Weblate

Component configuration:

```
../devel/integration
```

2.5.
Preferences

- Hide completed translations on the dashboard
- Translation editor mode
  - Full editor
- Zen editor mode
  - Top to bottom
- Number of nearby strings
  - 15
  - Number of nearby strings to show in each direction in the full editor
- Show secondary translations in the Zen mode
- Hide source if a secondary translation exists

Editor link

Enter a custom URL to be used as link to the source code. You can use {{branch}}, {{filename}} and {{line}} as filename and line placeholders.

Special characters

You can specify additional special visual keyboard characters to be shown while translating. It can be useful for characters you use frequently, but are hard to type on your keyboard.

Default dashboard view

- Watched translations
- Suggested translations

Save
<table>
<thead>
<tr>
<th>Translation</th>
<th>Translated</th>
<th>Unfinished</th>
<th>Unfinished words</th>
<th>Checks</th>
<th>Suggestions</th>
</tr>
</thead>
<tbody>
<tr>
<td>WeblateOrg/Android — Czech</td>
<td>76%</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WeblateOrg/Django — Hungarian</td>
<td>69%</td>
<td>8</td>
<td>109</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>WeblateOrg/Django — Czech</td>
<td>96%</td>
<td>1</td>
<td>12</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>WeblateOrg/Django — Hebrew</td>
<td>92%</td>
<td>2</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WeblateOrg/Django — Hebrew</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WeblateOrg/Django — Hungarian</td>
<td>96%</td>
<td>2</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WeblateOrg/Django — Czech</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WeblateOrg/Language names —</td>
<td>81%</td>
<td>4</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hungarian</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WeblateOrg/Language names —</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hebrew</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WeblateOrg/Language names —</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Czech</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WeblateOrg/WeblateOrg — Czech</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WeblateOrg/WeblateOrg —</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hungarian</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WeblateOrg/WeblateOrg —</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hebrew</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Powered by Weblate 4.11  About Weblate  Legal  Contact  Documentation  Donate to Weblate
Component configuration

Web:

VCS

editor://open/?file={{filename}}&line={{line}}

URL

Nette documentation

Special characters

Additional special characters to include in the `Component configuration`.

FAULT_AUTO_WATCH

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

Available:
Choose:

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

Save

Component watched notification
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
Rental error
Do not notify
Comment on source translation
Instant notification
Suggested comment
Instant notification
New language
Do not notify
New translation component
Do not notify
New assertion
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
Translating string
Do not notify
Approving string
Do not notify
Pending suggestions
Do not notify
Unreviewed strings
Do not notify

Save
| Weblate | Weblate | GitLab | GitHub | Google | Facebook | Bitbucket | OAuth 2.0 |
Username: testuser
Full name: Webiote Test
E-mail: webiote@example.org

You can add another e-mail address below.
Your name and e-mail will appear as commit authorship.

Current user identities

<table>
<thead>
<tr>
<th>Identity</th>
<th>User ID</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password</td>
<td>testuser</td>
<td>[Change password]</td>
</tr>
<tr>
<td>E-mail</td>
<td><a href="mailto:webiote@example.org">webiote@example.org</a></td>
<td>[Disconnect]</td>
</tr>
<tr>
<td>Google</td>
<td><a href="mailto:webiote@example.org">webiote@example.org</a></td>
<td>[Disconnect]</td>
</tr>
<tr>
<td>GitHub</td>
<td>123456</td>
<td>[Disconnect]</td>
</tr>
<tr>
<td>Bitbucket</td>
<td>webiote</td>
<td>[Disconnect]</td>
</tr>
</tbody>
</table>

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]
ENABLE AVATARS https://gravatar.com/

API

API

API

IP

Weblate

Weblate

Component

Component

Component

Components

Languages

Info

Search

Insights

Files

Tools

Manage

Share

Component

Component

Component

Add new translation component
Weblate

3: Markdown

@mention

report-source

variants

Weblate Unicode Language Plural Rules:

(labels)

✓ (✏) ()

Unicode Language Plural Rules:
<table>
<thead>
<tr>
<th>English</th>
<th>Czech, One</th>
<th>Czech, Few</th>
<th>Czech, Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Singular</td>
<td>%count as word</td>
<td>%count as slovo</td>
<td>%count as slovo</td>
</tr>
<tr>
<td>Plural</td>
<td>%count as words</td>
<td>%count slovo</td>
<td>%count slovo</td>
</tr>
</tbody>
</table>

Plural formula: \( (n == 1) ? 0 : (n == 2 &amp; &amp; n < 1000) ? 2 : 1 \)

Needs editing?

Glossary

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

Add term to glossary

String information

<table>
<thead>
<tr>
<th>Screenshot content</th>
</tr>
</thead>
<tbody>
<tr>
<td>No screenshot currently associated.</td>
</tr>
</tbody>
</table>

Add screenshot

<table>
<thead>
<tr>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>No explanation currently provided.</td>
</tr>
</tbody>
</table>

Labels

No labels currently set.

<table>
<thead>
<tr>
<th>Flags</th>
</tr>
</thead>
<tbody>
<tr>
<td>python-format</td>
</tr>
</tbody>
</table>

Source string location

webkit/templates/translation.html:149

<table>
<thead>
<tr>
<th>String age</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 seconds ago</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source string age</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 seconds ago</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Translation file</th>
</tr>
</thead>
<tbody>
<tr>
<td>webkit/locale/cs/LC_MESSAGES/5/django.po, string S</td>
</tr>
</tbody>
</table>

Nearby strings

Comments

Automatic suggestions

Other languages

History

New comment

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

Scope

Translation comment, discussions with other translators

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

New comment

You can use Markdown and mention users by @username.

Save
### 2.18 User configured Special characters defined in the Weblate RTL SPECIAL_CHARS

- Alt+Home
- Alt+End
- Alt+PageUp or Ctrl+↑ or Alt+↑ or Cmd+↑
- Alt+PageDown or Ctrl+↓ or Alt+↓ or Cmd+↓
- Alt+Enter or Ctrl+Enter or Cmd+Enter
- Ctrl+Shift+Enter or Cmd+Shift+Enter
- Ctrl+E or Ctrl+Enter or Cmd+Enter
- Ctrl+U or Cmd+U
- Ctrl+M or Cmd+M
- Ctrl+1 to Ctrl+9 or Cmd+1 to Cmd+9
- Ctrl+M+1 to 9
- Ctrl+I+1 to 9 or Ctrl+1 to Ctrl+9 or Cmd+I+1 to 9 or Ctrl+1 to Ctrl+9
- Ctrl+J or Ctrl+J or Ctrl+J or Ctrl+J
- Ctrl+S or Ctrl+S or Ctrl+S or Ctrl+S
- Ctrl+O or Ctrl+O or Ctrl+O or Ctrl+O
- Ctrl+Y or Ctrl+Y or Ctrl+Y or Ctrl+Y

### 3 User configured:

User configured **Special characters** defined in the **SPECIAL_CHARS**

Weblate **SPECIAL_CHARS**
<table>
<thead>
<tr>
<th>Language</th>
<th>Target string</th>
</tr>
</thead>
<tbody>
<tr>
<td>Czech</td>
<td>Soubory</td>
</tr>
<tr>
<td>Hungarian</td>
<td>Fájlok</td>
</tr>
<tr>
<td>English</td>
<td>Files</td>
</tr>
</tbody>
</table>
Weblate
Visual context for strings:

LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH

1

16
Keeping translations same across components
Bulk edit add-on
GET /api/translations/(string:project)/(string:component)/(string:language)/file/
Upload

The uploaded file will be merged with the current translation. If you want to overwrite already translated strings, don't forget to turn it on.

File

Choose File: No file chosen

File upload mode

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

Processing of strings needing edit

- Do not import

Conflict handling

- Update translated strings

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

Author name

Weblate Test

Author e-mail

weblate@example.org

Upload
POST /api/translations/{project}/{component}/{language}/file/
4.5.
↓
read-only

4.5.
↓
forbidden

4.5. Termination

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

Translation

English
Singular
%{count}s word

Plural
%{count}s words

Czech, One

Czech, Few
několik slov

Czech, Other
%{count} slov

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

Needs editing

Save and continue  Save and stay  Suggest  Skip

Things to check

⚠️ Python format
Following format strings are missing: %{count}s

Dismiss
For all languages

⚠️ Missing plurals
Some plural forms are untranslated

Dismiss
For all languages

Glossary

English  Czech
No related strings found in the glossary.

Add term to glossary

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
webate/templates/translation.html149

String age
9 seconds ago

Source string age
9 seconds ago

Translation file
webate/locale/cs/LCMESSAGESS/django.po, string 5

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

25
BBCode

weblate.checks.markup.BBCodeCheck
ignore-bbcodes

DuplicateCheck

weblate.checks.duplicate.DuplicateCheck
ignore-duplicate

GlossaryCheck

weblate.checks.glossary.GlossaryCheck
check-glossary
ignore-check-glossary
weblate.checks.chars.DoubleSpaceCheck
ignore-double-space
<table>
<thead>
<tr>
<th>Language</th>
<th>Singular</th>
<th>Plural</th>
</tr>
</thead>
<tbody>
<tr>
<td>English</td>
<td>%count%s word</td>
<td>%count%s words</td>
</tr>
<tr>
<td>Czech</td>
<td>%count%s slovo</td>
<td>%count%s slova</td>
</tr>
</tbody>
</table>

Plural formula: (n==1) ? 0 : (n==2 && n!=0) ? 1 : 2

**Needs editing**

**Glossary**

- No related strings found in the glossary.

**String information**

- No explanation currently provided.
- No labels currently set.
- Flags: python-format

**Source string location**

weblate/templates/translation.html:149

**String age**

5 seconds ago

**Source string age**

6 seconds ago

**Translation file**

weblate/locale/cs/LC_MESSAGES/5/django-po, string 5
AngularJS

weblate.checks.angularjs.AngularJSInterpolationCheck
angularjs-format
ignore-angularjs-format
Your balance is {{amount}} {{ currency }}

C

weblate.checks.format.CFormatCheck
c-format
ignore-c-format
There are %d apples
Your balance is %1$d %2$s

C#

weblate.checks.format.CSharpFormatCheck
c-sharp-format
ignore-c-sharp-format
There are {0} apples

ECMAScript

weblate.checks.format.ESTemplateLiteralsCheck
es-format
ignore-es-format
There are ${number} apples

C printf format

C# printf format

ECMAScript Template literals

AngularJS <https://angular.jp/guide/interpolation>`_
**i18next**

[i18next](https://www.i18next.com/translation-function/interpolation)

**ICU MessageFormat**

[International Components for Unicode MessageFormat](https://unicode.org/reports/tr35)

**Java**
Java MessageFormat

Java MessageFormat

weblate.checks.format.JavaMessageFormatCheck
java-messageformat
auto-java-messageformat
ignore-java-messageformat
There are {0} apples

JavaScript

JavaScript

weblate.checks.format.JavaScriptFormatCheck
davascript-format
ignore-javascript-format
There are %d apples

Lua

Lua

weblate.checks.format.LuaFormatCheck
lua-format
ignore-lua-format
There are %d apples

Object Pascal

Object Pascal

weblate.checks.format.ObjectPascalFormatCheck
object-pascal-format
ignore-object-pascal-format
There are %d apples
4.0 %.

weblate.checks.format.PercentPlaceholdersCheck
percent-placeholders
ignore-percent-placeholders
%number%

Perl

Perl

weblate.checks.format.PerlFormatCheck
perl-format
ignore-perl-format
There are %d apples
Your balance is %1$d %2$s

PHP

PHP

weblate.checks.format_PHPFormatCheck
php-format
ignore-php-format
There are %d apples
Your balance is %1$d %2$s

Python

Python

weblate.checks.format.PythonBraceFormatCheck
python-brace-format
ignore-python-brace-format
{} {amount} {currency}
Python

weblate.checks.format.PythonFormatCheck
python-format
ignore-python-format
There are %d apples
%(amount)d %(currency)s

Qt

weblate.checks.qt.QtFormatCheck
qt-format
ignore-qt-format
%1

Qt

weblate.checks.qt.QtPluralCheck
qt-plural-format
ignore-qt-plural-format
There are %Ln apple(s)

Ruby

weblate.checks.ruby.RubyFormatCheck
ruby-format
ignore-ruby-format
There are %d apples
%1$s %2$s
Your balance is %+2<amount>f %<currency>
Your balance is %{amount} %{currency}
Scheme

weblate.checks.format.SchemeFormatCheck
scheme-format
ignore-scheme-format

Weblate

Scheme

weblate.checks.format.SchemeFormatCheck
scheme-format
ignore-scheme-format

Scheme Format Check

weblate.checks.format.SchemeFormatCheck
scheme-format
ignore-scheme-format

Vue I18n

weblate.checks.format.VueFormattingCheck
vue-format
ignore-vue-format

Vue I18n

weblate.checks.format.VueFormattingCheck
vue-format
ignore-vue-format

Vue I18n

weblate.checks.consistency.TranslatedCheck
ignore-translated

Vue I18n

weblate.checks.consistency.TranslatedCheck
ignore-translated

Vue I18n

weblate.checks.consistency.TranslatedCheck
ignore-translated
Keeping translations same across components

**Kashida**

\[\text{weblate.checks.chars.KashidaCheck} \]
\[\text{ignore-kashida} \]

Kashida Wikipedia

**Markdown**

\[\text{weblate.checks.markup.MarkdownLinkCheck} \]
\[\text{ignore-md-link} \]

Markdown

\[\text{weblate.checks.markup.MarkdownRefLinkCheck} \]
\[\text{ignore-md-relink} \]

Markdown

\[\text{weblate.checks.markup.MarkdownSyntaxCheck} \]
\[\text{ignore-md-syntax} \]

Markdown span
weblate.checks.chars.MaxLengthCheck
max-length
ignore-max-length
max-length:100

replacements:
xml-text

weblate.checks.render.MaxSizeCheck
max-size
ignore-max-size
max-size:500:2, font-family:ubuntu, font-size:22

Component configuration

replacements:
xml-text

weblate.checks.chars.EscapedNewlineCountingCheck
ignore-escaped-newline


36
weblate.checks.chars.EndColonCheck
ignore-end-colon

weblate.checks.chars.EndEllipsisCheck
ignore-end-ellipsis

weblate.checks.chars.EndExclamationCheck
ignore-end-exclamation

weblate.checks.chars.EndStopCheck
ignore-end-stop

Wikipedia
weblate.checks.chars.PunctuationSpacingCheck
ignore-punctuation-spacing

Wikipedia:

weblate.checks.placeholders.RegexCheck
regex
ignore-regex

regex:^foo|bar$

weblate.checks.consistency.SamePluralsCheck
ignore-same-plurals

weblate.checks.chars.BeginNewlineCheck
ignore-begin-newline

3.9
weblate.checks.chars.BeginSpaceCheck
ignore-begin-space

weblate.checks.chars.EndNewlineCheck
ignore-end-newline

weblate.checks.chars.EndSpaceCheck
ignore-end-space

weblate.checks.same.SameCheck
ignore-same

strict-same

Component configuration:

---

40
HTML

**HTML 3.9**

```
weblate.checks.markup.SafeHTMLCheck
safe-html
ignore-safe-html
```

**HTML**

```
autofixer
```

**Markdown**

```
weblate.checks.markup.SafeHTMLCheck
```

**HTML Mozilla Bleach**

**URL**

**URL 3.5**

```
weblate.checks.markup.URLCheck
url
ignore-url
```

**XML**

**XML 2.8**

```
weblate.checks.markup.XMLTagsCheck
ignore-xml-tags
```

**XML**

```
weblate.checks.markup.XMLValidityCheck
ignore-xml-invalid
```

41
weblate.checks.chars.ZeroWidthSpaceCheck
ignore-zero-width-space

weblate.checks.source.EllipsisCheck
ignore-ellipsis

weblate.checks.icu.ICUSourceCheck
icu-message-format
ignore-icu-message-format

ICU MessageFormat syntax

ICU MessageFormat 4.9

ICU MessageFormat 4.1

weblate.checks.source.LongUntranslatedCheck
ignore-long-untranslated
weblate.checks.source.MultipleFailingCheck
ignore-multiple-failures

weblate.checks.format.MultipleUnnamedFormatsCheck
ignore-unnamed-format

weblate.checks.source.OptionalPluralCheck
ignore-optional-plural

Python gettext:

```python
from gettext import ngettext
print(ngettext("Selected %d file", "Selected %d files", files) % files)
```
"this is a quoted string" and 'another quoted string'
AND (source:hello OR source:bar)

AND (source:hello OR source:bar)

AND (source:hello OR source:bar)
\texttt{r"regexp"\
\texttt{\[2-5]"} 2 \sim 5\texttt{source:\r"[2-5]\"} 46
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.
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
access control

Gettext fuzzify

Weblate

per-project access control

OFF
OFF
OFF

ON
2.18 Weblate 2.18

per-project access control

50
Development process

- Developers
  - Intermediate file

Localization process

- Editors
  - Monolingual base language file
- Translators
  - Translation language file
Weblate

1. Weblate Component configuration
   Git
   Weblate

2. Weblate Component configuration URL
   Weblate

3. Weblate Component configuration
   Weblate
   Weblate

Avoiding merge conflicts

SSH

SSH

API

```
# 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/

# Update weblate remote:
git remote update weblate

# Merge Weblate changes:
git merge weblate/main

# Resolve conflicts:
```

52
edit ...
git add ...
...
git commit

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

# Open Weblate for translation:
wlc unlock

**Weblate**

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

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

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

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

**gettext PO**

```
# 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 '*\.*'`; 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
```

---

**Avoiding merge conflicts**

---

53
Weblate Project configuration

Component configuration

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

# Keeping translations same across components

Weblate Project configuration: see Component configuration.

## Adding translation projects and components

- `vcs/<project>/<component>/` DATA_DIR
- Git SSH
- Git Weblate
- Git Weblate
- Git Weblate

```bash
1. git submodule
2. git submodule add git@example.com:project-translations.git path/to/translations
3. git submodule update --remote path/to/translations
```

54
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.

### Component configuration

```bash
1. `git mv`;
2. `Component configuration`;
3. `report-source`
```
Weblate

Weblate pre-commit:

updating-target-files

Gettext PO:

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

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

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

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

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

---

**Weblate**
**Git**
**Mercurial**
**VCS**

---

**Weblate vcs-git**: [ref](vcs-github)
**Gerrit**
**Subversion**
**Mercurial**

---

```
57
```
Weblate

Weblate VCS

VCS:

list_translators

Weblate PO

Weblate PO 4.2

Weblate sr_Latn zh_Hant

Weblate POSIX

Weblate translate-toolkit

Translation Related File Formats

Weblate translate-toolkit

Translation Related File Formats

Weblate

GNU gettext

XLIFF

Apple iOS ID

Android

Weblate

Weblate
<table>
<thead>
<tr>
<th>Tool</th>
<th>Android</th>
<th>Apple iOS</th>
<th>PHP</th>
<th>JSON</th>
<th>JSON i18next</th>
<th>Java</th>
<th>mi18n</th>
<th>GWT</th>
<th>Joomla</th>
<th>Qt Linguist .ts</th>
<th>Arduino</th>
<th>Windows RC XML</th>
<th>Excel OpenXML</th>
<th>HTML Document</th>
<th>IDML</th>
<th>INI</th>
<th>Inno Setup INI</th>
<th>TermBase eXchange</th>
<th>Stringsdict</th>
<th>Fluent</th>
</tr>
</thead>
<tbody>
<tr>
<td>GNU gettext</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>gettext</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>XLIFF</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Java</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>mi18n</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>GWT</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Joomla</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Qt Linguist .ts</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Android</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Apple iOS</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>PHP</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>JSON</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>JSON i18next</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>go-i18n JSON</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>ARB</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>WebExtension</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>XML</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>CSV</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>YAML</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Ruby YAML</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>DTD</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>XML</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Windows RC XML</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Excel OpenXML</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>XML</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>HTML Document</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>IDML</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>ini</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Inno Setup INI</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>TermBase eXchange</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Stringsdict</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Fluent</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>
3.10.

Weblate XLIFF

Android

GNU gettext

libre

gettext PO:

#: 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"

Weblate Component configuration

```python
po/*.po
po/empty.pot
po/messages.pot
```

Gettext PO:

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

Weblate PO:

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

XML

weblate-flags

max-length:N

XLIFF

maxwidth

```html
<path>http://docs.oasis-open.org/xliff/v1.2/os/xliff-core.html#maxwidth</path>
```

Laravel

```bash
composer.phar Laravel
```

Webstate

```bash
composer.phar Webstate
```

60
```
#: weblate/media/js/bootstrap-datepicker.js:1421
msgid "day-tuesday"
msgstr "Úterý"

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

```
#: 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"
```

```
[[[ Weblate Component configuration ]]
po/*.po
po/en.po
po/messages.pot

Gettext PO file (monolingual)

[[[ XLIFF ]]
XML Localization Interchange File Format (XLIFF)

[[[ state ]]
approved="yes"
XLIFF Weblate

```

```
[[[ 3.3 ]]
Weblate 3.3 state

state
approved="yes" Weblate XLIFF

```

```
Should.

```

61
```
<trans-unit id="10" approved="yes">
  <source xml:space="preserve">hello</source>
  <target xml:space="preserve">Hello, world!</target>
</trans-unit>

weblate-flags:

<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>

font: Source Sans Pro (Source_Sans_Pro : fonts):

Weblate XLIFF Component configuration
localizations/*.xliff
localizations/en-US.xliff

XLIFF Wikipedia:
font attribute in XLIFF 1.2
maxwidth attribute in XLIFF 1.2
Java

Weblate ISO-8859-1 UTF-8 UTF-16 Unicode

Weblate Component configuration
src/app/Bundle_*.properties
src/app/Bundle.properties
Empty

Java ISO-8859-1

Java properties Wikipedia

mi18n

mi18n 4.7

mi18n JavaScript

GWT

GWT

GWT Component configuration
src/app/Bundle_*.properties
src/app/Bundle.properties
Empty
GWT
Weblate Component configuration

INI: language/*.ini
INI: language/en.ini
INI: Empty
INI: INI

INI Files
Joomla
Java
Inno Setup INI

Inno Setup INI

INI: language/*.isl
INI: language/en.islu
INI: Empty
INI: INI

INI Files
Unicode
ANSI

Joomla

INI: language/*/com_foobar.ini
INI: language/en-GB/com_foobar.ini
INI: Empty
INI: INI

INI Files
Mozilla and Java properties files

64
Qt Linguist .ts

Qt Linguist .ts

```
[...]
```

Webate Component configuration

```
i18n/app.*.ts
i18n/app.de.ts
```

Qt Linguist manual

```
Qt Linguist manual
```

Android

```
Android
```

```
Android string resources
```

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

strings.xml:

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

```
<string name="several_strings_0">First string</string>
<string name="several_strings_1">Second string</string>
```

strings.xml:

[https://gist.github.com/paour/11291062](https://gist.github.com/paour/11291062)
Apple iOS

iOS iPhone/iPad  Apple

Weblate Component configuration

Resources/*.lproj/Localizable.strings
Resources/en.lproj/Localizable.strings
Resources/Base.lproj/Localizable.strings

Empty

iOS (UTF-8)

Stringsdict Apple "strings files" documentation Mac OSX strings

PHP

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

Weblate Component configuration

lang/*/texts.php
lang/en/texts.php

Laravel PHP

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

PHP Localization in Laravel

JSON

2.0

2.16: Weblate 2.16 translate-toolkit 2.2.4 JSON 4.3: JSON i18next
go-i18n JSON

vue-i18n
react-intl

json

i18next

vue

go-i18n

66
WebExtension JSON
ARB
JSON
:
{
"Hello, world!\n": "Ahoj světe!\n",
"Orangutan has %d banana.\n": "",
"Try Weblate at https://demo.weblate.org/!\n": "",
"Thank you for using Weblate.": ""
}
:
{

}

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

: JSON
name

JSON

JSON

Weblate
:

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

Weblate Component conﬁguration
langs/translation-*.json
langs/translation-en.json
Empty
JSON
:
JSON updating-target-ﬁles JSON

,

JSON i18next ����
2.17
i18next

: Weblate 2.17

translate-toolkit 2.2.5

JavaScript

Weblate

i18next
: Weblate

i18next JSON v3

v2

v1

v4
:
{

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

}
67

i18next JSON


Weblate Component configuration
langs/*.json
langs/en.json
Empty

English:

i18next JSON

go-i18n JSON

ARB

WebExtension JSON
Weblate
Component configuration
_locales/*/messages.json
_locales/en/messages.json
Empty
WebExtension JSON

JSON
Google chrome.i18n
Mozilla Extensions Internationalization

/XML 2.3

/XML .resx
Microsoft .NET Resource files (.resx)
/updating-target-files
addon-weblate.cleanup.generic

/NET Resource files (.resx)
/update-target-files
/addon-weblate.cleanup.generic

/CSS

/NET Resource files (.resx)
/update-target-files
/addon-weblate.cleanup.generic

/CSV

/update-target-files
/addon-weblate.cleanup.generic

69
Thank you for using Weblate.

Děkujeme za použití Weblate.

### CSV

**Weblate Component configuration**

- locale/*.csv
- locale/en.csv

### YAML

**YAML Weblate Component configuration**

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

### Ruby YAML

```ruby
weblate:
  hello: ""
  orangutan: ""
  try: ""
  thanks: ""
```
Weblate Component configuration
locale/*.dtd
locale/en.dtd
Empty DTD

YAML

DTD

DTD 2.18
DTD

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

Weblate Component configuration
locale/*.dtd
locale/en.dtd
Empty DTD

Mozilla DTD format

XML 3.9
XML

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

Weblate Component configuration
locale/*.xml
locale/en.xml
Empty Flat XML

Flat XML

Windows RC

Windows RC 4.1

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

```
lang/*.rc
lang/en-US.rc
```

**Windows RC files**

---

**Triple-T gradle-play-publisher**

Fastlane

**F-Droid**

---

**Weblate Component configuration**

```
fastlane/android/metadata/*
fastlane/android/metadata/en-US
```

**changelogs:** :ref:`custom-checks`

---

**Weblate Component configuration**

```
/*.srt
/en.srt
```

**SubRip**

---

**MicroDVD**

**Advanced Substation Alpha**

**Substation Alpha**

---

**Subtitles**

---
Excel Open XML

3.2 XML.

Excel Open XML.xlsx source context Excel

XLSX target context XLSX

HTML

4.1 HTML.

DokuWiki MediaWiki

Simple Text Documents

OpenDocument

4.1 HTML.

OpenDocument Format

IDML
TermBase eXchange

TBX Wikipedia

Stringsdict

Apple iOS

Fluent

Fluent Web
Translation Related File Formats

Weblate: Git, GitHub, GitLab, Bitbucket, Codeberg, Gerrit, Subversion, Mercurial

VCS Weblate URL: https://github.com/WeblateOrg/weblate.git

Hosted Weblate

GitHub: hosted@weblate.org

SSH: git@github.com:WeblateOrg/weblate.git

SSH: Weblate push user

SSH: Weblate
Weblate SSH

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

```
AAAAB3NzaC1yc2EAAAADAQABAAABAMCAQc+Xqg6sARHq3WspBYXGtjSwePr7e9zPe6hA8UI/QQmogBiy7v+ni3HNwQYmjqQ81EEEs41r9bJm/x7MXY6Q1aw0mzf
```

[Download private key]

Known host keys

<table>
<thead>
<tr>
<th>Hostname</th>
<th>Keytype</th>
<th>Fingerprint</th>
</tr>
</thead>
<tbody>
<tr>
<td>github.com</td>
<td>ssh-ed25519</td>
<td>-H3evw4V4TuJhbpZ5s5zlDA0PFMsHdk4JhC9U</td>
</tr>
<tr>
<td>github.com</td>
<td>ed25519</td>
<td>p2QAMKnc17/WeI0rThV98/RBUSFyU3/Li4qGUGQ</td>
</tr>
<tr>
<td>github.com</td>
<td>ssh-rsa</td>
<td>nTHzg6KcJg/JWZ7E6GOcRsRmTdsEJcAR/uKuWE/8598</td>
</tr>
</tbody>
</table>

Add host key

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

[Submit]

Webate SSH

Webate SSH

SSH
SSH: GitHub

Weblate: SSH

GitHub

SSH

SSH

Weblate

SSH

Weblate

SSH

Weblate

SSH

Creating an access token for command-line use

GitHub

SSH

SSH

SSH

SSH
Weblate URL

```
weblate :// project/component
```

VCS:

```
GitHub URL: https://user:your_access_token@github.com/WeblateOrg/weblate.git

URL: https://user@example.com:%24password%23@bitbucket.org/…
```

HTTPS

```
GitHub URL: https://user:your_access_token@github.com/WeblateOrg/weblate.git
```

HTTP/HTTPS VCS

```
http_proxy https_proxy all_proxy
```

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

The cURL manpage

```
```

Git

```
```

```
```

78
Webtlate VCS HOME=DATA_DIR/home DATA_DIR/home/.git

remote helpers
Bazaar  Mercurial  GitHub  GitHub  Bazaar  Launchpad  gnuhhello

bzr::lp:gnuhello

Mercurial selenic.com: http://selenic.com/repo/hello

Git  Mercurial  tip

GitHub 2.3

GitLab 3.9

Pushing changes from Weblate

Pushing changes from Weblate
### Pagure

4.3.2 API

Git

Git

Git

Pagure: Pushing changes from Weblate

PAGURE_CREDENTIALS

PAGURE_USERNAME

PAGURE_TOKEN

### Gerrit

2.2 API

git-review

Gerrit

Gerrit

### Mercurial

2.1 API

Mercurial

Weblate

### Subversion

2.8 API

Subversion

Web328

Subversion

Perl

### Subversion

2.19 API

Subversion

Subversion

URL

Subversion

URL

# Use DATA_DIR as configured in Weblate settings.py, it is /app/data in...

HOME=$DATA_DIR/home svn co https://svn.example.com/example

DATA_DIR
Git

Weblate VCS

Weblate Git VCS Weblate

Web   Git  VCS  Weblate

Weblate REST API

2.6: REST API Weblate 2.6

API /api/ URL Django REST framework Weblate

API 100

Authorization:

ANY /

API

format -- Accept: Accept  REST  Accept: Accept  json  API  Web  Accept: Accept


Accept -- Accept: Accept

Authorization: Token YOUR-TOKEN Accept: Accept

Content-Type: Accept: Accept

Allow -- HTTP: HTTP: JSON: HTTP: JSON

detail(string) -- 200 OK HTTP HTTP

count(int) -- Accept: Accept

next(string) -- Accept: Accept

previous(string) -- Accept: Accept

results(string) -- Accept: Accept

url(string) -- Accept: Accept

web_url(string) -- Accept: Accept

200 OK

201 Created

204 No Content

400 Bad Request

403 Forbidden

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

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

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

CURL MK: 
curl -H "Authorization: Token TOKEN" 
https://example.com/api/

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 MK:

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

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

CURL JSON

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

API

```
GET /api/
```

**header**

- Host: example.com
- Accept: application/json, text/javascript
- Authorization: Token YOUR-TOKEN

**response**

```
HTTP/1.1 200 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/"
}
```
GET /api/users/

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

POST /api/users/

username (string) -- 
full_name (string) -- 
email (string) -- 
is_superuser (boolean) -- 
is_active (boolean) -- 
is_bot (boolean) -- Is user bot? (optional) (used for project scoped tokens)

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

username (string) -- 
full_name (string) -- 
email (string) -- 
is_superuser (boolean) -- 
is_active (boolean) -- 
is_bot (boolean) -- Is user bot? (optional) (used for project scoped tokens)

date_joined (string) -- 
groups (array) -- 

JSON:

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

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

username (string) -- 
full_name (string) -- 
email (string) --
subscription_id (int) -- ID

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

**int**: subscription_id

**username**(string) -- ID

**subscription_id**(int) -- ID

**JSON**

**notification**(string) -- ID

**scope**(int) -- ID

**frequency**(int) -- ID

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

**int**: subscription_id

**username**(string) -- ID

**subscription_id**(int) -- ID

**JSON**

**notification**(string) -- ID

**scope**(int) -- ID

**frequency**(int) -- ID

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

**int**: subscription_id

**username**(string) -- ID

**subscription_id**(int) -- ID

**subscription_id**(int) -- int

GET /api/groups/

**id**(int) -- ID

**JSON**

**name**(string) -- ID

**project_selection**(int) -- ID

**language_selection**(int) -- ID

**defining_project**(str) -- ID

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

**POST** /api/groups/

**name**(string) -- ID

**project_selection**(int) -- ID

**language_selection**(int) -- ID

**defining_project**(str) -- ID

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

**id**(int) -- ID

**name**(string) -- ID

**project_selection**(int) -- ID

**language_selection**(int) -- ID

**roles**(array) -- ID

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

**projects**(array) -- ID

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

4.0
components (array) -- GET /api/components/(string:project)/
(string:component)/

componentlists (array) -- GET /api/component-lists/(str:slug)/

defining_project (str) -- GET /api/projects/(string:project)/

JSON example:

```json
{
   "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)/

<table>
<thead>
<tr>
<th>id(int)</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>name(string)</td>
<td>JSON name</td>
</tr>
<tr>
<td>project_selection(int)</td>
<td></td>
</tr>
<tr>
<td>language_selection(int)</td>
<td></td>
</tr>
</tbody>
</table>

PATCH /api/groups/(int: id)/

<table>
<thead>
<tr>
<th>id(int)</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>name(string)</td>
<td>JSON name</td>
</tr>
<tr>
<td>project_selection(int)</td>
<td></td>
</tr>
<tr>
<td>language_selection(int)</td>
<td></td>
</tr>
</tbody>
</table>

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

| id(int) | ID |

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

| id(int) | ID |
| string role_id | ID |

POST /api/groups/(int: id)/components/
id (int) -- `ID`

string component_id -- `component_id`

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

id (int) -- `ID`

component_id (int) -- `component_id`

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

id (int) -- `ID`

string project_id -- `project_id`

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

id (int) -- `ID`

project_id (int) -- `project_id`

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

id (int) -- `ID`

string language_code -- `language_code`

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

id (int) -- `ID`

language_code (string) -- `language_code`

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

id (int) -- `ID`

string component_list_id -- `component_list_id`

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

id (int) -- `ID`

component_list_id (int) -- `component_list_id`

GET /api/roles/

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

POST /api/roles/

name (string) -- `name`

permissions (array) -- `permissions`
GET /api/roles/(int: id)/

id(int) -- int ID

name(string) -- JSON

permissions(array) -- JSON

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

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

id(int) -- int ID

name(string) -- JSON

permissions(array) -- JSON

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

id(int) -- int ID

name(string) -- JSON

permissions(array) -- JSON

DELETE /api/roles/(int: id)/

id(int) -- int ID

GET /api/languages/

code(string) -- JSON

name(string) -- JSON

direction(string) -- JSON

plural(object) -- JSON

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

language(string) -- JSON

code(string) -- JSON
direction (string) -- 
plural (object) -- 
aliases (array) -- 

JSON example:

```json
{
    "code": "en",
    "direction": "ltr",
    "name": "English",
    "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) /

name (string) --

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

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

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

```
```
translated_words (int) -- 
translated_words_percent (int) -- 
translated_chars (int) -- 
translated_chars_percent (int) -- 
total_chars (int) -- 
fuzzy (int) -- 
fuzzy_percent (int) -- 
failing (int) -- 
failing -- 

GET /api/projects/

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

POST /api/projects/

name (string) -- 
slug (string) -- 
web (string) -- Web

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

project (string) -- URL

JSON

name (string) -- 
slug (string) -- 
web (string) -- Web

components_list_url (string) -- URL

repository_url (string) -- URL

changes_list_url (string) -- URL

translation_review (boolean) -- 
source_review (boolean) -- 
set_language_team (boolean) -- "Language-Team"

enable_hooks (boolean) -- 
instructions (string) -- 
language_aliases (string) -- 

JSON:

```json
{
    "name": "Hello",
    "slug": "hello",
    "url": "http://example.com/api/projects/hello/",
    "web": "https://weblate.org/",
    "web_url": "http://example.com/projects/hello/"
}
```
PATCH /api/projects/(string: project)/

GET /api/changes/

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

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

GET /api/changes/(int:id)/

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

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

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

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

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

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

JSON:

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

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

JSON:

```
92
```
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"}

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

JSON request to create a new component from Git:

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

{
  "branch": "main",
  "file_format": "po",
  "filemask": "po/*.po",
  "name": "Weblate",
  "slug": "weblate",
  "repo": "https://github.com/WeblateOrg/hello.git",
  "template": "",
  "new_base": "po/hello.pot",
  "vcs": "git"
}

JSON request to create a new component from another one:

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

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

GET /api/projects/(string: project)/languages/
3.8 API

project (string) -- [HTML] URL [HTML]
results (array) -- [JSON]
language (string) -- [JSON]

code (string) -- [JSON]
total (int) -- [JSON]
translated (int) -- [JSON]
translated_percent (float) -- [JSON]
total_words (int) -- [JSON]
translated_words (int) -- [JSON]
words_percent (float) -- [JSON]

GET /api/projects/(string: project)/statistics/
3.8 API

project (string) -- [HTML] URL [HTML]

95
total(int) -- integer
translated(int) -- integer
translated_percent(float) -- float
total_words(int) -- integer
translated_words(int) -- integer
words_percent(float) -- float

### API

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

**GET /api/components/**

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

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

**project (string) -- integer URL**

**component (string) -- integer URL**

**project (object) -- integer URL**

**name (string) -- integer**

**slug (string) -- integer**

**vcs (string) -- integer**

**repo (string) -- integer**

**git_export (string) -- integer URL**

**branch (string) -- integer**

**push_branch (string) -- integer push**

**filemask (string) -- integer**

**template (string) -- integer**

**edit_template (string) -- integer**

**intermediate (string) -- integer**

**new_base (string) -- integer**

**file_format (string) -- integer**

**license (string) -- integer**

**agreement (string) -- integer**

**new_lang (string) -- integer**

**language_code_style (string) -- integer**

**source_language (object) -- integer GET /api/languages/(string:language)/**

**push (string) -- integer URL**

**check_flags (string) -- integer**

**priority (string) -- integer**

**enforced_checks (string) -- integer**

**restricted (string) -- integer**

**repoweb (string) -- integer**

**report_source_bugs (string) -- integer**
merge_style(string) -- [ ]
commit_message(string) -- [ ]
add_message(string) -- [ ]
delete_message(string) -- [ ]
merge_message(string) -- [ ]
addon_message(string) -- [ ]
allow_translation_propagation(string) -- [ ]
enable_suggestions(string) -- [ ]
suggestion_voting(string) -- [ ]
suggestion_autoaccept(string) -- [ ]
push_on_commit(string) -- [ ]
commit_pending_age(string) -- [ ]
auto_lock_error(string) -- [ ]
language_regex(string) -- [ ]
variant_regex(string) -- [ ]
repository_url (string) -- URL: GET /api/components/(string:project)/(string:component)/repository/
translations_url (string) -- URL: GET /api/components/(string:project)/(string:component)/translations/
lock_url (string) -- URL: GET /api/components/(string:project)/(string:component)/lock/
changes_list_url (string) -- URL: GET /api/components/(string:project)/(string:component)/changes/
task_url(string) -- URL: GET /api/tasks/(str:uuid)/
JSON JSON:

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

project (string)-- URL
component (string)-- URL
source_language (string)--

ame (string)--
slug (string)--
repo (string)-- VCS URL

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

JSON:

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"
}

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",
         "name": "English",
         "url": "http://example.com/api/languages/en/",
      }
   }
}
```plaintext
PUT /api/components/(string: project) /
string: component/ PUT

project (string) -- [PROJECT URL](http://example.com/projects/hello/)
component (string) -- [COMPONENT URL](http://example.com/api/projects/hello/weblate/)
branch (string) -- [VCS URL](file:///home/nijel/work/weblate-hello)
file_format (string) -- [FILEFORMAT]
filemask (string) -- [FILEMASK]
name (string) -- [NAME]
slug (string) -- [SLUG]
repo (string) -- [VCS URL](http://example.com/projects/hello/weblate/)
template (string) -- [TEMPLATE]
new_base (string) -- [NEW_BASE]
vcs (string) -- [VCS]

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

project (string) -- [PROJECT URL](http://example.com/projects/hello/)
component (string) -- [COMPONENT URL](http://example.com/api/projects/hello/weblate/)
GET /api/components/(string: project) /
string: component/changes/ [GET JSON]

project (string) -- [PROJECT URL](http://example.com/projects/hello/)
component (string) -- [COMPONENT URL](http://example.com/api/projects/hello/weblate/)
results (array) -- [RESULTS]
GET /api/changes/(int:id)/

GET /api/components/(string: project) /
string: component/file/ [GET ZIP]

project (string) -- [PROJECT URL](http://example.com/projects/hello/)
component (string) -- [COMPONENT URL](http://example.com/api/projects/hello/weblate/)
format (string) -- [FILEFORMAT]; zip: [FILEFORMAT]; zip
q (string) -- [QUERY]
GET /api/components/(string: project) /
string: component/screenshots/ [GET ZIP]
```

99
GET /api/screenshots/(int:id)/
GET /api/components/(string:project)/
string: component/lock/

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

GET /api/projects/(string:project)/repository/
GET /api/components/(string:project)/(string:component)/lock/

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

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

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

{"locked": true}
```

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

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

{"locked":true}
needs_commit (boolean)
needs_merge (boolean)
needs_push (boolean)
remote_commit (string)
status (string)
merge_failure -- null
POST /api/components/(string: project)/
string: component/repository/VCS

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

project (string)
component (string)

operation (string) -- push, pull, commit, reset, cleanup

result (boolean)

CURL:

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

JSON:

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

{"operation":"pull"}

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/

project (string)
component (string)

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

project (string)
component (string)

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

101
**project** (string) -- URL

**component** (string) -- URL

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

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

**project** (string) -- URL

**component** (string) -- URL

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

**result** (object) --

**CURL**:

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

**JSON**:

```json
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**:

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

{
   "failing_checks": 0,
   "failing_checks_percent": 0,
   "failing_checks_words": 0,
   "filename": "po/cs.po",
   "fuzzy": 0,
   "fuzzy_percent": 0.0,
   "fuzzy_words": 0,
   "have_comment": 0,
   "have_suggestion": 0,
   "is_template": false,
   "is_source": false,
   "language": {
      "code": "cs",
      "direction": "ltr",
      "name": "Czech",
      "url": "http://example.com/api/languages/cs/",
      "web_url": "http://example.com/languages/cs/"
   },
   "language_code": "cs",
}  
```

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

HTTP 2.7 OK.

project (string) -- URL 
component (string) -- URL 
JSON

results (array) -- 
GET /api/translations/(string:project)/ (string:component)/(string:language)/statistics/

GET /api/components/(string: project) /
string: component/links/

HTTP 4.5 OK.

project (string) -- URL 
component (string) -- URL 
JSON

projects (array) -- 
GET /api/projects/(string:project)/

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

HTTP 4.5 OK.

project (string) -- URL 
component (string) -- URL 

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

HTTP 4.5 OK.

project (string) -- URL 
component (string) -- URL 
project_slug(string) -- 
GET /api/translations/

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

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

project(string)-- URL
component(string)-- URL
language(string)-- JSON

component (object): GET /api/components/(string:project)/(string:component)/

failing_checks(int)--
failing_checks_percent(float)--
failing_checks_words(int)--
filename(string)--
fuzzy(int)--
fuzzy_percent (float)--
fuzzy_words (int)--
have_comment(int)--
have_suggestion(int)--
is_template(boolean)--
language (object): GET /api/languages/(string:language)/
language_code (string)--
last_author (string)--
last_change (timestamp)--
revision (string)--
share_url (string)-- URL

total(int)--
total_words (int)--
translate_url (string)-- URL
translated(int)--
translated_percent(float)--
translated_words (int)--

repository_url (string)-- URL
file_url (string)-- URL
changes_list_url (string)-- URL
units_list_url (string)-- URL

JSON: 104
component (string) -- URL
language (string) -- URL

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

project (string) -- URL
component (string) -- URL
language (string) -- URL

results (array) -- JSON
GET /api/changes/ (int:id)/

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

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

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

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

project (string) -- URL
component (string) -- URL
language (string) -- URL

results (array) -- JSON
GET /api/units/ (int:id)/

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

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

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

VCS format: ref: download

API format: VCS

API format: VCS
format -- po, mo, xlf, xlf11, tbx, csv, xlsx, json, aresource, strings

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

project (string) -- URL
component (string) -- URL
language (string) --
string conflict -- ignore, replace-translated, replace-approved
file file --
string email --
string author --
string method -- translate, approve, suggest, fuzzy, replace, source, add, delete
string fuzzy --
CURL:
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/

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

project (string) -- URL
component (string) -- URL
language (string) --
POST /api/translations/(string: project)/
string: component/string: language/repository/ VCS

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

project (string) -- URL
component (string) -- URL
language (string) --
JSON
operation (string) -- push, pull, commit, reset, cleanup
result (boolean) --
GET /api/translations/(string: project)/
string: component/string: language/statistics/

2.7

project (string) -- URL
component (string) -- URL
language (string) --
JSON

107
code (string) --  
failing (int) --  
failing_percent (float) --  
fuzzy (int) --  
fuzzy_percent (float) --  
total_words (int) --  
translated_words (int) --  
last_author (string) --  
last_change (timestamp) --  
name (string) --  
total (int) --  
translated (int) --  
translated_percent (float) --  
url (string) --  
url_translate (string) --  

unit  

Translate Toolkit  XLIFF

GET /api/units/

GET /api/units/(int:id)/

id (int) -- ID

translation (string) -- URL

source (array) --

previous_source (string) --

target (array) --

id_hash (string) --

content_hash (string) --

location (string) --

context (string) --

note (string) --

flags (string) --

state (int) -- 0 - 10 - 20 - 30 - 100 - 108

fuzzy (boolean) --

translated (boolean) --

approved (boolean) --

position (int) --

has_suggestion (boolean) --

has_comment (boolean) --

has_failing_check (boolean) --
num_words (int) --

priority (int) -- 100

id (int) --

explanation (string) -- Additional info on source strings
extra_flags (string) --
web_url (string) -- URL
source_unit (string) -- GET /api/units/(int:id)/
pending (boolean) -- whether the unit is pending for write
timestamp (timestamp) -- string age

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

PUT /api/units/(int: id)/

DELETE /api/units/(int: id)/

GET /api/changes/

GET /api/changes/(int:id)/

GET /api/units/

GET /api/units/(int:id)/

POST /api/units/

GET /api/units/(int:id)/

GET /api/changes/

GET /api/changes/(int:id)/

GET /api/units/

GET /api/units/(int:id)/
id(int)-- ID
unit(string)-- URL
translation(string)-- URL
component(string)-- URL
user(string)-- URL
author(string)-- URL
timestamp(timestamp)--
action(int)--
action_name(string)--
target(string)--
id(int)--

2.14

GET /api/screenshots/

GET /api/screenshots/(int:id)/

id(int)-- ID

name(string)--

component(string)--

file_url(string)--

units(array)--

GET /api/screenshots/(int:id)/file/

id(int)-- ID

POST /api/screenshots/(int:id)/file/

file image--

CURL:

curl -X POST \
    -F image=@image.png \
    -H "Authorization: Token TOKEN" \
    http://example.com/api/screenshots/1/file/

POST /api/screenshots/(int:id)/units/

id(int)-- ID

string unit_id-- ID

name(string)--
translation(string) -- URL
file_url(string) -- URL: GET /api/screenshots/(int:id)/file/
units(array) GET /api/units/(int:id)/
DELETE /api/screenshots/(int: id)/units/
int: unit_id

id(int) -- ID
unit_id -- ID
POST /api/screenshots/

file image --
string name
string project_slug
string component_slug
string language_code

JSON name
component
file_url -- URL
units(array) GET /api/units/(int:id)/
PATCH /api/screenshots/(int: id)/

id(int) -- ID
name
component
file_url -- URL
units(array) GET /api/units/(int:id)/
PUT /api/screenshots/(int: id)/

id(int) -- ID
name
component
file_url -- URL
units(array) GET /api/units/(int:id)/
DELETE /api/screenshots/(int: id)/

id(int) -- ID
### 4.0 API

GET /api/component-lists/

#### GET /api/component-lists/(str:slug)/

slug (string) -- project identifier

#### GET /api/component-lists/ (str: slug)/

slug (string) -- project identifier

### 4.1 API

GET /api/addons/

#### GET /api/addons/(int: id)/

id (int) --Addon ID

name (string) -- Name

component (string) -- Component URL

configuration (object) -- Configuration

#### POST /api/components/ (string: project) /

project slug (string) -- Project slug

component slug (string) -- Component slug

name (string) -- Name

configuration (object) -- Configuration

#### PATCH /api/addons/ (int: id)/

id (int) --Addon ID

configuration (object) -- Configuration

#### PUT /api/addons/ (int: id) /

id (int) --Addon ID

configuration (object) -- Configuration

#### DELETE /api/addons/ (int: id)/

id (int) --Addon ID
CREATE JSON NAME
slug(string)-- JSON
name(string)-- JSON
show_dashboard(boolean)-- JSON
components(array)-- JSON
auto_assign(array)-- JSON
PUT /api/component-lists/(str: slug)/
slug(string)-- JSON
name(string)-- JSON
slug(string)-- JSON
show_dashboard(boolean)-- JSON
PATCH /api/component-lists/(str: slug)/
slug(string)-- JSON
name(string)-- JSON
slug(string)-- JSON
show_dashboard(boolean)-- JSON
DELETE /api/component-lists/(str: slug)/
slug(string)-- JSON
POST /api/component-lists/(str: slug)/components/
slug(string)-- JSON
string component_id-- ID
DELETE /api/component-lists/(str: slug)/components/
slug(string)-- JSON
component_slug(string)-- JSON

113
GET /api/tasks/

GET /api/tasks/(str: uuid) /

type: uuid

completed: boolean

progress: int

result: object

log: string

GET /api/metrics/

units: int

units_translated: int

users: int

changes: int

projects: int

components: int

translations: int

languages: int

checks: int

configuration_errors: int

suggestions: int

celery_queues: object

name: string

VCS Weblate

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

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

string: component

2.6 POST /api/components/(string:project)/(string:component)/repository/ACL

VCS GitHub

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

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

VCS GitHub

POST /hooks/github/

GitHub
Automatically receiving changes from GitHub
GitHub
GitHub Webhooks
Weblate
POST /hooks/gitlab/
GitLab
Automatically receiving changes from GitLab
GitLab
https://docs.gitlab.com/ee/user/project/integrations/webhooks.html
GitLab Webhooks
Weblate
POST /hooks/bitbucket/
Bitbucket
Automatically receiving changes from Bitbucket
Bitbucket
https://support.atlassian.com/bitbucket-cloud/docs/manage-webhooks/
Bitbucket Webhooks
Weblate
POST /hooks/pagure/
Pagure
Pagure
https://docs.pagure.org/pagure/usage/using_webhooks.html
Pagure Webhooks
Weblate
POST /hooks/azure/
Azure Repos
Azure Repos
Azure Repos Web Hooks
Weblate
POST /hooks/gitea/
Gitea
Automatically receiving changes from Gitea Repos
Gitea
https://docs.gitea.io/en-us/webhooks/
Gitea Webhooks
Weblate
POST /hooks/gitee/
Gitee Webhook
Automatically receiving changes from Gitee Repos

Gitee
https://gitee.com/help/categories/40

ENABLE_HOOKS

Weblate

GET /exports/stats/(string: project)/

string: component/

format (string) -- HTML json csv

2.6

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

statistics/

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

(string:language)/statistics/

ACL

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

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/
\nel/"
}
Docker

Weblate Docker Hub: https://hub.docker.com/u/weblate

Docker Hub:

```
docker pull weblate/wlc
```

Docker Weblate localhost API URL API_KEY

```
docker run --rm weblate/wlc
```

```
WLC_ARGS
```

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

You might want to pass your `API_KEY` 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
```

```
~/.config/weblate
```

for other locations

```
[weblate]
url = https://hosted.weblate.org/api/

[keys]
https://hosted.weblate.org/api/ = APIKEY
```

```
wlc ls
wlc commit sandbox/hello-world
```

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

Weblate Python Weblate REST API Weblate

```
wlc
```

118
---format {csv, json, text, html}

---url URL
API URL: https://hosted.weblate.org/api/

---key KEY
API KEY: Weblate

---config PATH
---config-section SECTION

version
list-languages
list-projects
list-components
list-translations
show
ls

commit
pull
push
reset
cleanup
repo
statistics
lock-status
lock
unlock
0.5: wlc 0.5

Weblate
changes
0.7: wlc 0.7 Weblate 2.10

download
0.7: wlc 0.7

--convert

--output

upload
0.9: wlc 0.9

--overwrite

--input

--method

--fuzzy
```
process```` approve````

--author-name

--author-email

```
--help
```

1.6: .ini

Windows

XDG

XDG_CONFIG_HOME

XDG_CONFIG_DIRS

[weblate]

--config-section

key

Weblate API URL

url

API URL http://127.0.0.1:8000/api/

translation

INI

120
[weblate]
url = https://hosted.weblate.org/api/
key = APIKEY
translation = weblate/application

[keys]
https://hosted.weblate.org/api/ = APIKEY

[vcs]
weblate = wlc

$ wlc version
version: 0.1

$ 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/

$ wlc upload project/component/language --input /tmp/hello.po

$ cat .weblate
[weblate]
[url] = https://hosted.weblate.org/api/
[key] = APIKEY
[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/

$ wlc commit
Weblate Python API

pip install wlc

wlc

WeblateException

exception: wlc.WeblateException

Weblate

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

key (str) -- key
url (str) -- API URL
config (wlc.config.WeblateConfig) -- API URL

get (path)

post (path, **kwargs)

WeblateConfig

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

section (str) -- XDG
load (path=None)

path (str) -- XDG

config/wlc
wlc.main

wlc.main.main(settings=None, stdout=None, args=None)

settings (list) --
stdout (object) -- sys.stdout
args (list) -- sys.args

@wlc.main.register_command(command)
main() main() main() main()

Command

class wlc.main.Command(args, config, stdout=None)

Weblate

Docker

Docker  Weblate  PostgreSQL  Weblate  Weblate  Weblate

2 GB  RAM
2 CPU
1 GB HDD or SSD

Weblate  Weblate  Weblate  Weblate  Weblate  Weblate

2 GB  RAM
2 CPU
1 GB HDD or SSD

1. weblate-docker:
   git clone https://github.com/WeblateOrg/docker-compose.git weblate-docker
   cd weblate-docker

2. docker-compose.override.yml:
   ref:`docker-environment`
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

3. Weblate:

```
docker-compose up
```

Enjoy your Weblate deployment, it's accessible on port 80 of the weblate container.

**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.

**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.

**Invoking management commands**

**Choosing Docker hub tag**

You can use following tags on Docker hub, see [https://hub.docker.com/r/weblate/weblate/tags/](https://hub.docker.com/r/weblate/weblate/tags/) for full list of available ones.

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

Every image is tested by our CI before it gets published, so even the `bleeding` version should be quite safe to use.
Docker container with HTTPS support

Please see [Instructions] for generic deployment instructions, this section only mentions differences compared to it.

Using own SSL certificates

In case you have own SSL certificate you want to use, simply place the files into the Weblate data volume (see Docker container volumes):

- `ssl/fullchain.pem` containing the certificate including any needed CA certificates
- `ssl/privkey.pem` containing the private key

Both of these files must be owned by the same user as the one starting the docker container and have file mask set to 600 (readable and writable only by the owning user).

Additionally, Weblate container will now accept SSL connections on port 4443, you will want to include the port forwarding for HTTPS in docker compose override:

```yaml
version: '3'
services:
  weblate:
    ports:
      - 80:8080
      - 443:4443
```

If you already host other sites on the same server, it is likely ports 80 and 443 are used by a reverse proxy, such as NGINX. To pass the HTTPS connection from NGINX to the docker container, you can use the following configuration:

```yaml
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.

Automatic SSL certificates using Let's Encrypt

In case you want to use Let's Encrypt automatically generated SSL certificates on public installation, you need to add a reverse HTTPS proxy an additional Docker container, `https-portal` will be used for that. This is made use of in the `docker-compose-https.yml` file. Then create a `docker-compose-https.override.yml` file with your settings:

```yaml
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
```

125
Whenever invoking `docker-compose` you need to pass both files to it, and then do:

```
docker-compose -f docker-compose-https.yml -f docker-compose-https.override.yml build
docker-compose -f docker-compose-https.yml -f docker-compose-https.override.yml up
```

### Upgrading the Docker container

Usually it is good idea to only update the Weblate container and keep the PostgreSQL container at the version you have, as upgrading PostgreSQL is quite painful and in most cases does not bring many benefits.

```
4.10-1  Weblate 4.10-1  Docker PostgreSQL 10  Django 4.0
```

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

```
# Fetch latest versions of the images
docker-compose pull
# Stop and destroy the containers
docker-compose down
# Spawn new containers in the background
docker-compose up -d
# Follow the logs during upgrade
docker-compose logs -f
```

The Weblate database should be automatically migrated on first startup, and there should be no need for additional manual actions.

---

*Note:* Upgrades across 3.0 are not supported by Weblate. If you are on 2.x series and want to upgrade to 3.x, first upgrade to the latest 3.0.1-x (at time of writing this it is the 3.0.1-7) image, which will do the migration and then continue upgrading to newer versions.

```
docker-compose pull
```

---

### PostgreSQL

```
https://github.com/docker-library/postgres/issues/37
```

1. **Weblate**:

    ```
docker-compose stop weblate cache
    ```

2. **PostgreSQL**:

    ```
docker-compose exec database pg_dumpall --clean --username weblate >...
    ```

3. **PostgreSQL**:

    ```
docker-compose stop database
    ```

4. **PostgreSQL**:

    ```
docker-compose rm -v database
docker volume remove weblate_postgres-data
```
5. PostgreSQL file: `docker-compose.yml`

6. 

docker-compose up -d database

7. 

cat backup.sql | docker-compose exec -T database psql --username weblate --

8. 

docker-compose up -d

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.

WEBLATE_ADMIN_PASSWORD WEBLATE_ADMIN_NAME WEBLATE_ADMIN_EMAIL

Number of processes and memory consumption

The number of worker processes for both uWSGI and Celery is determined automatically based on number of CPUs. This works well for most cloud virtual machines as these typically have few CPUs and good amount of memory.

In case you have a lot of CPU cores and hit out of memory issues, try reducing number of workers:

```
environment:
WEBLATE_WORKERS: 2
```

You can also fine-tune individual worker categories:

```
environment:
WEBLATE_WORKERS CELERY_MAIN_OPTIONS:
WEB_WORKERS: 4
CELERY_NOTIFY_OPTIONS: --concurrency 2
CELERY_TRANSLATE_OPTIONS: --concurrency 1
CELERY_MEMORY_OPTIONS, CELERY_BACKUP_OPTIONS, CELERY_BEAT_OPTIONS, WEB_WORKERS
```

Scaling horizontally

```
4.6 REPO.
```

You can run multiple Weblate containers to scale the service horizontally. The /app/data volume has to be shared by all containers, it is recommended to use cluster filesystem such as GlusterFS for this. The /app/cache volume should be separate for each container.

Each Weblate container has defined role using WEBLATE_SERVICE environment variable. Please follow carefully the documentation as some of the services should be running just once in the cluster and the ordering of the services matters as well.

You can find example setup in the docker-compose repo as docker-compose-split.yml.
Docker environment variables

Many of Weblate's settings can be set in the Docker container using environment variables:

Generic settings

**WEBLATE_DEBUG**
Configures Django debug mode using `DEBUG`.

```
environment:
  WEBLATE_DEBUG: 1
```

**WEBLATE_LOGLEVEL**
Configures the logging verbosity.

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

**WEBLATE_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).

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

**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`.

> 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.

**WEBLATE_ADMIN_PASSWORD_FILE**
Sets the path to a file containing the password for the `admin` user.

**WEBLATE_SERVER_EMAIL**
The email address that error messages are sent from.
**WEBLATE_DEFAULT_FROM_EMAIL**

**DEFAULT_FROM_EMAIL**

**WEBLATE_CONTACT_FORM**

**CONTACT_FORM**

**WEBLATE_ALLOWED_HOSTS**

`ALLOWED_HOSTS` HTTP `*` 

**WEBLATE_REGISTRATION_OPEN**

Configures whether registrations are open by toggling `REGISTRATION_OPEN`.

**WEBLATE_REGISTRATION_ALLOW_BACKENDS**

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

**WEBLATE_TIME_ZONE**

Configures the used time zone in Weblate, see `TIME_ZONE`.

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

**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.

Please see `ENABLE_HTTPS` documentation for possible caveats.

This does not make the Weblate container accept HTTPS connections, you need to configure that as well, see `Docker container with HTTPS support` for examples.

**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`.

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`.

```
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.

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

`WEBLATE_REQUIRE_LOGIN`
Enables `REQUIRE_LOGIN` to enforce authentication on whole Weblate.

```
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`.

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

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

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

---

`130`
WEBLATE_PAGURE_USERNAME
PAGURE_USERNAME
Pagure merge-requests

WEBLATE_PAGURE_TOKEN
PAGURE_TOKEN
Pagure API

WEBLATE_DEFAULT_PULL_MESSAGE
DEFAULT_PULL_MESSAGE

WEBLATE_SIMPLIFY_LANGUAGES
Configures the language simplification policy, see SIMPLIFY_LANGUAGES.

WEBLATE_DEFAULT_ACCESS_CONTROL
Configures the default for new projects, see DEFAULT_ACCESS_CONTROL.

WEBLATE_DEFAULT_RESTRICTED_COMPONENT
Configures the default value for for new components, see DEFAULT_RESTRICTED_COMPONENT.

WEBLATE_DEFAULT_TRANSLATION_PROPAGATION
Configures the default value for for new components, see DEFAULT_TRANSLATION_PROPAGATION.

WEBLATE_DEFAULT_COMMITER_EMAIL
Default_COMMITER_EMAIL.

WEBLATE_DEFAULT_COMMITER_NAME
Default_COMMITER_NAME.

WEBLATE_DEFAULT_SHARED_TM
Default_SHARED_TM

WEBLATE_AKISMET_API_KEY
AKISMET_API_KEY.

WEBLATE_GPG_IDENTITY
Configures GPG signing of commits, see WEBLATE_GPG_IDENTITY.

WEBLATE_URL_PREFIX
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.

WEBLATE_LICENSE_FILTER
LICENSE_FILTER.

WEBLATE_LICENSE_REQUIRED
LICENSE_REQUIRED.

WEBLATE_WEBSITE_REQUIRED
WEBSITE_REQUIRED.

WEBLATE_HIDE_VERSION
HIDE_VERSION.
You can set configuration for any rate limiter scopes. To do that add `WEBLATE_` prefix to any of setting described in

```
WEBLATE_RATELIMIT_ATTEMPTS
WEBLATE_RATELIMIT_WINDOW
WEBLATE_RATELIMIT_LOCKOUT
```

```
WEBLATE_API_RATELIMIT_ANON
WEBLATE_API_RATELIMIT_USER
```

API:

```
WEBLATE_ENABLE_AVATARS
WEBLATE_LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH
WEBLATE_SSH_EXTRA_ARGS
WEBLATE_BORG_EXTRA_ARGS
```

Machine translation settings

```
WEBLATE_MT_APERTIUM_API
Enables Apertium machine translation and sets `MT_APERTIUM_API`
WEBLATE_MT_AWS_REGION
WEBLATE_MT_AWS_ACCESS_KEY_ID
WEBLATE_MT_AWS_SECRET_ACCESS_KEY
```

Environment:

```
environment:
    WEBLATE_MT_AWS_REGION: us-east-1
    WEBLATE_MT_AWS_ACCESS_KEY_ID: AKIASOFODNN7EXAMPLE
    WEBLATE_MT_AWS_SECRET_ACCESS_KEY: wJalrXUtnFEMI/K7MDENG/bPxRfiCyEYQYQ
```

132
WEBLATE_MT_DEEPL_KEY
Enables DeepL machine translation and sets MT_DEEPL_KEY

WEBLATE_MT_DEEPL_API_URL
DeepL API URL: MT_DEEPL_API_URL

WEBLATE_MT_LIBRETRANSLATE_KEY
Enables LibreTranslate machine translation and sets MT_LIBRETRANSLATE_KEY

WEBLATE_MT_LIBRETRANSLATE_API_URL
LibreTranslate API URL: MT_LIBRETRANSLATE_API_URL

WEBLATE_MT_GOOGLE_KEY
Enables Google Translate and sets MT_GOOGLE_KEY

WEBLATE_MT_GOOGLE_CREDENTIALS
Enables Google Translate API V3 (Advanced) and sets MT_GOOGLE_CREDENTIALS

WEBLATE_MT_GOOGLE_PROJECT
Enables Google Translate API V3 (Advanced) and sets MT_GOOGLE_PROJECT

WEBLATE_MT_GOOGLE_LOCATION
Enables Google Translate API V3 (Advanced) and sets MT_GOOGLE_LOCATION

WEBLATE_MT_MICROSOFT_COGNITIVE_KEY
Enables Microsoft Cognitive Services Translator and sets MT_MICROSOFT_COGNITIVE_KEY

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

WEBLATE_MT_MICROSOFT_REGION
Sets MT_MICROSOFT_REGION

WEBLATE_MT_MICROSOFT_BASE_URL
Sets MT_MICROSOFT_BASE_URL

WEBLATE_MT_MODERNMT_KEY
Enables ModernMT and sets MT_MODERNMT_KEY.

WEBLATE_MT_MYMEMORY_ENABLED
Enables MyMemory machine translation and sets MT_MYMEMORY_EMAIL to WEBLATE_ADMIN_EMAIL.

WEBLATE_MT_GLOSBE_ENABLED
Glosbe translator

WEBLATE_MT_MICROSOFT_TERMINOLOGY_ENABLED
Microsoft Terminology Service

WEBLATE_MT_SAP_BASE_URL
WEBLATE_MT_SAP_SANDBOX_APIKEY
WEBLATE_MT_SAP_USERNAME
WEBLATE_MT_SAP_PASSWORD
WEBLATE_MT_SAP_USE_MT
Configures SAP Translation Hub machine translation.

WEBLATE_MT_GLOSBE_ENABLED:
WEBLATE_MT_MYMEMORY_ENABLED: 1
WEBLATE_MT_MICROSOFT_TERMINOLOGY_ENABLED: 1
WEBLATE_MT_SAP_BASE_URL: "https://example.hana.ondemand.com/translationhub/api/v1/"
WEBLATE_MT_SAP_USERNAME: "user"
WEBLATE_MT_SAP_PASSWORD: "password"
WEBLATE_MT_SAP_USE_MT: 1
Authentication settings

**LDAP**

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Example Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>WEBLATE_AUTH_LDAP_SERVER_URI</td>
<td>ldap://ldap.example.org</td>
</tr>
<tr>
<td>WEBLATE_AUTH_LDAP_USER_DN_TEMPLATE</td>
<td>uid=%(user)s,ou=People,dc=example,dc=net</td>
</tr>
<tr>
<td>WEBLATE_AUTH_LDAP_BIND_DN</td>
<td>CN=ldap,CN=Users,DC=example,DC=com</td>
</tr>
<tr>
<td>WEBLATE_AUTH_LDAP_BIND_PASSWORD</td>
<td>password</td>
</tr>
<tr>
<td>WEBLATE_AUTH_LDAP_USER_ATTR_MAP</td>
<td>full_name:name,email:mail</td>
</tr>
<tr>
<td>WEBLATE_AUTH_LDAP_USER_SEARCH</td>
<td>CN=Users,DC=example,DC=com</td>
</tr>
<tr>
<td>WEBLATE_AUTH_LDAP_USER_SEARCH_UNION</td>
<td>ou=users,dc=example,dc=com</td>
</tr>
<tr>
<td>WEBLATE_AUTH_LDAP_USER_SEARCH_UNION_DELIMITER</td>
<td></td>
</tr>
<tr>
<td>WEBLATE_AUTH_LDAP_CONNECTION_OPTION_REFERRALS</td>
<td>0</td>
</tr>
<tr>
<td>WEBLATE_AUTH_LDAP_USER_SEARCH_FILTER</td>
<td>(sAMAccountName=%(user)s)</td>
</tr>
</tbody>
</table>

Example for direct bind:

```ini
[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
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
```

Example for search and bind:

```ini
[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
WEBLATE_AUTH_LDAP_USER_SEARCH_FILTER: (sAMAccountName=%(user)s)
```

Example for union search and bind:

```ini
[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
WEBLATE_AUTH_LDAP_USER_SEARCH_UNION: ou=users,dc=example,dc=com
```

Example with search and bind against Active Directory:

```ini
[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)
```
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

Bitbucket

WEBLATE_SOCIAL_AUTH_BITBUCKET_KEY
WEBLATE_SOCIAL_AUTH_BITBUCKET_SECRET

Facebook

WEBLATE_SOCIAL_AUTH_FACEBOOK_KEY
WEBLATE_SOCIAL_AUTH_FACEBOOK_SECRET

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

GitLab

WEBLATE_SOCIAL_AUTH_gitlab_KEY
WEBLATE_SOCIAL_AUTH_gitlab_SECRET
WEBLATE_SOCIAL_AUTH_gitlab_API_URL

Azure Active Directory

WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_KEY
WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_SECRET
Enables Azure Active Directory authentication, see Microsoft Azure Active Directory.
Azure Active Directory with Tenant support

WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_KEY
WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_SECRET
WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_TENANT_ID
Enables Azure Active Directory authentication with Tenant support, see Microsoft Azure Active Directory.

Keycloak

WEBLATE_SOCIAL_AUTH_KEYCLOAK_KEY
WEBLATE_SOCIAL_AUTH_KEYCLOAK_SECRET
WEBLATE_SOCIAL_AUTH_KEYCLOAK_PUBLIC_KEY
WEBLATE_SOCIAL_AUTH_KEYCLOAK_ALGORITHM
WEBLATE_SOCIAL_AUTH_KEYCLOAK_AUTHORIZATION_URL
WEBLATE_SOCIAL_AUTH_KEYCLOAK_ACCESS_TOKEN_URL
Enables Keycloak authentication, see documentation.

Linux vendors

You can enable authentication using Linux vendors authentication services by setting following variables to any value.

WEBLATE_SOCIAL_AUTH_FEDORA
WEBLATE_SOCIAL_AUTH_OPENSUSE
WEBLATE_SOCIAL_AUTH_UBUNTU

Slack

WEBLATE_SOCIAL_AUTH_SLACK_KEY
SOCIAL_AUTH_SLACK_SECRET
Enables Slack authentication, see Slack.

SAML

Self-signed SAML keys are automatically generated on first container startup. In case you want to use own keys, place the certificate and private key in /app/data/ssl/saml.crt and /app/data/ssl/saml.key.

WEBLATE_SAML_IDP_ENTITY_ID
WEBLATE_SAML_IDP_URL
WEBLATE_SAML_IDP_X509CERT
SAML Identity Provider settings, see SAML.

Other authentication settings

WEBLATE_NO_EMAIL_AUTH
Disables e-mail authentication when set to any value. See documentation.
**PostgreSQL database setup**

`docker-compose.yml` Weblate PostgreSQL

- **POSTGRES_PASSWORD**
  PostgreSQL password.

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

- **POSTGRES_USER**
  PostgreSQL username.

- **POSTGRES_DATABASE**
  PostgreSQL database name.

- **POSTGRES_HOST**
  PostgreSQL server hostname or IP address. Defaults to database.

- **POSTGRES_PORT**
  PostgreSQL server port. Defaults to none (uses the default value).

- **POSTGRES_SSL_MODE**
  Configure how PostgreSQL handles SSL in connection to the server, for possible choices see [SSL Mode Descriptions](#).

- **POSTGRES_ALTER_ROLE**
  Configures name of role to alter during migrations, see Weblate PostgreSQL [](#).

- **POSTGRES_CONN_MAX_AGE**
  4.8.1
  The lifetime of a database connection, as an integer of seconds. Use 0 to close database connections at the end of each request (this is the default behavior).

  Enabling connection persistence will typically cause more open connections to the database. Please adjust your database configuration prior enabling.

```yaml
environment:
  POSTGRES_CONN_MAX_AGE: 3600
```

- **CONN_MAX_AGE**, Persistent connections

- **POSTGRES_DISABLE_SERVER_SIDE_CURSORS**
  4.9.1
  Disable server side cursors in the database. This is necessary in some pgbouncer setups.

```yaml
environment:
  POSTGRES_DISABLE_SERVER_SIDE_CURSORS: 1
```

- **DISABLE_SERVER_SIDE_CURSORS**, Transaction pooling and server-side cursors
Database backup settings

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.

- `REDIS_HOST`
The Redis server hostname or IP address. Defaults to `cache`.
- `REDIS_PORT`
The Redis server port. Defaults to `6379`.
- `REDIS_DB`
The Redis database number, defaults to `1`.
- `REDIS_PASSWORD`
The Redis server password, not used by default.
- `REDIS_TLS`
Enables using SSL for Redis connection.
- `REDIS_VERIFY_SSL`
Can be used to disable SSL certificate verification for Redis connection.

Email server setup

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

Example TLS configuration:

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

Example SSL configuration:

```
environment:
  WEBLATE_EMAIL_HOST: smtp.example.com
  WEBLATE_EMAIL_PORT: 465
  WEBLATE_EMAIL_HOST_USER: user
  WEBLATE_EMAIL_HOST_PASSWORD: pass
  WEBLATE_EMAIL_USE_TLS: 0
  WEBLATE_EMAIL_USE_SSL: 1
```

- `WEBLATE_EMAIL_HOST`
Mail server hostname or IP address.
- `WEBLATE_EMAIL_PORT`
Mail server port, defaults to `25`.
- `WEBLATE_EMAIL_USE_SSL`
- `WEBLATE_EMAIL_USE_TLS`
WEBLATE_EMAIL_HOST_USER
EMAIL_HOST_USER

WEBLATE_EMAIL_HOST_PASSWORD
EMAIL_HOST_PASSWORD

WEBLATE_EMAIL_HOST_PASSWORD_FILE

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.

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.

WEBLATE_EMAIL_PORT WEBLATE_EMAIL_USE_SSL EMAIL_USE_TLS

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

WEBLATE_AUTO_UPDATE
Web site

AUTO_UPDATE

This is a Boolean setting (use "true" or "false").

WEBLATE_GET_HELP_URL
GET_HELP_URL

WEBLATE_STATUS_URL
STATUS_URL

WEBLATE_LEGAL_URL
LEGAL_URL

WEBLATE_PRIVACY_URL
PRIVACY_URL
Error reporting

It is recommended to collect errors from the installation systematically, see [Rollbar](#). To enable support for Rollbar, set the following:

**ROLLBAR_KEY**
Your Rollbar post server access token.

**ROLLBAR_ENVIRONMENT**
Your Rollbar environment, defaults to `production`.

To enable support for Sentry, set following:

**SENTRY_DSN**
Your Sentry DSN.

**SENTRY_ENVIRONMENT**
Your Sentry Environment (optional).

**CDN**

**WEBLATE_LOCALIZE_CDN_URL**
**WEBLATE_LOCALIZE_CDN_PATH**

Configuration for JavaScript 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
```

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

```
weblate-cdn
```

**WEBLATE_LOCALIZE_CDN_URL**
**WEBLATE_LOCALIZE_CDN_PATH**

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**

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

**CHECK_LIST**
**AUTOFIX_LIST**
**WEBLATE_ADDONS**
**INSTALLED_APPS**
WEBLATE WORKERS

4.6.1

Base number of worker processes running in the container. When not set it is determined automatically on container startup based on number of CPU cores available.

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 (\texttt{--concurrency 16}) or use different pool implementation (\texttt{--pool=gevent}).

By default, the number of concurrent workers is based on \texttt{WEBLATE WORKERS}.

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

WEB WORKERS

Configure how many uWSGI workers should be executed.

It defaults to \texttt{WEBLATE WORKERS}.

```
environment:
  WEB_WORKERS: 32
```

WEBLATE SERVICE

Defines which services should be executed inside the container. Use this for \textit{Scaling horizontally}.

```
WEBLATE SERVICE:
  celery-beat
  Celery task scheduler, only one instance should be running. This container is also responsible for the database structure migrations and it should be started prior others.
  celery-backup
  Celery worker for backups, only one instance should be running.
  celery-celery
  Generic Celery worker.
  celery-memory
  Celery worker.
  celery-notify
  Celery worker.
  celery-translate
  Celery worker.
  web
  Web worker.
```

141
Docker container volumes

There are two volumes (data and cache) exported by the Weblate container. The other service containers (PostgreSQL or Redis) have their data volumes as well, but those are not covered by this document.

The data volume is used to store Weblate persistent data such as cloned repositories or to customize Weblate installation.

The placement of the Docker volume on host system depends on your Docker configuration, but usually it is stored in /var/lib/docker/volumes/weblate-docker_weblate-data/_data/ (the path consist of name of your docker-compose directory, container, and volume names). In the container it is mounted as /app/data.

The cache volume is mounted as /app/cache and is used to store static files. Its content is recreated on container startup and the volume can be mounted using ephemeral filesystem such as tmpfs.

When creating the volumes manually, the directories should be owned by UID 1000 as that is user used inside the container.

Further configuration customization

You can further customize Weblate installation in the data volume, see Docker container volumes.

Custom configuration files

You can additionally override the configuration in /app/data/settings-override.py (see Docker container volumes). This is executed at the end of built-in settings, after all environment settings are loaded, and you can adjust or override them.

Replacing logo and other static files

The static files coming with Weblate can be overridden by placing into /app/data/python/customize/static (see Docker container volumes). For example creating /app/data/python/customize/static/favicon.ico will replace the favicon.

This approach can be also used to override Weblate templates. For example .docx documents can be placed into /app/data/python/customize/templates/legal/documents.

Alternatively you can also include own module (see Customizing Weblate) and add it as separate volume to the Docker container, for example:

```yaml
weblate:
  volumes:
    - weblate-data:/app/data
    - ./weblate_customization/weblate_customization:/app/data/python/
  environment:
    WEBLATE_ADD_APPS: weblate_customization
```
**Adding own Python modules**

You can place own Python modules in `/app/data/python/` (see Docker container volumes) and they can be then loaded by Weblate, most likely by using Custom configuration files.

### Customizing Weblate

#### Debian Ubuntu

Weblate

- 2 GB RAM
- 2 CPU
- 1 GB HDD or SSD

Weblate

- CPU 4 GB RAM
- 100-300 MB

### Python

#### Debian Ubuntu

```
apt install libxml2-dev libxslt-dev libfreetype6-dev libjpeg-dev libz-dev libyaml-dev
libffi-dev libcairo-dev gir1.2-pango-1.0 libgirepository1.0-dev libacl1-dev libssl-dev libpq-dev libjpeg62-turbo-dev build-essential python3-gdbm python3-dev python3-pip python3-virtualenv virtualenv git
```

```
apt install tesseract-ocr libtesseract-dev libleptonica-dev
apt install libldap2-dev libldap-common libpq-dev libpq-dev libjpeg62-turbo-dev build-essential python3-gdbm python3-dev python3-pip python3-virtualenv virtualenv git
```

#### Ubuntu

```
# Web server option 1: NGINX and uWSGI
apt install nginx uwsgi uwsgi-plugin-python3

# Web server option 2: Apache with `mod_wsgi`
apt install apache2 libapache2-mod-wsgi-py3

# Caching backend: Redis
apt install redis-server

# Database server: PostgreSQL
apt install postgresql postgresql-contrib
```
# SMTP server
apt install exim4

## Python

1. Webate virtualenv:
   ```
   virtualenv --python=python3 ~/weblate-env
   ```

2. Webate virtualenv:
   ```
   . ~/weblate-env/bin/activate
   ```

3. Webate virtualenv:
   ```
   pip install "Weblate[all]"
   ```

## Weblate

1. Webate Linux libffi:
   ```
   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...)
   ```

   PyPI:
   ```
   pip install --force-reinstall --no-binary :all: cffi
   ```

## Weblate

1. ~/weblate-env/lib/python3.7/site-packages/weblate/settings_example.py
2. settings.py
3. Weblate PostgreSQL
4. weblate migrate
5. weblate createadmin
6. weblate collectstatic
7. weblate compress
8. JavaScript CSS
9. Celery

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

weblate runserver

Weblate
http://localhost:8000/

Weblate virtualenv
`weblate`

Ctrl+C

/manage/performance/

URL:

weblate check --deploy

SUSE

VCS

SUSE

openSUSE

2 GB RAM
2 CPU
1 GB HDD or SSD

4 GB RAM

100 MB 300 MB

145
Python:

```
zypper install \n   libxslt-devel libxm2-devel freetype-devel libjpeg-devel zlib-devel \n   libyaml-devel libffi-devel cairo-devel pango-devel \n   gobject-introspection-devel libacl-devel python3-pip python3-virtualenv \n   python3-devel git
```

```
zypper install tesseract-ocr tesseract-devel leptonica-devel
zypper install libldap2-devel libtasl2-devel
zypper install libxmlsec1-devel
```

Weblate

```
# 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:

```
# Weblate virtualenv: 
virtualenv --python=python3 ~/weblate-env

2. Weblate virtualenv:
. ~/weblate-env/bin/activate

3. Weblate: 
   pip install "Weblate[all]"
```

PyPI:

```
ffi_prep_closure(): bad user_data (it seems that the version of the libffi_`
```
pip install --force-reinstall --no-binary :all: cffi

**Weblate**

```bash
- ./weblate-env/bin/activate
weblate
```

1. ```~/weblate-env/lib/python3.7/site-packages/weblate/settings_example.py```  
2. ```settings.py```  
3. ```weblate migrate```  
4. ```weblate createadmin```  
5. ```weblate migrate```  
6. ```weblate collectstatic```  
7. ```weblate compress```  
8. ```weblate runserver```  

**Web ref:**

```
http://localhost:8000/
```

**VCS**

```
http://localhost:8000/create/project/  Project
```

**Component**

```
http://localhost:8000/manage/performance/  --deploy
```
RedHat® Fedora® CentOS

**Weblate**

- 2 GB RAM
- 2 CPU
- 1 GB HDD or SSD

**Python**

```
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
```

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

dnf install libldap2-devel libssasl2-devel

dnf install libxmlsec1-devel
```

**Weblate**

```
# 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

1. Weblate virtualenv:
   
   ```bash
   virtualenv --python=python3 ~/weblate-env
   ```

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

3. Weblate virtualenv:
   
   ```bash
   pip install "Weblate[all]"
   ```

   **Note:** Weblate Linux libffi:
   ```bash
   ffi_prep_closure(): bad user_data (it seems that the version of the libffi... compile-time)
   ```

4. Weblate PyPI cffi:
   
   ```bash
   pip install --force-reinstall --no-binary :all: cffi
   ```

### Weblate

1. ~/weblate-env/lib/python3.7/site-packages/weblate/settings_example.py
   
   ```bash
   ~/weblate-env/lib/python3.7/site-packages/weblate/settings.py
   ```

2. Weblate PostgreSQL:
   
   ```bash
   weblate migrate
   ```

3. Weblate Django:
   
   ```bash
   weblate createadmin
   ```

4. Weblate static-files:
   
   ```bash
   weblate collectstatic
   ```

5. Weblate ref: server ref: static-files:
   
   ```bash
   ```

6. Weblate JavaScript CSS:
   
   ```bash
   weblate compress
   ```

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

8. Weblate ref: server ref: static-files:
   
   ```bash
   ```
weblate runserver

1. http://localhost:8000/create/project/ Project configuration
2. Web Component configuration
3. macOS

Web 2 GB RAM
2 CPU
1 GB HDD or SSD

Weblate CPU 4 GB RAM 100 300 MB

Python:

brew install python pango cairo gobject-introspection libffi glib libyaml
pip3 install virtualenv

export PKG_CONFIG_PATH="/usr/local/opt/libffi/lib/pkgconfig"
# 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

1. Weblate virtualenv:

   virtualenv --python=python3 ~/weblate-env

2. Weblate virtualenv (Activate):

   . ~/weblate-env/bin/activate

3. Weblate virtualenv (Install):

   pip install "Weblate[all]"

### libffi

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)

PyPI:

   pip install --force-reinstall --no-binary :all: cffi

## Weblate

1. Weblate virtualenv (Activate):

   . ~/weblate-env/bin/activate

2. Weblate settings.py:

   ~/weblate-env/lib/python3.7/site-packages/weblate/settings_example.py

3. Weblate PostgreSQL:

   ~/weblate-env/lib/python3.7/site-packages/weblate/settings.py
weblate migrate

weblate createadmin

5. Web ref: server ref: static-files:

weblate collectstatic

6. JavaScript CSS:

weblate compress

7. Celery:

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

8. Weblate runserver

http://localhost:8000/

Weblate virtualenv:

weblate check --deploy


Weblate virtualenv:

Ctrl+C

2. VCS Component configuration:

Debian Ubuntu SUSE openSUSE RedHat Fedora CentOS

2. Git Weblate tarball:

git clone https://github.com/WeblateOrg/weblate.git weblate-src

3. Weblate virtualenv:
. ~/weblate-env/bin/activate
pip install -e weblate-src

4. weblate/settings_example.py weblate/settings.py
5. weblate/settings.py Django
6. Weblate Weblate
7. Django

weblate migrate
weblate collectstatic
weblate compress

OpenShift

OpenShift Weblate Weblate PostgreSQL PERSISTENT VOLUME CLAIM PVC
https://github.com/WeblateOrg/openshift/

Web

template.yml

CLI

   -n <PROJECT>

   -n <PROJECT> weblate

# If the template is already uploaded
$ oc process --parameters -n <PROJECT> weblate
  -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/ \
  -p REDIS_IMAGE=docker-registry.default.svc:5000/openshift/redis:3.2 \
  | oc create -f

Weblate ConfigMap WebLATE_SITE_DOMAIN ConfigMap
WEBLATE_ADMIN_PASSWORD
Secret

$ oc delete all -l app=<APPLICATION_NAME>
$ oc delete configmap -l app=<APPLICATION_NAME>
$ oc delete secret -l app=<APPLICATION_NAME>
    # ATTENTION! The following command is only optional and will permanently delete all of your data.
$ oc delete pvc -l app=<APPLICATION_NAME>
$ oc delete all -l app=weblate \
    && oc delete secret -l app=weblate \
    && oc delete configmap -l app=weblate \
    && oc delete pvc -l app=weblate

Kubernetes

Kubernetes Helm Weblate
PostgreSQL PERSISTENT VOLUME CLAIM

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

Docker environment variables:

Docker:

Virtualenv:

Debian Ubuntu
SUSE openSUSE
RedHat Fedora CentOS
macOS

OpenShift
Kubernetes

Weblate Linux FreeBSD macOS Unix
Weblate Windows

Weblate:

PostgreSQL
Redis
Celery
SMTP

Python:

Python 3.6

requirements.txt

https://www.djangoproject.com/
https://docs.celeryproject.org/
https://toolkit.translatehouse.org/
https://github.com/WeblateOrg/translation-finder
https://python-social-auth.readthedocs.io/
https://www.djangoproject.com/
Weblate
requirements-optional.txt

https://www.mercurial-scm.org/
https://github.com/viraptor/phply
https://github.com/sirfz/tesserocr
https://github.com/Nekmo/python-akismet
https://pypi.org/project/ruamel.yaml/
https://docs.python-zeep.org/
https://pypi.org/project/aeidon/
https://projectfluent.org/

```bash
pip install "Weblate[PHP,Fluent]"
```

```bash
pip install "Weblate[all]"
```

```bash
pip install Weblate
```

Webilate PostgreSQL MySQL MariaDB

https://git-scm.com/
https://cairographics.org/ Pango Cairo
https://pypi.org/project/git-review/
https://git-scm.com/docs/git-svn
https://github.com/tesseract-ocr/tesseract
https://github.com/licensee/licensee

Python

```
pip install wheels
```

156
Weblate: Michal Čihař PGP ID:

63CB 1DF1 EF2A C0EE 5A32 9C27 B313 42B7 511D

[https://keybase.io/nijel]

PGP .asc:

```bash
$ gpg --verify Weblate-3.5.tar.xz.asc
```

```text
gpg: assuming signed data in 'Weblate-3.5.tar.xz'
gpg: Signature made Ne 3. března 2019, 16:43:15 CET
```

```text
Can't check signature: public key not found
```

GPG:

```bash
$ gpg --auto-key-locate wkd --locate-keys michal@cihar.com
```

```text
pub rsa4096 2009-06-17 [SC]
  63CB1DF1EF2AC0EE5A329C27B31342B7511D
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]
```

Michal Čihař PGP ID:

```bash
$ gpg --import wmxth3chu9jfxdxywj1skpmhsj311mzm
```

```text
1 :
```

```bash
$ gpg --keyserver hkp://pgp.mit.edu --recv-keys
```

```text
87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
```

```text
Michal Čihař <michal@cihar.com>" imported
gpg: Total number processed: 1
gpg: unchanged: 1
```

GPG:

```bash
$ gpg --verify Weblate-3.5.tar.xz.asc
```

```text
gpg: assuming signed data in 'Weblate-3.5.tar.xz'
gpg: Signature made Ne 3. března 2019, 16:43:15 CET
```

```text
using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
```

```text
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>
```

```text
WARNING: This key is not certified with a trusted signature!
gpg: There is no indication that the signature belongs to the...
```

Primary key fingerprint: 63CB 1DF1 EF2A C0EE 5A32 9C27 B313 42B7 511D
Validating other keys on your public keyring

```
$ gpg --verify Weblate-3.5.tar.xz.asc
gpg: assuming signed data in 'Weblate-3.5.tar.xz'
gpg: Signature made Sun Mar 3 16:43:15 2019 CET
gpg: using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: Good signature from "Michal Čihař <michal@cihar.com>" [ultimate]
gpg: aka "Michal Čihař <nijel@debian.org>" [ultimate]
gpg: aka "[jpeg image of size 8848]" [ultimate]
gpg: aka "Michal Čihař (Braiins) <michal.cihar@braiins.cz>
```

```
$ 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]
```

### Weblate

<table>
<thead>
<tr>
<th>Weblate</th>
<th>WSGI</th>
<th>Celery</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weblate</td>
<td>Celery</td>
<td>Celery</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Docker</th>
<th>/app/data</th>
<th>weblate</th>
<th>UID</th>
<th>1000</th>
</tr>
</thead>
</table>

### PostgreSQL

<table>
<thead>
<tr>
<th>PostgreSQL</th>
<th>Webate</th>
<th></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>PostgreSQL contrib</th>
<th></th>
</tr>
</thead>
</table>

### PostgreSQL notes
**PostgreSQL**

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
```

Weblate PostgreSQL

```python
CREATE EXTENSION IF NOT EXISTS pg_trgm WITH SCHEMA weblate;
```

**Weblate PostgreSQL**

settings.py

```python
DATABASES = {
    "default": {
        # Database engine
        "ENGINE": "django.db.backends.postgresql",
        # Database name
        "NAME": "weblate",
        # Database user
        "USER": "weblate",
        # Name of role to alter to set parameters in PostgreSQL, # use in case role name is different than user used for authentication.
        "ALTER_ROLE": "weblate",
        # Database password
        "PASSWORD": "password",
        # Set to empty string for localhost
        "HOST": "database.example.com",
        # Set to empty string for default
        "PORT": ",",
    }
}
```

Weblate PostgreSQL

Azure Database for PostgreSQL

```psycopg2.errors.UndefinedObject: role "weblate@hostname" does not exist.
```

**MySQL**

Weblate MySQL 5.7.8

MariaDB 10.2.7

```python
errors.OperandTypeMismatch: Cannot assign Unicode data to InnoDB table column.
```

**MySQL**

MariaDB

```python
utf8mb4_unicode_ci
```

unreadable

READ COMMITTED

page 159
SQL

MySQL 8.x

MariaDB 10.5.x

/etc/my.cnf.d/server.cnf

8 GB RAM

MySQL/MariaDB

Weblate

innodb_file_per_table

MySQL/MariaDB

settings.py

DATABASES = {
    "default": {
        # Database engine
        "ENGINE": "django.db.backends.mysql",
        # Database name
        "NAME": "weblate",
        # Database user
        "USER": "weblate",
        # Database password
        "PASSWORD": "password",
        # Set to empty string for localhost
        "HOST": "127.0.0.1",
        # Set to empty string for default
        "PORT": "3306",
        # In case you wish to use additional
        # connection options
        "OPTIONS": {},
    }
}

GRANT ALL ON weblate.* to 'weblate'@'localhost' IDENTIFIED BY 'password';

FLUSH PRIVILEGES;
Not receiving e-mails from Weblate

Configuring outgoing e-mail in Docker container

Weblate  SMTP

EMAIL_HOST
EMAIL_HOST_PASSWORD
EMAIL_USE_TLS
EMAIL_USE_SSL
EMAIL_HOST_USER
EMAIL_PORT

Django
SMTP AUTH extension not supported by server

Not receiving e-mails from Weblate

Weblate  WSGI
REMOTE_ADDR

HTTP

IP
IP_BEHIND_REVERSE_PROXY
IP_PROXY_HEADER
IP_PROXY_OFFSET

IP_PROXY_HEADER
IP_BEHIND_REVERSE_PROXY
IP_PROXY_OFFSET
SECURE_PROXY_SSL_HEADER

HTTP

Weblate  VCS

settings.py

import os

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

ADMINS

ALLOWED_HOSTS

ALLOWED_HOSTS = ["demo.weblate.org"]
ALLOWED_HOSTS = [
    "*
]

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

DATABASES =
    Weblate

DEBUG

DEFAULT_FROM_EMAIL

SECRET_KEY

SERVER_EMAIL

weblate migrate
weblate migrate --noinput
createadmin
admin

weblate migrate

162
Django:

```bash
weblate check --deploy
```

Django DEBUG:

```python
DEBUG = False
```

Django ADMINS:

```python
ADMINS = ("Your Name", "your_email@example.com"),
```

Django HTTPS SITE_DOMAIN WEBLATE_SITE_DOMAIN ENABLE_HTTPS:

```python
HTTPS = SITE_DOMAIN, WEBLATE_SITE_DOMAIN, ENABLE_HTTPS
```

---

163
HTTPS

Enable HTTPS: True

SECURE_HSTS_SECONDS

SECURE_HSTS_SECONDS = 0

CACHES

CACHES = {
    "default": {
        "BACKEND": "django_redis.cache.RedisCache",
        "LOCATION": "redis://127.0.0.1:6379/0",
        "OPTIONS": {
            "CLIENT_CLASS": "django_redis.client.DefaultClient",
            "PARSER_CLASS": "redis.connection.HiredisParser",
        },
    },
}

Celery

Celery is a task queue that scales.
Django's cache framework

Django Weblate

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,
      },
    },
}

ENABLE_AVATARS
AVATAR_URL_PREFIX

Weblate

SERVER_EMAIL
DEFAULT_FROM_EMAIL

SERVER_EMAIL = "admin@example.org"
DEFAULT_FROM_EMAIL = "weblate@example.org"

Weblade EMAIL_BACKEND
EMAIL_BACKEND = django.core.mail.backends.dummy.EmailBackend

HTTP
Invalid HTTP_HOST header: '1.1.1.1'. You may need to add '1.1.1.1' to ALLOWED_HOSTS.

Docker

ALLOWED_HOSTS
SECRET_KEY

2.1: Weblate

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

TEMPLATES = [
    {
        "BACKEND": "django.template.backends.django.DjangoTemplates",
        "DIRS": [os.path.join(BASE_DIR, "templates"), ]
        "OPTIONS": {
            "context_processors": ["django.contrib.auth.context_processors.auth",
                                   "django.template.context_processors.debug",
                                   "django.template.context_processors.i18n",
                                   "django.template.context_processors.request",
                                   "django.template.context_processors.csrf",
                                   "django.contrib.messages.context_processors.messages",
                                   "weblate.trans.context_processors.weblate_context",
        ]
        "loaders": ["django.template.loaders.cached.Loader",
                   "django.template.loaders.filesystem.Loader",
                   "django.template.loaders.app_directories.Loader"
                   ]
    },
]
Celery

Lazy commits: commit_pending

AUTO_UPDATE

JSON: dump_memory

cleanuptrans

3.2

Celery

UTF-8

LANG="C.UTF-8"

Ubuntu

/etc/apache2/envvars

export LANG='en_US.UTF-8'
export LC_ALL='en_US.UTF-8'

CentOS

/etc/sysconfig/httpd

LANG='en_US.UTF-8'

Debian

/local/share/ca-certificates

update-ca-certificates

CA

requests.py

import os
os.environ["REQUESTS_CA_BUNDLE"] = "/etc/ssl/certs/ca-certificates.crt"

weblate compress
Common Deployment Scenarios

Web

Weblate

Docker

Weblate

SSL

WSGI

NGINX

uWSGI

Celery

uwsgi

DATA_DIR

Celery

Weblate

Django

uWSGI

fcgi

uwsgi

Web servers

Django

Web

weblate runserver

Django

Weblate Django uWSGI fcgi Django

NAS

NGINX

uWSGI

Apache

Gunicorn

uwsgi

Celery

uwsgi

DATA_DIR

Celery

Docker

Weblate Django uWSGI fcgi Django

weblate runserver

Django

Web

weblate runserver

Django

Web
2.4 Weblate & Django

Django

```
weblate collectstatic --noinput STATIC_ROOT /static
```

Web

```
STATIC_ROOT /static
DATA_DIR /home/weblate/data
```

NGINX uWSGI Apache Gunicorn

How to deploy Django

How to deploy static files

Web

```
webate.middleware.SecurityMiddleware
Content-Security-Policy HTTP X-XSS-Protection

CSP_SCRIPT_SRC
CSP_IMG_SRC
CSP_CONNECT_SRC
CSP_STYLE_SRC
CSP_FONT_SRC
```

NGINX uWSGI

```
# This example assumes Weblate is installed in virtualenv in /home/weblate/
# and DATA_DIR is set to /home/weblate/data, please adjust paths to match...
# your setup.

server {
    listen 80;
    server_name weblate;
    # Not used
    root /var/www/html;

    location ~ ^/favicon.ico$ {
        # DATA_DIR/static/favicon.ico
        alias /home/weblate/data/static/favicon.ico;
        expires 30d;
    }

    location /static/ {
        # DATA_DIR/static/
        alias /home/weblate/data/static/;
        expires 30d;
    }

    location /media/ {
        # DATA_DIR/media/
        alias /home/weblate/data/media/;
        expires 30d;
    }

    location / {
```
include uwsgi_params;
    # Needed for long running operations in admin interface
uwsgi_read_timeout 3600;
    # Adjust based to uwsgi configuration:
uwsgi_pass unix:///run/uwsgi/app/weblate/socket;
    # uwsgi_pass 127.0.0.1:8080;
}

uWSGI [uwsgi]

# This example assumes Weblate is installed in virtualenv in /home/weblate/
# and DATA_DIR is set to /home/weblate/data, please adjust paths to match
# your setup.
[uwsgi]
plugins = python3
master = true
protocol = uwsgi
socket = 127.0.0.1:8080
wsgi-file = /home/weblate/weblate-env/lib/python3.9/site-packages/
    /weblate/wsgi.py

# Add path to Weblate checkout if you did not install
# Weblate by pip
# python-path = /path/to/weblate

# In case you're using virtualenv uncomment this:
# virtualenv = /home/weblate/weblate-env

# Needed for OAuth/OpenID
buffer-size = 8192

# Reload when consuming too much of memory
reload-on-rss = 250

# Increase number of workers for heavily loaded sites
workers = 8

# Enable threads for Sentry error submission
enable-threads = true

# Child processes do not need file descriptors
close-on-exec = true

# Avoid default 0000 umask
umask = 0022

# Run as weblate user
uid = weblate
gid = weblate

# Enable harakiri mode (kill requests after some time)
# harakiri = 3600
# harakiri-verbose = true

# Enable uWSGI stats server
# stats = :1717
# stats-http = true

# Do not log some errors caused by client disconnects
ignore-sigpipe = true
ignore-write-errors = true
disable-write-exception = true
# VirtualHost for Weblate
#
# This example assumes Weblate is installed in virtualenv in /home/weblate/
# and DATA_DIR is set to /home/weblate/data, please adjust paths to match...
#
<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
  WSGIPassAuthorization On
  WSGIScriptAlias / /home/weblate/weblate-env/lib/python3.7/site-
  packages/weblate/wsgi.py process-group=weblate
  <Directory /home/weblate/weblate-env/lib/python3.7/site-packages/>
    <Files wsgi.py>
      Require all granted
    </Files>
  </Directory>
</VirtualHost>
# VirtualHost for Weblate using gunicorn on localhost:8000
# This example assumes Weblate is installed in virtualenv in /home/weblate/
# and DATA_DIR is set to /home/weblate/data, please adjust paths to match...
# your setup.
<VirtualHost *:80>
  ServerAdmin admin@weblate.example.org
  ServerName weblate.example.org

  # DATA_DIR/static/favicon.ico
  Alias /favicon.ico /home/weblate/data/static/favicon.ico

  # DATA_DIR/static/
  Alias /static/ /home/weblate/data/static/
  <Directory /home/weblate/data/static/>
    Require all granted
  </Directory>

  # DATA_DIR/media/
  Alias /media/ /home/weblate/data/media/
  <Directory /home/weblate/data/media/>
    Require all granted
  </Directory>

  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>

How to use Django with Gunicorn
ServerAdmin admin@weblate.example.org
ServerName weblate.example.org

# DATA_DIR/static/favicon.ico
Alias /weblate/favicon.ico /home/weblate/data/static/favicon.ico

# DATA_DIR/static/
Alias /weblate/static/ /home/weblate/data/static/
<Directory /home/weblate/data/static/>
    Require all granted
</Directory>

# DATA_DIR/media/
Alias /weblate/media/ /home/weblate/data/media/
<Directory /home/weblate/data/media/>
    Require all granted
</Directory>

# Path to your Weblate virtualenv
WSGIDaemonProcess weblate python-home=/home/weblate/weblate-env_
    user=weblate request-timeout=600
WSGIProcessGroup weblate
WSGIApplicationGroup %{GLOBAL}

WSGIScriptAlias /weblate /home/weblate/weblate-env/lib/python3.7/site-
    packages/weblate/wsgi.py process-group=weblate
WSGIPassAuthorization On

<Directory /home/weblate/weblate-env/lib/python3.7/site-packages/
    weblate/>
    <Files wsgi.py>
        Require all granted
    </Files>
</Directory>

</VirtualHost>

# weblate/settings.py

URL_PREFIX = "'/weblate"

## Celery

CELERY 3.2.4

Weblate ≥ Celery ≥ Celery ≥ Celery ≥ Celery

CELERY_TASK_ALWAYS_EAGER = False
CELERY_BROKER_URL = "redis://localhost:6379"
CELERY_RESULT_BACKEND = CELERY_BROKER_URL

Redis broker configuration in Celery
Celery WSGI

```bash
$ ./weblate/examples/celery start
$ ./weblate/examples/celery stop
```

Eager WSGI

```
eager wsgi Celery
```

Celery

```
[CELERY_TASK_ALWAYS_EAGER] = True
[CELERY_BROKER_URL] = "memory://"
[CELERY_TASK_EAGER_PROPAGATES] = True
```

Linux

```
/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} \n  --logfile=${CELERYD_LOG_FILE} --loglevel=${CELERYD_LOG_LEVEL} ${CELERYD_OPTS}"
ExecStop=/bin/sh -c '"${CELERY_BIN} multi stopwait ${CELERYD_NODES} \n  --pidfile=${CELERYD_PID_FILE}"'
ExecReload=/bin/sh -c '"${CELERY_BIN} multi restart ${CELERYD_NODES} \n  -A ${CELERY_APP} --pidfile=${CELERYD_PID_FILE} \n  --logfile=${CELERYD_LOG_FILE} --loglevel=${CELERYD_LOG_LEVEL} ${CELERYD_OPTS}"

[Install]
WantedBy=multi-user.target
```

```
# 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"
```

174
# Extra command-line arguments to the worker,
# increase concurrency if you get weblate.E019

# 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"

/etc/logrotate.d/celery

/var/log/celery/*.log {
  weekly
  missingok
  rotate 12
  compress
  notifempty
}

Celery beat

Weblate settings.py

Celery beat

Celery

Celery queues

celery_queues

celery:userguide/configuration\celery:userguide/workers\celery:userguide/daemonizing\celery:userguide/monitoring

Weblate

Weblate Kubernetes /healthz/ URL

Docker URL

GET /api/metrics/ API

`Munin <https://github.com/WeblateOrg/munin>`_
Weblate

**Sentry**

Weblate

@student

settings.py

```
SENTRY_DSN = "https://id@your.sentry.example.com/"
```

**Rollbar**

Weblate

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'",
}
```

Weblate

Weblate

Weblate

```
# Export current data
weblate dumpdata > /tmp/weblate.dump
# Import dump
weblate loaddata /tmp/weblate.dump
```
Weblate

Weblate:
- Docker
- OpenShift
- Kubernetes

Bitnami Weblate


Cloudron Weblate

Cloudron Web Weblate Cloudron Web Weblate package repo

Install with YunoHost

yunohost app install https://github.com/YunoHost-Apps/weblate_ynh
Weblate

Docker

Upgrading the Docker container

1. WSGI
   
   `pip install -U "Weblate[all]"`

2. Weblate
   
   `pip install -U Weblate`

3. Git
   
   `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`

4. Weblate
   
   `weblate migrate --noinput`

5. Weblate
   
   `weblate collectstatic --noinput --clear`

6. JavaScript, CSS
   
   `weblate compress`

7. Git
   
   `weblate compilemessages`

8. Weblate
   
   `weblate check --deploy`

9. Celery
   
   `Celery`

10. Weblate

---

178
4.2 4.3 API

INSTALLED_APPS

CHECK_LIST

4.3 API

INSTALLED_APPS

SESSION_COOKIE_AGE_AUTHENTICATED
SESSION_COOKIE_AGE

hub lab GitHub GitLab: :setting: `GITHUB_CREDENTIALS` `GITLAB_CREDENTIALS`

4.3.1 API:

Celery memory CELERY_TASK_ROUTES

4.3.2 API:

post_update skip_push

4.4 4.5 API

INSTALLED_APPS

weblate.metrics

4.5.1 API:

pyahocorasick

180
4.5 4.6 4.7 4.8 4.9 4.10

**WEBLATE_FORMATS**

**API**

- **Weblate URL**
  
  POST /api/projects/(string:project)/components/

**PASSWORD_HASHERS**

- Argon2

---

4.6 4.7 4.8 4.9 4.10

**settings_example.py**

```python
MIDDLEWARE:
    settings.MIDDLEWARE
    settings.ExampleMiddleware
```

**DeepL**

```python
MT_DEEPL_API_URL = 'https://api.deepl.com/v2/translate'
```

**Django 3.2**

---

4.7 4.8 4.9 4.10

---

4.8 4.9 4.10

---

4.9 4.10

---

**CSRF_TRUSTED_ORIGINS**

- Docker

---

181
4.10 4.11 4.11.1

Weblate Python 3.7

Weblate Web API

Weblate chardet charset-normalizer

Changed in 4.11.1: There is a change in REST_FRAMEWORK setting (removal of one of the backends in DEFAULT_AUTHENTICATION_CLASSES).

Python 2 Python 3

Weblate Python 3.6 Python 3

Weblate: Python 2 Python 3 — Web 3.11.1

PostgreSQL

PostgreSQL Weblate Weblate PostgreSQL Web Celery

PostgreSQL

Django JSON pgloader

Django JSON

pgloader

PostgreSQL

PostgreSQL settings.py

DATABASES = {
    "default": {
        "ENGINE": "django.db.backends.mysql",
        "NAME": "weblate",
        "USER": "weblate",
        "PASSWORD": "password",
        "HOST": "database.example.com",
        "PORT": ",
        "OPTIONS": {
            "TIME_ZONE": "UTC",
            "USE_TZ": true
        }
    }
}
In case of using an older MySQL server, which has MyISAM as a default storage:

```python
# '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:
"connect_timeout": 28800,
```

```json
"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": "",
}
```

**PostgreSQL**

1. PostgreSQL settings.py

2. PostgreSQL:

   ```bash
   weblate migrate --database=postgresql
   weblate sqlflush --database=postgresql | weblate dbshell --database=postgresql
   ```

3. PostgreSQL:

   ```bash
   weblate dumpdata --all --output weblate.json
   weblate loaddata weblate.json --database=postgresql
   ```

4. PostgreSQL: DATABASES

   ```bash
   pgloader
   ```

   1. PostgreSQL
   
   2. PostgreSQL:

      ```bash
      weblate migrate
      weblate sqlflush | weblate dbshell
      ```

   3. pgloader

      ```bash
      pgloader
      ```

      ```sql
      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';
      ```
Pootle

Weblate

BorgBackup

3.9

Weblate

BorgBackup

4.4.1: PostgreSQL / MySQL/MariaDB

Borg

Weblate

14.6

6.6

6.6
Backup service: /tmp/tx5y65uwebate

Backup service credentials
Feb 25, 2022

Backup repository: /tmp/tx5y65uwebate

Passphrase: ptO&rg5YPk&Fgfv7fZ8y24FpZ044eqbdFrgqJU%F&X)9E

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
Feb 25, 2022

Backup performed
Feb 26, 2022

Repository initialization
Feb 25, 2022

Activate support package

The support packages include priority email 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/repo for backups from remote SSH backups.

Add
Weblate

SSH:

-borg init

The backup creation can be customized using `BORG_EXTRA_ARGS`.

Weblate

1. https://weblate.org/support/#backup
2. Docker
3. Borg
4. Weblate
5. Borg:

Docker Compose:

```
services:
  weblate:
    volumes:
    - /home/weblate/data:/app/data
    - /home/weblate/borgbackup:/borgbackup
```

UID 1000

General

186
1. Prepare a server where your backups will be stored.
2. Install the SSH server on it (you will get it by default with most Linux distributions).
3. Install BorgBackup on that server; most Linux distributions have packages available (see Installation).
4. Choose an existing user or create a new user that will be used for backing up.
5. Add Weblate SSH key to the user so that Weblate can SSH to the server without a password (see Weblate SSH).
6. Weblate user@host:/path/to/backups
   ssh://user@host:port/path/to/backups

| BORGBACKUP | Weblate SSH | General |

**BorgBackup**

1. `borg list REPOSITORY`
2. `borg extract REPOSITORY::ARCHIVE`
3. Weblate `backup`:
   Dumped data for backups
4. Weblate `backups/settings.py`:
   Dumped data for backups
   When using Docker container, the settings file is already included in the container and you should restore the original environment variables. The `environment.yml` file might help you with this (see Dumped data for backups).
5. Weblate `DATA_DIR`:
   When using Docker container place the data into the data volume, see Docker container volumes.
   Please make sure the files have correct ownership and permissions, see.

| BORG | ...

$ borg list /tmp/xxx
Enter passphrase for key /tmp/xxx:
2019-09-26T14:56:08
Thu, 2019-09-26 14:56:08...
"[de0ef13643635d090e9896bdaceb92a023050749ad3f3350e788f1a65576a5]"

$ borg extract /tmp/xxx::2019-09-26T14:56:08
Enter passphrase for key /tmp/xxx:

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")
You can restore this backup in a newer Weblate release, it will perform all the necessary migrations when running in `migrate`. Please consult `Weblate` on more detailed info on how to upgrade between versions.

**Django**

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()
```

If you have enough backup space, simply back up the whole `DATA_DIR`. This is a safe bet even if it includes some files you don't want. The following sections describe what you should back up and what you can skip in detail.

**Dumped data for backups**

**4.7**

The environment dump was added as `environment.yml` to help in restoring in the Docker environments.

Stored in `DATA_DIR/backups`.

Weblate dumps various data here, and you can include these files for more complete backups. The files are updated daily (requires a running Celery beats server, see *Celery*). Currently, this includes:

- Weblate settings as `settings.py` (there is also expanded version in `settings-expanded.py`).
- PostgreSQL dumps as `database.sql`.
- Environment dump as `environment.yml`.

The database backups are saved as plain text by default, but they can also be compressed or entirely skipped using `DATABASE_BACKUP`.

To restore the database backup load it using database tools, for example:

```
psql --file=database.sql weblate
```
Version control repositories

Stored in `DATA_DIR/vcs`.

The version control repositories contain a copy of your upstream repositories with Weblate changes. If you have *enabled* for all your translation components, all Weblate changes are included upstream. No need to back up the repositories on the Weblate side as they can be cloned again from the upstream location(s) with no data loss.

SSH and GPG keys

Stored in `DATA_DIR/ssh` and `DATA_DIR/home`.

If you are using SSH or GPG keys generated by Weblate, you should back up these locations. Otherwise you will lose the private keys and you will have to regenerate new ones.

User uploaded files

Stored in `DATA_DIR/media`.

**Visual context for strings**

Celery tasks

The Celery task queue might contain some info, but is usually not needed for a backup. At most you will lose updates not yet been processed to translation memory. It is recommended to perform the fulltext or repository update upon restoration anyhow, so there is no problem in losing these.

**Restoring manual backup**

1. Restore all data you have backed up.
2. Update all repositories using `updategit`.

```
weblate updategit --all
```
Moving a Weblate installation

Relocate your installation to a different system by following the backing up and restoration instructions above.

Python 2  Python 3  PostgreSQL

Weblate  python-social-auth
REGISTRATION_OPEN

Django  Pootle

Weblate FORCE_EMAIL_VALIDATION

Django

Welcome to Python Social Auth’s documentation!

Welcome to Django Framework

Docker

SOCIAL_AUTH_OPENSUSE_FORCE_EMAIL_VALIDATION = True

Pipeline

Most of the authentication backends require HTTPS. Once HTTPS is enabled in your web server please configure Weblate to report it properly using ENABLE_HTTPS, or by WEBLATE_ENABLE_HTTPS in the Docker container.

Python Social Auth backend
# 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",  
)

# Social auth backends setup
SOCIAL_AUTH_GITHUB_KEY = "GitHub Client ID"
SOCIAL_AUTH_GITHUB_SECRET = "GitHub Client Secret"
SOCIAL_AUTH_GITHUB_SCOPE = ["user:email"]

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/`.

# Authentication configuration
AUTHENTICATION_BACKENDS = (  
    "social_core.backends.bitbucket.BitbucketOAuth",  
    "social_core.backends.email.EmailAuth",  
    "weblate.accounts.auth.WeblateUserBackend",  
)

# Social auth backends setup
SOCIAL_AUTH_BITBUCKET_KEY = "Bitbucket Client ID"
SOCIAL_AUTH_BITBUCKET_SECRET = "Bitbucket Client Secret"
SOCIAL_AUTH_BITBUCKET_VERIFIED_EMAILS_ONLY = True
Google OAuth 2

```python
# 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"
```

Facebook OAuth 2

```python
# 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"]
```
GitLab OAuth 2

GitLab OAuth 2

**URL**

https://gitlab.com/profile/applications

```
URL @ https://WEBLATE SERVER/accounts/complete/gitlab/
read_user
```

```
# 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/'
```

---

Microsoft Azure Active Directory

**URL**

https://WEBLATE SERVER/accounts/complete/azuread-oauth2/
https://WEBLATE SERVER/accounts/complete/azuread-tenant-oauth2/

```
# 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 = ""
```

---

193
Microsoft Azure Active Directory

Slack

Slack OAuth 2 URL https://api.slack.com/apps>

# Authentication configuration
AUTHENTICATION_BACKENDS = (  
    "social_core.backends.slack.SlackOAuth2",  
    "social_core.backends.email.EmailAuth",  
    "weblate.accounts.auth.WeblateUserBackend",  
)

# Social auth backends setup
SOCIAL_AUTH_SLACK_KEY = ""
SOCIAL_AUTH_SLACK_SECRET = ""

You can override the authentication method display name and icon using settings as SOCIAL_AUTH_<NAME>_IMAGE and SOCIAL_AUTH_<NAME>_TITLE. For example overriding naming for Auth0 would look like:

SOCIAL_AUTH_AUTH0_IMAGE = "custom.svg"
SOCIAL_AUTH_AUTH0_TITLE = "Custom auth"

opensUSE Open ID URL https://WEBLATE SERVER/accounts/complete/slug/
settings.py

AUTH_PASSWORD_VALIDATORS =

# Social auth backends setup
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": "MIIDJCCAvagAwIBAgIvBMdA ... B8bml+ev0pe1z+FyF5sQA==",
        "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"
}

SAML 4.1.1

Weblate IDP

SOCIAL_AUTH_SAML_ENABLED_IDPS = [weblate]

SAML XML URL https://accounts/metadata/saml/

SOCIAL_AUTH_SAML_SP_ENTITY_ID, SOCIAL_AUTH_SAML_TECHNICAL_CONTACT, SOCIAL_AUTH_SAML_SUPPORT_CONTACT

# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.email.EmailAuth",
    "social_core.backends.saml.SAMLAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)
The default configuration extracts user details from following attributes, configure your IDP to provide them:

<table>
<thead>
<tr>
<th>SAML URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>urn:oid:2.5.4.3</td>
</tr>
<tr>
<td>urn:oid:2.5.4.42</td>
</tr>
<tr>
<td>urn:oid:2.5.4.4</td>
</tr>
<tr>
<td>urn:oid:0.9.2342.19200300.100.1.3</td>
</tr>
<tr>
<td>urn:oid:0.9.2342.19200300.100.1.1</td>
</tr>
</tbody>
</table>

**Docker**

**weblate**

IDP IDP Relay

**Configuring SAML in Docker**

**SAML**

**LDAP**

**django-auth-ldap**

```python
# Using PyPI
pip install django-auth-ldap>=1.3.0

# Using apt-get
apt-get install python-django-auth-ldap
```

**Python LDAP 3.1.0**

```
AttributeError: 'module' object has no attribute '_trace_level'
```

**Django**

```
# Add LDAP backed, keep Django one if you want to be able to sign in
# even without LDAP for admin account
AUTHENTICATION_BACKENDS = {
    "django_auth_ldap.backend.LDAPBackend",
    "weblate.accounts.auth.WeblateUserBackend",
}

# LDAP server address
AUTH_LDAP_SERVER_URI = "ldaps://ldap.example.net"

# DN to use for authentication
AUTH_LDAP_USER_DN_TEMPLATE = "cn=%(user)s,o=Example"
# Depending on your LDAP server, you might use a different DN
# like:
# AUTH_LDAP_USER_DN_TEMPLATE = 'ou=users,dc=example,dc=com'

# List of attributes to import from LDAP upon sign in
# Weblate stores full name of the user in the full_name attribute
AUTH_LDAP_USER_ATTR_MAP = {
    "full_name": "name",
    # Use the following if your LDAP server does not have full name
    # Weblate will merge them later
    # 'first_name': 'givenName',
    # 'last_name': 'sn',
    # Email is required for Weblate (used in VCS commits)
    "email": "mail",
}

# Hide the registration form
REGISTRATION_OPEN = False
```
AUTHENTICATION_BACKENDS = [
'social_core.backends.email.EmailAuth',
'weblate.accounts.auth.WeblateUserBackend',
]

AUTHENTICATION_BACKENDS = ['django_auth_ldap.backends.ldap.LDAPBackend',
'weblate.accounts.auth.WeblateUserBackend']

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)"
)

import ldap
from django_auth_ldap.config import LDAPSearch,

AUTH_LDAP_BIND_DN = "CN=ldap,CN=Users,DC=example,DC=com"
AUTH_LDAP_BIND_PASSWORD = "password"

# User and group search objects and types
AUTH_LDAP_USER_SEARCH = LDAPSearch(
    "CN=Users,DC=example,DC=com", ldap.SCOPE_SUBTREE,
    "(sAMAccountName=\%(user)s)"
)

# Make selected group a superuser in Weblate
AUTH_LDAP_USER_FLAGS_BY_GROUP = {
    "is_superuser": "CN=weblate_AdminUsers,OU=Groups,DC=example,DC=com",
}

# Map groups from AD to Weblate
AUTH_LDAP_GROUP_SEARCH = LDAPSearch(
    "OU=Groups,DC=example,DC=com", ldap.SCOPE_SUBTREE,
    "(objectClass=group)"
)

AUTH_LDAP_GROUP_TYPE = NestedActiveDirectoryGroupType()

AUTH_LDAP_FIND_GROUP_PERMS = True

# Optionally enable group mirroring from LDAP to Weblate
# AUTH_LDAP_MIRROR_GROUPS = True

Django Authentication Using LDAP Authentication
**CAS**

**django-cas-ng**

**1**

**Webate**

**CAS**

**2**

**Weblate**

**CAS**

**docker-cas-ng**

**1**

**pip install django-cas-ng**

**settings.py**

```python
# Add CAS backed, keep the Django one if you want to be able to sign in
# even without LDAP for the admin account
AUTHENTICATION_BACKENDS = (
    "django_cas_ng.backends.CASBackend",
    "weblate.accounts.auth.WeblateUserBackend",
)

# CAS server address
CAS_SERVER_URL = "https://cas.example.net/cas/

# Add django_cas_ng somewhere in the list of INSTALLED_APPS
INSTALLED_APPS = (... , "django_cas_ng")
```

**urls.py**

```python
from django_cas_ng.signals import cas_user_authenticated
from django.dispatch import receiver

@receiver(cas_user_authenticated)
def update_user_email_address(sender, user=None, attributes=None, **kwargs):
    # If your CAS server does not always include the email attribute
    # you can wrap the next two lines of code in a try/catch block.
    user.email = attributes["email"]
    user.save()
```

**Django CAS NG**

**CAS**

**Weblate**

**Django CAS NG**

```python
from django_cas_ng.signals import cas_user_authenticated
from django.dispatch import receiver

@receiver(cas_user_authenticated)
def update_user_email_address(sender, user=None, attributes=None, **kwargs):
    # If your CAS server does not always include the email attribute
    # you can wrap the next two lines of code in a try/catch block.
    user.email = attributes["email"]
    user.save()
```

**Django**

**Webate**

**Django**

**CAS**

**Django**

**Webate**

**user backend**

**LDAP**

**CAS**

**Django**

**Webate**

**authentication**

**Django**

**Webate**

**user**

**backend**

**AUTHENTICATION_BACKENDS**

```python
AUTHENTICATION_BACKENDS = {
    # Add authentication backend here
    "weblate.accounts.auth.WeblateUserBackend",
}

INSTALLED_APPS += {
    # Install authentication app here
}
```
Also, besides adding an existing user to the project, it is possible to invite new ones. Any new user will be created immediately, but the account will remain inactive until signing in with a link in the invitation sent via an e-mail. It is not required to have any site-wide privileges in order to do so, access management permission on the project’s scope (e.g. a membership in the Administration team) would be sufficient.

If the invited user missed the validity of the invitation, they can set their password using invited e-mail address in the password reset form as the account is created already.

The same kind of invitations are available site-wide from the management interface on the Users tab.
In case some users behave badly in your project, you have an option to block them from contributing. The blocked user still will be able to see the project if he has permissions for that, but he won’t be able to contribute.

Webate manage users

By default this prevents Weblate from granting access provided by Users and Viewers default groups due to these groups’ own configuration. This doesn’t prevent you from granting permissions to those projects site-wide by altering default groups, creating a new one, or creating additional custom settings for individual component as described in below.

One of the main benefits of managing permissions through the Weblate user interface is that you can delegate it to other users without giving them the superuser privilege. In order to do so, add them to the Administration team of the project.

This section contains an overview of some common configurations you may be interested in.

To manage permissions for a whole instance at once, add users to appropriate default groups:

- Users (this is done by default by the automatic group assignment).
- Reviewers (if you are using review workflow with dedicated reviewers).
- Managers (if you want to delegate most of the management operations to somebody else).

You may also grant some additional permissions of your choice to the default groups. For example, you may want to give a permission to manage screenshots to all the Users.

You can define some new custom groups as well. If you want to keep managing your permissions site-wide for these groups, choose an appropriate value for the Project selection (e.g. All projects or All public projects).
You can create your own dedicated groups to manage permissions for distinct objects such as languages, components, and projects. Although these groups can only grant additional privileges, you can’t revoke any permission granted by site-wide or per-project groups by adding another custom group.

If you want (for whatever reason) to allow translation to a specific language (let’s say Czech) only to a closed set of reliable translators while keeping translations to other languages public, you will have to:

1. Remove the permission to translate Czech from all the users. In the default configuration this can be done by altering the Users default group.

   Table 2

<table>
<thead>
<tr>
<th>Permission</th>
</tr>
</thead>
<tbody>
<tr>
<td>All but Czech</td>
</tr>
</tbody>
</table>

2. Add a dedicated group for Czech translators.

   Table 3

<table>
<thead>
<tr>
<th>Permission</th>
</tr>
</thead>
<tbody>
<tr>
<td>Czech</td>
</tr>
<tr>
<td>All but Czech</td>
</tr>
</tbody>
</table>

3. Add users you wish to give the permissions to into this group.

As you can see, permissions management this way is powerful, but can be quite a tedious job. You can’t delegate it to another user, unless granting superuser permissions.

---

**Weblate:**

A group can have no roles assigned to it, in that case access to browse the project by anyone is assumed (see below).
The scope of the permission assigned by the roles in the groups are applied by the following rules:

If the group specifies any Component list, all the permissions given to members of that group are granted for all the components in the component lists attached to the group, and an access with no additional permissions is granted for all the projects these components are in. Components and Projects are ignored.

If the group specifies any Components, all the permissions given to the members of that group are granted for all the components attached to the group, and an access with no additional permissions is granted for all the projects these components are in. Projects are ignored.

Otherwise, if the group specifies any Projects, either by directly listing them or by having Projects selection set to a value like All public projects, all those permissions are applied to all the projects, which effectively grants the same permissions to access all projects unrestricted components.

The restrictions imposed by a group’s Languages are applied separately, when it’s verified if a user has an access to perform certain actions. Namely, it’s applied only to actions directly related to the translation process itself like reviewing, saving translations, adding suggestions, etc.

Let’s say there is a project foo with the components: foo/bar and foo/baz and the following group:

Table4Group Spanish Admin-Reviewers

Members of that group will have following permissions (assuming the default role settings):

General (browsing) access to the whole project foo including both components in it: foo/bar and foo/baz.

Review strings in foo/bar Spanish translation (not elsewhere).

Manage VCS for the whole foo/bar repository e.g. commit pending changes made by translators for all languages.

Django admin interface

The most common use-case for the feature is to assign all new users to some default group. In order to do so, you will probably want to keep the default value (^.*$) in the regular expression field. Another use-case for this option might be to give some additional privileges to employees of your company by default. Assuming all of them use corporate e-mail addresses on your domain, this can be accomplished with an expression like ^.*@mycompany.com.

As for now, there is no way to bulk-add already existing users to some group via the user interface. For that, you may resort to using the REST API.
After installation, a default set of groups is created (see VCS).
If you want to use your Weblate installation in a less public manner, i.e. allow new users on an invitational basis only, it can be done by configuring Weblate in such a way that only known users have an access to it. In order to do so, you can set `REGISTRATION_OPEN` to `False` to prevent registrations of any new users, and set `REQUIRE_LOGIN` to `/.\*` to require logging-in to access all the site pages. This is basically the way to lock your Weblate installation.

You can use built-in invitations to add new users.

### 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 `Component configuration` defined in `Component configuration`) appear in the VCS repository.

You can share cloned VCS repositories using `Weblate URL`. Using this feature is highly recommended when you have many components sharing the same VCS. It improves performance and decreases required disk space.
Adding translation projects and components

**3.2 Add new projects:** An interface for adding projects and components is included, and you no longer have to use Django.

**3.4 Add new components:** 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](https://hosted.weblate.org/see)), 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:

![Billing Plan](image)

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:

- Creates component from remote version control repository.
- Creates additional component to existing one by choosing different files.
- Creates additional component to existing one, just for different branch.
- 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 REST API.
- Upload single document or translation file and translate that.
- 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 the same repository.

First you need to fill in name and repository location:

<table>
<thead>
<tr>
<th>Component name</th>
<th>Language names</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display name</td>
<td></td>
</tr>
</tbody>
</table>

URL slug:

Name used in URLs and filenames.

| Use as a glossary |                   |

Project:

WeblateOrg

Source language:

English

Version control system:

Git

Version control system to use to access your repository with translations.

Source code repository:

https://github.com/WeblateOrg/demo.git

URL of a repository, see weblate://project/component for sharing with other component.

Repository branch:

Repository branch to translate

Continue

On the next page, you are presented with a list of discovered translatable resources:

<table>
<thead>
<tr>
<th>Add new translation component</th>
<th></th>
</tr>
</thead>
</table>

Choose translation files to import:

- Specify configuration manually
- File format Android String Resource, file mask app/src/main/res/values-<string>.xml
- File format gettext PO file, file mask weblate/langdata/locale/<LC_MESSAGES>/django.po
- File format gettext PO file, file mask weblate/locale/<LC_MESSAGES>/django.pot
- File format gettext PO file, file mask weblate/locale/<LC_MESSAGES>/djangojs.po

Continue
As a last step, you review the translation component info and fill in optional details:
Create component

Add new translation component

Project
WebsiteOrg

Component name
Language names
Display name
URL slug
language names
None used in URL 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/WebsiteOrg/...ino.git

URL of a repository, use website:/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.

Branch for pushing changes, leave empty to use repository branch
Repository browser
https://github.com/WebsiteOrg/...ino.git[branch]

Use repository browser, use [branch] for branch, [filename] and [file] as filename and file placeholders. You might want to strip leading directory by using [filename]{parent/}

File format
gettext PO file

File mask
project.\*.po

Path of files to translate relative to repository root, use * instead of language code, for example: .po or locale/\*.LC_MESSAGES\*/language.po.

Mandatory base language file
application/\*.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 translation.

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
application/\*.strings.xml

Filename of file used for creating new translations. For gettext show pot file.

Translation license

OMG General Public License v3 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.

Custom domain name used to generate the filename for translations created by Website.

Language filter

[YourRegex]

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.
Project configuration

These basic attributes set up and inform translators of a project:

Verbose project name, used to display the project name.

URL

Project name suitable for URLs.

Web

URL where translators can find more info about the project.
This is a required parameter unless turned off by WEBSITE_REQUIRED.

Text describing localization process in the project, and any other information useful for translators. Markdown can be used for text formatting or inserting links.

"Language-Team"

Whether Weblate should manage the Language-Team header (this is a GNU gettext only feature right now).

Whether to use shared translation memory, see DEFAULT_SHARED_TM for more details.

Whether to contribute to shared translation memory, see DEFAULT_SHARED_TM for more details.
Configure per project access control, see `DEFAULT_ACCESS_CONTROL` for more details.
Default value can be changed by `DEFAULT_ACCESS_CONTROL`.

Enable review workflow for translations, see `review`.

Enable review workflow for source strings, see `review`.

Whether unauthenticated users are to be used for this repository.

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`

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.

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.

You can find some examples of typical configurations in the `Component configuration`.

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:
Verbose component name, used to display the component name.

Component name suitable for URLs.

**Component project**

*Project configuration* where the component belongs.

**VCS to use**, see [VCS](#) for details.

Pushing changes from Weblate

**VCS repository used to pull changes.**

See [VCS](#) for more details on specifying URLs.

**URL**

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.

See [VCS](#) for more details on how to specify a repository URL and *Pushing changes from Weblate* for more details on pushing changes from Weblate.

URL of repository browser used to display source files (location of used messages). When empty, no such links will be generated. You can use `{{parentdir}}`.

For example on GitHub, use something like: `https://github.com/WeblateOrg/hello/blob/{{branch}}/{{filename}}#L{{line}}`

In case your paths are relative to different folder (path contains `..`), you might want to strip leading directory by `parentdir` filter (see [filter](#)): `https://github.com/WeblateOrg/hello/blob/{{branch}}/{{filename|parentdir}}#L{{line}}`
URL

URL where changes made by Weblate are exported. This is important when Weblate is not used, or when there is a need to manually merge changes. You can use Git to automate this for Git repositories.

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.

Branch for pushing changes, leave empty to use . For linked repositories, this is not used and setting from linked component applies.

This is currently only supported for Git, GitLab and GitHub, it is ignored for other VCS integrations.

Mask of files to translate, including path. It should include one "*" replacing language code (see 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.

In case your filename contains special characters such as [, ], these need to be escaped as [[]] or [[]].

When set, the source strings are based on this file, but all other languages are based on . In case the string is not translated into the source language, translating to other languages is prohibited. This provides

215
Base file used to generate new translations, e.g. .pot file with gettext.

Adding-translation:

Translation file format, see also...

You can turn off propagation of translations to this component from other components within the 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.

Keeping translations same across components

Whether translation suggestions are accepted for this component.

Turns on vote casting for suggestions, see ...

Automatically accept voted suggestions, see ...

Webate
List of checks which can not be ignored, see [source](#).

**Enforce**: Enforcing the check does not automatically enable it, you still should enabled it using [Additional info on source strings](#) or [source](#).

License of the translation (does not need to be the same as the source code license).

How to handle requests for creation of new languages. Available options:

- User can select desired language and the project maintainers will receive a notification about this. It is up to them to add (or not) the language to the repository.
- 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).
- User can select language and Weblate automatically creates the file for it and translation can begin. There will be no option for user to start new translation.

- The project admins can add new translations even if it is disabled here when it is possible (either or the file format supports starting from an empty file).

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.

**Adding new strings**: [source](#)
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 as VCS, especially when pushing to a different branch.

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.

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.

Message used when committing a translation, see DEFAULT_ADD_MESSAGE.

COMMIT_PENDING_HOURS

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.

COMMIT_PENDING_HOURS

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.

There are other situations where pending changes might be committed, see Lazy commits.

Locks the component (and linked components, see Weblate URL) 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.
Language used for source strings. Change this if you are translating from something else than English.

**Note:** 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 [variants](#).

---

**Weblate:** You need to list language codes as they appear in the filename.

Some examples of filtering:

<table>
<thead>
<tr>
<th>Filter description</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selected languages only</td>
<td>^(cs</td>
</tr>
<tr>
<td>Exclude languages</td>
<td>^(?!(it</td>
</tr>
<tr>
<td>Filter two letter codes only</td>
<td>^.+$</td>
</tr>
<tr>
<td>Exclude non language files</td>
<td>^(?!(blank)$).+$</td>
</tr>
<tr>
<td>Include all files (default)</td>
<td>^[^.]+$</td>
</tr>
</tbody>
</table>

---

**Weblate:** variants

**Weblate, Git, Mercurial, VCS, alerts**

---

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.
4.5. The glossary will be accessible in all projects defined by \texttt{GOOGLEPROJECTS}.

It is recommended to enable \texttt{GOOGLEGLOSSTOR} on glossaries in order to allow adding new words to them.

\texttt{GOOGLEGLOSSTOR}:

Display color for a glossary used when showing word matches.

\texttt{GOOGLEGLOSSTOR}:

Weblate \texttt{THE_DJANGO_TEMPLATE_LANGUAGE} \texttt{COMPONENT_CONFIGURATION} The Django Template Language \texttt{THE_DJANGO_TEMPLATE_LANGUAGE}.

\texttt{THE_DJANGO_TEMPLATE_LANGUAGE}: Component configuration
<table>
<thead>
<tr>
<th>Component</th>
<th>Title</th>
</tr>
</thead>
</table>

```{% if stats.translated_percent > 80 %}Well translated!{% endif %}

{% replace component "-" " " %}

{% replace component|capfirst "-" " " %}

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 }}

... Django

VCS Weblate ref:production

Celery

CPU 221
Weblate ISO 639-1

--
cs_CZ
cs

xx_XX (generated):

Weblate

/languages/ URL
Weblate Weblate

weblate migrate

UPDATE LANGUAGES setuplang

Weblate

Gettext

GNU gettext utilities: Plural forms
Language Plural Rules by the Unicode Consortium
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 Weblate).
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).
Updating repositories

You should set up some way of updating backend repositories from their source.

Automatically receiving changes from GitHub
Automatically receiving changes from GitLab
Automatically receiving changes from Bitbucket
Pagure
Automatically receiving changes from Azure Repos

Avoiding merge conflicts

The merge conflicts from Weblate arise when same file was changed both in Weblate and outside it. There are two approaches to deal with that - avoid edits outside Weblate or integrate Weblate into your updating process, so that it flushes changes prior to updating the files outside Weblate.

The first approach is easy with monolingual files - you can add new strings within Weblate and leave whole editing of the files there. For bilingual files, there is usually some kind of message extraction process to generate translatable files from the source code. In some cases this can be split into two parts - one for the extraction generates template (for example gettext POT is generated using `xgettext`) and then further process merges it into actual translations (the gettext PO files are updated using `msgmerge`). You can perform the second step within Weblate and it will make sure that all pending changes are included prior to this operation.
The second approach can be achieved by using Weblate REST API to force Weblate to push all pending changes and lock the translation while you are doing changes on your side.

The script for doing updates can look like this:

```
# Lock Weblate translation
wlc lock
# Push changes from Weblate to upstream repository
wlc push
# Pull changes from upstream repository to your local copy
git pull
# Update translation files, this example is for Django
./manage.py makemessages --keep-pot -a
# Push changes to upstream repository
git commit -m 'Locale updates' -- locale
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
```

The example uses Weblate, which needs configuration (API keys) to be able to control Weblate remotely. You can also achieve this using any HTTP client instead of wlc, e.g. curl, see Weblate REST API.

Weblate comes with native support for GitHub.

Automatically receiving changes from GitHub

Weblate comes with native support for GitHub.

If you are using Hosted Weblate, the recommended approach is to install the Weblate app, that way you will get the correct setup without having to set much up. It can also be used for pushing changes back.

To receive notifications on every push to a GitHub repository, add the Weblate Webhook in the repository settings (Webhooks) as shown on the image below:
For the payload URL, append /hooks/github/ to your Weblate URL, for example for the Hosted Weblate service, this is https://hosted.weblate.org/hooks/github/.

You can leave other values at default settings (Weblate can handle both content types and consumes just the push event).

**POST /hooks/github/**

**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/).
POST /hooks/bitbucket/

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/).

POST /hooks/gitlab/

Pagure

Pagure 3.3.

Webhooks

Activate Web-hooks

https://hosted.weblate.org/hooks/pagure/
**POST /hooks/pagure/**

Alternatively receiving changes from Azure Repos

**3.8 Weblate**

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.

**Web hooks in Azure DevOps manual**
Automatically receiving changes from Gitea Repos

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.

Webhooks in Gitea manual

Automatically receiving changes from Gitee Repos

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.

Webhooks in Gitee manual

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.

Pushing changes from Weblate

Each translation component can have a push URL set up (see URL), and in that case Weblate will be able to push change to the remote repository. Weblate can also be configured to automatically push changes on every commit (this is default, see AUTO_UPDATE). 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 used, more details are found in that chapter.

<table>
<thead>
<tr>
<th>Desired setup</th>
<th>URL</th>
<th>push</th>
</tr>
</thead>
<tbody>
<tr>
<td>No push</td>
<td>empty</td>
<td>empty</td>
</tr>
<tr>
<td>Push directly</td>
<td>Git</td>
<td>SSH URL</td>
</tr>
<tr>
<td>Push directly</td>
<td>Git</td>
<td>SSH URL</td>
</tr>
<tr>
<td>GitHub pull request from fork</td>
<td>GitHub</td>
<td>empty</td>
</tr>
<tr>
<td>GitHub pull request from branch</td>
<td>GitHub</td>
<td>SSH URL</td>
</tr>
<tr>
<td>GitLab merge request from fork</td>
<td>GitLab</td>
<td>empty</td>
</tr>
<tr>
<td>GitLab merge request from branch</td>
<td>GitLab</td>
<td>SSH URL</td>
</tr>
<tr>
<td>Pagure pull request</td>
<td>Pagure</td>
<td>empty</td>
</tr>
<tr>
<td>Pagure pull request</td>
<td>Pagure</td>
<td>SSH URL</td>
</tr>
</tbody>
</table>

You can also enable automatic pushing of changes after Weblate commits, this can be done in Repository maintenance.

See Component configuration for setting up SSH keys, and Lazy commits for info about when Weblate decides to commit changes.

Can be empty in case supports pushing.
**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:

**Interacting with others**

Weblate makes it easy to interact with others using its API.

**Webate REST API**

**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 [Component configuration](#Component configuration)).

The changes in this mode are committed once any of the following conditions are fulfilled:

- Somebody else changes an already changed string.
- An explicit commit is requested.
- Change is older than period defined as [Component configuration](#Component configuration).

| Note | Commits are created for every component. So in case you have many components you will still see lot of commits. You might utilize Git 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",
        # Committing 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,
    }
}

Processing repository with scripts

The way to customize how Weblate interacts with the repository is [Component configuration]. Consult [Component configuration] for info on how to execute external scripts through add-ons.

Keeping translations same across components

Once you have multiple translation components, you might want to ensure that the same strings have same translation. This can be achieved at several levels.

Translation propagation

With [Component configuration] enabled (what is the default, see Component configuration), all new translations are automatically done in all components with matching strings. Such translations are properly credited to currently translating user in all components.

Notice: 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 check fires whenever the strings are different. You can utilize this to review such differences manually and choose the right translation.

Automatic translation based on different components can be way to synchronize the translations across components. You can either trigger it manually (see Component configuration) or make it run automatically on repository update using add-on (see Component configuration).
<table>
<thead>
<tr>
<th>Language</th>
<th>Translated</th>
<th>Unfinished</th>
<th>Unfinished words</th>
<th>Checks</th>
<th>Suggestions</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Czech</td>
<td>✔️</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hebrew</td>
<td>✔️</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hungarian</td>
<td>✔️</td>
<td>81%</td>
<td>4</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>English</td>
<td>✔️</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Start new translation

Powered by Weblate 4.11  About Weblate  Legal  Contact  Documentation  Donate to Weblate
Component configuration:

**permission access control**: 

- WeblateOrg/Language names

**Licensing**

Please pay attention to the licensing info, as this specifies how translations can be used.

By registering you agree to use your name and e-mail in the commits, and provide your contribution under the license defined by each localization project.

You have agreed to the following as a contributor:

- WeblateOrg/Language names

**Licenses for individual translations**

- **GNU General Public License v3.0 or later**
  - WeblateOrg/Django
  - WeblateOrg/WebateOrg
  - WeblateOrg/Language names

- **MIT License**
  - WeblateOrg/Android
Additional info on source strings

Edit additional string info

Explanation

Additional explanation to clarify meaning or usage of the string.

Labels

- General purpose
- Next update

Additional labels can be defined in the project settings.

Translation flags

Additional comma-separated flags to influence Weblate behavior.

Save and continue

Save
Automatic překlad prostřednictvím strojového překladi má použít aktuální strojového překladi pro získání nejlepších možných překladů a použije je na tento projekt.

Automatic translation uses active machine translation engines to get the best possible translations and applies them in this project.

You can add new translation strings here, it will automatically appear in all translations.

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.

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.

The latest commit or merge got not work properly as the latest index for this translation is not yet up to date.

Review

Frenen translations touched by other users.

Start review

Percent

Total

Paling check

Last activity

Last change

Last author

What is %?

What is %?

The string uses three dots (…) instead of an ellipsis character (…).
Strings prioritization

2.0

String priority can be changed to offer higher priority strings for translation earlier by using the priority flag.

This can be used to order the flow of translation in a logical manner.

2.4

3.3: Previously called Quality checks flags, it no longer configures only checks.

The string flags are also inherited from the at Component configuration and flags from the translation file (see).

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

2.9

You can upload a screenshot showing a given source string in use within your program. This helps translators understand where it is used, and how it should be translated.
In addition to *Additional info on source strings*, screenshots have a separate management interface under the *Tools* menu. Upload screenshots, assign them to source strings manually, or use optical character recognition to do so.

Once a screenshot is uploaded, this interface handles management and source string association:
null
# 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):
    name = _("Foobar")

    def fix_single_target(self, target, source, unit):
        if "foo" in target:
            return target.replace("foo", "bar"), True
        return target, False
ICU MessageFormat

ICU MessageFormat

replacements:%s:"John Doe"

variants:

AngularJS
C
C#
ECMAScript
i18next
ICU MessageFormat
Java
Java MessageFormat
JavaScript
Lua
Object Pascal
Perl
PHP
Python
Qt
Qt Python
Ruby
Scheme
Vue I18n

Markdown
HTML
URL

Ignore all quality checks.

BBCode
AngularJS
C
C#
ECMAScript
i18next
ICU MessageFormat syntax

Component configuration: GNU gettext

3.11

Component configuration: Weblate

3.7

Weblate: TrueType/OpenType
### Font group

<table>
<thead>
<tr>
<th>Name</th>
<th>default-font</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default font</td>
<td>Source Sans 3 Bold</td>
</tr>
<tr>
<td>Japanese</td>
<td>language override Droid Sans Fallback Regular</td>
</tr>
<tr>
<td>Korean</td>
<td>language override Droid Sans Fallback Regular</td>
</tr>
</tbody>
</table>

### Add language override

- **Language**: 
- **Font**: 

**Save**

### Edit font group

- **Font group name**: default-font
- **Default font**: Source Sans 3 Bold  
  - Default font is used unless a language override matches.

**Save**
<table>
<thead>
<tr>
<th>Font Group</th>
<th>Default Font</th>
<th>Language Overrides</th>
</tr>
</thead>
<tbody>
<tr>
<td>default-font</td>
<td>Source Sans 3 Bold</td>
<td>Japanese: Droid Sans Fallback Regular, Korean: Droid Sans Fallback Regular</td>
</tr>
</tbody>
</table>

Add font group

Font group name

Identifier you will use in checks to select this font group. Avoid whitespaces and special characters.

Default font

---------

Default font is used unless per language override matches.
1. `weblate.checks.Check`

2. `check`

3. `check_single`

```
# 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.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")

    def check_single(self, source, target, unit):
        return "foo" in target

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")
```python
# 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
```

Project configuration

amaGama

- Weblate machinery.tmservice.AmagamaTranslation

Installing amaGama

Apertium

- Weblate machinery.apertium.ApertiumAPYTranslation

AWS

- Weblate machinery.aws.AWSTranslation

MT_SERVICES:

- Project configuration
- amaGama
- Apertium
- AWS
Baidu API machine translation

3.2 Baidu

This service uses an API and you need to obtain an ID and API key from Baidu to use it.

Turn on this service by adding `weblate.machinery.baidu.BaiduTranslation` to `MT_SERVICES` and set `MT_BAIDU_ID` and `MT_BAIDU_SECRET`.

MT_BAIDU_ID, MT_BAIDU_SECRET Baidu Translate API

DeepL

2.20 DeepL

DeepL is a paid service providing good machine translation for a few languages. You need to purchase DeepL API subscription or you can use legacy DeepL Pro (classic) plan.

Turn on this service by adding `weblate.machinery.deepl.DeepLTranslation` to `MT_SERVICES` and set `MT_DEEPL_KEY`.

MT_DEEPL_KEY DeepL website DeepL pricing DeepL API documentation

LibreTranslate

4.7.1 LibreTranslate

LibreTranslate is a free and open-source service for machine translations. The public instance requires an API key, but LibreTranslate can be self-hosted and there are several mirrors available to use the API for free.

Turn on this service by adding `weblate.machinery.libretranslate.LibreTranslateTranslation` to `MT_SERVICES` and set `MT_LIBRETRANSLATE_API_URL`, `MT_LIBRETRANSLATE_KEY`.

MT_LIBRETRANSLATE_KEY, MT_LIBRETRANSLATE_API_URL LibreTranslate website, LibreTranslate repository, LibreTranslate mirrors

Glosbe

Free dictionary and translation memory for almost every living language.

The API is gratis to use, but subject to the used data source license. There is a limit of calls that may be done from one IP in a set period of time, to prevent abuse.

Turn on this service by adding `weblate.machinery.glosbe.GlosbeTranslation` to `MT_SERVICES`.

Glosbe website
Google Translate

This service uses the Google Translation API, and you need to obtain an API key and turn on billing in the Google API console.

To turn on this service, add `weblate.machinery.google.GoogleTranslation` to `MT_SERVICES` and set `MT_GOOGLE_KEY`.

**MT_GOOGLE_KEY**

Google Translate documentation

Google Translate API V3 (Advanced)

This service differs from the former one in how it authenticates. To enable service, add `weblate.machinery.googlev3.GoogleV3Translation` to `MT_SERVICES` and set `MT_GOOGLE_CREDENTIALS` and `MT_GOOGLE_PROJECT`.

If `location` fails, you may also need to specify `MT_GOOGLE_LOCATION`.

**MT_GOOGLE_CREDENTIALS**
**MT_GOOGLE_PROJECT**
**MT_GOOGLE_LOCATION**

Microsoft Cognitive Services Translator

This service differs from the former one in how it authenticates. To enable service, add `weblate.machinery.microsoft.MicrosoftCognitiveTranslation` to `MT_SERVICES` and set `MT_MICROSOFT_COGNITIVE_KEY`.

Translator Text API V2

The key you use with Translator API V2 can be used with API 3.

Translator Text API V3

The key you use with Translator API V3 can be used with API 3.

Microsoft Terminology Service

The Microsoft Terminology Service API can be accessed via `weblate.machinery.microsoftterminology.MicrosoftTerminologyService` in `MT_SERVICES`.
ModernMT

ModernMT 4.2

Turn this service on by adding `weblate.machinery.modernmt.ModernMTTranslation` to `MT_SERVICES` and configure `MT_MODERNMT_KEY`.

ModernMT API, `MT_MODERNMT_KEY`, `MT_MODERNMT_URL`

MyMemory

Huge translation memory with machine translation.

Free anonymous usage is currently limited to 100 requests/day, or to 1000 requests/day when you provide a contact e-mail address in `MT_MYMEMORY_EMAIL`. You can also ask them for more.

Turn on this service by adding `weblate.machinery.mymemory.MyMemoryTranslation` to `MT_SERVICES` and set `MT_MYMEMORY_EMAIL`.

MyMemory website

NetEase Sight API machine translation

NetEase 3.3

This service uses an API, and you need to obtain key and secret from NetEase.

Turn on this service by adding `weblate.machinery.youdao.NeteaseSightTranslation` to `MT_SERVICES` and set `MT_NETEASE_KEY` and `MT_NETEASE_SECRET`.

NetEase Sight Translation Platform

tmserver

You can run your own translation memory server by using the one bundled with Translate-toolkit and let Weblate talk to it. You can also use it with an amaGama server, which is an enhanced version of tmserver.

1. First you will want to import some data to the translation memory:

2. Turn on this service by adding `weblate.machinery.tmserver.TMServerTranslation` to `MT_SERVICES`.

   ```
   build_tmdb -d /var/lib/tm/db -s en -t cs locale/cs/LC_MESSAGES/django.po
   build_tmdb -d /var/lib/tm/db -s en -t de locale/de/LC_MESSAGES/django.po
   build_tmdb -d /var/lib/tm/db -s en -t fr locale/fr/LC_MESSAGES/django.po
   ```

3. Start tmserver to listen to your requests:

   ```
   tmserver -d /var/lib/tm/db
   ```

4. Configure Weblate to talk to it:

   ```
   MT_TMSERVER = "http://localhost:8888/tmserver/
   ```

   tmserver Installing amaGama Amagama Amagama Translation Memory
Yandex Translate

Yandex Translate API
Powered by Yandex.Translate

This service uses a Translation API, and you need to obtain an API key from Yandex.

Turn on this service by adding `weblate.machinery.yandex.YandexTranslation` to `MT_SERVICES`, and set `MT_YANDEX_KEY`.

MT_YANDEX_KEY

Yandex Translate API
Powered by Yandex.Translate

Youdao Zhiyun API machine translation

Youdao Zhiyun Natural Language Translation Service

This service uses an API, and you need to obtain an ID and an API key from Youdao.

Turn on this service by adding `weblate.machinery.youdao.YoudaoTranslation` to `MT_SERVICES` and set `MT_YOUDAO_ID` and `MT_YOUDAO_SECRET`.

MT_YOUDAO_ID
MT_YOUDAO_SECRET
Youdao Zhiyun Natural Language Translation Service

Weblate

Weblate can be the source of machine translations as well. It is based on the Woosh fulltext engine, and provides both exact and inexact matches.

Turn on these services by adding `weblate.machinery.weblatetm.WeblateTranslation` to `MT_SERVICES`.

Weblate Translation Memory

Weblate Translation Memory API

The can be used as a source for machine translation suggestions as well.

Turn on these services by adding `weblate.memory.machine.WeblateMemory` to the `MT_SERVICES`.
This service is turned on by default.

SAP Translation Hub

SAP Translation Hub API

You need to have a SAP account (and the SAP Translation Hub enabled in the SAP Cloud Platform) to use this service.

`weblate.machinery.saptranslationhub.SAPTranslationHub` to `MT_SERVICES`

MT_SAP_BASE_URL
MT_SAP_SANDBOX_APIKEY
MT_SAP_USERNAME
MT_SAP_PASSWORD
MT_SAP_USE_MT

To access the Sandbox API, you need to set `MT_SAP_BASE_URL` and `MT_SAP_SANDBOX_APIKEY`.

To access the productive API, you need to set `MT_SAP_BASE_URL`, `MT_SAP_USERNAME` and `MT_SAP_PASSWORD`.

MT_SAP_BASE_URL
MT_SAP_SANDBOX_APIKEY
MT_SAP_USERNAME
MT_SAP_PASSWORD
MT_SAP_USE_MT

SAP Translation Hub API

252
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:

```python
# Copyright © 2012–2022 Michal Čihař <michal@cihar.com>
# This file is part of Weblate <https://weblate.org/>
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.
# You should have received a copy of the GNU General Public License
# along with this program. If not, see <https://www.gnu.org/licenses/>.
# """Machine translation example.""

import dictionary
from weblate.machinery.base import MachineTranslation

class SampleTranslation(MachineTranslation):
    """Sample machine translation interface.""
    name = "Sample"

def download_languages(self):
    """Return list of languages your machine translation supports.""
    return {"cs"}

def download_translations(self, source, language, text, unit, user, search, threshold=75,):
    """Return tuple with translations.""
    for t in dictionary.translate(text):
        yield {"text": t, "quality": 100, "service": self.name, "source": text}

You can list your own class in `MT_SERVICES` and Weblate will start using that.

253
2.19 API

API

DEFAULT_ADDONS

install_addon
# Installed add-ons

There are no add-ons currently installed.

## Available add-ons

<table>
<thead>
<tr>
<th>Add-on</th>
<th>Description</th>
<th>Install Status</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Automatic translation</strong></td>
<td>Automatically translates strings using machine translation or other components.</td>
<td><img src="Install" alt="Install" /></td>
</tr>
<tr>
<td><strong>Add missing languages</strong></td>
<td>Ensures a consistent set of languages is used for all components within a project.</td>
<td><img src="Install" alt="Install" /></td>
</tr>
<tr>
<td><strong>Component discovery</strong></td>
<td>Automatically adds or removes project components based on file changes in the version control system.</td>
<td><img src="Install" alt="Install" /></td>
</tr>
<tr>
<td><strong>Bulk edit</strong></td>
<td>Bulk edit flags, labels, or states of strings.</td>
<td><img src="Install" alt="Install" /></td>
</tr>
<tr>
<td><strong>Statistics generator</strong></td>
<td>Generates a file containing detailed info about the translation status.</td>
<td><img src="Install" alt="Install" /></td>
</tr>
<tr>
<td><strong>Prefill translation with source</strong></td>
<td>Fills in translation strings with source string.</td>
<td><img src="Install" alt="Install" /></td>
</tr>
<tr>
<td><strong>Pseudolocale generation</strong></td>
<td>Generates a translation by adding prefix and suffix to source strings automatically.</td>
<td><img src="Install" alt="Install" /></td>
</tr>
<tr>
<td><strong>Contributors in comment</strong></td>
<td>Updates the comment part of the PO file header to include contributor names and years of contributions.</td>
<td><img src="Install" alt="Install" /></td>
</tr>
<tr>
<td><strong>Customize gettext output</strong></td>
<td>Allows customization of gettext output behavior, for example line wrapping.</td>
<td><img src="Install" alt="Install" /></td>
</tr>
<tr>
<td><strong>Generate MO files</strong></td>
<td>Automatically generates a MO file for every changed PO file.</td>
<td><img src="Install" alt="Install" /></td>
</tr>
<tr>
<td><strong>Update PO files to match POT (msgmerge)</strong></td>
<td>Updates all PO files (as configured by &quot;file mask&quot;) to match the POT file (as configured by &quot;Template for new translations&quot;) using msgmerge.</td>
<td><img src="Install" alt="Install" /></td>
</tr>
<tr>
<td><strong>Squash Git commits</strong></td>
<td>Squash Git commits prior to pushing changes.</td>
<td><img src="Install" alt="Install" /></td>
</tr>
<tr>
<td><strong>Stale comment removal</strong></td>
<td>Set a timeframe for removal of comments.</td>
<td><img src="Install" alt="Install" /></td>
</tr>
<tr>
<td><strong>Stale suggestion removal</strong></td>
<td>Set a timeframe for removal of suggestions.</td>
<td><img src="Install" alt="Install" /></td>
</tr>
</tbody>
</table>

*Some add-ons will ask for additional configuration during installation.*
3.9 web.
weblate.autotranslate.autotranslate

mode [suggest translate fuzzy]
   suggest -- [suggest]
   translate -- [translate]
   fuzzy -- [fuzzy]

filter_type [all nottranslated todo fuzzy]
   all -- [all]
   nottranslated -- [nottranslated]
   todo -- [todo]
   fuzzy -- [fuzzy]
   check:inconsistent -- [check:inconsistent]

auto [others mt]
   others -- [others]
   mt -- [mt]

component [en]
   en -- [en]

gines thresholds

update daily

BACKGROUND_TASKS

JavaScript CDN

4.2 CDN
weblate.cdn.cdnjs

threshold [0 256]
css_selector [CSS]
cookie_name [Cookie]
files [HTML URL]

daily post-commit post-update

JavaScript HTML HTML JavaScript HTML HTML HTML HTML

weblate-cdn cdn-addon-config cdn-addon-extract cdn-addon-html

cdn-addon-config weblate-cdn cdn-addon-extract cdn-addon-html

256
4.4
weblate.cleanup.blank

post-commit post-update

Weblate

weblate.cleanup.generic

pre-commit post-update

Weblate

weblate.consistency.languages

daily post-add

24 1 Weblate

weblate.discovery.discovery

match
file_format
name_template
base_file_temp
new_base_temp
language_regex
copy_addons
remove
confirm

post-update

VCS
import_project
<table>
<thead>
<tr>
<th>Component</th>
<th>Matched files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Django</td>
<td><code>website/locale/zh/C_MESSAGES/django.po</code> (zh) <code>website/locale/zh/C_MESSAGES/django.po</code> (zh) <code>website/locale/zh/C_MESSAGES/django.po</code> (zh)</td>
</tr>
<tr>
<td>Django</td>
<td><code>website/locale/nl/C_MESSAGES/django.po</code> (nl) <code>website/locale/nl/C_MESSAGES/django.po</code> (nl) <code>website/locale/nl/C_MESSAGES/django.po</code> (nl)</td>
</tr>
</tbody>
</table>

- **Please review and confirm the matched components.**

### The following components would be created

- **Django**
  - `website/locale/zh/C_MESSAGES/django.po` (zh)
  - `website/locale/nl/C_MESSAGES/django.po` (nl)

**Regular expression to match translation files against**

```
website/locale/[a-z]{4}/C_MESSAGES/[a-z]{2}/LC_MESSAGES/[a-z]{2}/component/[a-zA-Z]+.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**

- `website/locale/[[component]]/pot/`

**Filename for use for creating new translations. For gettext choose .pot file.**

**Language filter**

```
[\w\W]*\[a-z]{2}\W*
```

**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 nonexistent files**

### Regular expression to match translation files has to contain two named groups to match component and language, some examples:

<table>
<thead>
<tr>
<th>Regular expression</th>
<th>Example matched files</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>{\w+}/language/{\w+}$/LC_MESSAGES/{\w+}.po</code></td>
<td><code>cs/application.po</code> <code>cs/website.po</code> <code>de/application.po</code> <code>de/website.po</code></td>
<td>One folder per language containing translation files for components.</td>
</tr>
<tr>
<td><code>locale/{\w+}/LC_MESSAGES/{\w+}/component/[a-zA-Z]+.po</code></td>
<td><code>locale/cs/C_MESSAGES/application.po</code> <code>locale/cs/C_MESSAGES/website.po</code> <code>locale/de/C_MESSAGES/application.po</code> <code>locale/de/C_MESSAGES/website.po</code></td>
<td>Short structure for storing gettext PO files.</td>
</tr>
<tr>
<td><code>src/locale/{\w+}/C_FILES/{\w+}/component/[a-zA-Z]+.po</code></td>
<td><code>src/locale/cs/C_FILES/application.cs.po</code> <code>src/locale/cs/C_FILES/website.cs.po</code> <code>src/locale/de/C_FILES/application.de.po</code> <code>src/locale/de/C_FILES/website.de.po</code></td>
<td>Using both component and language name within filename.</td>
</tr>
<tr>
<td><code>locale/{\w+}/LC_MESSAGES/{\w+}/component/[a-zA-Z]+.po</code></td>
<td><code>locale/cs/C_MESSAGES/application.cs.po</code> <code>locale/cs/C_MESSAGES/website.cs.po</code> <code>locale/de/C_MESSAGES/application.de.po</code> <code>locale/de/C_MESSAGES/website.de.po</code></td>
<td>Using language in both path and filename.</td>
</tr>
<tr>
<td><code>res/values-{\w+}/strings-{\w+}/component/[a-zA-Z]+.xml</code></td>
<td><code>res/values-cs/strings-about.xml</code> <code>res/values-cs/strings-help.xml</code> <code>res/values-de/strings-about.xml</code> <code>res/values-de/strings-help.xml</code></td>
<td>Android resource strings, split into several files.</td>
</tr>
</tbody>
</table>

You can use Django Template markup in both component name and the monolingual base filename, for example:

- `[[ component ]]
- `Component filename match
- `Component filename with uppercase case first letter

**Save**
Component discovery add-on uses Weblate URL. 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 Version control system field (in Manage Settings Version control system) of each respective component. This saves time with configuration and system resources too.

### 3.11 weblate.flags.bulk

```
q
state -1 -- 10
      10 -- 20
      20 -- 30
add_flags
remove_flags
add_labels
remove_labels
```

### 3.11 update

```
NOT has:label Table6

NOT has:label Table7
```

### 3.11 weblate.flags.same_edit

```
post-create
ref: component-check_flags
```

### 3.11 VCS

```
ref: component-check_flags
```
weblate.flags.source_edit
post-create
VCS Weblate ""

weblate.flags.target_edit
post-create
VCS Weblate ""

weblate.generate.generate

filename

pre-commit
Django:
locale/{{ language_code }}.json

{
  "language": "{{ language_code }}",
  "strings": "{{ stats.all }}",
  "translated": "{{ stats.translated }}",
  "last_changed": "{{ stats.last_changed }}",
  "last_author": "{{ stats.last_author }}"
}

weblate.generate.prefill
updatedaily

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.

261
4.5 Add-on

**weblate.generate.pseudolocale**

<table>
<thead>
<tr>
<th>Source</th>
<th>Target</th>
<th>Prefix</th>
<th>Var_prefix</th>
<th>Suffix</th>
<th>Var_suffix</th>
<th>Var_multiplier</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>All strings in this translation will be overwritten</td>
<td>Variable part multi-How many times to repeat the variable part depending on the length of the source string.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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.

The strings will be generated using following pattern:

```
Fixed string prefix Variable string prefix Source string Variable string suffix Fixed string suffix
```

**Example:**

You can use this add-on to start translation to a new locale of an existing language or similar language. Once you add the translation to the component, follow to the add-on. **Example:** If you have `fr` and want to start `fr_CA` translation, simply set `fr` as the source, `fr_CA` as the target, and leave the prefix and suffix blank.

Uninstall the add-on once you have the new translation filled to prevent Weblate from changing the translations made after the copying.

---

**weblate.gettext.authors**

**pre-commit**

**PO**

```bash
# Pavel Borecki <pavel@example.com>, 2018, 2019.
# Filip Hron <filip@example.com>, 2018, 2019.
# anonymous <noreply@weblate.org>, 2019.
```

---

262
configure ALL_LINGUAS

weblate.gettext.configure

post-adddaily

configure.in configure.ac configure.ac ALL_LINGUAS

gettext

weblate.gettext.customize

width gettext width 77 --no-wrap

77 --

65535 --

-1 --

post-load

gettext:

77

MO

weblate.gettext.mo

path

pre-commit

MO PO PO

MO
POT PO (msgmerge)

weblate.gettext.msgmerge

previous msgid no_location fuzzy

msgmerge POT PO

post-update

msgmerge POT PO

Git

weblate.git.squash

squash all -- language -- file -- author

ap- pend_trailers -- RFC 822 "Co-authored-by: ...

commit_message

post-commit

JSON

weblate.json.customize

sort_keys indent style

spaces -- Spaces tabs -- Tabs

post-load
Java

weblate.properties.sort
pre-commit
Java

weblate.removal.comments
daily

weblate.removal.suggestions
daily

RESX

weblate.resx.update
post-update

Weblate
YAML 3.10.2

weblate.yaml.customize

indent YAML

width

80 -- 80
100 -- 100
120 -- 120
180 -- 180
65535 --

line_break

dos -- DOS (\r\n)
unix -- UNIX (\n)
mac -- MAC (\r)

post-load

WEBLATE_ADDONS

weblate.addons.base.BaseAddon

from django.utils.translation import gettext_lazy as _
from weblate.addons.events import EVENT_PRE_COMMIT
from weblate.addons.scripts import BaseScriptAddon
class ExamplePreAddon(BaseScriptAddon):
    # Event used to trigger the script
    events = (EVENT_PRE_COMMIT,)
    # Name of the addon, has to be unique
    name = "weblate.example.pre"
    # Verbose name and long description
    verbose = _("Execute script before commit")
    description = _("This add-on executes a script.")

    # Script to execute
    script = "/bin/true"
    # File to add in commit (for pre commit event)
    # does not have to be set
    add_file = "po/{{ language_code }}.po"
Component configuration

VCS Weblate VCS
Gulp

```sh
#!/bin/sh
gulp --gulpfile gulp-i18n-extract.js
git commit -m 'Update source strings' src/languages/en.lang.json
```

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, view while translating.
For installation tips, see Weblate Translation Memory, which is turned on by default.

Translation memory scopes

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.

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.

**Managing translation memory**

**User interface**

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.

---

WIKI: Translation memory in JSON can be imported into Weblate, TMX is provided for interoperability with other tools.

---

### Weblate

![Weblate User Interface]

**Translation memory status**

- **Number of your entries:** 0
- **Download as JSON**
- **Download as TMX**
- **Delete**

**Import translation memory**

- **File:** Choose File  |
  No file chosen

You can upload a TMX or JSON file.

**Upload**
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):

- Exports the memory into JSON
- Imports TMX or JSON files into the translation memory

settings.py

Django

mod_wsgi

Apache

Django's documentation

AKISMET_API_KEY

Weblate Akismet API

ANONYMOUS_USER_NAME

AUDITLOG_EXPIRY

Weblate

AUTH_LOCK_ATTEMPTS

Weblate
AUTO_UPDATE

3.2
3.11

Weblate
don/off

on/off

on/off

on/off

on/off

on/off

AVATAR_URL_PREFIX

AVATAR_URL_PREFIX = 'https://www.gravatar.com/
AVATAR_URL_PREFIX = 'https://www.libravatar.org/

AVATAR_URL_PREFIX

AUTH_TOKEN_VALID

AUTH_TOKEN_VALID

AUTH_PASSWORD_DAYS

AUTH_PASSWORD_DAYS

AUTH_PASSWORD_DAYS

AUTH_PASSWORD_DAYS

AUTOFIX_LIST

AUTOFIX_LIST

AUTH_PASSWORD_EXPIRY

...
safe-html

```python
AUTOFIX_LIST = (
    "weblate.trans.autofixes.whitespace.SameBookendingWhitespace",
    "weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis",
)
```

**BACKGROUND_TASKS**

**BASE_DIR**

Weblate

**BASIC_LANGUAGES**

Weblate
BORG_EXTRA_ARGS

```
borg create
```

```
BORG_EXTRA_ARGS = ["--exclude", "vcs/*"]
```

Weblate

```
borg create
```

CSP_SCRIPT_SRC, CSP_IMG_SRC, CSP_CONNECT_SRC, CSP_STYLE_SRC, CSP_FONT_SRC

```
Weblate

Content-Security-Policy

Matomo, Google, Sentry...

# Enable Cloudflare Javascript optimizations
CSP_SCRIPT_SRC = ["ajax.cloudflare.com"]
```

CHECK_LIST

```
check  Python
```

```
CHECK_LIST = ()
```

```
```

```
updatechecks
```

273
COMMENT_CLEANUP_DAYS

3.6

COMMIT_PENDING_HOURS

2.10

Component configuration: commit_pending

CONTACT_FORM

4.6

Configures how e-mail from the contact form is being sent. Choose a configuration that matches your mail server configuration.

The sender is used in as Reply-To, this is the default behaviour.

The sender is used in as From. Your mail server needs to allow sending such e-mails.

DATA_DIR

Weblate: Web Interface to crowdsource translations

VCS: Version Control System

DATABASE: database

SSH: Secure Shell

STATIC_ROOT: Django: STATIC_ROOT

The Docker container uses a separate volume for this, see Docker container volumes.

MEDIA_ROOT: Django: MEDIA_ROOT

Celery: Celery

User-uploaded fonts, see.

Dumped data for backups

Weblate uWSGI: www-data

www-data: sudo chown www-data:www-data -R $DATA_DIR

BASE_DIR: $BASE_DIR/data

BASE_DIR: Weblate
DATABASE_BACKUP

3.1

\text{"plain"}
\text{"compressed"}
\text{"none"}

Weblate

DEFAULT_ACCESS_CONTROL

3.3

$configured_settings:

ACL

Weblate

DEFAULT_AUTO_WATCH

4.5

Configures whether \textit{Automatically watch projects on contribution} should be turned on for new users. Defaults to \texttt{True}.

DEFAULT_RESTRICTED_COMPONENT

4.1

$configured_settings

DEFAULT_ADD_MESSAGE, DEFAULT_ADDON_MESSAGE, DEFAULT_COMMIT_MESSAGE, DEFAULT_DELETE_MESSAGE

Component configuration

Component configuration
DEFAULT_ADDONS

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

install_addon

DEFAULT_COMMITER_EMAIL

```
2.4 .

noreply@weblate.org
```

DEFAULT_COMMITER_NAME

```
2.4 .

Weblate
```

DEFAULT_COMMITER_EMAIL

DEFAULT_LANGUAGE

```
4.3.2 .

en
```
```
DEFAULT_MERGE_STYLE

3.4

rebase - merge

Component configuration

DEFAULT_SHARED_TM

3.2

Configures default value of and .

DEFAULT_TRANSLATION_PROPAGATION

2.5

True

Component configuration

DEFAULT_PULL_MESSAGE

Configures the default title and message for pull requests.

ENABLE_AVATARS

Gravatar -

Component configuration

ENABLE_HOOKS

Component configuration

ENABLE_HTTPS

Weblake HTTPS HTTP URL

HTTPS Django - Cookie HSTS

HTTPS URL

HTTPS Django SSL

Forwarded-Proto For-
warded

Django SSL

SECURE_PROXY_SSL_HEADER

Component configuration

SESSION_COOKIE_SECURE

CSRF_COOKIE_SECURE

SECURE_SSL_REDIRECT

SECURE_PROXY_SSL_HEADER
```
ENABLE_SHARING

GET_HELP_URL

4.5.2 URL
Weblate URL

4.3
GitLab

GETLAB_CREDENTIALS

4.3
GitLab

GITLAB_CREDENTIALS = {
    "gitlab.com": {
        "username": "weblate",
        "token": "your-api-token",
    },
    "gitlab.example.com": {
        "username": "weblate",
        "token": "another-api-token",
    },
}

GETLAB_USERNAME

GitLab

GETLAB_TOKEN

4.3
GitLab

API

GETLAB_CREDENTIALS = GitLab

GITHUB_CREDENTIALS

4.3
GitHub

GITHUB_CREDENTIALS = {
    "api.github.com": {
        "username": "weblate",
        "token": "your-api-token",
    },
}
"github.example.com": {
    "username": "weblate",
    "token": "another-api-token",
},

GITHUB_USERNAME

GitHub

GITHUB_CREDENTIALS

GITHUB_TOKEN

GitHub

API

GITHUB_CREDENTIALS

GOOGLE_ANALYTICS_ID

Google

ID

HIDE_REPO_CREDENTIALS

Web

URL

https://user:password@git.example.com/repo.git
https://git.example.com/repo.git

VCS

HIDE_VERSION

4.3.1

IP_BEHIND_REVERSE_PROXY

2.14

True

IP_PROXY_HEADER

IP

279
IP_PROXY_HEADER

2.14 [IP_BEHIND_REVERSE_PROXY] IP_PROXY_OFFSET

IP_PROXY_OFFSET

2.14 [SECURE_PROXY_SSL_HEADER] IP_BEHIND_REVERSE_PROXY IP_PROXY_OFFSET

LEGAL_URL

3.5 [Weblate URL]

Weblate [LEGAL_URL] = "https://weblate.org/terms/"

PRIVACY_URL

LICENSE_EXTRA

[LICENSE_EXTRA] = [
   "AGPL-3.0",
   "GNU Affero General Public License v3.0",
   "https://www.gnu.org/licenses/agpl-3.0-standalone.html",
]
### LICENSE_FILTER

**AGPL-3.0**

**GPL-3.0-or-later**

```python
LICENSE_FILTER = set()
```

### LICENSE_REQUIRED

Component configuration

### LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH

* 10

```python
False
```

### LOCALIZE_CDN_URL / LOCALIZE_CDN_PATH

**Hosted Weblate**

https://weblate-cdn.com/

**JavaScript CDN**

**LOCALIZE_CDN_URL**

**LOCALIZE_CDN_PATH**

Weblate
LOGIN_REQUIRED_URLS

LOGIN_REQUIRED_URLS = {r"/(.*)$",)
REST_FRAMEWORK["DEFAULT_PERMISSION_CLASSES"] = [
    "rest_framework.permissions.IsAuthenticated"
]

LOGIN_REQUIRED_URLS_EXCEPTIONS

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
)

MATOMO_SITE_ID

MATOMO_SITE_ID = 1
MATOMO_URL = "https://example.matomo.cloud/"
MT_SERVICES

When using Docker container, this configuration is automatically generated based on provided API keys, see Machine translation settings.

```
MT_SERVICES = {
    "weblate.machinery.apertium.ApertiumAPYTranslation",
    "weblate.machinery.deepl.DeepLTranslation",
    "weblate.machinery.glosbe.GlosbeTranslation",
    "weblate.machinery.google.GoogleTranslation",
    "weblate.machinery.libretranslate.LibreTranslateTranslation",
    "weblate.machinery.microsoft.MicrosoftCognitiveTranslation",
    "weblate.machinery.microsoftterminology.MicrosoftTerminologyService",
    "weblate.machinery.mymemory.MyMemoryTranslation",
    "weblate.machinery.tmserver.AmagamaTranslation",
    "weblate.machinery.tmserver.TMServerTranslation",
    "weblate.machinery.yandex.YandexTranslation",
    "weblate.machinery.weblatetm.WeblateTranslation",
    "weblate.memory.machine.WeblateMemory",
}
```

MT_APERTIUM_APY

Apertium-APy URL: https://wiki.apertium.org/wiki/ Apertium-apy

MT_AWS_ACCESS_KEY_ID

Amazon ID:

MT_AWS_SECRET_ACCESS_KEY

API Key:
MT_AWS_REGION

Amazon

MT_BAIDU_ID

Baidu Zhiyun API ID: https://api.fanyi.baidu.com/api/trans/product/index

Baidu API machine translation

MT_BAIDU_SECRET

Baidu Zhiyun API Key: https://api.fanyi.baidu.com/api/trans/product/index

Baidu API machine translation

MT_DEEPL_API_URL


XXX auth_key "Bonjour" as JSON URL 3

DeepL API

MT_DEEPL_KEY

DeepL API Key: https://www.deepl.com/pro.html

DeepL API
MT_LIBRETRANSLATE_API_URL

4.7.1 API URL

LibreTranslate API URL

Requires an API key to use outside of the website.

Mirrors are documented on the LibreTranslate GitHub repository, some of which can be used without authentication:

https://github.com/LibreTranslate/LibreTranslate#user-content-mirrors

MT_LIBRETRANSLATE_KEY

4.7.1 API Key

API key for the LibreTranslate instance specified in MT_LIBRETRANSLATE_API_URL.

MT_GOOGLE_KEY

Google Translate API v2 API register at https://cloud.google.com/translate/docs

Google Translate

MT_GOOGLE_CREDENTIALS

Google API v3 JSON OS https://cloud.google.com/docs/authentication/getting-started

MT_GOOGLE_PROJECT

Google API v3 ID https://cloud.google.com/appengine/docs/standard/nodejs/building-app/creating-project

Google Translate API V3 (Advanced)

MT_MICROSOFT_BASE_URL

"Base URLs" Azure api.cognitive.microsofttranslator.com

Azure China api.translator.azure.cn
**MT_NETEASE_KEY**

NetEase Sight API App key
https://sight.youdao.com/

NetEase Sight API machine translation

**MT_NETEASE_SECRET**

NetEase Sight API App secret
https://sight.youdao.com/

NetEase Sight API machine translation

**MT_TMSERVER**

tmserver URL

**MT_YANDEX_KEY**

Yandex Translate API API key
https://yandex.com/dev/translate/

Yandex Translate machine translation

**MT_YOUDAO_ID**

Youdao Zhiyun API ID
https://ai.youdao.com/product-fanyi-text.s

Youdao Zhiyun API machine translation

**MT_YOUDAO_SECRET**

Youdao Zhiyun Client secret
https://ai.youdao.com/product-fanyi-text.s

Youdao Zhiyun API machine translation

**MT_SAP_BASE_URL**

SAP Translation Hub API URL

SAP Translation Hub
MT_SAP_SANDBOX_APIKEY

API key

SAP Translation Hub

MT_SAP_USERNAME

SAP

SAP Translation Hub

MT_SAP_PASSWORD

SAP

SAP Translation Hub

MT_SAP_USE_MT

True or False

SAP Translation Hub

NEARBY_MESSAGES

DEFAULT_PAGE_LIMIT

4.7

PAGURE_CREDENTIALS

4.3.2

Pagure

Pagure

Pagure

Pagure

Pagure

PAGURE_USERNAME and

PAGURE_TOKEN

PAGURE_CREDENTIALS = {
    "pagure.io": {
        "username": "weblate",
        "token": "your-api-token",
    },
    "pagure.example.com": {
        "username": "weblate",
        "token": "another-api-token",
    },
}
PAGURE_USERNAME

4.3.2

Pagure

PAGURE_CREDENTIALS

Pagure

PAGURE_TOKEN

4.3.2

Pagure

Pagure API

PRIVACY_URL

4.8.1

Weblate

PRIVACY_URL = "https://weblate.org/terms/"

LEGAL_URL

RATELIMIT_ATTEMPTS

3.2

RATELIMIT_WINDOW

RATELIMIT_LOCKOUT

RATELIMIT_WINDOW

3.2

5

RATELIMIT_ATTEMPTS

RATELIMIT_LOCKOUT

RATELIMIT_ATTEMPTS

RATELIMIT_LOCKOUT
RATELIMIT_LOCKOUT

3.2sec.

600sec.

600sec.

3.2sec.

RATELIMIT_ATTEMPTS

RATELIMIT_WINDOW

REGISTRATION_ALLOW_BACKENDS

4.1

REGISTRATION_OPEN

REGISTRATION_ALLOW_BACKENDS

REGISTRATION_EMAIL_MATCH

REGISTRATION_CAPTCHA

2.17

REGISTRATION_EMAIL_MATCH

REGISTRATION_OPEN

LDAP

REGISTRATION_ALLOW_BACKENDS

REGISTRATION_EMAIL_MATCH
REPOSITORY_ALERT_THRESHOLD

REQUIRE_LOGIN

SENTRY_DSN

SESSION_COOKIE_AGE_AUTHENTICATED

SIMPLIFY_LANGUAGES

SITE_DOMAIN

# Production site with domain name
SITE_DOMAIN = "weblate.example.com"

# Local development with IP address and port
SITE_DOMAIN = "127.0.0.1:8000"
SITE_TITLE

Web

SPECIAL_CHARS

SPECIAL_CHARS = ("\t", "\n", "\u00a0", "...")

SINGLE_PROJECT

SINGLE_PROJECT = "test"

SSH_EXTRA_ARGS

SSH_EXTRA_ARGS = "-oKexAlgorithms=+diffie-hellman-group1-sha1"

The string is evaluated by shell, so make sure to quote any whitespace and special characters.

STATUS_URL

Weblate URL
SUGGESTION_CLEANUP_DAYS

3.2.1 None

UPDATE_LANGUAGES

4.3.2 setuplang

URL_PREFIX

Weblate WSGI WSGIScriptAlias

URL_PREFIX = "/translations"

VCS_BACKENDS

VCS_BACKENDS = ("weblate.vcs.git.GitRepository",)

VCS_CLONE_DEPTH

Weblate fatal: protocol error: expected old/new/ref, got 'shallow <commit hash>'
VCS_CLONE_DEPTH = 0

WEBLATE_ADDONS

```
WEBLATE_ADDONS = {
    # Built-in add-ons
    "weblate.addons.gettext.GenerateMoAddon",
    "weblate.addons.gettext.UpdateLinguasAddon",
    "weblate.addons.gettext.UpdateConfigureAddon",
    "weblate.addons.gettext.MsgmergeAddon",
    "weblate.addons.gettext.GettextCustomizeAddon",
    "weblate.addons.gettext.GettextAuthorComments",
    "weblate.addons.cleanup.CleanupAddon",
    "weblate.addons.consistency.LanguageConsistencyAddon",
    "weblate.addons.discovery.DiscoveryAddon",
    "weblate.addons.flags.SourceEditAddon",
    "weblate.addons.flags.TargetEditAddon",
    "weblate.addons.flags.SameEditAddon",
    "weblate.addons.flags.BulkEditAddon",
    "weblate.addons.generate.GenerateFileAddon",
    "weblate.addons.json.JSONCustomizeAddon",
    "weblate.addons.properties.PropertiesSortAddon",
    "weblate.addons.removal.RemoveComments",
    "weblate.addons.removal.RemoveSuggestions",
    "weblate.addons.removal.RemoveSuggestionSortAddon",
    "weblate.addons.git.GitSquashAddon",
    "weblate.addons.yaml.YAMLCustomizeAddon",
    "weblate.addons.cdn.CDNJSAddon",
    # Add-on you want to include
    "weblate.addons.example.ExampleAddon",
}
```
WEBLATE_GPG_IDENTITY

Weblate ᗕ Git _gitlab.com  IDENTITY:

```
WEBLATE_GPG_IDENTITY = "Weblate <weblate@example.com>"
```

GnuPG  DATA_DIR

```
home/.gnupg
```

GnuPG  DATA_DIR

```
```

WEBSITE_REQUIRED

Defines whether ᗕ Web ᗕ has to be specified when creating a project. Turned on by default as that suits public server setups.

```
WEBLATE_GPG_IDENTITY

import os
import platform
from logging.handlers import SysLogHandler

# Title of site to use
SITE_TITLE = "Weblate"

# Site domain
SITE_DOMAIN = ""

# Whether site uses https
ENABLE_HTTPS = False

# Debug 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_
        "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_
            "connect_timeout": 28800,
            # Persistent connections
            "CONN_MAX_AGE": 0,
            # Disable server-side cursors, might be needed with pgbouncer
            "DISABLE_SERVER_SIDE_CURSORS": False,
        }
    }
}

BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))

# Data directory
DATA_DIR = os.path.join(BASE_DIR, "data")

# Local time zone for this installation. Choices can be found here:
# http://en.wikipedia.org/wiki/List_of_tz_zones_by_name
# although not all choices may be available on all operating systems.
# In a Windows environment this must be set to your system time zone.
TIME_ZONE = "UTC"

# Language code for this installation. All choices can be found here:
# http://www.i18nguy.com/unicode/language-identifiers.html
LANGUAGE_CODE = "en-us"

LANGUAGES = {
    ("ar", "العربية"),
    ("az", "Azərbaycan"),
    ("be", "Беларуская"),
    ("bg", "Bългарски"),
    ("br", "Brezhoneg"),
    ("ca", "Català"),
    ("cs", "Čeština"),
    ("da", "Dansk"),
    ("de", "Deutsch"),
    ("en", "English"),
}
SITE_ID = 1

# If you set this to False, Django will make some optimizations so as not
# to load the internationalization machinery.
USE_I18N = True

# If you set this to False, Django will not format dates, numbers and
# calendars according to the current locale.
USE_L10N = True

# If you set this to False, Django will not use timezone-aware datetimes.
USE_TZ = True

# Type of automatic primary key, introduced in Django 3.2
DEFAULT_AUTO_FIELD = "django.db.models.AutoField"

# URL prefix to use, please see documentation for more details
URL_PREFIX = ""

# Absolute filesystem path to the directory that will hold user-uploaded_-
# files.
MEDIA_ROOT = os.path.join(DATA_DIR, "media")

# URL that handles the media served from MEDIA_ROOT. Make sure to use a
# trailing slash.
MEDIA_URL = f"{URL_PREFIX}/media/"

# Absolute path to the directory static files should be collected to.
# Don't put anything in this directory yourself; store your static files
# in apps' "static/" subdirectories and in STATICFILES_DIRS.
STATIC_ROOT = os.path.join(DATA_DIR, "static")
# URL prefix for static files.
STATIC_URL = f"/{URL_PREFIX}/static/"

# Additional locations of static files
STATICFILES_DIRS = (  
    # Put strings here, like "/home/html/static" or "C:/www/django/static".  
    # Always use forward slashes, even on Windows.  
    # Don't forget to use absolute paths, not relative paths.  
)

# List of finder classes that know how to find static files in  
# various locations.
STATICFILES_FINDERS = (  
    "django.contrib.staticfiles.finders.FileSystemFinder",  
    "django.contrib.staticfiles.finders.AppDirectoriesFinder",  
    "compressor.finders.CompressorFinder",  
)

# Make this unique, and don't share it with anybody.  
# You can generate it using weblate/examples/generate-secret-key
SECRET_KEY = 

TEMPLATES = [  
    {  
        "BACKEND": "django.template.backends.django.DjangoTemplates",  
        "OPTIONS": {  
            "context_processors": [  
                "django.contrib.auth.context_processors.auth",  
                "django.template.context_processors.debug",  
                "django.template.context_processors.i18n",  
                "django.template.context_processors.request",  
                "django.template.context_processors.csrf",  
                "django.contrib.messages.context_processors.messages",  
                "weblate.trans.context_processors.weblate_context",  
            ],  
            "APP_DIRS": True,  
        },  
    }  
]

# GitHub username and token for sending pull requests.  
# Please see the documentation for more details.
GITHUB_USERNAME = None
GITHUB_TOKEN = None

# GitLab username and token for sending merge requests.  
# Please see the documentation for more details.
GITLAB_USERNAME = None
GITLAB_TOKEN = None

# Authentication configuration
AUTHENTICATION_BACKENDS = (  
    "social_core.backends.email.EmailAuth",  
    # "social_core.backends.google.GoogleOAuth2",  
    # "social_core.backends.github.GithubOAuth2",  
    # "social_core.backends.bitbucket.BitbucketOAuth",  
    # "social_core.backends.suse.OpenSUZEOpenId",  
    # "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_KEY = ""
SOCIAL_AUTH_BITBUCKET_SECRET = ""
SOCIAL_AUTH_BITBUCKET_VERIFIED_EMAILS_ONLY = True
SOCIAL_AUTH_FACEBOOK_KEY = ""
SOCIAL_AUTH_FACEBOOK_SECRET = ""
SOCIAL_AUTH_FACEBOOK_SCOPE = ["email", "public_profile"]
SOCIAL_AUTH_FACEBOOK_PROFILE_EXTRA_PARAMS = {"fields": "id,name,email"}
SOCIAL_AUTH_GOOGLE_OAUTH2_KEY = ""
SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET = ""

# Social auth settings
SOCIAL_AUTH_PIPELINE = (
    "social_core.pipeline.social_auth.social_details",
    "social_core.pipeline.social_auth.social_uid",
    "social_core.pipeline.social_auth.auth_allowed",
    "social_core.pipeline.social_auth.social_user",
    "weblate.accounts.pipeline.store_params",
    "weblate.accounts.pipeline.verify_open",
    "weblate.accounts.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.load_extra_data",
    "weblate.accounts.pipeline.store_email",
    "weblate.accounts.pipeline.notify_connect",
    "weblate.accounts.pipeline.notify_disconnect",
    "weblate.accounts.pipeline.reauthenticate",
    "weblate.accounts.pipeline.verify_username",
    "social_core.pipeline.user.created",
    "weblate.accounts.pipeline.cleanup_next",
)

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",
    "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_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"},
    {"NAME": "django.contrib.auth.password_validation.MinimumLengthValidator", "OPTIONS": {"min_length": 10}},
    {"NAME": "django.contrib.auth.password_validation.CommonPasswordValidator"},
    {"NAME": "django.contrib.auth.password_validation.NumericPasswordValidator"},
    {"NAME": "weblate.accounts.password_validation.CharsPasswordValidator"},
    {"NAME": "weblate.accounts.password_validation.PastPasswordsValidator"},
]

# 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.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.checks",
    "weblate.formats",
    "weblate.glossary",
    "weblate.machinery",
    "weblate.trans",
    "weblate.lang",
    "weblate_language_data",
    "weblate.memory",
    "weblate.screenshots",
    "weblate.fonts",
    "weblate.accounts",
    "weblate.configuration",
    "weblate.utils",
    "weblate.vcs",
    "weblate.wladmin",
    "weblate.metrics",
    "weblate",
    # Optional: Git exporter
    "weblate.gitexport",
    # Standard Django modules
    "django.contrib.auth",
    "django.contrib.contenttypes",
    "django.contrib.sessions",
    "django.contrib.messages",
    "django.contrib.staticfiles",
    "django.contrib.admin.apps.SimpleAdminConfig",
    "django.contrib.admindocs",
    "django.contrib.sitemaps",
    "django.contrib.humanize",
    # Third party Django modules
    "social_django",
    "crispy_forms",
    "compressor",
    "rest_framework",
    "rest_framework.authtoken",
    "django_filters",
]

# Custom exception reporter to include some details
DEFAULT_EXCEPTION_REPORTER_FILTER = "weblate.trans.debug.WeblateExceptionReporterFilter"

# Default logging of Weblate messages
# to syslog in production (if available)
# - otherwise to console
# - you can also choose "logfile" to log into separate file
# after configuring it below

# Detect if we can connect to syslog
HAVE_SYSLOG = False
if platform.system() != "Windows":
    try:
        handler = SysLogHandler(address="/dev/log", facility=sysLogHandler.
            LOG_LOCAL2)
handler.close()
HAVE_SYSLOG = True
eexcept OSError:
HAVE_SYSLOG = False

if DEBUG or not HAVE_SYSLOG:
DEFAULT_LOG = "console"

else:
DEFAULT_LOG = "syslog"
DEFAULT_LOGLEVEL = "DEBUG" if DEBUG else "INFO"

# A sample logging configuration. The only tangible logging
# performed by this configuration is to send an email to
# the site admins on every HTTP 500 error when DEBUG=False.
# See http://docs.djangoproject.com/en/stable/topics/logging for
# more details on how to customize your logging configuration.
LOGGING = {
  "version": 1,
  "disable_existing_loggers": True,
  "filters": {"require_debug_false": {"()": "django.utils.log.RequireDebugFalse"}},
  "formatters": {
    "syslog": {"format": "weblate[%(process)d]: %(levelname)s
%(message)s"},
    "simple": {"format": "%a %(asctime)s %(levelname)s %(name)s
%(message)s"},
  "logfile": {"format": [%a %d %b %Y %H:%M:%S %z]
%(levelname)s
%(name)s
%(asctime)s
%(message)s"},
  "django.server": {
    "()": "django.utils.log.ServerFormatter",
    "format": [%a %d %b %Y %H:%M:%S]
%(levelname)s
%(asctime)s
%(message)s"},
  "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": {
#    "handlers": [DEFAULT_LOG],
#    "level": "DEBUG",
# },

"weblate": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},

# Logging VCS operations
webate.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.deep1.DeepLTranslation",
    # "weblate.machinery.glosbe.GlosbeTranslation",
    # "weblate.machinery.google.GoogleTranslation",
    # "weblate.machinery.googlev3.GoogleV3Translation",
    # "weblate.machinery.libretranslate.LibreTranslateTranslation",
    # "weblate.machinery.microsoft.MicrosoftCognitiveTranslation",
    # "weblate.machinery.microsoftterminology.
    # "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.saptranslatiationhub.SAPTranslationHub",
    # "weblate.machinery.youdao.YoudaoTranslation",
    # "weblate.machinery.youdao.YoudaoTranslation",
    # "weblate.memory.machine.WeblateTranslation",
    # "weblate.memory.machine.WeblateMemory",
}

# Machine translation API keys

"APERTIUM_APY": None

# DeepL API key
"DEEPL_KEY": None

# LibreTranslate

# URL of the Apertium APy server
"APERTIUM_APY": "https://apertium-apy.com",

# DeepL API key
"DEEPL_KEY": "YOUR_DEEPL_API_KEY",

# LibreTranslate
MT_LIBRETRANSLATE_API_URL = None
MT_LIBRETRANSLATE_KEY = None

# Microsoft Cognitive Services Translator API, register at
# https://portal.azure.com/
MT_MICROSOFT_COGNITIVE_KEY = None
MT_MICROSOFT_REGION = None

# ModernMT
MT_MODERNMT_KEY = None

# MyMemory identification email, see
# https://mymemory.translated.net/doc/spec.php
MT_MYMEMORY_EMAIL = None

# Optional MyMemory credentials to access private translation memory
MT_MYMEMORY_USER = None
MT_MYMEMORY_KEY = None

# Google API key for Google Translate API v2
MT_GOOGLE_KEY = None

# Google Translate API3 credentials and project id
MT_GOOGLE_CREDENTIALS = None
MT_GOOGLE_PROJECT = None

# Baidu app key and secret
MT_BAIDU_ID = None
MT_BAIDU_SECRET = None

# Youdao Zhiyun app key and secret
MT_YOUDAO_ID = None
MT_YOUDAO_SECRET = None

# Netease Sight (Jianwai) app key and secret
MT_NETEASE_KEY = None
MT_NETEASE_SECRET = None

# API key for Yandex Translate API
MT_YANDEX_KEY = None

# tmserver URL
MT_TMSERVER = None

# SAP Translation Hub
MT_SAP_BASE_URL = None
MT_SAP_SANDBOX_APIKEY = None
MT_SAP_USERNAME = None
MT_SAP_PASSWORD = None
MT_SAP_USE_MT = True

# Use HTTPS when creating redirect URLs for social authentication, see
# documentation for more details:
# 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/#csrftoken-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

# Referer only for same origin links
SECURE_REFERRER_POLICY = "same-origin"

# SSL redirect URL exemption list
SECURE_REDIRECT_EXEMPT = (r"healthz/$",)

# Session cookie
SESSION_COOKIE_AGE = 1000
SESSION_COOKIE_AGE_AUTHENTICATED = 1209600
SESSION_COOKIE_SAMESITE = "lax"

# Increase allowed upload size
DATA_UPLOAD_MAX_MEMORY_SIZE = 5000000

# Apply session cookie settings to language cookie as well
LANGUAGE_COOKIE_SECURE = SESSION_COOKIE_SECURE
LANGUAGE_COOKIE_HTTPONLY = SESSION_COOKIE_HTTPONLY
LANGUAGE_COOKIE_AGE = SESSION_COOKIE_AGE_AUTHENTICATED * 10
LANGUAGE_COOKIE_SAMESITE = SESSION_COOKIE_SAMESITE

# Some security headers
SECURE_BROWSER_XSS_FILTER = True
X_FRAME_OPTIONS = "DENY"
SECURE_CONTENT_TYPE_NOSNIFF = True

# Optionally enable HSTS
SECURE_HSTS_SECONDS = 31536000 if ENABLE_HTTPS else 0
SECURE_HSTS_PRELOAD = ENABLE_HTTPS
SECURE_HSTS_INCLUDE_SUBDOMAINS = ENABLE_HTTPS

# HTTPS detection behind reverse proxy
SECURE_PROXY_SSL_HEADER = None

# URL of login
LOGIN_URL = f"{URL_PREFIX}/accounts/login/"

# URL of logout
LOGOUT_URL = f"{URL_PREFIX}/accounts/logout/"

# Default location for login
LOGIN_REDIRECT_URL = f"{URL_PREFIX}/"

# Anonymous user name
ANONYMOUS_USER_NAME = "anonymous"

# Reverse proxy settings
IP_PROXY_HEADER = "HTTP_X_FORWARDED_FOR"
IP_BEHIND.Reverse_PROXY = False
IP_PROXY_OFFSET = 0

# Sending HTML in mails
EMAIL_SEND_HTML = True

# Subject of emails includes site title
EMAIL_SUBJECT_PREFIX = f"[{SITE_TITLE}] "

# Enable remote hooks
ENABLE_HOOKS = True

# By default the length of a given translation is limited to the length of
# the source string * 10 characters. Set this option to False to allow
# longer
LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH = True

# Use simple language codes for default language/country combinations
SIMPLIFY_LANGUAGES = True
# Render forms using bootstrap
CRISPY_TEMPLATE_PACK = "bootstrap3"

# List of quality checks
# CHECK_LIST = (  
#     "weblate.checks.same.SameCheck",
#     "weblate.checks.chars.BeginNewlineCheck",
#     "weblate.checks.chars.EndNewlineCheck",
#     "weblate.checks.chars.BeginSpaceCheck",
#     "weblate.checks.chars.EndSpaceCheck",
#     "weblate.checks.chars.DoubleSpaceCheck",
#     "weblate.checks.chars.EndStopCheck",
#     "weblate.checks.chars.EndColonCheck",
#     "weblate.checks.chars.EndQuestionCheck",
#     "weblate.checks.chars.EndEllipsisCheck",
#     "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.chars.ZeroWidthSpaceCheck",
#     "weblate.checks.render.MaxSizeCheck",
#     "weblate.checks.render.XMLValidityCheck",
#     "weblate.checks.markup.BBCodeCheck",
#     "weblate.checks.chars.NewLineCountCheck",
#     "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.checksplaceholders.PlaceholderCheck",
#     "weblate.checksplaceholders.RegexCheck",
#     "weblate.checks.duplicate.DuplicateCheck",
#     "weblate.checks.source.OptionalPluralCheck",
#     "weblate.checks.source.EllipsisCheck",
#     "weblate.checks.source.LongUntranslatedCheck",
#     "weblate.checks.format.MultipleUnnamedFormatsCheck",
#     )
# List of automatic fixups
# AUTOFIX_LIST = (
#     "weblate.checks.glossary.GlossaryCheck",
#     "weblate.checks.glossary.GlossaryCheck",
# )

# List of automatic fixups
AUTOFIX_LIST = (  
    "weblate.checks.glossary.GlossaryCheck",  
    "weblate.checks.glossary.GlossaryCheck",  
    "weblate.trans.autofixes.whitespace.SameBookendingWhitespace",  
    "weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis",  
    "weblate.trans.autofixes.chars.RemoveZeroSpace",  
    "weblate.trans.autofixes.chars.RemoveControlChars",  
)  

# List of enabled addons
WEBLATE_ADDONS = (  
    "weblate.addons.gettext.GenerateMoAddon",  
    "weblate.addons.gettext.UpdateLinguasAddon",  
    "weblate.addons.gettext.UpdateConfigureAddon",  
    "weblate.addons.gettext.MsgmergeAddon",  
    "weblate.addons.gettext.GettextCustomizeAddon",  
    "weblate.addons.gettext.GettextAuthorComments",  
    "weblate.addons.cleanup.CleanupAddon",  
    "weblate.addons.cleanup.RemoveBlankAddon",  
    "weblate.addons.consistency.LanguageConsistencyAddon",  
    "weblate.addons.discovery.DiscoveryAddon",  
    "weblate.addons.autotranslate.AutoTranslateAddon",  
    "weblate.addons.flags.SourceEditAddon",  
    "weblate.addons.flags.TargetEditAddon",  
    "weblate.addons.flags.SameEditAddon",  
    "weblate.addons.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.cleanup.RemoveBlankAddon",  
)  

# E-mail address that error messages come from.
SERVER_EMAIL = "noreply@example.com"

# Default email address to use for various automated correspondence from
# the site managers. Used for registration emails.
DEFAULT_FROM_EMAIL = "noreply@example.com"

# List of URLs your site is supposed to serve
ALLOWED_HOSTS = ["*"]

# Configuration for caching
CACHES = {
    "default": {
        "BACKEND": "django_redis.cache.RedisCache",  
        "LOCATION": "redis://127.0.0.1:6379/1",  
        "OPTIONS": {
            "CLIENT_CLASS": "django_redis.client.DefaultClient",  
            "PARSER_CLASS": "redis.connection.HiredisParser",  
        },  
        "KEY_PREFIX": "weblate",  
    },  
    "default": {
        "BACKEND": "django_redis.cache.RedisCache",  
        "LOCATION": "redis://127.0.0.1:6379/1",  
        "OPTIONS": {
            "CLIENT_CLASS": "django_redis.client.DefaultClient",  
            "PARSER_CLASS": "redis.connection.HiredisParser",  
        },  
        "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": [
        if REQUIRE_LOGIN
        else "rest_framework.permissions.IsAuthenticatedOrReadOnly"
    ],
    "DEFAULT_AUTHENTICATION_CLASSES": {
        "rest_framework.authentication.TokenAuthentication",
        "rest_framework.authentication.BearerAuthentication",
        "rest_framework.authentication.SessionAuthentication",
    },
    "DEFAULT_THROTTLE_CLASSES": {
        "weblate.api.throttling.UserRateThrottle",
        "weblate.api.throttling.AnonRateThrottle",
    },
    "DEFAULT_THROTTLE_RATES": {"anon": "100/day", "user": "5000/hour"},
    "DEFAULT_PAGINATION_CLASS": "rest_framework.pagination.PageNumberPagination",
    "PAGE_SIZE": 20,
    "VIEW_DESCRIPTION_FUNCTION": "weblate.api.views.get_view_description",
    "UNAUTHENTICATED_USER": "weblate.auth.models.get_anonymous",
}

# Fonts CDN URL
FONTS_CDN_URL = None

# Django compressor offline mode
COMPRESS_OFFLINE = False
# COMPRESS_OFFLINE_CONTEXT = [
#    {"fonts_cdn_url": FONTS_CDN_URL, "STATIC_URL": STATIC_URL, "LANGUAGE_"BIDI": True},
#    {"fonts_cdn_url": FONTS_CDN_URL, "STATIC_URL": STATIC_URL, "LANGUAGE_"BIDI": False},
#]

# Require login for all URLs
if REQUIRE_LOGIN:
    LOGIN_REQUIRED_URLS = (r"/(.*)$",)

# In such case you will want to include some of the exceptions
# LOGIN_REQUIRED_URLS_EXCEPTIONS = (r"/(.*)$",)
# rf"{URL_PREFIX}/api/(.*)$", # Allowing access to API
# rf"{URL_PREFIX}/js/i18n/$", # JavaScript localization
# rf"{URL_PREFIX}/contact/$", # Optional for contact form
# rf"{URL_PREFIX}/legal/(.*)$", # Optional for legal app
#
# Silence some of the Django system checks
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.send_mails": {"queue": "notify"},
    "weblate.accounts.tasks.notify_*": {"queue": "notify"},
    "weblate.utils.tasks.settings_backup": {"queue": "backup"},
    "weblate.utils.tasks.database_backup": {"queue": "backup"},
    "weblate.wladmin.tasks.backup_service": {"queue": "backup"},
    "weblate.memory.tasks.*": {"queue": "memory"},
}

# Enable plain database backups
DATABASE_BACKUP = "plain"

# Enable auto updating
AUTO_UPDATE = False

# PGP commits signing
WEBLATE_GPG_IDENTITY = None

# Third party services integration
MATOMO_SITE_ID = None
MATOMO_URL = None
GOOGLE_ANALYTICS_ID = None
SENTRY_DSN = None
SENTRY_ENVIRONMENT = SITE_DOMAIN
AKISMET_API_KEY = None
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).

**Invoking management commands**

As mentioned before, invocation depends on how you installed Weblate.

If using virtualenv for Weblate, you can either specify the full path to weblate, or activate the virtualenv prior to invoking it:

```
# Direct invocation
~/weblate-env/bin/weblate
# Activating virtualenv adds it to search path
. ~/weblate-env/bin/activate
weblate
```

If you are using source code directly (either from a tarball or Git checkout), the management script is ./manage.py available in the Weblate sources. To run it:

```
python ./manage.py list_versions
```

If you've installed Weblate using the pip or pip3 installer, or by using the ./setup.py script, the weblate is installed to your path (or virtualenv path), from where you can use it to control Weblate:

```
weblate list_versions
```

For the Docker image, the script is installed like above, and you can run it using docker exec:

```
docker exec --user weblate <container> weblate list_versions
```

For docker-compose the process is similar, you just have to use docker-compose exec:

```
docker-compose exec --user weblate weblate --user 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
```

---

### add_suggestions

```
weblate add_suggestions <project> <component> <language> <file>
```

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).

```
weblate --author michal@cihar.com add_suggestions weblate application cs /tmp/suggestions-cs.po
```
auto_translate

weblate auto_translate <project> <component> <language>

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 \underline{\textit{inconsistent}}).

--add
Automatically add language if a given translation does not exist.

--mt MT
Use machine translation instead of other components as machine translations.

--threshold THRESHOLD
Similarity threshold for machine translation, defaults to 80.

--mode MODE
Specify translation mode, default is translate but fuzzy or suggest can be used.

weblate auto_translate --user nijel --inconsistent --source weblate/application weblate website cs

celery_queues

weblate celery_queues

Displays length of Celery task queues.

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.

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.
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.

---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.
---

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.

---

cleanup_ssh_keys

-weblate cleanup_ssh_keys

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.

---

createadmin

-weblate createadmin

Creates an admin account with a random password, unless it is specified.

--password PASSWORD
Provides a password on the command-line, to not generate a random one.

--no-password
Do not set password, this can be useful with --update.

--username USERNAME
Use the given name instead of admin.

--email USER@EXAMPLE.COM
Specify the admin e-mail address.

--name
Specify the admin name (visible).

--update
Update the existing user (you can use this to change passwords).

2.9 Added parameters --username, --email, --name and --update.
dump_memory

weblate dump_memory

Export a JSON file containing Weblate Translation Memory content.

dumpuserdata

weblate dumpuserdata <file.json>

Dumps userdata to a file for later use by importuserdata.

This comes in handy when migrating or merging Weblate instances.

import_demo

weblate import_demo

Creates a demo project with components based on <https://github.com/WeblateOrg/demo>.
This can be useful when developing Weblate.

import_json

weblate import_json <json-file>

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.

The parameters --ignore and --update are there to deal with already imported components. Example of JSON file:

```json
[
  {
    "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"
  }
]
```
import_memory

weblate import_memory <file>

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.

TMX  Weblate

import_project

weblate import_project <project> <gitrepo> <branch> <filemask>

The import_project command is now based on the 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.

Documentation: {{ component }}

--base-file-template TEMPLATE

{{ component }}/res/values/string.xml

--new-base-template TEMPLATE

{{ component }}/ts/en.ts

--file-format FORMAT

You can also specify the file format to use (see Component configuration), 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
  '*/**.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'
```

**Note:**
More detailed examples can be found in the starting chapter, alternatively you might want to use `import_json`.
importuserdata

`weblate importuserdata <file.json>`
Imports user data from a file created by `dumpuserdata`

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
```

install_addon

```
3.2 NEW.
```

`weblate install_addon --addon ADDON <project|project/component>`

--addon ADDON
Name of the add-on to install. For example `weblate.gettextcustomize`.

--configuration CONFIG

--update

You can either define which project or component to install the add-on in (for example `weblate/application`), or use --all to include all existing components.
To install `gettext` for all components:

```
weblate install_addon --addon weblate.gettext.customize --config '{"width =>": -1}' --update --all
```

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].

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.

list_versions

weblate list_versions
Lists all Weblate dependencies and their versions.

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.

Note: 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.

lock_translation

weblate lock_translation <project|project/component>
Prevents further translation of a component.

Note: 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.

unlock_translation

move_language

weblate move_language source target
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.

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.
pushgit

```bash
weblate pushgit <project|project/component>
```

---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.

---

Weblate pushes changes automatically if `Component configuration` is turned on, which is the default.

unlock_translation

```bash
weblate unlock_translation <project|project/component>
```

Unlocks a given component, making it available for translation.

---

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.

---

lock_translation

setupgroups

```bash
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 `Component configuration`.

---

setuplang

```bash
weblate setuplang
```

Updates list of defined languages in Weblate.

---no-update
Turns off automatic updates of existing languages (only adds new ones).
updatechecks

`weblate updatechecks <project|project/component>`

Updates all checks for all strings.

**Note:** 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.

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.

**Note:** Usually it is better to configure hooks in the repository to trigger `updategit`, instead of regular polling by `updategit`.

4.0

Web

Manage

Post announcement
Translations will be used only if they reach 60%.

Post announcement

Message

You can use Markdown and mention users by @username.

Category

Info (light blue)

Category defines color used for the message.

Expiration date

mm/dd/yyyy

The message will be not shown after this date. Use it to announce string freeze and translation deadline for next release.

Notify users

The message is shown for all translations within the project, until its given expiry, or permanently until it is deleted.

Add
Add Announcement

Required fields are marked in bold.

Message: 
Translations will be used only if they reach 40%.

You can use Markdown and mention users by @username.

Project: Webtranslate

Component: 

Language: 

Category: Info (lightblue)
Category defines color used for the message.

Notify users

The message will be not shown after this date. Use it to announce string freeze and translation deadline for next release.

To days: Today

Save and add another  Save and continue editing  SAVE
1. Define **Automatic component list assignment** with `^.*$` as regular expression in both the project and the component fields, as shown on this image:
Weblate administration

Add Component list

Required fields are marked in bold.

Component list name: All components

URL slug: all-components

Show on dashboard
When enabled this component list will be shown as a tab on the dashboard

Components:
Available components

Choose all

INSTALLED_APPS

weblate.gitexport

PROJECT REGULAR EXPRESSION
COLUMN REGULAR EXPRESSION
SLIDED

*.*

ADD ANOTHER AUTOMATIC COMPONENT LIST ASSIGNMENT

SAVE

1. settings.py

INSTALLED_APPS += ("weblate.gitexport",)

2. weblate migrate

323
Weblate URL Component configuration

```sh
git clone https://example.org/git/weblate/main/
```

Access control

```
REQUIRE_LOGIN
```

User profile API

```
```
1. Add `weblate.legal` to installed apps in `settings.py`:

```python
INSTALLED_APPS += ("weblate.legal",)

# Optional:
# Social auth pipeline to confirm TOS upon registration/subsequent sign in
SOCIAL_AUTH_PIPELINE += ("weblate.legal.pipeline.tos_confirm",)

# Middleware to enforce TOS confirmation of signed in users
MIDDLEWARE += [
    "weblate.legal.middleware.RequireTOSMiddleware",
]
```

2. `weblate migrate`

3. `weblate/legal/templates/legal/`

   Weblate UI

   ENABLE_AVATARS

   Weblate UI features:

   - Gravatar
   - Libravatar

   AVATAR_URL_PREFIX
   ENABLE_AVATARS

Akismet

1. `akismet` Python

2. Akismet API

3. `AKISMET_API_KEY` `WEBLATE_AKISMET_API_KEY` Docker

   Akismet features:

   IP

   Enable AKISMET_API_KEY WEBLATE_AKISMET_API_KEY
1. Turn on `WEBLATE_GPG_IDENTITY`. (Weblate will generate a GnuPG key when needed and will use it to sign all translation commits.)

2. Alternatively you can also import existing keys into Weblate, just set `HOME=$DATA_DIR/home` when invoking gpg.

3.2: `WEBLATE_GPG_IDENTITY`

3.4.6: `WEBLATE_RATELIMIT_WINDOW` `WEBLATE_RATELIMIT_ATTEMPTS` `WEBLATE_RATELIMIT_CONTACT_ATTEMPTS` `WEBLATE_RATELIMIT_LOCKOUT` `WEBLATE_RATELIMIT_TRANSLATE_ATTEMPTS`
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.

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.

Creating a Python module

If you are not familiar with Python, you might want to look into Python For Beginners, explaining the basics and pointing to further tutorials.

To write some custom Python code (called a module), a place to store it is needed, either in the system path (usually something like /usr/lib/python3.7/site-packages/) or in the Weblate directory, which is also added to the interpreter search path.

Better yet, turn your customization into a proper Python package:
1. Create a folder for your package (we will use weblate_customization).
2. Within it, create a setup.py file to describe the package:

```python
from setuptools import setup

setup(
    name="weblate_customization",
    version="0.0.1",
    author="Your name",
    author_email="yourname@example.com",
    description="Sample Custom check for Weblate.",
    license="GPLv3+",
    keywords="Weblate check example",
)```
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.

### Changing the logo

1. **Django**: Creating a Python module

   Branding appears in the following files:
   - `/webseite.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`:

   ```python
   INSTALLED_APPS = (
       # Add your customization as first
       "weblate_customization",
       # Weblate apps are here...
   )
   ```

3. Run `weblate collectstatic --noinput`, to collect static files served to clients.

   **Note:**

   How to manage static files (e.g. images, JavaScript, CSS)

### Weblate

1. Python: Creating a Python module

2. Python: `WEBLATE_ADDONS`/`CHECK_LIST`/`AUTOFIX_LIST`:

   ```python
   # Checks
   CHECK_LIST += ("weblate_customization.checks.FooCheck",)
   
   # Autofixes
   AUTOFIX_LIST += ("weblate_customization.autofix.FooFixer",)
   
   # Add-ons
   WEBLATE_ADDONS += ("weblate_customization.addons.ExamplePreAddon",)
   ```

**Note:**
Weblate

Web version: 4.11 — 3f413ebe1a0d81a751e9b5b3bae88b69be6a0c7f
Support status: Community support

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

Django

SSH: alerts
Weblate administration

Add Project

Required fields are marked in bold.

Project name: wwwlib.org
Display name

URL slug: weblate.org
Name used in URLs and filenames.

Project website: https://weblate.org/
Main version of translated project.

Translation instructions: https://weblate.org/contribute/

You can use Markdown and mention users @username.

- Set "Language-Team" header
  Let Weblate update the "Language-Team" file header of your project.

- Use shared translation memory
  Uses the pool of shared translations between projects.

- Contribute to shared translation memory
  Contributes to the pool of shared translations between projects.

Access control:

Protected
How to restrict access to this project is detailed in the documentation.

- Enable reviews
  Requires dedicated reviewers to approve translations.

- Enable source reviews
  Requires dedicated reviewers to approve source strings.

- Enable hooks
  Whether to allow updating this repository by remote hooks.

Language aliases:

Comma-separated list of language code mappings, for example: en-US,es,MX,es

Save and add another  Save and continue editing  SAVE

Project configuration
<table>
<thead>
<tr>
<th>Department code</th>
<th>Name</th>
<th>Address</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>333</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component configuration</td>
<td>ID</td>
<td>334</td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td>----</td>
<td>-----</td>
<td></td>
</tr>
</tbody>
</table>
Component configuration

Weblate

Weblate Libre

Weblate URL

Weblate SSH URL

3.8
Weblate

Weblate 4.5.2

Weblate

Weblate: <https://weblate.org/discover/>
3. Confirm the service activation in your Weblate and turn on the discovery listing in your Weblate management page using *Enable discovery* button:

![Weblate Management Page](image)

You can customize the listing by providing a text and image (570 x 260 pixels) at <https://weblate.org/user/>.

**Legal documents**

*Note*: 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.

**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.
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):

See https://wiki.python.org/moin/PythonSoftwareFoundationLicenseFaq#Is_Python_subject_to_export_laws.3F

Optionally used by Weblate

Used by Git

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)

Export Controls (EAR) on Open Source Software

Weblate

Contributing to Weblate modules

Weblate

Weblate

Weblate

Weblate

Weblate

Weblate

Weblate

Weblate

Weblate

Weblate

Weblate

Weblate

Weblate

Weblate

Weblate

Weblate

Weblate

Weblate

Weblate

Weblate

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.
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.

You can translate the docs.

The language definitions are in the weblate-language-data repository.
You are welcome to add missing language definitions to languages.csv, other files are generated from that file.

**Weblate**

[issue](https://github.com/WeblateOrg/Weblate/issues) | [GitHub](https://github.com/WeblateOrg/Weblate) | [discussions](https://github.com/WeblateOrg/Weblate/discussions)

**Website**

[Donate](https://donate.freedombox.org) | [Freedombox](https://freedombox.org) | [GNU Solidario (GNU Health)](https://www.gnu.org) | [BallotReady](https://ballotready.org) | [MyExpenses.Mobi](https://myexpenses.org)

Weblate contributors:

Yashiro Ccs
Cheng-Chia Tseng
Timon Reinhard
Cassidy James
Loic Dachary
Marozed
https://freedombox.org/
GNU Solidario (GNU Health)
BallotReady
Richard Nespithal
MyExpenses.Mobi
**Weblate**

1. **Weblate**: 
   ```
git clone https://github.com/WeblateOrg/weblate.git
cd weblate
   ```

2. **virtualenv**: 
   ```
virtualenv .venv
.venv/bin/activate
   ```

3. **Weblate**: 
   ```
pip install -e .
pip install -r requirements-dev.txt
   ```

4. **Weblate**: 
   ```
weblate runserver
   ```

5. **Celery Worker**: 
   ```
./weblate/examples/celery start
   ```

6. **Local testing**: 
   ```
./scripts/test-database
./manage.py test
   ```

**Docker**

**Docker**

```bash
./rundev.sh
```

```bash
Docker ps
```

**Dockerfile**

```bash
./docker-compose.yml
docker-compose up -d
```

**Adding translation projects and components**

```bash
docker-compose exec weblate machine python manage.py test
docker-compose exec weblate machine python manage.py test
```

```bash
./rundev.sh test --failfast weblate.machine
```

```bash
Docker docker ps
```
./rundev.sh logs

Docker:

PyCharm  Weblate

GitHub PyCharm IDE :
IDE virtualenv

2 PyCharm Django

Django

Languages & Frameworks > Django

Enable Django Support
Django project root: Weblate/
Settings: weblate/settings_test.py
Do not use Django test runner:
Manage py tasks:
Manage script: manage.py
Environment variables:
Folder pattern to track files: migrations

Show structure

OK Cancel Apply Help
Be careful with the property called *No reload*: It prevents the server from being reloaded live if you modify files. This allows the existing debugger breakpoints to persist, when they normally would be discarded upon reloading the server.
Coding guidelines

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.

Coding standard and linting the code

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.

pre-commit run --all

Debugging Weblate

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.
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 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.

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.

Celery

Check the Celery queue status, either in celery or using celery_queues

Look in the Celery logs for errors (see Weblate logs)

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 under the Tools tab.

These send e-mails directly, so this verifies that your SMTP configuration is correct (see ). 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.

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 .

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 helps you to notice such failures easier.

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 and get pinpointed and detailed tracebacks in the error collection tool.
Weblate

Django

Sphinx

Docker

Weblate

`Django<https://www.djangoproject.com/>`

CSS Javascript

Weblate

accounts

addons

api

Django REST framework

auth

billing

tests

fonts

formats

translate-toolkit

gitexport

lang

legal

machinery

memory

screenshots

OCR

trans

utils
class weblate.addons.base.BaseAddon(storage=None)

classmethod can_install(component, user)

configure(settings)

daily(component)

classmethod get_add_form(user, component, **kwargs)

get_settings_form(user, **kwargs)

post_add(translation)

post_commit(component)

post_push(component)

post_update(component, previous_head: str, skip_push: bool)

previous_head(str) -- HEAD

skip_push (bool) -- commit_and_push --- commit_pending

pre_commit(translation, author)

pre_push(component)

pre_update(component)

save_state()

store_post_load(translation, store)

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)

# Copyright © 2012–2022 Michal Čihař <michal@cihar.com>
# This file is part of Weblate <https://weblate.org/>
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program. If not, see <https://www.gnu.org/licenses/>.

from django.utils.translation import gettext_lazy as _

from weblate.addons.base import BaseAddon
from weblate.addons.events import EVENT_PRE_COMMIT

class ExampleAddon(BaseAddon):
    # Filter for compatible components, every key is
    # matched against property of component
    compat = {"file_format": ["po", "po-mono"]}
    # List of events add-on should receive
    events = (EVENT_PRE_COMMIT,)
    # 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

Weblate

Bootstrap\jQuery

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.

yarn scripts/yarn

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 .

## Translation topic in the Django documentation

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'

Weblate depends on third-party components for many things. In case you find a vulnerability affecting one of those components in general, please report it directly to the respective project.
Some of these are:
Django
Django REST framework
Python Social Auth

**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.

**Continuous integration**

Current test results can be found on GitHub Actions and coverage is reported on Codecov.

There are several jobs to verify different aspects:
- Unit tests
- Documentation build and external links
- Migration testing from all supported releases
- Code linting
- Setup verification (ensures that generated dist files do not miss anything and can be tested)

The configuration for the CI is in `.github/workflows` directory. It heavily uses helper scripts stored in `ci` directory. The scripts can be also executed manually, but they require several environment variables, mostly defining Django settings file to use and database connection. The example definition of that is in `scripts/test-database`:

```bash
# Simple way to configure test database from environment
# Database backend to use postgresql / mysql / mariadb
export CI_DATABASE=${1:-postgresql}

# Database server configuration
export CI_DB_USER=weblate
export CI_DB_PASSWORD=weblate
export CI_DB_HOST=127.0.0.1

# Django settings module to use
export DJANGO_SETTINGS_MODULE=weblate.settings_test
```

The simple execution can look like:

```
. scripts/test-database
./ci/run-migrate
./ci/run-test
./ci/run-docs
```

**Local testing**

To run a testsuite locally, use:

```
DJANGO_SETTINGS_MODULE=weblate.settings_test ./manage.py test
```

**Notes:** 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 `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.
```

The tests can also be executed inside developer docker container, see Docker [Weblate](https://weblate.org/docs/docker/). See Django [weblate](https://weblate.org/docs/django/) for more info on running and writing tests for Django.

---

Weblate [JSON Schema](https://weblate.org/schemas/weblate-memory.schema.json)

```json
{
  "category": 1,
  "origin": "test.tmx",
  "source": "Hello",
  "source_language": "en",
  "minLength": 1
}
```

---

352
<table>
<thead>
<tr>
<th>Table 8 - Weblate User Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>target</strong></td>
</tr>
<tr>
<td>Ahoj</td>
</tr>
<tr>
<td>minLength 1</td>
</tr>
<tr>
<td><strong>target_language</strong></td>
</tr>
<tr>
<td>ISO 639-1 / ISO 639-2 / IETF BCP 47</td>
</tr>
<tr>
<td>cs</td>
</tr>
<tr>
<td>^[^ ]+$</td>
</tr>
<tr>
<td>False</td>
</tr>
</tbody>
</table>

**Weblate User Data**

https://weblate.org/schemas/weblate-userdata.schema.json

**basic**

- **username**
- **full_name**: Weblate
- **email**: noreply@example.com
- **date_joined**: 2019-11-18T18:53:54.862Z

**profile**

- **language**: cs
- **suggested**: 1
- **translated**: 24
- **uploaded**: 1
- **hide_completed**: boolean

953
<table>
<thead>
<tr>
<th>secondary_in_zen</th>
<th>True</th>
</tr>
</thead>
<tbody>
<tr>
<td>hide_source_secondary</td>
<td>True</td>
</tr>
<tr>
<td>editor_link</td>
<td>True</td>
</tr>
<tr>
<td>translate_mode</td>
<td>True</td>
</tr>
<tr>
<td>zen_mode</td>
<td>True</td>
</tr>
<tr>
<td>special_chars</td>
<td>True</td>
</tr>
<tr>
<td>dashboard_view</td>
<td>True</td>
</tr>
<tr>
<td>dashboard_component_list</td>
<td>True</td>
</tr>
<tr>
<td>languages</td>
<td>True</td>
</tr>
<tr>
<td>secondary_languages</td>
<td>True</td>
</tr>
<tr>
<td>watched</td>
<td>True</td>
</tr>
<tr>
<td>auditlog</td>
<td>True</td>
</tr>
</tbody>
</table>
### Table 9

<table>
<thead>
<tr>
<th>address</th>
<th>IP</th>
<th>user_agent</th>
<th>timestamp</th>
<th>activity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>127.0.0.1</td>
<td>PC / Linux / Firefox</td>
<td>2019-11-18T18:58:30.845Z</td>
<td></td>
</tr>
</tbody>
</table>

Weblate

- `dumpuserdata`

Weblate

- GitHub <https://github.com/WeblateOrg/weblate/milestones>

- Docker

- GitHub

- Docker

1. `./scripts/list-translated-languages`
2. `./scripts/prepare-release`
3. `make -j 12 -C docs update-screenshots`
4. `wlc push; git remote update; git merge origin/weblate`
5. `./scripts/create-release --tag`
6. Docker
7. GitHub
8. Docker

355
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.

There might be vulnerabilities in third-party libraries which do not affect Weblate, so those are not addressed by releasing bugfix versions of Weblate.

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).

Continuous integration

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
- `translation-finder`, used to discover translatable files in the repository
- `language-data`, language definitions for Weblate
Coding guidelines

Webate

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.

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
```

CI:

The CI integration is very similar to Weblate testsuite and continuous integration.

Coding standard and linting the code

```
flake8 .pre-commit-config.yaml
```

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.

```
pre-commit run --all
```

Weblate

file formats

"Webate" "web" "translate"
Weblate 4.12

Not yet released.
Added support for Amharic in https://github.com/WeblateOrg/graphics
Added support for Burmese in https://github.com/WeblateOrg/graphics
Extended options of the https://github.com/WeblateOrg/graphics add-on.
Added ignore-all-checks flag to ignore all quality checks on a string.
Avoid https://github.com/WeblateOrg/graphics add-on to trigger failing checks.
All changes in detail.

Weblate 4.11.2

2022-3-4
Fixed corrupted MO files in the binary release.
All changes in detail.

Weblate 4.11.1

2022-3-4
Fixed missing sanitizing of arguments to Git and Mercurial - CVE-2022-23915, see GHSA-3872-f48p-pxqj for more details.
CSV
API
README
CHANGELOG
Weblate 4.11

2022 2 25

Fixes stored XSS - CVE-2022-24710, see GHSA-6jp6-9rf9-gc66 for more details.

API

ICU MessageFormat syntax

Indicate lock and contributor agreement on other occurrences listing.

Fixed updating PO files with obsolete strings or missing plurals.

Improved squash add-on compatibility with Gerrit.

Automatically initialize user languages based on the Accept-Language header.

Weblate Python 3.7

Sticky header on translations listing to improve navigation.

Java

Fixed Git operation with non-ascii branch names.

guilabel: fast-forward

Docker container automatically enables TLS/SSL for outgoing e-mail when needed.

borgbackup 1.2

Automatically translated

Weblate 4.10.1

Released on December 22nd 2021.

Documented changes introduced by upgrading to Django 4.0.

Fixed displaying of Automatically translated label.

Fixed API display of branch in components with a shared repository.

Fixed manually editing page when browsing changes.

Kashida

The Weblate Docker container now uses Python 3.10.
Weblate 4.10

Released on December 16th 2021.

DeepL Languages 

GitHub XLSX

GitHub API

API

Weblate 4.9.1

Released on November 19th 2021.

Fixed upload of monolingual files after changing template.

API

Mitigate issues with GitHub SSH key change.

Weblate 4.9

Released on November 10th 2021.

The safe-html can now understand Markdown when used with md-text.

The maxLength tag now ignores XML markup when used with xml-text.

Lowered app store title length to 30 to assist with upcoming Google policy changes.

SSH EXTRA_ARGS

ICU MessageFormat

Highlight unusual whitespace characters in the strings.

Added option to stay on translated string while editing.

BORG EXTRA_ARGS

Added API endpoint to download all component translations as a ZIP file.

Python 3.10
**Weblate 4.8.1**

Released on September 10th 2021.

- **Django**
- **JavaScript**
- **API**

Added `PRIVACY_URL` setting to add privacy policy link to the footer.
Hide member e-mail addresses from project admins.
Improved gettext PO merging in case of conflicts.
Improved glossary highlighting.
Improved `safe-html` flag behavior with XML checks.

---

**Weblate 4.8**

Released on August 21th 2021.

- **Apple stringsdict**

The exact search operator is now case-sensitive with PostgreSQL.

Documentation improvements.
Performance improvements.
Improved squash add-on compatibility with Gerrit.

Fixed squash add-on sometimes skipping parsing upstream changes.
Preserve file extension for downloads.

---

**Weblate 4.7.2**

Released on July 15th 2021.

- **API**

Fixed Git exporter URLs after a domain change.
Windows RC
Weblate 4.7.1

Released on June 30th 2021.
LibreTranslate
mi18n

Weblate 4.7

Released on June 17th 2021.
gettext PO, object-pascal-format, Object Pascal, mi18n, SAML
Fixed Gerrit integration to better handle corner cases.
Weblate now requires Django 3.2.
DeepL API

Weblate 4.6.2

Released on May 8th 2021.
RTL, Git

362
**Weblate 4.6.1**

Released on May 2nd 2021.
Remove obsolete spam protection code.
Update list of user interface languages in Docker.
Pagure

**Weblate 4.6**

Released on April 19th 2021.
The auto_translate management command has now a parameter for specifying translation mode.
Added date filtering when browsing changes.
Improved activity charts.
Sender for contact form e-mails can now be configured.
Docker API

API for creating components now automatically uses Weblate URL.
Simplified state indication while listing strings.
Renamed Argon2 to clarify the purpose.
Initial support for Scaling horizontally the Docker deployment.

**Weblate 4.5.3**

Released on April 1st 2021.
Fixed possible loss of newly added strings on replace upload.
**Weblate 4.5.2**

Released on March 26th 2021.

Lua

Ignore format strings in the check.
Allow uploading screenshot from a translate page.
Added forced file synchronization to the repository maintenance.

Several performance improvements.

---

**Weblate 4.5.1**

Released on March 5th 2021.

Fixed editing of glossary flags in some corner cases.

TMX

API PO Markdown

Improved bulk edit performance.
Fixed preserving "Needs editing" and "Approved" states for ODF files.

---

**Weblate 4.5**

Released on February 19th 2021.

gettext PO lua-format

Fixed multiple unnamed variables check behavior with multiple format flags.
Dropped mailing list field on the project in favor of generic instructions for translators.

TermBase eXchange

Strings can now be added and removed in bilingual formats as well.

Amazon Translate
Java, Java MessageFormat

Dropped specific API for glossaries as component API is used now.
Added simplified interface to toggle some of the flags.

Moved text direction toggle to get more space for the visual keyboard.
Added check whether translation matches the glossary.

Weblate 4.4.2
Released on January 14th 2021.
Fixed corruption of one distributed MO file.

Weblate 4.4.1
Released on January 13th 2021.
Fixed displaying help for project settings.
PO
Fixed cleanup add-on behavior with HTML, ODF, IDML and Windows RC formats.
CSV
Use content compression for file downloads.
Improved user experience on importing from ZIP file.
Avoid duplicate pull requests on Pagure.
Reimplemented translation editor to use native browser textarea.
Added API for add-ons.

Weblate 4.4
Released on December 15th 2020.
Weblate now requires Django 3.1.
CodeMirror

365
Syntax highlighting in translation editor for XML, HTML, Markdown and reStructuredText.

Improved support for non-standard language codes.

The user is now presented with a filtered list of languages when adding a new translation.

Extended search capabilities for changes in history.

Improved billing detail pages and Libre hosting workflow.

API

Added tasks API.

Improved display of user defined special characters.

Improved naming of ZIP downloads.

Weblate 4.3.2

Released on November 4th 2020.

Pagure

Markdown

Simplified setup of Git repositories with different default branch than "master".

Newly created internal repositories now use main as the default branch.

reStructuredText

Fixed CodeMirror display issues in some situations.

Renamed Template group to "Sources" to clarify its meaning.

Fixed GitLab pull requests on repositories with longer paths.

Weblate 4.3.1

Released on October 21st 2020.

Improve hooks compatibility with Bitbucket Server.

Reduced memory usage.
Weblate 4.3

Released on October 15th 2020.
Include user stats in the API.
Fixed component ordering on paginated pages.
Rewritten support for GitHub and GitLab pull requests.
Fixed configuration of enforced checks.
Improve documentation about built-in backups.
Vue I18n
Generic placeholders check now supports regular expressions.

Vue I18n
Added support for interacting with multiple GitLab or GitHub instances.
Extended API to cover project updates, unit updates and removals and glossaries.
Unit API now properly handles plural strings.
Component creation can now handle ZIP file or document upload.
Consolidated API response status codes.
markdown

markdown
Improved JSON, YAML and CSV formats compatibility.

markdown
Improved performance of file downloads.
Improved repository management view.
Automatically enable java-format for Android.
Python 3.9
Fixed translating HTML files under certain conditions.

markdown

Weblate 4.2.2

Released on September 2nd 2020.
JSON
Fixed login redirect for some authentication configurations.
Fixed LDAP authentication with group sync.
Git
Fixed creating local VCS components using API.
**Weblate 4.2.1**

Released on August 21st 2020.

Android

[Android](https://weblate.org/docs/4.2.1/android.html)

**XLIFF**

[XLIFF](https://weblate.org/docs/4.2.1/xliff.html)

Allow setting up localization CDN in Docker image.

**Weblate 4.2**

Released on August 18th 2020.

Improved user pages and added listing of users.

Dropped support for migrating from 3.x releases, migrate through 4.1 or 4.0.

Added exports into several monolingual formats.

Improved activity charts.

Number of displayed nearby strings can be configured.

Simplified main navigation (replaced buttons with icons).

Improved language code handling in Google Translate integration.

The Git squash add-on can generate `Co-authored-by:` trailers.

Improved query search parser.

Improved user feedback from format strings checks.

Improved performance of bulk state changes.

Added compatibility redirects after project or component renaming.

Added notifications for strings approval, component locking and license change.

Added support for ModernMT.

Allow to avoid overwriting approved translations on file upload.

Dropped support for some compatibility URL redirects.

**ECMAScript**

[ECMAScript](https://weblate.org/docs/4.2/ecmascript.html)

Removed leading dot from JSON unit keys.

**Celery**

[Celery](https://weblate.org/docs/4.2/celery.html)

Allow to configure `Content-Security-Policy` HTTP headers.

Added support for aliasing languages at project level.

New add-on to help with HTML or JavaScript localization, see JavaScript CDN.

The Weblate domain is now configured in the settings, see `SITE_DOMAIN`.

**Weblate 4.1.1**

Released on June 19th 2020.

Fixed changing autofix or add-ons configuration in Docker.

Fixed possible crash in "About" page.

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 [47].

Weblate 4.1

Released on June 15th 2020.
Added support for creating new translations with included country code.
Added support for searching source strings with screenshot.
Extended info available in the stats insights.
Improved search editing on “Translate” pages.
Improve handling of concurrent repository updates.
Include changes count in credits.
Fixed UI language selection in some cases.
Allow to whitelist registration methods with registrations closed.
Improved lookup of related terms in glossary.
Improved translation memory matches.
Group same machinery results.
Add direct link to edit screenshot from translate page.
Improved removal confirmation dialog.
Include templates in ZIP download.
Add support for Markdown and notification configuration in announcements.
Extended details in check listings.
Consistently use dismissed as state of dismissed checks.
Fix 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.
Added new quality checks: , , .
Reintroduced support for wiping translation memory.
Fixed option to ignore source checks.
Added support for configuring different branch for pushing changes.
API [48]
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 [49].
API now supports filtering of changes.
Added support for sharing glossaries between projects.
**Weblate 4.0.4**

Released on May 7th 2020.
Fixed test suite execution on some Python 3.8 environments.
Typo fixes in the documentation.
Fixed creating components using API in some cases.
Fixed JavaScript errors breaking mobile navigation.
Fixed crash on displaying some checks.
Fixed screenshots listing.
Fixed monthly digest notifications.
Fixed intermediate translation behavior with units non-existing in translation.

**Weblate 4.0.3**

Released on May 2nd 2020.
Fixed possible crash in reports.
User mentions in comments are now case insensitive.
Fixed PostgreSQL migration for non superusers.
Fixed changing the repository URL while creating component.
Fixed crash when upstream repository is gone.

**Weblate 4.0.2**

Released on April 27th 2020.
Improved performance of translation stats.
Improved performance of changing labels.
Improved bulk edit performance.
Improved translation memory performance.
Fixed possible crash on component deletion.
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.
Weblate 4.0.1

Released on April 16th 2020.
Fixed package installation from PyPI.

Weblate 4.0

Released on April 16th 2020.
Weblate now requires Python 3.6 or newer.
Added management overview of component alerts.
Added component alert for broken repository browser URLs.
Improved sign in and registration pages.
Project access control and workflow configuration integrated to project settings.
Added check and highlighter for i18next interpolation and nesting.
Added check and highlighter for percent placeholders.
Record source string changes in history.
Upgraded Microsoft Translator to version 3 API.
Reimplemented translation memory backend.
Added support for several `is:` lookups in LINTER.
Allow to make `is:*` avoid internal blacklist.
Improved comments extraction from monolingual po files.
Renamed whiteboard messages to announcements.
Fixed occasional problems with registration mails.
Fixed editing monolingual XLIFF source file.
Added support for exact matching in LINTER.
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.

Weblate 3.x series

Weblate 3.11.3

Released on March 11th 2020.
Fixed searching for fields with certain priority.
Fixed predefined query for recently added strings.
Fixed searching returning duplicate matches.
Gmail LINTER
Fixed reverting changes from the history.
Added links to events in digest notifications.
Fixed email for account removal confirmation.
Added support for Slack authentication in Docker container.
Avoid sending notifications for not subscribed languages.
Include Celery queues in performance overview.
Raised bleach dependency to address CVE-2020-6802.
Fixed listing project level changes in history.
Fixed stats invalidation in some corner cases.
Fixed searching for certain string states.
Improved format string checks behavior on missing percent.
Fixed authentication using some third party providers.

**Weblate 3.11.2**

Released on February 22nd 2020.

Fixed some strings wrongly reported as having no words.

**Weblate 3.11.1**

Released on February 20th 2020.

Documented Celery setup changes.
Improved filename validation on component creation.
Fixed minimal versions of some dependencies.
Fixed adding groups with certain Django versions.
Fixed manual pushing to upstream repository.
Improved glossary matching.

**Weblate 3.11**

Released on February 17th 2020.

Allow using VCS push URL during component creation via API.

Fixed links in notifications e-mails.
Improved look of plaintext e-mails.

Display ignored checks and allow to make them active again.

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.

Added option for...
Increased default validity of confirmation links.
Improved Matomo integration.
Fixed to correctly handle source string change.
Extended automatic updates configuration by AUTO_UPDATE.

Weblate 3.10.3

Released on January 18th 2020.
Support for translate-toolkit 2.5.0.

Weblate 3.10.2

Released on January 18th 2020.
Add lock indication to projects.
Fixed CSS bug causing flickering in some web browsers.
Improved repository matching for GitHub and Bitbucket hooks.
Fixed data migration on some Python 2.7 installations.
Allow configuration of Git shallow cloning.
Improved background notification processing.
Fixed broken form submission when navigating back in web browser.
New add-on to configure YAML formatting.
Fixed same plurals check to not fire on single plural form languages.
Fixed regex search on some fields.

Weblate 3.10.1

Released on January 9th 2020.
Extended API with translation creation.
Fixed several corner cases in data migrations.
Compatibility with Django 3.0.
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.

Weblate 3.10

Released on December 20th 2019.
Improved application user interface.
Added doublespace check.
Fixed creating new languages.
Avoid sending auditlog notifications to deleted e-mails.
Added support for Markdown in comments.
Allow placing translation instruction text in project info.
Improved support for Mercurial.
Improved Git repository fetching performance.
Add search lookup for age of string.

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.

**Weblate 3.9.1**

Released on October 28th 2019.
Remove some unneeded files from backups.
Fixed potential crash in reports.
Fixed cross database migration failure.
Added support for force pushing Git repositories.
Reduced risk of registration token invalidation.

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.

**Weblate 3.9**

Released on October 15th 2019.
Include Weblate metadata in downloaded files.

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.
Display add-on parameters on overview.
Sentry is now supported through modern Sentry SDK instead of Raven.
Changed example settings to be better fit for production environment.
Added automated backups using BorgBackup.
Split cleanup add-on for RESX to avoid unwanted file updates.
Added advanced search capabilities.
Allow users to download their own reports.
Added localization guide to help configuring components.
Added support for GitLab merge requests.
Improved display of repository status.
Perform automated translation in the background.

Weblate 3.8

Released on August 15th 2019.
Added support for simplified creating of similar components.
Added support for parsing translation flags from the XML based file formats.
Log exceptions into Celery log.
Improved look of notification e-mails.
Fixed password reset behavior.
Improved performance on most of translation pages.
Fixed listing of languages not known to Weblate.
Add support for replacing file content with uploaded.
Add support for translating non VCS based content.
Added OpenGraph widget image to use on social networks.
Added support for animated screenshots.
Improved handling of monolingual XLIFF files.
Avoid sending multiple notifications for single event.
Add support for filtering changes.
Extended predefined periods for reporting.
Added webhook support for Azure Repos.
New opt-in notifications on pending suggestions or untranslated strings.
Add one click unsubscribe link to notification e-mails.
Fixed false positives with Has been translated check.
New management interface for admins.
String priority can now be specified using flags.
Added language management views.
Add checks for Qt library and Ruby format strings.
Added configuration to better fit single project installations.
Notify about new string on source string change on monolingual translations.
Added separate view for translation memory with search capability.

**Weblate 3.7.1**

Released on June 28th 2019.
Documentation updates.
Fixed some requirements constraints.
Updated language database.
Localization updates.
Various user interface tweaks.
Improved handling of unsupported but discovered translation files.
More verbosely report missing file format requirements.

**Weblate 3.7**

Released on June 21st 2019.
Added separate Celery queue for notifications.
Use consistent look with application for API browsing.
Include approved stats in the reports.
Report progress when updating translation component.
Allow to abort running background component update.
Extend template language for filename manipulations.
Use templates for editor link and repository browser URL.
Indicate max length and current characters count when editing translation.
Refreshed landing page for new contributors.
msgmerge:

Delay opening SMTP connection when sending notifications.
Improved error logging.
Allow custom location in MO generating add-on.
Added add-ons to cleanup old suggestions or comments.
Added option to enable horizontal mode in the Zen editor.
Improved import performance with many linked components.
Fixed examples installation in some cases.

Added new horizontal stats widget.
Improved format strings check on plurals.
Added font management tool.
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.
**Weblate 3.6.1**

Released on April 26th 2019.
Improved handling of monolingual XLIFF files.
Fixed digest notifications in some corner cases.
Fixed add-on script error alert.
Fixed generating MO file for monolingual PO files.
Fixed display of uninstalled checks.
Indicate administered projects on project listing.
Allow update to recover from missing VCS repository.

**Weblate 3.6**

Released on April 20th 2019.
Add support for downloading user data.
Improved instructions for resolving merge conflicts.
Cleanup add-on is now compatible with app store metadata translations.
Configurable language code syntax when adding new translations.
Warn about using Python 2 with planned termination of support in April 2020.
Extract special characters from the source string for visual keyboard.
Extended contributor stats to reflect both source and target counts.
Admins and consistency add-ons can now add translations even if disabled for users.
Fixed description of toggle disabling Language-Team header manipulation.
Notify users mentioned in comments.
Removed file format autodetection from component setup.
Fixed generating MO file for monolingual PO files.
Added digest notifications.
Added support for muting component notifications.
Added notifications for new alerts, whiteboard messages or components.
Notifications for administered projects can now be configured.
Improved handling of three letter language codes.

**Weblate 3.5.1**

Released on March 10th 2019.
Fixed Celery systemd unit example.
Fixed notifications from HTTP repositories with login.
Fixed race condition in editing source string for monolingual translations.
Include output of failed add-on execution in the logs.
Improved validation of choices for adding new language.
Allow to edit file format in component settings.
Update installation instructions to prefer Python 3.
Performance and consistency improvements for loading translations.
Microsoft Terminology service Zeep Localization updates.

377
**Weblate 3.5**

Released on March 3rd 2019.
Improved performance of built-in translation memory.
Added interface to manage global translation memory.
Improved alerting on bad component state.
Added user interface to manage whiteboard messages.
Add-on commit message now can be configured.
Reduce number of commits when updating upstream repository.
Fixed possible metadata loss when moving component between projects.
Improved navigation in the Zen mode.
Added several new quality checks (Markdown related and URL).
Added support for app store metadata files.
Added support for toggling GitHub or Gerrit integration.
Kashida 🇪🇸
Added option to squash commits based on authors.
Improved support for XLSX file format.
Compatibility with Tesseract 4.0.
Billing add-on now removes projects for unpaid billings after 45 days.

**Weblate 3.4**

Released on January 22nd 2019.
Added support for XLIFF placeholders.
Celery can now utilize multiple task queues.
Added support for renaming and moving projects and components.
Include characters counts in reports.
Added guided adding of translation components with automatic detection of translation files.
Customizable merge commit messages for Git.
Added visual indication of component alerts in navigation.
Improved performance of loading translation files.

Kashida 🇪🇸

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.

**Weblate 3.3**

Released on November 30th 2018.
Added support for component and project removal.
Improved performance for some monolingual translations.
Added translation component alerts to highlight problems with a translation.
Expose XLIFF string resname as context when available.
Added support for XLIFF states.
Added check for non writable files in DATA_DIR.
Improved CSV export for changes.
**Weblate 3.2.2**

Released on October 20th 2018.
Remove no longer needed Babel dependency.
Updated language definitions.
Improve documentation for add-ons, LDAP and Celery.
Fixed enabling new dos-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.

**Weblate 3.2.1**

Released on October 10th 2018.
Document dependency on backports.csv on Python 2.7.
Fix running tests under root.
Improved error handling in gitexport module.
Fixed progress reporting for newly added languages.
Correctly report Celery worker errors to Sentry.
Fixed creating new translations with Qt Linguist.
Fixed occasional fulltext index update failures.
Improved validation when creating new components.
Added support for cleanup of old suggestions.

**Weblate 3.2**

Released on October 6th 2018.
Add install_addon management command for automated add-on installation.

- Added support for export and import of Excel files.
- Improve component cleanup in case of multiple component discovery add-ons.

Microsoft Terminology

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.

Users and projects can import their existing translation memories.
Better management of related strings for screenshots.
Added support for checking Java MessageFormat.
See 3.2 milestone on GitHub for detailed list of addressed issues.

**Weblate 3.1.1**

Released on July 27th 2018.
Fix testsuite failure on some setups.

**Weblate 3.1**

Released on July 27th 2018.
Upgrades from older version than 3.0.1 are not supported.
Allow to override default commit messages from settings.
Improve webhooks compatibility with self hosted environments.
Added support for Amazon Translate.
Compatibility with Django 2.1.
Django system checks are now used to diagnose problems with installation.
Removed support for soon shutdown libravatar service.
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.

**Weblate 3.0.1**

Released on June 10th 2018.
Fixed possible migration issue from 2.20.
Localization updates.
Removed obsolete hook examples.
Improved caching documentation.
Fixed displaying of admin documentation.
Improved handling of long language names.

**Weblate 3.0**

Released on June 1st 2018.
Rewritten access control.
Several code cleanups that lead to moved and renamed modules.
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.
Support for more variables in commit message templates.
Add support for providing additional textual context.
Weblate 2.x series

Weblate 2.20

Released on April 4th 2018.
Improved speed of cloning subversion repositories.
Changed repository locking to use third party library.
Added support for downloading only strings needing action.
Added support for searching in several languages at once.
New add-on to configure gettext output wrapping.
New add-on to configure JSON formatting.
Added support for authentication in API using RFC 6750 compatible Bearer authentication.
Added support for automatic translation using machine translation services.
Added support for HTML markup in whiteboard messages.
Added support for mass changing state of strings.
Translate-toolkit at least 2.3.0 is now required, older versions are no longer supported.

Added support for DeepL machine translation service.
Machine translation results are now cached inside Weblate.

Weblate 2.19.1

Released on February 20th 2018.
Fixed migration issue on upgrade from 2.18.
Improved file upload API validation.

Weblate 2.19

Released on February 15th 2018.
Fixed imports across some file formats.
Display human friendly browser information in audit log.
Added TMX exporter for files.
Various performance improvements for loading translation files.
Added option to disable access management in Weblate in favor of Django one.
Improved glossary lookup speed for large strings.
Compatibility with django_auth_ldap 1.3.0.
Configuration errors are now stored and reported persistently.
Honor ignore flags in whitespace autofixer.
Improved compatibility with some Subversion setups.

Added support for SAP Translation Hub service.
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.
Weblate 2.18

Released on December 15th 2017.
Extended contributor stats.
Improved configuration of special characters virtual keyboard.
Added support for DTD file format.
Changed keyboard shortcuts to less likely collide with browser/system ones.
Improved support for approved flag in XLIFF files.
Added support for not wrapping long strings in gettext PO files.
Added button to copy permalink for current translation.
Dropped support for Django 1.10 and added support for Django 2.0.
Removed locking of translations while translating.
Added support for adding new strings to monolingual translations.
Added support for translation workflows with dedicated reviewers.

Weblate 2.17.1

Released on October 13th 2017.
Fixed running testsuite in some specific situations.

Weblate 2.17

Released on October 13th 2017.
Weblate by default does shallow Git clones now.
Improved performance when updating large translation files.
Added support for blocking certain e-mails from registration.
Users can now delete their own comments.
Added preview step to search and replace feature.
Client side persistence of settings in search and upload forms.
Extended search capabilities.
More fine grained per project ACL configuration.
Default value of BASE_DIR has been changed.
Added two step account removal to prevent accidental removal.
Project access control settings is now editable.
Added optional spam protection for suggestions using Akismet.

Weblate 2.16

Released on August 11th 2017.
Various performance improvements.
Added support for WebExtension JSON format.
Fixed git exporter authentication.
Improved CSV import in certain situations.
Improved look of Other translations widget.
The max-length checks is now enforcing length of text in form.
Make the commit_pending age configurable per component.
Various user interface cleanups.
Fixed component/project/site wide search for translations.

**Weblate 2.15**

Released on June 30th 2017.
Show more related translations in other translations.
Add option to see translations of current string to other languages.
Use 4 plural forms for Lithuanian by default.
Fixed upload for monolingual files of different format.
Improved error messages on failed authentication.
Keep page state when removing word from glossary.
Added Perl format quality check.
Added support for rejecting reused passwords.
Extended toolbar for editing RTL languages.

**Weblate 2.14.1**

Released on May 24th 2017.
Fixed possible error when paginating search results.
Fixed migrations from older versions in some corner cases.
Fixed possible CSRF on project watch and unwatch.
The password reset no longer authenticates user.
Fixed possible CAPTCHA bypass on forgotten password.

**Weblate 2.14**

Released on May 17th 2017.
Add glossary entries using AJAX.
The logout now uses POST to avoid CSRF.
The API key token reset now uses POST to avoid CSRF.
Weblate sets Content-Security-Policy by default.
The local editor URL is validated to avoid self-XSS.
The password is now validated against common flaws by default.
Notify users about important activity with their account such as password change.
The CSV exports now escape potential formulas.
Various minor improvements in security.
Suggestion content is stored in the history.
Store important account activity in audit log.
Ask for password confirmation when removing account or adding new associations.
Show time when suggestion has been made.
There is new quality check for trailing semicolon.
Ensure that search links can be shared.
Included source string information and screenshots in the API.
Allow to overwrite translations through API upload.
Weblate 2.13.1

Released on Apr 12th 2017.
Fixed listing of managed projects in profile.
Fixed migration issue where some permissions were missing.
Fixed listing of current file format in translation download.
Return HTTP 404 when trying to access project where user lacks privileges.

Weblate 2.13

Released on Apr 12th 2017.
Fixed quality checks on translation templates.
Added quality check to trigger on losing translation.
Add option to view pending suggestions from user.
Default dashboard for unauthenticated users can be configured.
Add option to browse 25 random strings for review.
History now indicates string change.
Better error reporting when adding new translation.
Added per language search within project.
Group ACLs can now be limited to certain permissions.
The per project ACLs are now implemented using Group ACL.
Added more fine grained privileges control.
Various minor UI improvements.

Weblate 2.12

Released on Mar 3rd 2017.
Improved admin interface for groups.
Added support for Yandex Translate API.
Improved speed of site wide search.
Added project and component wide search.
Added project and component wide search and replace.
Added support for opening source files in local editor.
Added support for configuring visual keyboard with special characters.
Improved screenshot management with OCR support for matching source strings.
Default commit message now includes translation information and URL.
Added support for Joomla translation format.
Improved reliability of import across file formats.
**Weblate 2.11**

Released on Jan 31st 2017.
Include language detailed information on language page.
Mercurial backend improvements.
Added option to specify translation component priority.
More consistent usage of Group ACL even with less used permissions.
Added WL_BRANCH variable to hook scripts.
Improved developer documentation.
Better compatibility with various Git versions in Git exporter add-on.
Included per project and component stats.
Added language code mapping for better support of Microsoft Translate API.
Moved fulltext cleanup to background job to make translation removal faster.
Fixed displaying of plural source for languages with single plural form.
Improved error handling in import_project.
Various performance improvements.

**Weblate 2.10.1**

Released on Jan 20th 2017.
Do not leak account existence on password reset form (CVE-2017-5537).

**Weblate 2.10**

Released on Dec 15th 2016.
Added quality check to check whether plurals are translated differently.
Fixed GitHub hooks for repositories with authentication.
Added optional Git exporter module.
Support for Microsoft Cognitive Services Translator API.
Simplified project and component user interface.
Added automatic fix to remove control characters.
Added per language overview to project.
Added support for CSV export.
Added CSV download for stats.
Added basic API for changes and strings.
Added support for Apertium APy server for machine translations.

**Weblate 2.9**

Released on Nov 4th 2016.
Extended parameters for createadmin management command.
Extended import_.json to be able to handle with existing components.
Added support for YAML files.
Project owners can now configure translation component and project details.
Use "Watched" instead of "Subscribed" projects.
Projects can be watched directly from project page.
Added multi language status widget.
Record suggestion deletion in history.
Improved UX of languages selection in profile.
Fixed showing whiteboard messages for component.
Show source string comment more prominently.
Automatically install Gettext PO merge driver for Git repositories.
Added search and replace feature.

**Weblate 2.8**

Released on Aug 31st 2016.
Documentation improvements.
Translations.
Updated bundled JavaScript libraries.
Added list_translators management command.
Django 1.8 is no longer supported.
Fixed compatibility with Django 1.10.
Added Subversion support.
Separated XML validity check from XML mismatched tags.
Fixed API to honor HIDE_REPO_CREDENTIALS settings.
Show source change in Zen mode.
Alt+PageUp/PageDown/Home/End now works in Zen mode as well.
Add tooltip showing exact time of changes.
Add option to select filters and search from translation page.
Added UI for translation removal.
Improved behavior when inserting placeables.
Fixed auto locking issues in Zen mode.

**Weblate 2.7**

Released on Jul 10th 2016.
Removed Google web translate machine translation.
Improved commit message when adding translation.
Fixed Google Translate API for Hebrew language.
Compatibility with Mercurial 3.8.
Added import_json management command.
Correct ordering of listed translations.
Show full suggestion text, not only a diff.
Extend API (detailed repository status, statistics, …).
Testsuite no longer requires network access to test repositories.
**Weblate 2.6**

Released on Apr 28th 2016.
Fixed validation of components with language filter.
Improved support for XLIFF files.
Fixed machine translation for non English sources.
Added REST API.
Django 1.10 compatibility.
Added categories to whiteboard messages.

**Weblate 2.5**

Released on Mar 10th 2016.
Fixed automatic translation for project owners.
Improved performance of commit and push operations.
Added support for merging comments on file upload.
Added support for some GNU extensions to C printf format.
Documentation improvements.
Added support for generating translator credits.
Added support for generating contributor stats.
Site wide search can search only in one language.
Improve quality checks for Armenian.
Support for starting translation components without existing translations.
Support for adding new translations in Qt TS.
Improved support for translating PHP files.
Performance improvements for quality checks.

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.
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.
Documentation improvements.
Added support for configurable automatic group assignments.
Improved adding of new translations.

Weblate 2.4

Released on Sep 20th 2015.
Improved support for PHP files.
Ability to add ACL to anonymous user.
Improved configurability of import_project command.
Added CSV dump of history.
Avoid copy/paste errors with whitespace characters.
Added support for Bitbucket webhooks.
Tighter control on fuzzy strings on translation upload.
Several URLs have changed, you might have to update your bookmarks.
Hook scripts are executed with VCS root as current directory.
Hook scripts are executed with environment variables describing current component.
Add management command to optimize fulltext index.
Added support for error reporting to Rollbar.
Projects now can have multiple owners.
Project owners can manage themselves.
Added support for javascript-format used in gettext PO.
Support for adding new translations in XLIFF.
Improved file format autodetection.
Extended keyboard shortcuts.
Improved dictionary matching for several languages.
Improved layout of most of pages.
Support for adding words to dictionary while translating.
Added support for filtering languages to be managed by Weblate.
Added support for translating and importing CSV files.
Rewritten handling of static files.
Direct login/registration links to third-party service if that's the only one.
Commit pending changes on account removal.
Add management command to change site name.
Add option to configure default committer.
Add hook after adding new translation.
Add option to specify multiple files to add to commit.
**Weblate 2.3**

Released on May 22nd 2015.
Dropped support for Django 1.6 and South migrations.
Support for adding new translations when using Java Property files.
Allow to accept suggestion without editing.
Improved support for Google OAuth 2.0.
Added support for Microsoft .resx files.
Tuned default robots.txt to disallow big crawling of translations.
Simplified workflow for accepting suggestions.
Added project owners who always receive important notifications.
Allow to disable editing of monolingual template.
More detailed repository status view.
Direct link for editing template when changing translation.
Allow to add more permissions to project owners.
Zen

**Weblate 2.2**

Released on Feb 19th 2015.
Performance improvements.
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.

**Weblate 2.1**

Released on Dec 5th 2014.
Added support for Mercurial repositories.
Replaced Glyphicon font by Awesome.
Added icons for social authentication services.
Better consistency of button colors and icons.
Documentation improvements.
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.

**Weblate 2.0**

Released on Nov 6th 2014.
New responsive UI using Bootstrap.
Rewritten VCS backend.
Documentation improvements.
Added whiteboard for site wide messages.
Configurable strings priority.
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.

**Weblate 1.x series**

**Weblate 1.9**

Released on May 6th 2014.
Django 1.6 compatibility.
No longer maintained compatibility with Django 1.4.
Management commands for locking/unlocking translations.
Improved support for Qt TS files.
Users can now delete their account.
Avatars can be disabled.
Merged first and last name attributes.
Avatars are now fetched and cached server side.
Added support for shields.io badge.

**Weblate 1.8**

Released on November 7th 2013.
Please check manual for upgrade instructions.
Nicer listing of project summary.
Better visible options for sharing.
More control over anonymous users privileges.
Supports login using third party services, check manual for more details.
Users can login by e-mail instead of username.
Documentation improvements.
Improved source strings review.
Searching across all strings.
Better tracking of source strings.
Captcha protection for registration.

**Weblate 1.7**

Released on October 7th 2013.
Please check manual for upgrade instructions.
Support for checking Python brace format string.
Per component customization of quality checks.
Detailed per translation stats.
Changed way of linking suggestions, checks and comments to strings.
Users can now add text to commit message.
Support for subscribing on new language requests.
Support for adding new translations.

Support for Pango + Cairo Pillow
Add status badge widget.

Changes in dictionary are now logged in history.

**Weblate 1.6**

Released on July 25th 2013.
Nicer error handling on registration.
Browsing of changes.
Fixed sorting of machine translation suggestions.
Improved support for MyMemory machine translation.
Added support for Amagama machine translation.
Various optimizations on frequently used pages.
Highlights searched phrase in search results.
Support for automatic fixups while saving the message.
Tracking of translation history and option to revert it.
Added support for Google Translate API.
Added support for managing SSH host keys.
Various form validation improvements.
Various quality checks improvements.
Performance improvements for import.
Added support for voting on suggestions.
Cleanup of admin interface.
**Weblate 1.5**

Released on April 16th 2013.
Please check manual for upgrade instructions.
Added public user pages.
Better naming of plural forms.
Added support for TBX export of glossary.
Added support for Bitbucket notifications.
Activity charts are now available for each translation, language or user.
Extended options of import_project admin command.
Compatible with Django 1.5.
Avatars are now shown using libravatar.
Added possibility to pretty print JSON export.
Various performance improvements.

- 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.

**Weblate 1.4**

Released on January 23rd 2013.
Fixed deleting of checks/comments on string deletion.
Added option to disable automatic propagation of translations.
Added option to subscribe for merge failures.
Correctly import on projects which needs custom ttkit loader.
Added sitemaps to allow easier access by crawlers.
Provide direct links to string in notification e-mails or feeds.
Various improvements to admin interface.
Provide hints for production setup in admin interface.
Added per language widgets and engage page.
Improved translation locking handling.

- 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.
**Weblate 1.3**

Released on November 16th 2012.
Compatibility with PostgreSQL database backend.
Removes languages removed in upstream git repository.
Improved quality checks processing.
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.

**Weblate 1.2**

Released on August 14th 2012.
Weblate now uses South for database migration, please check upgrade instructions if you are upgrading.
Fixed minor issues with linked git repos.
New introduction page for engaging people with translating using Weblate.
Added widgets which can be used for promoting translation projects.
Added option to reset repository to origin (for privileged users).
Project or component can now be locked for translations.
Possibility to disable some translations.
Configurable options for adding new translations.
Configuration of git commits per project.
Simple antispam protection.
Better layout of main page.
Support for automatically pushing changes on every commit.
Support for e-mail notifications of translators.
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.
Checking of requirements during setup.
Documentation improvements.
Weblate 1.1

Released on July 4th 2012.
Improved several translations.
Better validation while creating component.
Added support for shared git repositories across components.
Do not necessary commit on every attempt to pull remote repo.
Added support for offloading indexing.

Weblate 1.0

Released on May 10th 2012.
Improved validation while adding/saving component.
Experimental support for Android component files (needs patched ttkit).
Updates from hooks are run in background.
Improved installation instructions.
Improved navigation in dictionary.

Weblate 0.x series

Weblate 0.9

Released on April 18th 2012.
Fixed import of unknown languages.
Improved listing of nearby messages.
Improved several checks.
Documentation updates.
Added definition for several more languages.
Various code cleanups.
Documentation improvements.
Changed file layout.
Update helper scripts to Django 1.4.
Improved navigation while translating.
Better handling of po file renames.
Better validation while creating component.
Integrated full setup into syncdb.
Added list of recent changes to all translation pages.
Check for untranslated strings ignores format string only messages.

Weblate 0.8

Released on April 3rd 2012.
Replaced own full text search with Whoosh.
Various fixes and improvements to checks.
New command updatechecks.
Lot of translation updates.
Added dictionary for storing most frequently used terms.
Added /admin/report/ for overview of repositories status.
Machine translation services no longer block page loading.
Management interface now contains also useful actions to update data.
Records log of changes made by users.
Ability to postpone commit to Git to generate less commits from single user.
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.

**Weblate 0.7**

Released on February 16th 2012.
Direct support for GitHub notifications.
Added support for cleaning up orphaned checks and translations.
Displays nearby strings while translating.
Displays similar strings while translating.
Improved searching for string.

**Weblate 0.6**

Released on February 14th 2012.
Added various checks for translated messages.
Tunable access control.
Improved handling of translations with new lines.
Added client side sorting of tables.
Please check upgrading instructions in case you are upgrading.

**Weblate 0.5**

Released on February 12th 2012.
Support for machine translation using following online services:
- Apertium
- Microsoft Translator
- MyMemory
Several new translations.
Improved merging of upstream changes.
Better handle concurrent git pull and translation.
Propagating works for fuzzy changes as well.
Propagating works also for file upload.
Fixed file downloads while using FastCGI (and possibly others).
wlc
wlc.config
wlc.main
/ ANY /api/
GET /api/
GET /api/addons/
GET /api/addons/(int:id)/
PUT /api/addons/(int:id)/
DELETE /api/addons/(int:id)/
PATCH /api/addons/(int:id)/
GET /api/changes/
GET /api/changes/(int:id)/
GET /api/component-lists/
GET /api/component-lists/(str:slug)/
POST /api/component-lists/(str:slug)/components/
PUT /api/component-lists/(str:slug)/
DELETE /api/component-lists/(str:slug)/
DELETE /api/component-lists/(str:slug)/components/(str:component_slug)/
PATCH /api/component-lists/(str:slug)/
GET /api/components/
GET /api/components/(string:project)/(string:component)/
GET /api/components/(string:project)/(string:component)/changes/
GET /api/components/(string:project)/(string:component)/file/
GET /api/components/(string:project)/(string:component)/links/
GET /api/components/(string:project)/(string:component)/lock/
GET /api/components/(string:project)/(string:component)/monolingual_base/
GET /api/components/(string:project)/(string:component)/new_template/
GET /api/components/(string:project)/(string:component)/repository/
GET /api/components/(string:project)/(string:component)/screenshots/
GET /api/components/(string:project)/(string:component)/statistics/
GET /api/components/(string:project)/(string:component)/translations/
POST /api/components/(string:project)/(string:component)/addons/
POST /api/components/(string:project)/(string:component)/links/
POST /api/components/(string:project)/(string:component)/lock/
POST /api/components/(string:project)/(string:component)/repository/
POST /api/components/(string:project)/(string:component)/translations/
PUT /api/components/(string:project)/(string:component)/
DELETE /api/components/(string:project)/(string:component)/
DELETE /api/components/(string:project)/(string:component)/links/(string:project_slug)/
PATCH /api/components/(string:project)/(string:component)/
GET /api/groups/
GET /api/groups/(int:id)/
POST /api/groups/
POST /api/groups/(int:id)/componentlists/
POST /api/groups/(int:id)/components/
POST /api/groups/(int:id)/languages/
POST /api/groups/(int:id)/projects/
GET /exports/stats/(string:project)/(string:component)/
GET /hooks/update/(string:project)/
GET /hooks/update/(string:project)/(string:component)/
POST /hooks/azure/
POST /hooks/bitbucket/
POST /hooks/gitea/
POST /hooks/gitee/
POST /hooks/github/
POST /hooks/gitlab/
POST /hooks/pagure/