AUTH_SAML_ORG_INFO = {
    "en-US": {
        "name": "example",
        "displayname": "Example Inc.",
        "url": "http://example.com"
    }
}
SOCIAL_AUTH_SAML_TECHNICAL_CONTACT = {
    "givenName": "Tech Gal",
    "emailAddress": "technical@example.com"
}
SOCIAL_AUTH_SAML_SUPPORT_CONTACT = {
    "givenName": "Support Guy",
    "emailAddress": "support@example.com"
}
```

Die Standardkonfiguration extrahiert Benutzerdetails aus den folgenden Attributen; konfigurieren Sie Ihre IDP so, dass sie diese bereitstellt:

Attribut	SAML-URI-Referenz
Vollständiger Name	`urn:oid:2.5.4.3`
Vorname	`urn:oid:2.5.4.42`
Nachname	`urn:oid:2.5.4.4`
E-Mail	`urn:oid:0.9.2342.19200300.100.1.3`
Benutzername	`urn:oid:0.9.2342.19200300.100.1.1`

Hinweis: The example above and the Docker image define an IDP called `weblate`. You might need to configure this string as *Relay* in your IDP.

Siehe auch:

Configuring SAML in Docker, SAML

2.5.6 LDAP-Authentifizierung

Die LDAP-Authentifizierung lässt sich am besten mit dem Paket *django-auth-ldap* erreichen. Sie können es mit den üblichen Mitteln installieren:

```
# Using PyPI
pip install django-auth-ldap>=1.3.0

# Using apt-get
apt-get install python-django-auth-ldap
```

Hinweis: Dieses Paket ist im Docker-Container enthalten, siehe *Installing using Docker*.

Bemerkung: 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.

Sobald Sie das Paket installiert haben, können Sie es mit der Django-Authentifizierung verbinden:

```
# 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
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    # 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
```

Bemerkung: You should remove `'social_core.backends.email.EmailAuth'` from the `AUTHENTICATION_BACKENDS` setting, otherwise users will be able to set their password in Weblate, and authenticate using that. Keeping `'weblate.accounts.auth.WeblateUserBackend'` is still needed in order to make permissions and facilitate anonymous users. It will also allow you to sign in using a local admin account, if you have created it (e.g. by using `createadmin`).

Using bind password

If you can not use direct bind for authentication, you will need to use search, and provide a user to bind for the search. For example:

```
import ldap
from django_auth_ldap.config import LDAPSearch

AUTH_LDAP_BIND_DN = ""
AUTH_LDAP_BIND_PASSWORD = ""
AUTH_LDAP_USER_SEARCH = LDAPSearch(
    "ou=users,dc=example,dc=com", ldap.SCOPE_SUBTREE, "(uid=%(user)s)"
)
```

Integration von Active Directory

```
import ldap
from django_auth_ldap.config import LDAPSearch, NestedActiveDirectoryGroupType

AUTH_LDAP_BIND_DN = "CN=ldap,CN=Users,DC=example,DC=com"
AUTH_LDAP_BIND_PASSWORD = "password"

# User and group search objects and types
AUTH_LDAP_USER_SEARCH = LDAPSearch(
    "CN=Users,DC=example,DC=com", ldap.SCOPE_SUBTREE, "(sAMAccountName=%(user)s)"
)

# Make selected group a superuser in Weblate
AUTH_LDAP_USER_FLAGS_BY_GROUP = {
    # is_superuser means user has all permissions
    "is_superuser": "CN=weblate_AdminUsers,OU=Groups,DC=example,DC=com",
}

# Map groups from AD to Weblate
AUTH_LDAP_GROUP_SEARCH = LDAPSearch(
    "OU=Groups,DC=example,DC=com", ldap.SCOPE_SUBTREE, "(objectClass=group)"
)
AUTH_LDAP_GROUP_TYPE = NestedActiveDirectoryGroupType()
AUTH_LDAP_FIND_GROUP_PERMS = True
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
# Optionally enable group mirroring from LDAP to Weblate
# AUTH_LDAP_MIRROR_GROUPS = True
```

Siehe auch:

Django Authentication Using LDAP, Authentication

2.5.7 CAS-Authentifizierung

Die CAS-Authentifizierung kann mit einem Paket wie *django-cas-ng* erreicht werden.

Schritt eins ist die Offenlegung des E-Mail-Feldes des Benutzers über CAS. Dies muss auf dem CAS-Server selbst konfiguriert werden und setzt voraus, dass Sie mindestens CAS v2 verwenden, da CAS v1 Attribute überhaupt nicht unterstützt.

Der zweite Schritt ist die Aktualisierung von Weblate zur Verwendung Ihres CAS-Servers und Ihrer Attribute.

Um *django-cas-ng* zu installieren:

```
pip install django-cas-ng
```

Sobald Sie das Paket installiert haben, können Sie es an das Django-Authentifizierungssystem anbinden, indem Sie die Datei `settings.py` ändern:

```
# Add CAS backed, keep the Django one if you want to be able to sign in
# even without LDAP for the admin account
AUTHENTICATION_BACKENDS = (
    "django_cas_ng.backends.CASBackend",
    "weblate.accounts.auth.WeblateUserBackend",
)

# CAS server address
CAS_SERVER_URL = "https://cas.example.net/cas/"

# Add django_cas_ng somewhere in the list of INSTALLED_APPS
INSTALLED_APPS = (..., "django_cas_ng")
```

Finally, a signal can be used to map the e-mail field to the user object. For this to work you have to import the signal from the *django-cas-ng* package and connect your code with this signal. Doing this in settings file can cause problems, therefore it's suggested to put it:

- In your app config's `django.apps.AppConfig.ready()` method
- In the project's `urls.py` file (when no models exist)

```
from django_cas_ng.signals import cas_user_authenticated
from django.dispatch import receiver


@receiver(cas_user_authenticated)
def update_user_email_address(sender, user=None, attributes=None, **kwargs):
    # If your CAS server does not always include the email attribute
    # you can wrap the next two lines of code in a try/catch block.
    user.email = attributes["email"]
    user.save()
```

Siehe auch:

Django CAS NG

2.5.8 Konfigurieren der Django-Authentifizierung von Drittanbietern

Generell sollte jedes Django-Authentifizierungs-Plugin mit Weblate funktionieren. Folgen Sie einfach den Anweisungen für das Plugin und denken Sie daran, das Weblate-Benutzer-Backend installiert zu lassen.

Siehe auch:

LDAP-Authentifizierung, *CAS-Authentifizierung*

Typically the installation will consist of adding an authentication backend to `AUTHENTICATION_BACKENDS` and installing an authentication app (if there is any) into `INSTALLED_APPS`:

```
AUTHENTICATION_BACKENDS = (
    # Add authentication backend here
    "weblate.accounts.auth.WeblateUserBackend",
)

INSTALLED_APPS += (
    # Install authentication app here
)
```

2.6 Zugriffssteuerung

Weblate verfügt über ein sehr präzises Berechtigungssystem, um Benutzerrechte für die ganze Instanz oder nur in begrenztem Umfang zuzuweisen.

Geändert in Version 3.0: Vor Weblate 3.0 beruhte die Berechtigungsstruktur auf Django, ist nun aber eine speziell für Weblate erstellte. Sollten Sie eine ältere Version verwenden, konsultieren Sie bitte die Dokumentation zu jener Version, die Informationen hier gelten für sie nicht.

2.6.1 Einfache Zugriffskontrolle

Wenn Sie nicht die gesamte Weblate-Installation administrieren und nur Zugriff auf die Verwaltung bestimmter Projekte haben (wie bei Hosted Weblate), beschränken sich Ihre Möglichkeiten zur Verwaltung der Zugriffskontrolle auf folgende Einstellungen. Wenn Sie keine komplexe Einrichtung benötigen, sind diese für Sie ausreichend.

Projekt-Zugriffssteuerung

Bemerkung: Diese Funktion ist für Projekte im Freien Hosting-Angebot auf Hosted Weblate nicht verfügbar.

Sie können den Zugriff des Benutzers auf einzelne Projekte einschränken, indem Sie eine andere *Zugriffskontrolle*-Einstellung wählen. Verfügbare Optionen sind:

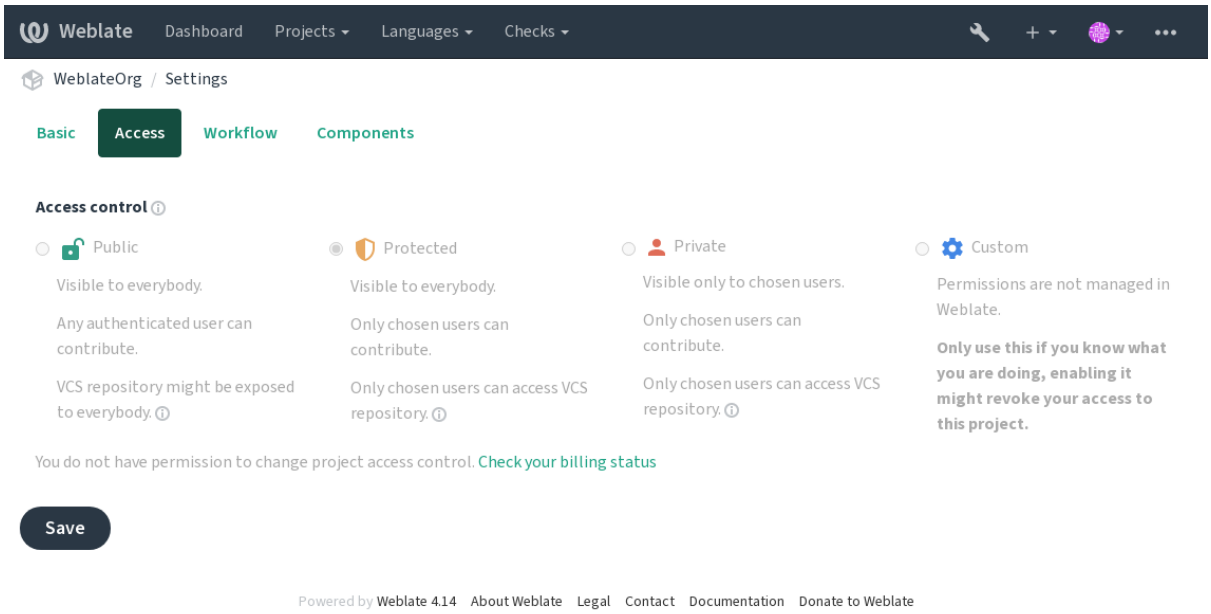
Öffentlich
: Öffentlich sichtbar, übersetzbar für alle angemeldeten Benutzer.

Geschützt
: Öffentlich sichtbar, aber nur ausgewählte Benutzer können übersetzen.

Privat
: Nur ausgewählte Benutzer können es sehen und übersetzen.

Benutzerdefiniert
: Die Funktionen von *Benutzerverwaltung* werden deaktiviert; standardmäßig ist es allen Benutzern untersagt, irgendwelche Aktionen am Projekt durchzuführen. Sie müssen alle Berechtigungen mit *Benutzerdefinierte Zugriffskontrolle* einrichten.

Die *Zugriffssteuerung* kann im Reiter *Zugriff* der Konfiguration (*Verwaltung* ↓ *Einstellungen*) des jeweiligen Projekts geändert werden.



Der Standardwert kann durch `DEFAULT_ACCESS_CONTROL` geändert werden.

Bemerkung: Auch bei *Privat*-Projekten werden einige Informationen über Ihr Projekt offengelegt: Statistik und Sprachzusammenfassung für die gesamte Instanz enthalten trotz der Einstellung der Zugriffskontrolle Zählungen für alle Projekte. Ihr Projektname und andere Informationen können dadurch nicht offengelegt werden.

Bemerkung: Der tatsächliche Satz an Berechtigungen, der standardmäßig für Benutzer in den Projekten *Public*, *Protected* und *Private* verfügbar ist, kann vom Administrator der Weblate-Instanz mit *custom settings* neu definiert werden.

Warnung: Wenn Sie die Zugriffskontrolle *Benutzerdefiniert* aktivieren, entfernt Weblate alle *speziellen Gruppen*, die es für ein ausgewähltes Projekt erstellt hat. Wenn Sie dies ohne Admin-Rechte für die gesamte Weblate-Instanz tun, verlieren Sie sofort Ihren Zugang zur Verwaltung des Projekts.

Siehe auch:

Zugriffssteuerung

Verwaltung der Zugriffssteuerung nach Projekt

Benutzer mit der Berechtigung *Projektzugriff verwalten* (siehe Privilegien) können Benutzer in Projekten verwalten, indem sie die Benutzer zu Teams hinzufügen. Sie können Benutzer einer der folgenden Gruppen zuordnen.

Folgende Teams werden für jedes Projekt automatisch gebildet:

Für *Public*, *Protected* und *Private* Projekte:

Verwaltung
: Enthält alle für das Projekt verfügbaren Berechtigungen.

Überprüfung (nur wenn *Review Workflow* eingeschaltet ist)
: Darf Übersetzungen bei der Überprüfung freigeben.

Nur für *geschützte* und *private* Projekte:

Übersetzen
Darf das Projekt übersetzen und offline erstellte Übersetzungen hochladen.

Quellcode
Kann Ausgangszeichenketten (wenn in den *project settings* erlaubt) und Infos zu Ausgangszeichenketten bearbeiten.

Sprachen
Darf übersetzte Sprachen verwalten (Übersetzungen hinzufügen oder entfernen).

Glossar
Darf das Glossar verwalten (Einträge ergänzen oder entfernen oder hochladen).

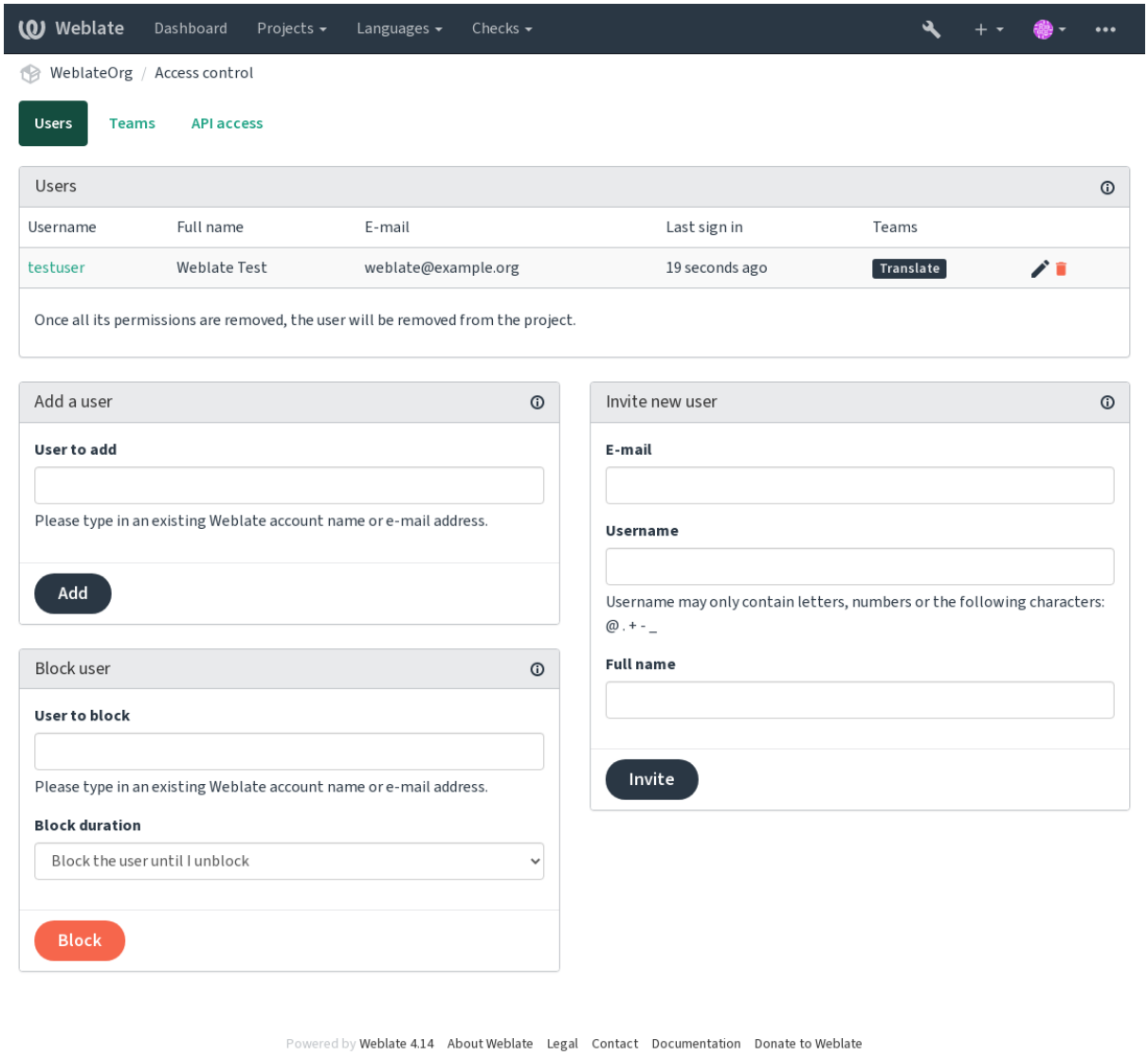
Speicher
Darf den Übersetzungsspeicher verwalten.

Bildschirmfotos
Darf Bildschirmfotos verwalten (sie hinzufügen, entfernen und mit Ausgangszeichenketten verbinden).

Automatische Übersetzung
Darf automatische Übersetzungen verwenden.

VCS
Darf das VCS verwalten und auf das exportierte Repository zugreifen.

Abrechnung
Darf auf Abrechnungsdaten und -einstellungen zugreifen (siehe *Abrechnung*).



Diese Funktionen sind auf der Seite *Zugriffssteuerung* verfügbar, die über das Projektmenü *Verwaltung* ↓ *Benutzer* aufgerufen werden kann.

Neue Benutzereinladung

Neben dem Hinzufügen eines bestehenden Benutzers zum Projekt ist es auch möglich, neue einzuladen. Jeder neue Benutzer wird sofort erstellt, aber das Konto bleibt bis zur Anmeldung mit einem Link in der Einladung, die per E-Mail gesendet wird, inaktiv. Dazu sind keine standortweiten Berechtigungen erforderlich, eine Zugriffsverwaltungsberechtigung für den Umfang des Projekts (z. B. eine Mitgliedschaft im Team *Administration*) wäre ausreichend.

Hinweis: Wenn der eingeladene Benutzer die Gültigkeit der Einladung versäumt hat, kann er sein Passwort mit der eingeladenen E-Mail-Adresse im Formular zum Zurücksetzen des Passworts festlegen, da das Konto bereits erstellt wurde.

Neu in Version 3.11: Es ist möglich, die E-Mail für Benutzereinladungen erneut zu senden (wodurch zuvor gesendete Einladungen ungültig werden).

Die gleiche Art von Einladungen sind Plattformweit über die *management interface* auf der Registerkarte *Users* verfügbar.

Benutzer sperren

Neu in Version 4.7.

Für den Fall, dass sich einige Benutzer in Ihrem Projekt schlecht benehmen, haben Sie die Möglichkeit, sie für Beiträge zu sperren. Der gesperrte Benutzer kann das Projekt immer noch sehen, wenn er die entsprechenden Berechtigungen hat, aber er kann nichts dazu beitragen.

Berechtigungsverwaltung pro Projekt

Sie können Ihre Projekte auf *Protected* oder *Private* einstellen und *manage users* pro Projekt in der Weblate-Benutzeroberfläche.

Standardmäßig verhindert dies, dass Weblate den Zugriff von *Benutzer* und *Betrachter* *default groups* aufgrund der eigenen Konfiguration dieser Gruppen gewährt. Dies hindert Sie jedoch nicht daran, die Berechtigungen für diese Projekte plattformweit zu gewähren, indem Sie die Standardgruppen ändern, eine neue Gruppe erstellen oder zusätzliche benutzerdefinierte Einstellungen für einzelne Komponenten vornehmen, wie in *Benutzerdefinierte Zugriffskontrolle* unten beschrieben.

One of the main benefits of managing permissions through the Weblate user interface is that you can delegate it to other users without giving them the superuser privilege. In order to do so, add them to the *Administration* team of the project.

2.6.2 Benutzerdefinierte Zugriffskontrolle

Bemerkung: Diese Funktion ist für Projekte im Freien Hosting-Angebot auf Hosted Weblate nicht verfügbar.

Die Berechtigungsstruktur basiert auf Gruppen und Rollen, wobei Rollen eine Reihe von Berechtigungen festlegen und Gruppen sie mit Benutzern und Übersetzungen verknüpfen, weitere Einzelheiten dazu unter *Benutzer, Rollen, Gruppen, und Berechtigungen*.

Die leistungsstärksten Funktionen der Zugriffssteuerung von Weblate sind bis jetzt nur über das *Django admin interface* erreichbar. Sie können es zur Verwaltung der Berechtigungen jedes Projekts benutzen. Sie müssen es nicht zwingend auf *Custom access control* umstellen, um es zu verwenden. Sie müssen allerdings Superuser-Rechte besitzen, um es zu nutzen.

Wenn Sie nicht an den Details der Implementierung interessiert sind und nur eine einfache Konfiguration auf der Grundlage der Standardeinstellungen erstellen möchten oder keinen plattformweiten Zugriff auf die gesamte Weblate-Installation haben (wie bei Hosted Weblate), lesen Sie bitte den Abschnitt *Einfache Zugriffskontrolle*.

Allgemeine Einstellungen

Dieser Abschnitt enthält einen Überblick über einige gängige Konfigurationen, die Sie interessieren könnten.

Plattformweite Rechteverwaltung

Um Berechtigungen für eine ganze Instanz auf einmal zu verwalten, fügen Sie Benutzer zu den entsprechenden *Standardgruppen* hinzu:

- *Benutzer* (dies geschieht standardmäßig durch *automatic group assignment*).
- *Prüfer* (wenn Sie *review workflow* mit eigenen Prüfern verwenden).
- *Manager* (wenn Sie die meisten Verwaltungsaufgaben an jemand anderen delegieren wollen).

Sie sollten alle Projekte als *Public* konfigurieren (siehe *Projekt-Zugriffssteuerung*), da sonst die plattformweiten Berechtigungen, die durch die Mitgliedschaft in den Gruppen *Benutzer* und *Prüfer* gegeben sind, keine Wirkung haben.

Sie können den Standardgruppen auch einige zusätzliche Berechtigungen Ihrer Wahl erteilen. So können Sie z. B. allen „Benutzern“ die Berechtigung zum Verwalten von Bildschirmfotos erteilen.

Sie können auch einige neue benutzerdefinierte Gruppen definieren. Wenn Sie Ihre Berechtigungen für diese Gruppen weiterhin plattformweit verwalten möchten, wählen Sie einen geeigneten Wert für *Projektauswahl* (z.B. *Alle Projekte* oder *Alle öffentlichen Projekte*).

Eigene Berechtigungen für Sprachen, Komponenten oder Projekte

Sie können Ihre eigenen Gruppen erstellen, um die Berechtigungen für bestimmte Objekte wie Sprachen, Komponenten und Projekte zu verwalten. Obwohl diese Gruppen nur zusätzliche Berechtigungen gewähren, können Sie keine Berechtigung, die von plattformweiten oder projektspezifischen Gruppen gewährt wurde, durch Hinzufügen einer anderen benutzerdefinierten Gruppe aufheben.

Beispiel:

Wenn Sie (aus welchen Gründen auch immer) die Übersetzung in eine bestimmte Sprache (sagen wir mal *Tschechisch*) nur einer geschlossenen Gruppe von zuverlässigen Übersetzern erlauben wollen, während die Übersetzungen in andere Sprachen öffentlich bleiben, müssen Sie das tun:

1. Entziehen Sie allen Benutzern die Erlaubnis, *Tschechisch* zu übersetzen. In der Standardkonfiguration kann dies durch Ändern der *Benutzer default group* erfolgen.

Tab. 1: Gruppe *Benutzer*

Sprachauswahl	*Wie vorgegeben*
Sprachen	Alle außer *Tschechisch*

2. Fügen Sie eine eigene Gruppe für *Tschechisch*-Übersetzer hinzu.

Tab. 2: Gruppe *Tschechische Übersetzer*

Rollen	*Hauptbenutzer*
Projektauswahl	*Alle öffentlichen Projekte*
Sprachauswahl	*Wie vorgegeben*
Sprachen	*Tschechisch*

3. Fügen Sie die Benutzer, denen Sie die Berechtigungen erteilen möchten, zu dieser Gruppe hinzu.

Wie Sie sehen, ist die Verwaltung von Berechtigungen auf diese Weise zwar sehr leistungsfähig, kann aber auch ziemlich mühsam sein. Sie können sie nicht an einen anderen Benutzer delegieren, es sei denn, Sie erteilen Superuser-Berechtigungen.

Benutzer, Rollen, Gruppen, und Berechtigungen

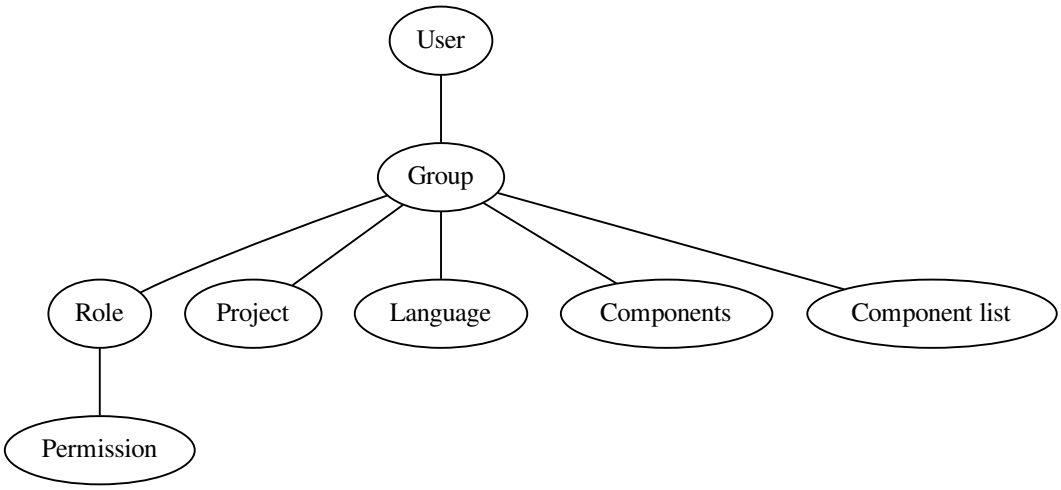
Die Legitimierungsmodelle bestehen aus verschiedenen Objekten:

Berechtigung
Durch Weblate bestimmte individuelle Berechtigung. Sie können keine individuellen Berechtigungen zuteilen, dies kann nur durch Rollenzuordnungen geschehen.

Rolle
Eine Rolle bestimmt eine Zusammenstellung von Genehmigungen. Dies ermöglicht die Wiederverwendung dieser Zusammenstellungen an verschiedenen Orten, zur Vereinfachung der Verwaltung.

Benutzer
Der Benutzer kann zu mehreren Gruppen gehören.

Gruppe
Die Gruppe verbindet Rollen, Benutzer und Legitimierungsobjekte (Projekte, Sprachen, und Komponentenlisten).



Bemerkung: Einer Gruppe können auch keine Rollen zugewiesen werden. In diesem Fall wird davon ausgegangen, dass jeder das Projekt durchsuchen kann (siehe unten).

Kontrolle des Zugriffs auf ein Projekt

Ein Benutzer muss Mitglied einer Gruppe sein, die mit dem Projekt oder einer Komponente in diesem Projekt verknüpft ist. Eine Mitgliedschaft allein genügt, es sind keine speziellen Genehmigungen erforderlich, um ein Projekt zu durchsuchen (dies wird in der Standardgruppe *Betrachter* angewendet, siehe *Liste der Gruppen*).

Kontrolle des Zugriffs auf eine Komponente

Ein Benutzer kann auf nicht eingeschränkte Komponenten zugreifen, sobald er auf das Projekt der Komponente zugreifen kann (und über alle Rechte verfügt, die dem Benutzer für das Projekt gewährt wurden). Wenn *Restricted access* eingeschaltet ist, erfordert der Zugriff auf die Komponente explizite Berechtigungen für die Komponente (oder eine Komponentenliste, in der die Komponente enthalten ist).

Umfang der Gruppen

Der Umfang der von den Rollen in den Gruppen zugewiesenen Berechtigungen wird nach den folgenden Regeln angewendet:

- Wenn die Gruppe eine *Komponentenliste* angibt, werden alle Berechtigungen, die Mitgliedern dieser Gruppe gewährt werden, für alle Komponenten in den Komponentenlisten, die der Gruppe zugeordnet sind, gewährt, und ein Zugriff ohne zusätzliche Berechtigungen wird für alle Projekte gewährt, in denen sich diese Komponenten befinden. *Komponenten* und *Projekte* werden ignoriert.
- Wenn die Gruppe einen *Komponenten* angibt, werden alle Berechtigungen, die den Mitgliedern dieser Gruppe erteilt wurden, für alle Komponenten gewährt, die der Gruppe zugeordnet sind, und ein Zugriff ohne zusätzliche Berechtigungen wird für alle Projekte gewährt, in denen diese Komponenten enthalten sind. *Projekte* werden ignoriert.

- Andernfalls, wenn die Gruppe irgendwelche *Projekte* angibt, entweder durch direkte Auflistung oder indem *Projektauswahl* auf einen Wert wie *Alle öffentlichen Projekte* gesetzt wird, werden alle diese Berechtigungen auf alle Projekte angewandt, was effektiv die gleichen Berechtigungen für den Zugriff auf alle Projekte *unrestricted components* gewährt.
- Die durch das *Sprachen* einer Gruppe auferlegten Einschränkungen, werden separat angewandt, wenn überprüft wird, ob ein Benutzer die Berechtigung für bestimmte durchzuführende Aktionen hat. Sie werden nämlich nur auf Aktionen angewendet, die direkt mit dem Übersetzungsprozess selbst zusammenhängen, wie z. B. das Überprüfen, Speichern von Übersetzungen, Hinzufügen von Vorschlägen usw.

Hinweis: Verwenden Sie *Sprachauswahl* oder *Projektauswahl* um automatisch alle Sprachen oder Projekte einzuschließen.

Beispiel:

Angenommen, es gibt ein Projekt `foo` mit den Komponenten: `foo/bar` und `foo/baz` und die folgende Gruppe:

Tab. 3: Gruppe ` Administrator-Prüfer Spanisch`

Rollen	*Review Strings*, *Repository verwalten*
Komponenten	foo/bar
Sprachen	*Spanisch*

Die Mitglieder dieser Gruppe haben folgende Rechte (unter Annahme der Standardrolleneinstellungen):

- Allgemeiner Zugriff (zur Durchsicht) auf das gesamte Projekt `foo` einschließlich der beiden Komponenten darin: `foo/bar` und `foo/baz`.
- Zeichenketten in der spanischen Übersetzung von `foo/bar` überprüfen (nirgendwo sonst).
- VCS für das gesamte `foo/bar`-Repository verwalten, z. B. ausstehende Änderungen von Übersetzern für alle Sprachen übertragen.

Automatische Gruppenzuordnungen

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.

Der häufigste Anwendungsfall für diese Funktion besteht darin, alle neuen Benutzer einer Standardgruppe zuzuordnen. Um dies zu erreichen, werden Sie wahrscheinlich den Standardwert (`^.*$`) im Feld für den regulären Ausdruck beibehalten wollen. Ein weiterer Anwendungsfall für diese Option könnte darin bestehen, den Mitarbeitern Ihres Unternehmens standardmäßig einige zusätzliche Privilegien zu gewähren. Unter der Annahme, dass alle Mitarbeiter eine Firmen-E-Mail-Adresse in Ihrer Domäne verwenden, lässt sich dies mit einem Ausdruck wie `^.*@mycompany.com` bewerkstelligen.

Bemerkung: Die automatische Gruppenzuordnung zu ‚Benutzer' und ‚Betrachter' wird immer neu erstellt, wenn Sie von einer Weblate-Version auf eine andere aktualisieren. Wenn Sie dies abstellen möchten, setzen Sie den regulären Ausdruck auf `^$` (der auf nichts zutrifft).

Bemerkung: Im Moment gibt es noch keine Möglichkeit, bereits bestehende Benutzer über die Benutzeroberfläche zu einer Gruppe hinzuzufügen. Hierfür können Sie auf die *REST API* zurückgreifen.

Standardgruppen und -rollen

Nach der Installation wird ein Standardsatz von Gruppen erstellt (Siehe *Liste der Gruppen*).

Diese Rollen und Gruppen werden bei der Installation erstellt. Die eingebauten Rollen werden durch die Datenbankmigration beim Upgrade immer auf dem neuesten Stand gehalten. Sie können sie nicht ändern. Bitte definieren Sie eine neue Rolle, wenn Sie Ihre eigenen Berechtigungen festlegen möchten.

Liste der Berechtigungen und integrierten Rollen

Zielgruppe	Berechtigung	Rollen
Abrechnung (siehe Abrechnung)	Abrechnungsinformationen anzeigen	*Verwaltung*, *Abrechnung*
Änderungen	Änderungen herunterladen	*Verwaltung*
Kommentare	Kommentar erstellen	*Administration*, *Edit source*, *Pow*
	Kommentar löschen	*Verwaltung*
	Kommentar lösen	*Verwaltung*, *Zeichenketten überp*
Komponente	Komponenteneinstellungen bearbeiten	*Verwaltung*
	Komponente sperren, um Übersetzungen zu verhindern	*Verwaltung*
Glossar	Glossareintrag hinzufügen	*Verwaltung*, *Glossar verwalten*,
	Glossareintrag bearbeiten	*Verwaltung*, *Glossar verwalten*,
	Glossareintrag löschen	*Verwaltung*, *Glossar verwalten*,
	Glossareinträge hochladen	*Verwaltung*, *Glossar verwalten*,
Automatische Vorschläge	Automatische Vorschläge verwenden	*Administration*, *Edit source*, *Pow*
Übersetzungsspeicher	Übersetzungsspeicher bearbeiten	*Verwaltung*, *Übersetzungsspeich*
	Übersetzungsspeicher löschen	*Verwaltung*, *Übersetzungsspeich*
Projekte	Projekteinstellungen bearbeiten	*Verwaltung*
	Projektzugang verwalten	*Verwaltung*
Auswertungen	Berichte herunterladen	*Verwaltung*
Bildschirmfotos	Bildschirmfoto hinzufügen	*Verwaltung*, *Bildschirmfotos ver*
	Bildschirmfoto bearbeiten	*Verwaltung*, *Bildschirmfotos ver*
	Bildschirmfoto löschen	*Verwaltung*, *Bildschirmfotos ver*
Ausgangszeichenketten	Zusätzliche Zeichenketteninformationen bearbeiten	*Administration*, *Edit source*
Zeichenketten	Neue Zeichenkette hinzufügen	*Verwaltung*
	Entfernen einer Zeichenkette	*Verwaltung*
	Fehlerhafte Prüfung ablehnen	*Administration*, *Edit source*, *Pow*
	Zeichenkette bearbeiten	*Administration*, *Edit source*, *Pow*
	Zeichenketten überprüfen	*Verwaltung*, *Zeichenketten überp*
	Zeichenkette bearbeiten, wenn Vorschläge erzwungen werden	*Verwaltung*, *Zeichenketten überp*
	Ausgangszeichenketten bearbeiten	*Administration*, *Edit source*, *Pow*
Vorschläge	Vorschlag annehmen	*Administration*, *Edit source*, *Pow*
	Vorschlag hinzufügen	*Administration*, *Edit source*, *Ada*
	Vorschlag löschen	*Verwaltung*, *Hauptbenutzer*
	Abstimmung über Vorschlag	*Administration*, *Edit source*, *Pow*
Übersetzungen	Sprache zur Übersetzung hinzufügen	*Verwaltung*, *Hauptbenutzer*, *Spr*
	Automatische Übersetzung durchführen	*Verwaltung*, *Automatische Vorsc*
	Vorhandene Übersetzung löschen	*Verwaltung*, *Sprachen verwalten*
	Übersetzungsdatei herunterladen	*Administration*, *Edit source*, *Acc*
	Mehrere Sprachen zur Übersetzung hinzufügen	*Verwaltung*, *Sprachen verwalten*
Uploads	Autor der hochgeladenen Übersetzung definieren	*Verwaltung*
	Vorhandene Zeichenketten mit Hochladung überschreiben	*Administration*, *Edit source*, *Pow*
	Übersetzungen hochladen	*Administration*, *Edit source*, *Pow*
VCS	Zugriff auf das interne Repository	*Administration*, *Access repositor*
	Commit für Änderungen am internen Repository durchführen	*Verwaltung*, *Repository verwalte*
	Änderung aus dem internen Repository pushen	*Verwaltung*, *Repository verwalte*
	Änderungen im internen Repository zurücksetzen	*Verwaltung*, *Repository verwalte*

Tab. 4 – Fortsetzung der vorherigen Seite

Zielgruppe	Berechtigung	Rollen
	Speicherort des Upstream-Repositorys anzeigen	*Administration*, *Access repositor*
	Aktualisierung des internen Repositorys	*Verwaltung*, *Repository verwalte*
Plattformweite Berechtigungen	Managementoberfläche nutzen	
	Neue Projekte hinzufügen	
	Sprachdefinitionen hinzufügen	
	Sprachdefinitionen verwalten	
	Gruppen verwalten	
	Benutzer verwalten	
	Rollen verwalten	
	Mitteilungen verwalten	
	Übersetzungsspeicher verwalten	
	Übersetzungsmaschinen verwalten	
	Komponentenlisten verwalten	

Bemerkung: Plattformweite Berechtigungen werden keiner Standardrolle gewährt. Diese sind sehr mächtig und kommen dem Superuser-Status recht nahe. Die meisten von ihnen betreffen alle Projekte in Ihrer Weblate-Installation.

Liste der Gruppen

Die folgenden Gruppen werden bei der Installation (oder nach dem Ausführen von `setupgroups`) erstellt, und Sie können diese nach Belieben ändern. Wenn Sie die Gruppen jedoch löschen oder umbenennen, werden sie bei der Migration neu erstellt.

Gäste

Legt die Berechtigungen für nicht authentifizierte Benutzer fest.

Diese Gruppe enthält ausschließlich anonyme Benutzer (siehe `ANONYMOUS_USER_NAME`).

Sie können Rollen aus dieser Gruppe entfernen, um die Berechtigungen für nicht-authentifizierte Benutzer einzuschränken.

Standard-Rollen: *Vorschlag hinzufügen*, *Auf Repository zugreifen*

Betrachter

Diese Rolle stellt die Sichtbarkeit der öffentlichen Projekte für alle Benutzer sicher. Standardmäßig sind alle Benutzer Mitglieder dieser Gruppe.

Standardmäßig werden alle neuen Konten durch *automatic group assignment* zu Mitgliedern dieser Gruppe, wenn sie ihr beitreten.

Standard-Rollen: keine

Benutzer

Standardgruppe für alle Benutzer.

Standardmäßig werden alle neuen Konten durch *automatic group assignment* zu Mitgliedern dieser Gruppe, wenn sie ihr beitreten.

Standard-Rollen: *Hauptbenutzer*

Prüfer

Gruppe für Prüfer (siehe *Übersetzungs-Workflows*).

Standard-Rollen: *Zeichenketten überprüfen*

Manager

Gruppe für Administratoren.

Standard-Rollen: *Verwaltung*

Warnung: Entfernen Sie niemals die vordefinierten Weblate-Gruppen und -Benutzer, da dies zu unerwarteten Problemen führen kann! Wenn Sie keine Verwendung für sie haben, können Sie stattdessen alle ihre Berechtigungen entfernen.

2.6.3 Zusätzliche Zugriffsbeschränkungen

Wenn Sie Ihre Weblate-Installation weniger öffentlich nutzen möchten, d.h. neue Benutzer nur auf Einladungsbasis zulassen möchten, können Sie Weblate so konfigurieren, dass nur bekannte Benutzer darauf zugreifen können. Um dies zu tun, müssen Sie `REGISTRATION_OPEN` auf `False` setzen, um das registrieren neuer Nutzer zu verhindern, und `REQUIRE_LOGIN` auf `/.*` damit Sie sich anmelden müssen, um auf alle Seiten der Website zugreifen zu können.

Hinweis: Sie können integrierte :*Neue Benutzereinladung* verwenden, um neue Benutzer hinzuzufügen.

2.7 Übersetzungsprojekte

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 *Dateimaske* defined in *Component configuration*) appear in the VCS repository.

Weblate supports a wide range of translation formats (both bilingual and monolingual ones) supported by Translate Toolkit, see *Supported file formats*.

Bemerkung: You can share cloned VCS repositories using *Weblate internal URLs*. Using this feature is highly recommended when you have many components sharing the same VCS. It improves performance and decreases required disk space.

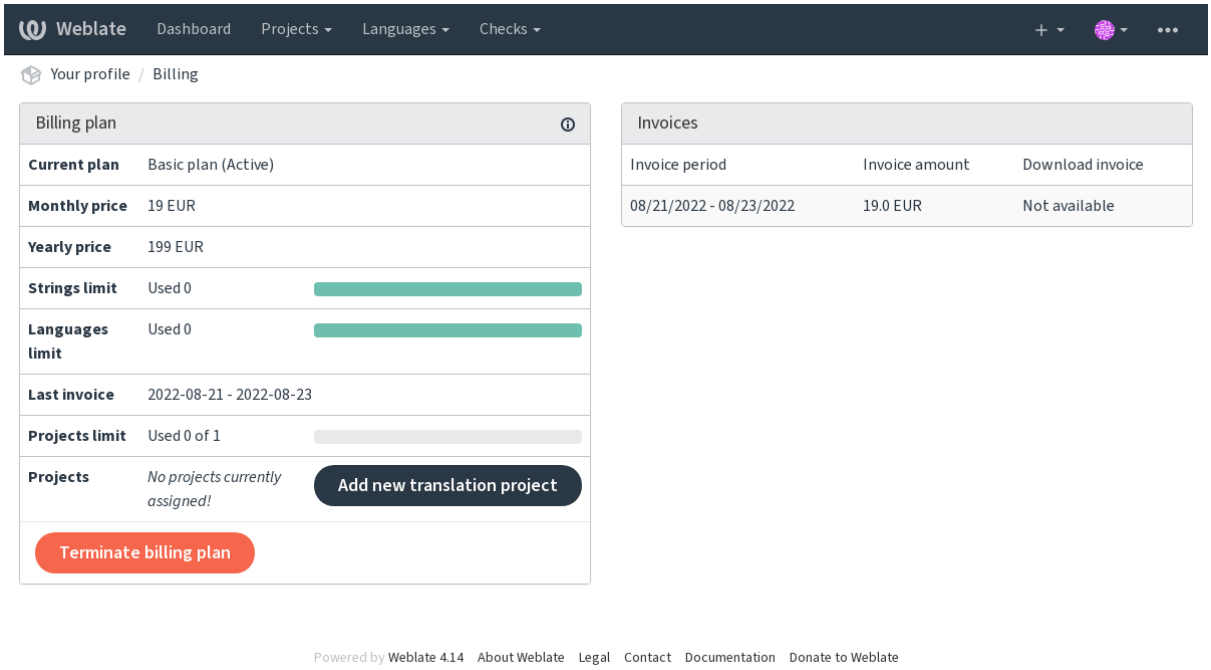
2.7.2 Adding translation projects and components

Geändert in Version 3.2: An interface for adding projects and components is included, and you no longer have to use *The Django admin interface*.

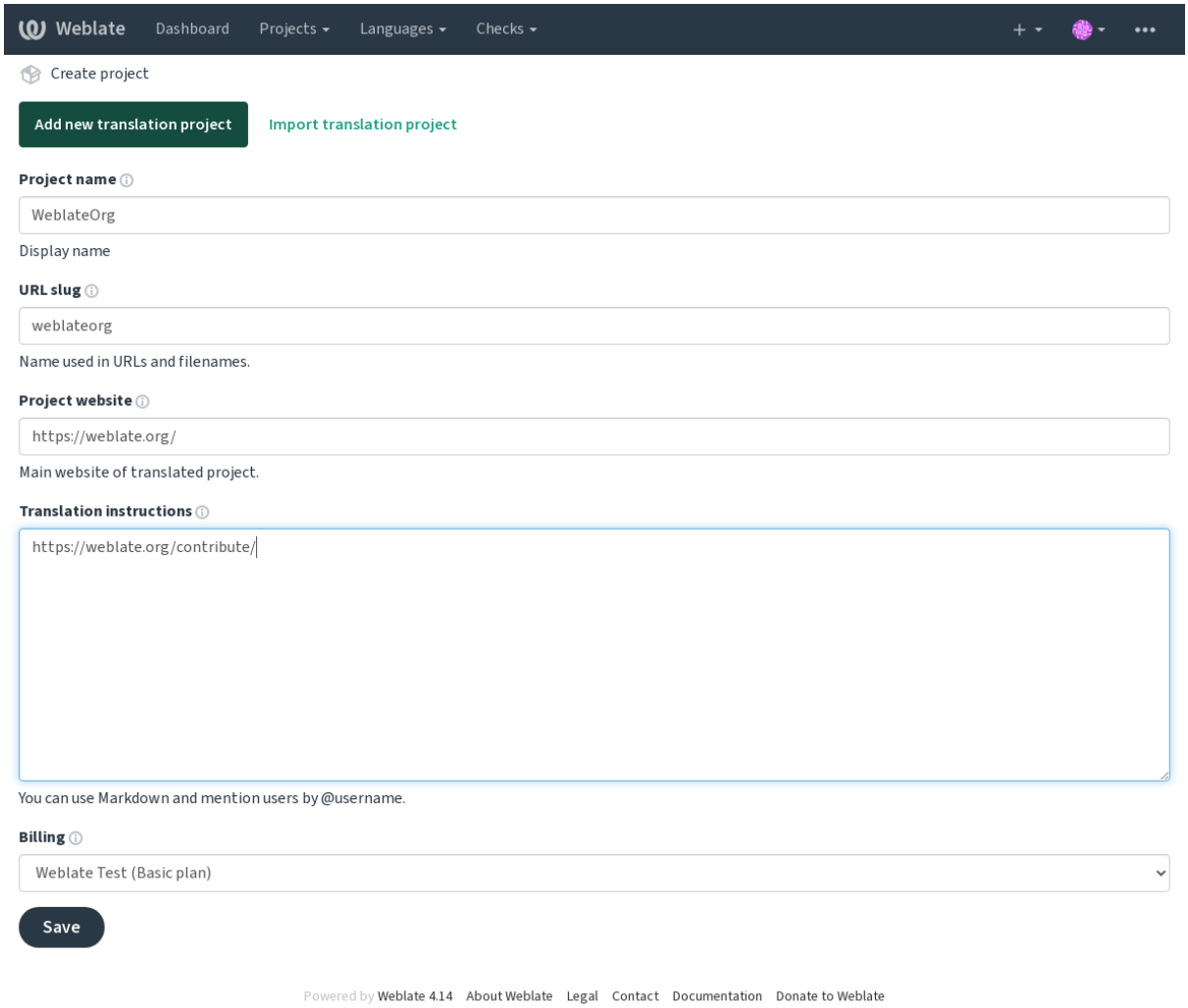
Geändert in Version 3.4: The process of adding components is now multi staged, with automated discovery of most parameters.

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/ see *Abrechnung*), you can also create those based on your plans allowance from the user account that manages billing.

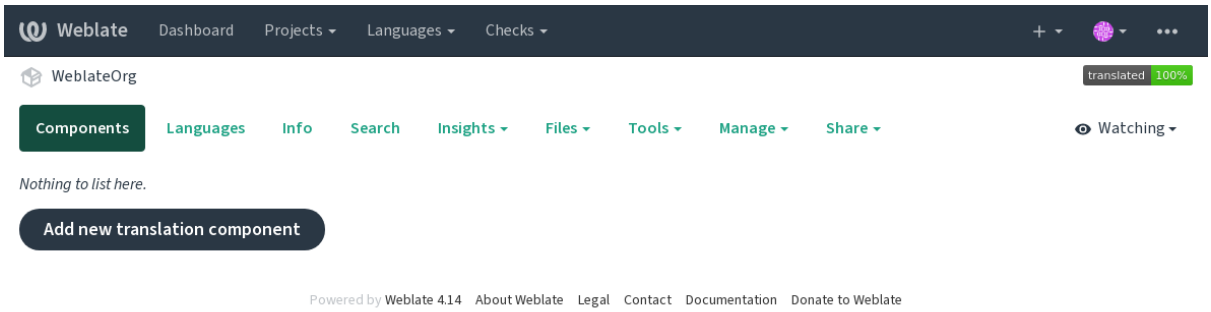
You can view your current billing plan on a separate page:



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:

Von der Versionskontrolle
Creates component from remote version control repository.

Aus vorhandener Komponente
Creates additional component to existing one by choosing different files.

Zusätzlicher Zweig
Creates additional component to existing one, just for different branch.

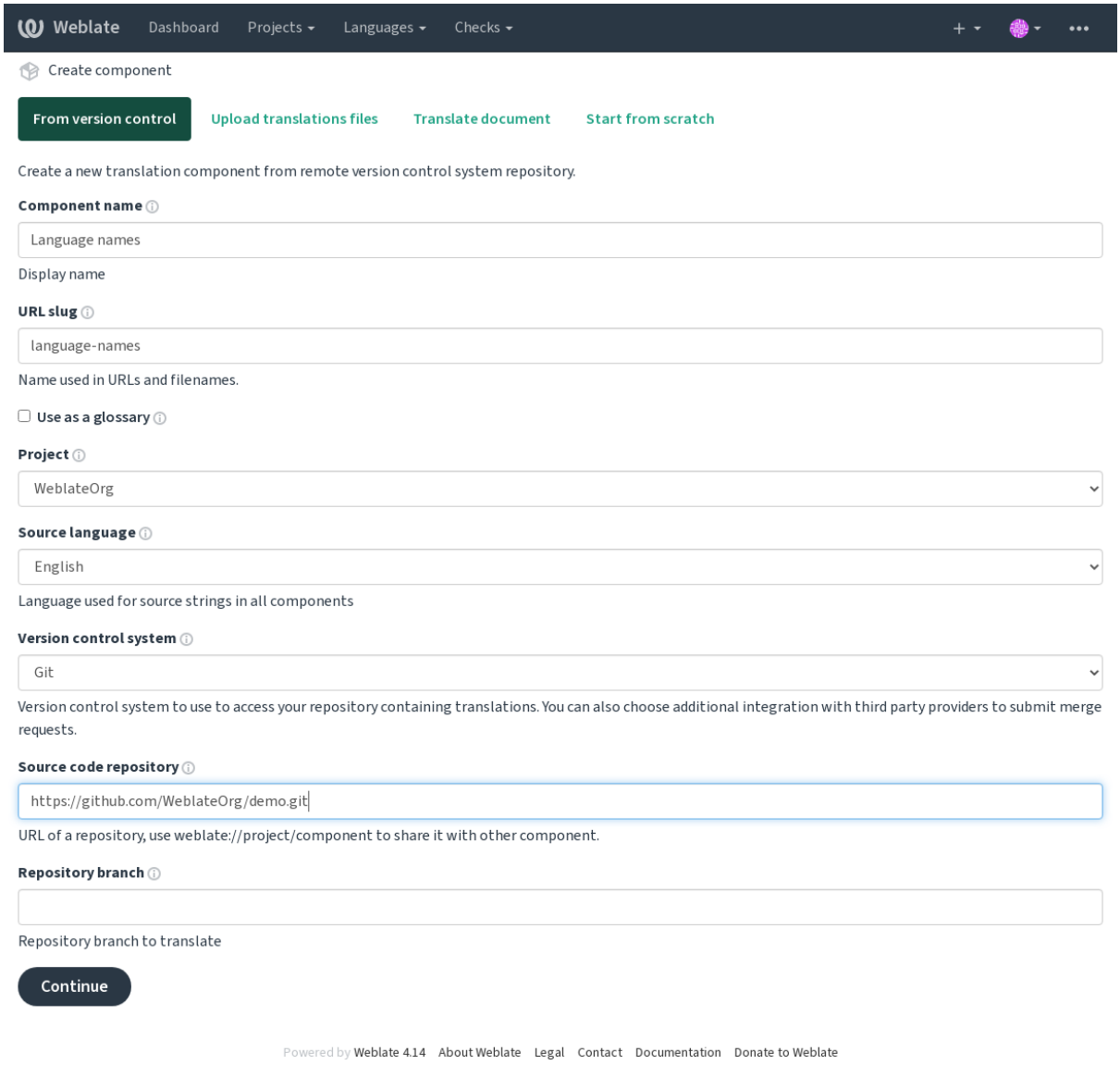
Übersetzungsdateien hochladen
Upload translation files to Weblate in case you do not have version control or do not want to integrate it with Weblate. You can later update the content using the web interface or *Weblate's REST API*.

Dokument übersetzen
Laden Sie ein einzelnes Dokument oder eine Übersetzungsdatei hoch und übersetzen Sie diese.

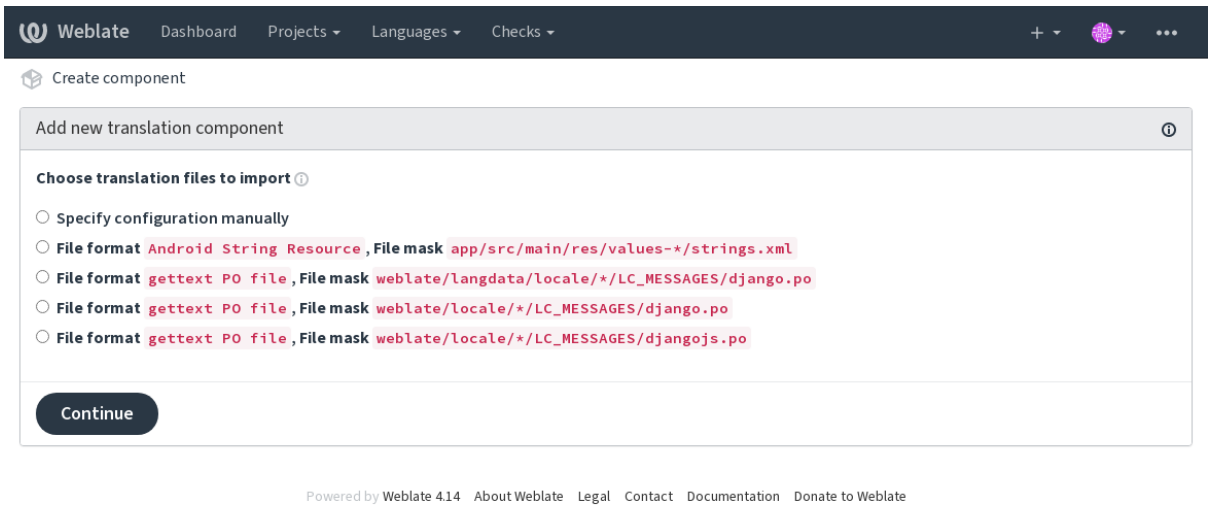
Bei Null anfangen
Create blank translation project and add strings manually.

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:

Weblate Dashboard Projects Languages Checks

Create component

Add new translation component

Project

WeblateOrg

Component name

Language names

Display name

URL slug

language-names

Name used in URLs and filenames.

Version control system

Git

Version control system to use to access your repository containing translations. You can also choose additional integration with third party providers to submit merge requests.

Source code repository

https://github.com/WeblateOrg/demo.git

URL of a repository, use weblate://project/component to share it with other component.

Repository branch

Repository branch to translate

Repository push URL

URL of a push repository, pushing is turned off if empty.

Push branch

Branch for pushing changes, leave empty to use repository branch

Repository browser

https://github.com/WeblateOrg/demo/blob/{{branch}}/{{filename}}#L{{line}}

Link to repository browser, use {{branch}} for branch, {{filename}} and {{line}} as filename and line placeholders. You might want to strip leading directory by using {{filename|parentdir}}.

File format

gettext PO file

File mask

app/src/main/res/values-*/strings.xmlweblate/langdata/locale/*/LC_MESSAGES/django.po

Path of files to translate relative to repository root, use * instead of language code, for example: po/*.po or locale/*/LC_MESSAGES/django.po.

Monolingual base language file

app/src/main/res/values/strings.xml

Filename of translation base file, containing all strings and their source; it is recommended for monolingual translation formats.

☑ **Edit base file**

Whether users will be able to edit the base file for monolingual translations.

Intermediate language file

Filename of intermediate translation file. In most cases this is a translation file provided by developers and is used when creating actual source strings.

Template for new translations

weblate/langdata/locale/django.pot

Filename of file used for creating new translations. For gettext choose .pot file.

Translation license

GNU General Public License v3.0 or later

Adding new translation

Create new language file

How to handle requests for creating new translations.

Language code style

Default based on the file format

Customize language code used to generate the filename for translations created by Weblate.

Language filter

^(cs|he|hu)$

Regular expression used to filter translation files when scanning for file mask.

Source language

English

Language used for source strings in all components

☐ **Use as a glossary**

You will be able to edit more options in the component settings after creating it.

Save

Powered by Weblate 4.14 About Weblate Legal Contact Documentation Donate to Weblate

Siehe auch:

The Django admin interface, *Project configuration*, *Component configuration*

2.7.3 Project configuration

Create a translation project and then add a new component for translation in it. The project is like a shelf, in which real translations are stacked. All components in the same project share suggestions and their dictionary; the translations are also automatically propagated through all components in a single project (unless turned off in the component configuration), see *Übersetzungsspeicher*.

Siehe auch:

/devel/integration

These basic attributes set up and inform translators of a project:

Projektname

Verbose project name, used to display the project name.

URL-Kurzbegriff

Project name suitable for URLs.

Projektseite

URL where translators can find more info about the project.

This is a required parameter unless turned off by `WEBSITE_REQUIRED`.

Übersetzungsanweisungen

Text describing localization process in the project, and any other information useful for translators. Markdown can be used for text formatting or inserting links.

Kopfzeile „Language-Team“ setzen

Whether Weblate should manage the `Language-Team` header (this is a *GNU gettext* only feature right now).

Gemeinsamen Übersetzungsspeicher verwenden

Whether to use shared translation memory, see *Geteilter Übersetzungspeicher* for more details.

The default value can be changed by `DEFAULT_SHARED_TM`.

Zu einem gemeinsamen Übersetzungsspeicher beitragen

Whether to contribute to shared translation memory, see *Geteilter Übersetzungspeicher* for more details.

The default value can be changed by `DEFAULT_SHARED_TM`.

Zugriffssteuerung

Configure per project access control, see *Projekt-Zugriffssteuerung* for more details.

Der Standardwert kann durch `DEFAULT_ACCESS_CONTROL` geändert werden.

Begutachtung aktivieren

Enable review workflow for translations, see *Zugehörige Prüfer*.

Quellenüberprüfung aktivieren

Enable review workflow for source strings, see *Überprüfungen der Ausgangszeichenketten*.

Siehe auch:

report-source, *Kommentare*

Hooks aktivieren

Whether unauthenticated *Benachrichtigungs-Hooks* are to be used for this repository.

Siehe auch:

Zwischensprachedatei, *Qualitäts-Gateway für die Ausgangszeichenketten*, *Bilingual and monolingual formats*, *Language definitions*

Sprachaliasnamen

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`

Hinweis: 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.

Siehe auch:

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.

Siehe auch:

/devel/integration

You can find some examples of typical configurations in the *Supported file formats*.

Bemerkung: It is recommended to keep translation components to a reasonable size - split the translation by anything that makes sense in your case (individual apps or add-ons, book chapters or websites).

Weblate easily handles translations with 10000s of strings, but it is harder to split work and coordinate among translators with such large translation components.

Should the language definition for a translation be missing, an empty definition is created and named as „cs_CZ (generated)“. You should adjust the definition and report this back to the Weblate authors, so that the missing languages can be included in next release.

The component contains all important parameters for working with the VCS, and for getting translations out of it:

Name der Komponente

Verbose component name, used to display the component name.

Component slug

Component name suitable for URLs.

Component project

Project configuration where the component belongs.

Versionsverwaltung

VCS to use, see *Integration der Versionsverwaltung* for details.

Siehe auch:

Pushing changes from Weblate

Quellcode-Repository

VCS repository used to pull changes.

Siehe auch:

See *Accessing repositories* for more details on specifying URLs.

Hinweis: This can either be a real VCS URL or `weblate://project/component` indicating that the repository should be shared with another component. See *Weblate internal URLs* for more details.

Push-URL für Repository

Repository URL used for pushing. This setting is used only for *Git* and *Mercurial* and push support is turned off for these when this is empty.

For linked repositories, this is not used and setting from linked component applies.

Siehe auch:

See *Accessing repositories* for more details on how to specify a repository URL and *Pushing changes from Weblate* for more details on pushing changes from Weblate.

Paketquellnavigator

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}}`

Falls Ihre Pfade relativ zu verschiedenen Verzeichnissen sind (Pfad enthält `..`), sollten Sie das führende Verzeichnis durch den Filter `parentdir` entfernen (siehe *Template markup*): `https://github.com/WeblateOrg/hello/blob/{{branch}}/{{filename|parentdir}}#L{{line}}`

Exportierte Paketquelladresse

URL where changes made by Weblate are exported. This is important when *Kontinuierliche Lokalisierung* is not used, or when there is a need to manually merge changes. You can use *Git exporter* to automate this for Git repositories.

Repository-Zweig

Which branch to checkout from the VCS, and where to look for translations.

For linked repositories, this is not used and setting from linked component applies.

Push Branch

Branch for pushing changes, leave empty to use *Repository-Zweig*.

For linked repositories, this is not used and setting from linked component applies.

Bemerkung: This is currently only supported for Git, GitLab and GitHub, it is ignored for other VCS integrations.

Siehe auch:

Pushing changes from Weblate

Dateimaske

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 `[]]`.

Siehe auch:

Bilingual and monolingual formats, *What does mean „There are more files for the single language (en)“?*

Einsprachige Basis-Sprachdatei

Base file containing string definitions for *Einsprachige Komponenten*.

Siehe auch:

Bilingual and monolingual formats, *What does mean „There are more files for the single language (en)“?*

Basisdatei bearbeiten

Whether to allow editing the base file for *Einsprachige Komponenten*.

Zwischensprachedatei

Intermediate language file for *Einsprachige Komponenten*. 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 *Einsprachige Basis-Sprachdatei*. In case the string is not translated into the source language, translating to other languages is prohibited. This provides *Qualitäts-Gateway für die Ausgangszeichenketten*.

Siehe auch:

Qualitäts-Gateway für die Ausgangszeichenketten, *Bilingual and monolingual formats*, *What does mean „There are more files for the single language (en)“?*

Vorlage für neue Übersetzungen

Base file used to generate new translations, e.g. `.pot` file with gettext.

Hinweis: In many monolingual formats Weblate starts with empty file by default. Use this in case you want to have all strings present with empty value when creating new translation.

Siehe auch:

adding-translation, *Adding new translations*, *Neue Übersetzung hinzufügen*, *Bilingual and monolingual formats*, *What does mean „There are more files for the single language (en)“?*

Dateiformat

Translation file format, see also *Supported file formats*.

Adresse für Fehlerberichte bei Ausgangszeichenketten

Email address used for reporting upstream bugs. This address will also receive notification about any source string comments made in Weblate.

Verbreitung von Übersetzungen erlauben

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`.

Siehe auch:

Keeping translations same across components

Vorschläge aktivieren

Whether translation suggestions are accepted for this component.

Abstimmen über Vorschläge

Turns on vote casting for suggestions, see *Abstimmen über Vorschläge*.

Vorschläge automatisch annehmen

Automatically accept voted suggestions, see *Abstimmen über Vorschläge*.

Übersetzungsmarkierungen

Customization of quality checks and other Weblate behavior, see *Customizing behavior using flags*.

Erzwungene Qualitätsprüfungen

List of checks which can not be ignored, see *Enforcing checks*.

Bemerkung: Enforcing the check does not automatically enable it, you still should enabled it using *Customizing behavior using flags* in *Übersetzungsmarkierungen* or *Additional info on source strings*.

Lizenzierung der Übersetzung

License of the translation (does not need to be the same as the source code license).

Mitwirkungsvereinbarung

Die Benutzervereinbarung muss genehmigt werden, bevor der Benutzer die Komponenten übersetzen kann.

Neue Übersetzung hinzufügen

How to handle requests for creation of new languages. Available options:

Maintainer kontaktieren
Der Benutzer kann die gewünschte Sprache auswählen und die Projektbetreuer erhalten eine Benachrichtigung darüber. Es liegt an ihnen, ob Sie die Sprache zum Repository hinzufügen.

Geben Sie die URL für die Übersetzungsanweisungen an
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).

Neue Sprachdatei erstellen
User can select language and Weblate automatically creates the file for it and translation can begin.

Hinzufügen neuer Übersetzungen deaktivieren
There will be no option for user to start new translation.

Hinweis: The project admins can add new translations even if it is disabled here when it is possible (either *Vorlage für neue Übersetzungen* or the file format supports starting from an empty file).

Siehe auch:

adding-translation, *Adding new translations*

Zeichenketten verwalten

Neu in Version 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.

Siehe auch:

Bilingual and monolingual formats, adding-new-strings, `POST /api/translations/(string:project)/(string:component)/(string:language)/units/`

Stil des Sprachcodes

Anpassen des Sprachcodes, der zur Generierung des Dateinamens für von Weblate erstellte Übersetzungen verwendet wird.

Siehe auch:

Adding new translations, *Sprachkürzel*, *Parsing language codes*

Git-Strategie

You can configure how updates from the upstream repository are handled. The actual implementation depends on VCS, see *Integration der Versionsverwaltung*.

Umbasieren (rebase)
Rebases Weblate commits on top of upstream repository on update. This provides clean history without extra merge commits.

Rebasing can cause you trouble in case of complicated merges, so carefully consider whether or not you want to enable them.

You might need to enable force pushing by choosing *Git Push erzwingen* as *Versionsverwaltung*, especially when pushing to a different branch.

Zusammenführen
Upstream repository changes are merged into Weblate one. This setting utilizes fast-forward when possible. This is the safest way, but might produce a lot of merge commits.

Zusammenführen ohne Fast-Forward
Upstream repository changes are merged into Weblate one with doing a merge commit every time (even when fast-forward would be possible). Every Weblate change will appear as a merge commit in Weblate repository.

Default value can be changed by `DEFAULT_MERGE_STYLE`.

Commit, add, delete, merge, add-on, and merge request messages

Message used when committing a translation, see *Template markup*.

Default value can be changed by `DEFAULT_ADD_MESSAGE`, `DEFAULT_ADDON_MESSAGE`, `DEFAULT_COMMIT_MESSAGE`, `DEFAULT_DELETE_MESSAGE`, `DEFAULT_MERGE_MESSAGE`, `DEFAULT_PULL_MESSAGE`.

Bei Commit gleichzeitig Pushen

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.

Alter der Änderungen, das erreicht sein muss, bevor ein Commit erfolgt

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`.

Hinweis: There are other situations where pending changes might be committed, see *Lazy commits*.

Sperre bei Fehler

Locks the component (and linked components, see *Weblate internal URLs*) upon the first failed push or merge into its upstream repository, or pull from it. This avoids adding another conflicts, which would have to be resolved manually.

The component will be automatically unlocked once there are no repository errors left.

Ausgangssprache

Language used for source strings. Change this if you are translating from something else than English.

Hinweis: 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 *Zwischensprachedatei*.

Sprachen-Filter

Regulärer Ausdruck, der zum Filtern der Übersetzung bei der Suche nach Dateimaske verwendet wird. Er kann verwendet werden, um die Liste der von Weblate verwalteten Sprachen einzuschränken.

Bemerkung: You need to list language codes as they appear in the filename.

Some examples of filtering:

Filter description	Regulärer Ausdruck
Selected languages only	`^(cs|de|es)$`
Exclude languages	`^(?!(it|fr)$).+$`
Filter two letter codes only	`^..$`
Exclude non language files	`^(?!(blank)$).+$`
Include all files (default)	`^[^.]+$`

Varianten regulärer Ausdruck

Regular expression used to determine the variants of a string, see variants.

Bemerkung: Die meisten Felder können von Projektverantwortlichen oder Administratoren in der Weblate-Benutzeroberfläche bearbeitet werden.

Siehe auch:

Does Weblate support other VCSes than Git and Mercurial?, alerts

Priorität

Komponenten mit höherer Priorität werden Übersetzern zuerst angeboten.

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`.

Hinweis: This applies to project admins as well — please make sure you will not loose access to the component after toggling the status.

In Projekten teilen

You can choose additional projects where the component will be visible. Useful for shared libraries which you use in several projects.

Bemerkung: 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.

Verwendung als Glossar

Neu in Version 4.5.

Ermöglicht die Verwendung dieser Komponente als Glossar. Sie können mit *Glossarfarbe* konfigurieren, wie es aufgelistet werden soll.

Das Glossar ist in allen durch *In Projekten teilen* definierten Projekten zugänglich.

It is recommended to enable *Zeichenketten verwalten* on glossaries in order to allow adding new words to them.

Siehe auch:

Glossar

Glossarfarbe

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*
- **Mehrere Erweiterungen**
  - *Komponentenerkennung*
  - *Statistikgenerator*
  - *Ausführen von Skripten der Erweiterung*

There following variables are available in the component templates:

`{{ language_code }}`
Sprachkürzel

`{{ language_name }}`
Sprachenname

`{{ component_name }}`
Name der Komponente

`{{ component_slug }}`
Component slug

`{{ project_name }}`
Projektname

`{{ project_slug }}`
Project slug

`{{ url }}`
Translation URL

`{{ filename }}`
Übersetzungsdateiname

`{{ stats }}`
Translation stats, this has further attributes, examples below.

`{{ stats.all }}`
Total strings count

`{{ stats.fuzzy }}`
Count of strings needing review

`{{ stats.fuzzy_percent }}`
Percent of strings needing review

`{{ stats.translated }}`
Translated strings count

`{{ stats.translated_percent }}`
Translated strings percent

`{{ stats.allchecks }}`
Number of strings with failing checks

`{{ stats.allchecks_percent }}`
Percent of strings with failing checks

`{{ author }}`
Author of current commit, available only in the commit scope.

{{ addon_name }}
Name of currently executed add-on, available only in the add-on commit message.

The following variables are available in the repository browser or editor templates:

{{branch}}
current branch

{{line}}
line in file

{{filename}}
filename, you can also strip leading parts using the `parentdir` filter, for example `{{filename|parentdir}}`

You can combine them with filters:

```
{{ component|title }}
```

You can use conditions:

```
{% if stats.translated_percent > 80 %}Well translated!{% endif %}
```

There is additional tag available for replacing characters:

```
{% replace component "-" " " %}
```

You can combine it with filters:

```
{% replace component|capfirst "-" " " %}
```

There are also additional filter to manipulate with filenames:

```
Directory of a file: {{ filename|dirname }}
File without extension: {{ filename|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*)
- *Enable caching*
- *Use a powerful database engine*
- *Disable debug mode*

Check resource limits

If you are importing huge translations or repositories, you might be hit by resource limitations of your server.

- Check the amount of free memory, having translation files cached by the operating system will greatly improve performance.
- Disk operations might be bottleneck if there is a lot of strings to process—the disk is pushed by both Weblate and the database.
- Additional CPU cores might help improve performance of background tasks (see *Background tasks using Celery*).

Disable unneeded checks

Some quality checks can be quite expensive, and if not needed, can save you some time during import if omitted. See `CHECK_LIST` for info on configuration.

2.7.7 Automatic creation of components

In case your project has dozen of translation files (e.g. for different gettext domains, or parts of Android apps), you might want to import them automatically. This can either be achieved from the command-line by using `import_project` or `import_json`, or by installing the *Komponentenerkennung* add-on.

To use the add-on, you first need to create a component for one translation file (choose the one that is the least likely to be renamed or removed in future), and install the add-on on this component.

For the management commands, you need to create a project which will contain all components and then run `import_project` or `import_json`.

Siehe auch:

Management commands, *Komponentenerkennung*

2.8 Language definitions

Um verschiedene Übersetzungen korrekt darzustellen, werden Informationen über den Namen der Sprache, die Leserichtung, die Pluraldefinitionen und den Sprachcode benötigt.

2.8.1 Integrierte Sprachdefinitionen

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`.

Siehe auch:

Erweiterung der integrierten Sprachdefinitionen, Current language definitions

2.8.2 Parsing language codes

While parsing translations, Weblate attempts to map language code (usually the ISO 639-1 one) from the *Dateimaske* to any existing language object.

You can further adjust this mapping at project level by *Sprachaliasnamen*.

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.
- Suche nach integrierten Sprachaliasen.
- Nach Sprachnamen suchen.
- 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 *Contributing to Weblate*), so that the proper definition can be added to the upcoming Weblate release.

Hinweis: In case you see something unwanted as a language, you might want to adjust *Sprachen-Filter* to ignore such file when parsing translations.

Siehe auch:

Sprachkürzel, *Adding new translations*

2.8.3 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.

2.8.4 Mehrdeutige Sprachcodes und Makrosprachen

In many cases it is not a good idea to use macrolanguage code for a translation. The typical problematic case might be Kurdish language, which might be written in Arabic or Latin script, depending on actual variant. To get correct behavior in Weblate, it is recommended to use individual language codes only and avoid macrolanguages.

Siehe auch:

Macrolanguages definition, List of macrolanguages

2.8.5 Language definitions

Each language consists of following fields:

Sprachkürzel

Code zur Identifizierung der Sprache. Weblate bevorzugt Zwei-Buchstaben-Codes, wie sie in ISO 639-1 definiert sind, verwendet aber ISO 639-2 oder ISO 639-3 Codes für Sprachen, die keinen Zwei-Buchstaben-Code haben. Es kann auch erweiterte Codes unterstützen, wie in BCP 47 definiert.

Siehe auch:

Parsing language codes, *Adding new translations*

Sprachenname

Visible name of the language. The language names included in Weblate are also being localized depending on user interface language.

Leserichtung

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.

Pluralformel

Gettext compatible plural formula used to determine which plural form is used for given count.

Siehe auch:

Plural, GNU gettext utilities: Plural forms, Language Plural Rules by the Unicode Consortium

2.8.6 Adding new translations

Geändert in Version 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 *Android string resources*), while others expect to have all keys present (for example *GNU gettext*). The document-based formats (for example *OpenDocument Format*) start with a copy of the source document and all strings marked as needing editing. In some situations this really doesn't depend on the format, but rather on the framework you use to handle the translation (for example with *JSON files*).

When you specify *Vorlage für neue Übersetzungen* in *Component configuration*, Weblate will use this file to start new translations. Any exiting translations will be removed from the file when doing so.

When *Vorlage für neue Übersetzungen* 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 des Sprachcodes* allows you to customize language code used in generated filenames:

Standard basierend auf dem Dateiformat
Dependent on file format, for most of them POSIX is used.

POSIX-Stil mit Unterstrich als Trennzeichen
Typically used by gettext and related tools, produces language codes like `pt_BR`.

POSIX-Stil mit Unterstrich als Trennzeichen, einschließlich Ländercode
POSIX style language code including the country code even when not necessary (for example `cs_CZ`).

BCP-Stil mit Bindestrich als Trennzeichen
Typically used on web platforms, produces language codes like `pt-BR`.

BCP-Stil mit Bindestrich als Trennzeichen, einschließlich Ländercode
BCP style language code including the country code even when not necessary (for example `cs-CZ`).

BCP-Stil mit Bindestrich als Trennzeichen, Codes für ältere Sprachen
Uses legacy codes for Chinese and BCP style notation.

BCP-Stil mit Bindestrich als Trennzeichen, Kleinbuchstaben
BCP style notation, all in lower case (for examle `cs-cz`).

Stil der App-Store-Metadaten
Style suitable for uploading metadata to appstores. This should be suitable for both Apple App Store and Google Play Store.

Android-Stil
Only used in Android apps, produces language codes like `pt-rBR`.

Linux-Stil
Locales as used by Linux, uses legacy codes for Chinese and POSIX style notation.

Additionally, any mappings defined in *Sprachaliasnamen* are applied in reverse.

Bemerkung: Weblate recognizes any of these when parsing translation files, the above settings only influences how new files are created.

Siehe auch:

Sprachkürzel, *Parsing language codes*

2.9 Kontinuierliche Lokalisierung

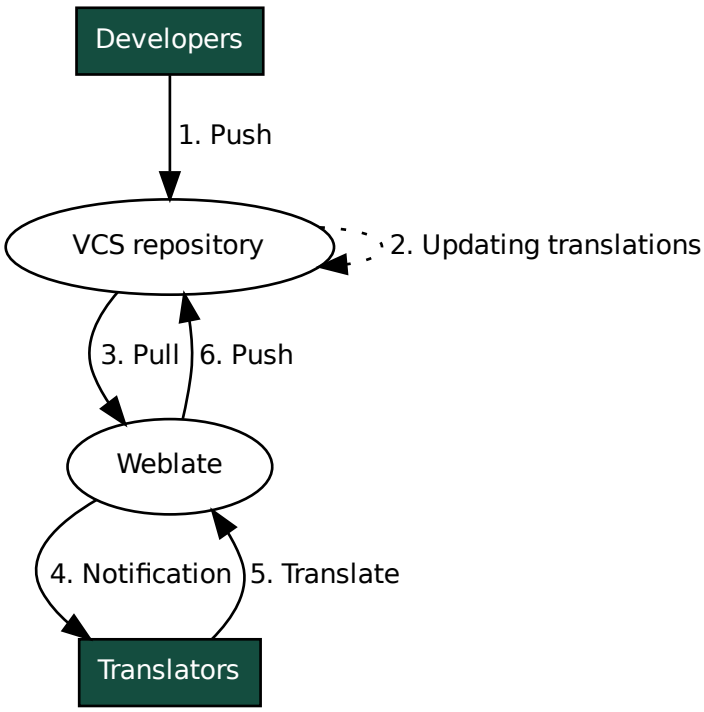
There is infrastructure in place so that your translation closely follows development. This way translators can work on translations the entire time, instead of working through huge amount of new text just prior to release.

Siehe auch:

/devel/integration describes basic ways to integrate your development with Weblate.

This is the process:

1. Developers make changes and push them to the VCS repository.
2. Optionally the translation files are updated (this depends on the file format, see *Why does Weblate still show old translation strings when I've updated the template?*).
3. Weblate pulls changes from the VCS repository, see *Updating repositories*.
4. Once Weblate detects changes in translations, translators are notified based on their subscription settings.
5. Translators submit translations using the Weblate web interface, or upload offline changes.
6. Once the translators are finished, Weblate commits the changes to the local repository (see *Lazy commits*) and pushes them back if it has permissions to do so (see *Pushing changes from Weblate*).



2.9.1 Updating repositories

You should set up some way of updating backend repositories from their source.

- Use *Benachrichtigungs-Hooks* to integrate with most of common code hosting services:
  - *Automatically receiving changes from GitHub*
  - *Automatically receiving changes from GitLab*
  - *Automatically receiving changes from Bitbucket*
  - *Automatically receiving changes from Pagure*
  - *Automatically receiving changes from Azure Repos*
- Manually trigger update either in the repository management or using *Weblate's REST API* or *Weblate Client*
- Enable `AUTO_UPDATE` to automatically update all components on your Weblate instance
- Execute `updategit` (with selection of project or `--all` to update all)

Whenever Weblate updates the repository, the post-update addons will be triggered, see *Erweiterungen*.

Avoiding merge conflicts

Die Merge-Konflikte von Weblate entstehen, wenn dieselbe Datei sowohl in Weblate als auch außerhalb von Weblate geändert wurde. Es gibt zwei Möglichkeiten, damit umzugehen: Entweder Sie vermeiden Bearbeitungen außerhalb von Weblate oder Sie integrieren Weblate in Ihren Aktualisierungsprozess, so dass die Änderungen vor der Aktualisierung der Dateien außerhalb von Weblate geleert werden.

Der erste Ansatz ist bei einsprachigen Dateien einfach - Sie können neue Zeichenketten innerhalb von Weblate hinzufügen und die gesamte Bearbeitung der Dateien dort belassen. Für zweisprachige Dateien gibt es in der Regel eine Art von Nachrichtenextraktionsprozess, um übersetzbare Dateien aus dem Quellcode zu erzeugen. In manchen Fällen kann dies in zwei Teile aufgeteilt werden - einer für die Extraktion erzeugt eine Vorlage (z.B. gettext POT wird mit **`xgettext`** erzeugt) und ein weiterer Prozess fügt sie in die tatsächlichen Übersetzungen ein (die gettext PO-Dateien werden mit **`msgmerge`** aktualisiert). Sie können den zweiten Schritt innerhalb von Weblate durchführen. Weblate sorgt dafür, dass alle anstehenden Änderungen vor diesem Vorgang berücksichtigt werden.

Der zweite Ansatz kann erreicht werden, indem man *Weblate's REST API* verwendet, um Weblate zu zwingen, alle anstehenden Änderungen zu pushen und die Übersetzung zu sperren, während man selbst Änderungen vornimmt.

The script for doing updates can look like this:

```
# Lock Weblate translation
wlc lock
# Push changes from Weblate to upstream repository
wlc push
# Pull changes from upstream repository to your local copy
git pull
# Update translation files, this example is for Django
./manage.py makemessages --keep-pot -a
git commit -m 'Locale updates' -- locale
# Push changes to upstream repository
git push
# Tell Weblate to pull changes (not needed if Weblate follows your repo
# automatically)
wlc pull
# Unlock translations
wlc unlock
```

If you have multiple components sharing same repository, you need to lock them all separately:

```
wlc lock foo/bar
wlc lock foo/baz
wlc lock foo/baj
```

Bemerkung: The example uses *Weblate Client*, which needs configuration (API keys) to be able to control Weblate remotely. You can also achieve this using any HTTP client instead of wlc, e.g. curl, see *Weblate's REST API*.

Siehe auch:

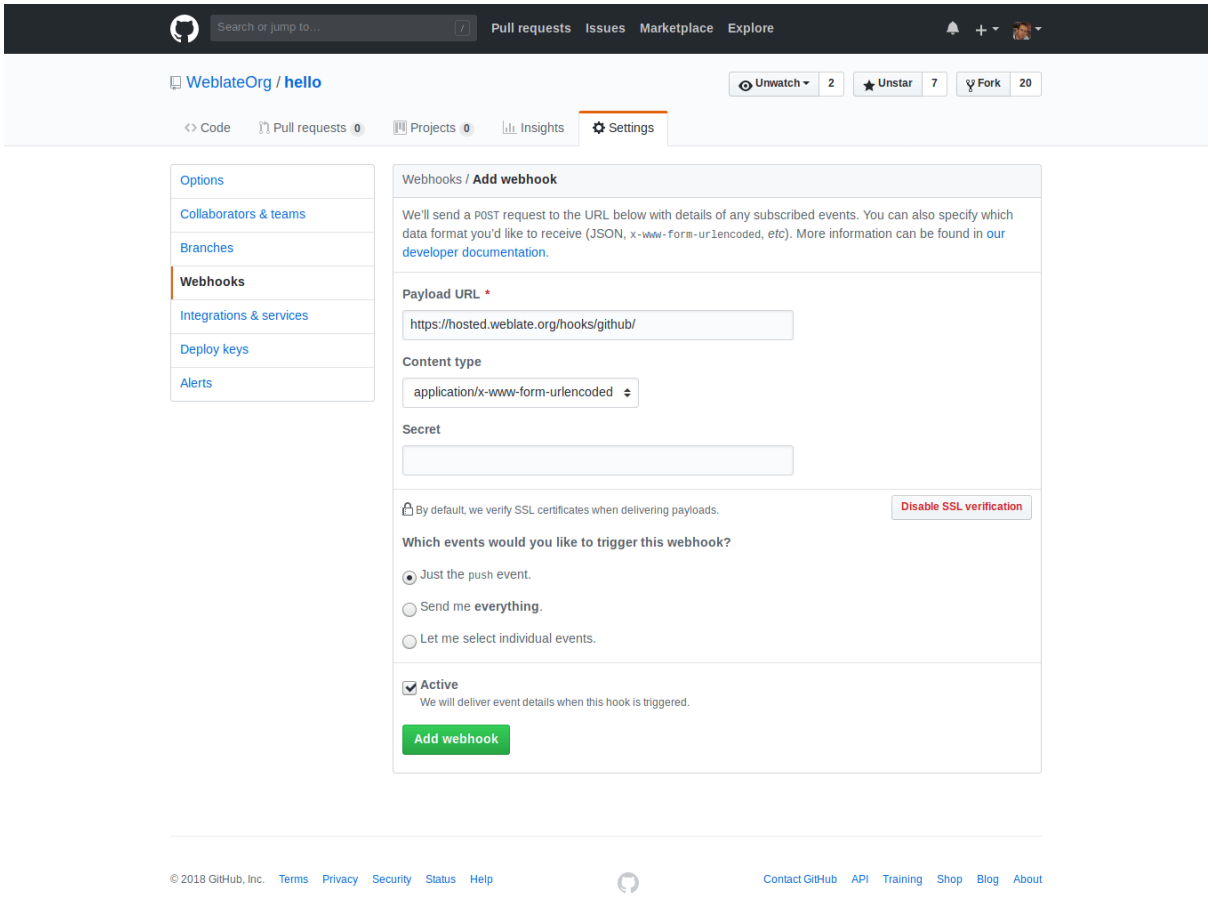
Weblate Client

Automatically receiving changes from GitHub

Weblate comes with native support for GitHub.

Wenn Sie Hosted Weblate verwenden, empfiehlt es sich, die Weblate-App zu installieren, damit Sie die korrekte Einrichtung erhalten, ohne viel einrichten zu müssen. Sie kann auch zum Zurückschieben von Änderungen verwendet werden.

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/`.

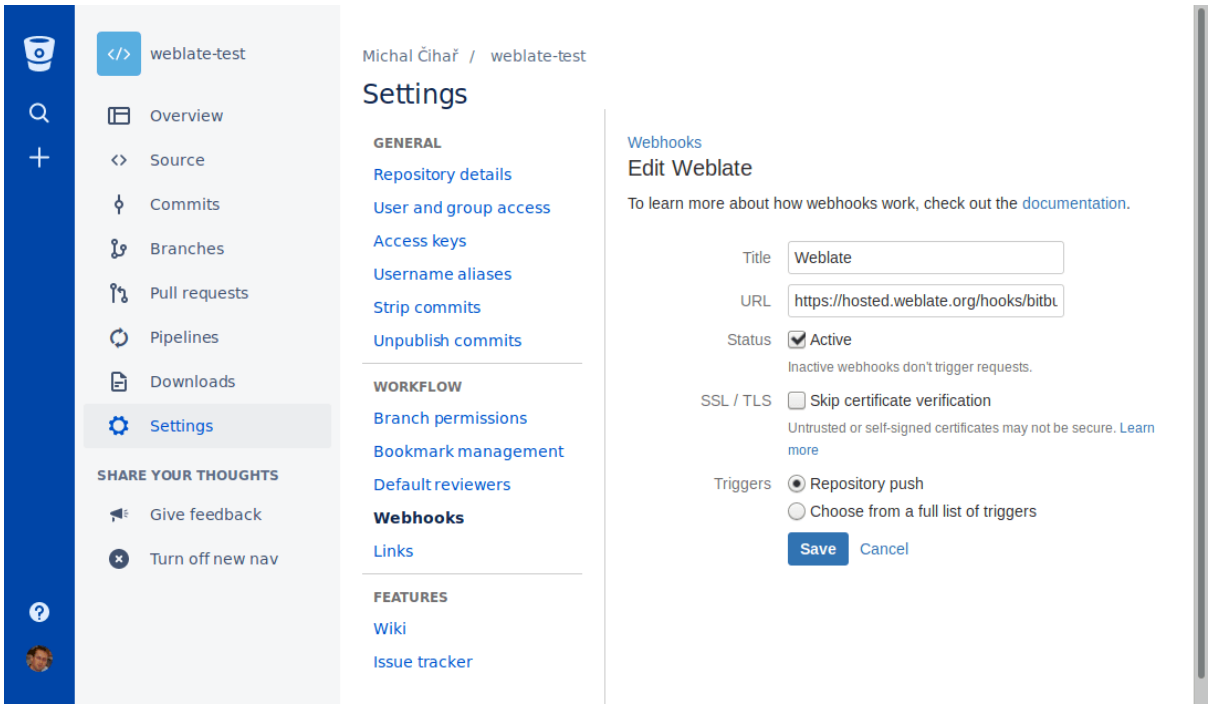
You can leave other values at default settings (Weblate can handle both content types and consumes just the *push* event).

Siehe auch:

`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/`).



Siehe auch:

`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/`).

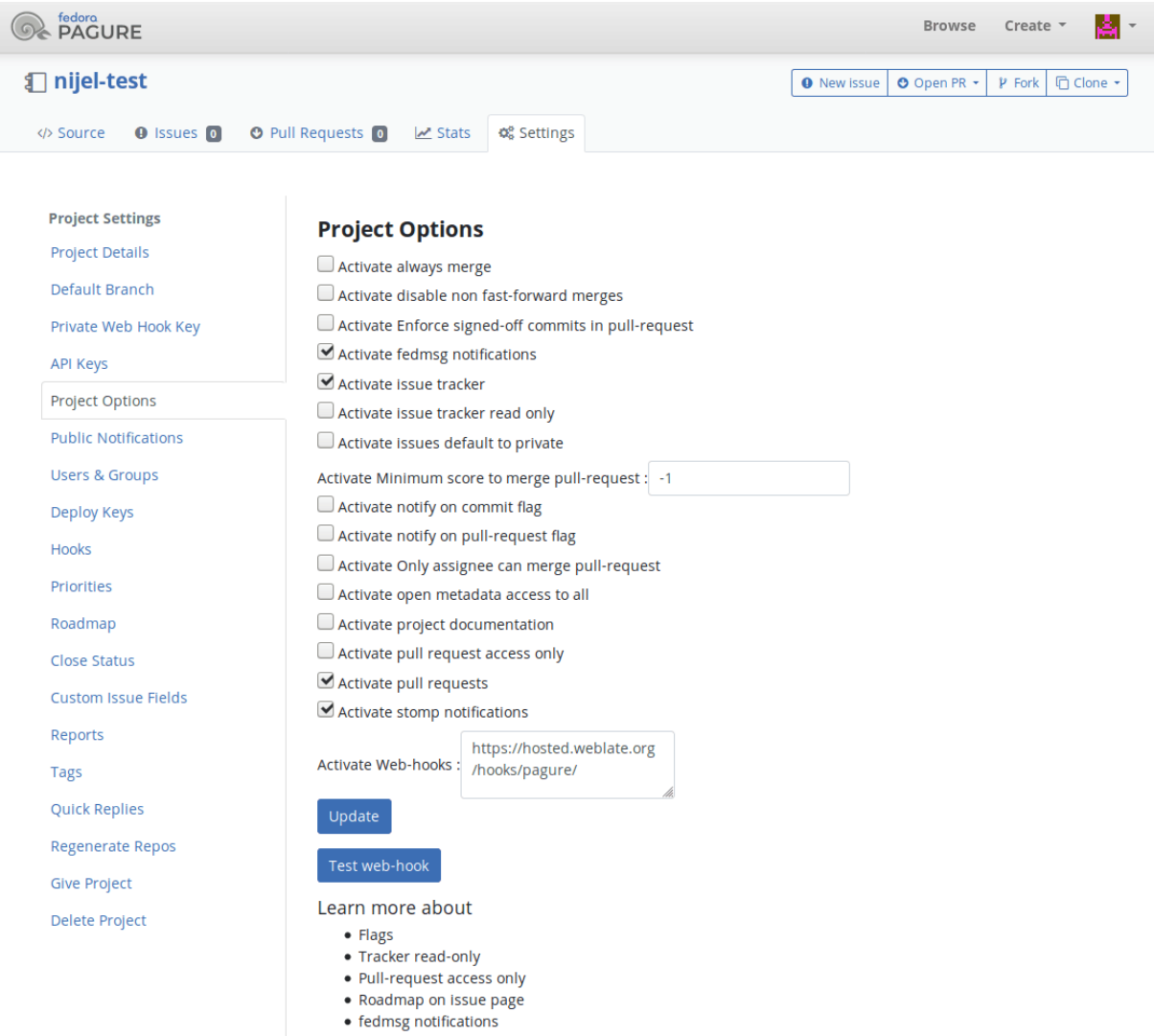
Siehe auch:

`POST /hooks/gitlab/`, *Accessing repositories from Hosted Weblate*

Automatically receiving changes from Pagure

Neu in Version 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*:



Siehe auch:

`POST /hooks/pagure/`, *Accessing repositories from Hosted Weblate*

Automatically receiving changes from Azure Repos

Neu in 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*.

Siehe auch:

Web hooks in Azure DevOps manual, `POST /hooks/azure/`, *Accessing repositories from Hosted Weblate*

Automatically receiving changes from Gitea Repos

Neu in Version 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*.

Siehe auch:

Webhooks in Gitea manual, `POST /hooks/gitea/`, *Accessing repositories from Hosted Weblate*

Automatically receiving changes from Gitee Repos

Neu in Version 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*.

Siehe auch:

Webhooks in Gitee manual, `POST /hooks/gitee/`, *Accessing repositories from Hosted Weblate*

Automatically updating repositories nightly

Weblate automatically fetches remote repositories nightly to improve performance when merging changes later. You can optionally turn this into doing nightly merges as well, by enabling `AUTO_UPDATE`.

2.9.2 Pushing changes from Weblate

Each translation component can have a push URL set up (see *Push-URL für Repository*), 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 *Bei Commit gleichzeitig Pushen*). 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 *Integration der Versionsverwaltung* used, more details are found in that chapter.

In case you do not want direct pushes by Weblate, there is support for *GitHub-Pull-Anfragen*, *GitLab Merge Requests*, *Pagure Merge Requests* pull requests or *Gerrit* reviews, you can activate these by choosing *GitHub*, *GitLab*, *Gerrit* or *Pagure* as *Versionsverwaltung* in *Component configuration*.

Overall, following options are available with Git, GitHub and GitLab:

Desired setup	*Versionsverwaltung*	*Push-URL für Repository*	*Push Branch*
No push	*Git*	*empty*	*empty*
Push directly	*Git*	SSH URL	*empty*
Push zu separatem Branch	*Git*	SSH URL	Branch name
GitHub pull request from fork	*GitHub-Pull-Anfragen*	*empty*	*empty*
GitHub pull request from branch	*GitHub-Pull-Anfragen*	SSH URL[1]	Branch name
GitLab merge request from fork	*GitLab Merge Requests*	*empty*	*empty*
GitLab merge request from branch	*GitLab Merge Requests*	SSH URL[1]	Branch name
Pagure merge request from fork	*Pagure Merge Requests*	*empty*	*empty*
Pagure merge request from branch	*Pagure Merge Requests*	SSH URL[1]	Branch name

[1] Can be empty in case *Quellcode-Repository* supports pushing.

Bemerkung: You can also enable automatic pushing of changes after Weblate commits, this can be done in *Bei Commit gleichzeitig Pushen*.

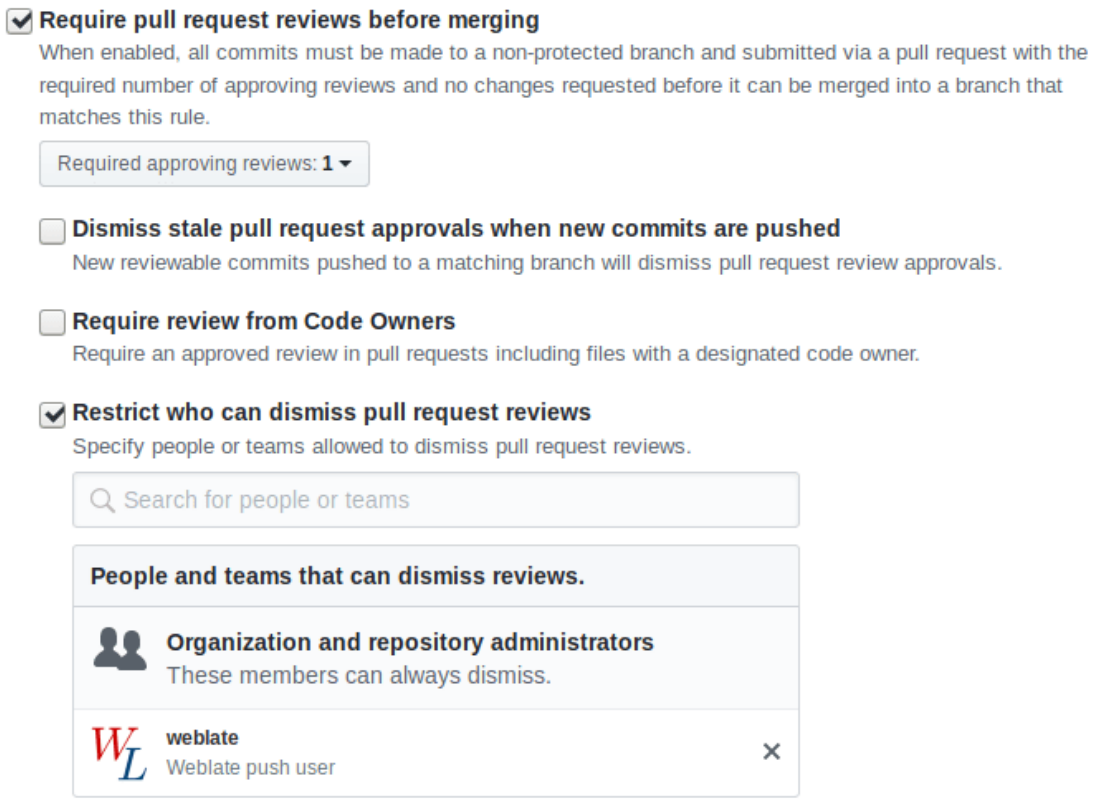
Siehe auch:

See *Accessing repositories* for setting up SSH keys, and *Lazy commits* for info about when Weblate decides to commit changes.

Protected branches

If you are using Weblate on protected branch, you can configure it to use pull requests and perform actual review on the translations (what might be problematic for languages you do not know). An alternative approach is to waive this limitation for the Weblate push user.

For example on GitHub this can be done in the repository configuration:



2.9.3 Interacting with others

Weblate makes it easy to interact with others using its API.

Siehe auch:

Weblate's REST API

2.9.4 Lazy commits

The behaviour of Weblate is to group commits from the same author into one commit if possible. This greatly reduces the number of commits, however you might need to explicitly tell it to do the commits in case you want to get the VCS repository in sync, e.g. for merge (this is by default allowed for the *Managers* group, see *Liste der Berechtigungen und integrierten Rollen*).

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.
- A file download is requested.
- Change is older than period defined as *Alter der Änderungen, das erreicht sein muss, bevor ein Commit erfolgt* on *Component configuration*.

Hinweis: Commits are created for every component. So in case you have many components you will still see lot of commits. You might utilize *Git-Commits konsolidieren* add-on in that case.

If you want to commit changes more frequently and without checking of age, you can schedule a regular task to perform a commit:

```
CELERY_BEAT_SCHEDULE = {
    # Unconditionally commit all changes every 2 minutes
    "commit": {
        "task": "weblate.trans.tasks.commit_pending",
        # Omitting hours will honor per component settings,
        # otherwise components with no changes older than this
        # won't be committed
        "kwargs": {"hours": 0},
        # How frequently to execute the job in seconds
        "schedule": 120,
    }
}
```

2.9.5 Processing repository with scripts

The way to customize how Weblate interacts with the repository is *Erweiterungen*. Consult *Ausführen von Skripten der Erweiterung* for info on how to execute external scripts through add-ons.

2.9.6 Keeping translations same across components

Wenn Sie mehrere Übersetzungskomponenten haben, möchten Sie möglicherweise sicherstellen, dass die selben Zeichenketten dieselbe Übersetzung haben. Dies kann auf mehreren Ebenen erreicht werden.

Translation propagation

Wenn *Verbreitung von Übersetzungen erlauben* aktiviert ist (was die Voreinstellung ist, siehe *Component configuration*), werden alle neuen Übersetzungen automatisch in allen Komponenten mit übereinstimmenden Zeichenketten durchgeführt. Solche Übersetzungen werden dem aktuell übersetzenden Benutzer in allen Komponenten korrekt gutgeschrieben.

Bemerkung: 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 *Inkonsistent* check fires whenever the strings are different. You can utilize this to review such differences manually and choose the right translation.

Automatische Übersetzung

Die automatische Übersetzung basierend auf verschiedenen Komponenten kann eine Möglichkeit sein, die Übersetzungen zwischen den Komponenten zu synchronisieren. Sie können es entweder manuell auslösen (siehe *Automatische Übersetzung*) oder es automatisch bei der Aktualisierung des Repositorys mit Hilfe einer Erweiterung laufen lassen (siehe *Automatische Übersetzung*).

2.10 Licensing translations

You can specify which license translations are contributed under. This is especially important to do if translations are open to the public, to stipulate what they can be used for.

You should specify *Component configuration* license info. You should avoid requiring a contributor license agreement, though it is possible.

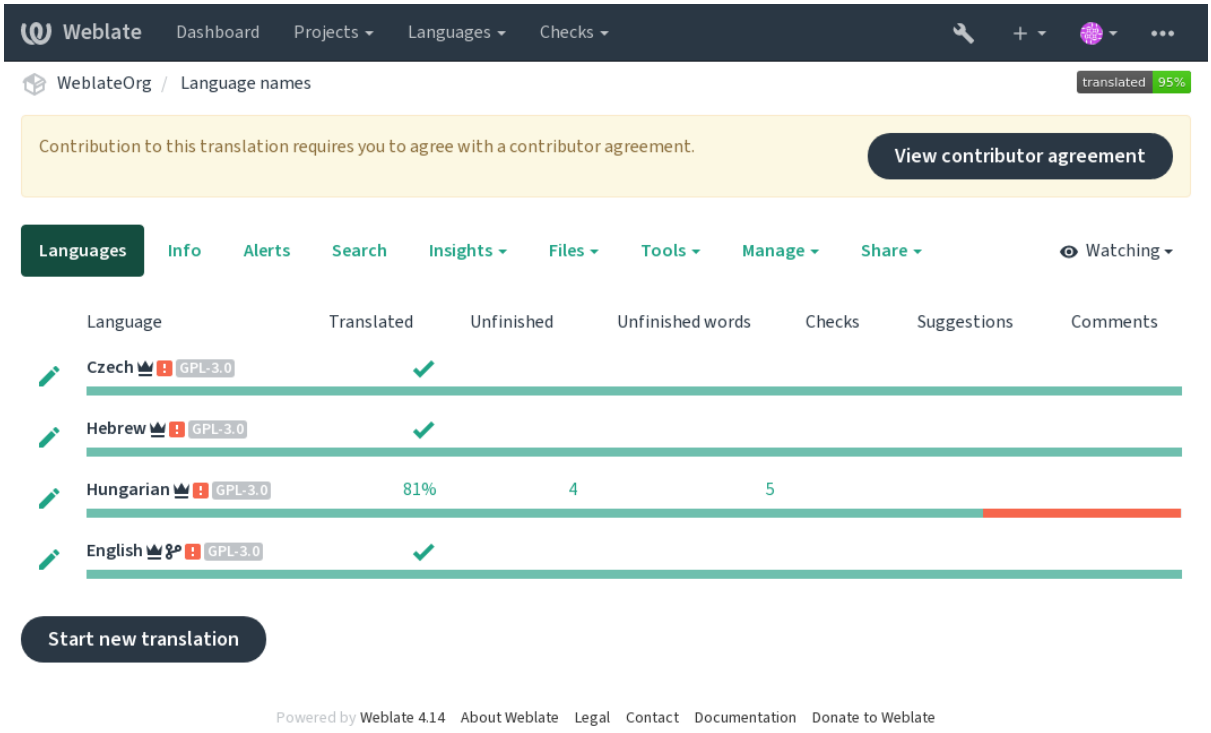
2.10.1 License info

Upon specifying license info (license name and URL), this info is shown in the translation info section of the respective *Component configuration*.

Usually this is best place to post licensing info if no explicit consent is required. If your project or translation is not libre you most probably need prior consent.

2.10.2 Mitwirkungsvereinbarung

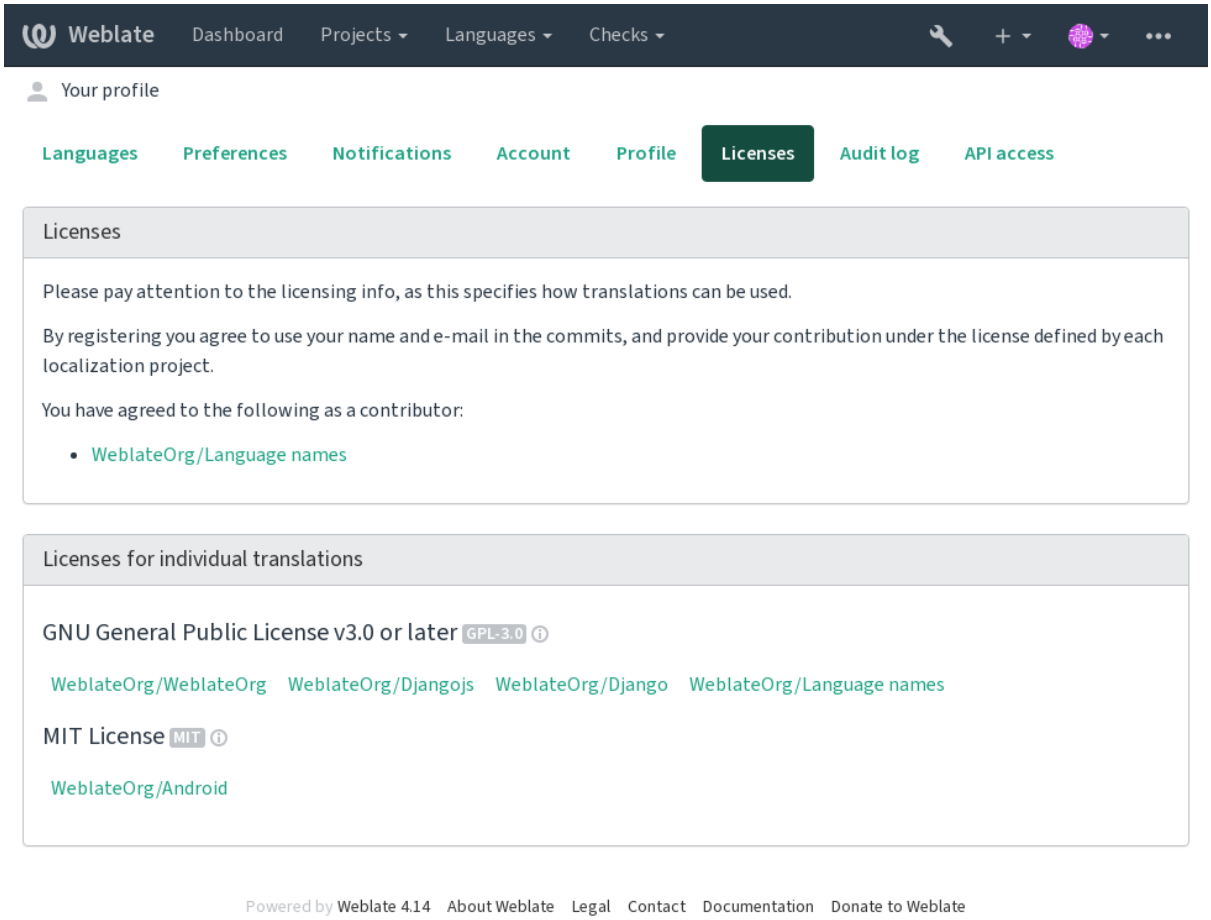
If you specify a contributor license agreement, only users who have agreed to it will be able to contribute. This is a clearly visible step when accessing the translation:



The entered text is formatted into paragraphs and external links can be included. HTML markup can not be used.

2.10.3 User licenses

Any user can review all translation licenses of all public projects on the instance from their profile:



2.11 Übersetzungsrichtlinien

2.11.1 Abstimmen über Vorschläge

Jeder kann standardmäßig Vorschläge hinzufügen, die von angemeldeten Benutzern akzeptiert werden. Die Vorschlagsabstimmung kann verwendet werden, um einen String zu verwenden, wenn mehr als ein angemeldeter Benutzer zustimmt, indem die Komponente mit *Vorschlagsabstimmung* eingerichtet wird, um die Abstimmung einzuschalten, und *Vorschläge automatisch annehmen*, um einen Schwellenwert für akzeptierte Vorschläge festzulegen (dies schließt eine Stimme des vorschlagenden Benutzers ein, wenn sie abgegeben wird).

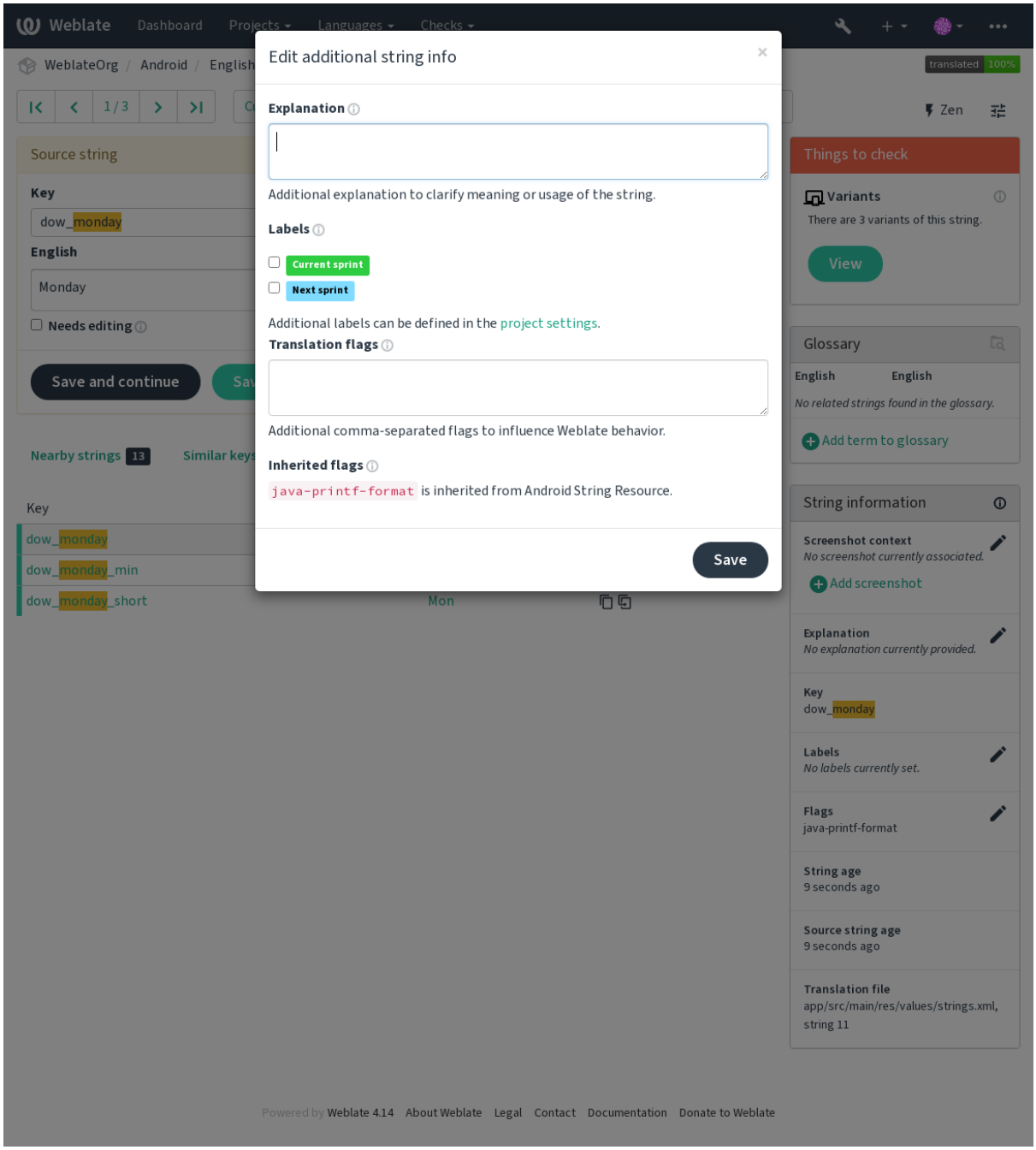
Bemerkung: Once automatic acceptance is set up, normal users lose the privilege to directly save translations or accept suggestions. This can be overridden with the *Edit string when suggestions are enforced permission*.

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:



Sie können direkt von der Übersetzungsoberfläche aus darauf zugreifen, indem Sie auf das Symbol „Bearbeiten“ neben *Screenshot context* oder *Flags* klicken.

Weblate Dashboard Projects Languages Checks

WeblateOrg / Django / Czech / Translate translated 96%

11 / 26 All strings Position and priority Zen

Translation

Explanation

Help text for automatic translation tool

English

Automatic translation via machine translation uses active machine translation engines to get the best possible translations and applies them in this project.

Czech

Automatický překlad prostřednictvím strojového překladu používá aktivní enginy strojového překladu pro získání nejlepších možných překladů a použije je na tento projekt.

Needs editing 169/1570 · 157

Save and continue | Save and stay | Suggest | Skip

Nearby strings 26 | Comments | Automatic suggestions | Other languages 4 | History

Context	English	Czech	Actions
	Files	Soubory	
	Automatic translation	Automatický překlad	
	Add new translation string	Add new translation string	
	Translation status	Stav překladu	
	%(count)s word	%(count)s slovo	
	Other components	Další součásti	
	Translation file	Soubor s překladem	
	Download	Stáhnout	
	Browse all translation changes	Procházet všechny změny v překladu.	
	Automatic translation takes existing translations in this project and applies them to the current component. It can be used to push translations to a different branch, to fix inconsistent translations or to translate a new component using translation memory.	Automatický překlad použije stávající překlady v projektu na tuto součást. Může být užitečný pro sloučení překladů z jiné větve, opravu nekonzistentních překladů nebo překlad nové součásti pomocí překladové paměti.	
	Automatic translation via machine translation uses active machine translation engines to get the best possible translations and applies them in this project.	Automatický překlad prostřednictvím strojového překladu používá aktivní enginy strojového překladu pro získání nejlepších možných překladů a použije je na tento projekt.	
	You can add new translation string here, it will automatically appear in all translations.	Zde můžete přidat nový řetězec k překladu, automaticky se objeví ve všech jazycích.	
	The uploaded file will be merged with the current translation. In case you want to overwrite already translated strings, don't forget to enable it.	Nahraný soubor bude sloučen se stávajícími překlady. Pokud chcete přepsat již přeložené řetězce, nezapomeňte to povolit.	
	The uploaded file will be merged with the current translation.	Nahraný soubor bude sloučen se stávajícími překlady.	
	The fulltext search might not work properly as the fulltext index for this translation is not yet up to date.	Fulltextové vyhledávání nemusí fungovat správě, protože fulltextový index pro tento překlad ještě není plně zpracován.	
	Review	Kontrola	
	Review translations touched by other users.	Zkontrolovat překlady od ostatních uživatelů.	
	Start review	Začít kontrolu	
	Percent	Procenta	
	Total	Celkem	
	Failing check	Neúspěšných kontrol	
	Last activity	Poslední aktivita	
	Last change	Poslední změna	
	Last author	Poslední autor	
Question for a mathematics-based CAPTCHA, the %s is an arithmetic problem	What is %s?	Kolik to je?	
	The string uses three dots (...) instead of an ellipsis character (…)		

Glossary

English	Czech	
machine translation	strojový překlad	WeblateOrg
project	projekt	WeblateOrg

Add term to glossary

String information

Screenshot context
No screenshot currently associated.
Add screenshot

Explanation
Help text for automatic translation tool

Labels
No labels currently set.

Flags
No flags currently set.

Source string location
weblate/templates/translation.html:212

String age
a second ago

Source string age
2 seconds ago

Translation file
weblate/locale/cs/LC_MESSAGES/djangojs.po, string 11

Powered by Weblate 4.14 About Weblate Legal Contact Documentation Donate to Weblate

Strings prioritization

Neu in Version 2.0.

String priority can be changed to offer higher priority strings for translation earlier by using the `priority` flag.

Hinweis: This can be used to order the flow of translation in a logical manner.

Siehe auch:

Qualitätsprüfungen

Übersetzungsmarkierungen

Neu in Version 2.4.

Geändert in Version 3.3: Previously called *Quality checks flags*, it no longer configures only checks.

Customization of quality checks and other Weblate behavior, see *Customizing behavior using flags*.

The string flags are also inherited from the *Übersetzungsmarkierungen* at *Component configuration* and flags from the translation file (see *Supported file formats*).

Siehe auch:

Qualitätsprüfungen, *Customizing behavior using flags*

Erläuterung

Geändert in Version 4.1: In previous versions this has been called *Extra context*.

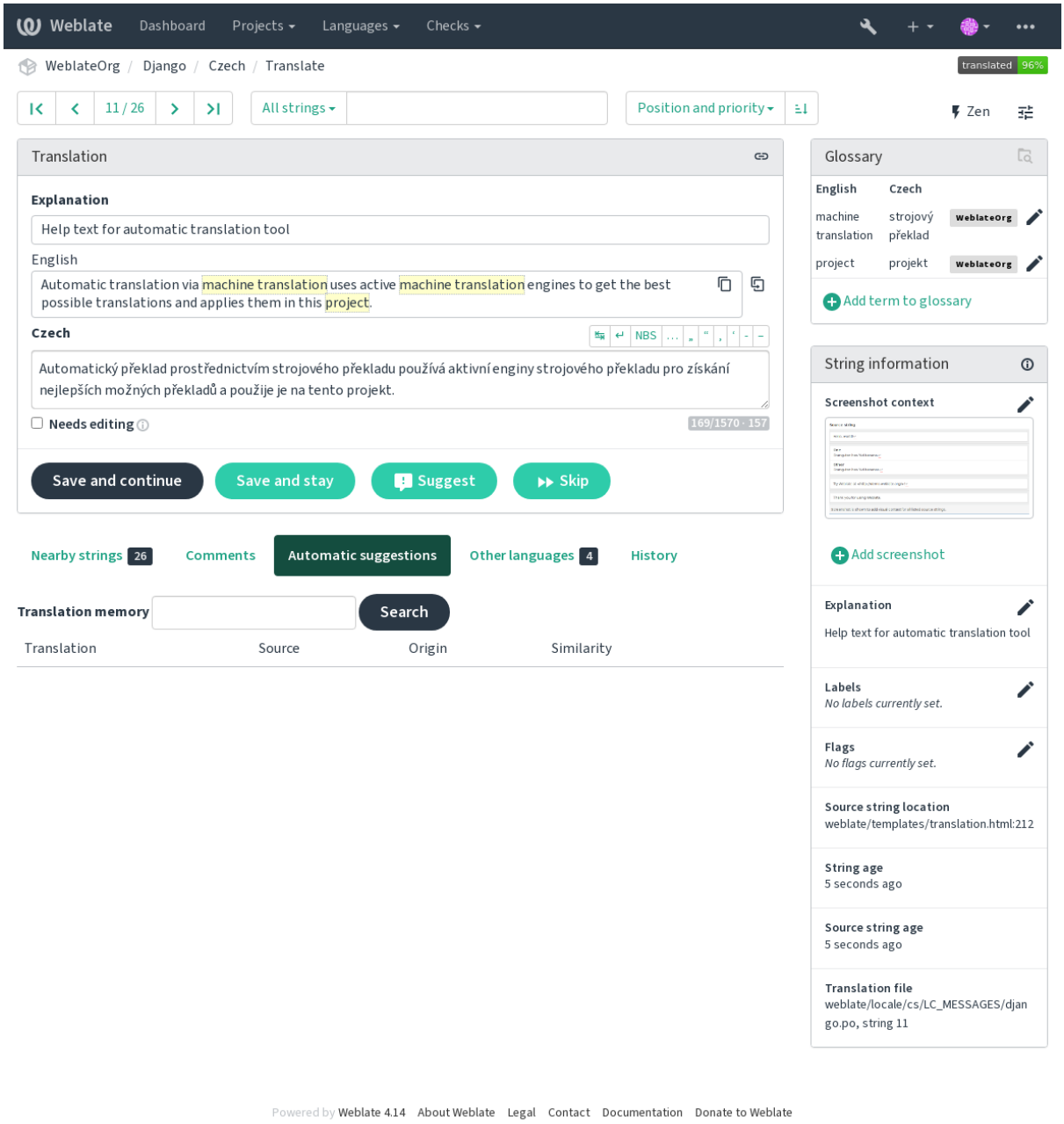
Use the explanation to clarify scope or usage of the translation. You can use Markdown to include links and other markup.

Visual context for strings

Neu in Version 2.9.

Sie können ein Bildschirmfoto hochladen, das die Verwendung einer bestimmten Ausgangszeichenkette in Ihrem Programm zeigt. Dies hilft den Übersetzern zu verstehen, wo sie verwendet wird und wie sie übersetzt werden sollte.

Das hochgeladene Bildschirmfoto wird in der Seitenleiste des Übersetzungskontexts angezeigt:



Zusätzlich zu *Additional info on source strings* haben Bildschirmfotos eine eigene Verwaltungsoberfläche unter dem Menü *Tools*. Laden Sie Bildschirmfotos hoch, ordnen Sie sie manuell den Ausgangszeichenketten zu oder verwenden Sie dazu die optische Zeichenerkennung.

Sobald ein Bildschirmfoto hochgeladen ist, übernimmt diese Schnittstelle die Verwaltung und die Zuordnung der Ausgangszeichenkette:

Weblate Dashboard Projects Languages Checks

WeblateOrg / Django / Screenshots / Automatic translation

Screenshot has been uploaded, you can now assign it to source strings.

Assigned source strings

English	Location	Assigned screenshots	Actions
No matching strings found.			

Screenshot is shown to add visual context for all listed source strings.

Assign source strings

English	Location	Assigned screenshots	Actions
No matching strings found.			

Source string search Search Automatically recognize

Image

Source string

Hello, world!

One
Orangutan has %d banana.

Other
Orangutan has %d bananas.

Try Weblate at <http://demo.weblate.org/>!

Thank you for using Weblate.

Screenshot is shown to add visual context for all listed source strings.

Edit screenshot

Screenshot name

Automatic translation

Image

Currently: screenshots/screenshot.png

Change:

Choose File No file chosen

Upload JPEG or PNG images up to 2000x2000 pixels.

Save

Screenshot details

Created	now
Uploaded by	testuser
Language	English

Delete screenshot

Deleting screenshot will remove it from all associated source strings.

Delete

Powered by Weblate 4.14 About Weblate Legal Contact Documentation Donate to Weblate

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`:

```
#
# Copyright © 2012-2022 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <https://www.gnu.org/licenses/>.
#

from django.utils.translation import gettext_lazy as _

from weblate.trans.autofixes.base import AutoFix


class ReplaceFooWithBar(AutoFix):
    """Replace foo with bar."""

    name = _("Foobar")

    def fix_single_target(self, target, source, unit):
        if "foo" in target:
            return target.replace("foo", "bar"), True
        return target, False
```

To install custom checks, provide a fully-qualified path to the Python class in the `AUTOFIX_LIST`, see *Custom quality checks, add-ons and auto-fixes*.

2.12.2 Customizing behavior using flags

You can fine-tune the Weblate behavior by using flags. This can be done on the source string level (see *Additional info on source strings*), or in the *Component configuration* (*Übersetzungsmarkierungen*). Some file formats also allow to specify flags directly in the format (see *Supported file formats*).

The flags are comma-separated, the parameters are separated with colon. You can use quotes to include whitespace or special chars in the string. For example:

```
placeholders:"special:value":"other value", regex:.*
```

Both single and double quotes are accepted, special characters are being escaped using backslash:

```
placeholders:"quoted \"string\"":'single \'quoted\''
```

Here is a list of flags currently accepted:

rst-text
Treat a text as an reStructuredText document, affects *Unveränderte Übersetzung*.

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 *Schreibgeschützte Zeichenketten*.

priority:N
Priority of the string. Higher priority strings are presented first for translation. The default priority is 100, the higher priority a string has, the earlier it is offered for translation.

max-length:N
Limit the maximal length for a string to N characters, see *Maximale Länge der Übersetzung*.

xml-text
Treat text as XML document, affects *XML-Syntax* and *XML-Auszeichnung*.

font-family:NAME
Define font-family for rendering checks, see *Managing fonts*.

font-weight:WEIGHT
Define font-weight for rendering checks, see *Managing fonts*.

font-size:SIZE
Define font-size for rendering checks, see *Managing fonts*.

font-spacing:SPACING
Define letter spacing for rendering checks, see *Managing fonts*.

icu-flags:FLAGS
Define flags for customizing the behavior of the *ICU MessageFormat* quality check.

icu-tag-prefix:PREFIX
Set a required prefix for XML tags for the *ICU MessageFormat* quality check.

placeholders:NAME:NAME2:...
Placeholder strings expected in translation, see *Platzhalter*.

replacements:FROM:TO:FROM2:TO2...
Replacements to perform when checking resulting text parameters (for example in *Maximaler Umfang der Übersetzung* or *Maximale Länge der Übersetzung*). 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
Regulärer Ausdruck zum Abgleich mit der Übersetzung, siehe *Regulärer Ausdruck*.

forbidden
Indicates forbidden translation in a glossary, see *Forbidden translations*.

strict-same
Make „Unchanged translation“ avoid using built-in words blacklist, see *Unveränderte Übersetzung*.

check-glossary
Aktiviere Qualitätsprüfung *Folgt nicht dem Glossar*.

angularjs-format
Aktiviere Qualitätsprüfung *AngularJS-Interpolations-Zeichenkette*.

c-format
Aktiviere Qualitätsprüfung *C-Format*.

c-sharp-format
Aktiviere Qualitätsprüfung *C#-Format*.

es-format
Aktiviere Qualitätsprüfung *ECMAScript-Buchstabenvorlagen*.

i18next-interpolation
Aktiviere Qualitätsprüfung *i18next Interpolation*.

icu-message-format
Aktiviere Qualitätsprüfung *ICU MessageFormat*.

java-printf-format
Enable the *Java-Format* quality check.

java-format
Aktiviere Qualitätsprüfung *Java-MessageFormat*.

javascript-format
Aktiviere Qualitätsprüfung *JavaScript-Format*.

lua-format
Aktiviere Qualitätsprüfung *Lua-Format*.

object-pascal-format
Aktiviere Qualitätsprüfung *Object Pascal-Format*.

percent-placeholders
Aktiviere Qualitätsprüfung *Platzhalter in Prozent*.

perl-format
Aktiviere Qualitätsprüfung *Perl-Format*.

php-format
Aktiviere Qualitätsprüfung *PHP-Format*.

python-brace-format
Aktiviere Qualitätsprüfung *Python-Brace-Format*.

python-format
Aktiviere Qualitätsprüfung *Python-Format*.

qt-format
Aktiviere Qualitätsprüfung *Qt-Format*.

qt-plural-format
Aktiviere Qualitätsprüfung *Qt-Plural-Format*.

ruby-format
Aktiviere Qualitätsprüfung *Ruby-Format*.

scheme-format
Aktiviere Qualitätsprüfung *Scheme-Format*.

vue-format
Aktiviere Qualitätsprüfung *Vue I18n-Formatierung*.

md-text
Text wie ein Markdown-Dokument behandeln. Aktivieren Sie die Qualitätsprüfungen *Markdown-Links*, *Markdown-Referenzen*, und *Markdown-Syntax*.

case-insensitive
Adjust checks behavior to be case-insensitive. Currently affects only *Platzhalter* quality check.

safe-html
Aktiviere Qualitätsprüfung *Unsicheres HTML*.

url
Die Zeichenkette sollte nur aus einer URL bestehen. Aktivieren Sie die Qualitätsprüfung *URL*.

ignore-all-checks
Ignore all quality checks.

ignore-bbcode
Überspringe Qualitätsprüfung *BBCode-Markup*.

ignore-duplicate
Überspringe Qualitätsprüfung *Aufeinanderfolgende doppelte Wörter*.

ignore-check-glossary
Überspringe Qualitätsprüfung *Folgt nicht dem Glossar*.

ignore-double-space
Überspringe Qualitätsprüfung *Doppeltes Leerzeichen*.

ignore-angularjs-format
Überspringe Qualitätsprüfung *AngularJS-Interpolations-Zeichenkette*.

ignore-c-format
Überspringe Qualitätsprüfung *C-Format*.

ignore-c-sharp-format
Überspringe Qualitätsprüfung *C#-Format*.

ignore-es-format
Überspringe Qualitätsprüfung *ECMAScript-Buchstabenvorlagen*.

ignore-i18next-interpolation
Überspringe Qualitätsprüfung *i18next Interpolation*.

ignore-icu-message-format
Überspringe Qualitätsprüfung *ICU MessageFormat*.

ignore-java-format
Überspringe Qualitätsprüfung *Java-MessageFormat*.

ignore-java-printf-format
Skip the *Java-Format* quality check.

ignore-javascript-format
Überspringe Qualitätsprüfung *JavaScript-Format*.

ignore-lua-format
Überspringe Qualitätsprüfung *Lua-Format*.

ignore-object-pascal-format
Überspringe Qualitätsprüfung *Object Pascal-Format*.

ignore-percent-placeholders
Überspringe Qualitätsprüfung *Platzhalter in Prozent*.

ignore-perl-format
Überspringe Qualitätsprüfung *Perl-Format*.

ignore-php-format
Überspringe Qualitätsprüfung *PHP-Format*.

ignore-python-brace-format
Überspringe Qualitätsprüfung *Python-Brace-Format*.

ignore-python-format
Überspringe Qualitätsprüfung *Python-Format*.

ignore-qt-format
Überspringe Qualitätsprüfung *Qt-Format*.

ignore-qt-plural-format
Überspringe Qualitätsprüfung *Qt-Plural-Format*.

ignore-ruby-format
Überspringe Qualitätsprüfung *Ruby-Format*.

ignore-scheme-format
Überspringe Qualitätsprüfung *Scheme-Format*.

ignore-vue-format
Überspringe Qualitätsprüfung *Vue I18n-Formatierung*.

ignore-translated
Überspringe Qualitätsprüfung *Ist übersetzt worden*.

ignore-inconsistent
Überspringe Qualitätsprüfung *Inkonsistent*.

ignore-kashida
Überspringe Qualitätsprüfung *Kashida-Buchstabe verwendet*.

ignore-md-link
Überspringe Qualitätsprüfung *Markdown-Links*.

ignore-md-reflink
Überspringe Qualitätsprüfung *Markdown-Referenzen*.

ignore-md-syntax
Überspringe Qualitätsprüfung *Markdown-Syntax*.

ignore-max-length
Überspringe Qualitätsprüfung *Maximale Länge der Übersetzung*.

ignore-max-size
Überspringe Qualitätsprüfung *Maximaler Umfang der Übersetzung*.

ignore-escaped-newline
Überspringe Qualitätsprüfung *Mismatched \n*.

ignore-end-colon
Überspringe Qualitätsprüfung *Nicht übereinstimmender Doppelpunkt*.

ignore-end-ellipsis
Überspringe Qualitätsprüfung *Nicht übereinstimmende Auslassungspunkte*.

ignore-end-exclamation
Überspringe Qualitätsprüfung *Nicht übereinstimmendes Ausrufezeichen*.

ignore-end-stop
Überspringe Qualitätsprüfung *Nicht übereinstimmender Punkt*.

ignore-end-question
Überspringe Qualitätsprüfung *Nicht übereinstimmendes Fragezeichen*.

ignore-end-semicolon
Überspringe Qualitätsprüfung *Nicht übereinstimmendes Semikolon*.

ignore-newline-count
Überspringe Qualitätsprüfung *Nicht übereinstimmende Zeilenumbrüche*.

ignore-plurals
Überspringe Qualitätsprüfung *Fehlende Pluralformen*.

ignore-placeholders
Überspringe Qualitätsprüfung *Platzhalter*.

ignore-punctuation-spacing
Überspringe Qualitätsprüfung *Satzzeichenabstand*.

ignore-regex
Überspringe Qualitätsprüfung *Regulärer Ausdruck*.

ignore-same-plurals
Überspringe Qualitätsprüfung *Identische Pluralformen*.

ignore-begin-newline
Überspringe Qualitätsprüfung *Zeilenumbruch am Anfang*.

ignore-begin-space
Überspringe Qualitätsprüfung *Leerzeichen am Anfang*.

ignore-end-newline
Überspringe Qualitätsprüfung *Zeilenumbruch am Ende*.

ignore-end-space
Überspringe Qualitätsprüfung *Leerzeichen am Ende*.

ignore-same
Überspringe Qualitätsprüfung *Unveränderte Übersetzung*.

ignore-safe-html
Überspringe Qualitätsprüfung *Unsicheres HTML*.

ignore-url
Überspringe Qualitätsprüfung *URL*.

ignore-xml-tags
Überspringe Qualitätsprüfung *XML-Auszeichnung*.

ignore-xml-invalid
Überspringe Qualitätsprüfung *XML-Syntax*.

ignore-zero-width-space
Überspringe Qualitätsprüfung *Leerzeichen ohne Breite*.

ignore-ellipsis
Überspringe Qualitätsprüfung *Auslassungspunkte*.

ignore-icu-message-format-syntax
Überspringe Qualitätsprüfung *ICU MessageFormat-Syntax*.

ignore-long-untranslated
Überspringe Qualitätsprüfung *Lange nicht übersetzt*.

ignore-multiple-failures
Überspringe Qualitätsprüfung *Mehrfach fehlgeschlagene Überpüfungen*.

ignore-unnamed-format
Überspringe Qualitätsprüfung *Mehrere unbenannte Variablen*.

ignore-optional-plural
Überspringe Qualitätsprüfung *Ohne Pluralformen*.

Bemerkung: 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

Neu in Version 3.11.

You can configure a list of checks which can not be ignored by setting *Erzwungene Qualitätsprüfungen* in *Component configuration*. Each listed check can not be dismissed in the user interface and any string failing this check is marked as *Needs editing* (see *Translation states*).

2.12.4 Managing fonts

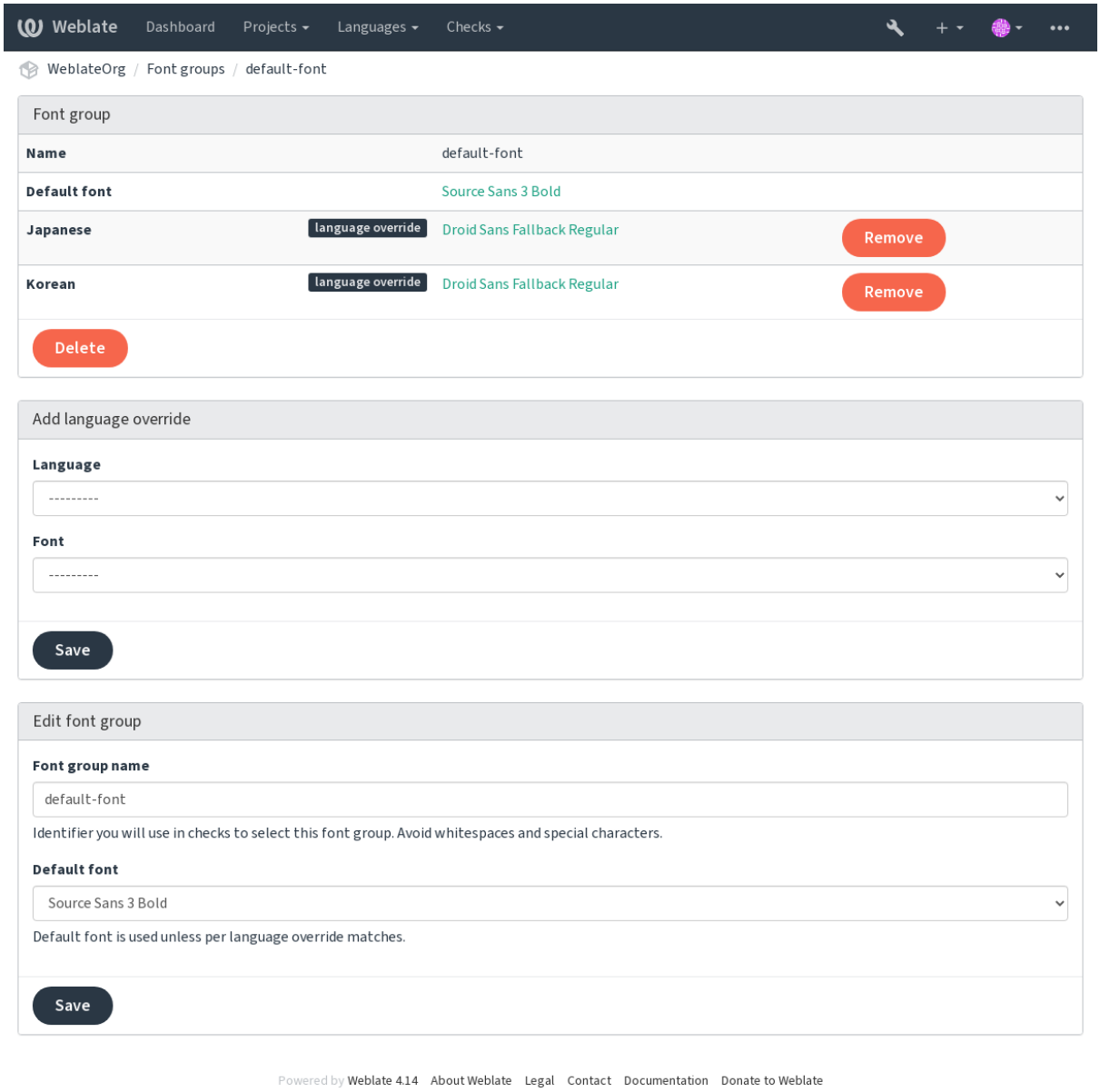
Neu in Version 3.7.

Hinweis: Fonts uploaded into Weblate are used purely for purposes of the *Maximaler Umfang der Übersetzung* check, they do not have an effect in Weblate user interface.

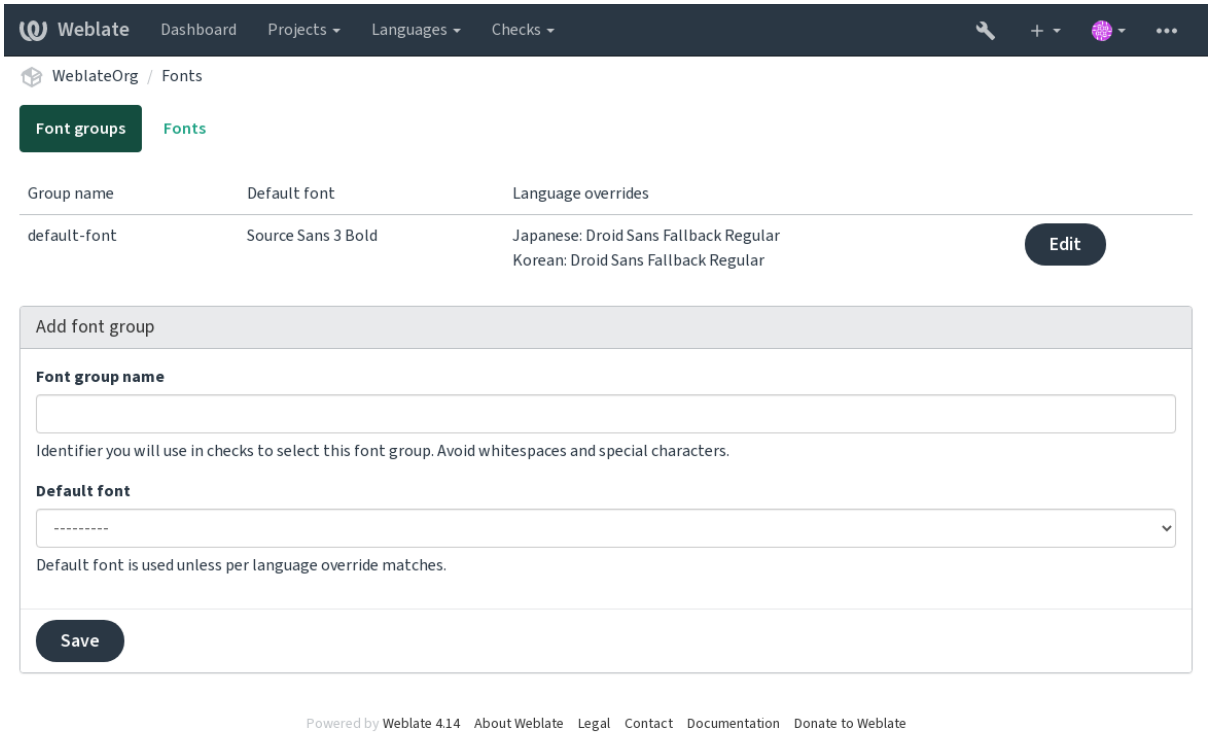
The *Maximaler Umfang der Übersetzung* check used to calculate dimensions of the rendered text needs font to be loaded into Weblate and selected using a translation flag (see *Customizing behavior using flags*).

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.

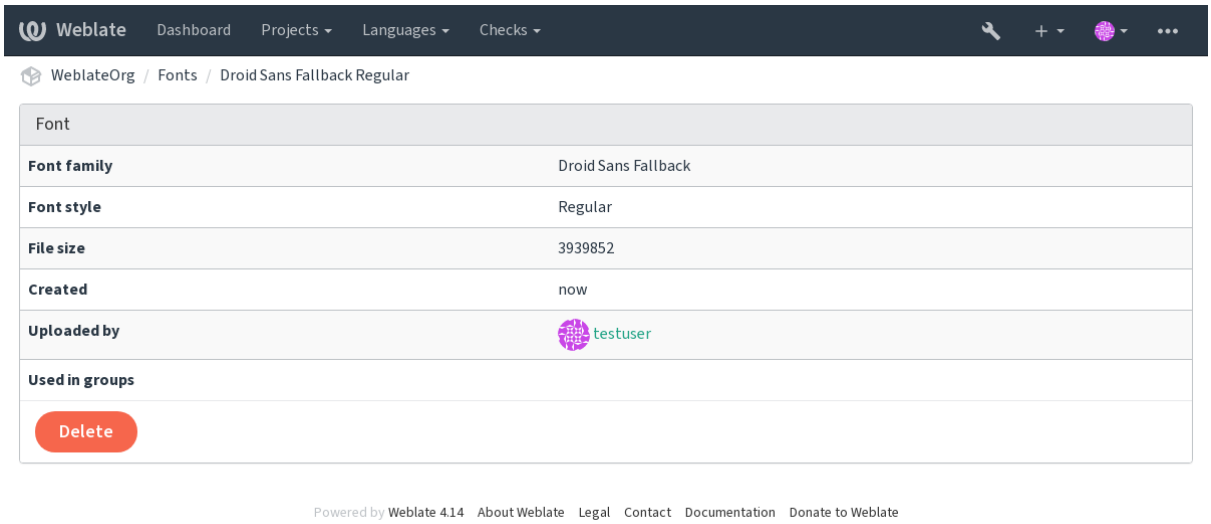
The font-groups allow you to define different fonts for different languages, which is typically needed for non-latin languages:



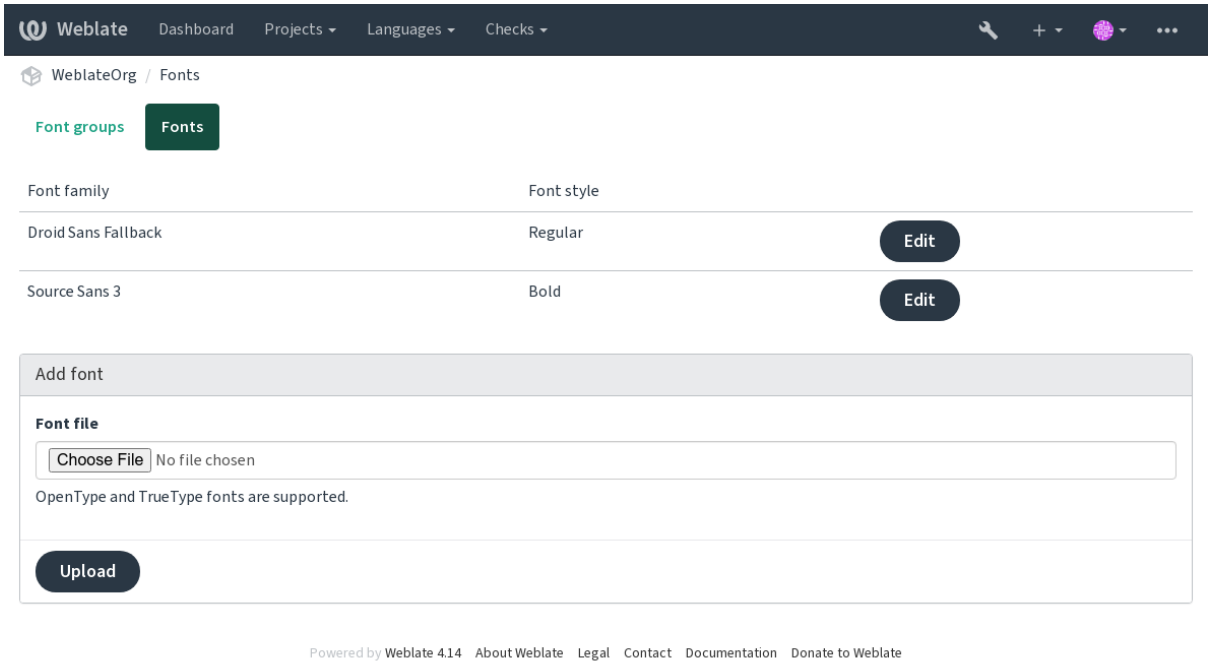
The font-groups are identified by name, which can not contain whitespace or special characters, so that it can be easily used in the check definition:



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`
Defines maximal width in pixels.

`font-family:ubuntu`
Defines font group to use by specifying its identifier.

`font-size:22`
Defines font size in pixels.

2.12.5 Writing own checks

A wide range of quality checks are built-in, (see *Qualitätsprüfungen*), though they might not cover everything you want to check. The list of performed checks can be adjusted using `CHECK_LIST`, and you can also add custom checks.

1. Subclass the *weblate.checks.Check*
2. Set a few attributes.
3. Implement either the `check` (if you want to deal with plurals in your code) or the `check_single` method (which does it for you).

Some examples:

To install custom checks, provide a fully-qualified path to the Python class in the `CHECK_LIST`, see *Custom quality checks, add-ons and auto-fixes*.

Checking translation text does not contain „foo“

This is a pretty simple check which just checks whether the translation is missing the string „foo“.

```
#
# Copyright © 2012-2022 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <https://www.gnu.org/licenses/>.
#
"""Simple quality check example."""

from django.utils.translation import gettext_lazy as _

from weblate.checks.base import TargetCheck


class FooCheck(TargetCheck):

    # Used as identifier for check, should be unique
    # Has to be shorter than 50 characters
    check_id = "foo"

    # Short name used to display failing check
    name = _("Foo check")

    # Description for failing check
    description = _("Your translation is foo")

    # Real check code
    def check_single(self, source, target, unit):
        return "foo" in target
```

Prüfen, ob sich die Pluralformen des tschechischen Übersetzungstextes unterscheiden

Prüfen Sie anhand der Sprachinfo, ob die beiden Pluralformen im Tschechischen nicht gleich sind.

```
#
# Copyright © 2012-2022 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
```

(Fortsetzung auf der nächsten Seite)

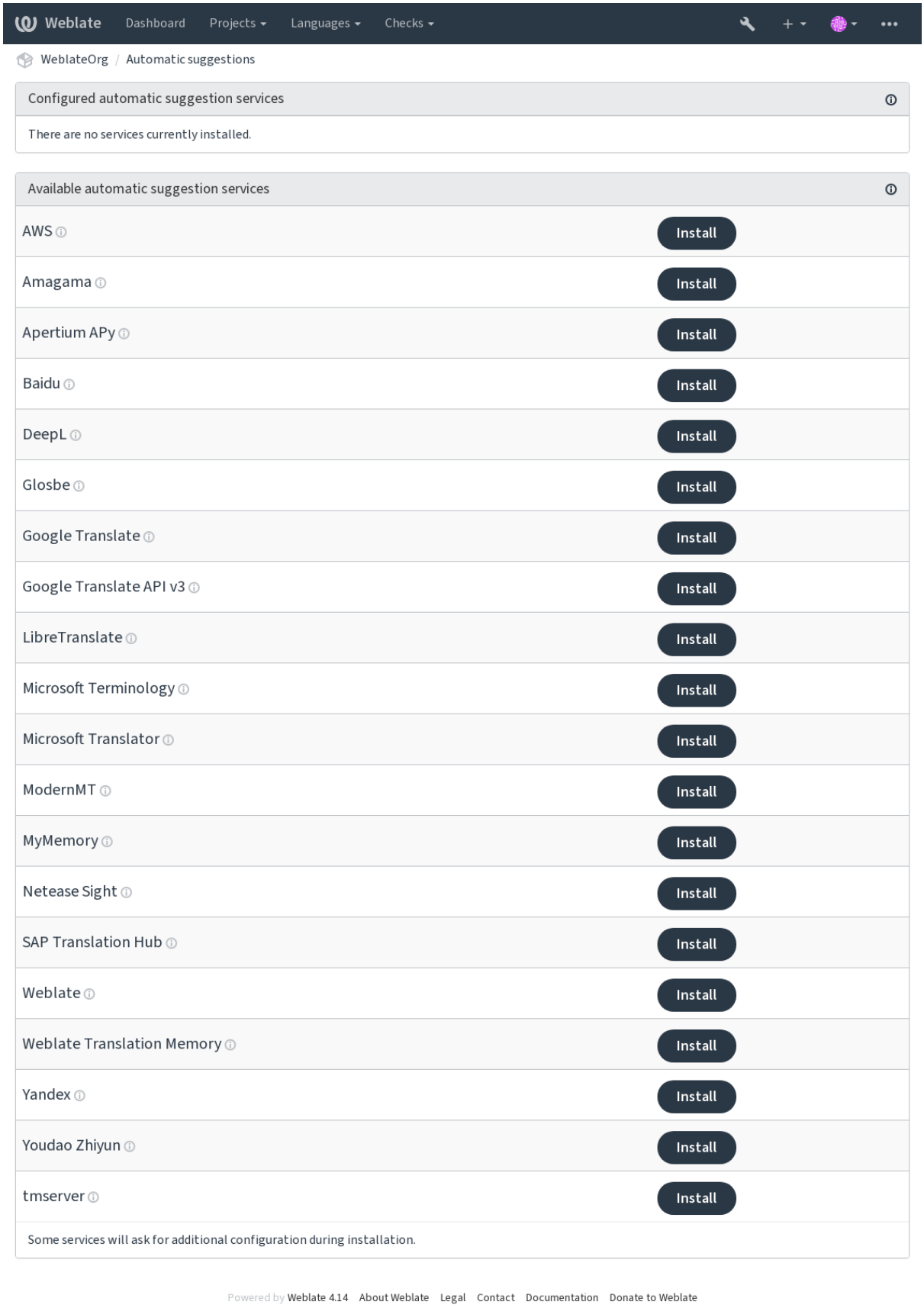
(Fortsetzung der vorherigen Seite)

```
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <https://www.gnu.org/licenses/>.
#
"""Quality check example for Czech plurals."""

from django.utils.translation import gettext_lazy as _

from weblate.checks.base import TargetCheck


class PluralCzechCheck(TargetCheck):

    # Used as identifier for check, should be unique
    # Has to be shorter than 50 characters
    check_id = "foo"

    # Short name used to display failing check
    name = _("Foo check")

    # Description for failing check
    description = _("Your translation is foo")

    # Real check code
    def check_target_unit(self, sources, targets, unit):
        if self.is_language(unit, ("cs",)):
            return targets[1] == targets[2]
        return False

    def check_single(self, source, target, unit):
        """We don't check target strings here."""
        return False
```

2.13 Configuring automatic suggestions

Geändert in Version 4.13: Prior to Weblate 4.13, the services were configured in the *Konfiguration*.

The support for several machine translation and translation memory services is built-in. Each service can be turned on by the administrator for whole site or at the project settings:



Bemerkung: They come subject to their terms of use, so ensure you are allowed to use them how you want.

The services translate from the source language as configured at *Component configuration*, see *Ausgangssprache*.

Siehe auch:

Automatische Vorschläge

2.13.1 Amagama

Dienst-ID
`amagama`

Konfiguration
This service has no configuration.

Special installation of *tmserver* run by the authors of Virtaal.

Siehe auch:

Installing amaGama, Amagama, amaGama Translation Memory

2.13.2 Apertium APy

Dienst-ID
`apertium-apy`

Konfiguration

`url`	API-URL	

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.

Siehe auch:

Apertium website, Apertium APy documentation

2.13.3 AWS

Neu in Version 3.1.

Dienst-ID
`aws`

Konfiguration

`key`	Zugangsschlüssel-ID	
`secret`	Geheimer API-Schlüssel	
`region`	Name der Region	

Amazon Translate is a neural machine translation service for translating text to and from English across a breadth of supported languages.

Siehe auch:

Amazon Translate Documentation

2.13.4 Baidu

Neu in Version 3.2.

Dienst-ID
`baidu`

Konfiguration

`key`	Client-ID	
`secret`	Client-Geheimnis	

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.

Siehe auch:

Baidu Translate API

2.13.5 DeepL

Neu in Version 2.20.

Dienst-ID
`deepl`

Konfiguration

`url`	API-URL	
`key`	API-Schlüssel	

DeepL is paid service providing good machine translation for a few languages. You need to purchase *DeepL API* subscription or you can use legacy *DeepL Pro (classic)* plan.

API-URL zur Verwendung mit dem DeepL-Dienst. Zum Zeitpunkt des Schreibens gibt es die v1-API sowie eine kostenlose und eine kostenpflichtige Version der v2-API.

`https://api.deepl.com/v2/` (Voreinstellung in Weblate)
Ist für die API-Nutzung des kostenpflichtigen Angebots gedacht, und das Abonnement ist nutzungsabhängig.

`https://api-free.deepl.com/v2/`
Ist für die API-Nutzung des kostenlosen Angebots gedacht, und das Abonnement ist nutzungsabhängig.

`https://api.deepl.com/v1/`
Ist für CAT-Tools gedacht und kann mit einem Benutzerabonnement verwendet werden.

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.

Am einfachsten finden Sie dies heraus, indem Sie eine URL wie die folgende in Ihrem Browser öffnen:

https://api.deepl.com/v2/translate?text=Hello&target_lang=FR&auth_key=XXX

Ersetzen Sie die XXX durch Ihren auth_key. Wenn Sie ein JSON-Objekt erhalten, das „Bonjour“ enthält, haben Sie die richtige URL; wenn nicht, probieren Sie die anderen drei aus.

Siehe auch:

DeepL-Website, DeepL-Preise, DeepL-API-Dokumentation

2.13.6 Glosbe

Dienst-ID
`glosbe`

Konfiguration
This service has no configuration.

Free dictionary and translation memory for almost every living language.

The API is gratis to use, but usage of the translations is subject to the license of the used data source. There is a limit of calls that may be done from one IP in a set period of time, to prevent abuse.

Siehe auch:

Glosbe website

2.13.7 Google Translate

Dienst-ID
`google-translate`

Konfiguration

`key`	API-Schlüssel	

Maschineller Übersetzungsdienst von 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.

Siehe auch:

Google Translate-Dokumentation

2.13.8 Google Translate-API v3

Dienst-ID
`google-translate-api-v3`

Konfiguration

`credentials`	Google Translate Kontoinformationen	
`project`	Google Translate-Projekt	
`location`	Google Translate-Gebiet	

Maschineller Übersetzungsdienst, der von Google Cloud-Diensten bereitgestellt wird.

Siehe auch:

Google translate documentation, Getting started with authentication on Google Cloud, Creating Google Translate project, Google Cloud App Engine locations

2.13.9 LibreTranslate

Neu in Version 4.7.1.

Dienst-ID
`libretranslate`

Konfiguration

`url`	API-URL	
`key`	API-Schlüssel	

LibreTranslate ist ein freier und quelloffener Dienst für maschinelle Übersetzungen. Die öffentliche Instanz erfordert einen API-Schlüssel, aber LibreTranslate kann selbst gehostet werden, und es gibt mehrere Mirrors, welche die API kostenlos nutzen.

`https://libretranslate.com/` (offizielle öffentliche Instanz)
Für die Nutzung außerhalb der Website ist ein API-Schlüssel erforderlich.

Siehe auch:

LibreTranslate-Website, LibreTranslate-Repository, LibreTranslate-Mirrors

2.13.10 Microsoft Terminology

Neu in Version 2.19.

Dienst-ID
`microsoft-terminology`

Konfiguration
This service has no configuration.

The Microsoft Terminology Service API allows you to programmatically access the terminology, definitions and user interface (UI) strings available in the Language Portal through a web service.

Siehe auch:

Microsoft Terminology Service API

2.13.11 Microsoft Translator

Neu in Version 2.10.

Dienst-ID
`microsoft-translator`

Konfiguration

`key`	API-Schlüssel	
`endpoint_url`	Endpunkt-URL der Anwendung	
`base_url`	Basis-URL der Anwendung	Verfügbare Auswahlmöglichkeiten: `api.cognitive.microsofttranslator.com` – Global (non-regional) `api-apc.cognitive.microsofttranslator.com` – Asia Pacific `api-eur.cognitive.microsofttranslator.com` – Europe `api-nam.cognitive.microsofttranslator.com` – North America `api.translator.azure.cn` – China
`region`	Anwendungsgebiet	

Machine translation service provided by Microsoft in Azure portal as a one of Cognitive Services.

Weblate implements Translator API V3.

Translator Text API V2

The key you use with Translator API V2 can be used with API 3.

Translator Text API V3

You need to register at Azure portal and use the key you obtain there. With new Azure keys, you also need to set `region` to locale of your service.

Hinweis: For Azure China, please use your endpoint from the Azure Portal.

Siehe auch:

Cognitive Services - Text Translation API, Microsoft Azure Portal, Base URLs, „Authenticating with a Multi-service resource“ „Authenticating with an access token“ section

2.13.12 ModernMT

Neu in Version 4.2.

Dienst-ID
`modernmt`

Konfiguration

`url`	API-URL	
`key`	API-Schlüssel	

Siehe auch:

ModernMT-API,

2.13.13 MyMemory

Dienst-ID
`mymemory`

Konfiguration

`email`	Kontakt E-Mail	
`username`	Benutzername	
`key`	API-Schlüssel	

Huge translation memory with machine translation.

Free, anonymous usage is currently limited to 100 requests/day, or to 1000 requests/day when you provide a contact e-mail address in `email`. You can also ask them for more.

Siehe auch:

MyMemory website

2.13.14 Netease Sight

Neu in Version 3.3.

Dienst-ID
`netease-sight`

Konfiguration

`key`	Client-ID	
`secret`	Client-Geheimnis	

Automatische Übersetzung bereitgestellt von NetEase.

Dieser Dienst verwendet eine API, und Sie müssen Schlüssel und Geheimnis von NetEase abrufen.

Siehe auch:

NetEase Sight Translation Platform

2.13.15 SAP Translation Hub

Dienst-ID
`sap-translation-hub`

Konfiguration

`url`	API-URL	
`key`	API-Schlüssel	
`username`	SAP-Benutzername	
`password`	SAP-Passwort	
`enable_mt`	Maschinelle Übersetzung aktivieren	

Machine translation service provided by SAP.

Sie müssen über ein SAP-Benutzerkonto verfügen (und den SAP Translation Hub in der SAP Cloud Platform aktiviert haben), um diesen Dienst nutzen zu können.

You can also configure whether to also use machine translation services, in addition to the term database.

Bemerkung: To access the Sandbox API, you need to set `url` and `key`.

To access the productive API, you need to set `url`, `username` and `password`.

Siehe auch:

SAP Translation Hub API, Building the Base URL of SAP Translation Hub

2.13.16 tmserver

Dienst-ID
`tmserver`

Konfiguration

`url`	API-URL	

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. Zuerst sollten Sie einige Daten in den Übersetzungsspeicher importieren:

```
build_tmdb -d /var/lib/tm/db -s en -t cs locale/cs/LC_MESSAGES/django.po
build_tmdb -d /var/lib/tm/db -s en -t de locale/de/LC_MESSAGES/django.po
build_tmdb -d /var/lib/tm/db -s en -t fr locale/fr/LC_MESSAGES/django.po
```

2. Starten Sie tmserver, um Ihre Anfragen abzuhören:

```
tmserver -d /var/lib/tm/db
```

3. Configure Weblate to talk to it, the default URL is `http://localhost:8888/tmserver/`.

Siehe auch:

tmserver Installing amaGama, Amagama, Amagama Translation Memory

2.13.17 Weblate

Dienst-ID
`weblate`

Konfiguration
This service has no configuration.

Weblate machine translation service can provide translations for strings that are already translated inside Weblate. It looks for exact matches in the existing strings.

2.13.18 Weblate-Übersetzungsspeicher

Neu in Version 2.20.

Dienst-ID
`weblate-translation-memory`

Konfiguration
This service has no configuration.

Use *Übersetzungsspeicher* as a machine translation service. Any string that has been translated in past (or uploaded to the translation memory) can be translated in this way.

2.13.19 Yandex

Dienst-ID
`yandex`

Konfiguration

`key`	API-Schlüssel	

Automatische Übersetzung bereitgestellt von Yandex.

This service uses a Translation API, and you need to obtain an API key from Yandex.

Siehe auch:

Yandex Translate API, Powered by Yandex.Translate

2.13.20 Youdao Zhiyun

Neu in Version 3.2.

Dienst-ID
`youdao-zhiyun`

Konfiguration

`key`	Client-ID	
`secret`	Client-Geheimnis	

Machine translation service provided by Youdao.

This service uses an API, and you need to obtain an ID and an API key from Youdao.

Siehe auch:

Youdao Zhiyun Natural Language Translation Service

2.13.21 Custom machine translation

You can also implement your own machine translation services using a few lines of Python code. This example implements machine translation in a fixed list of languages using `dictionary` Python module:

```
#
# Copyright © 2012-2022 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <https://www.gnu.org/licenses/>.
#
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
"""Machine translation example."""

import dictionary

from weblate.machinery.base import MachineTranslation


class SampleTranslation(MachineTranslation):
    """Sample machine translation interface."""

    name = "Sample"

    def download_languages(self):
        """Return list of languages your machine translation supports."""
        return {"cs"}

    def download_translations(
        self,
        source,
        language,
        text: str,
        unit,
        user,
        search: bool,
        threshold: int = 75,
    ):
        """Return tuple with translations."""
        for t in dictionary.translate(text):
            yield {"text": t, "quality": 100, "service": self.name, "source": text}
```

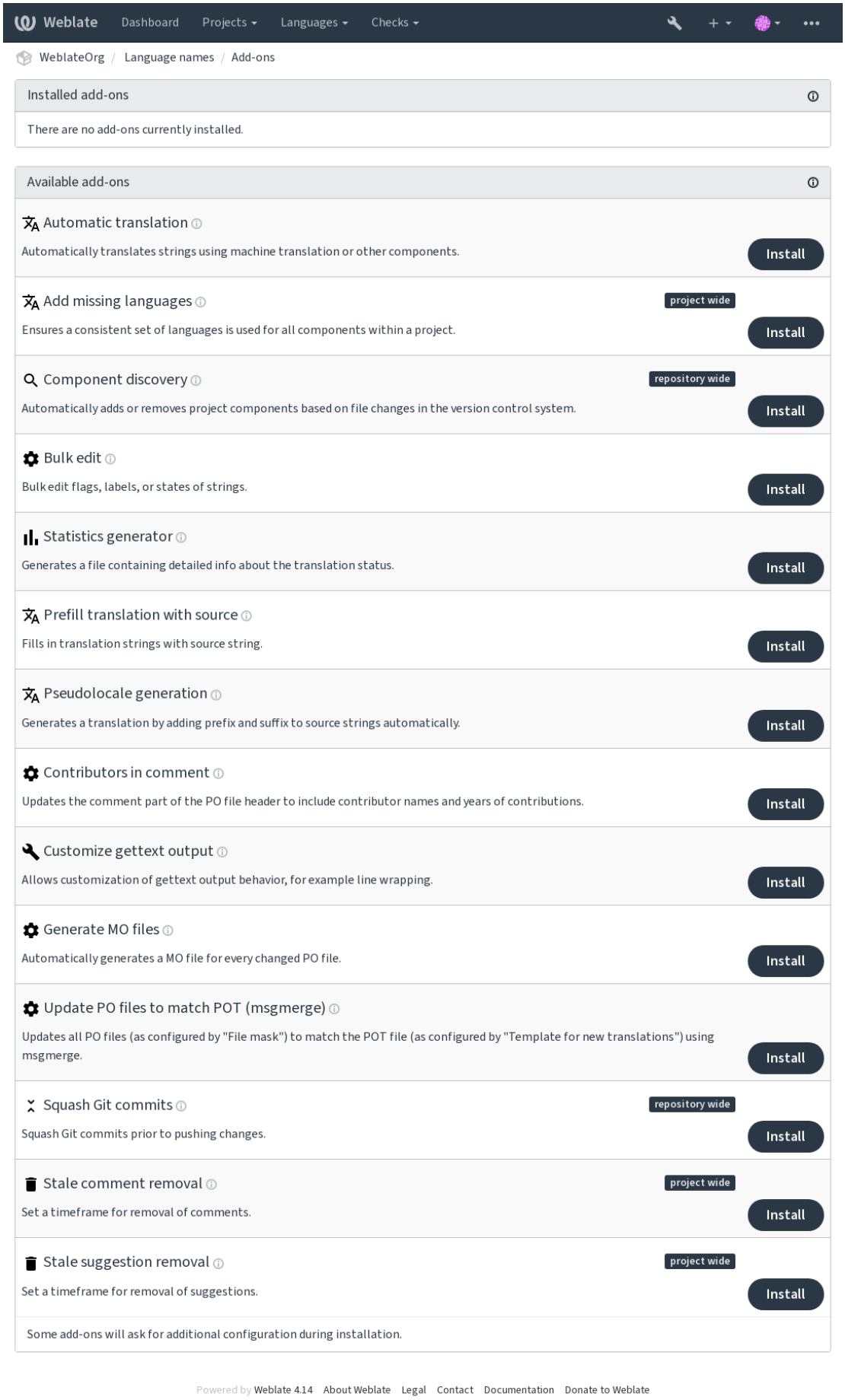
You can list your own class in `WEBLATE_MACHINERY` and Weblate will start using that.

2.14 Erweiterungen

Neu in Version 2.19.

Erweiterungen bieten Möglichkeiten, den Ablauf der Übersetzung anzupassen und zu automatisieren. Administratoren können Erweiterungen über das Menü *Verwaltung* ↓ *Erweiterungen* der jeweiligen Übersetzungskomponente hinzufügen und verwalten.

Hinweis: Sie können Add-ons auch mit *API*, `DEFAULT_ADDONS`, oder `install_addon` konfigurieren.


Weblate
Dashboard
Projects
Languages
Checks
WeblateOrg / Language names / Add-ons
Installed add-ons
There are no add-ons currently installed.
Available add-ons
Automatic translation
Automatically translates strings using machine translation or other components.
Install
Add missing languages
project wide
Ensures a consistent set of languages is used for all components within a project.
Install
Component discovery
repository wide
Automatically adds or removes project components based on file changes in the version control system.
Install
Bulk edit
Bulk edit flags, labels, or states of strings.
Install
Statistics generator
Generates a file containing detailed info about the translation status.
Install
Prefill translation with source
Fills in translation strings with source string.
Install
Pseudolocale generation
Generates a translation by adding prefix and suffix to source strings automatically.
Install
Contributors in comment
Updates the comment part of the PO file header to include contributor names and years of contributions.
Install
Customize gettext output
Allows customization of gettext output behavior, for example line wrapping.
Install
Generate MO files
Automatically generates a MO file for every changed PO file.
Install
Update PO files to match POT (msgmerge)
Updates all PO files (as configured by "File mask") to match the POT file (as configured by "Template for new translations") using msgmerge.
Install
Squash Git commits
repository wide
Squash Git commits prior to pushing changes.
Install
Stale comment removal
project wide
Set a timeframe for removal of comments.
Install
Stale suggestion removal
project wide
Set a timeframe for removal of suggestions.
Install
Some add-ons will ask for additional configuration during installation.
Powered by Weblate 4.14 About Weblate Legal Contact Documentation Donate to Weblate

2.14.1 Integrierte Erweiterungen

Automatische Übersetzung

Neu in Version 3.9.

Erweiterungs-ID
`weblate.autotranslate.autotranslate`

Konfiguration

`mode`	Automatischer Übersetzungsmodus	Verfügbare Auswahlmöglichkeiten: `suggest` – Als Vorschlag hinzufügen `translate` – Als Übersetzung hinzufügen `fuzzy` – Als „Zur Bearbeitung“ hinzufügen
`filter_type`	Suchfilter	Bitte beachten Sie, dass bei der Übersetzung aller Zeichenketten alle vorhandenen Übersetzungen verworfen werden. Verfügbare Auswahlmöglichkeiten: a``all`` – Alle Zeichenketten `nottranslated` – Nicht übersetzte Zeichenketten `todo` – Unfertige Zeichenketten `fuzzy` – Zur Bearbeitung markierte Zeichenketten `check:inconsistent` – Prüfung fehlgeschlagen: Inkonsistent
`auto_source`	Quelle der automatisierten Übersetzungen	Verfügbare Auswahlmöglichkeiten: `others` – Andere Übersetzungskomponenten `mt` – Automatische Übersetzung
`component`	Komponenten	Geben Sie den Slug einer Komponente ein, die als Ausgangszeichenkette verwendet werden soll. Lassen Sie das Feld leer, um alle Komponenten im aktuellen Projekt zu verwenden.
`engines`	Vorschläge aus automatischer Übersetzung	
`threshold`	Scoreschwellwert	

Trigger
Komponenten-Update, täglich

Übersetzt automatisch Zeichenketten mithilfe maschineller Übersetzung oder Übersetzungen aus anderen Komponenten.

Es wird ausgelöst:

- Wenn neue Zeichenketten in einer Komponente erscheinen.
- Einmal monatlich für jede Komponente, dies kann mit `BACKGROUND_TASKS` konfiguriert werden.

Siehe auch:

Automatische Übersetzung, *Keeping translations same across components*

JavaScript-Lokalisierung CDN

Neu in Version 4.2.

Erweiterungs-ID
`weblate.cdn.cdnjs`

Konfiguration

`threshold`	Übersetzungsgrenzwert	Schwellenwert für die Einbindung von Übersetzungen.
`css_selector`	CSS-Selektor	CSS-Selektor zur Erkennung lokalisierbarer Elemente.
`cookie_name`	Name des Sprachcookies	Name des Cookies, der die Sprachpräferenz speichert.
`files`	Zeichenketten aus HTML-Dateien extrahieren	Liste der Dateinamen im aktuellen Repository oder Remote-URLs, die nach übersetzbaren Zeichenkette analysiert werden sollen.

Trigger
daily, repository post-commit, repository post-update

Veröffentlicht Übersetzungen in einem Content Delivery-Netzwerk zur Verwendung in der JavaScript- oder HTML-Lokalisierung.

Kann verwendet werden, um statische HTML-Seiten zu übersetzen oder um die Übersetzung in den JavaScript-Code zu laden.

Erzeugt eine eindeutige URL für Ihre Komponente, die Sie in HTML-Seiten einfügen können, um diese zu lokalisieren. Siehe weblate-cdn für weitere Details.

Siehe auch:

cdn-addon-config, weblate-cdn, cdn-addon-extract, cdn-addon-html

Entferne leere Zeichenketten

Neu in Version 4.4.

Erweiterungs-ID
`weblate.cleanup.blank`

Konfiguration
Diese Erweiterung hat keine Konfiguration.

Trigger
repository post-commit, repository post-update

Entfernt Zeichenketten ohne Übersetzung aus den Übersetzungsdateien.

Verwenden Sie dies, um leere Zeichenketten in Übersetzungsdateien zu vermeiden (z. B. wenn Ihre Lokalisierungsbibliothek sie als fehlend anzeigt, anstatt auf die Quellzeichenfolge zurückzugreifen).

Siehe auch:

Does Weblate update translation files besides translations?

Übersetzungsdateien bereinigen

Erweiterungs-ID
`weblate.cleanup.generic`

Konfiguration
Diese Erweiterung hat keine Konfiguration.

Trigger
repository pre-commit, repository post-update

Aktualisieren Sie alle Übersetzungsdateien so, dass sie mit der einsprachigen Basisdatei übereinstimmen. Für die meisten Dateiformate bedeutet dies, veraltete Übersetzungsschlüssel, die in der Basisdatei nicht mehr vorhanden sind, zu entfernen.

Siehe auch:

Does Weblate update translation files besides translations?

Fehlende Sprachen hinzufügen

Erweiterungs-ID
`weblate.consistency.languages`

Konfiguration
Diese Erweiterung hat keine Konfiguration.

Trigger
daily, repository post-add

Stellt sicher, dass ein konsistenter Satz von Sprachen für alle Komponenten innerhalb eines Projekts verwendet wird.

Fehlende Sprachen werden alle 24 Stunden überprüft, und wenn neue Sprachen in Weblate hinzugefügt werden.

Im Gegensatz zu den meisten anderen wirkt sich diese Erweiterung auf das gesamte Projekt aus.

Hinweis: Auto-translate the newly added strings with *Automatische Übersetzung*.

Komponentenerkennung

Erweiterungs-ID
`weblate.discovery.discovery`

Konfiguration

`match`	Regulärer Ausdruck zum Abgleich von Übersetzungsdateien gegen	
`file_format`	Dateiformat	
`name_template`	Anpassen des Komponentennamens	
`base_file_template`	Definition des einsprachigen Basisdateinamens	Für zweisprachige Übersetzungsdateien leer lassen.
`new_base_template`	Definition der Basisdatei für neue Übersetzungen	Dateiname, der zum Anlegen neuer Übersetzungen verwendet wird. Für gettext .pot-Datei wählen.
`intermediate_template`	Zwischensprachedatei	Dateiname der Zwischenübersetzungsdatei. In den meisten Fällen handelt es sich um eine von Entwicklern bereitgestellte Übersetzungsdatei, die beim Erstellen der eigentlichen Ausgangszeichenketten verwendet wird.
`language_regex`	Sprachen-Filter	Regulärer Ausdruck, gegen den die Übersetzungsdateien beim Scannen nach Dateimaske gefiltert wird.
`copy_addons`	Addons der Hauptkomponente zu den neu erstellten klonen	
`remove`	Komponenten für nicht vorhandene Dateien entfernen	
`confirm`	Ich bestätige, dass die obigen Übereinstimmungen korrekt aussehen	

Trigger
Repository nach dem Update

Automatisches Hinzufügen oder Entfernen von Projektkomponenten basierend auf den Dateiänderungen im Versionskontrollsystem.

Wird jedes mal ausgelöst, wenn das VCS aktualisiert wird, und ähnelt ansonsten dem Verwaltungsbefehl `import_project`. Auf diese Weise können Sie mehrere Übersetzungskomponenten innerhalb eines VCS verfolgen.

Der Abgleich erfolgt über reguläre Ausdrücke, die eine komplexe Konfiguration ermöglichen, für die jedoch einige Kenntnisse erforderlich sind. Einige Beispiele für häufige Anwendungsfälle finden Sie im Hilfebereich der Erweiterung.

Sobald Sie auf *Save* klicken, wird eine Vorschau der passenden Komponenten angezeigt, anhand derer Sie überprüfen können, ob die Konfiguration tatsächlich Ihren Anforderungen entspricht:

Weblate Dashboard Projects Languages Checks

WeblateOrg / Language names / Add-ons / Component discovery

Configure add-on

- Please review and confirm the matched components.

Component	Matched files
The following components would be created	
Djangojs	`weblate/locale/hu/LC_MESSAGES/djangojs.po` (hu) `weblate/locale/cs/LC_MESSAGES/djangojs.po` (cs) `weblate/locale/he/LC_MESSAGES/djangojs.po` (he)
Django	`weblate/locale/hu/LC_MESSAGES/django.po` (hu) `weblate/locale/cs/LC_MESSAGES/django.po` (cs) `weblate/locale/he/LC_MESSAGES/django.po` (he)

☐ **I confirm the above matches look correct**

Regular expression to match translation files against

weblate/locale/(?P<language>[^/]*)/LC_MESSAGES/(?P<component>[^/]*)\.po

File format

gettext PO file

Customize the component name

{{ component|title }}

Define the monolingual base filename

Leave empty for bilingual translation files.

Define the base file for new translations

weblate/locale/{{ component }}.pot

Filename of file used for creating new translations. For gettext choose .pot file.

Intermediate language file

Filename of intermediate translation file. In most cases this is a translation file provided by developers and is used when creating actual source strings.

Language filter

^(cs|he|hu)$

Regular expression to filter translation files against when scanning for file mask.

☑ **Clone add-ons from the main component to the newly created ones**

☐ **Remove components for inexistent files**

The regular expression to match translation files has to contain two named groups to match component and language, some examples:

Regular expression	Example matched files	Description

`(?P<language>[^/.]*)/(?P<component>[^/]*)\.po`	`cs/application.po`
	`cs/website.po`
	`de/application.po`
	`de/website.po`
		One folder per language containing translation files for components.

`locale/(?P<language>[^/.]*)/LC_MESSAGES/(?P<component>[^/]*)\.po`	`locale/cs/LC_MESSAGES/application.po`
	`locale/cs/LC_MESSAGES/website.po`
	`locale/de/LC_MESSAGES/application.po`
	`locale/de/LC_MESSAGES/website.po`
		Usual structure for storing gettext PO files.

`src/locale/(?P<component>[^/]*)\.(?P<language>[^/.]*)\.po`	`src/locale/application.cs.po`
	`src/locale/website.cs.po`
	`src/locale/application.de.po`
	`src/locale/website.de.po`
		Using both component and language name within filename.

`locale/(?P<language>[^/.]*)/(?P<component>[^/]*)/(?P=language)\.po`	`locale/cs/application/cs.po`
	`locale/cs/website/cs.po`
	`locale/de/application/de.po`
	`locale/de/website/de.po`
		Using language in both path and filename.

`res/values-(?P<language>[^/.]*)/strings-(?P<component>[^/]*)\.xml`	`res/values-cs/strings-about.xml`
	`res/values-cs/strings-help.xml`
	`res/values-de/strings-about.xml`
	`res/values-de/strings-help.xml`
		Android resource strings, split into several files.

You can use Django template markup in both component name and the monolingual base filename, for example:

`{{ component }}`
Component filename match
`{{ component|title }}`
Component filename with upper case first letter

Save

Powered by Weblate 4.14 About Weblate Legal Contact Documentation Donate to Weblate

Hinweis: Component discovery add-on uses *Weblate internal URLs*. It's a convenient way to share VCS setup between multiple components. Linked components use the local repository of the main component set up by filling `weblate://project/main-component` into the *Quellcode-Repository* field (in *Manage* ↓ *Settings* ↓ *Version control system*) of each respective component. This saves time with configuration and system resources too.

Siehe auch:

Template markup

Massenbearbeitung

Neu in Version 3.11.

Erweiterungs-ID
`weblate.flags.bulk`

Konfiguration

`q`	Abfrage	
`state`	Zu setzender Status	Verfügbare Auswahlmöglichkeiten: `-1` – Nicht ändern `10` – Bearbeitung erforderlich `20` – Übersetzt `30` – Genehmigt
`add_flags`	Übersetzungsmarkierung zum Hinzufügen	
`remove_flags`	Übersetzungsmarkierung zum Entfernen	
`add_labels`	Labels zum Hinzufügen	
`remove_labels`	Labels zum Entfernen	

Trigger
Komponentenaktualisierung

Massenbearbeitung von Markierungen, Labels oder den Zeichenkettenstatus.

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.

Beispiele:

Tab. 5: Neue Zeichenketten automatisch beschriften

Suchanfrage	`NOT has:label`
Labels zum Hinzufügen	*recent*

Tab. 6: Marking all App-Store Metadatendateien changelog strings read-only

Suchanfrage	`language:en AND key:changelogs/`
Übersetzungsmarkierung zum Hinzufügen	`read-only`

Siehe auch:

Massenbearbeitung, *Customizing behavior using flags*, labels

Unveränderte Übersetzungen als „bearbeitungsbedürftig“ markieren

Neu in Version 3.1.

Erweiterungs-ID
`weblate.flags.same_edit`

Konfiguration
Diese Erweiterung hat keine Konfiguration.

Trigger
unit post-create

Jedes Mal wenn eine neue übersetzbare Zeichenkette aus dem VCS importiert wird und mit der Ausgangszeichenkette übereinstimmt, wird sie in Weblate als zu bearbeiten markiert. Dies ist besonders nützlich für Dateiformate, die nicht übersetzte Ausgangszeichenketten enthalten.

Hinweis: You might also want to tighthen the *Unveränderte Übersetzung* check by adding `strict-same` flag to *Übersetzungsmarkierungen*.

Siehe auch:

Translation states

Neue Ausgangszeichenketten als „bearbeitungsbedürftig“ markieren

Erweiterungs-ID
`weblate.flags.source_edit`

Konfiguration
Diese Erweiterung hat keine Konfiguration.

Trigger
unit post-create

Bei jedem Import einer neuen Ausgangszeichenkette aus dem VCS wird diese in Weblate als bearbeitungsbedürftig gekennzeichnet. Auf diese Weise können Sie die von den Entwicklern geschriebenen Quelltexte einfach filtern und bearbeiten.

Siehe auch:

Translation states

Neue Übersetzungen als „bearbeitungsbedürftig“ markieren

Erweiterungs-ID
`weblate.flags.target_edit`

Konfiguration
Diese Erweiterung hat keine Konfiguration.

Trigger
unit post-create

Bei jedem Import einer neuen übersetzbare Zeichenkette aus dem VCS wird diese in Weblate als bearbeitungsbedürftig gekennzeichnet. Auf diese Weise können Sie die von den Entwicklern geschriebenen Quelltexte einfach filtern und bearbeiten.

Siehe auch:

Translation states

Statistikgenerator

Erweiterungs-ID
`weblate.generate.generate`

Konfiguration

`filename`	Name der erzeugten Datei	
`template`	Inhalt der erzeugten Datei	

Trigger
Repository Pre-Commit

Erzeugt eine Datei mit detaillierten Informationen zum Übersetzungsstatus.

Sie können eine Django-Vorlage sowohl im Dateinamen als auch im Inhalt verwenden, siehe *Template markup* für eine detaillierte Beschreibung des Markups.

Zum Beispiel das Erzeugen einer Zusammenfassungsdatei für jede Übersetzung:

Name der erzeugten Datei
`locale/{{ language_code }}.json`

Inhalt

```
{
   "language": "{{ language_code }}",
   "strings": "{{ stats.all }}",
   "translated": "{{ stats.translated }}",
   "last_changed": "{{ stats.last_changed }}",
   "last_author": "{{ stats.last_author }}",
}
```

Siehe auch:

Template markup

Übersetzung mit Quelle vorbelegen

Neu in Version 4.11.

Erweiterungs-ID
`weblate.generate.prefill`

Konfiguration
Diese Erweiterung hat keine Konfiguration.

Trigger
Komponenten-Update, täglich

Füllt Übersetzungszeichenketten mit Quellzeichenketten aus.

All untranslated strings in the component will be filled with the source string, and marked as needing edit. Use this when you can not have empty strings in the translation files.

Erzeugung von Pseudolokalen

Neu in Version 4.5.

Erweiterungs-ID
`weblate.generate.pseudolocale`

Konfiguration

`source`	Ausgangszeichenketten	
`target`	Zielübersetzung	Alle Zeichenketten in dieser Übersetzung werden überschrieben
`prefix`	Festes Zeichenkettenpräfix	
`var_prefix`	Variables Zeichenkettenpräfix	
`suffix`	Festes Zeichenkettensuffix	
`var_suffix`	Variables Zeichenkettensuffix	
`var_multiplier`	Variabler Teilemultiplikator	Wie oft der variable Teil wiederholt werden soll, hängt von der Länge der Ausgangszeichenkette ab.

Trigger
Komponenten-Update, täglich

Erzeugt eine Übersetzung durch automatisches Hinzufügen von Präfix und Suffix zu Ausgangszeichenketten.

Pseudolocales are useful to find strings that are not prepared for localization. This is done by altering all translatable source strings to make it easy to spot unaltered strings when running the application in the pseudolocale language.

Finding strings whose localized counterparts might not fit the layout is also possible.

Using the variable parts makes it possible to look for strings which might not fit into the user interface after the localization - it extends the text based on the source string length. The variable parts are repeated by length of the text multiplied by the multiplier. For example `Hello world` with variable suffix _ and variable multiplier of 1 becomes `Hello world`_____ - the suffix is repeated once for each character in the source string.

Die Zeichenketten werden nach folgendem Muster erzeugt:

Fixed string prefix Variable string prefix Source string Variable string suffix Fixed string suffix

---

Hinweis: You can use real languages for testing, but there are dedicated pseudolocales available in Weblate - *en_XA* and *ar_XB*.

Hinweis: Sie können diese Erweiterung verwenden, um die Übersetzung in ein neues Gebietsschema einer bestehenden oder ähnlichen Sprache zu starten. Sobald Sie die Übersetzung zur Komponente hinzugefügt haben, folgen Sie der Erweiterung. *Beispiel:* Wenn Sie *fr* haben und die Übersetzung *fr_CA* starten wollen, setzen Sie einfach *fr* als Quelle, *fr_CA* als Ziel und lassen Sie das Präfix und Suffix leer.

Deinstallieren Sie die Erweiterung, sobald Sie die neue Übersetzung ausgefüllt haben, um zu verhindern, dass Weblate die nach dem Kopieren erstellten Übersetzungen verändert.

Mitwirkende im Kommentar

Erweiterungs-ID
`weblate.gettext.authors`

Konfiguration
Diese Erweiterung hat keine Konfiguration.

Trigger
Repository Pre-Commit

Aktualisiert den Kommentarteil in der Kopfzeile der PO-Datei, um den Namen des Mitwirkenden und das Datum des Beitrags aufzunehmen.

Die Kopfzeile der PO-Datei wird wie folgt aussehen:

```
# Michal Čihař <michal@cihar.com>, 2012, 2018, 2019, 2020.
# Pavel Borecki <pavel@example.com>, 2018, 2019.
# Filip Hron <filip@example.com>, 2018, 2019.
# anonymous <noreply@weblate.org>, 2019.
```

Aktualisieren Sie die Variable ALL_LINGUAS in der „configure“-Datei

Erweiterungs-ID
`weblate.gettext.configure`

Konfiguration
Diese Erweiterung hat keine Konfiguration.

Trigger
repository post-add, daily

Updates the ALL_LINGUAS variable in `configure`, `configure.in` or any `configure.ac` files, when a new translation is added.

Ausgabe von Gettext anpassen

Erweiterungs-ID
`weblate.gettext.customize`

Konfiguration

`width`	Umbruch langer Zeilen	Standardmäßig bricht gettext Zeilen mit 77 Zeichen und bei Zeilenvorschüben um. Mit dem Parameter –no-wrap wird der Umbruch nur bei Zeilenvorschub durchgeführt. Verfügbare Auswahlmöglichkeiten: `77` – Zeilenumbruch nach 77 Zeichen und bei Zeilenvorschüben `65535` – Zeilenumbruch nur bei Zeilenvorschüben `-1` – Kein Zeilenumbruch

Trigger
storage post-load

Ermöglicht die Anpassung des Ausgabeverhaltens von Gettext, z. B. Zeilenumbrüche.

Es bietet die folgenden Optionen:

- Zeilenumbruch nach 77 Zeichen und bei Zeilenumbrüchen
- Zeilenumbruch nur nach Zeilenvorschüben
- Kein Umbruch langer Zeilen

Bemerkung: Standardmäßig bricht gettext Zeilen bei 77 Zeichen und bei Zeilenvorschüben um. Mit dem Parameter `--no-wrap` wird der Umbruch nur bei Zeilenvorschüben durchgeführt.

LINGUAS-Datei aktualisieren

Erweiterungs-ID
`weblate.gettext.linguas`

Konfiguration
Diese Erweiterung hat keine Konfiguration.

Trigger
repository post-add, daily

Aktualisiert die LINGUAS-Datei, wenn eine neue Übersetzung hinzugefügt wird.

MO-Dateien erzeugen

Erweiterungs-ID
`weblate.gettext.mo`

Konfiguration

`path`	Pfad der erzeugten MO-Datei	Wenn nicht angegeben, wird der Ort der PO-Datei verwendet.

Trigger
Repository Pre-Commit

Erzeugt automatisch eine MO-Datei für jede geänderte PO-Datei.

The location of the generated MO file can be customized and the field for it uses *Template markup*.

PO-Dateien auf POT aktualisieren (msgmerge)

Erweiterungs-ID
`weblate.gettext.msgmerge`

Konfiguration

`previous`	Frühere msgids der übersetzten Zeichenketten beibehalten	
`no_location`	Speicherorte der übersetzten Zeichenketten entfernen	
`fuzzy`	Fuzzy-Matching verwenden	

Trigger
Repository nach dem Update

Aktualisiert alle PO-Dateien (wie durch *Dateimaske* konfiguriert), damit sie mit der POT-Datei (wie durch *Vorlage für neue Übersetzungen* konfiguriert) übereinstimmen, unter Verwendung von **`msgmerge`**.

Wird immer dann ausgelöst, wenn neue Änderungen aus dem Upstream-Repository übernommen werden. Die meisten msgmerge-Befehlszeilenoptionen können über die Konfiguration der Erweiterung eingerichtet werden.

Siehe auch:

Does Weblate update translation files besides translations?

Git-Commits konsolidieren

Erweiterungs-ID
`weblate.git.squash`

Konfiguration

`squash`	Commits zusammenfassen	Verfügbare Auswahlmöglichkeiten: `all` – Alle Commits in einem `language` – Pro Sprache `file` – Pro Datei `author` – Pro Autor
`append_trailers`	Fügen Sie der zusammengeführten Commit-Nachricht Endzeilen hinzu	Die Endzeilen ähneln RFC 822-E-Mail-Headern, am Ende eines ansonsten freien Teils der Commit-Nachricht, z. B. ‚Mitverfasst von: …‘.
`commit_message`	Commit-Nachricht	Diese Commit-Nachricht wird anstelle der zusammengeführten Commit-Nachrichten verwendet.

Trigger
repository post-commit

Git-Commits vor dem Pushen der Änderungen konsolidieren.

Git-Commits können vor dem Pushen von Änderungen in einem der folgenden Modi unterdrückt werden:

- Alle Commits in einen
- Pro Sprache
- Pro Datei
- Pro Übersetzer

Ursprüngliche Commit-Nachrichten werden erhalten, verlieren aber die Autorenschaft, es sei denn *Pro Autor* wird ausgewählt oder die Commit-Nachricht wird angepasst, um sie einzubeziehen.

Die ursprünglichen Commit-Nachrichten können optional mit einer eigenen Commit-Nachricht überschrieben werden.

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.

Ausgabe von JSON anpassen

Erweiterungs-ID
`weblate.json.customize`

Konfiguration

`sort_keys`	Nach JSON-Schlüssel sortieren	
`indent`	JSON Einrückung (Leerzeichen)	
`style`	JSON-Einrückungsstil	Verfügbare Auswahlmöglichkeiten: `spaces` – Leerzeichen `tabs` – Tabulatoren

Trigger
storage post-load

Ermöglicht die Anpassung des JSON-Ausgabeverhaltens, z. B. Einrückung und Sortierung.

Die Übersetzung der Java-Eigenschaften-Datei formatieren

Erweiterungs-ID
`weblate.properties.sort`

Konfiguration
Diese Erweiterung hat keine Konfiguration.

Trigger
Repository Pre-Commit

Formatiert und sortiert die Java-Eigenschaftendatei.

- Consolidates newlines to Unix ones.
- Uppercase formatting of Unicode escape sequences (in case they are present).
- Strips blank lines and comments.
- Sorts the strings by the keys.
- Drops duplicate strings.

Veralteten Kommentar entfernen

Neu in Version 3.7.

Erweiterungs-ID
`weblate.removal.comments`

Konfiguration

`age`	Tage aufzubewahren	

Trigger
täglich

Legen Sie einen Zeitrahmen für die Entfernung von Kommentaren fest.

Dies kann nützlich sein, um alte Kommentare zu entfernen, die möglicherweise nicht mehr aktuell sind. Seien Sie vorsichtig, denn wenn Kommentare alt werden, bedeutet das nicht, dass sie ihre Bedeutung verloren haben.

Entfernen von veraltetem Vorschlag

Neu in Version 3.7.

Erweiterungs-ID
`weblate.removal.suggestions`

Konfiguration

`age`	Tage aufzubewahren	
`votes`	Abstimmungsschwelle	Schwellwert für die Entfernung. Dieses Feld hat keine Auswirkung, wenn die Abstimmung deaktiviert ist.

Trigger
täglich

Legen Sie einen Zeitrahmen für die Entfernung von Vorschlägen fest.

Kann im Zusammenhang mit Übersetzungsvorschlägen (siehe *Peer-Review*) sehr nützlich sein, um Vorschläge zu entfernen, die innerhalb eines bestimmten Zeitraums nicht genügend positive Stimmen erhalten haben.

RESX-Dateien aktualisieren

Neu in Version 3.9.

Erweiterungs-ID
`weblate.resx.update`

Konfiguration
Diese Erweiterung hat keine Konfiguration.

Trigger
Repository nach dem Update

Aktualisieren Sie alle Übersetzungsdateien entsprechend der einsprachigen Upstream-Basisdatei. Nicht verwendete Zeichenketten werden entfernt und neue Zeichenketten werden als Kopien der Ausgangszeichenkette hinzugefügt.

Hinweis: Verwenden Sie *Übersetzungsdateien bereinigen*, wenn Sie nur veraltete Übersetzungsschlüssel entfernen wollen.

Siehe auch:

Does Weblate update translation files besides translations?

YAML-Ausgabe anpassen

Neu in Version 3.10.2.

Erweiterungs-ID
`weblate.yaml.customize`

Konfiguration

`indent`	YAML Einrückung	
`width`	Umbruch langer Zeilen	Verfügbare Auswahlmöglichkeiten: `80` – Zeilenumbruch bei 80 Zeichen `100` – Zeilenumbruch bei 100 Zeichen `120` – Zeilenumbruch bei 120 Zeichen `180` – Zeilenumbruch bei 180 Zeichen `65535` – Kein Zeilenumbruch
`line_break`	Zeilenumbrüche	Verfügbare Auswahlmöglichkeiten: `dos` – DOS (\r\n) `unix` – UNIX (\n) `mac` – MAC (\r)

Trigger
storage post-load

Ermöglicht die Anpassung des YAML-Ausgabeverhaltens, z. B. Zeilenlänge oder Zeilenvorschübe.

2.14.2 Liste der Erweiterungen anpassen

Die Liste der Erweiterungen wird über die **:Einstellung:`WEBLATE_ADDONS`** konfiguriert. Um eine weitere Erweiterung hinzuzufügen, geben Sie einfach den absoluten Klassennamen in dieser Einstellung an.

2.14.3 Schreiben einer Erweiterung

Sie können auch Ihre eigenen Erweiterungen schreiben, eine Unterklasse von `weblate.addons.base.BaseAddon` erstellen, um die Metadaten der Erweiterung zu definieren, und dann einen Callback implementieren, um die Verarbeitung durchzuführen.

Siehe auch:

Developing add-ons

2.14.4 Ausführen von Skripten der Erweiterung

Erweiterungen können auch verwendet werden, um externe Skripte auszuführen. Früher war dies in Weblate integriert, aber jetzt müssen Sie etwas Code schreiben, um Ihr Skript mit einer Erweiterung zu umschließen.

```
#
# Copyright © 2012-2022 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <https://www.gnu.org/licenses/>.
#
"""Example pre commit script."""


from django.utils.translation import gettext_lazy as _

from weblate.addons.events import EVENT_PRE_COMMIT
from weblate.addons.scripts import BaseScriptAddon


class ExamplePreAddon(BaseScriptAddon):
    # Event used to trigger the script
    events = (EVENT_PRE_COMMIT,)
    # Name of the addon, has to be unique
    name = "weblate.example.pre"
    # Verbose name and long descrption
    verbose = _("Execute script before commit")
    description = _("This add-on executes a script.")

    # Script to execute
    script = "/bin/true"
    # File to add in commit (for pre commit event)
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    # does not have to be set
    add_file = "po/{{ language_code }}.po"
```

Für Installationsanweisungen siehe *Custom quality checks, add-ons and auto-fixes*.

Das Skript wird ausgeführt, wobei das aktuelle Verzeichnis auf das Stammverzeichnis des VCS-Repositorys für eine beliebige Komponente gesetzt wird.

Zusätzlich sind die folgenden Umgebungsvariablen verfügbar:

WL_VCS
Versionsverwaltung verwendet.

WL_REPO
URL des Upstream-Repositorys.

WL_PATH
Absoluter Pfad zum VCS-Repository.

WL_BRANCH
Neu in Version 2.11.

In der aktuellen Komponente konfigurierter Repository-Branch.

WL_FILEMASK
Dateimaske für die aktuelle Komponente.

WL_TEMPLATE
Dateiname der Vorlage für einsprachige Übersetzungen (kann leer sein).

WL_NEW_BASE
Neu in Version 2.14.

Dateiname der Datei, die für die Erstellung neuer Übersetzungen verwendet wird (kann leer sein).

WL_FILE_FORMAT
In der aktuellen Komponente verwendetes Dateiformat.

WL_LANGUAGE
Sprache der aktuell bearbeiteten Übersetzung (nicht verfügbar für Hooks auf Komponentenebene).

WL_PREVIOUS_HEAD
Vorheriger HEAD nach der Aktualisierung (nur verfügbar nach Ausführung des Post-Update-Hooks).

WL_COMPONENT_SLUG
Neu in Version 3.9.

Component slug used to construct URL.

WL_PROJECT_SLUG
Neu in Version 3.9.

Project slug used to construct URL.

WL_COMPONENT_NAME
Neu in Version 3.9.

Name der Komponente.

WL_PROJECT_NAME
Neu in Version 3.9.

Projektname.

WL_COMPONENT_URL

Neu in Version 3.9.

URL der Komponente.

WL_ENGAGE_URL

Neu in Version 3.9.

URL des beteiligten Projekts.

Siehe auch:

Component configuration

Repository-Verarbeitung nach Aktualisierung

Kann verwendet werden, um Übersetzungsdateien zu aktualisieren, wenn sich der Upstream-Quellcode des VCS ändert. Um dies zu erreichen, denken Sie bitte daran, dass Weblate nur Dateien sieht, die an das VCS übertragen wurden, so dass Sie die Änderungen als Teil des Skripts übertragen müssen.

Zum Beispiel mit Gulp können Sie es mit folgendem Code machen:

```
#! /bin/sh
gulp --gulpfile gulp-i18n-extract.js
git commit -m 'Update source strings' src/languages/en.lang.json
```

Verarbeitung von Übersetzungen vor der Freigabe

Verwenden Sie das Commit-Skript, um eine Übersetzung automatisch zu ändern, bevor sie in das Repository übertragen wird.

Sie wird als einzelner Parameter übergeben, der aus dem Dateinamen der aktuellen Übersetzung besteht.

2.15 Übersetzungsspeicher

Neu in Version 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, *Automatische Vorschläge* view while translating.
- Automatically, by translating strings using *Automatische Übersetzung*, or *Automatische Übersetzung* add-on.

For installation tips, see *Weblate-Übersetzungsspeicher*, which is turned on by default.

2.15.1 Translation memory scopes

Neu in Version 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.

Geteilter Übersetzungspeicher

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

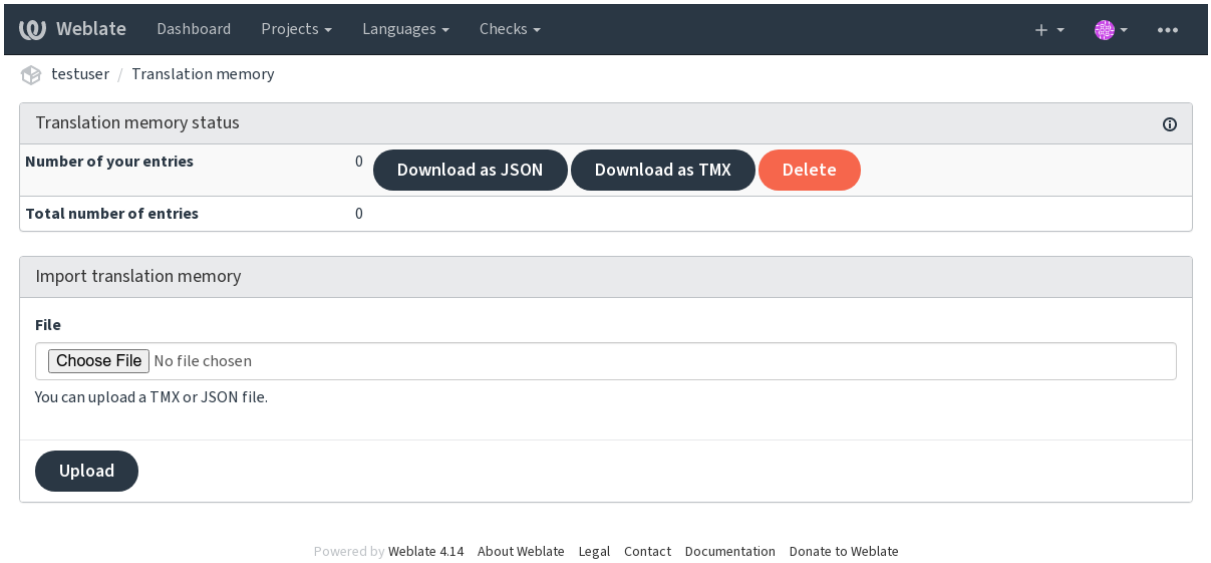
Neu in Version 3.2.

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.

Hinweis: Translation memory in JSON can be imported into Weblate, TMX is provided for interoperability with other tools.

Siehe auch:

Weblate-Übersetzungsspeicher-Schema



Verwaltungsoberfläche

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 Konfiguration

All settings are stored in `settings.py` (as is usual for Django).

Bemerkung: 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.

Siehe auch:

Please also check Django's documentation for parameters configuring Django itself.

2.16.1 AKISMET_API_KEY

Weblate can use Akismet to check incoming anonymous suggestions for spam. Visit akismet.com to purchase an API key and associate it with a site.

2.16.2 ANONYMOUS_USER_NAME

Username of users that are not signed in.

Siehe auch:

Zugriffssteuerung

2.16.3 AUDITLOG_EXPIRY

Neu in Version 3.6.

How many days Weblate should keep audit logs, which contain info about account activity.

Defaults to 180 days.

2.16.4 AUTH_LOCK_ATTEMPTS

Neu in Version 2.14.

Maximum number of failed authentication attempts before rate limiting is applied.

This is currently applied in the following locations:

- Sign in. Deletes the account password, preventing the user from signing in without requesting a new password.
- Password reset. Prevents new e-mails from being sent, avoiding spamming users with too many password reset attempts.

Defaults to 10.

Siehe auch:

Rate limiting

2.16.5 AUTO_UPDATE

Neu in Version 3.2.

Geändert in Version 3.11: The original on/off option was changed to differentiate which strings are accepted.

Updates all repositories on a daily basis.

Hinweis: Useful if you are not using *Benachrichtigungs-Hooks* to update Weblate repositories automatically.

Bemerkung: On/off options exist in addition to string selection for backward compatibility.

Options are:

`"none"`
No daily updates.

`"remote"` also `False`
Only update remotes.

`"full"` also `True`
Update remotes and merge working copy.

Bemerkung: This requires that *Background tasks using Celery* is working, and will take effect after it is restarted.

2.16.6 AVATAR_URL_PREFIX

Prefix for constructing avatar URLs as: `${AVATAR_URL_PREFIX}/avatar/${MAIL_HASH}?${PARAMS}`. The following services are known to work:

Gravatar (default), as per https://gravatar.com/

```
AVATAR_URL_PREFIX = 'https://www.gravatar.com/'
```

Libravatar, as per https://www.libravatar.org/

```
AVATAR_URL_PREFIX = 'https://www.libravatar.org/'
```

Siehe auch:

Avatar-Zwischenspeicherung, `ENABLE_AVATARS`, *Avatars*

2.16.7 AUTH_TOKEN_VALID

Neu in Version 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

Neu in Version 2.15.

How many days using the same password should be allowed.

Bemerkung: Password changes made prior to Weblate 2.15 will not be accounted for in this policy.

Defaults to 180 days.

2.16.9 AUTOFIX_LIST

List of automatic fixes to apply when saving a string.

Bemerkung: Provide a fully-qualified path to the Python class that implementing the autofixer interface.

Available fixes:

`weblate.trans.autofixes.whitespace.SameBookendingWhitespace`
Matches whitespace at the start and end of the string to the source.

`weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis`
Replaces trailing dots (…) if the source string has a corresponding ellipsis (…).

`weblate.trans.autofixes.chars.RemoveZeroSpace`
Removes zero-width space characters if the source does not contain any.

`weblate.trans.autofixes.chars.RemoveControlChars`
Removes control characters if the source does not contain any.

`weblate.trans.autofixes.html.BleachHTML`
Removes unsafe HTML markup from strings flagged as `safe-html` (see *Unsicheres HTML*).

You can select which ones to use:

```
AUTOFIX_LIST = (
    "weblate.trans.autofixes.whitespace.SameBookendingWhitespace",
    "weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis",
)
```

Siehe auch:

Automatische Korrekturen, *Custom automatic fixups*

2.16.10 BACKGROUND_TASKS

Neu in Version 4.5.2.

Defines how often lengthy maintenance tasks should be triggered for a component.

Right now this controls:

- *Automatische Übersetzung* Erweiterung
- *Checks and fixups* recalculation

Mögliche Auswahlmöglichkeiten:

- `monthly` (this is the default)
- `weekly`
- `daily`
- `never`

Bemerkung: Increasing the frequency is not recommended when Weblate contains thousands of components.

2.16.11 BASIC_LANGUAGES

Neu in Version 4.4.

Liste der Sprachen, die dem Benutzer zum Starten einer neuen Übersetzung angeboten wird. Wenn nicht angegeben, wird die integrierte Liste mit allen häufig verwendeten Sprachen verwendet, jedoch ohne länderspezifische Varianten.

This only limits non privileged users to add unwanted languages. The project admins are still presented with full selection of languages defined in Weblate.

Bemerkung: This does not define new languages for Weblate, it only filters existing ones in the database.

Beispiel:

```
BASIC_LANGUAGES = {"cs", "it", "ja", "en"}
```

Siehe auch:

Language definitions

2.16.12 BORG_EXTRA_ARGS

Neu in Version 4.9.

You can pass additional arguments to **borg create** when built-in backups are triggered.

Beispiel:

```
BORG_EXTRA_ARGS = ["--exclude", "vcs/"]
```

Siehe auch:

Sichern und Verschieben von Weblate, borg create

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.

Beispiel:

```
# Enable Cloudflare Javascript optimizations
CSP_SCRIPT_SRC = ["ajax.cloudflare.com"]
```

Siehe auch:

Content security policy, Content Security Policy (CSP)

2.16.14 CHECK_LIST

List of quality checks to perform on a translation.

Bemerkung: 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 *Qualitätsprüfungen* are turned on by default, from where you can change these settings. By default they are commented out in *Sample configuration* so that default values are used. New checks then carried out for each new Weblate version.

You can turn off all checks:

```
CHECK_LIST = ()
```

You can turn on only a few:

```
CHECK_LIST = (
    "weblate.checks.chars.BeginNewlineCheck",
    "weblate.checks.chars.EndNewlineCheck",
    "weblate.checks.chars.MaxLengthCheck",
)
```

Bemerkung: 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`.

Siehe auch:

Qualitätsprüfungen, *Customizing behavior using flags*

2.16.15 COMMENT_CLEANUP_DAYS

Neu in Version 3.6.

Delete comments after a given number of days. Defaults to `None`, meaning no deletion at all.

2.16.16 COMMIT_PENDING_HOURS

Neu in Version 2.10.

Number of hours between committing pending changes by way of the background task.

Siehe auch:

Component configuration, *Alter der Änderungen, das erreicht sein muss, bevor ein Commit erfolgt*, *Running maintenance tasks*, `commit_pending`

2.16.17 CONTACT_FORM

Neu in Version 4.6.

Legt fest, wie E-Mails aus dem Kontaktformular gesendet werden. Wählen Sie eine Konfiguration, die Ihrer Mail-Server-Konfiguration entspricht.

"reply-to"
Der Absender wird in `Reply-To` verwendet, dies ist das Standardverhalten.

"from"
Der Absender wird in `From` verwendet. Ihr Mail-Server muss den Versand solcher E-Mails erlauben.

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.

The following subdirectories usually exist:

home
Home directory used for invoking scripts.

ssh
SSH keys and configuration.

static
Default location for static Django files, specified by `STATIC_ROOT`. See *Serving static files*.

Der Docker-Container verwendet dafür ein eigenes Volume, siehe *Docker-Container-Volumes*.

media
Standardspeicherort für Django-Mediendateien, angegeben durch `MEDIA_ROOT`. Enthält hochgeladene Bildschirmfotos, siehe Bildschirmfotos.

vcs
Versionsverwaltung-Repositorys für Übersetzungen.

backups
Daily backup data, please check *Dumped data for backups* for details.

celery
Celery scheduler data, see *Background tasks using Celery*.

fonts:
User-uploaded fonts, see *Managing fonts*.

Bemerkung: This directory has to be writable by Weblate. Running it as uWSGI means the `www-data` user should have write access to it.

The easiest way to achieve this is to make the user the owner of the directory:

```
sudo chown www-data:www-data -R $DATA_DIR
```

Defaults to `/home/weblate/data`, but it is expected to be configured.

Siehe auch:

Filesystem permissions, *Sichern und Verschieben von Weblate*

2.16.19 DATABASE_BACKUP

Neu in Version 3.1.

Whether the database backups should be stored as plain text, compressed or skipped. The authorized values are:

- `"plain"`
- `"compressed"`
- `"none"`

Siehe auch:

Sichern und Verschieben von Weblate

2.16.20 DEFAULT_ACCESS_CONTROL

Neu in Version 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.

Siehe auch:

Projekt-Zugriffssteuerung, *Zugriffssteuerung*

2.16.21 DEFAULT_AUTO_WATCH

Neu in Version 4.5.

Configures whether *Automatically watch projects on contribution* should be turned on for new users. Defaults to `True`.

Siehe auch:

Benachrichtigungen

2.16.22 DEFAULT_RESTRICTED_COMPONENT

Neu in Version 4.1.

The default value for component restriction.

Siehe auch:

Restricted access, *Umfang der Gruppen*

2.16.23 DEFAULT_ADD_MESSAGE, DEFAULT_ADDON_MESSAGE, DEFAULT_COMMIT_MESSAGE, DEFAULT_DELETE_MESSAGE, DEFAULT_MERGE_MESSAGE

Default commit messages for different operations, please check *Component configuration* for details.

Siehe auch:

Template markup, *Component configuration*, *Commit, add, delete, merge, add-on, and merge request messages*

2.16.24 DEFAULT_ADDONS

Default add-ons to install on every created component.

Bemerkung: This setting affects only newly created components.

Example:

```
DEFAULT_ADDONS = {
    # Add-on with no parameters
    "weblate.flags.target_edit": {},
    # Add-on with parameters
    "weblate.autotranslate.autotranslate": {
        "mode": "suggest",
        "filter_type": "todo",
        "auto_source": "mt",
        "component": "",
        "engines": ["weblate-translation-memory"],
        "threshold": "80",
    },
}
```

Siehe auch:

`install_addon`, *Erweiterungen*, `WEBLATE_ADDONS`

2.16.25 DEFAULT_COMMITER_EMAIL

Neu in Version 2.4.

Committer e-mail address defaulting to `noreply@weblate.org`.

Siehe auch:

DEFAULT_COMMITER_NAME

2.16.26 DEFAULT_COMMITER_NAME

Neu in Version 2.4.

Committer name defaulting to `Weblate`.

Siehe auch:

DEFAULT_COMMITER_EMAIL

2.16.27 DEFAULT_LANGUAGE

Neu in Version 4.3.2.

Default source language to use for example in *Ausgangssprache*.

Defaults to *en*. The matching language object needs to exist in the database.

Siehe auch:

Language definitions, *Ausgangssprache*

2.16.28 DEFAULT_MERGE_STYLE

Neu in Version 3.4.

Merge style for any new components.

- *rebase* - default
- *merge*

Siehe auch:

Component configuration, *Git-Strategie*

2.16.29 DEFAULT_SHARED_TM

Neu in Version 3.2.

Configures default value of *Gemeinsamen Übersetzungsspeicher verwenden* and *Zu einem gemeinsamen Übersetzungsspeicher beitragen*.

2.16.30 DEFAULT_TRANSLATION_PROPAGATION

Neu in Version 2.5.

Default setting for translation propagation, defaults to `True`.

Siehe auch:

Component configuration, *Verbreitung von Übersetzungen erlauben*

2.16.31 DEFAULT_PULL_MESSAGE

Configures the default title and message for pull requests.

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.

Siehe auch:

Avatar-Zwischenspeicherung, `AVATAR_URL_PREFIX`, *Avatars*

2.16.33 ENABLE_HOOKS

Whether to enable anonymous remote hooks.

Siehe auch:

Benachrichtigungs-Hooks

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.

Siehe auch:

`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

Neu in Version 4.5.2.

URL where support for your Weblate instance can be found.

2.16.37 GITEA_CREDENTIALS

Neu in Version 4.12.

List for credentials for Gitea servers.

Hinweis: Use this in case you want Weblate to interact with more of them, for single Gitea endpoint stick with `GITEA_USERNAME` and `GITEA_TOKEN`.

```
GITEA_CREDENTIALS = {
    "try.gitea.io": {
        "username": "weblate",
        "token": "your-api-token",
    },
    "gitea.example.com": {
        "username": "weblate",
        "token": "another-api-token",
    },
}
```

2.16.38 GITEA_USERNAME

Neu in Version 4.12.

Gitea username used to send pull requests for translation updates.

Siehe auch:

`GITEA_CREDENTIALS`, *Gitea pull requests*

2.16.39 GITEA_TOKEN

Neu in Version 4.12.

Gitea personal access token used to make API calls to send pull requests for translation updates.

Siehe auch:

`GITEA_CREDENTIALS`, *Gitea pull requests*, Creating a Gitea personal access token

2.16.40 GITLAB_CREDENTIALS

Neu in Version 4.3.

List for credentials for GitLab servers.

Hinweis: Use this in case you want Weblate to interact with more of them, for single GitLab endpoint stick with `GITLAB_USERNAME` and `GITLAB_TOKEN`.

```
GITLAB_CREDENTIALS = {
    "gitlab.com": {
        "username": "weblate",
        "token": "your-api-token",
    },
    "gitlab.example.com": {
        "username": "weblate",
        "token": "another-api-token",
    },
}
```

2.16.41 GITLAB_USERNAME

GitLab username used to send merge requests for translation updates.

Siehe auch:

GITLAB_CREDENTIALS, *GitLab Merge Requests*

2.16.42 GITLAB_TOKEN

Neu in Version 4.3.

GitLab personal access token used to make API calls to send merge requests for translation updates.

Siehe auch:

GITLAB_CREDENTIALS, *GitLab Merge Requests*, GitLab: Personal access token

2.16.43 GITHUB_CREDENTIALS

Neu in Version 4.3.

List for credentials for GitHub servers.

Hinweis: 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",
    },
}
```

2.16.44 GITHUB_USERNAME

GitHub username used to send pull requests for translation updates.

Siehe auch:

`GITHUB_CREDENTIALS`, *GitHub-Pull-Anfragen*

2.16.45 GITHUB_TOKEN

Neu in Version 4.3.

GitHub personal access token used to make API calls to send pull requests for translation updates.

Siehe auch:

`GITHUB_CREDENTIALS`, *GitHub-Pull-Anfragen*, Erstellen eines persönlichen GitHub-Zugangstokens

2.16.46 GOOGLE_ANALYTICS_ID

Google Analytics ID to turn on monitoring of Weblate using Google Analytics.

2.16.47 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.

Bemerkung: This is turned on by default.

2.16.48 HIDE_VERSION

Neu in Version 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.

Bemerkung: Diese Funktion ist standardmäßig ausgeschaltet.

2.16.49 INTERLEDGER_PAYMENT_POINTERS

Neu in Version 4.12.1.

List of Interledger Payment Pointers (ILPs) for Web Monetization.

If multiple are specified, probabilistic revenue sharing is achieved by selecting one randomly.

Please check <https://webmonetization.org/> for more details.

Hinweis: The default value lets users fund Weblate itself.

2.16.50 IP_BEHIND_REVERSE_PROXY

Neu in Version 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*.

Warnung: Ensure you are actually using a reverse proxy and that it sets this header, otherwise users will be able to fake the IP address.

Bemerkung: This is not on by default.

Siehe auch:

Running behind reverse proxy, *Rate limiting*, *IP_PROXY_HEADER*, *IP_PROXY_OFFSET*

2.16.51 IP_PROXY_HEADER

Neu in Version 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`.

Siehe auch:

Running behind reverse proxy, *Rate limiting*, `SECURE_PROXY_SSL_HEADER`, *IP_BEHIND_REVERSE_PROXY*, *IP_PROXY_OFFSET*

2.16.52 IP_PROXY_OFFSET

Neu in Version 2.14.

Indicates which part of *IP_PROXY_HEADER* is used as client IP address.

Abhängig von Ihrer Einrichtung kann dieser Header aus mehreren IP-Adressen bestehen (z.B. `X-Forwarded-For: a, b, client-ip`) und Sie können hier konfigurieren, welche Adresse aus dem Header als Client-IP-Adresse verwendet wird.

Warnung: Setting this affects the security of your installation, you should only configure it to use trusted proxies for determining IP address.

Defaults to 0.

Siehe auch:

Running behind reverse proxy, *Rate limiting*, `SECURE_PROXY_SSL_HEADER`, *IP_BEHIND_REVERSE_PROXY*, *IP_PROXY_HEADER*

2.16.53 LEGAL_URL

Neu in Version 3.5.

URL where your Weblate instance shows its legal documents.

Hinweis: Useful if you host your legal documents outside Weblate for embedding them inside Weblate, please check *Rechtliche Grundlagen* for details.

Example:

```
LEGAL_URL = "https://weblate.org/terms/"
```

Siehe auch:

PRIVACY_URL

2.16.54 LICENSE_EXTRA

Additional licenses to include in the license choices.

Bemerkung: Each license definition should be tuple of its short name, a long name and an URL.

For example:

```
LICENSE_EXTRA = [
    (
        "AGPL-3.0",
        "GNU Affero General Public License v3.0",
        "https://www.gnu.org/licenses/agpl-3.0-standalone.html",
    ),
]
```

2.16.55 LICENSE_FILTER

Geändert in Version 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.

Bemerkung: This filter uses the short license names.

For example:

```
LICENSE_FILTER = {"AGPL-3.0", "GPL-3.0-or-later"}
```

Following disables the license alert:

```
LICENSE_FILTER = set()
```

Siehe auch:

alerts

2.16.56 LICENSE_REQUIRED

Defines whether the license attribute in *Component configuration* is required.

Bemerkung: This is off by default.

2.16.57 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.

Hinweis: Set this to `False` to allow longer translations (up to 10,000 characters) irrespective of source string length.

Bemerkung: Defaults to `True`.

2.16.58 LOCALIZE_CDN_URL and LOCALIZE_CDN_PATH

These settings configure the *JavaScript-Lokalisierung CDN* add-on. `LOCALIZE_CDN_URL` defines root URL where the localization CDN is available and `LOCALIZE_CDN_PATH` defines path where Weblate should store generated files which will be served at the `LOCALIZE_CDN_URL`.

Hinweis: On Hosted Weblate, this uses `https://weblate-cdn.com/`.

Siehe auch:

JavaScript-Lokalisierung CDN

2.16.59 LOGIN_REQUIRED_URLS

A list of URLs you want to require signing in. (Besides the standard rules built into Weblate).

Hinweis: This allows you to password protect a whole installation using:

```
LOGIN_REQUIRED_URLS = (r"/(.*)$",)
REST_FRAMEWORK["DEFAULT_PERMISSION_CLASSES"] = [
    "rest_framework.permissions.IsAuthenticated"
]
```

Hinweis: It is desirable to lock down API access as well, as shown in the above example.

Siehe auch:

`REQUIRE_LOGIN`

2.16.60 LOGIN_REQUIRED_URLS_EXCEPTIONS

List of exceptions for *LOGIN_REQUIRED_URLS*. If not specified, users are allowed to access the sign in page.

Some of exceptions you might want to include:

```
LOGIN_REQUIRED_URLS_EXCEPTIONS = (
    r"/accounts/(.*)$",  # Required for sign in
    r"/static/(.*)$",  # Required for development mode
    r"/widgets/(.*)$",  # Allowing public access to widgets
    r"/data/(.*)$",  # Allowing public access to data exports
    r"/hooks/(.*)$",  # Allowing public access to notification hooks
    r"/api/(.*)$",  # Allowing access to API
    r"/js/i18n/$",  # JavaScript localization
)
```

2.16.61 MATOMO_SITE_ID

ID of a site in Matomo (formerly Piwik) you want to track.

Bemerkung: This integration does not support the Matomo Tag Manager.

Siehe auch:

MATOMO_URL

2.16.62 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.

Hinweis: This integration does not support the Matomo Tag Manager.

For example:

```
MATOMO_SITE_ID = 1
MATOMO_URL = "https://example.matomo.cloud/"
```

Siehe auch:

MATOMO_SITE_ID

2.16.63 NEARBY_MESSAGES

How many strings to show around the currently translated string. This is just a default value, users can adjust this in *Benutzerprofil*.

2.16.64 DEFAULT_PAGE_LIMIT

Neu in Version 4.7.

Standardanzahl der Elemente, die bei aktivem Seitenumbruch angezeigt werden.

2.16.65 PAGURE_CREDENTIALS

Neu in Version 4.3.2.

List for credentials for Pagure servers.

Hinweis: 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",
    },
}
```

2.16.66 PAGURE_USERNAME

Neu in Version 4.3.2.

Pagure-Benutzername, der verwendet wird, um Merge Requests für Übersetzungsaktualisierungen zu senden.

Siehe auch:

`PAGURE_CREDENTIALS`, *Pagure Merge Requests*

2.16.67 PAGURE_TOKEN

Neu in Version 4.3.2.

Pagure personal access token used to make API calls to send merge requests for translation updates.

Siehe auch:

`PAGURE_CREDENTIALS`, *Pagure Merge Requests*, Pagure API

2.16.68 PRIVACY_URL

Neu in Version 4.8.1.

URL, unter der Ihre Weblate-Instanz ihre Datenschutzrichtlinie anzeigt.

Hinweis: Useful if you host your legal documents outside Weblate for embedding them inside Weblate, please check *Rechtliche Grundlagen* for details.

Example:

```
PRIVACY_URL = "https://weblate.org/terms/"
```

Siehe auch:

LEGAL_URL

2.16.69 PROJECT_BACKUP_KEEP_COUNT

Neu in Version 4.14.

Defines how many backups per project are kept on the server. It defaults to 3.

Siehe auch:

Project level backups

2.16.70 PROJECT_BACKUP_KEEP_DAYS

Neu in Version 4.14.

Defines how long the project backups will be kept on the server. Defaults to 30 days.

Siehe auch:

Project level backups

2.16.71 RATELIMIT_ATTEMPTS

Neu in Version 3.2.

Maximum number of authentication attempts before rate limiting is applied.

Defaults to 5.

Siehe auch:

Rate limiting, *RATELIMIT_WINDOW*, *RATELIMIT_LOCKOUT*

2.16.72 RATELIMIT_WINDOW

Neu in Version 3.2.

How long authentication is accepted after rate limiting applies.

An amount of seconds defaulting to 300 (5 minutes).

Siehe auch:

Rate limiting, *RATELIMIT_ATTEMPTS*, *RATELIMIT_LOCKOUT*

2.16.73 RATELIMIT_LOCKOUT

Neu in Version 3.2.

How long authentication is locked after rate limiting applies.

An amount of seconds defaulting to 600 (10 minutes).

Siehe auch:

Rate limiting, *RATELIMIT_ATTEMPTS*, *RATELIMIT_WINDOW*

2.16.74 REGISTRATION_ALLOW_BACKENDS

Neu in Version 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.

Example:

```
REGISTRATION_ALLOW_BACKENDS = ["azuread-oauth2", "azuread-tenant-oauth2"]
```

Hinweis: The backend names match names used in URL for authentication.

Siehe auch:

REGISTRATION_OPEN, *Legitimierung*

2.16.75 REGISTRATION_CAPTCHA

A value of either `True` or `False` indicating whether registration of new accounts is protected by CAPTCHA. This setting is optional, and a default of `True` will be assumed if it is not supplied.

If turned on, a CAPTCHA is added to all pages where a users enters their e-mail address:

- New account registration.
- Password recovery.
- Adding e-mail to an account.
- Contact form for users that are not signed in.

2.16.76 REGISTRATION_EMAIL_MATCH

Neu in Version 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$"
```

2.16.77 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*).

Bemerkung: If using third-party authentication methods such as *LDAP-Authentifizierung*, it just hides the registration form, but new users might still be able to sign in and create accounts.

Siehe auch:

REGISTRATION_ALLOW_BACKENDS, *REGISTRATION_EMAIL_MATCH*, *Legitimierung*

2.16.78 REPOSITORY_ALERT_THRESHOLD

Neu in Version 4.0.2.

Threshold for triggering an alert for outdated repositories, or ones that contain too many changes. Defaults to 25.

Siehe auch:

alerts

2.16.79 REQUIRE_LOGIN

Neu in Version 4.1.

This enables *LOGIN_REQUIRED_URLS* and configures REST framework to require authentication for all API endpoints.

Bemerkung: This is implemented in the *Sample configuration*. For Docker, use *WEBLATE_REQUIRE_LOGIN*.

2.16.80 SENTRY_DSN

Neu in Version 3.9.

Sentry DSN to use for *Collecting error reports*.

Siehe auch:

Django integration for Sentry

2.16.81 SESSION_COOKIE_AGE_AUTHENTICATED

Neu in Version 4.3.

Set session expiry for authenticated users. This complements `SESSION_COOKIE_AGE` which is used for unauthenticated users.

Siehe auch:

`SESSION_COOKIE_AGE`

2.16.82 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.

2.16.83 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.

Beispiele:

```
# Production site with domain name
SITE_DOMAIN = "weblate.example.com"

# Local development with IP address and port
SITE_DOMAIN = "127.0.0.1:8000"
```

Bemerkung: 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*.

Hinweis: On a Docker container, the site domain is configured through *WEBLATE_ALLOWED_HOSTS*.

Siehe auch:

Set correct site domain, *Allowed hosts setup*, *Correctly configure HTTPS* *WEBLATE_SITE_DOMAIN*, *ENABLE_HTTPS*

2.16.84 SITE_TITLE

Site title to be used for the website and sent e-mails.

2.16.85 SPECIAL_CHARS

Additional characters to include in the visual keyboard, *Visual keyboard*.

The default value is:

```
SPECIAL_CHARS = ("\t", "\n", "\u00a0", "…")
```

2.16.86 SINGLE_PROJECT

Neu in Version 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.

Geändert in Version 3.11: The setting now also accepts a project slug, to force displaying that single project.

Example:

```
SINGLE_PROJECT = "test"
```

2.16.87 SSH_EXTRA_ARGS

Neu in Version 4.9.

Allows to add custom parameters when Weblate is invoking SSH. This is useful when connecting to servers using legacy encryption or other non-standard features.

For example when SSH connection in Weblate fails with *Unable to negotiate with legacyhost: no matching key exchange method found. Their offer: diffie-hellman-group1-sha1*, you can enable that using:

```
SSH_EXTRA_ARGS = "-oKexAlgorithms=+diffie-hellman-group1-sha1"
```

Hinweis: The string is evaluated by shell, so make sure to quote any whitespace and special characters.

Siehe auch:

OpenSSH Legacy Options

2.16.88 STATUS_URL

The URL where your Weblate instance reports its status.

2.16.89 SUGGESTION_CLEANUP_DAYS

Neu in Version 3.2.1.

Automatically deletes suggestions after a given number of days. Defaults to `None`, meaning no deletions.

2.16.90 UPDATE_LANGUAGES

Neu in Version 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`.

Warnung: The languages display might become inconsistent with this. Weblate language definitions extend over time and it will not display language code for the defined languages.

Siehe auch:

Integrierte Sprachdefinitionen

2.16.91 URL_PREFIX

This setting allows you to run Weblate under some path (otherwise it relies on being run from the webserver root).

Bemerkung: 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`.

Hinweis: The prefix should start with a `/`.

Example:

```
URL_PREFIX = "/translations"
```

Bemerkung: This setting does not work with Django's built-in server, you would have to adjust `urls.py` to contain this prefix.

2.16.92 VCS_BACKENDS

Configuration of available VCS backends.

Bemerkung: Weblate tries to use all supported back-ends you have the tools for.

Hinweis: You can limit choices or add custom VCS back-ends by using this.

```
VCS_BACKENDS = ("weblate.vcs.git.GitRepository",)
```

Siehe auch:

Integration der Versionsverwaltung

2.16.93 VCS_CLONE_DEPTH

Neu in Version 3.10.2.

Configures how deep cloning of repositories Weblate should do.

Bemerkung: 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 *Erweiterungen*), you might want to increase the depth or turn off shallow clones completely by setting this to 0.

Hinweis: In case you get `fatal: protocol error: expected old/new/ref, got 'shallow <commit hash>'` error when pushing from Weblate, turn off shallow clones completely by setting:

```
VCS_CLONE_DEPTH = 0
```

2.16.94 WEBLATE_ADDONS

List of add-ons available for use. To use them, they have to be enabled for a given translation component. By default this includes all built-in add-ons, when extending the list you will probably want to keep existing ones enabled, for example:

```
WEBLATE_ADDONS = (
    # Built-in add-ons
    "weblate.addons.gettext.GenerateMoAddon",
    "weblate.addons.gettext.UpdateLinguasAddon",
    "weblate.addons.gettext.UpdateConfigureAddon",
    "weblate.addons.gettext.MsgmergeAddon",
    "weblate.addons.gettext.GettextCustomizeAddon",
    "weblate.addons.gettext.GettextAuthorComments",
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    "weblate.addons.cleanup.CleanupAddon",
    "weblate.addons.consistency.LangaugeConsistencyAddon",
    "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",
)
```

Bemerkung: Removing the add-on from the list does not uninstall it from the components. Weblate will crash in that case. Please uninstall add-on from all components prior to removing it from this list.

Siehe auch:

Erweiterungen, `DEFAULT_ADDONS`

2.16.95 WEBLATE_EXPORTERS

Neu in Version 4.2.

List of a available exporters offering downloading translations or glossaries in various file formats.

Siehe auch:

Supported file formats

2.16.96 WEBLATE_FORMATS

Neu in Version 3.0.

List of file formats available for use.

Bemerkung: The default list already has the common formats.

Siehe auch:

Supported file formats

2.16.97 WEBLATE_MACHINERY

Neu in Version 4.13.

List of machinery services available for use.

Siehe auch:

Configuring automatic suggestions

2.16.98 WEBLATE_GPG_IDENTITY

Neu in Version 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.

Siehe auch:

Signing Git commits with GnuPG

2.16.99 WEBSITE_REQUIRED

Defines whether *Projektseite* has to be specified when creating a project. Turned on by default as that suits public server setups.

2.17 Sample configuration

The following example is shipped as `weblate/settings_example.py` with Weblate:

```
#
# Copyright © 2012-2022 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <https://www.gnu.org/licenses/>.
#


import os
import platform
from logging.handlers import SysLogHandler

# Title of site to use
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
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,
        # Disable server-side cursors, might be needed with pgbouncer
        "DISABLE_SERVER_SIDE_CURSORS": False,
    }
}

# Data directory, you can use following for the development purposes:
# os.path.join(os.path.dirname(os.path.dirname(os.path.abspath(__file__))), "data")
DATA_DIR = "/home/weblate/data"

# Local time zone for this installation. Choices can be found here:
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
# 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", "ةيبرعلا"),
    ("az", "Azərbaycan"),
    ("be", "Беларуская"),
    ("be@latin", "Biełaruskaja"),
    ("bg", "Български"),
    ("br", "Brezhoneg"),
    ("ca", "Català"),
    ("cs", "Čeština"),
    ("da", "Dansk"),
    ("de", "Deutsch"),
    ("en", "English"),
    ("el", "Ελληνικά"),
    ("en-gb", "English (United Kingdom)"),
    ("es", "Español"),
    ("fi", "Suomi"),
    ("fr", "Français"),
    ("gl", "Galego"),
    ("he", "עברית"),
    ("hu", "Magyar"),
    ("hr", "Hrvatski"),
    ("id", "Indonesia"),
    ("is", "Íslenska"),
    ("it", "Italiano"),
    ("ja", "□□□"),
    ("kab", "Taqbaylit"),
    ("kk", "Қазақ тілі"),
    ("ko", "□□□"),
    ("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", "□□□"),
    ("tr", "Türkçe"),
    ("uk", "Українська"),
    ("zh-hans", "□□□□"),
    ("zh-hant", "□□□□"),
)

SITE_ID = 1

# If you set this to False, Django will make some optimizations so as not
# to load the internationalization machinery.
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
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-generate-secret-key
SECRET_KEY = ""

TEMPLATES = [
    {
        "BACKEND": "django.template.backends.django.DjangoTemplates",
        "OPTIONS": {
            "context_processors": [
                "django.contrib.auth.context_processors.auth",
                "django.template.context_processors.debug",
                "django.template.context_processors.i18n",
                "django.template.context_processors.request",
                "django.template.context_processors.csrf",
                "django.contrib.messages.context_processors.messages",
                "weblate.trans.context_processors.weblate_context",
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
            ],
        },
        "APP_DIRS": True,
    }
]


# GitHub username and token for sending pull requests.
# Please see the documentation for more details.
GITHUB_USERNAME = None
GITHUB_TOKEN = None

# GitLab username and token for sending merge requests.
# Please see the documentation for more details.
GITLAB_USERNAME = None
GITLAB_TOKEN = None

# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.email.EmailAuth",
    # "social_core.backends.google.GoogleOAuth2",
    # "social_core.backends.github.GithubOAuth2",
    # "social_core.backends.bitbucket.BitbucketOAuth2",
    # "social_core.backends.suse.OpenSUSEOpenId",
    # "social_core.backends.ubuntu.UbuntuOpenId",
    # "social_core.backends.fedora.FedoraOpenId",
    # "social_core.backends.facebook.FacebookOAuth2",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Custom user model
AUTH_USER_MODEL = "weblate_auth.User"

# Social auth backends setup
SOCIAL_AUTH_GITHUB_KEY = ""
SOCIAL_AUTH_GITHUB_SECRET = ""
SOCIAL_AUTH_GITHUB_SCOPE = ["user:email"]

SOCIAL_AUTH_GITHUB_ORG_KEY = ""
SOCIAL_AUTH_GITHUB_ORG_SECRET = ""
SOCIAL_AUTH_GITHUB_ORG_NAME = ""

SOCIAL_AUTH_GITHUB_TEAM_KEY = ""
SOCIAL_AUTH_GITHUB_TEAM_SECRET = ""
SOCIAL_AUTH_GITHUB_TEAM_ID = ""

SOCIAL_AUTH_BITBUCKET_OAUTH2_KEY = ""
SOCIAL_AUTH_BITBUCKET_OAUTH2_SECRET = ""
SOCIAL_AUTH_BITBUCKET_OAUTH2_VERIFIED_EMAILS_ONLY = True

SOCIAL_AUTH_FACEBOOK_KEY = ""
SOCIAL_AUTH_FACEBOOK_SECRET = ""
SOCIAL_AUTH_FACEBOOK_SCOPE = ["email", "public_profile"]
SOCIAL_AUTH_FACEBOOK_PROFILE_EXTRA_PARAMS = {"fields": "id,name,email"}

SOCIAL_AUTH_GOOGLE_OAUTH2_KEY = ""
SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET = ""

# Social auth settings
SOCIAL_AUTH_PIPELINE = (
    "social_core.pipeline.social_auth.social_details",
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    "social_core.pipeline.social_auth.social_uid",
    "social_core.pipeline.social_auth.auth_allowed",
    "social_core.pipeline.social_auth.social_user",
    "weblate.accounts.pipeline.store_params",
    "weblate.accounts.pipeline.verify_open",
    "social_core.pipeline.user.get_username",
    "weblate.accounts.pipeline.require_email",
    "social_core.pipeline.mail.mail_validation",
    "weblate.accounts.pipeline.revoke_mail_code",
    "weblate.accounts.pipeline.ensure_valid",
    "weblate.accounts.pipeline.remove_account",
    "social_core.pipeline.social_auth.associate_by_email",
    "weblate.accounts.pipeline.reauthenticate",
    "weblate.accounts.pipeline.verify_username",
    "social_core.pipeline.user.create_user",
    "social_core.pipeline.social_auth.associate_user",
    "social_core.pipeline.social_auth.load_extra_data",
    "weblate.accounts.pipeline.cleanup_next",
    "weblate.accounts.pipeline.user_full_name",
    "weblate.accounts.pipeline.store_email",
    "weblate.accounts.pipeline.notify_connect",
    "weblate.accounts.pipeline.password_reset",
)
SOCIAL_AUTH_DISCONNECT_PIPELINE = (
    "social_core.pipeline.disconnect.allowed_to_disconnect",
    "social_core.pipeline.disconnect.get_entries",
    "social_core.pipeline.disconnect.revoke_tokens",
    "weblate.accounts.pipeline.cycle_session",
    "weblate.accounts.pipeline.adjust_primary_mail",
    "weblate.accounts.pipeline.notify_disconnect",
    "social_core.pipeline.disconnect.disconnect",
    "weblate.accounts.pipeline.cleanup_next",
)

# Custom authentication strategy
SOCIAL_AUTH_STRATEGY = "weblate.accounts.strategy.WeblateStrategy"

# Raise exceptions so that we can handle them later
SOCIAL_AUTH_RAISE_EXCEPTIONS = True

SOCIAL_AUTH_EMAIL_VALIDATION_FUNCTION = "weblate.accounts.pipeline.send_validation"
SOCIAL_AUTH_EMAIL_VALIDATION_URL = f"{URL_PREFIX}/accounts/email-sent/"
SOCIAL_AUTH_LOGIN_ERROR_URL = f"{URL_PREFIX}/accounts/login/"
SOCIAL_AUTH_EMAIL_FORM_URL = f"{URL_PREFIX}/accounts/email/"
SOCIAL_AUTH_NEW_ASSOCIATION_REDIRECT_URL = f"{URL_PREFIX}/accounts/profile/#account
↪"
SOCIAL_AUTH_PROTECTED_USER_FIELDS = ("email",)
SOCIAL_AUTH_SLUGIFY_USERNAMES = True
SOCIAL_AUTH_SLUGIFY_FUNCTION = "weblate.accounts.pipeline.slugify_username"

# Password validation configuration
AUTH_PASSWORD_VALIDATORS = [
    {
        "NAME": "django.contrib.auth.password_validation.
↪UserAttributeSimilarityValidator"  # noqa: E501, pylint: disable=line-too-long
    },
    {
        "NAME": "django.contrib.auth.password_validation.MinimumLengthValidator",
        "OPTIONS": {"min_length": 10},
    },
    {"NAME": "django.contrib.auth.password_validation.CommonPasswordValidator"},
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    {"NAME": "django.contrib.auth.password_validation.NumericPasswordValidator"},
    {"NAME": "weblate.accounts.password_validation.CharsPasswordValidator"},
    {"NAME": "weblate.accounts.password_validation.PastPasswordsValidator"},
    # Optional password strength validation by django-zxcvbn-password
    # {
    #     "NAME": "zxcvbn_password.ZXCVBNValidator",
    #     "OPTIONS": {
    #         "min_score": 3,
    #         "user_attributes": ("username", "email", "full_name")
    #     }
    # },
]

# Password hashing (prefer Argon)
PASSWORD_HASHERS = [
    "django.contrib.auth.hashers.Argon2PasswordHasher",
    "django.contrib.auth.hashers.PBKDF2PasswordHasher",
    "django.contrib.auth.hashers.PBKDF2SHA1PasswordHasher",
    "django.contrib.auth.hashers.BCryptSHA256PasswordHasher",
]

# Allow new user registrations
REGISTRATION_OPEN = True

# Shortcut for login required setting
REQUIRE_LOGIN = False

# Middleware
MIDDLEWARE = [
    "weblate.middleware.RedirectMiddleware",
    "weblate.middleware.ProxyMiddleware",
    "django.middleware.security.SecurityMiddleware",
    "django.contrib.sessions.middleware.SessionMiddleware",
    "django.middleware.csrf.CsrfViewMiddleware",
    "weblate.accounts.middleware.AuthenticationMiddleware",
    "django.contrib.messages.middleware.MessageMiddleware",
    "django.middleware.clickjacking.XFrameOptionsMiddleware",
    "social_django.middleware.SocialAuthExceptionMiddleware",
    "weblate.accounts.middleware.RequireLoginMiddleware",
    "weblate.api.middleware.ThrottlingMiddleware",
    "weblate.middleware.SecurityMiddleware",
    "weblate.wladmin.middleware.ManageMiddleware",
]

ROOT_URLCONF = "weblate.urls"

# Django and Weblate apps
INSTALLED_APPS = [
    # Weblate apps on top to override Django locales and templates
    "weblate.addons",
    "weblate.auth",
    "weblate.checks",
    "weblate.formats",
    "weblate.glossary",
    "weblate.machinery",
    "weblate.trans",
    "weblate.lang",
    "weblate_language_data",
    "weblate.memory",
    "weblate.screenshots",
    "weblate.fonts",
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    "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.
LOGGING = {
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    "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,
        },
        # Logging database queries
        # "django.db.backends": {
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
        #     "handlers": [DEFAULT_LOG],
        #     "level": "DEBUG",
        # },
        "weblate": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
        # Logging VCS operations
        "weblate.vcs": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
        # Python Social Auth
        "social": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
        # Django Authentication Using LDAP
        "django_auth_ldap": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
        # SAML IdP
        "djangosaml2idp": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
    },
}

# Remove syslog setup if it's not present
if not HAVE_SYSLOG:
    del LOGGING["handlers"]["syslog"]

# List of machine translations
MT_SERVICES = (
    #     "weblate.machinery.apertium.ApertiumAPYTranslation",
    #     "weblate.machinery.baidu.BaiduTranslation",
    #     "weblate.machinery.deepl.DeepLTranslation",
    #     "weblate.machinery.glosbe.GlosbeTranslation",
    #     "weblate.machinery.google.GoogleTranslation",
    #     "weblate.machinery.googlev3.GoogleV3Translation",
    #     "weblate.machinery.libretranslate.LibreTranslateTranslation",
    #     "weblate.machinery.microsoft.MicrosoftCognitiveTranslation",
    #     "weblate.machinery.microsoftterminology.MicrosoftTerminologyService",
    #     "weblate.machinery.modernmt.ModernMTTranslation",
    #     "weblate.machinery.mymemory.MyMemoryTranslation",
    #     "weblate.machinery.netease.NeteaseSightTranslation",
    #     "weblate.machinery.tmserver.AmagamaTranslation",
    #     "weblate.machinery.tmserver.TMServerTranslation",
    #     "weblate.machinery.yandex.YandexTranslation",
    #     "weblate.machinery.saptranslationhub.SAPTranslationHub",
    #     "weblate.machinery.youdao.YoudaoTranslation",
    "weblate.machinery.weblatetm.WeblateTranslation",
    "weblate.memory.machine.WeblateMemory",
)

# Machine translation API keys

# URL of the Apertium APy server
MT_APERTIUM_APY = None

# DeepL API key
MT_DEEPL_KEY = None

# LibreTranslate
MT_LIBRETRANSLATE_API_URL = None
MT_LIBRETRANSLATE_KEY = None

# Microsoft Cognitive Services Translator API, register at
# https://portal.azure.com/
MT_MICROSOFT_COGNITIVE_KEY = None
MT_MICROSOFT_REGION = None

# ModernMT
MT_MODERNMT_KEY = None
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
# 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:
# https://python-social-auth-docs.readthedocs.io/en/latest/configuration/settings.
↪html#processing-redirects-and-urlopen
SOCIAL_AUTH_REDIRECT_IS_HTTPS = ENABLE_HTTPS

# Make CSRF cookie HttpOnly, see documentation for more details:
# https://docs.djangoproject.com/en/1.11/ref/settings/#csrf-cookie-httponly
CSRF_COOKIE_HTTPONLY = True
CSRF_COOKIE_SECURE = ENABLE_HTTPS
# Store CSRF token in session
CSRF_USE_SESSIONS = True
# Customize CSRF failure view
CSRF_FAILURE_VIEW = "weblate.trans.views.error.csrf_failure"
SESSION_COOKIE_SECURE = ENABLE_HTTPS
SESSION_COOKIE_HTTPONLY = True
# SSL redirect
SECURE_SSL_REDIRECT = ENABLE_HTTPS
SECURE_SSL_HOST = SITE_DOMAIN
# Sent referrrer only for same origin links
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
SECURE_REFERRER_POLICY = "same-origin"
# SSL redirect URL exemption list
SECURE_REDIRECT_EXEMPT = (r"healthz/$",)  # Allowing HTTP access to health check
# Session cookie age (in seconds)
SESSION_COOKIE_AGE = 1000
SESSION_COOKIE_AGE_AUTHENTICATED = 1209600
SESSION_COOKIE_SAMESITE = "Lax"
# Increase allowed upload size
DATA_UPLOAD_MAX_MEMORY_SIZE = 50000000

# Apply session coookie settings to language cookie as ewll
LANGUAGE_COOKIE_SECURE = SESSION_COOKIE_SECURE
LANGUAGE_COOKIE_HTTPONLY = SESSION_COOKIE_HTTPONLY
LANGUAGE_COOKIE_AGE = SESSION_COOKIE_AGE_AUTHENTICATED * 10
LANGUAGE_COOKIE_SAMESITE = SESSION_COOKIE_SAMESITE

# Some security headers
SECURE_BROWSER_XSS_FILTER = True
X_FRAME_OPTIONS = "DENY"
SECURE_CONTENT_TYPE_NOSNIFF = True

# Optionally enable HSTS
SECURE_HSTS_SECONDS = 31536000 if ENABLE_HTTPS else 0
SECURE_HSTS_PRELOAD = ENABLE_HTTPS
SECURE_HSTS_INCLUDE_SUBDOMAINS = ENABLE_HTTPS

# 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
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
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.icu.ICUMessageFormatCheck",
#     "weblate.checks.icu.ICUSourceCheck",
#     "weblate.checks.qt.QtFormatCheck",
#     "weblate.checks.qt.QtPluralCheck",
#     "weblate.checks.ruby.RubyFormatCheck",
#     "weblate.checks.consistency.PluralsCheck",
#     "weblate.checks.consistency.SamePluralsCheck",
#     "weblate.checks.consistency.ConsistencyCheck",
#     "weblate.checks.consistency.TranslatedCheck",
#     "weblate.checks.chars.EscapedNewlineCountingCheck",
#     "weblate.checks.chars.NewLineCountCheck",
#     "weblate.checks.markup.BBCodeCheck",
#     "weblate.checks.chars.ZeroWidthSpaceCheck",
#     "weblate.checks.render.MaxSizeCheck",
#     "weblate.checks.markup.XMLValidityCheck",
#     "weblate.checks.markup.XMLTagsCheck",
#     "weblate.checks.markup.MarkdownRefLinkCheck",
#     "weblate.checks.markup.MarkdownLinkCheck",
#     "weblate.checks.markup.MarkdownSyntaxCheck",
#     "weblate.checks.markup.URLCheck",
#     "weblate.checks.markup.SafeHTMLCheck",
#     "weblate.checks.placeholders.PlaceholderCheck",
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
#     "weblate.checks.placeholders.RegexCheck",
#     "weblate.checks.duplicate.DuplicateCheck",
#     "weblate.checks.source.OptionalPluralCheck",
#     "weblate.checks.source.EllipsisCheck",
#     "weblate.checks.source.MultipleFailingCheck",
#     "weblate.checks.source.LongUntranslatedCheck",
#     "weblate.checks.format.MultipleUnnamedFormatsCheck",
#     "weblate.checks.glossary.GlossaryCheck",
# )

# List of automatic fixups
# AUTOFIX_LIST = (
#     "weblate.trans.autofixes.whitespace.SameBookendingWhitespace",
#     "weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis",
#     "weblate.trans.autofixes.chars.RemoveZeroSpace",
#     "weblate.trans.autofixes.chars.RemoveControlChars",
# )

# List of enabled addons
# WEBLATE_ADDONS = (
#     "weblate.addons.gettext.GenerateMoAddon",
#     "weblate.addons.gettext.UpdateLinguasAddon",
#     "weblate.addons.gettext.UpdateConfigureAddon",
#     "weblate.addons.gettext.MsgmergeAddon",
#     "weblate.addons.gettext.GettextCustomizeAddon",
#     "weblate.addons.gettext.GettextAuthorComments",
#     "weblate.addons.cleanup.CleanupAddon",
#     "weblate.addons.cleanup.RemoveBlankAddon",
#     "weblate.addons.consistency.LangaugeConsistencyAddon",
#     "weblate.addons.discovery.DiscoveryAddon",
#     "weblate.addons.autotranslate.AutoTranslateAddon",
#     "weblate.addons.flags.SourceEditAddon",
#     "weblate.addons.flags.TargetEditAddon",
#     "weblate.addons.flags.SameEditAddon",
#     "weblate.addons.flags.BulkEditAddon",
#     "weblate.addons.generate.GenerateFileAddon",
#     "weblate.addons.generate.PseudolocaleAddon",
#     "weblate.addons.generate.PrefillAddon",
#     "weblate.addons.json.JSONCustomizeAddon",
#     "weblate.addons.properties.PropertiesSortAddon",
#     "weblate.addons.git.GitSquashAddon",
#     "weblate.addons.removal.RemoveComments",
#     "weblate.addons.removal.RemoveSuggestions",
#     "weblate.addons.resx.ResxUpdateAddon",
#     "weblate.addons.yaml.YAMLCustomizeAddon",
#     "weblate.addons.cdn.CDNJSAddon",
# )

# E-mail address that error messages come from.
SERVER_EMAIL = "noreply@example.com"

# Default email address to use for various automated correspondence from
# the site managers. Used for registration emails.
DEFAULT_FROM_EMAIL = "noreply@example.com"

# List of URLs your site is supposed to serve
ALLOWED_HOSTS = ["*"]

# Configuration for caching
CACHES = {
    "default": {
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
        "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},
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    {"fonts_cdn_url": FONTS_CDN_URL, "STATIC_URL": STATIC_URL, "LANGUAGE_BIDI":␣
↪False},
]

# Require login for all URLs
if REQUIRE_LOGIN:
    LOGIN_REQUIRED_URLS = (r"/(.*)$",)

# In such case you will want to include some of the exceptions
# LOGIN_REQUIRED_URLS_EXCEPTIONS = (
#    rf"{URL_PREFIX}/accounts/(.*)$",  # Required for login
#    rf"{URL_PREFIX}/admin/login/(.*)$",  # Required for admin login
#    rf"{URL_PREFIX}/static/(.*)$",  # Required for development mode
#    rf"{URL_PREFIX}/widgets/(.*)$",  # Allowing public access to widgets
#    rf"{URL_PREFIX}/data/(.*)$",  # Allowing public access to data exports
#    rf"{URL_PREFIX}/hooks/(.*)$",  # Allowing public access to notification hooks
#    rf"{URL_PREFIX}/healthz/$",  # Allowing public access to health check
#    rf"{URL_PREFIX}/api/(.*)$",  # Allowing access to API
#    rf"{URL_PREFIX}/js/i18n/$",  # JavaScript localization
#    rf"{URL_PREFIX}/contact/$",  # Optional for contact form
#    rf"{URL_PREFIX}/legal/(.*)$",  # Optional for legal app
#    rf"{URL_PREFIX}/avatar/(.*)$",  # Optional for avatars
# )

# Silence some of the Django system checks
SILENCED_SYSTEM_CHECKS = [
    # We have modified django.contrib.auth.middleware.AuthenticationMiddleware
    # as weblate.accounts.middleware.AuthenticationMiddleware
    "admin.E408"
]

# Celery worker configuration for testing
# CELERY_TASK_ALWAYS_EAGER = True
# CELERY_BROKER_URL = "memory://"
# CELERY_TASK_EAGER_PROPAGATES = True
# Celery worker configuration for production
CELERY_TASK_ALWAYS_EAGER = False
CELERY_BROKER_URL = "redis://localhost:6379"
CELERY_RESULT_BACKEND = CELERY_BROKER_URL

# Celery settings, it is not recommended to change these
CELERY_WORKER_MAX_MEMORY_PER_CHILD = 200000
CELERY_BEAT_SCHEDULE_FILENAME = os.path.join(DATA_DIR, "celery", "beat-schedule")
CELERY_TASK_ROUTES = {
    "weblate.trans.tasks.auto_translate*": {"queue": "translate"},
    "weblate.accounts.tasks.notify_*": {"queue": "notify"},
    "weblate.accounts.tasks.send_mails": {"queue": "notify"},
    "weblate.utils.tasks.settings_backup": {"queue": "backup"},
    "weblate.utils.tasks.database_backup": {"queue": "backup"},
    "weblate.wladmin.tasks.backup": {"queue": "backup"},
    "weblate.wladmin.tasks.backup_service": {"queue": "backup"},
    "weblate.memory.tasks.*": {"queue": "memory"},
}

# Enable plain database backups
DATABASE_BACKUP = "plain"

# Enable auto updating
AUTO_UPDATE = False

# PGP commits signing
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
WEBLATE_GPG_IDENTITY = None

# Third party services integration
MATOMO_SITE_ID = None
MATOMO_URL = None
GOOGLE_ANALYTICS_ID = None
SENTRY_DSN = None
SENTRY_ENVIRONMENT = SITE_DOMAIN
AKISMET_API_KEY = None
```

2.18 Management commands

Bemerkung: Running management commands under a different user than the one running your webserver can result in files getting wrong permissions, please check *Filesystem permissions* for more details.

You will find basic management commands (available as `./manage.py` in the Django sources, or as an extended set in a script called **`weblate`** installable atop Weblate).

2.18.1 Invoking management commands

As mentioned before, invocation depends on how you installed Weblate.

If using virtualenv for Weblate, you can either specify the full path to **`weblate`**, or activate the virtualenv prior to invoking it:

```
# Direct invocation
~/weblate-env/bin/weblate

# Activating virtualenv adds it to search path
. ~/weblate-env/bin/activate
weblate
```

If you are using source code directly (either from a tarball or Git checkout), the management script is `./manage.py` available in the Weblate sources. To run it:

```
python ./manage.py list_versions
```

If you've installed Weblate using the pip installer, or by using the `./setup.py` script, the **`weblate`** is installed to your path (or virtualenv path), from where you can use it to control Weblate:

```
weblate list_versions
```

For the Docker image, the script is installed like above, and you can run it using **`docker exec`**:

```
docker exec --user weblate <container> weblate list_versions
```

For **`docker-compose`** the process is similar, you just have to use **`docker-compose exec`**:

```
docker-compose exec --user weblate weblate weblate list_versions
```

In case you need to pass it a file, you can temporary add a volume:

```
docker-compose exec --user weblate /tmp:/tmp weblate weblate importusers /tmp/
↪users.json
```

Siehe auch:

Installing using Docker, *Installing on Debian and Ubuntu*, *Installing on SUSE and openSUSE*, *Installing on RedHat, Fedora and CentOS*, *Installieren aus Quellen*

2.18.2 add_suggestions

weblate add_suggestions <project> <component> <language> <file>

Neu in Version 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).

Example:

```
weblate --author michal@cihar.com add_suggestions weblate application cs /tmp/
↪suggestions-cs.po
```

2.18.3 auto_translate

weblate auto_translate <project> <component> <language>

Neu in Version 2.5.

Geändert in Version 4.6: Parameter für Übersetzungsmodus hinzugefügt.

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 *Inkonsistent*).

--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

Geben Sie den Übersetzungsmodus an, Standard ist `translate`, aber `fuzzy` oder `suggest` können verwendet werden.

Example:

```
weblate auto_translate --user nijel --inconsistent --source weblate/application␣
↪weblate website cs
```

Siehe auch:

Automatische Übersetzung

2.18.4 celery_queues

weblate celery_queues

Neu in Version 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.

Bemerkung: 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*.

Siehe auch:

Running maintenance tasks, `COMMIT_PENDING_HOURS`

2.18.8 cleanuptrans

weblate cleanuptrans

Cleans up orphaned checks and translation suggestions. There is normally no need to run this manually, as the cleanups happen automatically in the background.

Siehe auch:

Running maintenance tasks

2.18.9 cleanup_ssh_keys

weblate cleanup_ssh_keys

Neu in Version 4.9.1.

Performs cleanup of stored SSH host keys:

- Removes deprecated RSA keys for GitHub which might cause issues connecting to GitHub.
- Removes duplicate entries in host keys.

Siehe auch:

SSH repositories

2.18.10 createadmin

weblate createadmin

Creates an `admin` account with a random password, unless it is specified.

--password `PASSWORD`
: Provides a password on the command-line, to not generate a random one.

--no-password
: Do not set password, this can be useful with *–update*.

--username `USERNAME`
: Use the given name instead of `admin`.

--email `USER@EXAMPLE.COM`
: Specify the admin e-mail address.

--name
: Specify the admin name (visible).

--update
: Update the existing user (you can use this to change passwords).

Geändert in Version 2.9: Added parameters `--username`, `--email`, `--name` and `--update`.

2.18.11 dump_memory

weblate dump_memory

Neu in Version 2.20.

Export a JSON file containing Weblate Translation Memory content.

Siehe auch:

Übersetzungsspeicher, *Weblate-Übersetzungsspeicher-Schema*

2.18.12 dumpuserdata

weblate dumpuserdata <file.json>

Dumps userdata to a file for later use by `importuserdata`.

Hinweis: This comes in handy when migrating or merging Weblate instances.

2.18.13 import_demo

weblate import_demo

Neu in Version 4.1.

Creates a demo project with components based on <https://github.com/WeblateOrg/demo>. Make sure the celery tasks are running before running this command.

This can be useful when developing Weblate.

2.18.14 import_json

weblate import_json <json-file>

Neu in Version 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.

Geändert in Version 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"
  }
]
```

Siehe auch:

import_memory

2.18.15 import_memory

weblate import_memory <file>

Neu in Version 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.

Siehe auch:

Übersetzungsspeicher, *Weblate-Übersetzungsspeicher-Schema*

2.18.16 import_project

weblate import_project <project> <gitrepo> <branch> <filemask>

Geändert in Version 3.0: The import_project command is now based on the *Komponentenerkennung* add-on, leading to some changes in behavior and what parameters are accepted.

Batch imports components into project based on the file mask.

<project> names an existing project, into which the components are to be imported.

The *<gitrepo>* defines the Git repository URL to use, and *<branch>* signifies the Git branch. To import additional translation components from an existing Weblate component, use a *weblate://<project>/<component>* URL for the *<gitrepo>*.

The *<filemask>* defines file discovery for the repository. It can be either be made simple using wildcards, or it can use the full power of regular expressions.

The simple matching uses `**` for component name and `*` for language, for example: `**/*.po`

The regular expression has to contain groups named *component* and *language*. For example: `(?P<language>[^/]*)/(?P<component>[^-/]*)\.po`

The import matches existing components based on files and adds the ones that do not exist. It does not change already existing ones.

--name-template TEMPLATE

Customize the name of a component using Django template syntax.

For example: `Documentation: {{ component }}`

--base-file-template TEMPLATE

Customize the base file for monolingual translations.

For example: `{{ component }}/res/values/string.xml`

--new-base-template TEMPLATE

Customize the base file for addition of new translations.

For example: `{{ component }}/ts/en.ts`

--file-format FORMAT

You can also specify the file format to use (see *Supported file formats*), the default is auto-detection.

--language-regex REGEX

You can specify language filtering (see *Component configuration*) with this parameter. It has to be a valid regular expression.

--main-component

You can specify which component will be chosen as the main one—the one actually containing the VCS repository.

--license NAME

Specify the overall, project or component translation license.

--license-url URL

Specify the URL where the translation license is to be found.

--vcs NAME

In case you need to specify which version control system to use, you can do it here. The default version control is Git.

To give you some examples, let's try importing two projects.

First The Debian Handbook translations, where each language has separate a folder with the translations of each chapter:

```
weblate import_project \
    debian-handbook \
    git://anonscm.debian.org/debian-handbook/debian-handbook.git \
    squeeze/master \
    '*/**.po'
```

Then the Tanaguru tool, where the file format needs be specified, along with the base file template, and how all components and translations are located in single folder:

```
weblate import_project \
    --file-format=properties \
    --base-file-template=web-app/tgol-web-app/src/main/resources/i18n/%s-I18N.
↪properties \
    tanaguru \
    https://github.com/Tanaguru/Tanaguru \
    master \
    web-app/tgol-web-app/src/main/resources/i18n/**-I18N_*.properties
```

More complex example of parsing of filenames to get the correct component and language out of a filename like `src/security/Numerous_security_holes_in_0.10.1.de.po`:

```
weblate import_project \
    tails \
    git://git.tails.boum.org/tails master \
    'wiki/src/security/(?P<component>.*)\.(?P<language>[^.]*)\.po$'
```

Filtering only translations in a chosen language:

```
./manage import_project \
    --language-regex '^(cs|sk)$' \
    weblate \
    https://github.com/WeblateOrg/weblate.git \
    'weblate/locale/*/LC_MESSAGES/**.po'
```

Importing Sphinx documentation split to multiple files:

```
$ weblate import_project --name-template 'Documentation: %s' \
    --file-format po \
    project https://github.com/project/docs.git master \
    'docs/locale/*/LC_MESSAGES/**.po'
```

Importing Sphinx documentation split to multiple files and directories:

```
$ weblate import_project --name-template 'Directory 1: %s' \
    --file-format po \
    project https://github.com/project/docs.git master \
    'docs/locale/*/LC_MESSAGES/dir1/**.po'
$ weblate import_project --name-template 'Directory 2: %s' \
    --file-format po \
    project https://github.com/project/docs.git master \
    'docs/locale/*/LC_MESSAGES/dir2/**.po'
```

Siehe auch:

More detailed examples can be found in the starting chapter, alternatively you might want to use *import_json*.

2.18.17 importuserdata

weblate importuserdata <file.json>

Imports user data from a file created by *dumpuserdata*

2.18.18 importusers

weblate importusers --check <file.json>

Imports users from JSON dump of the Django auth_users database.

--check

With this option it will just check whether a given file can be imported and report possible conflicts arising from usernames or e-mails.

You can dump users from the existing Django installation using:

```
weblate dumpdata auth.User > users.json
```

2.18.19 install_addon

Neu in Version 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-kodierte Konfiguration einer Erweiterung.

--update
: Aktualisieren Sie die vorhandene Konfiguration der Erweiterung.

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 *Ausgabe von Gettext anpassen* for all components:

```
weblate install_addon --addon weblate.gettext.customize --config '{"width": -1}' --
↪update --all
```

Siehe auch:

Erweiterungen

2.18.20 list_languages

weblate list_languages <locale>

Lists supported languages in MediaWiki markup - language codes, English names and localized names.

This is used to generate <https://wiki.l10n.cz/Slovn%C3%ADk_s_n%C3%A1zvy_jazyk%C5%AF>.

2.18.21 list_translators

weblate list_translators <project|project/component>

Lists translators by contributed language for the given project:

```
[French]
Jean Dupont <jean.dupont@example.com>
[English]
John Doe <jd@example.com>
```

--language-code
: List names by language code instead of language name.

You can either define which project or component to use (for example `weblate/application`), or use `--all` to list translators from all existing components.

2.18.22 list_versions

weblate list_versions

Lists all Weblate dependencies and their versions.

2.18.23 loadpo

weblate loadpo <project|project/component>

Reloads translations from disk (for example in case you have done some updates in the VCS repository).

--force

Force update, even if the files should be up-to-date.

--lang LANGUAGE

Limit processing to a single language.

You can either define which project or component to update (for example `weblate/application`), or use `--all` to update all existing components.

Bemerkung: You seldom need to invoke this, Weblate will automatically load changed files for every VCS update. This is needed in case you manually changed an underlying Weblate VCS repository or in some special cases following an upgrade.

2.18.24 lock_translation

weblate lock_translation <project|project/component>

Prevents further translation of a component.

Hinweis: 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.

Siehe auch:

unlock_translation

2.18.25 move_language

weblate move_language source target

Neu in Version 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.

Example:

```
weblate move_language cze cs
```

After moving the content, you should check whether there is anything left (this is subject to race conditions when somebody updates the repository meanwhile) and remove the *(generated)* language.

2.18.26 pushgit

weblate pushgit <project|project/component>

Pushes committed changes to the upstream VCS repository.

--force-commit

Force commits any pending changes, prior to pushing.

You can either define which project or component to update (for example `weblate/application`), or use `--all` to update all existing components.

Bemerkung: Weblate pushes changes automatically if *Bei Commit gleichzeitig Pushen* in *Component configuration* is turned on, which is the default.

2.18.27 unlock_translation

weblate unlock_translation <project|project/component>

Unlocks a given component, making it available for translation.

Hinweis: 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.

Siehe auch:

lock_translation

2.18.28 setupgroups

weblate setupgroups

Configures default groups and optionally assigns all users to that default group.

--no-privs-update

Turns off automatic updating of existing groups (only adds new ones).

--no-projects-update

Prevents automatic updates of groups for existing projects. This allows adding newly added groups to existing projects, see *Projekt-Zugriffssteuerung*.

Siehe auch:

Liste der Berechtigungen und integrierten Rollen

2.18.29 setuplang

weblate setuplang

Updates list of defined languages in Weblate.

--no-update

Turns off automatic updates of existing languages (only adds new ones).

2.18.30 updatechecks

weblate updatechecks <project|project/component>

Updates all checks for all strings.

Hinweis: Useful for upgrades which do major changes to checks.

You can either define which project or component to update (for example `weblate/application`), or use `--all` to update all existing components.

2.18.31 updategit

weblate updategit <project|project/component>

Fetches remote VCS repositories and updates the internal cache.

You can either define which project or component to update (for example `weblate/application`), or use `--all` to update all existing components.

Bemerkung: Usually it is better to configure hooks in the repository to trigger *Benachrichtigungs-Hooks*, instead of regular polling by `updategit`.

2.19 Ankündigungen

Geändert in Version 4.0: In früheren Versionen wurde diese Funktion Whiteboard-Nachrichten genannt.

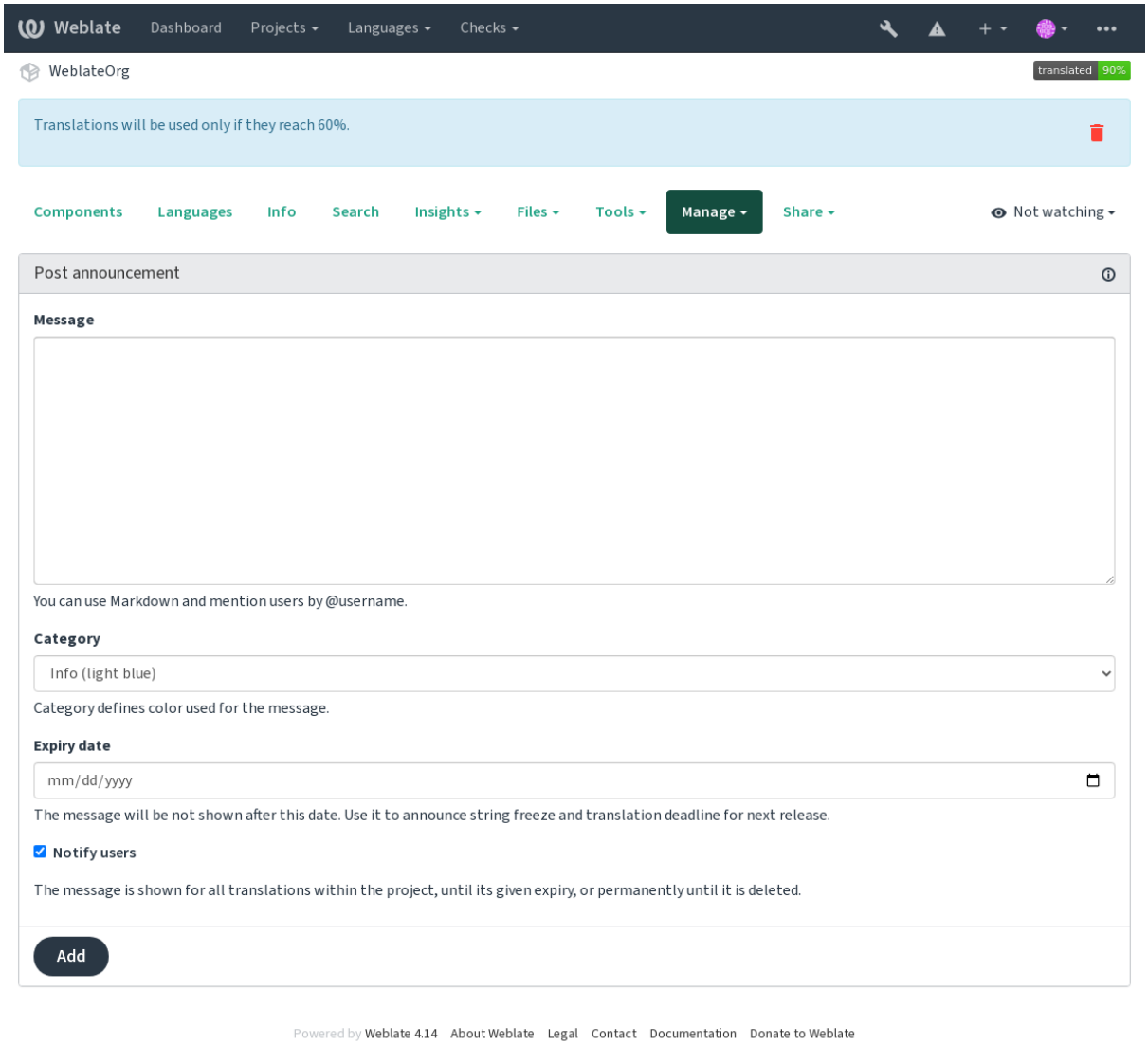
Provide info to your translators by posting announcements, site-wide, per project, component, or language.

Announce the purpose, deadlines, status, or specify targets for translation.

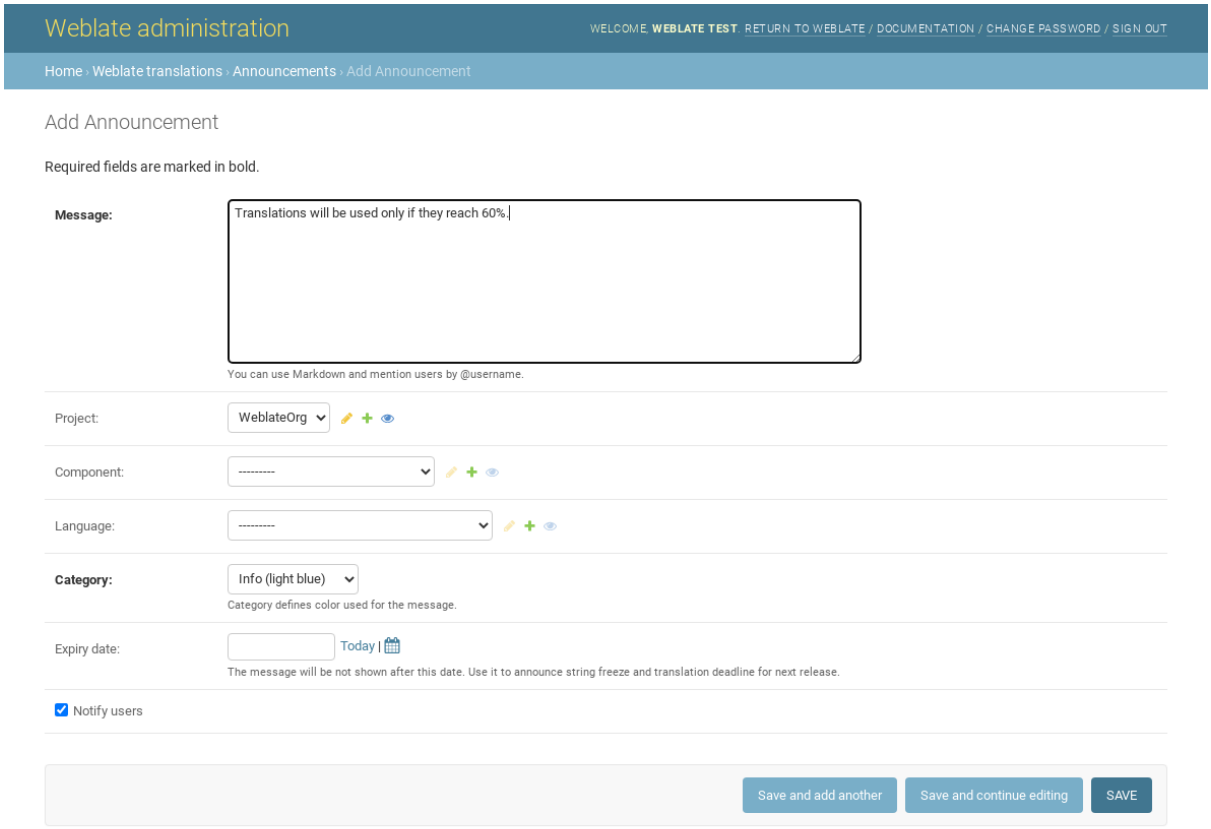
The users will receive notification on the announcements for watched projects (unless they opt out).

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*:



Sie kann auch über die Adminoberfläche hinzugefügt werden:



Die Ankündigungen werden dann in dem angegebenen Kontext angezeigt:

Kein Kontext angegeben

Wird auf der Übersicht (Einstiegsseite) angezeigt.

Projekt spezifiziert

Wird im Projekt angezeigt, einschließlich aller Komponenten und Übersetzungen.

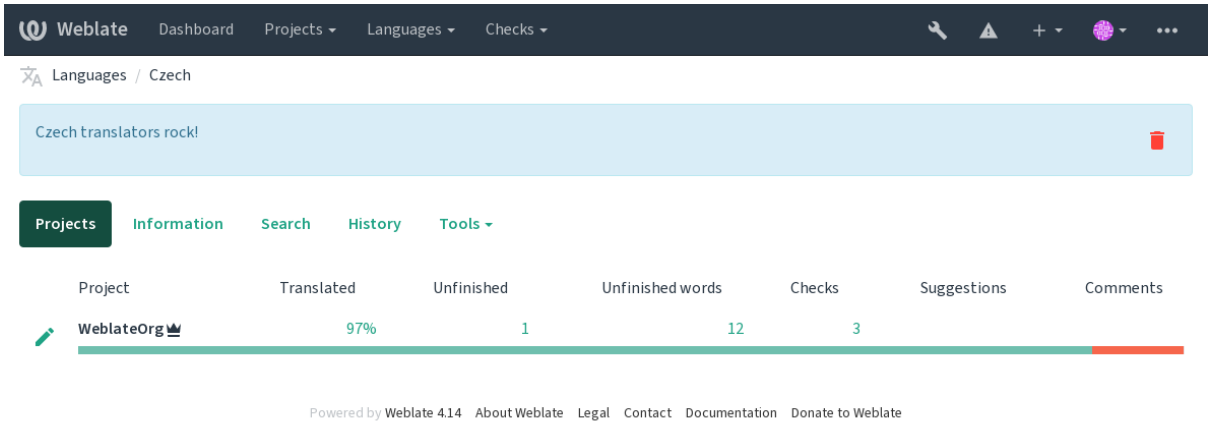
Komponente spezifiziert

Wird für eine bestimmte Komponente und alle ihre Übersetzungen angezeigt.

Sprache spezifiziert

Wird in der Sprachübersicht und allen Übersetzungen in dieser Sprache angezeigt.

So sieht es auf der Übersichtsseite der Sprachen aus:



2.20 Component Lists

Specify multiple lists of components to appear as options on the user dashboard, from which users can pick one as their default view. See *Übersicht* to learn more.

Geändert in Version 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.

Geändert in Version 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

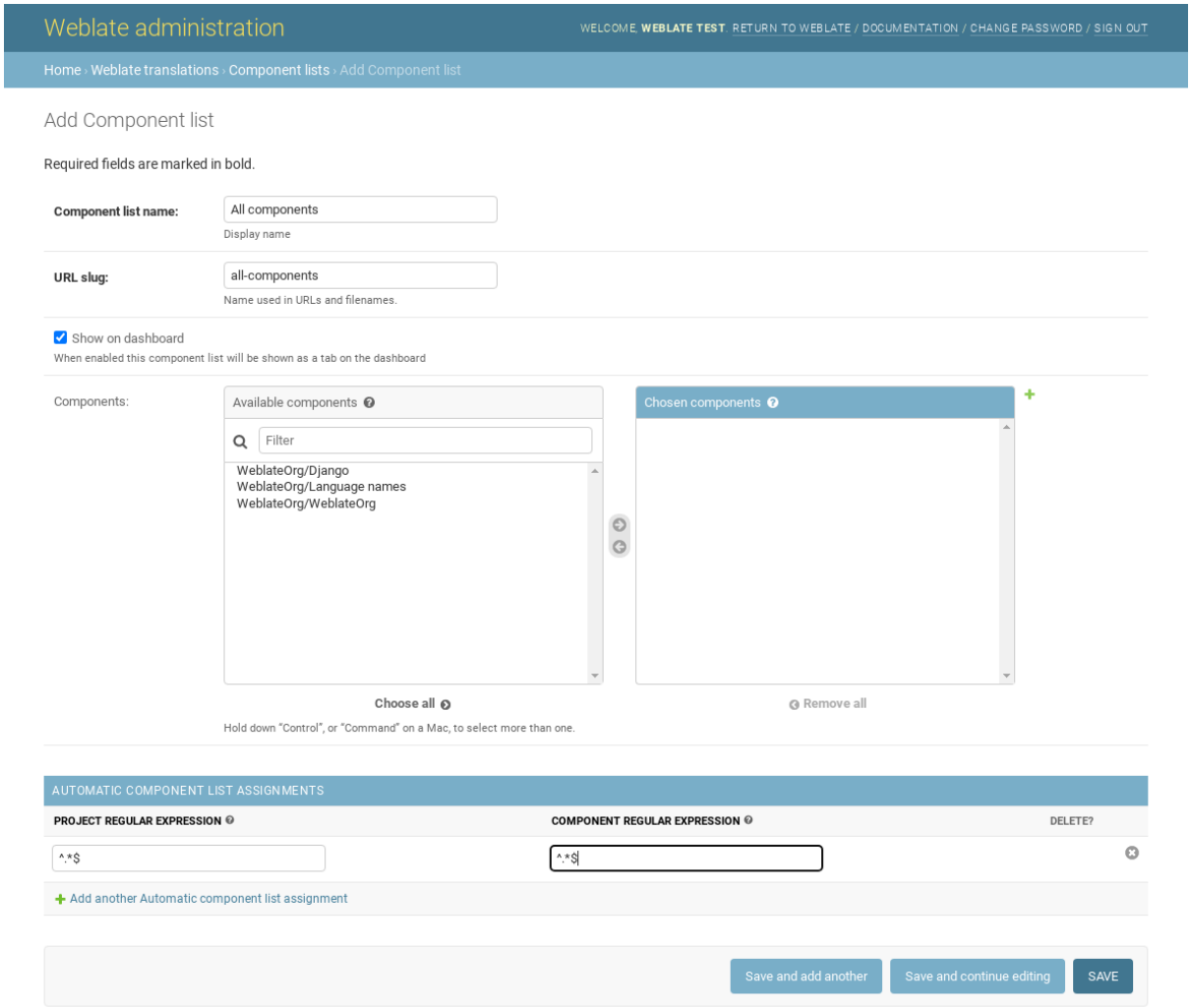
Neu in Version 2.13.

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.

Hinweis: 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

Neu in Version 2.10.

Provides you read-only access to the underlying Git repository using HTTP(S).

Installation

1. Add `weblate.gitexport` to installed apps in `settings.py`:

```
INSTALLED_APPS += ("weblate.gitexport",)
```

2. Export existing repositories by migrating your database after installation:

```
weblate migrate
```

Usage

The module automatically hooks into Weblate and sets the exported repository URL in the *Component configuration*. The repositories are accessible under the `/git/` part of the Weblate URL, for example `https://example.org/git/weblate/main/`.

Repositorys für öffentlich zugängliche Projekte können ohne Authentifizierung geklont werden:

```
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/'
```

Hinweis: Standardmäßig haben Mitglieder der Gruppe *Benutzer* und anonyme Benutzer über die Rollen *Auf Repository zugreifen* und *Hauptbenutzer* Zugriff auf die Repositorys für öffentliche Projekte.

2.21.2 Abrechnung

Neu in Version 2.4.

This is used on Hosted Weblate to define billing plans, track invoices and usage limits.

Installation

1. Add `weblate.billing` to installed apps in `settings.py`:

```
INSTALLED_APPS += ("weblate.billing",)
```

2. Run the database migration to optionally install additional database structures for the module:

```
weblate migrate
```

Usage

After installation you can control billing in the admin interface. Users with billing enabled will get new *Billing* tab in their *Benutzerprofil*.

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 Rechtliche Grundlagen

Neu in Version 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

Bemerkung: Legal documents for the Hosted Weblate service are available in this Git repository <https://github.com/WeblateOrg/wllegal/tree/main/wllegal/templates/legal/documents>.

Most likely these will not be directly usable to you, but might come in handy as a starting point if adjusted to meet your needs.

Installation

1. Add `weblate.legal` to installed apps in `settings.py`:

```
INSTALLED_APPS += ("weblate.legal",)

# Optional:

# Social auth pipeline to confirm TOS upon registration/subsequent sign in
SOCIAL_AUTH_PIPELINE += ("weblate.legal.pipeline.tos_confirm",)

# Middleware to enforce TOS confirmation of signed in users
MIDDLEWARE += [
    "weblate.legal.middleware.RequireTOSMiddleware",
]
```

2. Run the database migration to optionally install additional database structures for the module:

```
weblate migrate
```

3. Edit the legal documents in the `weblate/legal/templates/legal/` folder to match your service.

Usage

After installation and editing, the legal documents are shown in the Weblate UI.

2.21.4 Avatars

Avatars are downloaded and cached server-side to reduce information leaks to the sites serving them by default. The built-in support for fetching avatars from e-mails addresses configured for it can be turned off using `ENABLE_AVATARS`.

Weblate currently supports:

- Gravatar
- Libravatar

Siehe auch:

Avatar-Zwischenspeicherung, `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:

- Vorschläge von unautorisierten Benutzern
- Projekt- und Komponentenbeschreibungen und Links

Bemerkung: This (among other things) relies on IP address of the client, please see *Running behind reverse proxy* for properly configuring that.

Siehe auch:

Running behind reverse proxy, `AKISMET_API_KEY`, `WEBLATE_AKISMET_API_KEY`

2.21.6 Signing Git commits with GnuPG

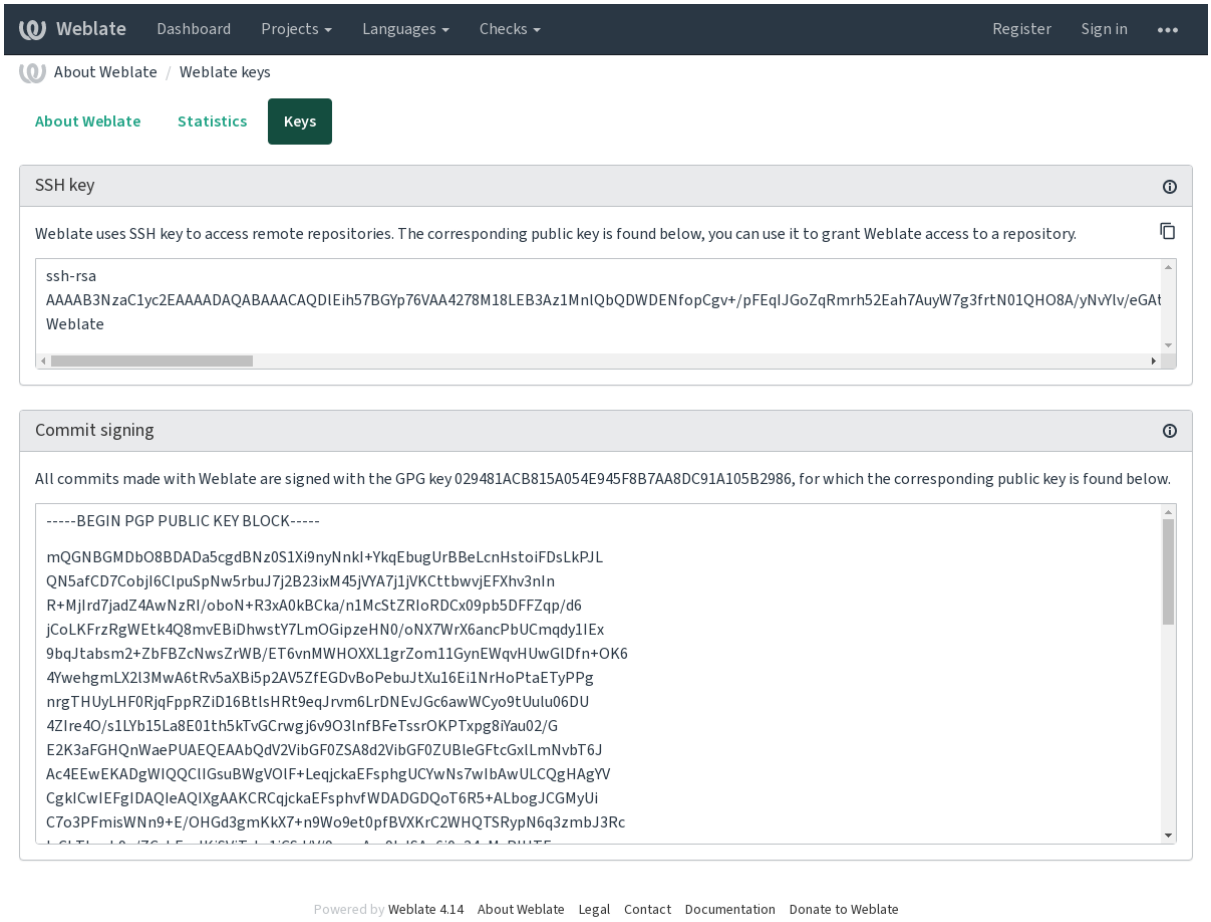
Neu in Version 3.1.

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.

Siehe auch:

WEBLATE_GPG_IDENTITY

2.21.7 Rate limiting

Geändert in Version 3.2: The rate limiting now accepts more fine-grained configuration.

Geändert in Version 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:

Name	Zielgruppe	Allowed attempts	Ratelimit window	Lockout period
Registrierung	`REGISTRATION`	5	300	600
Nachricht an Administratoren senden	`MESSAGE`	5	300	600
Passwort-Authentifizierung bei der Anmeldung	`LOGIN`	5	300	600
Plattformweite Suche	`SEARCH`	6	60	60
Translating	`TRANSLATE`	30	60	600
Adding to glossary	`GLOSSARY`	30	60	600
Beginn der Übersetzung in eine neue Sprache	`LANGUAGE`	2	300	600
Neues Projekt erstellen	`PROJECT`	5	600	600

If a user fails to sign in *`AUTH_LOCK_ATTEMPTS`* times, password authentication will be turned off on the account until having gone through the process of having its password reset.

The settings can be also applied in the Docker container by adding `WEBLATE_` prefix to the setting name, for example *`RATELIMIT_ATTEMPTS`* becomes *`WEBLATE_RATELIMIT_ATTEMPTS`*.

The API has separate rate limiting settings, see *API rate limiting*.

Siehe auch:

Rate limiting, *Running behind reverse proxy*, *API rate limiting*

2.21.8 Integration von Fedora-Messaging

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/> for setup instructions.

2.22 Customizing Weblate

Extend and customize using Django and Python. Contribute your changes upstream so that everybody can benefit. This reduces your maintenance costs; code in Weblate is taken care of when changing internal interfaces or refactoring the code.

Warnung: 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.

Siehe auch:

Contributing to Weblate

2.22.1 Creating a Python module

If you are not familiar with Python, you might want to look into Python For Beginners, explaining the basics and pointing to further tutorials.

To write some custom Python code (called a module), a place to store it is needed, either in the system path (usually something like `/usr/lib/python3.9/site-packages/`) or in the Weblate directory, which is also added to the interpreter search path.

Better yet, turn your customization into a proper Python package:

1. Create a folder for your package (we will use *weblate_customization*).
2. Within it, create a `setup.py` file to describe the package:

```
from setuptools import setup

setup(
    name="weblate_customization",
    version="0.0.1",
    author="Your name",
    author_email="yourname@example.com",
    description="Sample Custom check for Weblate.",
    license="GPLv3+",
    keywords="Weblate check example",
    packages=["weblate_customization"],
)
```

3. Create a folder for the Python module (also called `weblate_customization`) for the customization code.
4. Within it, create a `__init__.py` file to make sure Python can import the module.
5. This package can now be installed using **`pip install -e`**. More info to be found in Editable installs.
6. Once installed, the module can be used in the Weblate configuration (for example `weblate_customization.checks.FooCheck`).

Your module structure should look like this:

```
weblate_customization
├── setup.py
└── weblate_customization
    ├── __init__.py
    ├── addons.py
    └── checks.py
```

You can find an example of customizing Weblate at <https://github.com/WeblateOrg/customize-example>, it covers all the topics described below.

2.22.2 Changing the logo

1. Create a simple Django app containing the static files you want to overwrite (see *Creating a Python module*).

   Branding appears in the following files:

   `icons/weblate.svg`
   Logo shown in the navigation bar.

   `logo-*.png`
   Web icons depending on screen resolution and web-browser.

   `favicon.ico`
   Web icon used by legacy browsers.

   `weblate-*.png`
   Avatars for bots or anonymous users. Some web-browsers use these as shortcut icons.

email-logo.png
Used in notifications e-mails.

2. Add it to `INSTALLED_APPS`:

```
INSTALLED_APPS = (
    # Add your customization as first
    "weblate_customization",
    # Weblate apps are here…
)
```

3. Run `weblate collectstatic --noinput`, to collect static files served to clients.

Siehe auch:

How to manage static files (e.g. images, JavaScript, CSS), *Serving static files*

2.22.3 Custom quality checks, add-ons and auto-fixes

To install your code for *Custom automatic fixups*, *Writing own checks* or *Schreiben einer Erweiterung* in Weblate:

1. Place the files into your Python module containing the Weblate customization (see *Creating a Python module*).
2. Add its fully-qualified path to the Python class in the dedicated settings (`WEBLATE_ADDONS`, `CHECK_LIST` or `AUTOFIX_LIST`):

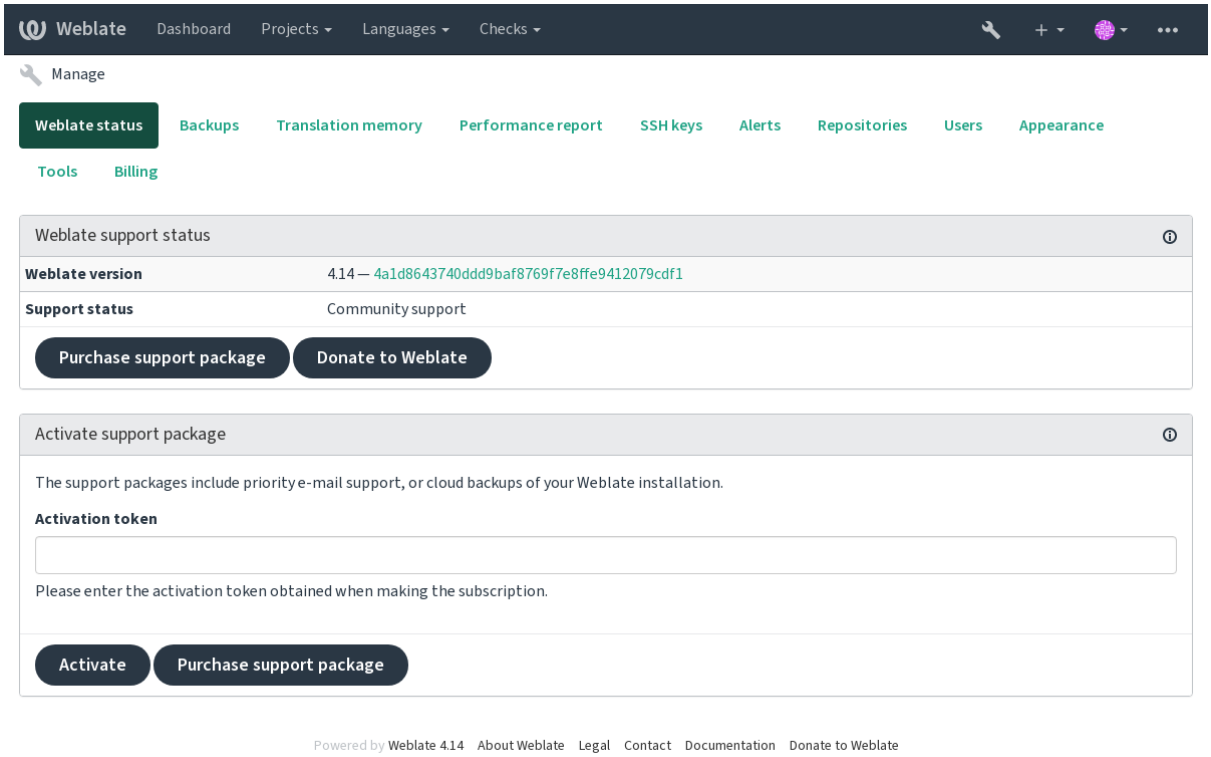
```
# Checks
CHECK_LIST += ("weblate_customization.checks.FooCheck",)

# Autofixes
AUTOFIX_LIST += ("weblate_customization.autofix.FooFixer",)

# Add-ons
WEBLATE_ADDONS += ("weblate_customization.addons.ExamplePreAddon",)
```

Siehe auch:

Custom automatic fixups, *Writing own checks*, *Schreiben einer Erweiterung*, *Ausführen von Skripten der Erweiterung*

2.23 Verwaltungsoberfläche

Die Verwaltungsoberfläche bietet Verwaltungseinstellungen unter der URL `/manage/`. Sie ist für Benutzer verfügbar, die mit Administratorrechten angemeldet sind, und kann über das Schraubenschlüssel-Symbol oben rechts aufgerufen werden:



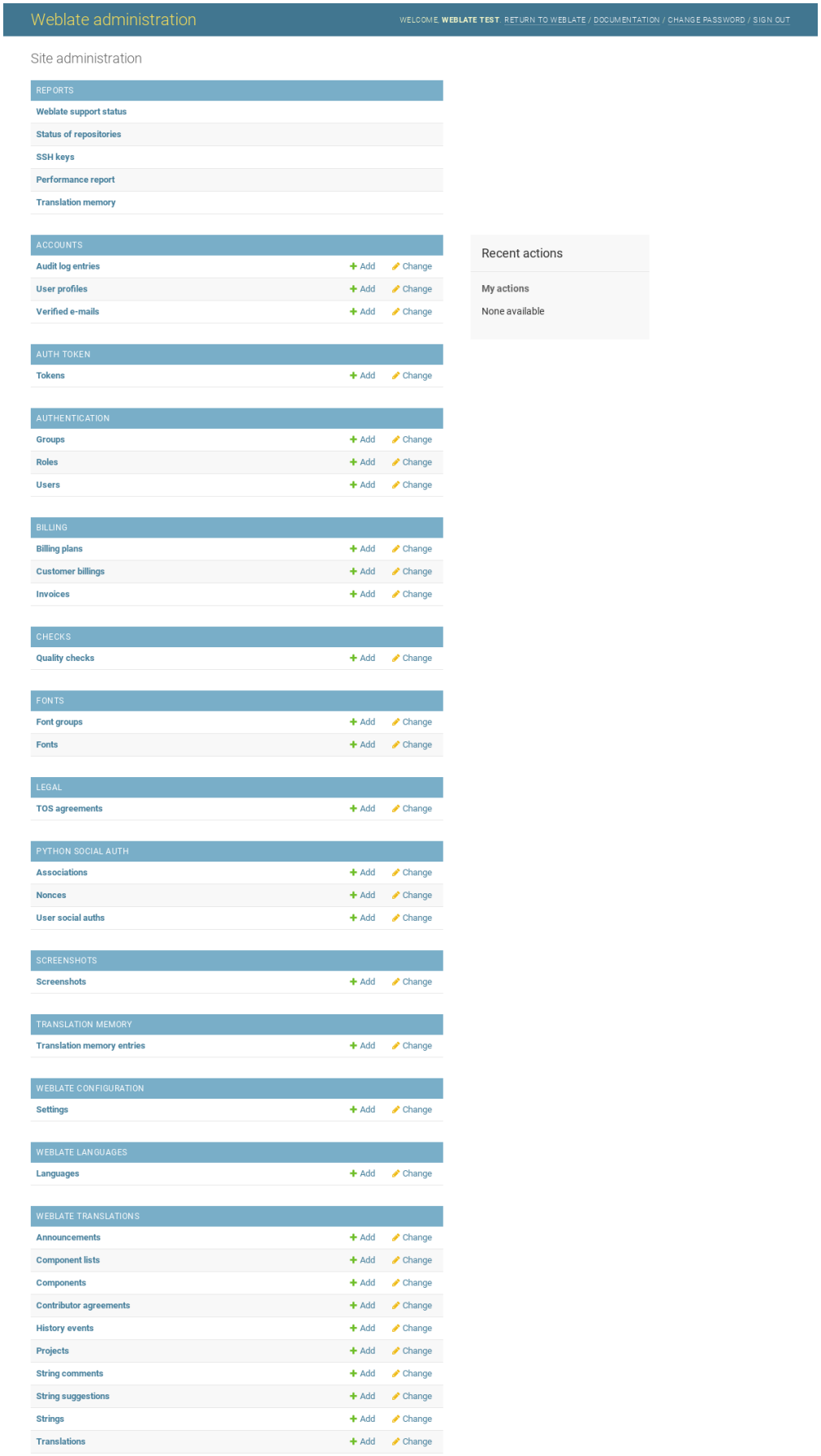
Sie enthält einen grundlegenden Überblick über Ihr Weblate:

- Support-Status, siehe *Getting support for Weblate*
- Backups, siehe *Sichern und Verschieben von Weblate*
- Gemeinsamer Übersetzungsspeicher, siehe Speicher
- Performance report to review Weblate health and length of Celery queues
- SSH-Schlüsselverwaltung, siehe *SSH repositories*
- Alerts overview for all components, see alerts

2.23.1 The Django admin interface

Warnung: 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
WELCOME, WEBLATE TEST. RETURN TO WEBLATE / DOCUMENTATION / CHANGE PASSWORD / SIGN OUT
Site administration
REPORTS
Weblate support status
Status of repositories
SSH keys
Performance report
Translation memory
ACCOUNTS
Audit log entries + Add Change
User profiles + Add Change
Verified e-mails + Add Change
AUTH TOKEN
Tokens + Add Change
AUTHENTICATION
Groups + Add Change
Roles + Add Change
Users + Add Change
BILLING
Billing plans + Add Change
Customer billings + Add Change
Invoices + Add Change
CHECKS
Quality checks + Add Change
FONTS
Font groups + Add Change
Fonts + Add Change
LEGAL
TOS agreements + Add Change
PYTHON SOCIAL AUTH
Associations + Add Change
Nonces + Add Change
User social auths + Add Change
SCREENSHOTS
Screenshots + Add Change
TRANSLATION MEMORY
Translation memory entries + Add Change
WEBLATE CONFIGURATION
Settings + Add Change
WEBLATE LANGUAGES
Languages + Add Change
WEBLATE TRANSLATIONS
Announcements + Add Change
Component lists + Add Change
Components + Add Change
Contributor agreements + Add Change
History events + Add Change
Projects + Add Change
String comments + Add Change
String suggestions + Add Change
Strings + Add Change
Translations + Add Change
Recent actions
My actions
None available

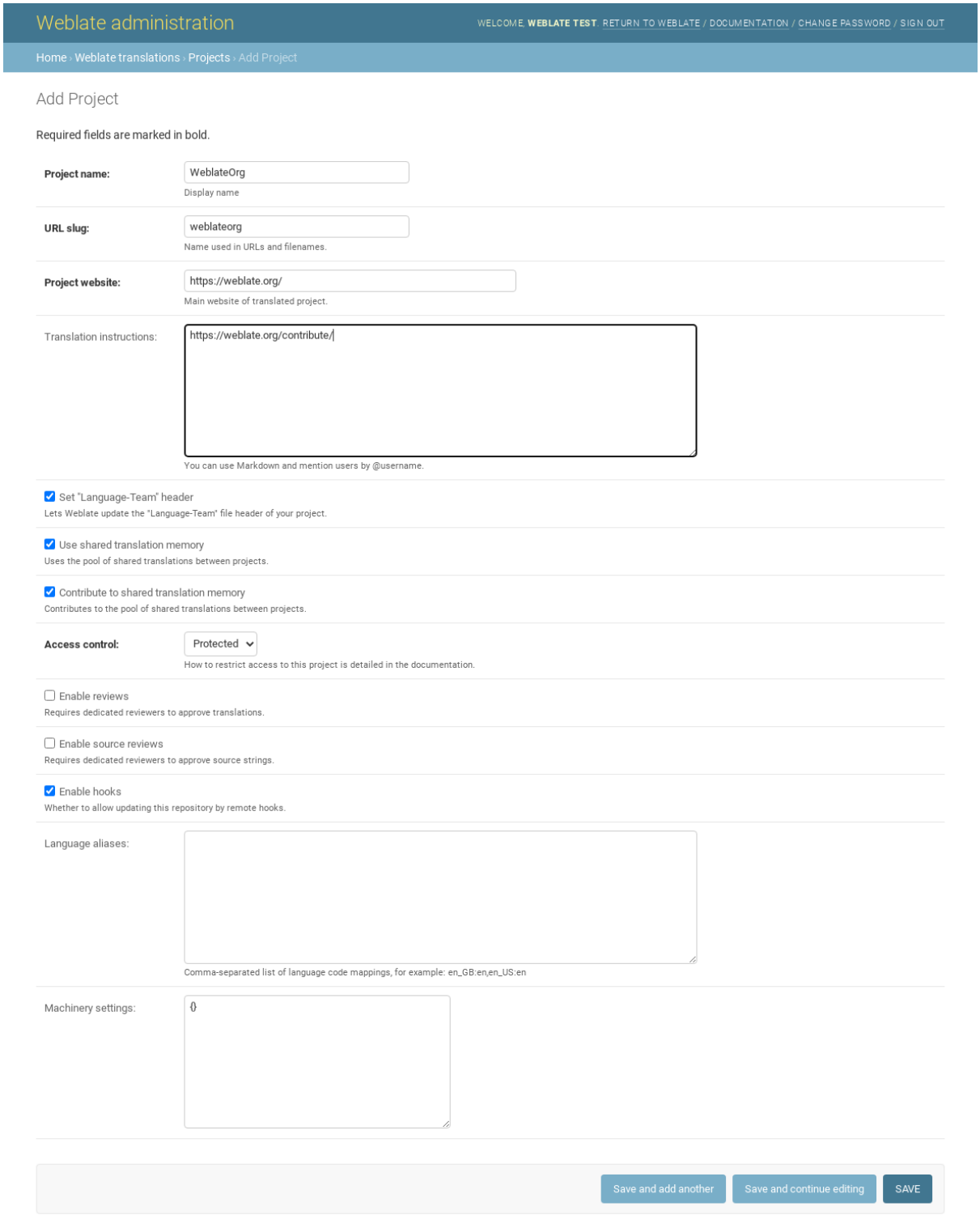
In the *Reports* section, you can check the status of your site, tweak it for *Production setup*, or manage SSH keys used to access *Accessing repositories*.

Manage database objects under any of the sections. The most interesting one is probably *Weblate translations*, where you can manage translatable projects, see *Project configuration* and *Component configuration*.

Weblate languages holds language definitions, explained further in *Language definitions*.

Hinzufügen eines Projekts

Das Hinzufügen eines Projekts dient als Container für alle Komponenten. Normalerweise erstellen Sie ein Projekt für ein Stück Software oder ein Buch (siehe *Project configuration* für Informationen zu den einzelnen Parametern):



Siehe auch:

Project configuration

Zweisprachige Komponenten

Once you have added a project, translation components can be added to it. (See *Component configuration* for info regarding individual parameters):

Weblate administration

WELCOME, **WEBLATE TEST**. RETURN TO WEBLATE / DOCUMENTATION / CHANGE PASSWORD / SIGN OUT

Home › Weblate translations › Components › Add Component

Add Component

IMPORT SPHEZ DOCUMENTATION

Required fields are marked in bold.

Component name: Language names
Display name

URL slug: language-names
Name used in URLs and filenames.

Project: WeblateOrg

Version control system: Git
Version control system to use to access your repository containing translations. You can also choose additional integration with third party providers to submit merge requests.

Source code repository: https://github.com/WeblateOrg/demo.git
URL of a repository, use weblate://project/component to share it with other component.

Repository push URL:
URL of a push repository, pushing is turned off if empty.

Repository browser: https://github.com/WeblateOrg/demo/blob/{{branch}}/{{filename}}#L{
Link to repository browser, use {{branch}} for branch, {{filename}} and {{pos}} as filename and line placeholders. You might want to strip leading directory by using {{filenameparent0}}.

Exported repository URL:
URL of repository where users can fetch changes from Weblate

Source string bug reporting address:
E-mail address for reports on errors in source strings. Leave empty for no e-mails.

Repository branch:
Repository branch to translate

Push branch:
Branch for pushing changes, leave empty to use repository branch

File mask: weblate/langdata/locale/*/LC_MESSAGES/djo
Path of files to translate relative to repository root, use * instead of language code, for example po/* po or locale/*/LC_MESSAGES/django.po.

Monolingual base language file:
Filename of translation base file, containing all strings and their source, it is recommended for monolingual translation formats.

☑ Edit base file
Whether users will be able to edit the base file for monolingual translations.

Intermediate language file:
Filename of intermediate translation file. In most cases this is a translation file provided by developers and is used when creating actual source strings.

Template for new translations: weblate/langdata/locale/django.pot
Filename of file used for creating new translations. For gettext choose .pot file.

File format: gettext PO file

☐ Locked
Locked component will not get any translation updates.

☑ Allow translation propagation
Whether translation updates in other components will cause automatic translation in this one

☑ Turn on suggestions
Whether to allow translation suggestions at all.

☐ Suggestion voting
Users can only vote for suggestions and can't make direct translations.

Autoaccept suggestions: 0
Automatically accept suggestions with this number of votes, use 0 to turn it off.

Translation flags:
Additional comma-separated flags to influence Weblate behavior.

Enforced checks: []
List of checks which can not be ignored.

Translation license: GNU General Public License v3.0 or later

Contributor agreement:
User agreement which needs to be approved before a user can translate this component.

Adding new translation: Create new language file
How to handle requests for creating new translations.

Language code style: Default based on the file format
Customize language code used to generate the filename for translations created by Weblate.

☐ Manage strings
Enables adding and removing strings straight from Weblate. If your strings are extracted from the source code or managed externally you probably want to keep it disabled.

Merge style: Rebase
Define whether Weblate should merge the upstream repository or rebase changes onto it.

Commit message when translating:

```
Translated using Weblate ({{ language_name }})

Currently translated at {{ stats.translated_percent }}% ({{ stats.translated }} of {{ stats.all }} strings)

Translation: {{ project_name }}/{{ component_name }}
Translate-URL: {{ url }}
```

You can use template language for various info, please consult the documentation for more details.

Commit message when adding translation:

```
Added translation using Weblate ({{ language_name }})
```

You can use template language for various info, please consult the documentation for more details.

Commit message when removing translation:

```
Deleted translation using Weblate ({{ language_name }})
```

You can use template language for various info, please consult the documentation for more details.

Commit message when merging translation:

```
Merge branch '{{ component_remote_branch }}' into Weblate.
```

You can use template language for various info, please consult the documentation for more details.

Commit message when add-on makes a change:

```
Update translation files

Updated by "{{ addon_name }}" hook in Weblate.

Translation: {{ project_name }}/{{ component_name }}
Translate-URL: {{ url }}
```

You can use template language for various info, please consult the documentation for more details.

Merge request message:

```
Translations update from {{ site_title }}

Translations update from {{ site_title }}({{ site_url }}) for {{ project_name }}/{{ component_name }}({{{url}}}.

{% if component_linked_childs %}
It also includes following components:
{% for linked in component_linked_childs %}
* {{ linked.project_name }}/{{ linked.name }}({{ linked.url }})
{% endfor %}
{% endif %}
```

You can use template language for various info, please consult the documentation for more details.

☑ Push on commit
Whether the repository should be pushed upstream on every commit.

Age of changes to commit: 24
Time in hours after which any pending changes will be committed to the VCS.

☑ Lock on error
Whether the component should be locked on repository errors.

Source language: English
Language used for source strings in all components.

Language filter: ^(nsh|hu)$
Regular expression used to filter translation files when scanning for file mask.

Variants regular expression:
Regular expression used to determine variants of a string.

Priority: Medium
Components with higher priority are offered first to translators.

☐ Restricted component
Restrict access to the component to only those explicitly given permission.

Share in projects: WeblateOrg
Choose additional projects where this component will be listed. Hold down "Control", or "Command" on a Mac, to select more than one.

☐ Use as a glossary

Glossary color: Silver

Remote revision:

Local revision:

Save and add another | Save and continue editing | SAVE

Siehe auch:

Component configuration, *Bilingual and monolingual formats*

Einsprachige Komponenten

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):

Weblate administration

WELCOME, **WEBLATE TEST**. RETURN TO WEBLATE / DOCUMENTATION / CHANGE PASSWORD / SIGN OUT

Home › Weblate translations › Components › Add Component

Add Component

IMPORT SPHINX DOCUMENTATION

Required fields are marked in bold.

Component name: Android
Display name

URL slug: android
Name used in URLs and filenames.

Project: WeblateOrg

Version control system: Git
Version control system to use to access your repository containing translations. You can also choose additional integration with third party providers to submit merge requests.

Source code repository: weblate://weblateorg/language-names
URL of a repository, use weblate://project/component to share it with other component.

Repository push URL:
URL of a push repository, pushing is turned off if empty.

Repository browser:
Link to repository browser, use {{branch}} for branch, {{filename}} and {{line}} as filename and line placeholders. You might want to strip leading directory by using {{filename:sequence:s}}.

Exported repository URL:
URL of repository where users can fetch changes from Weblate.

Source string bug reporting address:
E-mail address for reports on errors in source strings. Leave empty for no e-mails.

Repository branch:
Repository branch to translate.

Push branch:
Branch for pushing changes, leave empty to use repository branch.

File mask: app/src/main/res/values-*/strings.xml
Path of files to translate relative to repository root, use * instead of language code, for example: po/*.po or locale/*/LC_MESSAGES/djangojs.po.

Monolingual base language file: app/src/main/res/values/strings.xml
Filename of translation base file, containing all strings and their source, it is recommended for monolingual translation formats.

☑ Edit base file
Whether users will be able to edit the base file for monolingual translations.

Intermediate language file:
Filename of intermediate translation file. In most cases this is a translation file provided by developers and is used when creating actual source strings.

Template for new translations:
Filename of file used for creating new translations. For gettext choose .pot file.

File format: Android String Resource

☐ Locked
Locked component will not get any translation updates.

☑ Allow translation propagation
Whether translation updates in other components will cause automatic translation in this one.

☑ Turn on suggestions
Whether to allow translation suggestions at all.

☐ Suggestion voting
Users can only vote for suggestions and can't make direct translations.

Autoaccept suggestions: 0
Automatically accept suggestions with this number of votes, use 0 to turn it off.

Translation flags:
Additional comma-separated flags to influence Weblate behavior.

Enforced checks:
!
List of checks which can not be ignored.

Translation license: MIT License

Contributor agreement:
User agreement which needs to be approved before a user can translate this component.

Adding new translation: Create new language file
How to handle requests for creating new translations.

Language code style: Default based on the file format
Customize language code used to generate the filename for translations created by Weblate.

☐ Manage strings
Enables adding and removing strings straight from Weblate. If your strings are extracted from the source code or managed externally you probably want to keep it disabled.

Merge style: Rebase
Define whether Weblate should merge the upstream repository or rebase changes onto it.

Commit message when translating:
Translated using Weblate ({{ language_name }})

Currently translated at {{ stats.translated_percent }}% ({{ stats.translated }} of {{ stats.all }} strings)

Translation: {{ project_name }}/{{ component_name }}
Translate-URL: {{ url }}

You can use template language for various info, please consult the documentation for more details.

Commit message when adding translation:
Added translation using Weblate ({{ language_name }})

You can use template language for various info, please consult the documentation for more details.

Commit message when removing translation:
Deleted translation using Weblate ({{ language_name }})

You can use template language for various info, please consult the documentation for more details.

Commit message when merging translation:
Merge branch '{{ component_remote_branch }}' into Weblate.

You can use template language for various info, please consult the documentation for more details.

Commit message when add-on makes a change:
Update translation files

Updated by "{{ addon_name }}" hook in Weblate.

Translation: {{ project_name }}/{{ component_name }}
Translate-URL: {{ url }}

You can use template language for various info, please consult the documentation for more details.

Merge request message:
Translations update from {{ site_title }}

Translations update from {{ site_title }}({{ site_url }}) for {{ project_name }}/{{ component_name }}({{ url }}).

{% if component_linked_childs %}
Rule includes following components:
{% for linked in component_linked_childs %}
* {{ linked.project_name }}/{{ linked.name }}({{ linked.url }})
{% endfor %}
{% endif %}

You can use template language for various info, please consult the documentation for more details.

☑ Push on commit
Whether the repository should be pushed upstream on every commit.

Age of changes to commit: 24
Time in hours after which any pending changes will be committed to the VCS.

☑ Lock on error
Whether the component should be locked on repository errors.

Source language: English
Language used for source strings in all components.

Language filter: ^[^,]+$
Regular expression used to filter translation files when scanning for file mask.

Variants regular expression:
Regular expression used to determine variants of a string.

Priority: Medium
Components with higher priority are offered first to translators.

☐ Restricted component
Restrict access to the component to only those explicitly given permission.

Share in projects: WeblateOrg
Choose additional projects where this component will be listed. Hold down "Control", or "Command" on a Mac, to select more than one.

☐ Use as a glossary

Glossary color: Silver

Remote revision:

Local revision:

Save and add another | Save and continue editing | SAVE

Siehe auch:

Component configuration, *Bilingual and monolingual formats*

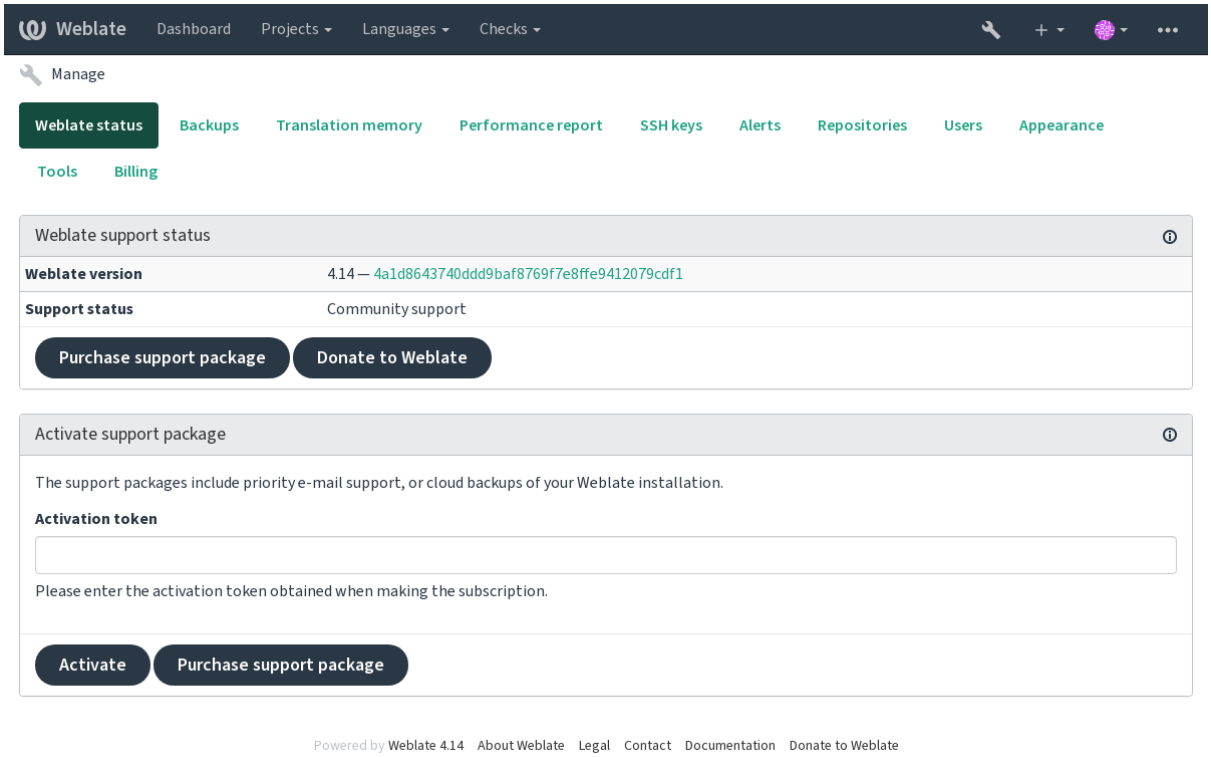
2.24 Getting support for Weblate

Weblate is copylefted libre software with community support. Subscribers receive priority support at no extra charge. Prepaid help packages are available for everyone. You can find more info about current support offerings at <https://weblate.org/support/>.

2.24.1 Integrating support

Neu in Version 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 *Entdecken Sie Weblate* 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
- *Von Weblate bereitgestellter Backup-Speicher*
- *Entdecken Sie Weblate*

Hinweis: Purchased support packages are already activated upon purchase, and can be used without integrating them.

2.24.4 Entdecken Sie Weblate

Neu in Version 4.5.2.

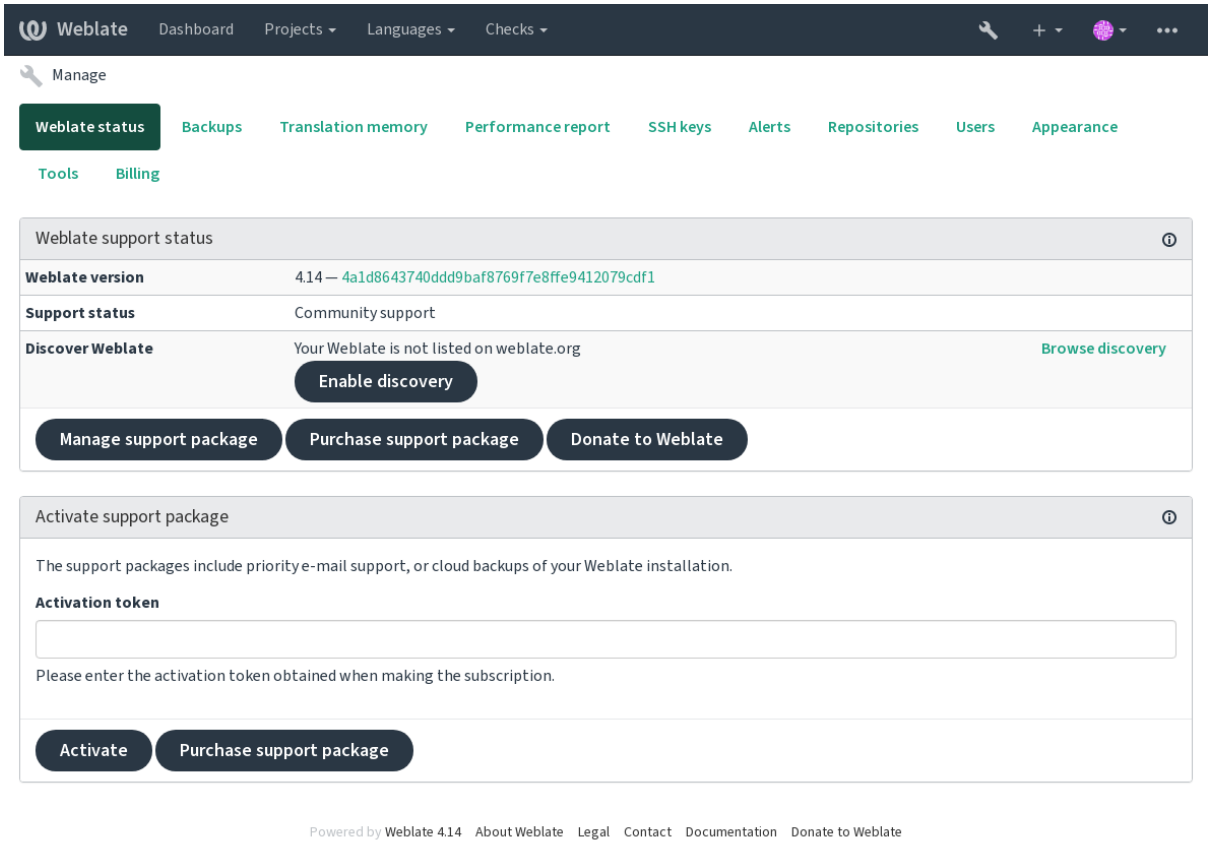
Bemerkung: Diese Funktion befindet sich derzeit in einer frühen Betaphase.

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.

Aufnahme in die Liste

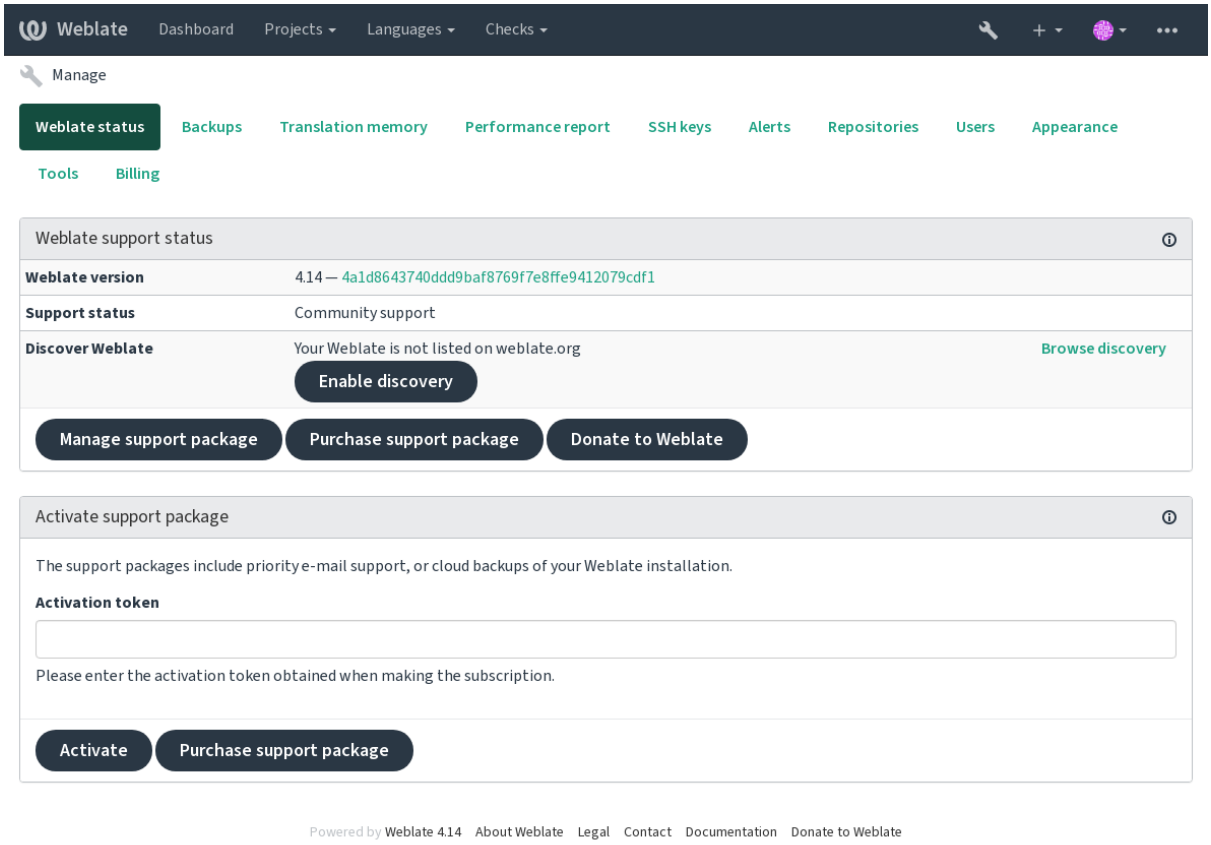
Hinweis: Participating in Discover Weblate makes Weblate submit some information about your server, please see *Data submitted to the Weblate*.

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:



Auflistung Ihres Servers ohne ein Support-Abonnement in Discover Weblate:

1. Registrieren Sie sich unter <https://weblate.org/user/>
2. Registrieren Sie Ihren Weblate-Server in der Discovery-Datenbank unter <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:



Auflistung anpassen

You can customize the listing by providing a text and image (570 x 260 pixels) at <https://weblate.org/user/>.

2.25 Legal documents

Bemerkung: Herein you will find various legal information you might need to operate Weblate in certain legal jurisdictions. It is provided as a means of guidance, without any warranty of accuracy or correctness. It is ultimately your responsibility to ensure that your use of Weblate complies with all applicable laws and regulations.

2.25.1 ITAR and other export controls

Weblate can be run within your own datacenter or virtual private cloud. As such, it can be used to store ITAR or other export-controlled information, however, end users are responsible for ensuring such compliance.

The Hosted Weblate service has not been audited for compliance with ITAR or other export controls, and does not currently offer the ability to restrict translations access by country.

2.25.2 US encryption controls

Weblate does not contain any cryptographic code, but might be subject export controls as it uses third party components utilizing cryptography for authentication, data-integrity and -confidentiality.

Most likely Weblate would be classified as ECCN 5D002 or 5D992 and, as publicly available libre software, it should not be subject to EAR (see Encryption items NOT Subject to the EAR).

Software components used by Weblate (listing only components related to cryptographic function):

Python
: See https://wiki.python.org/moin/PythonSoftwareFoundationLicenseFaq#Is_Python_subject_to_export_laws.3F

GnuPG
: Optionally used by Weblate

Git
: Optionally used by Weblate

curl
: Used by Git

OpenSSL
: Used by Python and cURL

The strength of encryption keys depends on the configuration of Weblate and the third party components it interacts with, but in any decent setup it will include all export restricted cryptographic functions:

- In excess of 56 bits for a symmetric algorithm
- Factorisation of integers in excess of 512 bits for an asymmetric algorithm
- Computation of discrete logarithms in a multiplicative group of a finite field of size greater than 512 bits for an asymmetric algorithm
- Discrete logarithms in a group different than above in excess of 112 bits for an asymmetric algorithm

Weblate doesn't have any cryptographic activation feature, but it can be configured in a way where no cryptography code would be involved. The cryptographic features include:

- Accessing remote servers using secure protocols (HTTPS)
- Generating signatures for code commits (PGP)

Siehe auch:

Export Controls (EAR) on Open Source Software

KAPITEL 3

Contributor docs

3.1 Contributing to 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:

- *Melden von Problemen in Weblate*
- *Starting contributing code to Weblate*
- *Contributing to Weblate modules*
- *Translating Weblate*
- *Zur Weblate-Dokumentation beitragen*
- *Weblate-Diskussionen*
- *Funding Weblate development*

3.1.1 Translating Weblate

Weblate wird laufend mit Hilfe von Weblate selbst <https://hosted.weblate.org/>`_übersetzt. Helfen Sie mit, Weblate in so vielen Sprachen wie möglich verfügbar zu machen. Das bringt Weblate näher an seine Benutzer!

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 Zur Weblate-Dokumentation beitragen

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.

Hinweis: You can translate the docs.

3.1.3 Erweiterung der integrierten Sprachdefinitionen

The language definitions are in the weblate-language-data repository.

Sie können fehlende Sprachdefinitionen gerne in `languages.csv` ergänzen, andere Dateien werden aus dieser Datei generiert.

3.1.4 Weblate-Diskussionen

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 Funding Weblate development

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.

Supporters who have funded Weblate

Liste der Weblate-Unterstützer:

- Yashiro Ccs
- Cheng-Chia Tseng
- Timon Reinhard
- Cassidy James
- Loic Dachary
- Marozed

- https://freedombox.org/
- GNU Solidario (GNU Health)
- BallotReady
- Richard Nespithal
- MyExpenses.Mobi

Möchten Sie in die Liste aufgenommen werden? Bitte beachten Sie die Optionen auf der Seite Spenden für Weblate.

3.2 Starting contributing code to Weblate

Understand the Weblate source code by going through *Weblate source code*, *Weblate frontend* and *Weblate-Interna*.

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 Running Weblate locally

The most comfortable approach to get started with Weblate development is to follow *Installieren aus Quellen*. It will get you a virtualenv with editable Weblate sources.

1. Klonen des Weblate-Quellcodes:

```
git clone https://github.com/WeblateOrg/weblate.git
cd weblate
```

2. Create a virtualenv:

```
virtualenv .venv
.venv/bin/activate
```

3. Install Weblate (for this you need some system dependencies, see *Installieren aus Quellen*):

```
pip install -e .
```

3. Install all dependencies useful for development:

```
pip install -r requirements-dev.txt
```

4. Start a development server:

```
weblate runserver
```

5. Depending on your configuration, you might also want to start Celery workers:

```
./weblate/examples/celery start
```

6. To run a test (see *Local testing* for more details):

```
. scripts/test-database
./manage.py test
```

Siehe auch:

Installieren aus Quellen

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.

Das Skript akzeptiert auch einige Parameter, um Tests auszuführen, führen Sie es mit dem Parameter `test` aus und geben Sie dann alle `test`-Parameter an, um zum Beispiel nur Tests im Modul `weblate.machine` auszuführen:

```
./rundev.sh test --failfast weblate.machine
```

Bemerkung: 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
```

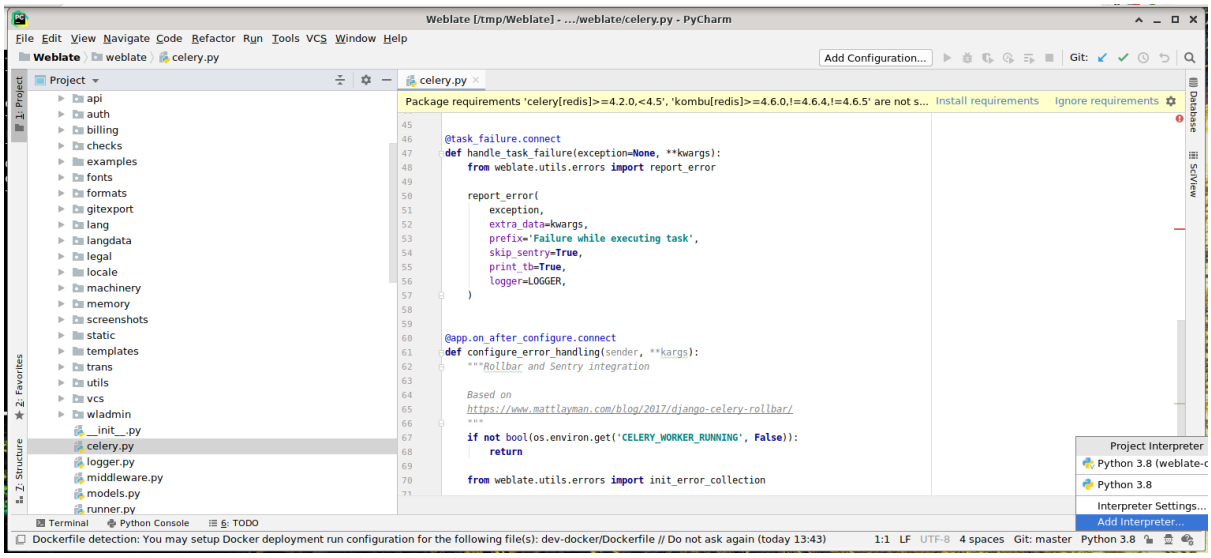
Wenn Sie das Skript ohne Argumente ausführen, wird der Docker-Container neu erstellt und neu gestartet.

Bemerkung: This is not a suitable setup for production, as it includes several hacks which are insecure, but they make development easier.

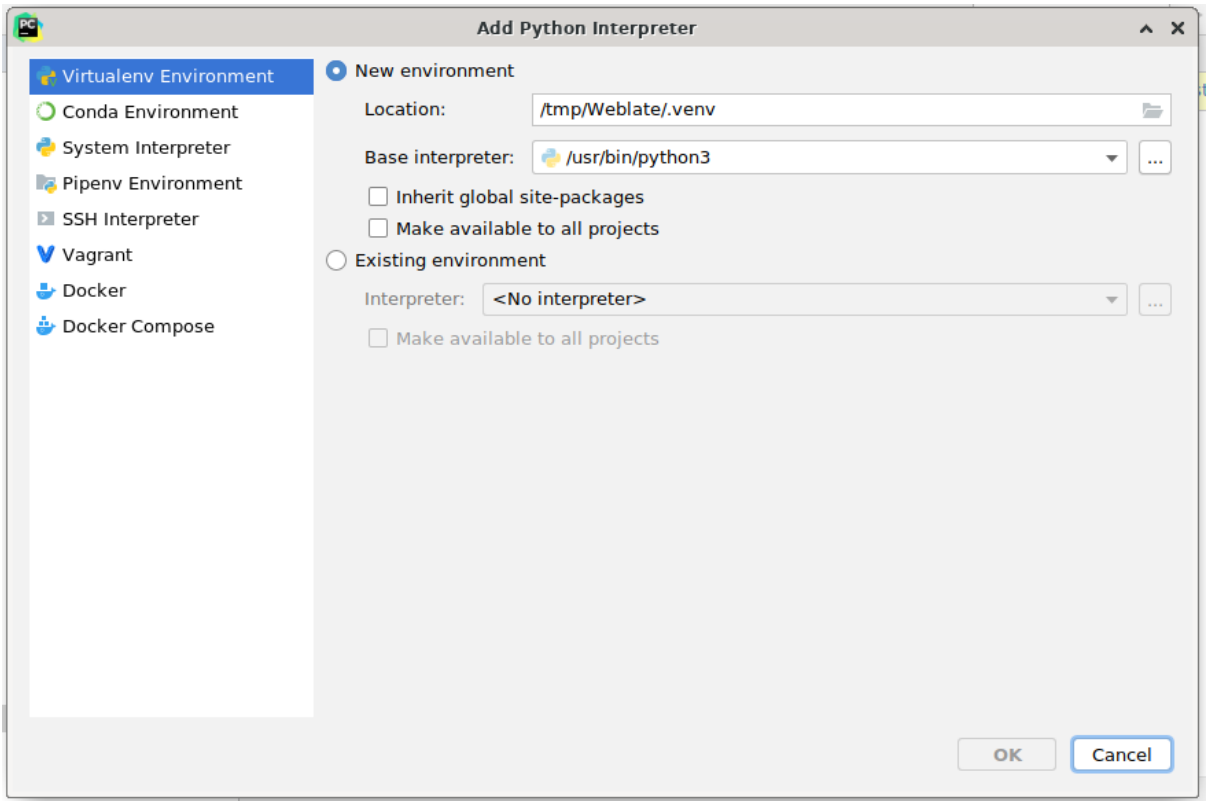
3.2.4 Coding Weblate with PyCharm

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:

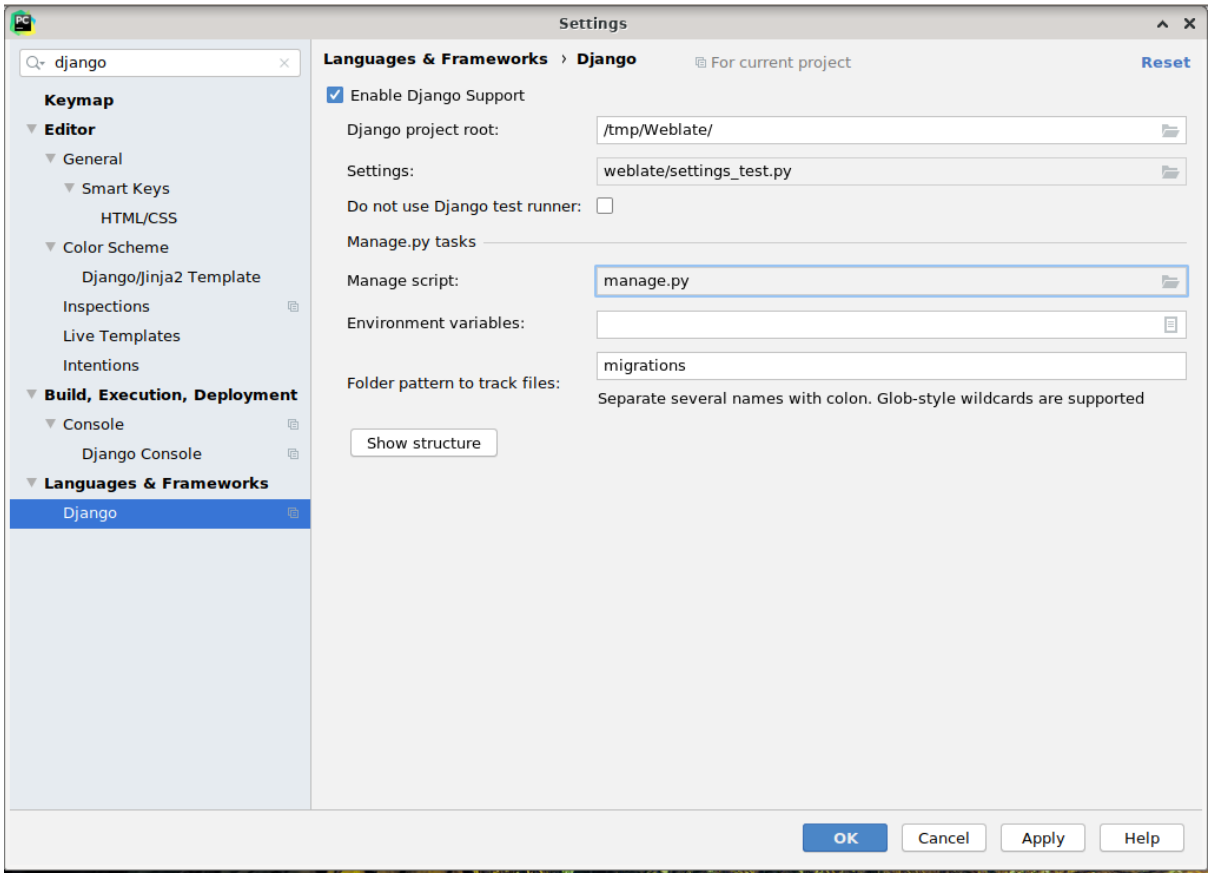


You can either choose to let PyCharm create the virtualenv for you, or select an already existing one:



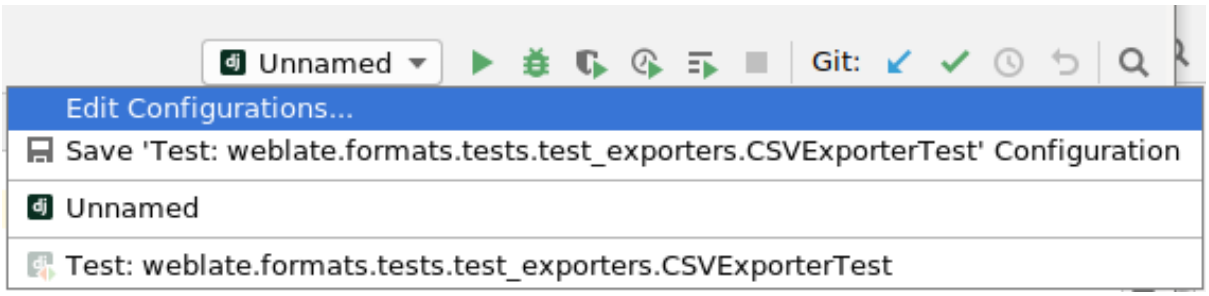
Vergessen Sie nicht, die Abhängigkeiten zu installieren, sobald der Interpreter eingestellt ist: Entweder über die Konsole (die Konsole der IDE verwendet standardmäßig direkt Ihre virtualenv), oder über die Benutzeroberfläche, wenn Sie eine Warnung über fehlende Abhängigkeiten erhalten.

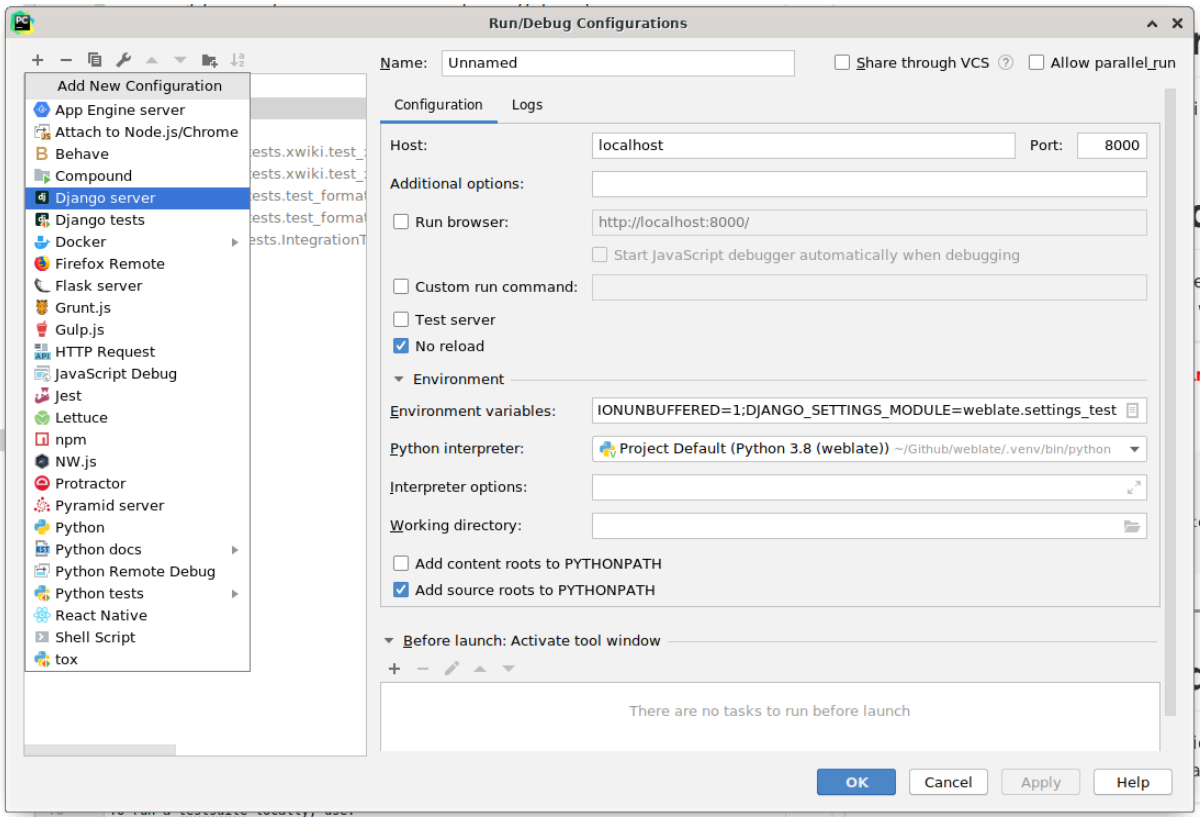
The second step is to set the right info to use Django natively inside PyCharm: The idea is to be able to immediately trigger the unit tests in the IDE. For that you need to specify the root path of the Django project and the path to its settings:



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:





Hinweis: Seien Sie vorsichtig mit der Property namens *No reload*: Diese verhindert, dass der Server live neu geladen wird, wenn Sie Dateien ändern. Dadurch bleiben die vorhandenen Debugger-Haltepunkte erhalten, obwohl sie normalerweise beim Neuladen des Servers verworfen würden.

3.2.5 Bootstrapping your devel instance

You might want to use `import_demo` to create demo translations and `createadmin` to make an admin user.

3.3 Weblate source code

Weblate is developed on GitHub. You are welcome to fork the code and open pull requests. Patches in any other form are welcome too.

Siehe auch:

Check out *Weblate-Interna* to see how Weblate looks from inside.

3.3.1 Coding guidelines

Any code for Weblate should be written with Security by Design Principles in mind.

Any code should come with documentation explaining the behavior. Don't forget documenting methods, complex code blocks, or user visible features.

Any new code should utilize **PEP 484** type hints. We're not checking this in our CI yet as existing code does not yet include them.

3.3.2 Coding standard and linting the code

The code should follow PEP-8 coding guidelines and should be formatted using **black** code formatter.

To check the code quality, you can use **flake8**, the recommended plugins are listed in `.pre-commit-config.yaml` and its configuration is placed in `setup.cfg`.

The easiest approach to enforce all this is to install pre-commit. The repository contains configuration for it to verify the committed files are sane. After installing it (it is already included in the `requirements-lint.txt`) turn it on by running `pre-commit install` in Weblate checkout. This way all your changes will be automatically checked.

You can also trigger check manually, to check all files run:

```
pre-commit run --all
```

3.4 Debugging Weblate

Bugs can behave as application crashes or as various misbehavior. You are welcome to collect info on any such issue and submit it to the issue tracker.

3.4.1 Debugmodus

Turning on debug mode will make the exceptions show in the web browser. This is useful to debug issues in the web interface, but not suitable for a production environment because it has performance consequences and might leak private data.

In a production environment, use `ADMINS` to receive e-mails containing error reports, or configure error collection using a third-party service.

Siehe auch:

Disable debug mode, *Properly configure admins*, *Collecting error reports*

3.4.2 Weblate logs

Weblate can produce detailed logs of what is going on in the background. In the default configuration it uses syslog and that makes the log appear either in `/var/log/messages` or `/var/log/syslog` (depending on your syslog daemon configuration).

The Celery process (see *Background tasks using Celery*) usually produces its own logs as well. The example system-wide setups logs to several files under `/var/log/celery/`.

Docker containers log to their output (as per usual in the Docker world), so you can look at the logs using `docker-compose logs`.

Siehe auch:

Sample configuration contains `LOGGING` configuration.

3.4.3 Not processing background tasks

A lot of things are done in the background by Celery workers. If things like sending out e-mails or component removal does not work, there might a related issue.

Dinge, die in diesem Fall zu überprüfen sind:

- Überprüfen Sie, ob der Celery-Prozess ausgeführt wird, siehe *Background tasks using Celery*
- Check the Celery queue status, either in *Verwaltungsoberfläche*, or using `celery_queues`
- Look in the Celery logs for errors (see *Weblate logs*)

3.4.4 Not receiving e-mails from Weblate

You can verify whether outgoing e-mail is working correctly by using the `sendtestemail` management command (see *Invoking management commands* for instructions on how to invoke it in different environments) or by using *Verwaltungsoberfläche* under the *Tools* tab.

These send e-mails directly, so this verifies that your SMTP configuration is correct (see *Configuring outgoing e-mail*). Most of the e-mails from Weblate are however sent in the background and there might be some issues with Celery involved as well, please see *Not processing background tasks* for debugging that.

3.4.5 Analyzing application crashes

In case the application crashes, it is useful to collect as much info about the crash as possible. This can be achieved by using third-party services which can collect such info automatically. You can find info on how to set this up in *Collecting error reports*.

3.4.6 Silent failures

Lots of tasks are offloaded to Celery for background processing. Failures are not shown in the user interface, but appear in the Celery logs. Configuring *Collecting error reports* helps you to notice such failures easier.

3.4.7 Performance issues

In case Weblate performs badly in some scenario, please collect the relevant logs showing the issue, and anything that might help figuring out where the code might be improved.

In case some requests take too long without any indication, you might want to install dogslow along with *Collecting error reports* and get pinpointed and detailed tracebacks in the error collection tool.

In case the slow performance is linked to the database, you can also enable logging of all database queries using following configuration after enabling `DEBUG`:

```
LOGGING["loggers"]["django.db.backends"] = {"handlers": ["console"], "level":
↪"DEBUG"}
```

3.5 Weblate-Interna

Bemerkung: Dieses Kapitel gibt Ihnen einen grundlegenden Überblick über die Interna von Weblate.

Weblate leitet den größten Teil seiner Codestruktur von Django ab und basiert auf diesem.

3.5.1 Struktur des Verzeichnisses

Schneller Überblick über die Verzeichnisstruktur des Weblate-Hauptrepositorys:

`docs`
Quellcode für diese Dokumentation, die mit Sphinx erstellt werden kann.

`dev-docker`
Docker-Code zum Betrieb des Entwicklungsservers, siehe *Running Weblate locally in Docker*.

`weblate`
Quellcode von Weblate als Django Anwendung, siehe *Weblate-Interna*.

`weblate/static`
Client-Dateien (CSS, Javascript und Bilder), siehe *Weblate frontend*.

3.5.2 Module

Weblate besteht aus mehreren Django-Anwendungen (einige sind optional, siehe *Optional Weblate modules*):

`accounts`

Benutzerkonto, Profile und Benachrichtigungen.

`addons`

Erweiterungen um das Verhalten von Weblate zu verändern, siehe *Erweiterungen*.

`api`

API basierend auf Django REST framework.

`auth`

Authentifizierung und Berechtigungen.

`billing`

Das optionale Modul *Abrechnung*.

`checks`

Translation string *Qualitätsprüfungen* module.

`fonts`

Font rendering checks module.

`formats`

File format abstraction layer based on translate-toolkit.

`gitexport`

Das optionale Modul *Git exporter*.

`lang`

Modul zur Definition von Sprach- und Pluralmodellen.

`legal`

Das optionale Modul *Rechtliche Grundlagen*.

`machinery`

Integration von automatischen Übersetzungsdiensten.

`memory`

Integrierter Übersetzungsspeicher, siehe :ref:'translation-memory'.

`screenshots`

Verwaltung von Bildschirmfotos und OCR-Modul.

`trans`

Hauptmodul für Übersetzungen.

`utils`

Verschiedene Hilfsprogramme.

`vcs`

Abstraktion der Versionsverwaltung.

`wladmin`

Anpassung der Django-Adminoberfläche.

3.6 Developing add-ons

Erweiterungen are way to customize localization workflow in Weblate.

class `weblate.addons.base.`**`BaseAddon`**(*storage=None*)

Base class for Weblate add-ons.

classmethod can_install(*component*, *user*)

Überprüfen Sie, ob die Erweiterung mit der angegebenen Komponente kompatibel ist.

configure(*settings*)

Konfiguration speichern.

daily(*component*)

Hook triggered daily.

classmethod get_add_form(*user*, *component*, ***kwargs*)

Konfigurationsformular zum Hinzufügen einer neuen Erweiterung zurückgeben.

get_settings_form(*user*, ***kwargs*)

Konfigurationsformular für diese Erweiterung zurückgeben.

post_add(*translation*)

Hook wird ausgelöst, nachdem eine neue Übersetzung hinzugefügt wurde.

post_commit(*component*)

Hook wird ausgelöst, nachdem Änderungen an das Repository übergeben wurden.

post_push(*component*)

Hook wird ausgelöst, nachdem das Repository Upstream gepusht wurde.

post_update (*component*, *previous_head: str*, *skip_push: bool*)

Hook triggered after repository is updated from upstream.

Parameter

- **previous_head** (*str*) – HEAD of the repository prior to update, can be blank on initial clone.
- **skip_push** (*bool*) – Ob die Erweiterungsoperation das Pushen von Änderungen im Upstream überspringen soll. Normalerweise können Sie dies den zugrunde liegenden Methoden als `commit_and_push` oder `commit_pending` übergeben.

pre_commit (*translation*, *author*)

Hook wird ausgelöst, bevor Änderungen an das Repository übergeben werden.

pre_push (*component*)

Hook triggered before repository is pushed upstream.

pre_update (*component*)

Hook triggered before repository is updated from upstream.

save_state ()

Speichern Sie Informationen über den Zustand der Erweiterung.

store_post_load (*translation*, *store*)

Hook wird ausgelöst, nachdem eine Datei analysiert wurde.

It receives an instance of a file format class as a argument.

This is useful to modify file format class parameters, for example adjust how the file will be saved.

unit_pre_create (*unit*)

Hook triggered before new unit is created.

Hier ein Beispiel für eine Erweiterung:

```
#
# Copyright © 2012-2022 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <https://www.gnu.org/licenses/>.
#


from django.utils.translation import gettext_lazy as _

from weblate.addons.base import BaseAddon
from weblate.addons.events import EVENT_PRE_COMMIT


class ExampleAddon(BaseAddon):
    # Filter for compatible components, every key is
```

(Fortsetzung auf der nächsten Seite)

(Fortsetzung der vorherigen Seite)

```
    # 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 Weblate frontend

The frontend is currently built using Bootstrap, jQuery and few third party libraries.

3.7.1 Unterstützte Browser

Weblate supports the latest, stable releases of all major browsers and platforms.

Alternative browsers which use the latest version of WebKit, Blink, or Gecko, whether directly or via the platform's web view API, are not explicitly supported. However, Weblate should (in most cases) display and function correctly in these browsers as well.

Older browsers might work, but some features might be limited.

3.7.2 Dependency management

The yarn package manager is used to update third party libraries. The configuration lives in `scripts/yarn` and there is a wrapper script `scripts/yarn-update` to upgrade the libraries, build them and copy to correct locations in `weblate/static/vendor`, where all third partly frontend code is located. The Weblate specific code should be placed directly in `weblate/static` or feature specific subdirectories (for example `weblate/static/editor`).

Adding new third-party library typically consists of:

```
# Add a yarn package
yarn --cwd scripts/yarn add PACKAGE
# Edit the script to copy package to the static folder
edit scripts/yarn-update
# Run the update script
./scripts/yarn-update
# Add files to git
git add .
```

3.7.3 Coding style

Weblate relies on Prettier for the code formatting for both JavaScript and CSS files.

We also use ESLint to check the JavaScript code.

3.7.4 Lokalisierung

Should you need any user visible text in the frontend code, it should be localizable. In most cases all you need is to wrap your text inside `gettext` function, but there are more complex features available:

```
document.write(gettext('this is to be translated'));

var object_count = 1 // or 0, or 2, or 3, ...
s = 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'
```

Siehe auch:

Translation topic in the Django documentation

3.7.5 Icons

Weblate currently uses material design icons. In case you are looking for new symbol, check Material Design Icons or Material Design Resources.

Additionally, there is `scripts/optimize-svg` to reduce size of the SVG as most of the icons are embedded inside the HTML to allow styling of the paths.

3.8 Melden von Problemen in Weblate

Weblate Issue Tracker wird auf GitHub gehostet.

Sie können dort gerne alle Probleme melden, die Sie haben, oder Verbesserungsvorschläge für Weblate machen. Es gibt verschiedene Vorlagen, die Sie bequem durch den Problembericht führen.

Wenn Sie ein Sicherheitsproblem in Weblate gefunden haben, lesen Sie bitte den Abschnitt Sicherheit unten.

Wenn Sie sich nicht sicher sind, ob es sich um einen Fehlerbericht oder eine Funktionsanfrage handelt, können Sie Diskussionen versuchen.

3.8.1 Sicherheitslücken

Um der Community Zeit zu geben, zu reagieren und zu aktualisieren, werden Sie dringend gebeten, alle Sicherheitsprobleme privat zu melden. HackerOne wird verwendet, um Sicherheitsprobleme zu behandeln, und kann direkt unter HackerOne gemeldet werden. Sobald Sie das Problem dort gemeldet haben, hat die Gemeinschaft nur eine begrenzte, aber ausreichende Zeit, um den Vorfall zu lösen.

Alternativ können Sie sich auch an security@weblate.org wenden, das ebenfalls auf HackerOne veröffentlicht wird.

Wenn Sie HackerOne, aus welchen Gründen auch immer, nicht verwenden wollen, können Sie den Bericht per E-Mail an michal@cihar.com senden. Sie können ihn mit diesem PGP-Schlüssel *3CB 1DF1 EF12 CF2A C0EE 5A32 9C27 B313 42B7 511D* verschlüsseln. Sie können den PGP-Schlüssel auch von Keybase beziehen.

Bemerkung: Weblate ist für viele Dinge von Komponenten Dritter abhängig. Falls Sie eine Schwachstelle finden, die eine dieser Komponenten betrifft, melden Sie diese bitte direkt an das entsprechende Projekt.

Einige davon sind:

- Django
- Django REST-Framework
- Python Social Auth

3.9 Weblate testsuite and continuous integration

Testsuites exist for most of the current code, increase coverage by adding testcases for any new functionality, and verify that it works.

3.9.1 Continuous integration

Current test results can be found on GitHub Actions and coverage is reported on Codecov.

There are several jobs to verify different aspects:

- Unit tests
- Documentation build and external links
- Migration testing from all supported releases
- Code linting
- Setup verification (ensures that generated dist files do not miss anything and can be tested)

The configuration for the CI is in `.github/workflows` directory. It heavily uses helper scripts stored in `ci` directory. The scripts can be also executed manually, but they require several environment variables, mostly defining Django settings file to use and database connection. The example definition of that is in `scripts/test-database`:

```
# Simple way to configure test database from environment

# Database backend to use postgresql / mysql / mariadb
export CI_DATABASE=${1:-postgresql}

# Database server configuration
export CI_DB_USER=weblate
export CI_DB_PASSWORD=weblate
export CI_DB_HOST=127.0.0.1

# Django settings module to use
export DJANGO_SETTINGS_MODULE=weblate.settings_test
```

The simple execution can look like:

```
. scripts/test-database
./ci/run-migrate
./ci/run-test
./ci/run-docs
```

3.9.2 Local testing

To run a testsuite locally, use:

```
DJANGO_SETTINGS_MODULE=weblate.settings_test ./manage.py test
```

Hinweis: You will need a database (PostgreSQL) server to be used for tests. By default Django creates separate database to run tests with `test_` prefix, so in case your settings is configured to use `weblate`, the tests will use `test_weblate` database. See *Database setup for Weblate* for setup instructions.

The `weblate/settings_test.py` is used in CI environment as well (see *Continuous integration*) and can be tuned using environment variables:

```
# Simple way to configure test database from environment

# Database backend to use postgresql / mysql / mariadb
export CI_DATABASE=${1:-postgresql}

# Database server configuration
export CI_DB_USER=weblate
export CI_DB_PASSWORD=weblate
export CI_DB_HOST=127.0.0.1

# Django settings module to use
export DJANGO_SETTINGS_MODULE=weblate.settings_test
```

Prior to running tests you should collect static files as some tests rely on them being present:

```
DJANGO_SETTINGS_MODULE=weblate.settings_test ./manage.py collectstatic
```

You can also specify individual tests to run:

```
DJANGO_SETTINGS_MODULE=weblate.settings_test ./manage.py test weblate.gitexport
```

Hinweis: The tests can also be executed inside developer docker container, see *Running Weblate locally in Docker*.

Siehe auch:

See Testing in Django for more info on running and writing tests for Django.

3.10 Data schemas

Weblate uses JSON Schema to define layout of external JSON files.

3.10.1 Weblate-Übersetzungsspeicher-Schema

https://weblate.org/schemas/weblate-memory.schema.json		
type	*array*	
items	*The Translation Memory Item*	
	type	*object*
	properties	
	• **category**	*The String Category*
		1 is global, 2 is shared, 10000000+ are project specific, 20000000+ are user specific

Fortsetzung auf der nächsten Seite

Tab. 1 – Fortsetzung der vorherigen Seite

		type	*integer*
		examples	1
		minimum	0
		default	1
	• **origin**	*The String Origin*	
		Filename or component name	
		type	*string*
		examples	test.tmx
			project/component
		default	
	• **source**	*The Source String*	
		type	*string*
		examples	Hello
		minLength	1
		default	
	• **source_language**	*The Source Language*	
		ISO 639-1 / ISO 639-2 / IETF BCP 47	
		type	*string*
		examples	en
		pattern	^[^]+$
		default	
	• **target**	*The Target String*	
		type	*string*
		examples	Ahoj
		minLength	1
		default	
	• **target_language**	*The Target Language*	
		ISO 639-1 / ISO 639-2 / IETF BCP 47	
		type	*string*
		examples	cs
		pattern	^[^]+$
		default	
	additionalProperties	False	
definitions			

Siehe auch:

Übersetzungsspeicher, `dump_memory`, `import_memory`

3.10.2 Weblate user data export

https://weblate.org/schemas/weblate-userdata.schema.json			
type	*object*		
properties			
• **basic**	*Basic*		
	type	*object*	
	properties		
	• **username**	*Username*	
		type	*string*
		examples	admin
		default	
	• **full_name**	*Full name*	
		type	*string*
		examples	Weblate-Administrator
		default	

Fortsetzung auf der nächsten Seite

Tab. 2 – Fortsetzung der vorherigen Seite

	• **email**	*E-mail*	
		type	*string*
		examples	noreply@example.com
		default	
		format	E-Mail
	• **date_joined**	*Date joined*	
		type	*string*
		examples	2019-11-18T18:53:54.862Z
		default	
		format	date-time
	additionalProperties	False	
• **profile**	*Profile*		
	type	*object*	
	properties		
	• **language**	*Language*	
		type	*string*
		examples	cs
		pattern	^[^]*$
		default	
	• **suggested**	*Number of suggested strings*	
		type	*integer*
		examples	1
		default	0
	• **translated**	*Number of translated strings*	
		type	*integer*
		examples	24
		default	0
	• **uploaded**	*Anzahl der hochgeladenen Bildschirmfotos*	
		type	*integer*
		examples	1
		default	0
	• **hide_completed**	*Hide completed translations on the dashboard*	
		type	*boolean*
		examples	False
		default	True
	• **secondary_in_zen**	*Show secondary translations in the Zen mode*	
		type	*boolean*
		examples	True
		default	True
	• **hide_source_secondary**	*Hide source if a secondary translation exists*	
		type	*boolean*
		examples	False
		default	True
	• **editor_link**	*Editor link*	
		type	*string*
		examples	
		pattern	^.*$
		default	
	• **translate_mode**	*Translation editor mode*	
		type	*integer*
		examples	0
		default	0
	• **zen_mode**	*Zen editor mode*	
		type	*integer*
		examples	0

Fortsetzung auf der nächsten Seite

Tab. 2 – Fortsetzung der vorherigen Seite

		default	0	
	• **special_chars**	*Special characters*		
		type	*string*	
		examples		
		pattern	^.*$	
		default		
	• **dashboard_view**	*Default dashboard view*		
		type	*integer*	
		examples	1	
		default	0	
	• **dashboard_component_list**	*Default component list*		
		default	null	
		anyOf	type	*null*
			type	*integer*
	• **languages**	*Translated languages*		
		type	*array*	
		default		
		items	*Language code*	
			type	*string*
			examples	cs
			pattern	^.*$
			default	
	• **secondary_languages**	*Secondary languages*		
		type	*array*	
		default		
		items	*Language code*	
			type	*string*
			examples	sk
			pattern	^.*$
			default	
	• **watched**	*Watched projects*		
		type	*array*	
		default		
		items	*Project slug*	
			type	*string*
			examples	weblate
			pattern	^.*$
			default	
	additionalProperties	False		
• **auditlog**	*Audit log*			
	type	*array*		
	default			
	items	*Items*		
		type	*object*	
		properties		
		• **address**	*IP address*	
			type	*string*
			examples	127.0.0.1
			pattern	^.*$
			default	
		• **user_agent**	*User agent*	
			type	*string*
			examples	PC / Linux / Firefox 70.0
			pattern	^.*$

Fortsetzung auf der nächsten Seite

Tab. 2 – Fortsetzung der vorherigen Seite

			default	
		• **timestamp**	*Timestamp*	
			type	*string*
			examples	2019-11-18T18:58:30.845Z
			default	
			format	date-time
		• **activity**	*Activity*	
			type	*string*
			examples	anmelden
			pattern	^.*$
			default	
		additionalProperties	False	
definitions				

Siehe auch:

Benutzerprofil, `dumpuserdata`

3.11 Releasing Weblate

3.11.1 Releasing schedule

Weblate hat einen zweimonatigen Veröffentlichungszyklus für neue Versionen (x.y). Darauf folgen in der Regel eine Reihe von Bugfix-Releases zur Behebung von Problemen, die sich eingeschlichen haben (x.y.z).

Die Änderung der Hauptversion zeigt an, dass der Upgrade-Prozess diese Version nicht überspringen kann - Sie müssen immer auf x.0 aktualisieren, bevor Sie auf höhere x.y-Versionen aktualisieren.

Siehe auch:

Upgrading Weblate

3.11.2 Release-Planung

Die Funktionen für kommende Versionen werden mit Hilfe von GitHub-Meilensteinen gesammelt, Sie können unsere Roadmap unter <https://github.com/WeblateOrg/weblate/milestones> einsehen.

3.11.3 Release-Prozess

Vor Release zu prüfende Dinge:

1. Prüfen Sie neu übersetzte Sprachen mit **`./scripts/list-translated-languages`**.
2. Endgültige Version mit **`./scripts/prepare-release`** einstellen.
3. Make sure screenshots are up to date **`make -j 12 -C docs update-screenshots`**.
4. Merge any possibly pending translations **`wlc push; git remote update; git merge origin/weblate`**

Perform the release:

5. Create a release **`./scripts/create-release --tag`** (see below for requirements).

Post release manual steps:

6. Update Docker image.

7. Close GitHub milestone.
8. Once the Docker image is tested, add a tag and push it.
9. Update Helm chart to new version.
10. Include new version in `.github/workflows/migrations.yml` to cover it in migration testing.
11. Increase version in the website download links.
12. Increase version in the repository by **`./scripts/set-version`**.

To create tags using the **`./scripts/create-release`** script you will need following:

- GnuPG with private key used to sign the release
- Push access to Weblate git repositories (it pushes tags)
- Configured **`hub`** tool and access to create releases on the Weblate repo
- SSH access to Weblate download server (the Website downloads are copied there)

3.12 Security and privacy

Tipp: Bei Weblate erhält Sicherheit eine Umgebung, welche die Privatsphäre unserer Benutzer wertschätzt.

Development of Weblate adheres to the Best Practices of the Linux Foundation's Core Infrastructure Initiative.

Siehe auch:

Sicherheitslücken

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.

Hinweis: There might be vulnerabilities in third-party libraries which do not affect Weblate, so those are not addressed by releasing bugfix versions of Weblate.

3.12.2 Docker container security

The Docker containers are regularly scanned using Anchore and Trivy security scanners.

This allows us to detect vulnerabilities early and release improvements quickly.

You can get the results of these scans at GitHub — they are stored as artifacts on our CI in the SARIF format (Static Analysis Results Interchange Format).

Siehe auch:

Continuous integration

3.13 Contributing to Weblate modules

Besides the main repository, Weblate consists of several Python modules. All these follow same structure and this documentation covers them all.

For example, this covers:

- wlc, Python client library, see *Weblate Client*
- translation-finder, used to discover translatable files in the repository
- language-data, language definitions for Weblate, see *Language definitions*

3.13.1 Coding guidelines

Any code for Weblate should be written with Security by Design Principles in mind.

Any code should come with documentation explaining the behavior. Don't forget documenting methods, complex code blocks, or user visible features.

Any new code should utilize **PEP 484** type hints. We're not checking this in our CI yet as existing code does not yet include them.

3.13.2 Running tests

The tests are executed using **`py.test`**. First you need to install test requirements:

```
pip install -r requirements-test.txt
```

You can then execute the testsuite in the repository checkout:

```
py.test
```

Siehe auch:

The CI integration is very similar to *Weblate testsuite and continuous integration*.

3.13.3 Coding standard and linting the code

The code should follow PEP-8 coding guidelines and should be formatted using **`black`** code formatter.

To check the code quality, you can use **`flake8`**, the recommended plugins are listed in `.pre-commit-config.yaml` and its configuration is placed in `setup.cfg`.

The easiest approach to enforce all this is to install pre-commit. The repository contains configuration for it to verify the committed files are sane. After installing it (it is already included in the `requirements-lint.txt`) turn it on by running `pre-commit install` in Weblate checkout. This way all your changes will be automatically checked.

You can also trigger check manually, to check all files run:

```
pre-commit run --all
```

Siehe auch:

Weblate source code

3.14 Über Weblate

3.14.1 Project goals

Web-based continuous localization tool with tight *Integration der Versionsverwaltung* supporting a wide range of *file formats*, making it easy for translators to contribute.

3.14.2 Projektname

„Weblate“ is a portmanteau of the words „web“ and „translate“.

3.14.3 Projektseite

Die Einstiegsseite ist https://weblate.org und es gibt einen Cloud-gehosteten Dienst unter https://hosted.weblate.org. Die Dokumentation kann unter https://docs.weblate.org nachgelesen werden.

3.14.4 Project logos

Die Projektlogos und andere Grafiken sind unter https://github.com/WeblateOrg/graphics verfügbar.

3.14.5 Leadership

Dieses Projekt wird von Michal Čihař betreut, den Sie unter michal@cihar.com erreichen können.

3.14.6 Authors

Weblate wurde von Michal Čihař gegründet. Seit seiner Gründung im Jahr 2012 haben Tausende von Menschen dazu beigetragen.

3.15 Lizenz

KAPITEL 4

Change History

4.1 Weblate 4.14

Released on August 22nd 2022.

- Nachverfolgung von Add-On-Änderungen in einem Verlauf.
- Fixed parsing translation from Windows RC, HTML and text files.
- Extended language code style configuration options.
- Added support for pluralse updated in recent CLDR releases.
- Reduced memory usage while updating components with a lot of translations.
- Added support for translation domain in SAP Translation Hub.
- Allow absolute links in source string locations.
- Improved operation behind some reverse proxies.
- Exteded API to cover translation memory.
- Improved document translation workflow.
- Improved reliability of HTML and text files translation.
- Added support for project level backups.
- Improved performance and memory usage of translation memory lookups.

Alle Änderungen im Detail.

4.2 Weblate 4.13.1

Veröffentlicht am 1. Juli 2022.

- Tracking-Vorschläge im Verlauf behoben.
- Fixed parsing reverse proxy info from Cloudflare.
- Make parse error lock a component from translating.
- Fixed configuring intermediate file in the discovery add-on.
- Fixed DeepL translations behavior with placeholders.
- Fixed untranslating strings via API.
- Added support for removing user from a group via API.
- Fixed audit log for user invitation e-mails.
- Fixed flag names for Java formatting strings.

Alle Änderungen im Detail.

4.3 Weblate 4.13

Veröffentlicht am 15. Juni 2022.

- Changed behavior of updating language names.
- Added pagination to projects listing.
- API for creating new units now returns information about newly created unit.
- Component discovery now supports configuring an intermediate language.
- Added fixed encoding variants to CSV formats.
- Changed handling of context and location for some formats to better fit underlying implementation.
- Added support for ResourceDictionary format.
- Improved progress bar colors for color blind.
- Fixed variants cleanup on string removal.
- Compatibility with Django 4.1.
- Added support for storing escaped XML elements in XLIFF.
- Improved formatting of placeholder check errors.
- Redirect /.well-known/change-password to /accounts/password/.
- Machine translation services are now configurable per project.
- Added separate permission for resolving comments and grant it to the *Review strings* role.
- Added support for storing alternative translations in the CSV file.
- The placeholders check can now be case-insensitive as well.

Alle Änderungen im Detail.

4.4 Weblate 4.12.2

Veröffentlicht am 11. Mai 2022.

- Fixed rebuilding project translation memory for some components.
- Fixed sorting components by untranslated strings.
- Fixed possible loss of translations while adding new language.
- Ensure Weblate SSH key is generated during migrations.

Alle Änderungen im Detail.

4.5 Weblate 4.12.1

Veröffentlicht am 29. April 2022.

- Fixed pull request message title.
- Improved syntax error handling in Fluent format.
- Fixed avatar display in notification e-mails.
- Add support for web monetization.
- Fixed removal of stale source strings when removing translations.

Alle Änderungen im Detail.

4.6 Weblate 4.12

Veröffentlicht am 20. April 2022.

- Added support for Amharic in *Nicht übereinstimmender Punkt*.
- Added support for Burmese in *Nicht übereinstimmendes Fragezeichen*.
- Extended options of the *Erzeugung von Pseudolokalen* add-on.
- Added `ignore-all-checks` flag to ignore all quality checks on a string.
- Avoid *Erzeugung von Pseudolokalen* add-on to trigger failing checks.
- Added support for *Gitea pull requests*.
- Added Linux style language code to *Stil des Sprachcodes*.
- Added support for rebuilding project translation memory.
- Improved API for creating components from a file.
- Add copy and clone buttons to other translations.
- Make merge request message configurable at component level.
- Improved maximal length restriction behavior with XML tags.
- Fixed loading Fluent files with additional comments.

Alle Änderungen im Detail.

4.7 Weblate 4.11.2

Veröffentlicht am 4. März 2022.

- Fixed corrupted MO files in the binary release.

Alle Änderungen im Detail.

4.8 Weblate 4.11.1

Veröffentlicht am 4. März 2022.

- Fixed missing sanitizing of arguments to Git and Mercurial - CVE-2022-23915, see GHSA-3872-f48p-pxqj for more details.
- Fixed loading fuzzy strings from CSV files.
- Added support for creating teams using the API.
- Fixed user mention suggestions display.
- The project tokens access can now be customized.

Alle Änderungen im Detail.

4.9 Weblate 4.11

Veröffentlicht am 25. Februar 2022.

- Fixes stored XSS - CVE-2022-24710, see GHSA-6jp6-9rf9-gc66 for more details.
- Installation von Erweiterungen über die API korrigiert.
- Renamed *Strings needing action* to *Unfinished strings*.
- Fixed false positives from *ICU MessageFormat-Syntax*.
- Indicate lock and contributor agreement on other occurrences listing.
- Fixed updating PO files with obsolete strings or missing plurals.
- Die Kompatibilität der Squash-Erweiterung mit Gerrit wurde verbessert.
- Automatically initialize user languages based on the Accept-Language header.
- Improved error handling on string removal.
- Weblate erfordert jetzt Python 3.7 oder neuer.
- Einige Schreiboperationen mit Projekt-Token-Authentifizierung wurden behoben.
- Die Nachverfolgung des Zeichenkettenstatus bei Änderungen der Zeichenkette im Repository wurde korrigiert.
- Zeichenkettenveränderungen im Repository verfolgen.
- Sticky header on translations listing to improve navigation.
- Nicht übersetzte Zeichenketten in *Java properties* behoben.
- Fixed Git operation with non-ascii branch names.
- Neue Erweiterung *Übersetzung mit Quelle vorbelegen*.
- *Merge without fast-forward Git-Strategie* hinzugefügt.
- Erweiterungstrigger für neu hinzugefügte Zeichenketten *Automatische Übersetzung* repariert.
- Improved punctuation checks for Burmese.

- Unterstützung zur Festlegung benutzerdefinerter Teams auf Projektebene, um Benutzerzugriffe zu gewähren, hinzugefügt, siehe *Verwaltung der Zugriffssteuerung nach Projekt*.
- Links zur Dokumentation bei Warnungen hinzugefügt.
- Docker-Container aktiviert bei Bedarf automatisch TLS/SSL für ausgehende E-Mails.
- Unterstützung für die Suche nach geklärten Kommentaren ergänzt.
- Unterstützung für borgbackup 1.2 hinzugefügt.
- Die Verwendung von *Automatisch übersetzt* wurde korrigiert.

Alle Änderungen im Detail.

4.10 Weblate 4.10.1

Veröffentlicht am 22. Dezember 2021.

- Dokumentierte Änderungen, die durch das Upgrade auf Django 4.0 eingeführt wurden.
- Die Anzeige von *Automatisch übersetzt* wurde korrigiert.
- Die API-Anzeige einer Branch in Komponenten mit einem gemeinsamen Repository wurde korrigiert.
- Verbesserte Analyse der fehlgeschlagenen Push-Warnung.
- Das manuelle Bearbeiten der Seite beim Durchsuchen von Änderungen wurde behoben.
- Die Genauigkeit von *Kashida-Buchstabe verwendet* wurde verbessert.
- Der Weblate-Docker-Container verwendet jetzt Python 3.10.

Alle Änderungen im Detail.

4.11 Weblate 4.10

Veröffentlicht am 16. Dezember 2021.

- Unterstützung für Formalitäten und Platzhalter mit DeepL hinzugefügt.
- Massenbearbeitung sowie Suchen und Ersetzen sind jetzt auf Projekt- und Sprachebene verfügbar.
- Filterung beim Suchen und Ersetzen hinzugefügt.
- Fixed: „Perform automatic translation“ privilege is no longer part of the *Languages* group.
- „Perform automatic translation“ is in the *Administration* and the new *Automatic translation* group.
- Das Erzeugen von XLSX-Dateien mit Sonderzeichen wurde korrigiert.
- Added ability to the GitHub authentication backend to check if the user belongs to a specific GitHub organization or team.
- Verbesserte Rückmeldung über ungültige an die API übergebene Parameter.
- Unterstützung für projektspezifische Zugriffstoken für die API hinzugefügt.
- Fixed string removal in some cases.
- Die Übersetzung neu hinzugefügter Zeichenketten wurde korrigiert.
- Label automatically translated strings to ease their filtering.

Alle Änderungen im Detail.

4.12 Weblate 4.9.1

Veröffentlicht am 19. November 2021.

- Fixed upload of monolingual files after changing template.
- Improved handling of whitespace in flags.
- Unterstützung für die Filterung in der Download-API hinzugefügt.
- Die Statistikanzeige beim Hinzufügen neuer Übersetzungen wurde korrigiert.
- Mitigate issues with GitHub SSH key change.

Alle Änderungen im Detail.

4.13 Weblate 4.9

Veröffentlicht am 10. November 2021.

- Provide more details for events in history.
- Verbesserte Darstellung des Verlaufs.
- Verbesserte Leistung der Übersetzungsseiten.
- Unterstützung für die Einschränkung des Downloads von Übersetzungsdateien wurde hinzugefügt.
- The `safe-html` can now understand Markdown when used with `md-text`.
- The `max-length` tag now ignores XML markup when used with `xml-text`.
- Fixed dimensions of rendered texts in *Maximaler Umfang der Übersetzung*.
- Lowered app store title length to 30 to assist with upcoming Google policy changes.
- Added support for customizing SSH invocation via *SSH_EXTRA_ARGS*.
- Qualitätsprüfungen für ICU-MessageFormat wurden hinzugefügt.
- Improved error condition handling in machine translation backends.
- Highlight unusual whitespace characters in the strings.
- Option hinzugefügt, um beim Bearbeiten auf der übersetzten Zeichenfolge zu bleiben.
- Added support for customizing Borg invocation via *BORG_EXTRA_ARGS*.
- Die Erzeugung von MO-Dateien für einsprachige Übersetzungen wurde korrigiert.
- Added API endpoint to download all component translations as a ZIP file.
- Unterstützung für Python 3.10 hinzugefügt.
- Added support for resending e-mail invitation from the management interface.

Alle Änderungen im Detail.

4.14 Weblate 4.8.1

Veröffentlicht am 10. September 2021.

- Das Entfernen von Benutzern in der Django-Adminoberfläche wurde behoben.
- Document add-on parameters in greater detail.
- Fixed JavaScript error in glossary.
- Begrenzung der Anzahl von Übereinstimmungen bei der Konsistenzprüfung.
- Improve handling of placeholders in machine translations.
- Die Erstellung von Erweiterungen über die API wurde behoben.
- Die Einstellung `PRIVACY_URL` wurde hinzugefügt, um einen Link zur Datenschutzrichtlinie in die Fußzeile einzufügen.
- Hide member e-mail addresses from project admins.
- Improved gettext PO merging in case of conflicts.
- Verbesserte Hervorhebung des Glossars.
- Improved `safe-html` flag behavior with XML checks.
- Commit-Meldungen für verknüpfte Komponenten korrigiert.

Alle Änderungen im Detail.

4.15 Weblate 4.8

Veröffentlicht am 21. August 2021.

- Unterstützung für das Apple-Stringsdict-Format wurde hinzugefügt.
- Der Operator für die exakte Suche unterscheidet jetzt bei PostgreSQL zwischen Groß- und Kleinschreibung.
- Das Speichern von Glossarerklärungen wurde in einigen Fällen korrigiert.
- Verbesserung der Dokumentation.
- Leistungsverbesserungen.
- Die Kompatibilität der Squash-Erweiterung mit Gerrit wurde verbessert.
- Das Hinzufügen von Zeichenketten zu einsprachigen Glossarkomponenten wurde korrigiert.
- Verbesserte Leistung bei der Handhabung von Varianten.
- Das gelegentliche Überspringen des Parsens von Upstream-Änderungen der Squash-Erweiterung behoben.
- Dateierweiterung für Downloads beibehalten.
- Unterstützung für das Fluent-Format wurde hinzugefügt.
- Unterstützung für die Verwendung von Tabulatoren zum Einrücken von JSON-Formaten wurde hinzugefügt.

Alle Änderungen im Detail.

4.16 Weblate 4.7.2

Veröffentlicht am 15. Juli 2021.

- Unterstützung von mehr Sprachaliasen, die für ein Projekt konfiguriert werden können.
- Die Validierung der Suchbegriffe in der API wurde korrigiert.
- Git-Exporter-URLs nach einem Domainwechsel korrigiert.
- Bereinigungsvorgang der Erweiterung für Windows RC-Dateien korrigiert.
- Möglicher Absturz bei der XLIFF-Aktualisierung behoben.

Alle Änderungen im Detail.

4.17 Weblate 4.7.1

Veröffentlicht am 30. Juni 2021.

- Verbessertes Popup beim Hinzufügen von Begriffen zum Glossar.
- Unterstützung für den automatischen Übersetzungsdienst LibreTranslate hinzugefügt.
- Es wurde eine Ratenbegrenzung für die Erstellung neuer Projekte hinzugefügt.
- Die Leistung von Dateiaktualisierungen wurde verbessert.

Alle Änderungen im Detail.

4.18 Weblate 4.7

Veröffentlicht am 17. Juni 2021.

- Verbesserte Zustandsprüfung der Konfiguration.
- Unterstützung für `object-pascal-format` in gettext PO hinzugefügt, siehe *Object Pascal-Format*.
- Umbenennung von *Nearby keys* in *Similar keys*, um den Zweck besser zu beschreiben.
- Unterstützung für *mi18n lang Dateien* hinzugefügt.
- Verbesserte Integration der SAML-Authentifizierung.
- Die Integration von *Gerrit* wurde korrigiert, um Eckfälle besser zu behandeln.
- Weblate benötigt jetzt Django 3.2.
- Das Einladen von Benutzern, wenn die E-Mail-Authentifizierung deaktiviert ist, wurde behoben.
- Verbesserte Sprachdefinitionen.
- Unterstützung für das Sperren von Benutzern für Beiträge zu einem Projekt wurde hinzugefügt.
- Die automatische Erstellung von Glossarsprachen wurde korrigiert.
- Erweiterte Dokumentation zu Erweiterungen.
- Leistungsverbesserungen für Komponenten mit verknüpften Repositorys.
- Unterstützung für kostenlose DeepL-API hinzugefügt.
- Die Benutzerverwaltung benötigt keine Django-Adminoberfläche mehr.

Alle Änderungen im Detail.

4.19 Weblate 4.6.2

Veröffentlicht am 8. Mai 2021.

- Absturz nach dem Verschieben einer gemeinsamen Komponente zwischen Projekten behoben.
- Das Hinzufügen neuer Zeichenketten zu leeren Eigenschaftendateien wurde korrigiert.
- Die Ausrichtung der Kopiersymbole in RTL-Sprachen wurde korrigiert.
- Erweiterte Zeichenkettenstatistik auf der Information-Reiterkarte.
- Die Behandlung von in Git ignorierten Übersetzungsdateien wurde korrigiert.
- Verbesserte Leistung der Metriken.
- Möglicher Fehler beim Speichern von Glossaren behoben.
- Das Verhalten der Konsistenzprüfung bei Sprachen mit unterschiedlichen Pluralregeln wurde korrigiert.

Alle Änderungen im Detail.

4.20 Weblate 4.6.1

Veröffentlicht am 2. Mai 2021.

- Veralteten Spamschutz-Code entfernt.
- Verbessern Sie die Genauigkeit der Pluralprüfung von Ausgangszeichenketten.
- Aktualisierung der Liste der Benutzeroberflächensprachen in Docker.
- Verbesserte Fehlermeldungen beim Erstellen von Pull-Anfragen.
- Die Erstellung von Pull-Anfragen auf Pagure wurde behoben.
- Das Auslösen von automatisch installierten Erweiterungen wurde behoben.
- Mögliche Caching-Probleme beim Upgrade behoben.
- Das Hinzufügen neuer Einheiten zu einsprachigen Übersetzungen mittels Hochladen wurde behoben.

Alle Änderungen im Detail.

4.21 Weblate 4.6

Veröffentlicht am 19. April 2021.

- Der Verwaltungsbefehl auto_translate hat jetzt einen Parameter zur Angabe des Übersetzungsmodus.
- Unterstützung für *Textdateien* hinzugefügt.
- Trends und Metriken für alle Objekte hinzugefügt.
- Added support for directly copying text from secondary languages.
- Datumsfilterung beim Durchsuchen von Änderungen hinzugefügt.
- Improved activity charts.
- Absender für Kontaktformular-E-Mails können jetzt konfiguriert werden.
- Verbesserte Parametervalidierung in der API der Komponentenerstellung.
- The rate limiting no longer applies to superusers.
- Die Leistung und Zuverlässigkeit der Erweiterung für die automatischen Vorschläge wurde verbessert.

- Die Ratenbegrenzung kann nun im Docker-Container angepasst werden.
- API zum Erstellen von Komponenten verwendet jetzt automatisch *Weblate internal URLs*.
- Vereinfachte Zustandsanzeige beim Auflisten von Zeichenketten.
- Passwort-Hashing verwendet jetzt standardmäßig Argon2.
- Vereinfachte Fortschrittsbalken, die den Übersetzungsstatus anzeigen.
- *Fehlende Sprachen hinzufügen* umbenannt, um den Zweck zu verdeutlichen.
- Das Speichern des Zeichenkettenstatus in XLIFF wurde korrigiert.
- Sprachübergreifende Suche hinzugefügt.
- Erstmalige Unterstützung für *Horizontale Skalierung* der Docker-Bereitstellung.

Alle Änderungen im Detail.

4.22 Weblate 4.5.3

Veröffentlicht am 1. April 2021.

- Metriksammlung korrigiert.
- Möglicher Absturz beim Hinzufügen von Zeichenketten behoben.
- Verbesserte Beispiele für Suchanfragen.
- Möglicher Verlust von neu hinzugefügten Zeichenketten beim Ersetzen des Hochladens behoben.

4.23 Weblate 4.5.2

Veröffentlicht am 26. März 2021.

- Konfigurierbarer Zeitplan für die automatische Übersetzung.
- Lua-Format-Prüfung hinzugefügt.
- Ignore format strings in the *Aufeinanderfolgende doppelte Wörter* check.
- Erlaubt das Hochladen von Bildschirmfotos von einer Übersetzungsseite.
- Added forced file synchronization to the repository maintenance.
- Fixed automatic suggestions for languages with a longer code.
- Verbesserte Leistung beim Hinzufügen neuer Zeichenketten.
- Several bug fixes in quality checks.
- Several performance improvements.
- Integration mit *Entdecken Sie Weblate* hinzugefügt.
- Fixed checks behavior with read-only strings.

Alle Änderungen im Detail.

4.24 Weblate 4.5.1

Veröffentlicht am 5. März 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.
- Bessere Handhabung beim Hochladen von einsprachigen PO über die API.
- Das Verhalten von Warnmeldungen bei Glossarkomponenten wurde verbessert.
- Verbesserte Markdown-Link-Prüfungen.
- Indicate glossary and source language in breadcrumbs.
- Paginierte Komponentenliste großer Projekte.
- Verbesserte Leistung beim Entfernen von Übersetzungen, Komponenten oder Projekten.
- Improved bulk edit performance.
- Fixed preserving „Needs editing“ and „Approved“ states for ODF files.
- Verbesserte Benutzeroberfläche zum Anpassen der Downloads von Übersetzungsdateien

Alle Änderungen im Detail.

4.25 Weblate 4.5

Veröffentlicht am 19. Februar 2021.

- Unterstützung für das in gettext PO verwendete `lua-format` hinzugefügt.
- Unterstützung für gemeinsame Nutzung einer Komponente zwischen Projekten hinzugefügt.
- Fixed multiple unnamed variables check behavior with multiple format flags.
- Dropped mailing list field on the project in favor of generic instructions for translators.
- Erweiterung zur Erzeugung von Pseudolokalen hinzugefügt.
- Unterstützung für TermBase eXchange-Dateien hinzugefügt.
- Unterstützung für die manuelle Definition von Varianten einer Zeichenkette unter Verwendung einer Markierung wurde hinzugefügt.
- Verbesserte Leistung der Konsistenzprüfungen.
- Verbesserte Leistung des Übersetzungsspeichers für lange Zeichenketten.
- Unterstützung für die Suche in Erklärungen hinzugefügt.
- Strings can now be added and removed in bilingual formats as well.
- Extend list of supported languages in Amazon Translate machine translation.
- Automatische Aktivierung von Prüfungen der Eigenschaften des Java MessageFormat.
- Es wurde eine neue Upload-Methode hinzugefügt, um neue Zeichenketten zu einer Übersetzung hinzuzufügen.
- Es wurde eine einfache Benutzeroberfläche zum Durchsuchen von Übersetzungen hinzugefügt.
- Glossare werden jetzt als reguläre Komponenten gespeichert.
- Dropped specific API for glossaries as component API is used now.
- Added simplified interface to toggle some of the flags.
- Unterstützung für nicht übersetzbare oder unzulässige Begriffe im Glossar hinzugefügt.

- Unterstützung für die Definition von Terminologie in einem Glossar wurde hinzugefügt.
- Der Umschalter für die Leserichtung wurde verschoben, um mehr Platz für die visuelle Tastatur zu erhalten.
- Option zum automatischen Beobachten von Projekten hinzugefügt, zu denen Benutzer beigetragen haben.
- Added check whether translation matches the glossary.
- Unterstützung für die Anpassung der Textfarbe der Navigation wurde hinzugefügt.

Alle Änderungen im Detail.

4.26 Weblate 4.4.2

Veröffentlicht am 14. Januar 2021.

- Fixed corruption of one distributed MO file.

4.27 Weblate 4.4.1

Veröffentlicht am 13. Januar 2021.

- Fixed reverting plural changes.
- Die Anzeige der Hilfe für Projekteinstellungen wurde korrigiert.
- Verbesserte Benutzerverwaltung.
- Verbesserte Handhabung des Kontexts in einsprachigen PO-Dateien.
- Das Verhalten der Bereinigungsvorgang-Erweiterung bei den Formaten HTML, ODF, IDML und Windows RC wurde korrigiert.
- Das Analysieren des Speicherorts aus CSV-Dateien wurde korrigiert.
- Use content compression for file downloads.
- Improved user experience on importing from ZIP file.
- Die Erkennung des Dateiformats beim Hochladen wurde verbessert.
- Avoid duplicate pull requests on Pagure.
- Verbesserte Leistung bei der Anzeige von Ghost-Übersetzungen.
- Reimplemented translation editor to use native browser textarea.
- Es wurde behoben, dass die Erweiterung für den Bereinigungsvorgang beim Hinzufügen von neuen Zeichenketten nicht mehr funktioniert.
- API für Erweiterungen hinzugefügt.

Alle Änderungen im Detail.

4.28 Weblate 4.4

Veröffentlicht am 15. Dezember 2020.

- Verbesserte Validierung beim Erstellen einer Komponente.
- Weblate benötigt jetzt Django 3.1.
- Unterstützung für die Anpassung des Erscheinungsbilds in der Verwaltungsoberfläche wurde hinzugefügt.
- Fixed read-only state handling in bulk edit.
- Verbesserte CodeMirror-Integration.
- Erweiterung hinzugefügt, um leere Zeichenketten aus Übersetzungsdateien zu entfernen.
- Der CodeMirror-Editor wird jetzt für Übersetzungen verwendet.
- Syntaxhervorhebung im Übersetzungseditor für XML, HTML, Markdown und reStructuredText.
- Platzhalter im Übersetzungseditor hervorheben.
- Verbesserte Unterstützung für nicht standardisierte Sprachcodes.
- Warnung bei Verwendung mehrdeutiger Sprachcodes hinzugefügt.
- Dem Benutzer wird nun eine gefilterte Liste von Sprachen angezeigt, wenn er eine neue Übersetzung hinzufügt.
- Erweiterte Suchmöglichkeiten für Änderungen in der Historie.
- Improved billing detail pages and Libre hosting workflow.
- Erweiterte Übersetzungsstatistik-API.
- Verbesserte Reiterkarte „Andere Übersetzungen“ beim Übersetzen.
- Added tasks API.
- Die Leistung des Dateiuploads wurde verbessert.
- Improved display of user defined special characters.
- Verbesserte Leistung der automatischen Übersetzung.
- Several minor improvements in the user interface.
- Improved naming of ZIP downloads.
- Option zum Erhalten von Benachrichtigungen über nicht beobachtete Projekte hinzugefügt.

  Alle Änderungen im Detail.

4.29 Weblate 4.3.2

Veröffentlicht am 4. November 2020.

- Fixed crash on certain component file masks.
- Die Genauigkeit der Prüfung auf aufeinanderfolgende doppelte Wörter wurde verbessert.
- Unterstützung für Pagure-Pull-Anfragen hinzugefügt.
- Verbesserte Fehlermeldungen bei fehlgeschlagenen Registrierungen.
- 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.
- GitLab-Pull-Anfragen für Repositorys mit längeren Pfaden wurden korrigiert.

Alle Änderungen im Detail.

4.30 Weblate 4.3.1

Veröffentlicht am 21. Oktober 2020.

- Verbesserte Leistung der automatischen Übersetzung.
- Fixed session expiry for authenticated users.
- Unterstützung für das Ausblenden von Versionsinformationen hinzufügen.
- Improve hooks compatibility with Bitbucket Server.
- Die Leistung bei Aktualisierungen des Übersetzungsspeichers wurde verbessert.
- Reduced memory usage.
- Verbesserte Leistung der Matrixansicht.
- Bestätigung vor dem Entfernen eines Benutzers aus einem Projekt hinzugefügt.

  Alle Änderungen im Detail.

4.31 Weblate 4.3

Veröffentlicht am 15. Oktober 2020.

- Include user stats in the API.
- Fixed component ordering on paginated pages.
- Define source language for a glossary.
- Überarbeitete Unterstützung für GitHub- und GitLab-Pull-Anfragen.
- Behobene Statistikzählungen nach dem Entfernen eines Übersetzungsvorschlags.
- Erweitertes öffentliches Benutzerprofil.
- Fixed configuration of enforced checks.
- Verbesserung der Dokumentation über integrierte Backups.
- Moved source language attribute from project to a component.
- „Vue I18n“-Formatierungsprüfung hinzugefügt.
- Die allgemeine Platzhalterprüfung unterstützt jetzt reguläre Ausdrücke.
- Verbessertes Aussehen des Matrixmodus.
- Suchmaschinen werden jetzt als automatische Vorschläge bezeichnet.
- Unterstützung für die Interaktion mit mehreren GitLab- oder GitHub-Instanzen wurde hinzugefügt.
- 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.

- Unterstützung von Markdown in der Mitwirkungsvereinbarung.
- Verbesserte Verfolgung von Ausgangszeichenketten.
- Verbesserte Kompatibilität mit JSON-, YAML- und CSV-Formaten.
- Unterstützung für das Entfernen von Zeichenketten hinzugefügt.
- Verbesserte Leistung beim Herunterladen von Dateien.
- Verbesserte Ansicht der Repository-Verwaltung.
- Automatisches Aktivieren des Java-Formats für Android.
- Unterstützung für lokalisierte Bildschirmfotos wurde hinzugefügt.
- Unterstützung für Python 3.9 hinzugefügt.
- Die Übersetzung von HTML-Dateien unter bestimmten Bedingungen wurde behoben.

Alle Änderungen im Detail.

4.32 Weblate 4.2.2

Veröffentlicht am 2. September 2020.

- Die Übereinstimmung von Ausgangszeichenketten für JSON-Formate wurde korrigiert.
- Die Umleitung der Anmeldung für einige Authentifizierungskonfigurationen wurde korrigiert.
- LDAP-Authentifizierung mit Gruppensynchronisierung wurde korrigiert.
- Absturz bei der Meldung des automatischen Übersetzungsfortschritts behoben.
- Fixed Git commit squashing with trailers enabled.
- Die Erstellung lokaler VCS-Komponenten über die API wurde korrigiert.

4.33 Weblate 4.2.1

Veröffentlicht am 21. August 2020.

- Fixed saving plurals for some locales in Android resources.
- Absturz in der Erweiterung für den Bereinigungsvorgang für einige XLIFF-Dateien behoben.
- Allow setting up localization CDN in Docker image.

4.34 Weblate 4.2

Veröffentlicht am 18. August 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.
- Unterstützung für das Sperren von Komponenten, bei denen Repository-Fehler auftreten, wurde hinzugefügt.
- Simplified main navigation (replaced buttons with icons).

- Improved language code handling in Google Translate integration.
- Die Git-Squash-Erweiterung kann `Co-authored-by:` Trailer erzeugen.
- 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.
- Prüfung für ECMAScript-Template-Literale hinzugefügt.
- Option zum Beobachten einer Komponente hinzugefügt.
- Removed leading dot from JSON unit keys.
- Removed separate Celery queue for translation memory.
- Erlaubt das Übersetzen aller Komponenten einer Sprache auf einmal.
- Allow to configure `Content-Security-Policy` HTTP headers.
- Added support for aliasing languages at project level.
- Neue Erweiterung zur Hilfe bei der HTML- oder JavaScript-Lokalisierung, siehe *JavaScript-Lokalisierung CDN*.
- The Weblate domain is now configured in the settings, see `SITE_DOMAIN`.
- Unterstützung für die Suche nach Komponente und Projekt hinzufügen.

4.35 Weblate 4.1.1

Veröffentlicht am 19. Juni 2020.

- Änderung der Autofix- oder Erweiterungs-Konfiguration in Docker behoben.
- 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 *SAML-Authentifizierung*.

4.36 Weblate 4.1

Veröffentlicht am 15. Juni 2020.

- Added support for creating new translations with included country code.
- Unterstützung für die Suche nach Ausgangszeichenketten mit Bildschirmfoto hinzugefügt.
- 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.
- Direkten Link zum Bearbeiten des Bildschirmfotos von der Übersetzungsseite hinzugefügt.
- Improved removal confirmation dialog.
- Include templates in ZIP download.
- Add support for Markdown and notification configuration in announcements.
- Extended details in check listings.
- Added support for new file formats: *Laravel PHP-Zeichenketten*, *HTML files*, *OpenDocument Format*, *IDML Format*, *Windows RC files*, *INI translations*, *Inno Setup INI translations*, *GWT-Eigenschaften*, *go-i18n JSON files*, *ARB File*.
- Consistently use dismissed as state of dismissed checks.
- Unterstützung für die Aktivierung der Konfiguration von Standarderweiterungen wurde hinzugefügt.
- 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: *Mehrere unbenannte Variablen*, *Lange nicht übersetzt*, *Aufeinanderfolgende doppelte Wörter*.
- 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.

4.37 Weblate 4.0.4

Veröffentlicht am 7. Mai 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.
- Die Auflistung der Bildschirmfotos wurde korrigiert.
- Fixed monthly digest notifications.
- Fixed intermediate translation behavior with units non existing in translation.

4.38 Weblate 4.0.3

Veröffentlicht am 2. Mai 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.

4.39 Weblate 4.0.2

Veröffentlicht am 27. April 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.

4.40 Weblate 4.0.1

Veröffentlicht am 16. April 2020.

- Fixed package installation from PyPI.

4.41 Weblate 4.0

Veröffentlicht am 16. April 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.
- Vorschläge für fehlgeschlagene Prüfungen anzeigen.
- Record source string changes in history.
- Upgraded Microsoft Translator to version 3 API.
- Reimplemented translation memory backend.
- Added support for several `is:` lookups in *Searching*.
- Allow to make *Unveränderte Übersetzung* avoid internal blacklist.
- Improved comments extraction from monolingual po files.
- Renamed whiteboard messages to announcements.
- Fixed occasional problems with registration mails.
- Die LINGUAS-Update-Erweiterung wurde verbessert, um mehr Syntaxvarianten zu behandeln.
- Fixed editing monolingual XLIFF source file.
- Added support for exact matching in *Searching*.
- Erweiterte API zur Abdeckung von Bildschirmfotos, Benutzern, Gruppen, Komponentenlisten und erweiterter Projekterstellung.
- 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.

4.42 Weblate 3.x series

4.42.1 Weblate 3.11.3

Veröffentlicht am 11. März 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.
- Unterstützung für Slack-Authentifizierung in Docker-Containern hinzugefügt.
- Avoid sending notifications for not subscribed languages.
- Include Celery queues in performance overview.
- Die Links zu den Dokumentationen für die Erweiterungen wurden korrigiert.
- 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.
- Die Authentifizierung bei einigen Drittanbietern wurde korrigiert.

4.42.2 Weblate 3.11.2

Veröffentlicht am 22. Februar 2020.

- Fixed rendering of suggestions.
- Fixed some strings wrongly reported as having no words.

4.42.3 Weblate 3.11.1

Veröffentlicht am 20. Februar 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.

4.42.4 Weblate 3.11

Veröffentlicht am 17. Februar 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.
- Anzeige von benachbarten Zeichenketten bei einsprachigen Übersetzungen.
- Unterstützung für die Gruppierung von Zeichenketten wurde hinzugefügt.
- Recommend upgrade to new Weblate versions in the system checks.
- Provide more detailed analysis for duplicate language alert.
- Include more detailed license info on the project pages.
- Automatically unshallow local copies if needed.
- Fixed download of strings needing action.
- New alert to warn about using the same file mask twice.
- Improve XML placeables extraction.
- The `SINGLE_PROJECT` can now enforce redirection to chosen project.
- Added option to resolve comments.
- Added bulk editing of flags.
- Added support for labels.
- Erweiterung für Massenbearbeitung hinzugefügt.
- Added option for *Enforcing checks*.
- Increased default validity of confirmation links.
- Improved Matomo integration.
- Fixed *Ist übersetzt worden* to correctly handle source string change.
- Extended automatic updates configuration by `AUTO_UPDATE`.
- LINGUAS-Erweiterungen synchronisieren jetzt vollständig die Übersetzungen in Weblate.

4.42.5 Weblate 3.10.3

Veröffentlicht am 18. Januar 2020.

- Support for translate-toolkit 2.5.0.

4.42.6 Weblate 3.10.2

Veröffentlicht am 18. Januar 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.
- Neue Erweiterung zur Konfiguration der YAML-Formatierung.
- Fixed same plurals check to not fire on single plural form languages.
- Fixed regex search on some fields.

4.42.7 Weblate 3.10.1

Veröffentlicht am 9. Januar 2020.

- Extended API with translation creation.
- Fixed several corner cases in data migrations.
- Compatibility with Django 3.0.
- Die Leistung der Datenbereinigung wurde verbessert.
- 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.

4.42.8 Weblate 3.10

Veröffentlicht am 20. Dezember 2019.

- Improved application user interface.
- Added doublespace 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.

4.42.9 Weblate 3.9.1

Veröffentlicht am 28. Oktober 2019.

- Einige nicht benötigte Dateien aus Backups entfernt.
- Fixed potential crash in reports.
- Fixed cross database migration failure.
- Added support for force pushing Git repositories.
- Reduced risk of registration token invalidation.
- Fixed account removal hitting rate limiter.
- Added search based on priority.
- Fixed possible crash on adding strings to JSON file.
- Safe HTML check and fixup now honor source string markup.
- Avoid sending notifications to invited and deleted users.
- Fix SSL connection to redis in Celery in Docker container.

4.42.10 Weblate 3.9

Veröffentlicht am 15. Oktober 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.
- Erweiterung für automatische Übersetzung zu Bootstrap-Übersetzungen hinzugefügt.
- Extend automatic translation to add suggestions.
- Erweiterungsparameter werden in der Übersicht angezeigt.
- Sentry is now supported through modern Sentry SDK instead of Raven.
- Changed example settings to be better fit for production environment.
- Automatische Backups mit BorgBackup hinzugefügt.
- Bereinigungsvorgang für die RESX-Erweiterung zur Vermeidung unerwünschter Dateiaktualisierungen aufgeteilt.
- 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.

4.42.11 Weblate 3.8

Veröffentlicht am 15. August 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.
- Verbesserte Leistung von Erweiterungen mit Repository-Spezifikation.
- 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.
- Unterstützung des Klonens von Erweiterungen für erkannte Komponenten wurde hinzugefügt.
- 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.
- Unterstützung für animierte Bildschirmfotos hinzugefügt.
- 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.

4.42.12 Weblate 3.7.1

Veröffentlicht am 28. Juni 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.

4.42.13 Weblate 3.7

Veröffentlicht am 21. Juni 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.
- Unterstützung für die Konfiguration der msgmerge-Erweiterung wurde hinzugefügt.
- Delay opening SMTP connection when sending notifications.
- Improved error logging.
- Allow custom location in MO generating add-on.
- Erweiterungen um alte Vorschläge oder Kommentare zu bereinigen hinzugefügt.
- 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.

4.42.14 Weblate 3.6.1

Veröffentlicht am 26. April 2019.

- Improved handling of monolingual XLIFF files.
- Fixed digest notifications in some corner cases.
- Fehlermeldung des Erweiterungsskripts behoben.
- 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.

4.42.15 Weblate 3.6

Veröffentlicht am 20. April 2019.

- Add support for downloading user data.
- Erweiterungen werden jetzt automatisch bei der Installation ausgelöst.
- Improved instructions for resolving merge conflicts.
- Die Bereinigungserweiterung ist jetzt mit der Übersetzung von App-Store-Metadaten kompatibel.
- 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.
- Administratoren und Konsistenz-Erweiterungen können jetzt Übersetzungen hinzufügen, auch wenn sie für Benutzer deaktiviert sind.
- 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.
- Zusammenfassungs-Benachrichtigungen hinzugefügt.
- 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.

4.42.16 Weblate 3.5.1

Veröffentlicht am 10. März 2019.

- Fixed Celery systemd unit example.
- Fixed notifications from HTTP repositories with login.
- Fixed race condition in editing source string for monolingual translations.
- Die Ausgabe der fehlgeschlagenen Erweiterungsausführung wird in die Protokolle aufgenommen.
- 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.

4.42.17 Weblate 3.5

Veröffentlicht am 3. März 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.
- Commit-Nachricht für eine Erweiterung kann jetzt konfiguriert werden.
- 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.
- Abrechnungserweiterung entfernt jetzt Projekte für unbezahlte Abrechnungen nach 45 Tagen.

4.42.18 Weblate 3.4

Veröffentlicht am 22. Januar 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.
- Neue Erweiterung zu Squash-Commits vor dem 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.

4.42.19 Weblate 3.3

Veröffentlicht am 30. November 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.

4.42.20 Weblate 3.2.2

Veröffentlicht am 20. Oktober 2018.

- Remove no longer needed Babel dependency.
- Updated language definitions.
- Verbesserung der Dokumentation für Erweiterungen, LDAP und Celery.
- Fixed enabling new dos-eol and auto-java-messageformat flags.
- Fixed running setup.py test from PyPI package.
- Improved plurals handling.
- Fixed translation upload API failure in some corner cases.
- Fixed updating Git configuration in case it was changed manually.

4.42.21 Weblate 3.2.1

Veröffentlicht am 10. Oktober 2018.

- Document dependency on backports.csv on Python 2.7.
- Fix running tests under root.
- Improved error handling in gitexport module.
- Fixed progress reporting for newly added languages.
- Correctly report Celery worker errors to Sentry.
- Fixed creating new translations with Qt Linguist.
- Fixed occasional fulltext index update failures.
- Improved validation when creating new components.
- Added support for cleanup of old suggestions.

4.42.22 Weblate 3.2

Veröffentlicht am 6. Oktober 2018.

- Hinzufügen des Verwaltungsbefehls install_addon für die automatische Installation von Erweiterungen.
- 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.
- Der integrierte Übersetzungsspeicher speichert jetzt automatisch durchgeführte Übersetzungen.
- Users and projects can import their existing translation memories.
- Bessere Verwaltung von zusammenhängenden Zeichenketten für Bildschirmfotos.
- Added support for checking Java MessageFormat.

See 3.2 milestone on GitHub for detailed list of addressed issues.

4.42.23 Weblate 3.1.1

Veröffentlicht am 27. Juli 2018.

- Fix testsuite failure on some setups.

4.42.24 Weblate 3.1

Veröffentlicht am 27. Juli 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.

- Neue Erweiterung, um unveränderte Übersetzungen als bearbeitungsbedürftig zu markieren.
- 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.

4.42.25 Weblate 3.0.1

Veröffentlicht am 10. Juni 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.

4.42.26 Weblate 3.0

Veröffentlicht am 1. Juni 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.
- Neue Erweiterung, um sicherzustellen, dass alle Komponenten in einem Projekt die gleichen Übersetzungen haben.
- Support for more variables in commit message templates.
- Add support for providing additional textual context.

4.43 Weblate 2.x series

4.43.1 Weblate 2.20

Veröffentlicht am 4. April 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.
- Unterstützung für die Authentifizierung in API mit RFC 6750 kompatibler Bearer-Authentifizierung hinzugefügt.
- 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.
- Integrierter Übersetzungsspeicher hinzugefügt.
- Added componentlists overview to dashboard and per component list overview pages.
- Added support for DeepL machine translation service.
- Machine translation results are now cached inside Weblate.
- Unterstützung für die Neuordnung von bestätigten Änderungen wurde hinzugefügt.

4.43.2 Weblate 2.19.1

Veröffentlicht am 20. Februar 2018.

- Fixed migration issue on upgrade from 2.18.
- Improved file upload API validation.

4.43.3 Weblate 2.19

Veröffentlicht am 15. Februar 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.
- Verbesserte integrierte automatische Übersetzung.
- 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.

- Erweiterungen zum Anpassen von Übersetzungs-Workflows hinzugefügt.

4.43.4 Weblate 2.18

Veröffentlicht am 15. Dezember 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.

4.43.5 Weblate 2.17.1

Veröffentlicht am 13. Oktober 2017.

- Fixed running testsuite in some specific situations.
- Locales updates.

4.43.6 Weblate 2.17

Veröffentlicht am 13. Oktober 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.

4.43.7 Weblate 2.16

Veröffentlicht am 11. August 2017.

- Various performance improvements.
- Added support for nested JSON format.
- Added support for WebExtension JSON format.
- Die Authentifizierung des Git-Exporters wurde korrigiert.
- 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.
- Die Komponenten-, Projekt- und plattformweite Suche nach Übersetzungen wurde korrigiert.

4.43.8 Weblate 2.15

Veröffentlicht am 30. Juni 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.
- Verbesserte Fehlermeldungen bei fehlgeschlagener Authentifizierung.
- 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.

4.43.9 Weblate 2.14.1

Veröffentlicht am 24. Mai 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.

4.43.10 Weblate 2.14

Veröffentlicht am 17. Mai 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.
- Die Authentifizierungsversuche sind nun ratenbegrenzt.
- 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.
- Aufnahme von Informationen zur Ausgangszeichenkette und Bildschirmfotos in die API.
- Allow to overwrite translations through API upload.

4.43.11 Weblate 2.13.1

Veröffentlicht am 12. April 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.
- HTTP 404 zurückgeben, wenn versucht wird, auf ein Projekt zuzugreifen, für das der Benutzer keine Berechtigungen hat.

4.43.12 Weblate 2.13

Veröffentlicht am 12. April 2017.

- Fixed quality checks on translation templates.
- Added quality check to trigger on losing translation.
- Add option to view pending suggestions from user.
- Add option to automatically build component lists.
- Default dashboard for unauthenticated users can be configured.
- Add option to browse 25 random strings for review.
- History now indicates string change.

- Better error reporting when adding new translation.
- Added per language search within project.
- Group ACLs can now be limited to certain permissions.
- The per project ACLs are now implemented using Group ACL.
- Eine feinere Kontrolle der Berechtigungen wurde hinzugefügt.
- Various minor UI improvements.

4.43.13 Weblate 2.12

Veröffentlicht am 3. März 2017.

- Improved admin interface for groups.
- Added support for Yandex Translate API.
- Verbesserte Geschwindigkeit der plattformweiten Suche.
- 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.
- Verbesserte Bildschirmfoto-Verwaltung mit OCR-Unterstützung für übereinstimmende Ausgangszeichenketten.
- Default commit message now includes translation information and URL.
- Added support for Joomla translation format.
- Improved reliability of import across file formats.

4.43.14 Weblate 2.11

Veröffentlicht am 31. Januar 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.

4.43.15 Weblate 2.10.1

Veröffentlicht am 20. Januar 2017.

- Do not leak account existence on password reset form (CVE-2017-5537).

4.43.16 Weblate 2.10

Veröffentlicht am 15. Dezember 2016.

- Added quality check to check whether plurals are translated differently.
- GitHub-Hooks für Repositorys mit Authentifizierung korrigiert.
- 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.
- Matrixansicht für schnellen Überblick über alle Übersetzungen hinzugefügt.
- Added basic API for changes and strings.
- Added support for Apertium APy server for machine translations.

4.43.17 Weblate 2.9

Veröffentlicht am 4. November 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.
- Unterstützung für das Hochladen von visuellem Kontext (Bildschirmfotos) für Übersetzungen wurde hinzugefügt.

4.43.18 Weblate 2.8

Veröffentlicht am 31. August 2016.

- Verbesserung der Dokumentation.
- 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.

4.43.19 Weblate 2.7

Veröffentlicht am 10. Juli 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.

4.43.20 Weblate 2.6

Veröffentlicht am 28. April 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.

4.43.21 Weblate 2.5

Veröffentlicht am 10. März 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.
- Verbesserung der Dokumentation.
- Added support for generating translator credits.
- Added support for generating contributor stats.
- Die plattformweite Suche kann nur in einer Sprache suchen.
- 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.
- Die plattformweite Suche nach fehlgeschlagenen Qualitätsprüfungen wurde korrigiert.
- Added option to specify source language.
- Improved support for XLIFF files.
- Extended list of options for import_project.
- Improved targeting for whiteboard messages.
- Support for automatic translation across projects.
- Optimized fulltext search index.
- Added management command for auto translation.
- Added placeables highlighting.
- Added keyboard shortcuts for placeables, checks and machine translations.
- Improved translation locking.
- Added quality check for AngularJS interpolation.
- Added extensive group based ACLs.
- Clarified terminology on strings needing edit (formerly fuzzy).
- Clarified terminology on strings needing action and untranslated strings.
- Support for Python 3.
- Dropped support for Django 1.7.
- Dropped dependency on msginit for creating new gettext PO files.
- Added configurable dashboard views.
- Improved notifications on parse errors.
- Added option to import components with duplicate name to import_project.

- Improved support for translating PHP files.
- Added XLIFF export for dictionary.
- Added XLIFF and gettext PO export for all translations.
- Verbesserung der Dokumentation.
- Added support for configurable automatic group assignments.
- Improved adding of new translations.

4.43.22 Weblate 2.4

Veröffentlicht am 20. September 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.

4.43.23 Weblate 2.3

Veröffentlicht am 22. Mai 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.

4.43.24 Weblate 2.2

Veröffentlicht am 19. Februar 2015.

- Leistungsverbesserungen.
- Fulltext search on location and comments fields.
- New SVG/JavaScript-based activity charts.
- Support for Django 1.8.
- Support for deleting comments.
- Added own SVG badge.
- Added support for Google Analytics.
- Improved handling of translation filenames.
- Added support for monolingual JSON translations.
- Record component locking in a history.
- Support for editing source (template) language for monolingual translations.
- Added basic support for Gerrit.

4.43.25 Weblate 2.1

Veröffentlicht am 5. Dezember 2014.

- Added support for Mercurial repositories.
- Replaced Glyphicon font by Awesome.
- Icons für Authentifizierungsdienste der sozialen Medien hinzugefügt.
- Better consistency of button colors and icons.
- Verbesserung der Dokumentation.
- Various bugfixes.
- Automatic hiding of columns in translation listing for small screens.
- Changed configuration of filesystem paths.
- Improved SSH keys handling and storage.
- Improved repository locking.
- Customizable quality checks per source string.
- Allow to hide completed translations from dashboard.

4.43.26 Weblate 2.0

Veröffentlicht am 6. November 2014.

- New responsive UI using Bootstrap.
- Rewritten VCS backend.
- Verbesserung der Dokumentation.
- Whiteboard für plattformweite Meldungen hinzugefügt.
- Configurable strings priority.
- Added support for JSON file format.
- Fixed generating mo files in certain cases.
- Added support for GitLab notifications.
- Added support for disabling translation suggestions.
- Django 1.7 support.
- ACL projects now have user management.
- Extended search possibilities.
- Give more hints to translators about plurals.
- Fixed Git repository locking.
- Compatibility with older Git versions.
- Improved ACL support.
- Added buttons for per language quotes and other special characters.
- Support for exporting stats as JSONP.

4.44 Weblate 1.x series

4.44.1 Weblate 1.9

Veröffentlicht am 6. Mai 2014.

- Django 1.6 compatibility.
- No longer maintained compatibility with Django 1.4.
- Management commands for locking/unlocking translations.
- Improved support for Qt TS files.
- Users can now delete their account.
- Avatars can be disabled.
- Merged first and last name attributes.
- Avatars are now fetched and cached server side.
- Added support for shields.io badge.

4.44.2 Weblate 1.8

Veröffentlicht am 7. November 2013.

- Please check manual for upgrade instructions.
- Nicer listing of project summary.
- Better visible options for sharing.
- Mehr Kontrolle über anonyme Benutzerberechtigungen.
- Supports login using third party services, check manual for more details.
- Users can login by e-mail instead of username.
- Verbesserung der Dokumentation.
- Improved source strings review.
- Searching across all strings.
- Better tracking of source strings.
- Captcha protection for registration.

4.44.3 Weblate 1.7

Veröffentlicht am 7. Oktober 2013.

- Please check manual for upgrade instructions.
- Support for checking Python brace format string.
- Per component customization of quality checks.
- Detailed per translation stats.
- Changed way of linking suggestions, checks and comments to strings.
- Users can now add text to commit message.
- Support for subscribing on new language requests.
- Support for adding new translations.

- Widgets and charts are now rendered using Pillow instead of Pango + Cairo.
- Add status badge widget.
- Dropped invalid text direction check.
- Changes in dictionary are now logged in history.
- Performance improvements for translation view.

4.44.4 Weblate 1.6

Veröffentlicht am 25. Juli 2013.

- Nicer error handling on registration.
- Browsing of changes.
- Fixed sorting of machine translation suggestions.
- Improved support for MyMemory machine translation.
- Added support for Amagama machine translation.
- Various optimizations on frequently used pages.
- Highlights searched phrase in search results.
- Support for automatic fixups while saving the message.
- Tracking of translation history and option to revert it.
- Added support for Google Translate API.
- Added support for managing SSH host keys.
- Various form validation improvements.
- Various quality checks improvements.
- Performance improvements for import.
- Added support for voting on suggestions.
- Cleanup of admin interface.

4.44.5 Weblate 1.5

Veröffentlicht am 16. April 2013.

- Please check manual for upgrade instructions.
- Added public user pages.
- Better naming of plural forms.
- Added support for TBX export of glossary.
- Added support for Bitbucket notifications.
- Activity charts are now available for each translation, language or user.
- Extended options of import_project admin command.
- Compatible with Django 1.5.
- Avatars are now shown using 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.
- New interface for machine translations.
- Added support for monolingual po files.
- Extend amount of cached metadata to improve speed of various searches.
- Now shows word counts as well.

4.44.6 Weblate 1.4

Veröffentlicht am 23. Januar 2013.

- Fixed deleting of checks/comments on string deletion.
- Added option to disable automatic propagation of translations.
- Added option to subscribe for merge failures.
- Correctly import on projects which needs custom ttkit loader.
- Added sitemaps to allow easier access by crawlers.
- Provide direct links to string in notification e-mails or feeds.
- Various improvements to admin interface.
- Provide hints for production setup in admin interface.
- Added per language widgets and engage page.
- Improved translation locking handling.
- Show code snippets for widgets in more variants.
- Indicate failing checks or fuzzy strings in progress bars.
- More options for formatting commit message.
- Fixed error handling with machine translation services.
- Improved automatic translation locking behaviour.
- Support for showing changes from previous source string.
- Added support for substring search.
- Various quality checks improvements.
- Support for per project ACL.
- Basic code coverage by unit tests.

4.44.7 Weblate 1.3

Veröffentlicht am 16. November 2012.

- Compatibility with PostgreSQL database backend.
- Removes languages removed in upstream git repository.
- Improved quality checks processing.
- Neue Prüfungen hinzugefügt (BB-Code, XML-Markup und Zeilenumbrüche).
- Support for optional rebasing instead of merge.

- Possibility to relocate Weblate (for example to run it under /weblate path).
- Support for manually choosing file type in case autodetection fails.
- Better support for Android resources.
- Support for generating SSH key from web interface.
- More visible data exports.
- New buttons to enter some special characters.
- Support for exporting dictionary.
- Support for locking down whole Weblate installation.
- Checks for source strings and support for source strings review.
- Support for user comments for both translations and source strings.
- Better changes log tracking.
- Changes can now be monitored using RSS.
- Improved support for RTL languages.

4.44.8 Weblate 1.2

Veröffentlicht am 14. August 2012.

- Weblate now uses South for database migration, please check upgrade instructions if you are upgrading.
- Fixed minor issues with linked git repos.
- New introduction page for engaging people with translating using Weblate.
- Added widgets which can be used for promoting translation projects.
- Added option to reset repository to origin (for privileged users).
- Project or component can now be locked for translations.
- Possibility to disable some translations.
- Configurable options for adding new translations.
- Configuration of git commits per project.
- Simple antispam protection.
- Better layout of main page.
- Support for automatically pushing changes on every commit.
- Support for e-mail notifications of translators.
- List only used languages in preferences.
- Improved handling of not known languages when importing project.
- Support for locking translation by translator.
- Optionally maintain `Language-Team` header in po file.
- Include some statistics in about page.
- Supports (and requires) django-registration 0.8.
- Zwischenspeichern der Anzahl von Zeichenketten mit fehlgeschlagenen Qualitätsprüfungen.
- Checking of requirements during setup.
- Verbesserung der Dokumentation.

4.44.9 Weblate 1.1

Veröffentlicht am 4. Juli 2012.

- Improved several translations.
- Better validation while creating component.
- Added support for shared git repositories across components.
- Do not necessary commit on every attempt to pull remote repo.
- Added support for offloading indexing.

4.44.10 Weblate 1.0

Veröffentlicht am 10. Mai 2012.

- Improved validation while adding/saving component.
- Experimental support for Android component files (needs patched ttkit).
- Updates from hooks are run in background.
- Improved installation instructions.
- Improved navigation in dictionary.

4.45 Weblate 0.x series

4.45.1 Weblate 0.9

Veröffentlicht am 18. April 2012.

- Fixed import of unknown languages.
- Improved listing of nearby messages.
- Improved several checks.
- Documentation updates.
- Added definition for several more languages.
- Various code cleanups.
- Verbesserung der Dokumentation.
- Changed file layout.
- Update helper scripts to Django 1.4.
- Improved navigation while translating.
- Better handling of po file renames.
- Better validation while creating component.
- Integrated full setup into syncdb.
- Added list of recent changes to all translation pages.
- Check for untranslated strings ignores format string only messages.

4.45.2 Weblate 0.8

Veröffentlicht am 3. April 2012.

- Replaced own full text search with Whoosh.
- Various fixes and improvements to checks.
- New command updatechecks.
- Lot of translation updates.
- Added dictionary for storing most frequently used terms.
- Added /admin/report/ for overview of repositories status.
- Machine translation services no longer block page loading.
- Management interface now contains also useful actions to update data.
- Records log of changes made by users.
- Ability to postpone commit to Git to generate less commits from single user.
- Possibility to browse failing checks.
- Automatic translation using already translated strings.
- New about page showing used versions.
- Django 1.4 compatibility.
- Ability to push changes to remote repo from web interface.
- Added review of translations done by others.

4.45.3 Weblate 0.7

Veröffentlicht am 16. Februar 2012.

- Direct support for GitHub notifications.
- Added support for cleaning up orphaned checks and translations.
- Displays nearby strings while translating.
- Displays similar strings while translating.
- Improved searching for string.

4.45.4 Weblate 0.6

Veröffentlicht am 14. Februar 2012.

- Added various checks for translated messages.
- Tunable access control.
- Improved handling of translations with new lines.
- Added client side sorting of tables.
- Please check upgrading instructions in case you are upgrading.

4.45.5 Weblate 0.5

Veröffentlicht am 12. Februar 2012.

- **Support for machine translation using following online services:**
  - Apertium
  - Microsoft Translator
  - MyMemory
- Several new translations.
- Improved merging of upstream changes.
- Better handle concurrent git pull and translation.
- Propagating works for fuzzy changes as well.
- Propagating works also for file upload.
- Fixed file downloads while using FastCGI (and possibly others).

4.45.6 Weblate 0.4

Veröffentlicht am 8. Februar 2012.

- Added usage guide to documentation.
- Fixed API hooks not to require CSRF protection.

4.45.7 Weblate 0.3

Veröffentlicht am 8. Februar 2012.

- Better display of source for plural translations.
- New documentation in Sphinx format.
- Displays secondary languages while translating.
- Improved error page to give list of existing projects.
- New per language stats.

4.45.8 Weblate 0.2

Veröffentlicht am 7. Februar 2012.

- Improved validation of several forms.
- Warn users on profile upgrade.
- Remember URL for login.
- Naming of text areas while entering plural forms.
- Automatic expanding of translation area.

4.45.9 Weblate 0.1

Veröffentlicht am 6. Februar 2012.

- Initial release.

Python-Modulindex

W

HTTP Routing Table

/

/api

/api/addons

/api/changes

/api/component-lists

/api/components

/api/groups

/api/languages

/api/memory

/api/metrics

/api/projects

/api/roles

/api/screenshots

/api/tasks

/api/translations

/api/units

/api/users

/exports

/hooks

Stichwortverzeichnis

Sonderzeichen

A

B

C

D

E

F

G

H

I

Q

R

S

T

U

V

W

X

Y