# Documentation pour l’utilisateur

## 1 Documentation pour l’utilisateur

1.1 Bases de Weblate ........................................... 1
1.2 Inscription et profil utilisateur .............................. 1
1.3 Traduction à l’aide de Weblate ............................. 10
1.4 Téléchargement et télèversement des traductions .......... 20
1.5 Glossaire .................................................... 23
1.6 Contrôles de qualité et corrections ......................... 25
1.7 Recherche .................................................... 54
1.8 Flux de travail de traduction ............................... 59
1.9 Foire aux questions ......................................... 63
1.10 Formats de fichiers pris en charge ......................... 71
1.11 Intégration avec le système de contrôle de versions .... 93
1.12 API REST de Weblate .................................... 100
1.13 Client Weblate .............................................. 141
1.14 API Python de Weblate ................................... 146

## 2 Documentation pour l’administrateur

2.1 Instructions de configuration ............................... 149
2.2 Déploiements Weblate .................................... 212
2.3 Mise à niveau de Weblate ................................ 213
2.4 Sauvegarder et déplacer Weblate ......................... 222
2.5 Authentification ............................................ 228
2.6 Contrôle d’accès ............................................ 239
2.7 Projets de traduction ..................................... 248
2.8 Définitions de langue .................................... 265
2.9 Traduction en continu .................................... 268
2.10 Licence des traductions .................................. 277
2.11 Processus de traduction .................................. 279
2.12 Contrôles de qualité et corrections ....................... 285
2.13 Configuring automatic suggestions ...................... 295
2.14 Extensions ................................................ 306
2.15 Mémoire de traduction .................................. 324
2.16 Configuration .............................................. 326
2.17 Configuration d’exemple ................................ 351
2.18 Commandes de gestion ................................... 367
2.19 Annonces .................................................... 378
2.20 Liste des composants ..................................... 381
2.21 Modules Weblate optionnels .............................. 382
2.22 Personnaliser Weblate .................................... 387
2.23 Interface de gestion ..................................... 389
2.24 Getting support for Weblate ............................. 398
3 Documentation pour le contributeur

3.1 Contribuer à Weblate ......................................................................................... 403
3.2 Commencer à contribuer au code de Weblate ............................................... 404
3.3 Code source de Weblate ................................................................................. 409
3.4 Déboguer Weblate ......................................................................................... 410
3.5 Weblate internals ......................................................................................... 412
3.6 Développement de greffons ......................................................................... 413
3.7 Interface de Weblate ..................................................................................... 414
3.8 Reporting issues in Weblate ......................................................................... 416
3.9 Weblate testsuite and continuous integration ............................................... 416
3.10 Schémas de données ..................................................................................... 418
3.11 Publication de Weblate ................................................................................. 422
3.12 Sécurité et confidentialité ............................................................................. 423
3.13 Contributing to Weblate modules ................................................................. 423
3.14 À propos de Weblate ................................................................................... 424
3.15 Licence ........................................................................................................... 425

4 Historique des modifications

4.1 Weblate 4.16 ............................................................................................... 426
4.2 Weblate 4.15.2 ............................................................................................. 426
4.3 Weblate 4.15.1 ............................................................................................. 427
4.4 Weblate 4.15 ............................................................................................... 427
4.5 Weblate 4.14.2 ............................................................................................. 427
4.6 Weblate 4.14.1 ............................................................................................. 428
4.7 Weblate 4.14 ............................................................................................... 428
4.8 Weblate 4.13.1 ............................................................................................. 428
4.9 Weblate 4.13 ............................................................................................... 429
4.10 Weblate 4.12.2 ............................................................................................. 429
4.11 Weblate 4.12.1 ............................................................................................. 429
4.12 Weblate 4.12 ............................................................................................... 430
4.13 Weblate 4.11.2 ............................................................................................. 430
4.14 Weblate 4.11.1 ............................................................................................. 430
4.15 Weblate 4.11 ............................................................................................... 430
4.16 Weblate 4.10.1 ............................................................................................. 431
4.17 Weblate 4.10 ............................................................................................... 431
4.18 Weblate 4.9.1 ............................................................................................... 432
4.19 Weblate 4.9 ............................................................................................... 432
4.20 Weblate 4.8.1 ............................................................................................... 432
4.21 Weblate 4.8 ............................................................................................... 433
4.22 Weblate 4.7.2 ............................................................................................... 433
4.23 Weblate 4.7.1 ............................................................................................... 433
4.24 Weblate 4.7 ............................................................................................... 433
4.25 Weblate 4.6.2 ............................................................................................... 434
4.26 Weblate 4.6.1 ............................................................................................... 434
4.27 Weblate 4.6 ............................................................................................... 434
4.28 Weblate 4.5.3 ............................................................................................... 435
4.29 Weblate 4.5.2 ............................................................................................... 435
4.30 Weblate 4.5.1 ............................................................................................... 435
4.31 Weblate 4.5 ............................................................................................... 435
4.32 Weblate 4.4.2 ............................................................................................... 436
4.33 Weblate 4.4.1 ............................................................................................... 436
4.34 Weblate 4.4 ............................................................................................... 436
4.35 Weblate 4.3.2 ............................................................................................... 437
4.36 Weblate 4.3.1 ............................................................................................... 437
4.37 Weblate 4.3 ............................................................................................... 438
4.38 Weblate 4.2.2 ............................................................................................... 438
<table>
<thead>
<tr>
<th>Section</th>
<th>Version</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.39</td>
<td>Weblate 4.2.1</td>
<td>438</td>
</tr>
<tr>
<td>4.40</td>
<td>Weblate 4.2</td>
<td>439</td>
</tr>
<tr>
<td>4.41</td>
<td>Weblate 4.1.1</td>
<td>439</td>
</tr>
<tr>
<td>4.42</td>
<td>Weblate 4.1</td>
<td>440</td>
</tr>
<tr>
<td>4.43</td>
<td>Weblate 4.0.4</td>
<td>440</td>
</tr>
<tr>
<td>4.44</td>
<td>Weblate 4.0.3</td>
<td>441</td>
</tr>
<tr>
<td>4.45</td>
<td>Weblate 4.0.2</td>
<td>441</td>
</tr>
<tr>
<td>4.46</td>
<td>Weblate 4.0.1</td>
<td>441</td>
</tr>
<tr>
<td>4.47</td>
<td>Weblate 4.0</td>
<td>441</td>
</tr>
<tr>
<td>4.48</td>
<td>Weblate 3.x</td>
<td>442</td>
</tr>
<tr>
<td>4.49</td>
<td>Weblate 2.x</td>
<td>451</td>
</tr>
<tr>
<td>4.50</td>
<td>Weblate 1.x</td>
<td>459</td>
</tr>
<tr>
<td>4.51</td>
<td>Weblate 0.x</td>
<td>462</td>
</tr>
</tbody>
</table>

Index des modules Python 465

Table de routage HTTP 466

Index 469
1.1 Bases de Weblate

1.1.1 Structure du projet et des composants

Dans Weblate, les traductions sont organisées en projets et composants. Chaque projet peut contenir un certain nombre de composants et ceux-ci contiennent des traductions dans des langues individuelles. Le composant correspond à un fichier traduisible (par exemple GNU gettext ou Android string resources). Les projets sont là pour vous aider à organiser les composants en ensembles logiques (par exemple pour regrouper toutes les traductions utilisées dans une application).

En interne, chaque projet dispose par défaut de traductions de chaînes communes qui se propagent à travers les autres composants du projet. Cela allège le fardeau de la traduction répétitive et multiversion. La propagation de la traduction peut être désactivée par Configuration des composants en utilisant Permettre la propagation de la traduction au cas où les traductions devraient diverger.

Voir aussi :
../devel/integration

1.2 Inscription et profil utilisateur

1.2.1 S’inscrire

Tout le monde peut parcourir les projets, voir les traductions ou suggérer des traductions par défaut. Seuls les utilisateurs enregistrés sont autorisés à enregistrer les modifications et sont crédités pour chaque traduction effectuée.

Vous pouvez vous inscrire en suivant les étapes suivantes :

1. Remplissez le formulaire d’inscription avec vos informations d’identification.
2. Activez l’inscription en suivant le lien figurant dans le courriel que vous recevez.
3. Ajustez éventuellement votre profil pour choisir les langues que vous connaissez.
1.2.2 Tableau de bord

Lorsque vous vous connectez, vous verrez un aperçu des projets et des composants, ainsi que leur progression respective en matière de traduction.

Nouveau dans la version 2.5.

Les composants des projets que vous regardez sont affichés par défaut et référencés avec vos langues préférées.

**Indication :** Vous pouvez passer d’une vue à l’autre en utilisant les onglets de navigation.

Le menu comporte les options suivantes :

— **Projets > Parcourir tous les projets** dans le menu principal affichant l'état de la traduction pour chaque projet de l'instance Weblate.

— En sélectionnant une langue dans le menu principal **Langues**, vous verrez l'état de la traduction de tous les projets, filtrés par une de vos langues principales.

— **Traductions surveillées** dans le tableau de bord affichera l'état de la traduction des projets que vous surveillez, filtré par vos langues principales.

En outre, la liste déroulante peut également afficher un nombre quelconque de **liste de composants**, ensembles de composants de projet préconfigurés par l'administrateur Weblate, voir **Liste des composants**.

Vous pouvez configurer votre tableau de bord personnel dans la section **Préférences** des paramètres de votre profil d'utilisateur.
1.2.3 Profil utilisateur

Le profil utilisateur est accessible en cliquant sur l'icône d'utilisateur en haut à droite du menu supérieur, puis sur le menu Paramètres.

Le profil utilisateur contient vos préférences. Le nom et l'adresse courriel utilisés dans les archivages du système de contrôle des versions, alors conservez ces informations exactes.

Note : Toutes les sélections de langue ne proposent que les langues actuellement traduites.

Indication : Demandez ou ajoutez d'autres langues que vous souhaitez traduire en cliquant sur le bouton pour les rendre également disponibles.

Langues

1.2.4 Interface language

Choose the language you want to display the UI in.

Langues traduites

Choisissez les langues que vous préférez traduire, et elles seront proposées sur la page principale des projets surveillés, afin que vous puissiez accéder plus facilement à toutes ces traductions dans chacune de ces langues.
Vous pouvez définir les langues secondaires qui vous sont présentées à titre indicatif lors de la traduction. L'image suivante montre un exemple où l’hébreu est affiché comme langue secondaire :

**Langues secondaires**

Vous pouvez définir les langues secondaires qui vous sont présentées à titre indicatif lors de la traduction. L’image suivante montre un exemple où l’hébreu est affiché comme langue secondaire :
1.2.5 Préférences

Affichage du tableau de bord par défaut

Dans l’onglet Préférences, vous pouvez choisir laquelle des vues du tableau de bord disponibles doit être présentée par défaut. Si vous choisissez Liste de composants, vous devez sélectionner la liste des composants qui sera affichée dans la liste déroulante Liste de composants par défaut.

Voir aussi :

Liste des composants

Lien vers l’éditeur

Un lien vers le code source est affiché dans le navigateur web configuré dans la Configuration des composants par défaut.

Indication : En définissant Lien vers l’éditeur, vous utilisez votre éditeur local pour ouvrir le fichier de code source des chaînes traduites du système de contrôle des versions. Vous pouvez utiliser Balisage de modèle.

En général, quelque chose comme editor://open/?file={{filename}}&line={{line}} est une bonne option.
Voir aussi :
Vous pouvez trouver plus d'informations sur l'enregistrement de protocoles URL personnalisés pour l'éditeur dans la documentation de Nette documentation.

Caractères spéciaux
Additional special characters to include in the Clavier visuel.

1.2.6 Notifications
Abonnez-vous aux différentes notifications de l'onglet Notifications. Les notifications pour les événements sélectionnés sur les projets surveillés ou administrés vous seront envoyées par courriel.
Certaines notifications ne sont envoyées que pour des événements dans vos langues (par exemple au sujet de nouvelles chaînes à traduire), tandis que d'autres se déclenchent au niveau des composants (par exemple des erreurs de fusion). Ces deux groupes de notifications sont visuellement séparés dans les paramètres.
Vous pouvez basculer les notifications pour les projets surveillés et les projets administrés et il est possible de les modifier (ou de les mettre en sourdine) pour chaque projet et chaque composant. Visitez la page d'aperçu du composant et sélectionnez le choix approprié dans le menu Surveillé.

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

Note : Vous ne recevrez pas de notifications pour vos propres actions.

Indication : Sending out notifications is limited, you will not receive more than 1000 e-mails per day. Any further notifications for you will be discarded.
1.2. Inscription et profil utilisateur
1.2.7 Compte

L’onglet *Compte* vous permet de configurer les détails de base du compte, de connecter divers services que vous pouvez utiliser pour vous connecter à Weblate, de supprimer complètement votre compte ou de télécharger vos données utilisateur (voir *Export des données utilisateur Weblate*).

**Note** : La liste des services dépend de la configuration de votre Weblate, mais elle peut inclure des sites populaires tels que GitLab, GitHub, Google, Facebook, Bitbucket, ainsi que d’autres fournisseurs OAuth 2.0.
1.2. Inscription et profil utilisateur
1.2.8 Profil

Tous les champs de cette page sont facultatifs et peuvent être supprimés à tout moment, et en les remplissant, vous nous donnez votre accord pour partager ces données partout où votre profil utilisateur apparaît.

The commit e-mail will be used instead of your account e-mail in version control commits. Use this to avoid leaking your real e-mail there. Be aware that using different e-mail can disconnect your contributions on other servers (for example your contributions will no longer link to your profile on GitHub). The private e-mail can be turned on site-wide using \texttt{PRIVATE\_COMMIT\_EMAIL\_OPT\_IN}.

L’avatar peut être affiché pour chaque utilisateur (en fonction du paramètre \texttt{ENABLE\_AVATARS}). Les images sont obtenues en utilisant https://gravatar.com/.

1.2.9 Licences

1.2.10 Accès par API

You can get or reset your API access token here.

1.2.11 Journal d’audit

Le journal d’audit permet de suivre les actions effectuées avec votre compte. Il enregistre l’adresse IP et le navigateur pour chaque action importante effectuée avec votre compte. Les actions critiques déclenchent également une notification à l’adresse courriel principale.

Voir aussi :

\textit{Running behind reverse proxy}

1.3 Traduction à l’aide de Weblate

Nous vous remercions de l’intérêt que vous portez à la traduction à l’aide de Weblate. Les projets peuvent être configurés pour une traduction directe ou en acceptant les suggestions faites par les utilisateurs sans compte.

Globalement, il existe deux modes de traduction :
— Le projet accepte les traductions directes
— The project only accepts suggestions, which are automatically validated once a defined number of votes is reached

Please see \textit{Flux de travail de traduction} for more info on translation workflow.

Options pour la visibilité des projets de traduction :
— Publicly visible
— Visible uniquement pour un certain groupe de traducteurs

Voir aussi :

\textit{Contrôle d’accès, Flux de travail de traduction}
1.3.1 Projets de traduction

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

1.3.2 Liens de traduction

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

**Note**: Les droits réels peuvent varier en fonction de la configuration de votre Weblate.

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

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

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

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

Voir aussi :

report-source, *Relectures des chaînes sources, Activer la révision des chaînes sources*

1.3.5 Variantes

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

Voir aussi :

variants, *Variantes*

1.3.6 Libellés

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

Following labels are used by Weblate:

**Traduit automatiquement**

La chaîne a été traduite à l'aide de *Traduction automatique*.

**La source doit être vérifiée**

La chaîne a été marquée pour révision à l'aide de *Relectures des chaînes sources*.

Voir aussi :

labels

1.3.7 Traduction

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

Tous les caractères blancs spéciaux sont soulignés en rouge et indiqués par des symboles gris. Les espaces consécutives sont également soulignées en rouge pour alerter le traducteur d’un éventuel problème de formatage.

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

Below the translation, translators will find suggestion made by others, to be accepted (√), accepted with changes (✉), or deleted (☴).
Pluriels

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

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

Voir aussi :

Forme plurielle

![Translation interface](image)
**Alternative translations**

Nouveau dans la version 4.13.

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

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

**Raccourcis clavier**

Modifié dans la version 2.18 : Les raccourcis clavier ont été réorganisés dans la version 2.18 pour éviter toute collision avec les paramètres par défaut du navigateur ou du système d'exploitation.

Les raccourcis clavier suivants peuvent être utilisés lors de la traduction :

<table>
<thead>
<tr>
<th>Raccourcis clavier</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alt+Home</td>
<td>Naviguer vers la première traduction dans la recherche en cours.</td>
</tr>
<tr>
<td>Alt+End</td>
<td>Naviguer jusqu'à la dernière traduction dans la recherche en cours.</td>
</tr>
<tr>
<td>Alt+PageUp or Ctrl+↑ or Alt++ or Cmd++</td>
<td>Naviguer vers la traduction précédente dans la recherche en cours.</td>
</tr>
<tr>
<td>Alt+PageDown or Ctrl-↓ or Alt-↓ or Cmd-↓</td>
<td>Naviguer vers la traduction suivante dans la recherche en cours.</td>
</tr>
<tr>
<td>Ctrl+Enter or Cmd+Entrée</td>
<td>Submit current form; this is same as pressing <em>Save and continue</em> while editing translation.</td>
</tr>
<tr>
<td>Ctrl+Shift+Enter or Cmd+Shift+Entrée</td>
<td>Unmark translation as needing edit and submit it.</td>
</tr>
<tr>
<td>Alt+Enter or Option+Enter</td>
<td>Submit the string as a suggestion; this is same as pressing <em>Suggest</em> while editing translation.</td>
</tr>
<tr>
<td>Ctrl+E or Cmd+E</td>
<td>Placer le focus dans l'éditeur de traduction.</td>
</tr>
<tr>
<td>Ctrl+U or Cmd+U</td>
<td>Placer le focus dans l'éditeur de commentaire.</td>
</tr>
<tr>
<td>Ctrl+M or Cmd+M</td>
<td>Affiche l'onglet <em>Automatic suggestions</em>, consultez <em>Suggestions automatiques</em>.</td>
</tr>
<tr>
<td>Ctrl+1 to Ctrl+9 or Cmd+1 to Cmd+9</td>
<td>Copie l'élément numéroté depuis la chaîne source.</td>
</tr>
<tr>
<td>Ctrl+M+1 to 9 or Cmd+M+1 to 9</td>
<td>Copie la traduction automatique du numéro donné dans la traduction actuelle.</td>
</tr>
<tr>
<td>Ctrl+I+1 to 9 or Cmd+I+1 to 9</td>
<td>Ignorer un élément de la liste des contrôles de qualité défaillants.</td>
</tr>
<tr>
<td>Ctrl+J or Cmd+J</td>
<td>Affiche l'onglet <em>Chaînes à proximité</em>.</td>
</tr>
<tr>
<td>Ctrl+S or Cmd+S</td>
<td>Focus search field.</td>
</tr>
<tr>
<td>Ctrl+O or Cmd+O</td>
<td>Copy source string.</td>
</tr>
<tr>
<td>Ctrl+Y or Cmd+Y</td>
<td>Toggle the <em>Needs editing</em> checkbox.</td>
</tr>
</tbody>
</table>
Clavier visuel

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

Les symboles affichés se répartissent en trois catégories :
—— User configured Caractères spéciaux defined in the Profil utilisateur
—— Les caractères par langue fournis par Weblate (par exemple, les guillemets ou les caractères spécifiques à RTL)
—— Les caractères configurés en utilisant SPECIAL_CHARS

Contexte de traduction

This contextual description provides related info about the current string.

Attributs de la chaîne
Des éléments comme l'ID de message, le contexte (msgctxt) ou l'emplacement dans le code source.

Captures d'écran
Des captures d'écran peuvent être téléversées sur Weblate pour mieux informer les traducteurs sur l'endroit et de la manière dont la chaîne est utilisée, voir Visual context for strings.

Chaînes à proximité
Affiche les messages à proximité dans le fichier de traduction. Ils sont généralement utilisés dans un contexte similaire et s'avèrent utiles pour maintenir la cohérence de la traduction.
Autres occurrences
Dans le cas où un message apparaît à plusieurs endroits (par exemple, plusieurs composants), cet onglet les affiche tous s’ils sont jugés incohérents (voir Incohérence). Vous pouvez choisir lequel utiliser.

Mémoire de traduction
Regarder les chaînes de caractères similaires traduites dans le passé, voir Mémoire de traduction.

Glossaire
Affiche les termes du glossaire du projet utilisés dans le message actuel.

 Modifications récentes
Liste des personnes qui ont récemment modifié ce message en utilisant Weblate.

Projet
Project info like instructions for translators, or a directory or link to the string in the version control system repository the project uses.
If you want direct links, the translation format has to support it.

Historique de la traduction
Chaque modification est enregistrée par défaut (sauf si désactivé dans les paramètres du composant) dans la base de données, et peut être annulée. En option, il est également possible d’annuler toute modification dans le système de contrôle de version sous-jacent.

Longueur de la chaîne traduite
Weblate can limit the length of a translation in several ways to ensure the translated string is not too long:
— The default limitation for translation is ten times longer than the source string. This can be turned off by LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH. In case you are hitting this, it might be also caused by a monolingual translation erroneously set up as bilingual one, making Weblate mistaking the translation key for the actual source string. See Formats monolingues et bilingues for more info.
— Longueur maximale en caractères définie par le fichier de traduction ou le drapeau, voir Taille maximum de la traduction.
— Taille maximale du rendu en pixels définie par des drapeaux, voir Taille maximale de la traduction.

1.3.8 Suggestions automatiques
Based on configuration and your translated language, Weblate provides suggestions from several machine translation tools and Mémoire de traduction. All machine translations are available in a single tab of each translation page.

Voir aussi :
Vous pouvez trouver la liste des outils pris en charge dans Configuring automatic suggestions.

1.3.9 Traduction automatique
Vous pouvez utiliser la traduction automatique pour amorcer une traduction basée sur des sources externes. Cet outil s’appelle Traduction automatique accessible dans le menu Outils, une fois que vous avez sélectionné un composant et une langue :

1.3. Traduction à l’aide de Weblate
Deux modes de fonctionnement sont possibles :
— Utilisation d'autres composants Weblate comme source de traduction.
— Utilisation de services de traduction automatique sélectionnés avec des traductions atteignant un certain seuil de qualité.

Vous pouvez également choisir les chaînes de caractères qui doivent être traduites automatiquement.

Avertissement : Soyez conscient que cela remplacera les traductions existantes si elles sont utilisées avec des filtres larges tels que Toutes les chaînes.

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

The automatically translated strings are labelled Automatically translated.

Voir aussi :
Keeping translations same across components
1.3.10 Limites de requêtes

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

Default limits and fine-tuning is described in the administrative manual, see Limites de requêtes.

1.3.11 Rechercher et remplacer

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

**Indication** : Rassurez-vous, il n'y a aucun risque de causer des dommages. Il s'agit d'un processus à deux étapes affichant d’abord un aperçu des chaînes modifiées avant que les modifications ne soient confirmées.

1.3.12 Modification en masse

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

- Changement de l'état des chaînes (par exemple pour approuver toutes les chaînes non révisées).
- Ajustement des drapeaux de traduction (voir Customizing behavior using flags)
- Ajustement des libellés de chaînes de caractères (voir labels)

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

**Voir aussi** :

Bulk edit add-on

1.3.13 Matrix View

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

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

1.3.14 Zen Mode

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

You can select the Zen editor as your default editor using the Préférences tab on your Profil utilisateur. Here you can also choose between having translations listed Top to bottom or Side by side depending on your personal preference.
1.4 Téléchargement et téléversement des traductions

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

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

1.4.1 Downloading translations

From the project or component dashboard, translatable files can be downloaded in the Files menu. The first option is to download the file in the original format as it is stored in the repository. In this case, any pending changes in the translation are getting committed and the up-to-date file is yield without any conversions.

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

- gettext PO
- XLIFF avec extensions gettext
- XLIFF 1.1
- TermBase eXchange
- Translation Memory eXchange
- gettext MO (only available when translation is using gettext PO)
- CSV
- Excel Open XML
- JSON (uniquement disponible pour les traductions mono-langues)
- Ressource Android (uniquement disponible pour les traductions mono-langues)
- Chaînes iOS (uniquement disponible pour les traductions mono-langues)

Indication : The content available in the converted files differs based on file format features, you can find overview in Fonctionnalités des types de traduction.
Voir aussi :

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

1.4.2 Uploading translations

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

Formats de fichiers pris en charge

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

Voir aussi :

*Formats de fichiers pris en charge, Téléchargement et téléversement des traductions*
**Import methods**

These are the choices presented when uploading translation files:

**Ajouter comme traduction** *(translate)*

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

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

**Ajouter comme suggestion** *(suggest)*

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

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

**Ajouter comme traduction à vérifier** *(fuzzy)*

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

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

**Remplacer le fichier de traduction existant** *(replace)*

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

**Mettre à jour les chaînes sources** *(source)*

Updates source strings in bilingual translation file. This is similar to what *Mettre à jour les fichiers PO afin qu’ils correspondent au POT (msgmerge)* does.

This option is supported only for some file formats.

**Add new strings** *(add)*

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

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

This option is available only with *Gérer les chaînes* turned on.

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

**Voir aussi :**

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

**Conflicts handling**

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

**Strings needing edit**

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

**Overriding authorship**

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

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

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

1.5.1 Gestion des glossaires

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

Use any component as a glossary by turning on Utiliser comme glossaire. You can create multiple glossaries for one project.

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

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

Vous pouvez parcourir tous les termes du glossaire :
ou les modifier comme n’importe quelle traduction.

### 1.5.2 Glossary terms

Les termes du glossaire sont traduits de la même manière que les chaînes de caractères ordinaires. Vous pouvez activer des fonctionnalités supplémentaires en utilisant le menu *Outils* pour chaque terme.
Untranslatable terms

Nouveau dans la version 4.5.

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

Voir aussi :

*Customizing behavior using flags*

Forbidden translations

Nouveau dans la version 4.5.

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

Voir aussi :

*Customizing behavior using flags*

Terminologie

Nouveau dans la version 4.5.

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

Voir aussi :

*Customizing behavior using flags*

Variantes

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

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

Voir aussi :

variants

1.6 Contrôles de qualité et corrections

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

Once submitting a translation with a failing check, this is immediately shown to the user :
Chapitre 1. Documentation pour l’utilisateur
1.6.1 Automatic fixups

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

Voir aussi :

AUTOFIX_LIST

1.6.2 Quality checks

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

Voir aussi :

CHECK_LIST, Customizing behavior using flags

1.6.3 Translation checks

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

Balisage BBCode

Summary
Les BBCodes dans la traduction ne correspondent pas à la source

Portée
translated strings

Check class
weblate.checks.markup.BBCodeCheck

Check identifier
bbcode

Flag to ignore
ignore-bbcode

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

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

Répétition de mots

Nouveau dans la version 4.1.

Summary
Le texte contient une répétition du même mot deux fois de suite :

Portée
translated strings

Check class
weblate.checks.duplicate.DuplicateCheck

Check identifier
duplicate
Flag to ignore

ignore-duplicate

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

**Indication:** This check includes language specific rules to avoid false positives. In case it triggers falsely in your case, let us know. See *Reporting issues in Weblate*.

---

### Non conforme au glossaire

Nouveau dans la version 4.5.

**Summary**
La traduction ne respecte pas les termes du glossaire.

**Portée**
translated strings

**Check class**
weblate.checks.glossary.GlossaryCheck

**Check identifier**
check_glossary

**Flag to enable**
check-glossary

**Flag to ignore**
ignore-check-glossary

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

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

**Voir aussi:**
Glossaire, Customizing behavior using flags, Drapeaux de traduction

---

### Double espace

**Summary**
La traduction contient un double espace

**Portée**
translated strings

**Check class**
weblate.checks.chars.DoubleSpaceCheck

**Check identifier**
double_space

**Flag to ignore**
ignore-double-space

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

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

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

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

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

**Indication**: In case specific format check is not available in Weblate, you can use generic Balises de remplacement.

Besides checking, this will also highlight the formatting strings to easily insert them into translated strings:
Chaîne d’interpolation AngularJS

Summary
Les chaînes d’interpolation AngularJS ne correspondent pas à la source

Portée
translated strings

Check class
weblate.checks.angularjs.AngularJSInterpolationCheck

Check identifier
angularjs_format

Flag to enable
angularjs-format

Flag to ignore
ignore-angularjs-format

Named format string example
Votre solde est {{amount}} {{ currency }}

Voir aussi :
Formatted strings, AngularJS text interpolation

Format C

Summary
La chaîne de format C ne correspond pas à celle de la source

Portée
translated strings

Check class
weblate.checks.format.CFormatCheck

Check identifier
c_format

Flag to enable
c-format

Flag to ignore
ignore-c-format

Simple format string example
Il y a %d pommes

Position format string example
Votre solde est %1$d %2$s

Voir aussi :

Formatted strings,
C format strings, C printf format
Format C#

Summary
La chaîne au format C# ne correspond pas à celle de la source

Portée
translated strings

Check class
weblate.checks.format.CSharpFormatCheck

Check identifier
c_sharp_format

Flag to enable
c-sharp-format

Flag to ignore
ignore-c-sharp-format

Position format string example
Il y a {0} pommes

Voir aussi :
Formatted strings, C# String Format

Modèle de littéraux ECMAScript

Summary
Les modèles littéraux ECMAScript ne correspondent pas à la source

Portée
translated strings

Check class
weblate.checks.format.ESTemplateLiteralsCheck

Check identifier
es_format

Flag to enable
es-format

Flag to ignore
ignore-es-format

Interpolation example
Il y a ${number} pommes

Voir aussi :
Formatted strings, Template literals

Interpolation i18next

Nouveau dans la version 4.0.

Summary
L’interpolation i18next ne correspond pas à la source

Portée
translated strings

Check class
weblate.checks.format.I18NextInterpolationCheck

Check identifier
i18next_interpolation
Flag to enable
   i18next-interpolation
Flag to ignore
   ignore-i18next-interpolation
Interpolation example
   Il y a {{number}} pommes
Nesting example
   Il y a $t(number) pommes

Voir aussi :

Formatted strings, i18next interpolation

**ICU MessageFormat**

Nouvea dans la version 4.9.

**Summary**

Erreurs de syntaxe et/ou incohérence de caractère de remplacement dans les chaînes ICU Message-Format.

**Portée**

translated strings

**Check class**

webate.checks.icu.ICUMessageFormatCheck

**Check identifier**

icu_message_format

**Flag to enable**

icu-message-format

**Flag to ignore**

ignore-icu-message-format

**Interpolation example**

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

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

```
icu-message-format, icu-flags:xml:-plural_selectors```

<table>
<thead>
<tr>
<th>xml</th>
<th>Enable support for simple XML tags. By default, XML tags are parsed loosely. Stray <code>&lt;</code> characters are ignored if they are not reasonably part of a tag.</th>
</tr>
</thead>
<tbody>
<tr>
<td>strict-xml</td>
<td>Enable support for strict XML tags. All <code>&lt;</code> characters must be escaped if they are not part of a tag.</td>
</tr>
<tr>
<td>-highlight</td>
<td>Disable highlighting placeholders in the editor.</td>
</tr>
<tr>
<td>-require_other</td>
<td>Disable requiring sub-messages to have an other selector.</td>
</tr>
<tr>
<td>-submessage_se</td>
<td>Skip checking that sub-message selectors match the source.</td>
</tr>
<tr>
<td>-types</td>
<td>Ne pas vérifier que les types de marqueurs de position correspondent à la source.</td>
</tr>
<tr>
<td>-extra</td>
<td>Skip checking that no placeholders are present that were not present in the source string.</td>
</tr>
<tr>
<td>-missing</td>
<td>Skip checking that no placeholders are missing that were present in the source string.</td>
</tr>
</tbody>
</table>

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

Voir aussi :

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

**Format Java**

**Summary**
La chaîne au format Java ne correspond pas à celle de la source

**Portée**
translated strings

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

**Check identifier**
`java_printf_format`

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

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

**Simple format string example**
Il y a %d pommes

**Position format string example**
Votre solde est %1$d %2$s

Modifié dans la version 4.14 : This used to be toggled by `java-format` flag, it was changed for consistency with GNU gettext.

Voir aussi :

Formatted strings, Java Format Strings

**MessageFormat Java**

**Summary**
La chaîne MessageFormat Java ne correspond pas à celle de la source

**Portée**
translated strings

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

**Check identifier**
`java_format`

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

**Flag to enable autodetection**
`auto-java-messageformat` enables check only if there is a format string in the source

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

**Position format string example**
Il y a `{0}` pommes
This check validates that format string is valid for the Java MessageFormat class. Besides matching format strings in the curly braces, it also verifies single quotes as they have a special meaning. Whenever writing single quote, it should be written as ' '. When not paired, it is treated as beginning of quoting and will not be shown when rendering the string.

Voir aussi :

Formatted strings, Java MessageFormat

**Format JavaScript**

---

**Summary**

La chaîne de format JavaScript ne correspond pas à celle de la source

**Portée**

translated strings

**Check class**

weblate.checks.format.JavaScriptFormatCheck

**Check identifier**

javascript_format

**Flag to enable**

javascript-format

**Flag to ignore**

ignore-javascript-format

**Simple format string example**

Il y a %d pommes

Voir aussi :

Formatted strings, JavaScript formatting strings

**Format Lua**

---

**Summary**

La chaîne de format Lua ne correspond pas à celle de la source

**Portée**

translated strings

**Check class**

weblate.checks.formatLuaFormatCheck

**Check identifier**

lua_format

**Flag to enable**

lua-format

**Flag to ignore**

ignore-lua-format

**Simple format string example**

Il y a %d pommes

Voir aussi :

Formatted strings, Lua formatting strings
Format Pascal objet

 Summary
 La chaîne au format Pascal objet ne correspond pas à la source
 Portée
 translated strings
 Check class
 weblate.checks.format.ObjectPascalFormatCheck
 Check identifier
 object_pascal_format
 Flag to enable
 object-pascal-format
 Flag to ignore
 ignore-object-pascal-format
 Simple format string example
 Il y a %d pommes

Voir aussi :
Formatted strings, Object Pascal formatting strings, Free Pascal formatting strings Delphi formatting strings

Balises de remplacement par caractères pour cent

Nouveau dans la version 4.0.

 Summary
 Les balises de remplacement par caractères pour cent ne correspondent pas à la source
 Portée
 translated strings
 Check class
 weblate.checks.format.PercentPlaceholdersCheck
 Check identifier
 percent_placeholders
 Flag to enable
 percent-placeholders
 Flag to ignore
 ignore-percent-placeholders
 Simple format string example
 Il y a %number% pommes

Voir aussi :
Formatted strings,

Format Perl

 Summary
 La chaîne au format Perl ne correspond pas à la source
 Portée
 translated strings
 Check class
 weblate.checks.format.PerlFormatCheck
 Check identifier
 perl_format
Flag to enable
perl-format

Flag to ignore
ignore-perl-format

Simple format string example
Il y a %d pommes

Position format string example
Votre solde est %1$d %2$s

Voir aussi :
Formatted strings, Perl sprintf, Perl Format Strings

Format PHP

Summary
La chaîne de format PHP ne correspond pas à celle de la source

Portée
translated strings

Check class
weblate.checks.format.PHPFormatCheck

Check identifier
php_format

Flag to enable
php-format

Flag to ignore
ignore-php-format

Simple format string example
Il y a %d pommes

Position format string example
Votre solde est %1$d %2$s

Voir aussi :
Formatted strings, PHP sprintf documentation, PHP Format Strings

Format d'accolade Python

Summary
La chaîne de format python ne correspond pas à celui de la source

Portée
translated strings

Check class
weblate.checks.format.PythonBraceFormatCheck

Check identifier
python_brace_format

Flag to enable
python-brace-format

Flag to ignore
ignore-python-brace-format

Simple format string
Il y a {} pommes
Named format string example
Votre solde est {amount} {currency}

Voir aussi :
Formatted strings, Python brace format, Python Format Strings

Format Python

Summary
La chaîne de format Python ne correspond pas à celle de la source

Portée
translated strings

Check class
weblate.checks.format.PythonFormatCheck

Check identifier
python_format

Flag to enable
python-format

Flag to ignore
ignore-python-format

Simple format string
Il y a %d pommes

Named format string example
Your balance is %(amount)d %(currency)s

Voir aussi :
Formatted strings, Python string formatting, Python Format Strings

Format Qt

Summary
La chaîne de format Qt ne correspond pas à la source

Portée
translated strings

Check class
weblate.checks.qt.QtFormatCheck

Check identifier
qt_format

Flag to enable
qt-format

Flag to ignore
ignore-qt-format

Position format string example
Il y a %1 pommes

Voir aussi :
Formatted strings, Qt QString ::arg()
Forme plurielle Qt

Summary
La chaîne au format Qt pluriel ne correspond pas à la source

Portée
translated strings

Check class
weblate.checks.qt.QtPluralCheck

Check identifier
qt_plural_format

Flag to enable
qt-plural-format

Flag to ignore
ignore-qt-plural-format

Plural format string example
Il y a %Ln pomme(s)

Voir aussi :
Formatted strings, Qt i18n guide

Format Ruby

Summary
La chaîne de format Ruby ne correspond pas à la source

Portée
translated strings

Check class
weblate.checks.ruby.RubyFormatCheck

Check identifier
ruby_format

Flag to enable
ruby-format

Flag to ignore
ignore-ruby-format

Simple format string example
Il y a %d pommes

Position format string example
Votre solde est %1$f %2$s

Named format string example
Votre solde est %+.2<amount>f %<currency>s

Named template string
Votre solde est %{amount} %{currency}

Voir aussi :
Formatted strings, Ruby Kernel#sprintf
Format Scheme

Summary
La chaîne de format Scheme ne correspond pas à la source

Portée
translated strings

Check class
webate.checks.format.SchemeFormatCheck

Check identifier
scheme_format

Flag to enable
scheme-format

Flag to ignore
ignore-scheme-format

Simple format string example
There are ~d apples

Voir aussi :
Formatted strings, Srfi 28, Chicken Scheme format, Guile Scheme formatted output

Formatage Vue I18n

Summary
Le formatage Vue I18n ne correspond pas à celui de la source

Portée
translated strings

Check class
webate.checks.format.VueFormattingCheck

Check identifier
vue_format

Flag to enable
evue-format

Flag to ignore
ignore-vue-format

Formatage nommé
Il y a {count} pommes

Formatage Rails I18n
Il y a %{count} pommes

Messages de paramètres régionaux liés
@:message.dio @:message.the_world!

Voir aussi :
Formatted strings, Vue I18n Formatting, Vue I18n Linked locale messages

1.6. Contrôles de qualité et corrections
A déjà été traduit

Summary
Cette chaîne a été traduite par le passé

Portée
all strings

Check class
weblate.checks.consistency.TranslatedCheck

Check identifier
translated

Flag to ignore
ignore-translated

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

Incohérence

Summary
Cette chaîne a différentes traductions dans ce projet ou n’est pas traduite dans certains composants.

Portée
all strings

Check class
weblate.checks.consistency.ConsistencyCheck

Check identifier
inconsistent

Flag to ignore
ignore-inconsistent

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

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

Cette vérification s'applique à tous les composants d'un projet dont l'option Permettre la propagation de la traduction est activée.

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

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

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

Voir aussi :

Keeping translations same across components
Présence d’un caractère kashida

Nouveau dans la version 3.5.

Summary
Les lettres décoratives kashida ne doivent pas être utilisées

Portée
translated strings

Check class
weblate.checks.chars.KashidaCheck

Check identifier
kashida

Flag to ignore
ignore-kashida

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

Voir aussi :
Kashida sur Wikipédia

Liens Markdown

Nouveau dans la version 3.5.

Summary
Les liens Markdown ne correspondent pas à la source

Portée
translated strings

Check class
weblate.checks.markup.MarkdownLinkCheck

Check identifier
md-link

Flag to enable
md-text

Flag to ignore
ignore-md-link

Les liens Markdown ne correspondent pas à la source.

Voir aussi :
Markdown links

Références Markdown

Nouveau dans la version 3.5.

Summary
Les références de liens Markdown ne correspondent pas à la source

Portée
translated strings

Check class
weblate.checks.markup.MarkdownRefLinkCheck

Check identifier
md-reflink

Flag to enable
md-text
Flag to ignore
  ignore-md-reflink
Markdown link references do not match source.
Voir aussi :
Markdown links

Syntaxe Markdown

Nouveau dans la version 3.5.

Summary
  La syntaxe Markdown ne correspond pas à la source
Portée
  translated strings
Check class
  weblate.checks.markup.MarkdownSyntaxCheck
Check identifier
  md-syntax
Flag to enable
  md-text
Flag to ignore
  ignore-md-syntax
La syntaxe Markdown ne correspond pas à la source
Voir aussi :
Markdown span elements

Taille maximum de la traduction

Summary
  La traduction ne doit pas dépasser la taille indiquée
Portée
  translated strings
Check class
  weblate.checks.chars.MaxLengthCheck
Check identifier
  max-length
Flag to enable
  max-length
Flag to ignore
  ignore-max-length
Checks that translations are of acceptable length to fit available space. This only checks for the length of translation characters.
Unlike the other checks, the flag should be set as a key:value pair like max-length:100.

Indication : This check looks at number of chars, what might not be the best metric when using proportional fonts to render the text. The Taille maximale de la traduction check does check actual rendering of the text.
The replacements: flag might be also useful to expand placeables before checking the string.
Quand le drapeau xml-text est aussi utilisé, le calcul de la longueur ignore les balises XML.
Taille maximale de la traduction

**Summary**
Le texte traduit ne doit pas dépasser une taille donnée

**Portée**
translated strings

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

**Check identifier**
max-size

**Flag to enable**
max-size

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

Nouveau dans la version 3.7.

Le texte rendu par la traduction ne doit pas dépasser une taille donnée. Elle fait le rendu du texte avec un retour à la ligne et vérifie s'il tient dans les limites données.

Cette vérification nécessite un ou deux paramètres - largeur maximale et nombre maximal de lignes. Si le nombre de lignes n'est pas indiqué, une seule ligne sera autorisée.

Vous pouvez également configurer la police utilisée à l'aide des directives `font-*` (voir *Customizing behavior using flags*), par exemple les drapeaux de traduction suivants indiquent que le texte rendu avec la police Ubuntu de taille 22 doit tenir sur deux lignes et 500 pixels :

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

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

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

Quand le drapeau `xml-text` est aussi utilisé, le calcul de la longueur ignore les balises XML.

**Voir aussi** :
*Gestion des polices, Customizing behavior using flags, Taille maximum de la traduction*

Pas de correspondance \n
**Summary**
Le nombre de littéraux \n de la traduction n'est pas identique à celui de la source

**Portée**
translated strings

**Check class**
weblate.checks.chars.EscapedNewlineCountingCheck

**Check identifier**
escape_newline

**Flag to ignore**
ignore-escaped-newline

Usually escaped newlines are important for formatting program output. Check fails if the number of `\n` literals in translation do not match the source.
**Incohérence de caractère deux-points**

**Summary**
La chaîne source et la traduction ne finissent pas toutes les deux par deux-points

**Portée**
translated strings

**Check class**
weblate.checks.chars.EndColonCheck

**Check identifier**
end_colon

**Flag to ignore**
ignore-end-colon

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

Voir aussi :
Colon on Wikipedia

**Incohérence de points de suspension**

**Summary**
La chaîne source et la traduction ne se finissent pas toutes les deux par des points de suspension

**Portée**
translated strings

**Check class**
weblate.checks.chars.EndEllipsisCheck

**Check identifier**
end_ellipsis

**Flag to ignore**
ignore-end-ellipsis

Vérifie que le caractère ... est répliqué entre la source et la traduction. Cette vérification ne porte que sur les ellipses réelles (…) et non sur les trois points (...).

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

Voir aussi :
Ellipsis on Wikipedia

**Incohérence de point d’exclamation**

**Summary**
La source et la traduction ne finissent pas toutes les deux par un point d’exclamation

**Portée**
translated strings

**Check class**
weblate.checks.chars.EndExclamationCheck

**Check identifier**
end_exclamation

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

Voir aussi :
Point d'exclamation sur Wikipédia

**Incohérence de point final**

Summary
La chaîne source et la traduction ne finissent pas toutes les deux par un point final

Portée
translated strings

Check class
weblate.checks.chars.EndStopCheck

Check identifier
ded_stop

Flag to ignore
ignore-end-stop

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

Voir aussi :
Point final sur Wikipédia

**Incohérence de point d’interrogation**

Summary
La source et la traduction ne finissent pas toutes les deux par un point d’interrogation

Portée
translated strings

Check class
weblate.checks.chars.EndQuestionCheck

Check identifier
ded_question

Flag to ignore
ignore-end-question

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

Voir aussi :
Point d’interrogation sur Wikipédia
Incohérence de point-virgule

Summary
La chaîne source et la traduction ne finissent pas toutes les deux par un point-virgule

Portée
translated strings

Check class
weblate.checks.chars.EndSemicolonCheck

Check identifier
d_end_semicolon

Flag to ignore
ignore-end-semicolon

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

Voir aussi :
Point-virgule sur Wikipedia

Incohérence dans les sauts de ligne

Summary
Le nombre de lignes de la traduction n’est pas identique à la source

Portée
translated strings

Check class
weblate.checks.chars.NewLineCountCheck

Check identifier
newline-count

Flag to ignore
ignore-newline-count

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

Pluriels manquants

Summary
Certaines formes plurielles n’ont pas été traduites

Portée
translated strings

Check class
weblate.checks.consistency.PluralsCheck

Check identifier
plurals

Flag to ignore
ignore-plurals

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

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

Nouveau dans la version 3.9.

Summary
Balises de remplacement absentes de la traduction

Portée
translated strings

Check class
weblate.checks.placeholders.PlaceholderCheck

Check identifier
placeholders

Flag to enable
placeholders

Flag to ignore
ignore-placeholders

Modifié dans la version 4.3: Vous pouvez utiliser des expressions rationnelles comme substitut.

Modifié dans la version 4.13: With the case-insensitive flag, the placeholders are not case-sensitive.

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

placeholders:$URL:$TARGET$:*some long text*

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

placeholders:r"^[^[ ] $"

You can also have case insensitive placeholders:

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

Voir aussi :
Customizing behavior using flags

Espacement de ponctuation

Nouveau dans la version 3.9.

Summary
Espace insécable manquante devant le signe de ponctuation double

Portée
translated strings

Check class
weblate.checks.chars.PunctuationSpacingCheck

Check identifier
punctuation_spacing

Flag to ignore
ignore-punctuation-spacing

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

Voir aussi :
Espacement français et anglais sur Wikipédia
Expression rationnelle

Nouveau dans la version 3.9.

**Summary**
La traduction ne respecte pas l’expression rationnelle

**Portée**
translated strings

**Check class**
weblate.checks.placeholders.RegexCheck

**Check identifier**
regex

**Flag to enable**
regex

**Flag to ignore**
ignore-regex

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

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

Pluriel identique

**Summary**
Les traductions au singulier et au pluriel sont identiques

**Portée**
translated strings

**Check class**
weblate.checks.consistency.SamePluralsCheck

**Check identifier**
same-plurals

**Flag to ignore**
ignore-same-plurals

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

Nouvelle ligne au début

**Summary**
La chaîne source et la traduction ne commencent pas toutes les deux par un saut de ligne

**Portée**
translated strings

**Check class**
weblate.checks.chars.BeginNewlineCheck

**Check identifier**
begin_newline

**Flag to ignore**
ignore-begin-newline

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

**Voir aussi :**

*Saut de ligne à la fin*
Espaces au début

Summary
La chaîne source et la traduction ne commencent pas toutes les deux par le même nombre d’espaces

Portée
translated strings

Check class
weblate.checks.chars.BeginSpaceCheck

Check identifier
begin_space

Flag to ignore
ignore-begin-space

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

Saut de ligne à la fin

Summary
La chaîne source et la traduction ne finissent pas toutes les deux par un saut de ligne

Portée
translated strings

Check class
weblate.checks.chars.EndNewlineCheck

Check identifier
end_newline

Flag to ignore
ignore-end-newline

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

Voir aussi :

Nouvelle ligne au début

Espace à la fin

Summary
La chaîne source et la traduction ne finissent pas toutes les deux par un espace

Portée
translated strings

Check class
weblate.checks.chars.EndSpaceCheck

Check identifier
end_space

Flag to ignore
ignore-end-space

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

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

Summary
La chaîne source et la chaîne traduite sont identiques

Portée
translated strings

Check class
weblate.checks.same.SameCheck

Check identifier
same

Flag to ignore
ignore-same

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

This check can help find strings mistakenly untranslated.

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

Voir aussi :
Configuration des composants, Customizing behavior using flags

HTML non sûr

Nouveau dans la version 3.9.

Summary
La traduction utilise du code HTML non sûr

Portée
translated strings

Check class
weblate.checks.markup.SafeHTMLCheck

Check identifier
safe-html

Flag to enable
safe-html

Flag to ignore
ignore-safe-html

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

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

Voir aussi :

The HTML check is performed by the Ammonia library.
URL

Nouveau dans la version 3.5.

Summary
La traduction ne contient pas d’URL

Portée
translated strings

Check class
weblate.checks.markup.URLCheck

Check identifier
url

Flag to enable
url

Flag to ignore
ignore-url

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

Balisage XML

Summary
Les balises XML dans la traduction ne correspondent pas à la source

Portée
translated strings

Check class
weblate.checks.markup.XMLTagsCheck

Check identifier
xml-tags

Flag to ignore
ignore-xml-tags

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

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

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

Syntaxe XML

Nouveau dans la version 2.8.

Summary
Cette traduction n’est pas un XML valide

Portée
translated strings

Check class
weblate.checks.markup/XMLValidityCheck

Check identifier
xml-invalid
**Flag to ignore**

`ignore-xml-invalid`

The XML markup is not valid.

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

---

**Espace sans chasse**

**Summary**

La traduction contient un caractère espace sans chasse

**Portée**

translated strings

**Check class**

`weblate.checks.chars.ZeroWidthSpaceCheck`

**Check identifier**

`zero-width-space`

**Flag to ignore**

`ignore-zero-width-space`

Zero-width space (`U+200B`) characters are used to break messages within words (word wrapping). As they are usually inserted by mistake, this check is triggered once they are present in translation. Some programs might have problems when this character is used.

**Voir aussi :**

Zero width space on Wikipedia

---

**1.6.4 Source checks**

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

**Points de suspension**

**Summary**

Cette chaîne contient trois points (…) au lieu du caractère points de suspension (…)  

**Portée**

chaînes sources

**Check class**

`weblate.checks.source.EllipsisCheck`

**Check identifier**

`ellipsis`

**Flag to ignore**

`ignore-ellipsis`

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

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

**Voir aussi :**

Ellipsis on Wikipedia
Syntaxe ICU MessageFormat

Nouveau dans la version 4.9.

Summary
Erreurs de syntaxe dans les chaines ICU MessageFormat.

Portée
chaînes sources

Check class
weblate.checks.icu.ICUSourceCheck

Check identifier
icu_message_format_syntax

Flag to enable
icu-message-format

Flag to ignore
ignore-icu-message-format

Voir aussi :
ICU MessageFormat

Ancienne chaîne non traduite

Nouveau dans la version 4.1.

Summary
Cette chaîne n’a pas été traduite depuis longtemps

Portée
chaînes sources

Check class
weblate.checks.source.LongUntranslatedCheck

Check identifier
long_untranslated

Flag to ignore
ignore-long-untranslated

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

Plusieurs vérifications en échec

Summary
Les traductions dans plusieurs langues ont des vérifications échouées

Portée
chaînes sources

Check class
weblate.checks.source.MultipleFailingCheck

Check identifier
multiple_failures

Flag to ignore
ignore-multiple-failures

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

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

Nouveau dans la version 4.1.

**Summary**
Il y a plusieurs variables non nommées dans la chaîne, ce qui rend impossible leur réorganisation par les traducteurs

**Portée**
chaînes sources

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

**Check identifier**
unnamed_format

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

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

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

Non pluralisé

**Summary**
Cette chaîne est utilisée comme un pluriel, sans utiliser les formes plurielles

**Portée**
chaînes sources

**Check class**
weblate.checks.source.OptionalPluralCheck

**Check identifier**
optionalplural

**Flag to ignore**
ignore-optionalplural

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

For example with Gettext in Python it could be:

```python
from gettext import ngettext

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

1.7 Recherche

Nouveau dans la version 3.9.

Des requêtes avancées utilisant des opérations booléennes, des parenthèses ou une recherche par champ spécifique peuvent être utilisées pour trouver les chaînes de caractères que vous souhaitez.

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

Toute phrase entrée dans la boîte de recherche est divisée en mots. Les chaînes de caractères contenant l’un d’eux sont affichées. Pour rechercher une phrase exacte, mettez la « phrase recherchée » entre guillemets (les guillemets simples (”) et doubles ( ») fonctionneront) : "ceci est une chaîne entre guillemets" ou 'une autre chaîne entre guillemets'.

1.7.2 Champs

source:TEXTE  
Source string case-insensitive search.

target:TEXTE  
Target string case-insensitive search.

context:TEXTE  
Context string case-insensitive search.

key:TEXTE  
Key string case-insensitive search.

note:TEXTE  
Source string description case-insensitive search.
location : TEXTE
  Location string case-insensitive search.

priority : NUMÉRO
  Priorité de chaîne de caractères.

id : NUMBER
  String unique identifier.

added : DATEHEURE
  Horodatage de la date à laquelle la chaîne a été ajoutée dans Weblate.

state : TEXTE
  Search for string states (approved, translated, needs-editing, empty, read-only), supports Opérateurs de champs.

pending : BOOLÉEN
  Chaîne en attente d’archivage dans le système de contrôle des versions.

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

is : TEXTE
  Search for pending translations (pending). Can also search for all string states (approved, translated, untranslated, needs-editing, read-only).

language : TEXTE
  Langue cible de la chaîne.

cOMPONENT : TEXTE
  Component slug or name case-insensitive search, see Identifiant du composant and Nom du composant.

project : TEXTE
  Identifiant du projet, voir Abrégé de l’URL.

changed_by : TEXTE
  La chaîne a été modifiée par l’auteur avec le nom d’utilisateur fourni.

changed : DATEHEURE
  Le contenu de la chaîne a été modifiée à la date fournie, prend en charge Opérateurs de champs.

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

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

check : TEXTE
  String has failing check, see Contrôles de qualité et corrections for check identifiers.

dismissed_check : TEXTE
  String has dismissed check, see Contrôles de qualité et corrections for check identifiers.

comment : TEXTE
  Recherche dans les commentaires des utilisateurs.

resolved_comment : TEXT
  Search in resolved comments.

comment_author : TEXTE
  Filtre par auteur de commentaire.

suggestion : TEXTE
  Recherche dans les suggestions.

suggestion_author : TEXTE
  Filtre par auteur de suggestion.

explanation : TEXT
  Rechercher dans les explications.
1.7.3 Opérateurs booléens

Vous pouvez combiner des recherches en utilisant AND, OR, NOT et des parenthèses pour former des requêtes complexes. Par exemple: `state:translated AND (source:hello OR source:bar)`

1.7.4 Opérateurs de champs

Vous pouvez spécifier des opérateurs, des plages ou des recherches partielles pour les recherches de date ou numériques:

- `state:>=translated`
  L'état est translated (traduit) ou mieux (approved (approuvé)).

- `changed:2019`

- `changed:[2019-03-01 to 2019-04-01]`
  Modifié entre les deux dates fournies.

1.7.5 Opérateurs exacts

Vous pouvez effectuer une recherche de correspondance exacte sur différents champs de chaîne de caractères en utilisant l'opérateur `. Par exemple, pour rechercher toutes les chaînes de caractères source correspondant exactement à `hello world`, utilisez `source:="hello world"`. Pour la recherche d'expressions d'un seul mot, vous pouvez sauter les guillemets. Par exemple, pour rechercher toutes les chaînes de caractères source correspondants à `hello`, vous pouvez utiliser `source:=hello`.

1.7.6 Recherche de modifications

Nouveau dans la version 4.4.

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

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

1.7.7 Expressions rationnelles

Partout où le texte est accepté, vous pouvez également spécifier une expression rationnelle sous la forme `r"regexp"`.

Par exemple, pour rechercher toutes les chaînes de caractères source qui contiennent un chiffre compris entre 2 et 5, utilisez `source:r"[2-5]"`.

1.7. Recherche
1.7.8 Requêtes prédéfinies

Vous pouvez choisir parmi un certain nombre de requêtes prédéfinies sur la page de recherche, ce qui vous permet d'accéder rapidement aux recherches les plus fréquentes:
1.7.9 Ordre des résultats

Il existe de nombreuses options pour ordonner les chaînes en fonction de vos besoins :

1.8 Flux de travail de traduction

L’utilisation de Weblate est un processus qui rapproche vos utilisateurs de vous, en vous rapprochant de vos traducteurs. C’est à vous de décider du nombre de fonctionnalités que vous souhaitez utiliser.

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

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

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

1.8.2 États de traduction

Chaque chaîne traduite peut être dans l’un des états suivants :

**Non traduit**
La traduction est vide. Selon le format du fichier, elle peut être stockée ou non.

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

**En attente de révision**
La traduction est effectuée, mais non relue. Elle est stockée dans le fichier comme une traduction valide.

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

This state is only available when reviews are enabled.

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

The states are represented in the translation files when possible.

**Indication** : In case file format you use does not support storing states, you might want to use *Marquer les traductions inchangées comme « À vérifier »* add-on to flag unchanged strings as needing editing.

Voir aussi :

*Fonctionnalités des types de traduction, Flux de travail de traduction*

1.8.3 Traduction directe

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

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

<table>
<thead>
<tr>
<th>Paramètre</th>
<th>Valeur</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activer les révisions</td>
<td>inactif</td>
<td>Configuré au niveau du projet.</td>
</tr>
<tr>
<td>Autoriser les suggestions</td>
<td>actif</td>
<td>Il est utile que les utilisateurs puissent suggérer quand ils ne sont pas sûrs.</td>
</tr>
<tr>
<td>Vote pour la suggestion</td>
<td>inactif</td>
<td></td>
</tr>
<tr>
<td>Accepter automatiquement les suggestions</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Groupe de traducteurs</td>
<td>Utilisateurs</td>
<td>Ou <em>Traduire avec per-project access control.</em></td>
</tr>
<tr>
<td>Reviewers group</td>
<td>N/A</td>
<td>Non utilisé.</td>
</tr>
</tbody>
</table>
1.8.4 Peer review

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

— *Any user* can add suggestions.
— *Any user* can vote for suggestions.
— Les suggestions deviennent des traductions lorsqu’elles reçoivent un nombre prédéterminé de votes.

<table>
<thead>
<tr>
<th>Paramètre</th>
<th>Valeur</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activer les révisions</td>
<td>inactif</td>
<td>Configuré au niveau du projet.</td>
</tr>
<tr>
<td>Autoriser les suggestions</td>
<td>actif</td>
<td></td>
</tr>
<tr>
<td>Vote pour la suggestion</td>
<td>inactif</td>
<td></td>
</tr>
<tr>
<td>Accepter automatiquement les suggestions</td>
<td>1</td>
<td>You can set higher value to require more peer reviews.</td>
</tr>
<tr>
<td>Groupe de traducteurs</td>
<td>Utilisa-teurs</td>
<td>Ou Traduire avec per-project access control.</td>
</tr>
<tr>
<td>Reviewers group</td>
<td>N/A</td>
<td>Not used, all translators review.</td>
</tr>
</tbody>
</table>

1.8.5 Dedicated reviewers

Nouveau dans la version 2.18 : The proper review workflow is supported since Weblate 2.18.

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

— *Any user* can edit unapproved translations.
— *Le réviseur* peut approuver/désapprouver des chaînes.
— *Reviewer* can edit all translations (including approved ones).
— Les suggestions permettent également de suggérer des modifications pour les chaînes approuvées.

<table>
<thead>
<tr>
<th>Paramètre</th>
<th>Valeur</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activer les révisions</td>
<td>actif</td>
<td>Configuré au niveau du projet.</td>
</tr>
<tr>
<td>Autoriser les suggestions</td>
<td>inactif</td>
<td>Il est utile que les utilisateurs puissent suggérer quand ils ne sont pas sûrs.</td>
</tr>
<tr>
<td>Vote pour la suggestion</td>
<td>inactif</td>
<td></td>
</tr>
<tr>
<td>Accepter automatiquement les suggestions</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Groupe de traducteurs</td>
<td>Utilisa-teurs</td>
<td>Ou Traduire avec per-project access control.</td>
</tr>
<tr>
<td>Reviewers group</td>
<td>Réviseur</td>
<td>Or Review with per-project access control.</td>
</tr>
</tbody>
</table>
1.8.6 Turning on reviews

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

- **Set "Language-Team" header**
  Lets Weblate update the "Language-Team" file header of your project.
- **Use shared translation memory**
  Uses the pool of shared translations between projects.
- **Contribute to shared translation memory**
  Contributes to the pool of shared translations between projects.
- **Enable hooks**
  Whether to allow updating this repository by remote hooks.

1.8.7 Quality gateway for the source strings

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

By setting *Fichier de langue intermédiaire*, this file will be used as source for the strings, but it will be edited to source language to polish it. Once the string is ready in the source language, it will be also available for translators to translate into additional languages.
Voir aussi :

Fichier de langue intermédiaire, Fichier de langue de base mono-langue, Formats monolingues et bilingues

### 1.8.8 Relectures des chaînes sources

With Activer la révision des chaînes sources enabled, the review process can be applied on the source strings. Once enabled, users can report issues in the source strings. The actual process depends on whether you use bilingual or monolingual formats.

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

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

Voir aussi :

Formats monolingues et bilingues, Dedicated reviewers, labels, Commentaires

### 1.9 Foire aux questions

#### 1.9.1 Configuration

Comment créer un flux de travail automatisé ?

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

1. Set up your Git repository to tell Weblate when there is any change, see Déclencheurs de notification for info on how to do it.
2. Set a push URL at your Configuration des composants in Weblate, this allows Weblate to push changes to your repository.
3. Turn on Pousser lors du commit on your Configuration des composants in Weblate, this will make Weblate push changes to your repository whenever they happen at Weblate.

Voir aussi :

Traduction en continu, Avoiding merge conflicts
How to access repositories over SSH?

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

How to fix merge conflicts in translations?

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

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

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

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

# Merge Weblate changes:
git merge weblate/main

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

# Rebase changes (if Weblate is configured to do rebases)
git rebase origin/main

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

# Open Weblate for translation:
wlc unlock
```

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

```bash
# 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
```

(suite sur la page suivante)
In case of gettext PO files, there is a way to merge conflicts in a semi-automatic way:

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

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

# Update Weblate remote:
git remote update weblate

# Merge Weblate changes:
git merge weblate/main

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

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

Voir aussi :

- How to export the Git repository that Weblate uses?
- Traduction en continu
- Avoiding merge conflicts
- Client Weblate

### How do I translate several branches at once?

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

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

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

Voir aussi :

- Keeping translations same across components
How to translate multi-platform projects?

Weblate supports a wide range of file formats (see Formats de fichiers pris en charge) and the easiest approach is to use the native format for each platform.

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

Voir aussi :

Keeping translations same across components

How to export the Git repository that Weblate uses?

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

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

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

What are the options for pushing changes back upstream?

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

— Weblate automatically pushes and merges changes (see Comment créer un flux de travail automatisé?).
— You manually tell Weblate to push (it needs push access to the upstream repository).
— Somebody manually merges changes from the Weblate git repository into the upstream repository.
— Somebody rewrites history produced by Weblate (e.g. by eliminating merge commits), merges changes, and tells Weblate to reset the content in the upstream repository.

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

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

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

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

```
    git submodule add git@example.com:project-translations.git path/to/translations
```
3. Link Weblate to this repository, it no longer needs access to the repository containing your source code.
4. You can update the main repository with translations from Weblate by:

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

Please consult the git submodule documentation for more details.
How can I check whether my Weblate is set up properly?

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

Voir aussi :
Surveiller Weblate, Monitoring Celery status

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

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

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

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

Voir aussi :
Configuration des composants

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

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

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

1. Verrouiller le composant dans Weblate.
2. Valider et fusionner les modifications en attente dans le dépôt.
3. Disable receiving webhooks the Configuration du projet; this prevents Weblate from immediately seeing changes in the repository.
4. Do any needed changes in the repo (for example using git mv), push them to the upstream repository.
5. Change the Configuration des composants to match the new setup; upon changing configuration, Weblate will fetch the updated repository and notice the changed locations while keeping existing strings.
6. Unlock the component and re-enable hooks in the project configuration.

1.9.2 Utilisation

How do I review the translations of others?

— There are several review based workflows available in Weblate, see Flux de travail de traduction.
— You can subscribe to any changes made in Notifications and then check others contributions as they come in by e-mail.
— There is a review tool available at the bottom of the translation view, where you can choose to browse translations made by others since a given date.

Voir aussi :
Flux de travail de traduction
How do I provide feedback on a source string?

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

Voir aussi:
report-source, Commentaires

How can I use existing translations while translating?

— Toutes les contributions dans Weblate peuvent être utilisées grâce au mémoire de traduction partagé.
— Vous pouvez importer un fichier de mémoire de traduction existant dans Weblate.
— Use the import functionality to load compendium as translations, suggestions or translations needing review. This is the best approach for a one-time translation using a compendium or a similar translation database.
— You can set up tmserver with all databases you have and let Weblate use it. This is good when you want to use it several times during translation.
— Another option is to translate all related projects in a single Weblate instance, which will make it automatically pick up translations from other projects as well.

Voir aussi:
Configuring automatic suggestions, Suggestions automatiques, Mémoire de traduction

Does Weblate update translation files besides translations?

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

Voir aussi:
updating-target-files

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

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

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

Voir aussi:
Définitions de langue

Can Weblate highlight changes in a fuzzy string?

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

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

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

For monolingual translations, Weblate can find the previous string by ID, so it shows the differences automatically.
Why does Weblate still show old translation strings when I’ve updated the template?

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

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

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

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

In case you want to do the update automatically, you can install add-on *Mettre à jour les fichiers PO afin qu’ils correspondent au POT* (msgmerge).

**Voir aussi:**

updating-target-files

### 1.9.3 Dépannage

Requests sometimes fail with « too many open files » error

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

The easiest way to do this is to run:

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

**Voir aussi:**

DATA_DIR

When accessing the site I get a « Bad Request (400) » error

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

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

**Voir aussi:**

Allowed hosts setup

1.9. Foire aux questions
What does mean « There are more files for the single language (en) »?

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

— In case the translation to the source language is desired, please change the Langue source in the component settings. You might want to use English (Developer) as a source language, or utilize Quality gateway for the source strings.
— Si le fichier de traduction de la langue source n’est pas nécessaire, veuillez le supprimer du dépôt.
— Si le fichier de traduction de la langue source est nécessaire, mais qu’il doit être ignoré par Weblate, veuillez ajuster le Filtre sur la langue pour l’exclure.

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

This can be caused by using obsolete language codes together with new one (ja and jp for Japanese) or including both country specific and generic codes (fr and fr_FR). See Parsing language codes for more details.

1.9.4 Fonctionnalités

Does Weblate support other VCSes than Git and Mercurial?

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

You can also use Assistants distants Git in Git to access other VCSes.

Weblate also supports VCS-less operation, see Fichiers locaux.

Note : For native support of other VCSes, Weblate requires using distributed VCS, and could probably be adjusted to work with anything other than Git and Mercurial, but somebody has to implement this support.

Voir aussi :

Intégration avec le système de contrôle de versions

How does Weblate credit translators?

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

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

Voir aussi :

list_translators, ./devel/reporting

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

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

Modifié dans la version 4.2 : Les traducteurs peuvent traduire l’ensemble des composants d’un projet dans une langue spécifique.
Why does Weblate use language codes such sr_Latn or zh_Hant?

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

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

**Note**: Weblate defaults to POSIX style language codes with underscore, see *Définitions de langue* for more details.

Voir aussi:

*Définitions de langue*, *Style de code-langue*, Adding new translations

### 1.10 Formats de fichiers pris en charge

Weblate prend en charge la plupart des formats de traduction compris par translate-toolkit, cependant chaque format étant légèrement différent, certains problèmes peuvent survenir avec des formats peu testés.

**Voir aussi**:

Translation Related File Formats

**Note**: Lorsque vous choisissez un format de fichier pour votre demande, il est préférable de conserver un format bien établi dans la boîte à outils/plateforme que vous utilisez. De cette façon, vos traducteurs pourront utiliser les outils auxquels ils sont habitués et contribueront plus probablement à votre projet.

#### 1.10.1 Formats monolingues et bilingues

Les formats monolingual et bilingual sont pris en charge. Les formats bilingues stockent deux langues dans un seul fichier — source et traduction (exemples principaux GNU gettext, XLIFF ou Apple iOS strings). À l'opposé, les formats mono-langue identifient la chaîne par un identifiant, et chaque fichier de langue contient uniquement la traduction correspondante pour une langue donnée (exemple Android string resources). Certains formats de fichiers sont utilisés dans les deux variantes, veuillez consulter la description détaillée ci-dessous.

Pour une utilisation correcte des fichiers monolingues, Weblate nécessite l'accès à un fichier contenant la liste complète des chaînes de caractères à traduire avec leur source — ce fichier est appelé *Fichier de langue de base mono-langue* dans Weblate, bien que le nom puisse varier selon votre paradigme.

De plus, ce flux de travail peut être étendu en utilisant un *Fichier de langue intermédiaire* pour inclure des chaînes fournies par les développeurs, mais qui ne doivent pas être utilisées telles quelles dans les chaînes finales.

#### 1.10.2 Détection automatique

Weblate peut détecter automatiquement plusieurs formats de fichiers répandus, mais cette détection peut nuire aux performances et limitera les fonctionnalités spécifiques à un format de fichier donné (par exemple l’ajout automatique de nouvelles traductions).
### 1.10.3 Fonctionnalités des types de traduction

Fonctionnalités de chacun des formats pris en charge :

<table>
<thead>
<tr>
<th>Format</th>
<th>Type</th>
<th>Pluriels</th>
<th>Descriptions</th>
<th>Contexte</th>
<th>Emplacement</th>
<th>Drapage</th>
<th>États supplémentaires</th>
</tr>
</thead>
<tbody>
<tr>
<td>GNU gettext</td>
<td>bilingue</td>
<td>oui</td>
<td>oui</td>
<td>oui</td>
<td>oui</td>
<td>oui^9</td>
<td>à vérifier</td>
</tr>
<tr>
<td>Gettext monolingue</td>
<td>monolingue</td>
<td>oui</td>
<td>oui</td>
<td>oui</td>
<td>oui</td>
<td>oui^9</td>
<td>à vérifier</td>
</tr>
<tr>
<td>XLIFF</td>
<td>les deux</td>
<td>oui</td>
<td>oui</td>
<td>oui</td>
<td>oui</td>
<td>oui^10</td>
<td>à vérifier, approuver</td>
</tr>
<tr>
<td>Java properties</td>
<td>les deux</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>mi18n lang files</td>
<td>monolingue</td>
<td>non</td>
<td>oui</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>Propriétés GWT</td>
<td>monolingue</td>
<td>oui</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>Joomla translations</td>
<td>monolingue</td>
<td>non</td>
<td>oui</td>
<td>non</td>
<td>oui</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>Qt Linguist.ts</td>
<td>les deux</td>
<td>oui</td>
<td>oui</td>
<td>non</td>
<td>oui</td>
<td>oui^10</td>
<td>à vérifier</td>
</tr>
<tr>
<td>Android string resources</td>
<td>monolingue</td>
<td>oui</td>
<td>oui^7</td>
<td>non</td>
<td>non</td>
<td>oui^7</td>
<td></td>
</tr>
<tr>
<td>Apple iOS strings</td>
<td>les deux</td>
<td>non</td>
<td>oui</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>Chaînes de caractères PHP</td>
<td>monolingue</td>
<td>non^11</td>
<td>oui</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>JSON files</td>
<td>monolingue</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>JSON i18nnext files</td>
<td>monolingue</td>
<td>oui</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>go-i18n JSON files</td>
<td>monolingue</td>
<td>oui</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>go18next JSON files</td>
<td>monolingue</td>
<td>oui</td>
<td>oui</td>
<td>oui</td>
<td>non</td>
<td>oui</td>
<td>non</td>
</tr>
<tr>
<td>ARB File</td>
<td>monolingue</td>
<td>oui</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>WebExtension JSON</td>
<td>monolingue</td>
<td>oui</td>
<td>oui</td>
<td>oui</td>
<td>oui</td>
<td>oui</td>
<td>non</td>
</tr>
<tr>
<td>XML resource files</td>
<td>monolingue</td>
<td>non</td>
<td>oui</td>
<td>non</td>
<td>non</td>
<td>oui^10</td>
<td></td>
</tr>
<tr>
<td>Resource-Dictionary files</td>
<td>monolingue</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>oui^10</td>
<td></td>
</tr>
<tr>
<td>CSV files</td>
<td>les deux</td>
<td>non</td>
<td>oui</td>
<td>oui</td>
<td>oui</td>
<td>oui</td>
<td>non</td>
</tr>
<tr>
<td>YAML files</td>
<td>monolingue</td>
<td>non</td>
<td>oui</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
</tbody>
</table>

suite sur la page suivante
Tableau 1 – suite de la page précédente

<table>
<thead>
<tr>
<th>Format</th>
<th>Type</th>
<th>Pluriels</th>
<th>Description</th>
<th>Contexte</th>
<th>Emplacement</th>
<th>Drapeaux</th>
<th>États supplémentaires</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ruby YAML files</td>
<td>mono-lingue</td>
<td>oui</td>
<td>oui</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>DTD files</td>
<td>mono-lingue</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>Flat XML files</td>
<td>mono-lingue</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>oui</td>
<td>oui</td>
<td>Page 73, 10</td>
</tr>
<tr>
<td>Windows RC files</td>
<td>mono-lingue</td>
<td>non</td>
<td>oui</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>Excel Open XML</td>
<td>mono-lingue</td>
<td>non</td>
<td>oui</td>
<td>oui</td>
<td>oui</td>
<td>non</td>
<td>à vérifier</td>
</tr>
<tr>
<td>Fichiers de métdonnées de l’App Store</td>
<td>mono-lingue</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>Subtitle files</td>
<td>mono-lingue</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>oui</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>HTML files</td>
<td>mono-lingue</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>OpenDocument Format</td>
<td>mono-lingue</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>IDML Format</td>
<td>mono-lingue</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>INI translations</td>
<td>mono-lingue</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>Traduction des fichier INI InnoSetup</td>
<td>mono-lingue</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>TermBase eXchange format</td>
<td>bilingue</td>
<td>non</td>
<td>oui</td>
<td>non</td>
<td>non</td>
<td>oui</td>
<td>10</td>
</tr>
<tr>
<td>Fichiers texte</td>
<td>mono-lingue</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>Stringsdict format</td>
<td>mono-lingue</td>
<td>oui</td>
<td>oui</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
<tr>
<td>Fluent format</td>
<td>mono-lingue</td>
<td>no 12</td>
<td>oui</td>
<td>non</td>
<td>non</td>
<td>non</td>
<td>non</td>
</tr>
</tbody>
</table>

1. Voir Formats monolingues et bilingues
2. Les pluriels sont nécessaires pour traduire correctement les chaînes de caractères à nombre variable.
3. Source string descriptions can be used to pass additional info about the string to translate.
4. Le contexte est utilisé pour différencier des chaînes identiques utilisées dans des endroits différents (par exemple Mer peut être utilisé comme abrégé de « mercredi » ou comme nom pour une étendue d’eau).
5. L’emplacement d’une chaîne dans le code source peut aider les traducteurs compétents à comprendre comment la chaîne est utilisée.
6. Voir Customizing behavior using flags
7. XML comment placed before the <string> element, parsed as a source string description.
8. Additional states supported by the file format in addition to « Untranslated » and « Translated ».
9. Les commentaires de type gettext sont utilisés comme des drapeaux.
10. Les drapeaux sont extraits de l’attribut non standard weblate-flags pour tous les formats basés sur XML. En outre, l’attribut max-length:N est pris en charge à travers l’attribut maxWidth attribute tel que défini dans la norme XLIFF, voir Specifying translation flags.
11. Les commentaires de type gettext sont utilisés comme des drapeaux.
12. Les pluriels ne sont pris en charge que pour Laravel qui les utilise dans la syntaxe des chaînes de caractères pour les définir, voir Localization in Laravel.

1.10. Formats de fichiers pris en charge
Chaines en lecture seule

Nouveau dans la version 3.10.

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

1.10.4 GNU gettext

Format le plus largement utilisé pour la traduction des logiciels libres.

Les informations contextuelles stockées dans le fichier sont prises en charge en ajustant ses en-têtes ou en établissant
des liens avec les fichiers sources correspondants.

Un fichier PO gettext bilingue ressemble généralement à ceci :

```ruby
#: 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"
mmsgid "None"
mmsgstr "Žádný"
```

**Configuration des composants typique de Weblate**

<table>
<thead>
<tr>
<th>Masque de fichier</th>
<th>po/*.po</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fichier de langue de base mono-langue</td>
<td>Vide</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
<td>po/messages.pot</td>
</tr>
<tr>
<td>Format de fichier</td>
<td>Fichier gettext PO</td>
</tr>
</tbody>
</table>

Voir aussi :

devel/gettext, devel/sphinx, Gettext on Wikipedia, PO Files, Mettre à jour la variable ALL_LINGUAS dans le fichier « configure », Personnaliser la sortie gettext, Mettre à jour le fichier LINGUAS, Générer des fichiers MO, Mettre à jour les fichiers PO afin qu’ils correspondent au POT (msgmerge)

Gettext monolingue

Certains projets décident d’utiliser gettext comme format monolingue — ils codifient uniquement les identifiants dans
leur code source et la chaîne doit ensuite être traduite dans toutes les langues, y compris en anglais. Cela est pris en
charge, mais il est nécessaire d’explicitement choisir ce format de fichier lors de l’importation du composant dans
Weblate.

Un fichier gettext PO monolingue ressemble généralement à ceci :

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

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

(suite sur la page suivante)
Tandis que le fichier de langue de base ressemblera à ceci :

```python
#: 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"
```

### Configuration des composants typique de Weblate

<table>
<thead>
<tr>
<th>Masque de fichier</th>
<th>po/<em>.</em>.po</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fichier de langue de base mono-langue</td>
<td>po/en.po</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
<td>po/messages.pot</td>
</tr>
<tr>
<td>Format de fichier</td>
<td>Fichier gettext PO (monolingue)</td>
</tr>
</tbody>
</table>

### 1.10.5 XLIFF

Format basé sur XML créé pour normaliser les fichiers de traduction, mais en fin de compte, ce n’est qu’une des trop nombreuses normes du domaine.

XML Localization Interchange File Format (XLIFF) est généralement utilisé comme fichier bilingue, mais Weblate prend également en charge la version monolingue.

Weblate supporte le format XLIFF dans plusieurs variantes :

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

- **XLIFF with placeables support**
  
  Standard XLIFF supporting placeables and other XML elements.

- **XLIFF with gettext extensions**
  
  XLIFF enriched by XLIFF 1.2 Representation Guide for Gettext PO to support plurals.

**Voir aussi :**

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

### États de traduction

Modifié dans la version 3.3 : Weblate ignored the state attribute prior to the 3.3 release.

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

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

While saving, Weblate doesn’t add those attributes unless necessary.
The state attribute is only added in case string is marked as needing edit.
— The approved attribute is only added in case string has been reviewed.
— In other cases the attributes are not added, but they are updated in case they are present.

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

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

Voir aussi :

Dedicated reviewers

Whitespace and newlines in XLIFF

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

Par exemple :

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

Specifying translation flags

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

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

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

Clés de chaîne

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

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

<table>
<thead>
<tr>
<th>Typical Weblate</th>
<th>Configuration des composants for bilingual XLIFF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
<td>localizations/*_.xliff</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
<td>Vide</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
<td>localizations/en-*.xliff</td>
</tr>
<tr>
<td>Format de fichier</td>
<td>XLIFF Translation File</td>
</tr>
</tbody>
</table>
Typical Weblate Configuration des composants for monolingual XLIFF

<table>
<thead>
<tr>
<th>Masque de fichier</th>
<th>localizations/*xliff</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fichier de langue de base mono-langue</td>
<td>localizations/en-US.xliff</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
<td>localizations/en-US.xliff</td>
</tr>
<tr>
<td>Format de fichier</td>
<td>XLIFF Translation File</td>
</tr>
</tbody>
</table>

Voir aussi :

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

1.10.6 Java properties

Native Java format for translations.

Java properties are usually used as monolingual translations.

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

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

<table>
<thead>
<tr>
<th>Configuration des composants typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
</tr>
<tr>
<td>Format de fichier</td>
</tr>
</tbody>
</table>

Voir aussi :

Java properties on Wikipedia, Mozilla and Java properties files, mi18n lang files, Propriétés GWT, updating-target-files, Formater le fichier de propriétés Java, Nettoyer les fichiers de traduction

1.10.7 mi18n lang files

Nouveau dans la version 4.7.

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

<table>
<thead>
<tr>
<th>Configuration des composants typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
</tr>
<tr>
<td>Format de fichier</td>
</tr>
</tbody>
</table>

Voir aussi :

mi18n, Mozilla and Java properties files, Java properties, updating-target-files, Formater le fichier de propriétés Java, Nettoyer les fichiers de traduction

1.10. Formats de fichiers pris en charge
1.10.8 Propriétés GWT

Native GWT format for translations.

GWT properties are usually used as monolingual translations.

<table>
<thead>
<tr>
<th>Configuration des composants typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
</tr>
<tr>
<td>Format de fichier</td>
</tr>
</tbody>
</table>

Voir aussi :
- GWT localization guide, GWT Internationalization Tutorial, Mozilla and Java properties files, updating-target-files, Formater le fichier de propriétés Java, Nettoyer les fichiers de traduction

1.10.9 INI translations

Nouveau dans la version 4.1.

INI file format for translations.

INI translations are usually used as monolingual translations.

<table>
<thead>
<tr>
<th>Configuration des composants typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
</tr>
<tr>
<td>Format de fichier</td>
</tr>
</tbody>
</table>

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

Voir aussi :
- INI Files, Java properties, Joomla translations, Traduction des fichier INI InnoSetup

1.10.10 Traduction des fichier INI InnoSetup

Nouveau dans la version 4.1.

Format des fichiers Inno Setup INI pour les traductions.

Les traductions de fichiers Inno Setup INI sont généralement utilisées comme des traductions mono-langue.

Note : The only notable difference to INI translations is in supporting %n and %t placeholders for line break and tab.
Note : Only Unicode files (.islu) are currently supported, ANSI variant (.isl) is currently not supported.

Voir aussi :
INI Files, Joomla translations, INI translations

1.10.11 Joomla translations

Nouveau dans la version 2.12.
Native Joomla format for translations.
Joomla translations are usually used as monolingual translations.

<table>
<thead>
<tr>
<th>Configuration des composants</th>
<th>Typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
<td>language/*/com_foobar.ini</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
<td>language/en-GB/com_foobar.ini</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
<td>Vide</td>
</tr>
<tr>
<td>Format de fichier</td>
<td>Joomla Language File</td>
</tr>
</tbody>
</table>

Voir aussi :
Mozilla and Java properties files, INI translations, Traduction des fichier INI InnoSetup

1.10.12 Qt Linguist .ts

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

| Typical Weblate Configuration des composants when using as bilingual |
|---------------------------|------------------|
| Masque de fichier         | i18n/app.*.ts |
| Fichier de langue de base mono-langue | Vide |
| Modèle pour les nouvelles traductions | i18n/app.de.ts |
| Format de fichier         | Qt Linguist Translation File |

| Typical Weblate Configuration des composants when using as monolingual |
|---------------------------|------------------|
| Masque de fichier         | i18n/app.*.ts |
| Fichier de langue de base mono-langue | i18n/app.en.ts |
| Modèle pour les nouvelles traductions | i18n/app.en.ts |
| Format de fichier         | Qt Linguist Translation File |

Voir aussi :
Qt Linguist manual, Qt .ts, Formats monolingues et bilingues

1.10. Formats de fichiers pris en charge
1.10.13 Android string resources

Android specific file format for translating applications.

Android string resources are monolingual, the *Fichier de langue de base mono-langue* is stored in a different location from the other files – res/values/strings.xml.

<table>
<thead>
<tr>
<th>Configuration des composants</th>
<th>typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
<td>res/values-*/strings.xml</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
<td>res/values/strings.xml</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
<td>Vide</td>
</tr>
<tr>
<td>Format de fichier</td>
<td>Android String Resource</td>
</tr>
</tbody>
</table>

**Configuration**

Masque de fichier : res/values-*/strings.xml

Fichier de langue de base mono-langue : res/values/strings.xml

Modèle pour les nouvelles traductions : Vide

Format de fichier : Android String Resource

**Voir aussi :**

Android string resources documentation, Android string resources

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

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

devient:

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

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

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

**Indication :** To avoid translating some strings, these can be marked as non-translatable. This can be especially useful for string references :

```xml
<string name="foobar" translatable="false">@string/foo</string>
```
1.10.14 Apple iOS strings

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

Apple iOS strings are usually used as monolingual.

<table>
<thead>
<tr>
<th>Configuration des composants typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
</tr>
<tr>
<td>Format de fichier</td>
</tr>
</tbody>
</table>

Voir aussi :
Stringsdict format, Apple « strings files » documentation, Message Catalog File Format in PWG 5100.13, Mac OSX strings

1.10.15 Chaînes de caractères PHP

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

Example file :

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

<table>
<thead>
<tr>
<th>Configuration des composants typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
</tr>
<tr>
<td>Format de fichier</td>
</tr>
</tbody>
</table>

Chaînes de caractères PHP Laravel

Modifié dans la version 4.1.

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

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

Voir aussi :
PHP, Localisation dans Laravel
1.10.16 JSON files

Nouveau dans la version 2.0.

Modifié dans la version 2.16 : Since Weblate 2.16 and with translate-toolkit at-least 2.2.4, nested structure JSON files are supported as well.

Modifié dans la version 4.3 : The structure of JSON file is properly preserved even for complex situations which were broken in prior releases.

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

Weblate currently supports several variants of JSON translations:
- Simple key / value files, used for example by vue-i18n or react-intl.
- Files with nested keys.
- JSON i18next files
- go-i18n JSON files
- gotext JSON files
- WebExtension JSON
- ARB File

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

Example file:

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

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

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

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

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

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

---

<table>
<thead>
<tr>
<th>Configuration des composants</th>
<th>typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
<td>langs/translation-*.json</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
<td>langs/translation-en.json</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
<td>Vide</td>
</tr>
<tr>
<td>Format de fichier</td>
<td>JSON nested structure file</td>
</tr>
</tbody>
</table>
Voir aussi :
JSON, updating-target-files, Personnaliser la sortie JSON, Nettoyer les fichiers de traduction,

1.10.17 JSON i18next files

Modifié dans la version 2.17 : Since Weblate 2.17 and with translate-toolkit at-least 2.2.5, i18next JSON files with plurals are supported as well.

Modifié dans la version 4.15.1 : Support for v4 variant of this format was added.

Indication : In case you use plurals, it is recommended to use v4 as that aligned plural handling with CLDR. Older versions have different plural rules for some languages which are not correct.

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

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

Note : Weblate supports the i18next JSON v3 and v4 variants. Please choose correct file format matching your environment.

The v2 and v1 variants are mostly compatible with v3, with exception of how plurals are handled.

Example file :

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

<table>
<thead>
<tr>
<th>Configuration des composants typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
</tr>
<tr>
<td>Format de fichier</td>
</tr>
</tbody>
</table>

Voir aussi :
JSON, i18next JSON Format, updating-target-files, Personnaliser la sortie JSON, Nettoyer les fichiers de traduction

1.10.18 go-i18n JSON files

Nouveau dans la version 4.1.

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

Modifié dans la version 4.16 : Weblate supports both v1 and v2 formats, please choose corresponding version when selecting file format.
### Configuration des composants typique de Weblate

<table>
<thead>
<tr>
<th>Masque de fichier</th>
<th>langs/*.json</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fichier de langue de base mono-langue</td>
<td>langs/en.json</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
<td>Vide</td>
</tr>
<tr>
<td>Format de fichier</td>
<td>go-i18n v1 JSON file</td>
</tr>
</tbody>
</table>

Voir aussi :

JSON, go-i18n, updating-target-files, Personnaliser la sortie JSON, Nettoyer les fichiers de traduction.

#### 1.10.19 gotext JSON files

Nouveau dans la version 4.15.1.

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

<table>
<thead>
<tr>
<th>Masque de fichier</th>
<th>internal/translations/locales/*/messages.gotext.json</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fichier de langue de base mono-langue</td>
<td>internal/translations/locales/en-GB/messages.gotext.json</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
<td>Vide</td>
</tr>
<tr>
<td>Format de fichier</td>
<td>gotext JSON file</td>
</tr>
</tbody>
</table>

Voir aussi :

JSON, I18n in Go : Managing Translations, updating-target-files, Personnaliser la sortie JSON, Nettoyer les fichiers de traduction.

#### 1.10.20 ARB File

Nouveau dans la version 4.1.

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

<table>
<thead>
<tr>
<th>Masque de fichier</th>
<th>lib/l10n/intl_*.arb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fichier de langue de base mono-langue</td>
<td>lib/l10n/intl_en.arb</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
<td>Vide</td>
</tr>
<tr>
<td>Format de fichier</td>
<td>ARB file</td>
</tr>
</tbody>
</table>

Voir aussi :

1.10.21 WebExtension JSON

Nouveau dans la version 2.16 : This is supported since Weblate 2.16 and with translate-toolkit at least 2.2.4.

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

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

Example file:

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

### Configuration des composants typique de Weblate

<table>
<thead>
<tr>
<th>Masque de fichier</th>
<th>_locales/*/messages.json</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fichier de langue de base mono-langue</td>
<td>_locales/en/messages.json</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
<td>Vide</td>
</tr>
<tr>
<td>Format de fichier</td>
<td>WebExtension JSON file</td>
</tr>
</tbody>
</table>

Voir aussi :

JSON, Google chrome.i18n, Mozilla Extensions Internationalization
### 1.10.22 .XML resource files

Nouveau dans la version 2.3.

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

<table>
<thead>
<tr>
<th>Configuration des composants typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
</tr>
<tr>
<td>Format de fichier</td>
</tr>
</tbody>
</table>

**Voir aussi :**

.NET Resource files (.resx), updating-target-files, Nettoyer les fichiers de traduction

### 1.10.23 ResourceDictionary files

Nouveau dans la version 4.13.

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

<table>
<thead>
<tr>
<th>Configuration des composants typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
</tr>
<tr>
<td>Format de fichier</td>
</tr>
</tbody>
</table>

**Voir aussi :**

Flat XML, Flat XML files, updating-target-files, Nettoyer les fichiers de traduction

### 1.10.24 CSV files

Nouveau dans la version 2.4.

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

- Files with header defining fields (location, source, target, ID, fuzzy, context, translator_comments, developer_comments). This is the recommended approach, as it is the least error prone. Choose CSV file as a file format.
- Files with two fields—source and translation (in this order). Choose Simple CSV file as a file format.
- Headerless files with fields in order defined by the translate-toolkit: location, source, target, ID, fuzzy, context, translator_comments, developer_comments. Choose CSV file as a file format.
- Remember to define Fichier de langue de base mono-langue when your files are monolingual (see Formats monolingues et bilingues).

**Indication** : By default, the CSV format does autodetection of file encoding. This can be unreliable in some corner cases and causes performance penalty. Please choose file format variant with encoding to avoid this (for example CSV file (UTF-8)).
Avertissement : The CSV format currently automatically detects the dialect of the CSV file. In some cases the automatic detection might fail and you will get mixed results. This is especially true for CSV files with newlines in the values. As a workaround it is recommended to omit quoting characters.

Example file:

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

| Configuration des composants typique de Weblate pour les fichiers CSV bilingues |
| Masque de fichier | locale/*.csv |
| Fichier de langue de base mono-langue | Vide |
| Modèle pour les nouvelles traductions | locale/en.csv |
| Format de fichier | CSV file |

Configuration des composants typique de Weblate pour les fichiers CSV monolingues

| Masque de fichier | locale/*.csv |
| Fichier de langue de base mono-langue | locale/en.csv |
| Modèle pour les nouvelles traductions | locale/en.csv |
| Format de fichier | Fichier CSV simplifié |

Multivalue CSV file

Nouveau dans la version 4.13.

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

Voir aussi :

CSV

1.10.25 YAML files

Nouveau dans la version 2.9.

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

Example of a YAML file:

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

| Configuration des composants typique de Weblate |
| Masque de fichier | translations/messages.*.yml |
| Fichier de langue de base mono-langue | translations/messages.en.yml |
| Modèle pour les nouvelles traductions | Vide |
| Format de fichier | YAML file |

Voir aussi :

YAML, Ruby YAML files
1.10.26 Ruby YAML files

Nouveau dans la version 2.9.
Ruby i18n YAML files with language as root node.
Example Ruby i18n YAML file :

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

Configuration des composants typique de Weblate

<table>
<thead>
<tr>
<th>Masque de fichier</th>
<th>translations/messages.*.yml</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fichier de langue de base mono-langued</td>
<td>translations/messages.en.yml</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
<td>Vide</td>
</tr>
<tr>
<td>Format de fichier</td>
<td>Ruby YAML file</td>
</tr>
</tbody>
</table>

Voir aussi :
YAML, YAML files

1.10.27 DTD files

Nouveau dans la version 2.18.
Example DTD file :

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

Configuration des composants typique de Weblate

<table>
<thead>
<tr>
<th>Masque de fichier</th>
<th>locale/*.dtd</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fichier de langue de base mono-langued</td>
<td>locale/en.dtd</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
<td>Vide</td>
</tr>
<tr>
<td>Format de fichier</td>
<td>DTD file</td>
</tr>
</tbody>
</table>

Voir aussi :
Mozilla DTD format
1.10.28 Flat XML files

Nouveau dans la version 3.9.

Example of a flat XML file:

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

<table>
<thead>
<tr>
<th>Configuration des composants typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
</tr>
<tr>
<td>Format de fichier</td>
</tr>
</tbody>
</table>

Voir aussi:

Flat XML

1.10.29 Windows RC files

Modifié dans la version 4.1 : Support for Windows RC files has been rewritten.

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

Example Windows RC file:

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

<table>
<thead>
<tr>
<th>Configuration des composants typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
</tr>
<tr>
<td>Format de fichier</td>
</tr>
</tbody>
</table>

Voir aussi:

Windows RC files
1.10.30 Fichiers de métadonnées de l’App Store

Nouveau dans la version 3.5.

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

- Triple-T gradle-play-publisher
- Fastlane
- F-Droid

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

<table>
<thead>
<tr>
<th>Configuration des composants</th>
<th>typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
<td>fastlane/android/metadata/*</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
<td>fastlane/android/metadata/en-US</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
<td>fastlane/android/metadata/en-US</td>
</tr>
<tr>
<td>Format de fichier</td>
<td>App store metadata files</td>
</tr>
</tbody>
</table>

**Indication:** In case you don’t want to translate certain strings (for example changelogs), mark them read-only (see *Customizing behavior using flags*). This can be automated by the *Modification en masse*.

1.10.31 Subtitle files

Nouveau dans la version 3.7.

Weblate peut traduire divers fichiers de sous-titres :

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

<table>
<thead>
<tr>
<th>Configuration des composants</th>
<th>typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
<td>path/*.srt</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
<td>path/en.srt</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
<td>path/en.srt</td>
</tr>
<tr>
<td>Format de fichier</td>
<td>SubRip subtitle file</td>
</tr>
</tbody>
</table>

**Voir aussi :**

Subtitle

1.10.32 Excel Open XML

Nouveau dans la version 3.2.

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

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

Nouveau dans la version 4.1.

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

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

**Voir aussi :**
HTML

1.10.34 Fichiers texte

Nouveau dans la version 4.6.

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

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

Il existe trois variantes de ce format :
— Fichier texte brut
— Fichier texte DokuWiki
— Fichier texte MediaWiki

**Voir aussi :**
Simple Text Documents

1.10.35 OpenDocument Format

Nouveau dans la version 4.1.

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

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

**Voir aussi :**
OpenDocument Format

1.10.36 IDML Format

Nouveau dans la version 4.1.

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

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

Nouveau dans la version 4.5.

TBX est un format XML pour l'échange de données de terminologie.

<table>
<thead>
<tr>
<th>Configuration des composants typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
</tr>
<tr>
<td>Format de fichier</td>
</tr>
</tbody>
</table>

Voir aussi :
TBX on Wikipedia, TBX, Glossaire

1.10.38 Stringsdict format

Nouveau dans la version 4.8.

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

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

<table>
<thead>
<tr>
<th>Configuration des composants typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
</tr>
<tr>
<td>Format de fichier</td>
</tr>
</tbody>
</table>

Voir aussi :
Apple iOS strings, Stringsdict File Format

1.10.39 Fluent format

Nouveau dans la version 4.8.

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

Fluent est un format texte monolingual qui se concentre sur la localisation asymétrique : une simple chaîne dans une langue peut se traduire en une complexe traduction multi-variant dans une autre langue.

<table>
<thead>
<tr>
<th>Configuration des composants typique de Weblate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Masque de fichier</td>
</tr>
<tr>
<td>Fichier de langue de base mono-langue</td>
</tr>
<tr>
<td>Modèle pour les nouvelles traductions</td>
</tr>
<tr>
<td>Format de fichier</td>
</tr>
</tbody>
</table>
Voir aussi :
Project Fluent website

1.10.40 Supporting other formats

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

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

Voir aussi :
Translation Related File Formats

1.11 Intégration avec le système de contrôle de versions

Weblate currently supports Git (with extended support for Requêtes de fusion GitHub, Requêtes de fusion GitLab, Gitea pull requests, Gerrit, Subversion and Bitbucket Server pull requests) and Mercurial as version control back-ends.

1.11.1 Accessing repositories

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

Accessing repositories from Hosted Weblate

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

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

Once the weblate user is added, you can configure Dépôt du code source and URL pour l'envoi du dépôt using the SSH protocol (for example git@github.com:WeblateOrg/weblate.git).

SSH repositories

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

Avertissement : On GitHub, each key can only be used once, see Dépôts GitHub and Accessing repositories from Hosted Weblate.

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

In case adjustment is needed, do so from the Weblate admin interface.
The Weblate SSH key

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

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

Note: The corresponding private SSH key cannot currently have a password, so make sure it is well protected.

Indication: Make a backup of the generated private Weblate SSH key.
Verifying SSH host keys

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

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

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

- Added key for github.com with fingerprint p2QAMKNC1YJYWeOT8z/L3/BuCuGCUbQm (ecdsa-sha2-nistp256), please verify that it is correct.
- Added key for github.com with fingerprint nThbg5tXUpJj0EIIkGOCspRomTwCARJuUkW655Y8 (ssh-rsa), please verify that it is correct.
- Added key for github.com with fingerprint +DDY3wvV67u1Jbhp2sz/fL3/Ab2P5sK/2/hx4KJjO (ssh-ec25519), please verify that it is correct.

---

1.11. Intégration avec le système de contrôle de versions
Dépôts GitHub

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

In case the `Pousser la branche` is not set, the project is forked and changes pushed through a fork. In case it is set, changes are pushed to the upstream repository and chosen branch.

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

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

Voir aussi :

Accessing repositories from Hosted Weblate

URLs internes de Weblate

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

**Avertissement** : La suppression du composant principal entraîne la suppression des composants liés.

Weblate ajuste automatiquement l'URL du dépôt lors de la création d’un composant s’il trouve un composant avec une configuration de dépôt correspondante. Vous pouvez la remplacer dans la dernière étape de la configuration.

Raisons de l’utiliser :

— Économise de l’espace disque sur le serveur, le dépôt n’est stocké qu’une seule fois.  
— Rend les mises à jour plus rapides, un seul dépôt est mis à jour.  
— Il n’y a qu’un seul dépôt exporté avec les traductions Weblate (voir `Exportateur Git`).  
— Certains greffons peuvent fonctionner sur plusieurs composants partageant un référentiel, par exemple `Squash les commits Git`.

Dépôts HTTPS

Pour accéder à des dépôts protégés en HTTPS, ajoutez le nom d’utilisateur et le mot de passe dans l’URL. Ne vous inquiétez pas, Weblate supprimera ces informations lorsque l’URL sera montrée aux utilisateurs (s’ils sont autorisés à voir l’URL du dépôt).

Par exemple, l’ajout de l’authentification dans l’URL de GitHub peut ressembler à : `https://user:your_access_token@github.com/WeblateOrg/weblate.git`.

**Note** : Si votre nom d’utilisateur ou votre mot de passe contient des caractères spéciaux, ceux-ci doivent être codés en URL, par exemple `https://user%40example.com:%24password%23@bitbucket.org/...`
Utilisation d’un proxy

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

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

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

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

Voir aussi :

The cURL manpage, Git config documentation

1.11.2 Git

**Indication** : Weblate nécessite Git 2.12 ou plus récent.

Voir aussi :

Voir Accessing repositories pour des informations sur la façon d’accéder aux différents types de dépôts.

Git avec force push

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

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

Personnalisation de la configuration Git

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

Assistants distants Git

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

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

Une fois que vous les avez installés, ces commandes peuvent être utilisées pour spécifier un dépôt dans Webhate.

Pour cloner le projet `gnuhello` depuis Launchpad en utilisant Bazaar :

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

Pour le dépôt `hello` deselenic.com en utilisant Mercurial :

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

1.11.3 Requêtes de fusion GitHub

Nouveau dans la version 2.3.

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

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

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

Voir aussi :
Pushing changes from Weblate, GITHUB_CREDENTIALS

1.11.4 Requêtes de fusion GitLab

Nouveau dans la version 3.9.

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

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

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

Voir aussi :
Pushing changes from Weblate, GITLAB_CREDENTIALS

1.11.5 Gitea pull requests

Nouveau dans la version 4.12.

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

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

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

Voir aussi :
Pushing changes from Weblate, GITEA_CREDENTIALS
1.11.6 Bitbucket Server pull requests

Nouveau dans la version 4.16.

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

**Avertissement** : This does not support Bitbucket Cloud API.

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

You need to configure API credentials (**BITBUCKETSERVER_CREDENTIALS**) in the Weblate settings to make this work. Once configured, you will see a Bitbucket Server option when selecting Système de contrôle de version.

**Voir aussi** :

*Pushing changes from Weblate, BITBUCKETSERVER_CREDENTIALS*

1.11.7 Requêtes de fusion Pagure

Nouveau dans la version 4.3.2.

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

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

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

**Voir aussi** :

*Pushing changes from Weblate, PAGURE_CREDENTIALS*

1.11.8 Gerrit

Nouveau dans la version 2.2.

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

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

1.11.9 Mercurial

Nouveau dans la version 2.1.

Mercurial is another VCS you can use directly in Weblate.

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

**Voir aussi** :

Voir Accessing repositories pour des informations sur la façon d’accéder aux différents types de dépots.
1.11.10 Subversion

Nouveau dans la version 2.8.

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

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

Modifié dans la version 2.19 : Before this, only repositories using the standard layout were supported.

### Identifiants pour Subversion

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

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

**Voir aussi :**

* `DATA_DIR`

1.11.11 Fichiers locaux

1.11.12 Git

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

Nouveau dans la version 3.8.

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

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

1.12 API REST de Weblate

Nouveau dans la version 2.6 : The REST API is available since Weblate 2.6.

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

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

```
ANY /
```

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

**Paramètres de requête**

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

**En-têtes de requête**

- `Accept` - the response content type depends on `Accept` header
- `Authorization` - optional token to authenticate as `Authorization: Token YOUR-TOKEN`

**En-têtes de réponse**

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

**Objet JSON de réponse**

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

**Codes d’état**

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

**Jetons d’authentification**

Modifié dans la version 4.10 : Project scoped tokens were introduced in the 4.10 release.

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

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

Exemple de requête :

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

Exemple de réponse :

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/"
}

Exemple CURL :

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

Passing Parameters Examples

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

Form request example :

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

operation=pull

JSON request example :

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

{"operation":"pull"}

Exemple CURL :

CURL JSON example :

```bash
```

**API rate limiting**

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

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

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

The status of rate limiting is reported in following headers:

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

Modifié dans la version 4.1 : Added ratelimiting status headers.

Voir aussi :

* Limite de requêtes, Limite de requêtes, WEBLATE_API_RATELIMIT_ANON, WEBLATE_API_RATELIMIT_USER

### 1.12.2 API Entry Point

**GET /api/**

The API root entry point.

**Exemple de requête :**

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

**Exemple de réponse :**

```
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
```
1.12.3 Utilisateurs

Nouveau dans la version 4.0.

**GET /api/users/**

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

**Voir aussi :**

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

**POST /api/users/**

Creates a new user.

**Paramètres**

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

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

Returns information about users.

**Paramètres**

- **username** *(string)* – Nom d’utilisateur de l’utilisateur

**Objet JSON de réponse**

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

**Example JSON data :**

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

Paramètres
— username (string) – Nom d'utilisateur de l'utilisateur

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

PATCH /api/users/(str: username) /
Changes the user parameters.

Paramètres
— username (string) – Nom d'utilisateur de l'utilisateur

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

DELETE /api/users/(str: username) /
Deletes all user information and marks the user inactive.

Paramètres
— username (string) – Nom d'utilisateur de l'utilisateur

POST /api/users/(str: username)/groups/
Associate groups with a user.

Paramètres
— username (string) – Nom d'utilisateur de l'utilisateur

Paramètres de la forme
— string group_id – The unique group ID

DELETE /api/users/(str: username)/groups/
Retirer l'utilisateur d'un groupe.

Paramètres
— username (string) – Nom d'utilisateur de l'utilisateur

Paramètres de la forme
— string group_id – The unique group ID

GET /api/users/(str: username)/statistics/
List statistics of a user.

Paramètres
— username (string) – Nom d'utilisateur de l'utilisateur

Objet JSON de réponse
The Weblate Manual, Version 4.16

— translated (int) — Nombre de traductions par utilisateur
— suggested (int) — Nombre de suggestions par utilisateur
— uploaded (int) — Nombre de téléversements par utilisateur
— commented (int) — Nombre de commentaires par utilisateur
— languages (int) — Nombre de langues que l'utilisateur peut traduire

GET /api/users/(str: username)/notifications/
List subscriptions of a user.

Paramètres
— username (string) — Nom d'utilisateur de l'utilisateur

POST /api/users/(str: username)/notifications/
Associate subscriptions with a user.

Paramètres
— username (string) — Nom d'utilisateur de l'utilisateur

Objet JSON de requête
— notification (string) — Name of notification registered
— scope (int) — Scope of notification from the available choices
— frequency (int) — Frequency choices for notifications

GET /api/users/(str: username)/notifications/
int: subscription_id/
Get a subscription associated with a user.

Paramètres
— username (string) — Nom d'utilisateur de l'utilisateur
— subscription_id (int) — ID de notification enregistré

PUT /api/users/(str: username)/notifications/
int: subscription_id/
Edit a subscription associated with a user.

Paramètres
— username (string) — Nom d'utilisateur de l'utilisateur
— subscription_id (int) — ID de notification enregistré

Objet JSON de requête
— notification (string) — Name of notification registered
— scope (int) — Scope of notification from the available choices
— frequency (int) — Frequency choices for notifications

PATCH /api/users/(str: username)/notifications/
int: subscription_id/
Edit a subscription associated with a user.

Paramètres
— username (string) — Nom d'utilisateur de l'utilisateur
— subscription_id (int) — ID de notification enregistré

Objet JSON de requête
— notification (string) — Name of notification registered
— scope (int) — Scope of notification from the available choices
— frequency (int) — Frequency choices for notifications

DELETE /api/users/(str: username)/notifications/
int: subscription_id/
Delete a subscription associated with a user.

Paramètres
— username (string) — Nom d'utilisateur de l'utilisateur
— subscription_id — Name of notification registered
— subscription_id — entier
1.12.4 Groupes

Nouveau dans la version 4.0.

GET /api/groups/

Returns a list of groups if you have permissions to see manage groups. If not, then you get to see only the groups the user is a part of.

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

POST /api/groups/

Creates a new group.

Paramètres
- **name** (string) – Nom du groupe
- **project_selection** (int) – Group of project selection from given options
- **language_selection** (int) – Group of languages selected from given options
- **defining_project** (str) – link to the defining project, used for *Gestion du contrôle d'accès par projet*; see GET /api/projects/(string:project)/

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

Returns information about group.

Paramètres
- **id** (int) – ID du groupe

Objet JSON de réponse
- **name** (string) – name of a group
- **project_selection** (int) – integer corresponding to group of projects
- **language_selection** (int) – integer corresponding to group of languages
- **roles** (array) – link to associated roles; see GET /api/roles/(int:id)/
- **projects** (array) – link to associated projects; see GET /api/projects/(string:project)/
- **components** (array) – link to associated components; see GET /api/components/(string:project)/(string:component)/
- **componentlists** (array) – link to associated componentlist; see GET /api/component-lists/(str:slug)/
- **defining_project** (str) – link to the defining project, used for *Gestion du contrôle d'accès par projet*; see GET /api/projects/(string:project)/

Example JSON data :

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

Paramètres
— id (int) – ID du groupe

Objet JSON de réponse
— name (string) – name of a group
— project_selection (int) – integer corresponding to group of projects
— language_selection (int) – integer corresponding to group of languages

PATCH /api/groups/(int: id)/
Changes the group parameters.

Paramètres
— id (int) – ID du groupe

Objet JSON de réponse
— name (string) – name of a group
— project_selection (int) – integer corresponding to group of projects
— language_selection (int) – integer corresponding to group of languages

DELETE /api/groups/(int: id)/
Deletes the group.

Paramètres
— id (int) – ID du groupe

POST /api/groups/(int: id)/roles/
Associate roles with a group.

Paramètres
— id (int) – ID du groupe

Paramètres de la forme
— string role_id – The unique role ID

POST /api/groups/(int: id)/components/
Associate components with a group.

Paramètres
— id (int) – ID du groupe

Paramètres de la forme
— string component_id – The unique component ID

DELETE /api/groups/(int: id)/components/(int: component_id)
Delete component from a group.

Paramètres
— id (int) – ID du groupe
— component_id (int) – The unique component ID

POST /api/groups/(int: id)/projects/
Associate projects with a group.

Paramètres
— id (int) – ID du groupe

Paramètres de la forme
— string project_id – The unique project ID

DELETE /api/groups/(int: id)/projects/(int: project_id)
Delete project from a group.

**Paramètres**
- `id (int)` – ID du groupe
- `project_id (int)` – The unique project ID

**POST /api/groups/(int: id)/languages/**
Associate languages with a group.

**Paramètres**
- `id (int)` – ID du groupe

**Paramètres de la forme**
- `string language_code` – The unique language code

**DELETE /api/groups/(int: id)/languages/**
string: `language_code`
Delete language from a group.

**Paramètres**
- `id (int)` – ID du groupe
- `language_code (string)` – The unique language code

**POST /api/groups/(int: id)/componentlists/**
Associate componentlists with a group.

**Paramètres**
- `id (int)` – ID du groupe

**Paramètres de la forme**
- `string component_list_id` – The unique componentlist ID

**DELETE /api/groups/(int: id)/componentlists/**
`int`: `component_list_id`
Delete componentlist from a group.

**Paramètres**
- `id (int)` – ID du groupe
- `component_list_id (int)` – The unique componentlist ID

### 1.12.5 Rôles

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

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

**POST /api/roles/**
Creates a new role.

**Paramètres**
- `name (string)` – Nom du rôle
- `permissions (array)` – List of codenames of permissions

**GET /api/roles/(int: id)/**
Returns information about a role.

**Paramètres**
- `id (int)` – ID de rôle

**Objet JSON de réponse**
- `name (string)` – Nom du rôle
- `permissions (array)` – list of codenames of permissions

**Example JSON data :**
The Weblate Manual, Version 4.16

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

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

Changes the role parameters.

**Paramètres**

- id (int) – ID du rôle

**Objet JSON de réponse**

- name (string) – Nom du rôle
- permissions (array) – list of codenames of permissions

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

Changes the role parameters.

**Paramètres**

- id (int) – ID du rôle

**Objet JSON de réponse**

- name (string) – Nom du rôle
- permissions (array) – list of codenames of permissions

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

Deletes the role.

**Paramètres**

- id (int) – ID du rôle

1.12.6 Langues

GET /api/languages/

Returns a list of all languages.

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

POST /api/languages/

Creates a new language.

**Paramètres**

- code (string) – Nom de la langue
- name (string) – Nom de la langue
- direction (string) – Orientation du texte
- population (int) – Nombre de locuteurs
- plural (object) – Language plural formula and number

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

Returns information about a language.

**Paramètres**

- language (string) – Code langue

**Objet JSON de réponse**

- code (string) – Code langue
- direction (string) – Orientation du texte
- plural (object) – Object of language plural information
- aliases (array) – Array of aliases for language
Objet JSON de requête
— **population** (*int*) – Nombre de locuteurs

Example JSON data :

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

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

Paramètres
— **language** (*string*) – Code de la langue

Objet JSON de requête
— **name** (*string*) – Nom de la langue
— **direction** (*string*) – Orientation du texte
— **population** (*int*) – Nombre de locuteurs
— **plural** (*object*) – Language plural details

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

Paramètres
— **language** (*string*) – Code de la langue

Objet JSON de requête
— **name** (*string*) – Nom de la langue
— **direction** (*string*) – Orientation du texte
— **population** (*int*) – Nombre de locuteurs
— **plural** (*object*) – Language plural details

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

Paramètres
— **language** (*string*) – Code de la langue

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

Paramètres
— **language** (*string*) – Code langue

Objet JSON de réponse
— **total** (*int*) – total number of strings
1.12.7 Projets

GET /api/projects/
Retourne une liste de tous les projets.

Voir aussi :
Les attributs des objets projet sont documentés à GET /api/projects/(string:project)/.

POST /api/projects/
Nouveau dans la version 3.9.
Crée un nouveau projet.

Paramètres
---
- **name (string)** - Nom du projet
- **slug (string)** - Identifiant du projet
- **web (string)** - Site Web du projet

GET /api/projects/(string: project) /
Retourne les informations sur un projet.

Paramètres
---
- **project (string)** - URL abrégée du projet

Objet JSON de réponse
---
- **name (string)** - Nom du projet
- **slug (string)** - Projet abrégé
- **web (string)** - Site Web du projet
- **components_list_url (string)** - URL de la liste des composants; voir GET /api/projects/(string:project)/components/
- **repository_url (string)** - URL du statut du répertoire; voir GET /api/projects/(string:project)/repository/
- **changes_list_url (string)** - URL de la liste des modifications; voir GET /api/projects/(string:project)/changes/
- **translation_review (boolean)** - Activer les révisions
- **source_review (boolean)** - Activer la révision des chaines sources
- **set_language_team (boolean)** - Définir l'en-tête «Language-Team»
- **enable_hooks (boolean)** - Activer les points d'ancrage
- **instructions (string)** - Directives de traduction
- **language_aliases (string)** - Alias de langue

Exemple JSON data :

```json
{
  "name": "Hello",
  "slug": "hello",
  "url": "http://example.com/api/projects/hello/",
}
```

(suite sur la page suivante)
PATCH /api/projects/(string: project) /
Nouveau dans la version 4.3.
Edit a project by a PATCH request.

Paramètres
— project (string) – URL abrégée du projet
— component (string) – URL abrégée du composant

PUT /api/projects/(string: project) /
Nouveau dans la version 4.3.
Edit a project by a PUT request.

DELETE /api/projects/(string: project) /
Nouveau dans la version 3.9.
Deletes a project.

GET /api/projects/(string: project) /changes/
Returns a list of project changes. This is essentially a project scoped GET /api/changes/ accepting same params.

Paramètres
— project (string) – URL abrégée du projet

Objet JSON de réponse
— results (array) – array of component objects; see GET /api/changes/(int:id)/

GET /api/projects/(string: project) /repository/
Returns information about VCS repository status. This endpoint contains only an overall summary for all repositories for the project. To get more detailed status use GET /api/components/(string:project)/ (string:component) /repository/.

Paramètres
— project (string) – URL abrégée du projet

Objet JSON de réponse
— needs_commit (boolean) – whether there are any pending changes to commit
— needs_merge (boolean) – whether there are any upstream changes to merge
— needs_push (boolean) – whether there are any local changes to push

Example JSON data :
```
{
  "needs_commit": true,
  "needs_merge": false,
  "needs_push": true
}
```

POST /api/projects/(string: project) /repository/
Performs given operation on the VCS repository.

Paramètres
— project (string) – URL abrégée du projet

Objet JSON de requête
— **operation**(*string*) – Operation to perform: one of push, pull, commit, reset, cleanup, file-sync

**Objet JSON de réponse**
— **result**(*boolean*) – result of the operation

**Exemple CURL**:

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

**JSON request example**:

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

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

**JSON response example**:

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

{"result":true}
```

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

Returns a list of translation components in the given project.

**Paramètres**
— **project**(*string*) – URL abrégée du projet

**Objet JSON de réponse**
— **results**(*array*) – array of component objects; see GET /api/components/(string:project)/(string:component)/

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

Nouveau dans la version 3.9.

Modifié dans la version 4.3: The zipfile and docfile parameters are now accepted for VCS-less components, see [Fichiers locaux].

Modifié dans la version 4.6: The cloned repositories are now automatically shared within a project using URLs internes de Weblate. Use disable_autoshare to turn off this.

Creates translation components in the given project.

**Indication**: Use URLs internes de Weblate when creating multiple components from a single VCS repository.

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

**Paramètres**
— **project**(*string*) – URL abrégée du projet
Paramètres de la forme

- **file zipfile** – ZIP file to upload into Weblate for translations initialization
- **file docfile** – Document à traduire
- **boolean disable_autoshare** – Disables automatic repository sharing via **URLs internes de Weblate**.

Objet JSON de requête

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

Objet JSON de réponse

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

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

**CURL form request example :**

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

**CURL JSON request example :**

```bash
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" \nhttp://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"
}
```
The Weblate Manual, Version 4.16

### JSON request to create a new component from another one:

```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

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

### JSON response example:

```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

{
    "branch": "main",
    "file_format": "po",
    "filemask": "po/*.po",
    "git_export": "",
    "license": "",
    "license_url": "",
    "name": "Weblate",
    "slug": "weblate",
    "project": {
        "name": "Hello",
        "slug": "hello",
        "source_language": {
            "code": "en",
            "direction": "ltr",
            "population": 15903439015,
            "name": "English",
            "url": "http://example.com/api/languages/en/",
            "web_url": "http://example.com/languages/en/"
        },
        "url": "http://example.com/api/projects/hello/",
        "web": "https://weblate.org/",
        "web_url": "http://example.com/projects/hello/"
    },
    "repo": "file:///home/nijel/work/weblate-hello",
    "template": "",
    "new_base": "",
    "url": "http://example.com/api/components/hello/weblate/",
    "vcs": "git",
    "web_url": "http://example.com/projects/hello/weblate/"
}
```

GET /api/projects/ (string: project) /languages/
Returns paginated statistics for all languages within a project.
Nouveaux dans la version 3.8.

Paramètres
— **project**(string) – URL abrégée du projet

Objet JSON de réponse
— **results**(array) – array of translation statistics objects
— **language**(string) – Nom de la langue
— **code**(string) – Code de la langue
— **total**(int) – total number of strings
— **translated**(int) – nombre de chaînes traduites
— **translated_percent**(float) – pourcentage de chaînes traduites
— **total_words**(int) – total number of words
— **translated_words**(int) – number of translated words
— **words_percent**(float) – percentage of translated words

GET /api/projects/(string: project)/statistics/
Returns statistics for a project.
Nouveaux dans la version 3.8.

Paramètres
— **project**(string) – URL abrégée du projet

Objet JSON de réponse
— **total**(int) – total number of strings
— **translated**(int) – nombre de chaînes traduites
— **translated_percent**(float) – pourcentage de chaînes traduites
— **total_words**(int) – total number of words
— **translated_words**(int) – number of translated words
— **words_percent**(float) – percentage of translated words

### 1.12.8 Composants

**Indication** : Utiliser **POST /api/projects/(string:project)/components/** pour créer des composants.

GET /api/components/
Returns a list of translation components.

**Voir aussi** :
Component object attributes are documented at **GET /api/components/(string:project)/
(string:component)/**.

GET /api/components/(string: project)/
**string**: component/
Returns information about translation component.

Paramètres
— **project**(string) – URL abrégée du projet
— **component**(string) – URL abrégée du composant

Objet JSON de réponse
— **project**(object) – the translation project; see **GET /api/projects/(string:project)/**
— **name**(string) – Nom du composant
— **slug**(string) – Identifiant du composant
— **vcs**(string) – Système de contrôle de version
— **repo**(string) – Dépôt du code source
— **git_export**(string) – URL de dépôt exportée
— branch (string) – Branch du dépôt
— push_branch (string) – Pousser la branche
— filemask (string) – Masque de fichier
— template (string) – Fichier de langue de base mono-langue
— edit_template (string) – Modifier le fichier de base
— intermediate (string) – Fichier de langue intermédiaire
— new_base (string) – Modèle pour les nouvelles traductions
— file_format (string) – Format de fichier
— license (string) – Licence associée à cette traduction
— agreement (string) – Accord de contribution
— new_lang (string) – Ajouter une nouvelle traduction
— language_code_style (string) – Style de code-langue
— source_language (object) – source language object; see GET /api/languages/(string:language)/
— push (string) – URL pour l'envoi du dépôt
— check_flags (string) – Drapeaux de traduction
— priority (string) – Priorité
— enforced_checks (string) – Vérifications forcées
— restricted (string) – Accès restreint
— repoweb (string) – Explorateur de dépôt
— report_source_bugs (string) – Adresse pour signaler une anomalie de chaîne source
— merge_style (string) – Style de fusion
— commit_message (string) – Commit, add, delete, merge, add-on, and merge request messages
— add_message (string) – Commit, add, delete, merge, add-on, and merge request messages
— delete_message (string) – Commit, add, delete, merge, add-on, and merge request messages
— merge_message (string) – Commit, add, delete, merge, add-on, and merge request messages
— addon_message (string) – Commit, add, delete, merge, add-on, and merge request messages
— pull_message (string) – Commit, add, delete, merge, add-on, and merge request messages
— allow_translation_propagation (string) – Permettre la propagation de la traduction
— enable_suggestions (string) – Autoriser les suggestions
— suggestion_voting (string) – Vote pour la suggestion
— suggestion_autoaccept (string) – Accepter automatiquement les suggestions
— push_on_commit (string) – Pousser lors du commit
— commit_pending_age (string) – Âge des modifications à commiter
— auto_lock_error (string) – Verrouiller en cas d'erreur
— language_regex (string) – Filtre sur la langue
— variant_regex (string) – Expression rationnelle des variantes
— repository_url (string) – URL to repository status; see GET /api/components/(string:project)/(string:component)/repository/
— translations_url (string) – URL to translations list; see GET /api/components/(string:project)/(string:component)/translations/
— lock_url (string) – URL to lock status; see GET /api/components/(string:project)/(string:component)/lock/
— changes_list_url (string) – URL to changes list; see GET /api/components/(string:project)/(string:component)/changes/
— task_url (string) – URL to a background task (if any); see GET /api/tasks/(str:uuid)/

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

PATCH /api/components/(string: project)/
  string: component/
Edit a component by a PATCH request.

Paramètres
  — project(string) – URL abrégée du projet
  — component(string) – URL abrégée du composant
  — source_language(string) – Project source language code (optional)

Objet JSON de requête
  — name(string) – name of component
  — slug(string) – slug of component
  — repo(string) – VCS repository URL

Exemple CURL :

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

JSON request example :
PATCH /api/projects/hello/components/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20
{
  "name": "new name"
}

JSON response example:

HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, POST, HEAD, OPTIONS
{
  "branch": "main",
  "file_format": "po",
  "filemask": "po/*.po",
  "git_export": "",
  "license": "",
  "license_url": "",
  "name": "new name",
  "slug": "weblate",
  "project": {
    "name": "Hello",
    "slug": "hello",
    "source_language": {
      "code": "en",
      "direction": "ltr",
      "population": 159034349015,
      "name": "English",
      "url": "http://example.com/api/languages/en/",
      "web_url": "http://example.com/languages/en/"
    },
    "url": "http://example.com/api/projects/hello/",
    "web": "https://weblate.org/",
    "web_url": "http://example.com/projects/hello/"
  },
  "repo": "file:///home/nijel/work/weblate-hello",
  "template": "",
  "new_base": "",
  "url": "http://example.com/api/components/hello/weblate/",
  "vcs": "git",
  "web_url": "http://example.com/projects/hello/weblate/"
}

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

Edit a component by a PUT request.

Paramètres
  — project(string) – URL abrégée du projet
  — component(string) – URL abrégée du composant

Objet JSON de requête
  — branch(string) – VCS repository branch
### file_format (string)
- file format of translations

### filename (string)
- mask of translation files in the repository

### name (string)
- name of component

### slug (string)
- slug of component

### repo (string)
- VCS repository URL

### template (string)
- base file for monolingual translations

### new_base (string)
- base file for adding new translations

### vcs (string)
- VCS repository URL

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

```
string: component/
```

Nouveau dans la version 3.9.

Deletes a component.

**Paramètres**

- **project (string)** – URL abrégée du projet
- **component (string)** – URL abrégée du composant

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

```
string: component/changes/
```

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

**Paramètres**

- **project (string)** – URL abrégée du projet
- **component (string)** – URL abrégée du composant

Objet JSON de réponse

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

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

```
string: component/file/
```

Nouveau dans la version 4.9.

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

**Paramètres**

- **project (string)** – URL abrégée du projet
- **component (string)** – URL abrégée du composant

**Paramètres de requête**

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

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

```
string: component/screenshots/
```

Returns a list of component screenshots.

**Paramètres**

- **project (string)** – URL abrégée du projet
- **component (string)** – URL abrégée du composant

Objet JSON de réponse

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

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

```
string: component/lock/
```

Returns component lock status.

**Paramètres**

- **project (string)** – URL abrégée du projet
- **component (string)** – URL abrégée du composant

Objet JSON de réponse

- **locked (boolean)** – whether component is locked for updates
Example JSON data:

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

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

Sets component lock status.

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

**Paramètres**
- project (string) – URL abrégée du projet
- component (string) – URL abrégée du composant

**Objet JSON de requête**
- lock – Boolean whether to lock or not.

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

**JSON request example:**

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

{"lock": true}
```

**JSON response example:**

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

{"locked": true}
```

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

Returns information about VCS repository status.

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

**Paramètres**
- project (string) – URL abrégée du projet
- component (string) – URL abrégée du composant

**Objet JSON de réponse**
- needs_commit (boolean) – whether there are any pending changes to commit
- needs_merge (boolean) – whether there are any upstream changes to merge
- needs_push (boolean) – whether there are any local changes to push
- remote_commit (string) – Remote commit information
— **status** (string) – VCS repository status as reported by VCS
— **merge_failure** – Text describing merge failure or null if there is none

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

Performs the given operation on a VCS repository.
See **POST /api/projects/ (string:project)/repository/** for documentation.

**Paramètres**
— **project** (string) – URL abrégée du projet
— **component** (string) – URL abrégée du composant

**Objet JSON de requête**
— **operation** (string) – Operation to perform: one of push, pull, commit, reset, cleanup

**Objet JSON de réponse**
— **result** (boolean) – result of the operation

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

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

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

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

{"result":true}
```

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

Downloads base file for monolingual translations.

**Paramètres**
— **project** (string) – URL abrégée du projet
— **component** (string) – URL abrégée du composant

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

Downloads template file for new translations.

**Paramètres**
— **project** (string) – URL abrégée du projet
— **component** (string) – URL abrégée du composant
GET /api/components/(string: project)/
string: component/translations/

Returns a list of translation objects in the given component.

**Paramètres**
- `project` (string) – URL abrégée du projet
- `component` (string) – URL abrégée du composant

**Objet JSON de réponse**
- `results` (array) – array of translation objects; see GET /api/translations/(string:project)/(string:component)/(string:language)/

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

Creates new translation in the given component.

**Paramètres**
- `project` (string) – URL abrégée du projet
- `component` (string) – URL abrégée du composant

**Objet JSON de requête**
- `language_code` (string) – translation language code; see GET /api/languages/(string:language)/

**Objet JSON de réponse**
- `result` (object) – new translation object created

**Exemple CURL :**

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

**JSON request example :**

```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 response example :**

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

{
    "failing_checks": 0,
    "failing_checks_percent": 0,
    "failing_checks_words": 0,
    "filename": "po/cs.po",
    "fuzzy": 0,
    "fuzzy_percent": 0.0,
    "fuzzy_words": 0,
    "have_comment": 0,
}
```
GET /api/components/(string: project)/
string: component/statistics/

Returns paginated statistics for all translations within component.

Nouveau dans la version 2.7.

Paramètres
— project(string) – URL abrégée du projet
— component(string) – URL abrégée du composant

Objet JSON de réponse
— results (array) – array of translation statistics objects; see GET /api/translations/(string:project)/(string:component)/{string:language}/statistics/

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

Retourne les projets liés à un composant.

Nouveau dans la version 4.5.

Paramètres
— project(string) – URL abrégée du projet
— component(string) – URL abrégée du composant

Objet JSON de réponse
— projects (array) – associated projects; see GET /api/projects/(string:project)/

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

Associe un projet à un composant.

Nouveau dans la version 4.5.

Paramètres
— project(string) – URL abrégée du projet
— component(string) – URL abrégée du composant

Paramètres de la forme
DELETE /api/components/(string: project)/
  string: component/links/string: project_slug/
Remove association of a project with a component.
Nouveau dans la version 4.5.

Paramètres
— project (string) – URL abrégée du projet
— component (string) – URL abrégée du composant
— project_slug (string) – L’identifiant du projet à supprimer

1.12.9 Traductions

GET /api/translations/
Returns a list of translations.
Voir aussi :
Translation object attributes are documented at GET /api/translations/(string:project)/
  (string:component)/(string:language)/.

GET /api/translations/(string: project)/
  string: component/string: language/
Returns information about a translation.

Paramètres
— project (string) – URL abrégée du projet
— component (string) – URL abrégée du composant
— language (string) – Translation language code

Objet JSON de réponse
— component (object) – component object; see GET /api/components/
  (string:project)/(string:component)/
— failing_checks (int) – nombre de chaînes qui échouent aux contrôles
— failing_checks_percent (float) – pourcentage de chaînes qui échouent aux contrôles
— failing_checks_words (int) – nombre de mots avec des contrôles échoués
— filename (string) – Nom du fichier de traduction
— fuzzy (int) – nombre de chaînes à vérifier
— fuzzy_percent (float) – percentage of fuzzy (marked for edit) strings
— fuzzy_words (int) – number of words in fuzzy (marked for edit) strings
— have_comment (int) – number of strings with comment
— have_suggestion (int) – number of strings with suggestion
— is_template (boolean) – si la traduction a une base mono-langue
— language (object) – source language object; see GET /api/languages/
  (string:language)/
— language_code (string) – language code used in the repository; this can be different
  from language code in the language object
— last_author (string) – name of last author
— last_change (timestamp) – last change timestamp
— revision (string) – revision hash for the file
— share_url (string) – URL for sharing leading to engagement page
— total (int) – total number of strings
— total_words (int) – total number of words
— translate_url (string) – URL for translating
— translated (int) – nombre de chaînes traduites
— translated_percent (float) – pourcentage de chaînes traduites
— translated_words (int) – number of translated words
— repository_url (string) – URL to repository status; see GET /api/translations/(string:project)/
  (string:component)/(string:language)/repository/
— file_url (string) – URL to file object; see GET /api/translations/(string:project)/(string:component)/(string:language)/file/
— changes_list_url (string) – URL to changes list; see GET /api/translations/(string:project)/(string:component)/(string:language)/changes/
— units_list_url (string) – URL to strings list; see GET /api/translations/(string:project)/(string:component)/(string:language)/units/

Example JSON data:

```
{
  "component": {
    "branch": "main",
    "file_format": "po",
    "filemask": "po/*.po",
    "git_export": "",
    "license": "",
    "license_url": "",
    "name": "Weblate",
    "new_base": "",
    "project": {
      "name": "Hello",
      "slug": "hello",
      "source_language": {
        "code": "en",
        "direction": "ltr",
        "population": 159034349015,
        "name": "English",
        "url": "http://example.com/api/languages/en/",
        "web_url": "http://example.com/languages/en/"
      },
      "url": "http://example.com/api/projects/hello/",
      "web": "https://weblate.org/",
      "web_url": "http://example.com/projects/hello/"
    },
    "repo": "file:///home/nijel/work/weblate-hello",
    "slug": "weblate",
    "template": "",
    "url": "http://example.com/api/components/hello/weblate/",
    "vcs": "git",
    "web_url": "http://example.com/projects/hello/weblate/"
  },
  "failing_checks": 3,
  "failing_checks_percent": 75.0,
  "failing_checks_words": 11,
  "filename": "po/cs.po",
  "fuzzy": 0,
  "fuzzy_percent": 0.0,
  "fuzzy_words": 0,
  "have_comment": 0,
  "have_suggestion": 0,
  "is_template": false,
  "language": {
    "code": "cs",
    "direction": "ltr",
    "population": 1303174280
    "name": "Czech",
    "url": "http://example.com/api/languages/cs/",
    "web_url": "http://example.com/languages/cs/"
  },
  "language_code": "cs",
}
```

(suite sur la page suivante)
DELETE /api/translations/(string: project) /
string: component/string: language/

Nouveau dans la version 3.9.
Deletes a translation.

Paramètres
— project (string) – URL abrégée du projet
— component (string) – URL abrégée du composant
— language (string) – Translation language code

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

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

Paramètres
— project (string) – URL abrégée du projet
— component (string) – URL abrégée du composant
— language (string) – Translation language code

Objet JSON de réponse
— results (array) – array of component objects; see GET /api/changes/ (int:id)/

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

Returns a list of translation units.

Paramètres
— project (string) – URL abrégée du projet
— component (string) – URL abrégée du composant
— language (string) – Translation language code
— q (string) – Search query string Recherche (optional)

Objet JSON de réponse
— results (array) – array of component objects; see GET /api/units/ (int:id)/

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

Ajouter une nouvelle unité.

Paramètres
— project (string) – URL abrégée du projet
— component (string) – URL abrégée du composant
— language (string) – Translation language code

Objet JSON de requête
— key (string) – Name of translation unit (used as key or context)
— value (array) – Source strings (use single string if not creating plural)
— state (int) – String state; see GET /api/units/(int:id)/

Objet JSON de réponse
— unit (object) – newly created unit; see GET /api/units/(int:id)/

Voir aussi :
Gérer les chaînes, adding-new-strings

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

Trigger automatic translation.

Paramètres
— project (string) – URL abrégée du projet
— component (string) – URL abrégée du composant
— language (string) – Translation language code

Objet JSON de requête
— mode (string) – Mode de traduction automatique
— filter_type (string) – Automatic translation filter type
— auto_source (string) – Automatic translation source - mt or others
— component (string) – Activez la contribution au mémoire de traduction partagé du projet afin d’avoir accès aux composants supplémentaires.
— engines (array) – Moteurs de traduction automatisée
— threshold (string) – Seuil de score

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

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

Note : This API endpoint uses different logic for output than rest of API as it operates on whole file rather than on data. Set of accepted format parameter differs and without such parameter you get translation file as stored in VCS.

Paramètres de requête
— format – File format to use; if not specified no format conversion happens: supported file formats : po, mo, xlliff, xlliff11, tbx, tmx, csv, xlsx, json, aresource, strings
— q (string) – Filter downloaded strings, see search, only applicable when conversion is in place (format is specified).

Paramètres
— project (string) – URL abrégée du projet
— component (string) – URL abrégée du composant
— language (string) – Translation language code

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

Upload new file with translations.

Paramètres
— project (string) – URL abrégée du projet
— component (string) – URL abrégée du composant
— language (string) – Translation language code

Paramètres de la forme
— string conflicts – How to deal with conflicts (ignore, replace-translated or replace-approved)
— file file – Fichier téléversé
— string email – Adresse courriel de l’auteur
— string author – Nom de l’auteur
— string method – Upload method (translate, approve, suggest, fuzzy, replace, source, add), see Import methods

1.12. API REST de Weblate
— *string fuzzy* – Fuzzy (marked for edit) strings processing (*empty*, *process*, *approve*)

**Exemple CURL :**
```bash
curl -X POST \
  -F file=@strings.xml \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/translations/hello/android/cs/file/
```

**GET /api/translations/(string: project)/**
**string: component/string: language/repository/**
Returns information about VCS repository status.
The response is same as for **GET /api/components/(string:project)/**
**(string:component)/repository/**.

**Paramètres**
— *project* (*string*) – URL abrégée du projet
— *component* (*string*) – URL abrégée du composant
— *language* (*string*) – Translation language code

**POST /api/translations/(string: project)/**
**string: component/string: language/repository/**
Performs given operation on the VCS repository.
See **POST /api/projects/(string:project)/repository/** for documentation.

**Paramètres**
— *project* (*string*) – URL abrégée du projet
— *component* (*string*) – URL abrégée du composant
— *language* (*string*) – Translation language code

**Objet JSON de requête**
— *operation* (*string*) – Operation to perform: one of push, pull, commit, reset, cleanup

**Objet JSON de réponse**
— *result* (*boolean*) – result of the operation

**GET /api/translations/(string: project)/**
**string: component/string: language/statistics/**
Returns detailed translation statistics.
Nouveau dans la version 2.7.

**Paramètres**
— *project* (*string*) – URL abrégée du projet
— *component* (*string*) – URL abrégée du composant
— *language* (*string*) – Translation language code

**Objet JSON de réponse**
— *name* (*string*) – Nom de la langue
— *code* (*string*) – Code de la langue
— *failing* (*int*) – number of failing checks
— *failing_percent* (*float*) – percentage of failing checks
— *fuzzy* (*int*) – nombre de chaînes à vérifier
— *fuzzy_percent* (*float*) – percentage of fuzzy (marked for edit) strings
— *total_words* (*int*) – total number of words
— *translated_words* (*int*) – number of translated words
— *last_author* (*string*) – name of last author
— *last_change* (*timestamp*) – date of last change
— *name* (*string*) – Nom de la langue
— *total* (*int*) – total number of strings
— *translated* (*int*) – nombre de chaînes traduites
— *translated_percent* (*float*) – pourcentage de chaînes traduites
— *url* (*string*) – URL to access the translation (engagement URL)
— *url_translate* (*string*) – URL to access the translation (real translation URL)
1.12.10 Mémoire


**GET /api/memory/**

Returns a list of memory results.

**DELETE /api/memory/ (int: memory_object_id)/**

Deletes a memory object

- **Paramètres**
  - memory_object_id – Memory Object ID

1.12.11 Unités

A *unit* is a single piece of a translation which pairs a source string with a corresponding translated string and also contains some related metadata. The term is derived from the Translate Toolkit and XLIFF.

Nouveau dans la version 2.10.

**GET /api/units/**

Returns list of translation units.

- **Paramètres**
  - q (string) – Search query string *Recherche* (optional)

Voir aussi :

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

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

Modifié dans la version 4.3 : The *target* and *source* are now arrays to properly handle plural strings.

Returns information about translation unit.

- **Paramètres**
  - id (int) – ID d’unité

**Objet JSON de réponse**

- translation (string) – URL of a related translation object
- source (array) – Chaîne source
- previous_source (string) – previous source string used for fuzzy matching
- target (array) – Chaîne cible
- id_hash (string) – unique identifier of the unit
- content_hash (string) – unique identifier of the source string
- location (string) – location of the unit in source code
- context (string) – translation unit context
- note (string) – translation unit note
- flags (string) – translation unit flags
- labels (array) – translation unit labels, available on source units
- state (int) – unit state, 0 - untranslated, 10 - needs editing, 20 - translated, 30 - approved, 100 - read only
- fuzzy (boolean) – si l’unité est approximative ou marquée pour révision
- translated (boolean) – si l’unité est traduite
- approved (boolean) – si la traduction est approuvée
- position (int) – unit position in translation file
- has_suggestion (boolean) – si l’unité a des suggestions
- has_comment (boolean) – si l’unité a des commentaires
- has_failing_check (boolean) – si l’unité a échoué aux contrôles
- num_words (int) – number of source words
- priority (int) – translation priority; 100 is default
- id (int) – Identifiant d’unité
- explanation (string) – String explanation, available on source units, see *Additional info on source strings*
— extra_flag (string) – Drapeaux de chaîne supplémentaires, disponibles sur les unités sources, voir Customizing behavior using flags
— web_url (string) – URL where the unit can be edited
— source_unit (string) – Source unit link; see GET /api/units/(int:id)/
— pending (boolean) – whether the unit is pending for write
— timestamp (timestamp) – string age

PATCH /api/units/(int: id)/
Nouveau dans la version 4.3.
Réaliser une mise à jour partielle sur l’unité de traduction.

Paramètres
— id (int) – ID d’unité

Objet JSON de requête
— state (int) – unit state, 0 - untranslated, 10 - needs editing, 20 - translated, 30 - approved
   (need review workflow enabled, see Dedicated reviewers)
— target (array) – Chaîne cible
— explanation (string) – String explanation, available on source units, see Additional info on source strings
— extra_flag (string) – Drapeaux de chaîne supplémentaires, disponibles sur les unités sources, voir Customizing behavior using flags

Objet JSON de réponse
— labels (array) – labels, available on source units

PUT /api/units/(int: id)/
Nouveau dans la version 4.3.
Réaliser une mise à jour complète sur l’unité de traduction.

Paramètres
— id (int) – ID d’unité

Objet JSON de requête
— state (int) – unit state, 0 - untranslated, 10 - needs editing, 20 - translated, 30 - approved
   (need review workflow enabled, see Dedicated reviewers)
— target (array) – Chaîne cible
— explanation (string) – String explanation, available on source units, see Additional info on source strings
— extra_flag (string) – Drapeaux de chaîne supplémentaires, disponibles sur les unités sources, voir Customizing behavior using flags

Objet JSON de réponse
— labels (array) – labels, available on source units

DELETE /api/units/(int: id)/
Nouveau dans la version 4.3.
Supprime une unité de traduction.

Paramètres
— id (int) – ID d’unité

1.12.12 Modifications

Nouveau dans la version 2.10.

GET /api/changes/
Modifié dans la version 4.1 : Filtering of changes was introduced in the 4.1 release.
Returns a list of translation changes.
Voir aussi :
Change object attributes are documented at GET /api/changes/(int:id)/.

Paramètres de requête
The Weblate Manual, Version 4.16

— **user** *(string)* – Username of user to filters
— **action** *(int)* – Action to filter, can be used several times
— **timestamp_after** *(timestamp)* – ISO 8601 formatted timestamp to list changes after
— **timestamp_before** *(timestamp)* – ISO 8601 formatted timestamp to list changes before

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

Returns information about translation change.

**Paramètres**
— **id** *(int)* – ID de la modification

**Objet JSON de réponse**
— **unit** *(string)* – URL of a related unit object
— **translation** *(string)* – URL of a related translation object
— **component** *(string)* – URL of a related component object
— **user** *(string)* – URL of a related user object
— **author** *(string)* – URL of a related author object
— **timestamp** *(timestamp)* – Horodatage de l'événement
— **action** *(int)* – numeric identification of action
— **action_name** *(string)* – text description of action
— **target** *(string)* – event changed text or detail
— **id** *(int)* – Identifiant de l'événement

### 1.12.13 Captures d'écran


**GET /api/screenshots/**

Returns a list of screenshot string information.

**Voir aussi :**

Screenshot object attributes are documented at **GET /api/screenshots/(int:id)/**.

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

Returns information about screenshot information.

**Paramètres**
— **id** *(int)* – ID de la capture d'écran

**Objet JSON de réponse**
— **name** *(string)* – name of a screenshot
— **component** *(string)* – URL of a related component object
— **file_url** *(string)* – URL to download a file; see **GET /api/screenshots/(int:id)/file/**
— **units** *(array)* – link to associated source string information; see **GET /api/units/(int:id)/**

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

Download the screenshot image.

**Paramètres**
— **id** *(int)* – ID de la capture d'écran

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

Replace screenshot image.

**Paramètres**
— **id** *(int)* – ID de la capture d'écran

**Paramètres de la forme**
— **file image** – Fichier téléversé

Exemple 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/

Associate source string with screenshot.

Paramètres
- id(int) – ID de la capture d'écran

Paramètres de la forme
- string unit_id – ID d'unité

Objet JSON de réponse
- name(string) – name of a screenshot
- translation(string) – URL of a related translation object
- file_url(string) – URL to download a file; see GET /api/screenshots/(int:id)/file/
- units(array) – link to associated source string information; see GET /api/units/(int:id)/

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

int: unit_id

Supprimer l'association de la chaîne source avec la capture d'écran.

Paramètres
- id(int) – ID de la capture d'écran
- unit_id – ID de l'unité de la chaîne source

POST /api/screenshots/

Create a new screenshot.

Paramètres de la forme
- file image – Fichier téléversé
- string name – Nom de la capture d'écran
- string project_slug – Identifiant du projet
- string component_slug – Identifiant du composant
- string language_code – Code langue

Objet JSON de réponse
- name(string) – name of a screenshot
- component(string) – URL of a related component object
- file_url(string) – URL to download a file; see GET /api/screenshots/(int:id)/file/
- units(array) – link to associated source string information; see GET /api/units/(int:id)/

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

Modifier les informations partielles sur la capture d'écran.

Paramètres
- id(int) – ID de la capture d'écran

Objet JSON de réponse
- name(string) – name of a screenshot
- component(string) – URL of a related component object
- file_url(string) – URL to download a file; see GET /api/screenshots/(int:id)/file/
- units(array) – link to associated source string information; see GET /api/units/(int:id)/

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

Modifier les informations complètes sur la capture d'écran.
Paramètres
— id (int) – ID de la capture d’écran

Objet JSON de réponse
— name (string) – name of a screenshot
— component (string) – URL of a related component object
— file_url (string) – URL to download a file; see GET /api/screenshots/ (int:id)/file/
— units (array) – link to associated source string information; see GET /api/units/ (int:id)/

DELETE /api/screenshots/(int: id)/
Supprimer la capture d’écran.

Paramètres
— id (int) – ID de la capture d’écran

1.12.14 Extensions

Nouveau dans la version 4.4.1.

GET /api/addons/
Returns a list of add-ons.
Voir aussi :
Add-on object attributes are documented at GET /api/addons/(int:id)/.

GET /api/addons/(int: id)/
Returns information about add-on information.

Paramètres
— id (int) – ID du module

Objet JSON de réponse
— name (string) – name of an add-on
— component (string) – URL of a related component object
— configuration (object) – Optional add-on configuration

Voir aussi :
Extensions

POST /api/components/(string: project)/
string: component/addons/
Creates a new add-on.

Paramètres
— project_slug (string) – Identifiant du projet
— component_slug (string) – Identifiant du composant

Objet JSON de requête
— name (string) – name of an add-on
— configuration (object) – Optional add-on configuration

PATCH /api/addons/(int: id)/
Edit partial information about add-on.

Paramètres
— id (int) – ID du module

Objet JSON de réponse
— configuration (object) – Optional add-on configuration

PUT /api/addons/(int: id)/
Edit full information about add-on.

Paramètres
— id (int) – ID du module
Objet JSON de réponse
— `configuration` (object) – Optional add-on configuration

DELETE /api/addons/(int: id) /
Delete add-on.
Paramètres
— `id` (int) – ID du module

1.12.15 Listes de composants

Nouveau dans la version 4.0.

GET /api/component-lists/
Returns a list of component lists.
Voir aussi :
Component list object attributes are documented at GET /api/component-lists/(str:slug)/.

GET /api/component-lists/(str: slug) /
Returns information about component list.
Paramètres
— `slug` (string) – Component list slug

Objet JSON de réponse
— `name` (string) – name of a component list
— `slug` (string) – slug of a component list
— `show_dashboard` (boolean) – whether to show it on a dashboard
— `components` (array) – link to associated components; see GET /api/components/(string:project)/(string:component)/
— `auto_assign` (array) – automatic assignment rules

PUT /api/component-lists/(str: slug) /
Changes the component list parameters.

Paramètres
— `slug` (string) – Component list slug

Objet JSON de requête
— `name` (string) – name of a component list
— `slug` (string) – slug of a component list
— `show_dashboard` (boolean) – whether to show it on a dashboard

PATCH /api/component-lists/(str: slug) /
Changes the component list parameters.

Paramètres
— `slug` (string) – Component list slug

Objet JSON de requête
— `name` (string) – name of a component list
— `slug` (string) – slug of a component list
— `show_dashboard` (boolean) – whether to show it on a dashboard

DELETE /api/component-lists/(str: slug) /
Deletes the component list.
Paramètres
— `slug` (string) – Component list slug

POST /api/component-lists/(str: slug)/components/
Associate component with a component list.
Paramètres
— `slug` (string) – Component list slug
Paramètres de la forme

- **string component_id** – ID du composant

DELETE /api/component-lists/ (str: slug) /components/

str: component_slug

Disassocie un composant de la liste de composants.

Paramètres

- **slug** (string) – Slug de la liste de composants
- **component_slug** (string) – Identifiant du composant

1.12.16 Glossaire

Modifié dans la version 4.5 : Glossaries are now stored as regular components, translations and strings, please use respective API instead.

1.12.17 Tâches

Nouveau dans la version 4.4.

GET /api/tasks/

L'affichage de la liste des tâches est actuellement non disponible.

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

Renvoie des informations à propos d'une tâche

Paramètres

- **uuid** (string) – Task UUID

Objet JSON de réponse

- **completed** (boolean) – Si la tâche est achevée
- **progress** (int) – Task progress in percent
- **result** (object) – Task result or progress details
- **log** (string) – Journal des tâches

1.12.18 Metrics

GET /api/metrics/

Returns server metrics.

Objet JSON de réponse

- **units** (int) – Number of units
- **units_translated** (int) – Number of translated units
- **users** (int) – Number of users
- **changes** (int) – Nombre de modifications
- **projects** (int) – Number of projects
- **components** (int) – Nombre de composants
- **translations** (int) – Number of translations
- **languages** (int) – Number of used languages
- **checks** (int) – Number of triggered quality checks
- **configuration_errors** (int) – Number of configuration errors
- **suggestions** (int) – Number of pending suggestions
- **celery_queues** (object) – Lengths of Celery queues, see Background tasks using Celery
- **name** (string) – Nom du serveur configuré
1.12.19 Déclencheurs de notification

Notification hooks allow external applications to notify Weblate that the VCS repository has been updated.

You can use repository endpoints for projects, components and translations to update individual repositories; see POST /api/projects/(string:project)/repository/ for documentation.

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

*string: component/*

Obsolette depuis la version 2.6 : Please use POST /api/components/(string:project)/(string:component)/repository/ instead which works properly with authentication for ACL limited projects.

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

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

* Obsolète depuis la version 2.6 : Please use POST /api/projects/(string:project)/repository/ instead which works properly with authentication for ACL limited projects.

Triggers update of all components in a project (pulling from VCS and scanning for translation changes).

**POST /hooks/github/**

Special hook for handling GitHub notifications and automatically updating matching components.

*Note*: GitHub includes direct support for notifying Weblate: enable Weblate service hook in repository settings and set the URL to the URL of your Weblate installation.

Voir aussi:

*Automatically receiving changes from GitHub*

For instruction on setting up GitHub integration


Generic information about GitHub Webhooks

**ENABLE_HOOKS**

For enabling hooks for whole Weblate

**POST /hooks/gitlab/**

Special hook for handling GitLab notifications and automatically updating matching components.

Voir aussi:

*Automatically receiving changes from GitLab*

For instruction on setting up GitLab integration

https://docs.gitlab.com/ee/user/project/integrations/webhooks.html

Generic information about GitLab Webhooks

**ENABLE_HOOKS**

For enabling hooks for whole Weblate

**POST /hooks/bitbucket/**

Special hook for handling Bitbucket notifications and automatically updating matching components.

Voir aussi:

*Automatically receiving changes from Bitbucket*

For instruction on setting up Bitbucket integration

https://support.atlassian.com/bitbucket-cloud/docs/manage-webhooks/

Generic information about Bitbucket Webhooks

**ENABLE_HOOKS**

For enabling hooks for whole Weblate
POST /hooks/pagure/
Nouveau dans la version 3.3.
Special hook for handling Pagure notifications and automatically updating matching components.

**Voir aussi**:

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

https://docs.pagure.org/pagure/usage/using_webhooks.html
  Generic information about Pagure Webhooks

**ENABLE_HOOKS**
  For enabling hooks for whole Weblate

POST /hooks/azure/
Nouveau dans la version 3.8.
Special hook for handling Azure DevOps notifications and automatically updating matching components.

**Note**: Please make sure that *Resource details to send* is set to *All*, otherwise Weblate will not be able to match your Azure repository.

**Voir aussi**:

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

  Generic information about Azure DevOps Web Hooks

**ENABLE_HOOKS**
  For enabling hooks for whole Weblate

POST /hooks/gitea/
Nouveau dans la version 3.9.
Special hook for handling Gitea Webhook notifications and automatically updating matching components.

**Voir aussi**:

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

https://docs.gitea.io/en-us/webhooks/
  Generic information about Gitea Webhooks

**ENABLE_HOOKS**
  For enabling hooks for whole Weblate

POST /hooks/gitee/
Nouveau dans la version 3.9.
Special hook for handling Gitee Webhook notifications and automatically updating matching components.

**Voir aussi**:

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

https://gitee.com/help/categories/40
  Generic information about Gitee Webhooks

**ENABLE_HOOKS**
  For enabling hooks for whole Weblate
1.12.20 Exportations

Weblate provides various exports to allow you to further process the data.

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

Paramètres de requête
— format (string) – Output format: either json or csv

Obsolète depuis la version 2.6 : Please use GET /api/components/(string:project)/
(string:component)/statistics/ and GET /api/translations/
(string:project)/(string:component)/(string:language)/statistics/ instead; it allows access to ACL controlled projects as well.
Retrieves statistics for given component in given format.

Exemple de requête :

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

Exemple de réponse :

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/

}]

(suite sur la page suivante)
1.12.21 Flux RSS

Changes in translations are exported in RSS feeds.

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

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

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

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

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

Voir aussi :
RSS on Wikipedia

1.13 Client Weblate

Nouveau dans la version 2.7 : There has been full wlc utility support ever since Weblate 2.7. If you are using an older version some incompatibilities with the API might occur.
1.13.1 Installation

The Weblate Client is shipped separately and includes the Python module. To use the commands below, you need to install `wlc`:

```
pip install wlc
```

1.13.2 Utilisation de Docker

Le client Weblate est également disponible sous forme d'image Docker.

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

Installation :

```
docker pull weblate/wlc
```

The Docker container uses Weblate's default settings and connects to the API deployed in localhost. The API URL and API_KEY can be configured through the arguments accepted by Weblate.

La commande pour lancer le conteneur utilise la syntaxe suivante :

```
docker run --rm weblate/wlc [WLC_ARGS]
```

Exemple :

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

You might want to pass your Fichiers de configuration to the Docker container, the easiest approach is to add your current directory as /home/weblate volume :

```
docker run --volume $PWD:/home/weblate --rm weblate/wlc show
```

1.13.3 Pour commencer

The wlc configuration is stored in ~/.config/weblate (see Fichiers de configuration for other locations), please create it to match your environment :

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

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

Vous pouvez alors invoquer des commandes sur le serveur par défaut :

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

Voir aussi :

Fichiers de configuration
1.13.4 Synopsis

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

Commands actually indicate which operation should be performed.

1.13.5 Description

Weblate Client is a Python library and command-line utility to manage Weblate remotely using API REST de Weblate. The command-line utility can be invoked as `wlc` and is built-in on `wlc`.

Arguments

The program accepts the following arguments which define output format or which Weblate instance to use. These must be entered before any command.

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

Spécifiez le format de sortie.

```
--url URL
```

Specify the API URL. Overrides any value found in the configuration file, see Fichiers de configuration. The URL should end with `/api/`, for example `https://hosted.weblate.org/api/`.

```
--key KEY
```

Specify the API user key to use. Overrides any value found in the configuration file, see Fichiers de configuration. You can find your key in your profile on Weblate.

```
--config PATH
```

Surcharge le chemin du fichier d’utilisation, voir Fichiers de configuration.

```
--config-section SECTION
```

Surcharge la section du fichier d’utilisation en cours d’utilisation, voir Fichiers de configuration.

Commandes

Les commandes suivantes sont disponibles :

```
version
```

Imprime la version actuelle.

```
list-languages
```

Liste les langues utilisées dans Weblate.

```
list-projects
```

Liste des projets dans Weblate.

```
list-components
```

Liste les composants dans Weblate.

```
list-translations
```

Liste les traductions dans Weblate.

```
show
```

Affiche l’objet Weblate (traduction, composant ou projet).

```
lis
```

Liste les objets Weblate (traduction, composant ou projet).
commit
Commits changes made in a Weblate object (translation, component or project).

pull
Met à jour Weblate avec les modifications du dépôt distant (traduction, composant ou projet).

push
Pousse les modifications Weblate vers le dépôt distant (traduction, composant ou projet).

reset
Nouveau dans la version 0.7 : Supporté depuis wlc 0.7.
Annule les modifications Weblate pour correspondre au dépôt distant (traduction, composant ou projet).

cleanup
Nouveau dans la version 0.9 : Supporté depuis wlc 0.9.
Removes any untracked changes in a Weblate object to match the remote repository (translation, component or project).

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

stats
Affiche les statistiques détaillées d’un objet Weblate (traduction, composant ou projet).

lock-status
Nouveau dans la version 0.5 : Supporté depuis wlc 0.5.
Affiche l’état du verrouillage.

lock
Nouveau dans la version 0.5 : Supporté depuis wlc 0.5.
Locks component from further translation in Weblate.

unlock
Nouveau dans la version 0.5 : Supporté depuis wlc 0.5.
Déverrouille la traduction d’un composant Weblate.

changes
Nouveau dans la version 0.7 : Supporté depuis wlc 0.7 et Weblate 2.10.
Affiche les modifications d’un objet.

download
Nouveau dans la version 0.7 : Supporté depuis wlc 0.7.
Télécharge un fichier de traduction.

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

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

upload
Nouveau dans la version 0.9 : Supporté depuis wlc 0.9.
Charge un fichier de traduction.

--overwrite
Overwrite existing translations upon uploading.

--input
Fichier à partir duquel le contenu est lu. S’il n’est pas spécifié, le contenu est lu à partir de l’entrée standard stdin.
--method
  Upload method to use, see Import methods.
--fuzzy
  Fuzzy (marked for edit) strings processing (empty, process, approve)
--author-name
  Nom de l'auteur, pour surcharger l'utilisateur actuellement connecté
--author-email
  Auteur du courriel, pour surcharger l'utilisateur actuellement connecté

**Indication :** Vous pouvez obtenir plus d’informations sur la commande en l’appelant avec l’option `--help`, par exemple: `wlc ls --help`.

### 1.13.6 Fichiers de configuration

**.weblate, .weblate.ini, weblate.ini**
  Modifié dans la version 1.6 : Les fichiers avec l’extension `.ini` sont également acceptés.
  Fichier de configuration par projet

**C:\Users\NAME\AppData\weblate.ini**
  Nouveau dans la version 1.6.
  Fichier de configuration utilisateur sous Windows.

**~/.config/weblate**
  Fichier de configuration utilisateur.

**/etc/xdg/weblate**
  System wide configuration file

Le programme suit les spécifications XDG, donc vous pouvez ajuster l’emplacement des fichiers de configuration par les variables d’environnement `XDG_CONFIG_HOME` ou `XDG_CONFIG_DIRS`. Sur Windows, le répertoire **APPDATA** est la localisation préférée pour le fichier de configuration.

Les paramètres suivants peuvent être configurés dans la section `[weblate]` (vous pouvez les personnaliser en utilisant `--config-section`):

- **key**
  Clé d’API pour accéder à Weblate.

- **url**
  URL du serveur API, par défaut `http://127.0.0.1:8000/api/`.

- **translation**
  Chemin vers la traduction par défaut - composant ou projet.

Le fichier de configuration est un fichier INI, par exemple :

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

De plus, les clés API peuvent être stockées dans la section `[keys]` :

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

Cela vous permet de stocker des clés dans vos paramètres personnels, tout en utilisant la configuration `.weblate` dans le répertoire de contrôle versionaire (VCS) afin que `wlc` sache lequel des serveurs il doit parler à.
## 1.13.7 Exemples

Imprimer la version actuelle du programme :

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

Lister tous les projets :

```
$ 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/
```

Téléverser un fichier de traduction :

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

You can also designate what project wlc should work on :

```
$ cat .weblate
[weblate]
url = https://hosted.weblate.org/api/
translation = weblate/application
```

```
$ wlc show
branch: main
file_format: po
source_language: en
filemask: weblate/locale/*/LC_MESSAGES/django.po
git_export: https://hosted.weblate.org/git/weblate/application/
license: GPL-3.0+
license_url: https://spdx.org/licenses/GPL-3.0+
name: Application
new_base: weblate/locale/django.pot
project: weblate
repo: git://github.com/WeblateOrg/weblate.git
slug: application
template:
url: https://hosted.weblate.org/api/components/weblate/application/
vcs: git
web_url: https://hosted.weblate.org/projects/weblate/application/
```

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

```
$ wlc commit
```

## 1.14 API Python de Weblate

### 1.14.1 Installation

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

```
pip install wlc
```
1.14.2 wlc

WeblateException

exception wlc.WeblateException
   Base class for all exceptions.

Weblate

class wlc.Weblate (key='', url=None, config=None)
   
   Paramètres
   — key (str) – User key
   — url (str) – API server URL, if not specified default is used
   — config (wlc.config.WeblateConfig) – Configuration object, overrides any other parameters.

Access class to the API, define API key and optionally API URL.

get (path)
   
   Paramètres
   — path (str) – Request path

Type renvoyé
   object

Performs a single API GET call.

post (path, **kwargs)
   
   Paramètres
   — path (str) – Request path

Type renvoyé
   object

Performs a single API GET call.

1.14.3 wlc.config

WeblateConfig

class wlc.config.WeblateConfig (section='wlc')
   
   Paramètres
   — section (str) – Configuration section to use

Configuration file parser following XDG specification.

load (path=) None)
   
   Paramètres
   — path (str) – Path from which to load configuration.

Loads configuration from a file, if none is specified, it loads from the wlc configuration file (~/.config/wlc) placed in your XDG configuration path (/etc/xdg/wlc).
1.14.4 wlc.main

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

Paramètres
— settings (list) – Settings to override as list of tuples
— stdout (object) – stdout file object for printing output, uses sys.stdout as default
— args (list) – Command-line arguments to process, uses sys.args as default

Main entry point for command-line interface.

@wlc.main.register_command(command)

Decorator to register Command class in main parser used by main().

Commande

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

Main class for invoking commands.
2.1 Instructions de configuration

2.1.1 Installer Weblate

Installing using Docker

With dockerized Weblate deployment you can get your personal Weblate instance up and running in seconds. All of Weblate’s dependencies are already included. PostgreSQL is set up as the default database.

Exigences matérielles

Weblate should run on any contemporary hardware without problems, the following is the minimal configuration required to run Weblate on a single host (Weblate, database and webserver) :

— 3 GB of RAM
— 2 CPU cores
— 1 GB of storage space

The more memory the better - it is used for caching on all levels (filesystem, database and Weblate).

Many concurrent users increases the amount of needed CPU cores. For hundreds of translation components at least 4 GB of RAM is recommended.

The typical database storage usage is around 300 MB per 1 million hosted words. Storage space needed for cloned repositories varies, but Weblate tries to keep their size minimal by doing shallow clones.

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

The following examples assume you have a working Docker environment, with `docker-compose` installed. Please check the Docker documentation for instructions.

1. Clone the weblate-docker repo:

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

2. Create a `docker-compose.override.yml` file with your settings. See [Docker environment variables](#) for full list of environment variables.

   ```yaml
   version: '3'
services:
   weblate:
   ports:
   - 80:8080
   environment:
   WEBLATE_EMAIL_HOST: smtp.example.com
   WEBLATE_EMAIL_HOST_USER: user
   WEBLATE_EMAIL_HOST_PASSWORD: pass
   WEBLATE_SERVER_EMAIL: weblate@example.com
   WEBLATE_DEFAULT_FROM_EMAIL: weblate@example.com
   WEBLATE_SITE_DOMAIN: weblate.example.com
   WEBLATE_ADMIN_PASSWORD: password for the admin user
   WEBLATE_ADMIN_EMAIL: weblate.admin@example.com
   ```

   **Note:** If `WEBLATE_ADMIN_PASSWORD` is not set, the admin user is created with a random password shown on first startup.

   The provided example makes Weblate listen on port 80, edit the port mapping in the `docker-compose.override.yml` file to change it.

3. Start Weblate containers:

   ```bash
docker-compose up
   ```

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

   Modifié dans la version 2.15-2 : The setup has changed recently, priorly there was separate web server container, since 2.15-2 the web server is embedded in the Weblate container.

   Modifié dans la version 3.7.1-6 : In July 2019 (starting with the 3.7.1-6 tag), the containers are not running as a root user. This has changed the exposed port from 80 to 8080.

   **Voir aussi :**

   *Invoking management commands*

**Choosing Docker hub tag**

You can use following tags on Docker hub, see https://hub.docker.com/r/weblate/weblate/tags/ for full list of available ones.
Every image is tested by our CI before it gets published, so even the bleeding version should be quite safe to use.

**Docker container with HTTPS support**

Please see *Installation* for generic deployment instructions, this section only mentions differences compared to it.

**Using own SSL certificates**

Nouveau dans la version 3.8-3.

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

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

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

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

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

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

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

    server_name <SITE_URL>;
    ssl_certificate /etc/letsencrypt/live/<SITE>/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/<SITE>/privkey.pem;

    location / {  
        proxy_set_header HOST $host;
        proxy_set_header X-Forwarded-Proto https;
    }
}
```

(suite sur la page suivante)
The Weblate database should be automatically migrated on first startup, and there should be no need for additional manual actions.

Automatic SSL certificates using Let’s Encrypt

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

```
version: '3'
services:  
  weblate:
    environment:
      WEBLATE_EMAIL_HOST: smtp.example.com
      WEBLATE_EMAIL_HOST_USER: user
      WEBLATE_EMAIL_HOST_PASSWORD: pass
      WEBLATE_SITE_DOMAIN: weblate.example.com
      WEBLATE_ADMIN_PASSWORD: password for admin user
  https-portal:
    environment:
      DOMAINS: 'weblate.example.com -> http://weblate:8080'
```

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

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

Upgrading the Docker container

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

Modifié dans la version 4.10-1 : Since Weblate 4.10-1, the Docker container uses Django 4.0 what requires PostgreSQL 10 or newer, please upgrade it prior to upgrading Weblate. See Upgrade from 4.9 to 4.10 and Upgrading PostgreSQL container.

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

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

Replace `<SITE_URL>`, `<SITE>` and `<EXPOSED_DOCKER_PORT>` with actual values from your environment.
Note: Upgrades across major versions are not supported by Weblate. For example, if you are on 3.x series and want to upgrade to 4.x, first upgrade to the latest 4.0.x-y image (at time of writing this it is the 4.0.4-5), which will do the migration and then continue upgrading to newer versions.

You might also want to update the docker-compose repository, though it’s not needed in most case. See Upgrading PostgreSQL container for upgrading the PostgreSQL server.

Upgrading PostgreSQL container

PostgreSQL containers do not support automatic upgrading between version, you need to perform the upgrade manually. Following steps show one of the options of upgrading.

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

1. Arrêtez le conteneur Weblate :

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

2. Backup the database :

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

3. Stop the database container :

   ```bash
docker-compose stop database
   ```

4. Supprimer le volume PostgreSQL :

   ```bash
docker-compose rm -v database
   docker volume remove weblate-docker_postgres-data
   ```

5. Adjust docker-compose.yml to use new PostgreSQL version.

6. Start the database container :

   ```bash
docker-compose up -d database
   ```

7. Restaurer la base de données à partir de la sauvegarde :

   ```bash
cat backup.sql | docker-compose exec -T database psql --username weblate --dbname postgres
   ```

8. (Optional) Update password for the Weblate user. This might be needed when migrating to PostgreSQL 14 or 15 as way of storing passwords has been changed :

   ```bash
docker-compose exec -T database psql --username weblate --dbname postgres -c "ALTER USER weblate WITH PASSWORD 'weblate'"
   ```

9. Démarrer tous les conteneurs restants :

   ```bash
docker-compose up -d
   ```
Connexion en tant qu’administrateur

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.

Voir aussi :
WEBLATE_ADMIN_PASSWORD, WEBLATE_ADMIN_NAME, WEBLATE_ADMIN_EMAIL

Number of processes and memory consumption

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

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

```
environment:
  WEBLATE_WORKERS: 2
```

You can also fine-tune individual worker categories:

```
environment:
  WEB_WORKERS: 4
  CELERY_MAIN_OPTIONS: --concurrency 2
  CELERY_NOTIFY_OPTIONS: --concurrency 1
  CELERY_TRANSLATE_OPTIONS: --concurrency 1
```

Voir aussi :
WEBLATE_WORKERS, CELERY_MAIN_OPTIONS, CELERY_NOTIFY_OPTIONS, CELERY_TRANSLATE_OPTIONS, CELERY_MEMORY_OPTIONS, CELERY_BACKUP_OPTIONS, CELERY_BEAT_OPTIONS, WEB_WORKERS

Scaling horizontally

Nouveau dans la version 4.6.

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 Configuration can be set in the Docker container using the environment variables described below.

If you need to define a setting not exposed through Docker environment variables, see Configuration beyond environment variables.
Paramètres généraux

**WEBLATE_DEBUG**

Configure Django debug mode using `DEBUG`.

Exemple :

```
environment:
  WEBLATE_DEBUG: 1
```

Voir aussi :
* Disable debug mode

**WEBLATE_LOGLEVEL**

Configure the logging verbosity.

**WEBLATE_LOGLEVEL_DATABASE**

Configure the logging of the database queries verbosity.

**WEBLATE_SITE_TITLE**

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

**WEBLATE_SITE_DOMAIN**

Configures the site domain. This parameter is required.

Voir aussi :
* Set correct site domain, SITE_DOMAIN

**WEBLATE_ADMIN_NAME**

**WEBLATE_ADMIN_EMAIL**

Configure 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).

Exemple :

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

Voir aussi :
* Connexion en tant qu’administrateur, Properly configure admins, ADMINS

**WEBLATE_ADMIN_PASSWORD**

Sets the password for the `admin` user.

— If not set and `admin` user does not exist, it is created with a random password shown on first container startup.

— If not set and `admin` user exists, no action is performed.

— If set the `admin` user is adjusted on every container startup to match `WEBLATE_ADMIN_PASSWORD, WEBLATE_ADMIN_NAME` and `WEBLATE_ADMIN_EMAIL`.

Avertissement : 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.

Voir aussi :
* Connexion en tant qu’administrateur, WEBLATE_ADMIN_PASSWORD, WEBLATE_ADMIN_PASSWORD_FILE, WEBLATE_ADMIN_NAME, WEBLATE_ADMIN_EMAIL

**WEBLATE_ADMIN_PASSWORD_FILE**

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

Voir aussi :
* `WEBLATE_ADMIN_PASSWORD`
**WEBLATE_SERVER_EMAIL**

The email address that error messages are sent from.

Voir aussi :
SERVER_EMAIL, Configure e-mail sending

**WEBLATE_DEFAULT_FROM_EMAIL**

Configures the address for outgoing e-mails.

Voir aussi :
DEFAULT_FROM_EMAIL, Configure e-mail sending

**WEBLATE_CONTACT_FORM**

Configures contact form behavior, see CONTACT_FORM.

**WEBLATE_ALLOWED_HOSTS**

Configures allowed HTTP hostnames using ALLOWED_HOSTS.

Defaults to * which allows all hostnames.

Exemple :

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

Voir aussi :
ALLOWED_HOSTS, Allowed hosts setup, Set correct site domain

**WEBLATE_REGISTRATION_OPEN**

Configures whether registrations are open by toggling REGISTRATION_OPEN.

Exemple :

```
environment:
  WEBLATE_REGISTRATION_OPEN: 0
```

**WEBLATE_REGISTRATION_ALLOW_BACKENDS**

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

Exemple :

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

**WEBLATE_REGISTRATION_REBIND**

Nouveau dans la version 4.16.

Configures REGISTRATION_REBIND.

**WEBLATE_TIME_ZONE**

Configures the used time zone in Weblate, see TIME_ZONE.

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

Exemple :

```
environment:
  WEBLATE_TIME_ZONE: Europe/Prague
```

**WEBLATE_ENABLE_HTTPS**

Makes Weblate assume it is operated behind a reverse HTTPS proxy, it makes Weblate use HTTPS in e-mail and API links or set secure flags on cookies.
Indication : Please see ENABLE_HTTPS documentation for possible caveats.

Note : 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.

Exemple :

```bash
environment:
  WEBLATE_ENABLE_HTTPS: 1
```

Voir aussi :
ENABLE_HTTPS Set correct site domain, WEBLATE_SECURE_PROXY_SSL_HEADER

WEBLATE_INTERLEDGER_PAYMENT_POINTERS

Nouveau dans la version 4.12.1.

Lets Weblate set the `meta[name=monetization]` field in the head of the document. If multiple are specified, chooses one randomly.

Voir aussi :
INTERLEDGER_PAYMENT_POINTERS

WEBLATE_IP_PROXY_HEADER

Lets Weblate fetch the IP address from any given HTTP header. Use this when using a reverse proxy in front of the Weblate container.

Enables IP_BEHIND_REVERSE_PROXY and sets IP_PROXY_HEADER.

Note : 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.

Exemple :

```bash
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.

Exemple :

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

Voir aussi :
SECURE_PROXY_SSL_HEADER

WEBLATE_REQUIRE_LOGIN

Enables REQUIRE_LOGIN to enforce authentication on whole Weblate.

Exemple :

```bash
environment:
  WEBLATE_REQUIRE_LOGIN: 1
```

WEBLATE_LOGIN_REQUIRED_URLS_EXCEPTIONS
WEBLATE_ADD_LOGIN_REQUIRED_URLS_EXCEPTIONS

Adds URL exceptions for authentication required for the whole Weblate installation using \texttt{LOGIN_REQUIRED_URLS_EXCEPTIONS}.

You can either replace whole settings, or modify default value using \texttt{ADD} and \texttt{REMOVE} variables.

WEBLATE_ADD_LOGIN_REQUIRED_URLS_EXCEPTIONS

You can either replace whole settings, or modify default value using \texttt{ADD} and \texttt{REMOVE} variables.

WEBLATE_GOOGLE_ANALYTICS_ID

Configures ID for Google Analytics by changing \texttt{GOOGLE_ANALYTICS_ID}.

WEBLATE_GITHUB_USERNAME

WEBLATE_GITHUB_TOKEN

WEBLATE_GITHUB_HOST

Configures GitHub pull-requests integration by changing \texttt{GITHUB_CREDENTIALS}.

Voir aussi :

\textit{Requêtes de fusion GitHub}

WEBLATE_GITLAB_USERNAME

WEBLATE_GITLAB_TOKEN

WEBLATE_GITLAB_HOST

Configures GitLab merge-requests integration by changing \texttt{GITLAB_CREDENTIALS}.

Voir aussi :

\textit{Requêtes de fusion GitLab}

WEBLATE_GITEA_USERNAME

WEBLATE_GITEA_TOKEN

WEBLATE_GITEA_HOST

Configures Gitea pull-requests integration by changing \texttt{GITEA_CREDENTIALS}.

Voir aussi :

\textit{Gitea pull requests}

WEBLATE_PAGURE_USERNAME

WEBLATE_PAGURE_TOKEN

WEBLATE_PAGURE_HOST

Configures Pagure merge-requests integration by changing \texttt{PAGURE_CREDENTIALS}.

Voir aussi :

\textit{Requêtes de fusion Pagure}

WEBLATE_BITBUCKETSERVER_USERNAME

WEBLATE_BITBUCKETSERVER_TOKEN

WEBLATE_BITBUCKETSERVER_HOST

Configures Bitbucket Server pull-requests integration by changing \texttt{BITBUCKETSERVER_CREDENTIALS}.

Voir aussi :

\textit{Bitbucket Server pull requests}

WEBLATE_DEFAULT_PULL_MESSAGE

Configures the default title and message for pull requests via API by changing \texttt{DEFAULT_PULL_MESSAGE}.

Voir aussi :

\texttt{DEFAULT_PULL_MESSAGE}
WEBLATE_SIMPLIFY_LANGUAGES
Configures the language simplification policy, see SIMPLIFY_LANGUAGES.

WEBLATE_DEFAULT_ACCESS_CONTROL
Configures the default Contrôle d'accès for new projects, see DEFAULT_ACCESS_CONTROL.

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

WEBLATE_DEFAULT_TRANSLATION_PROPAGATION
Configures the default value for Permettre la propagation de la traduction for new components, see DEFAULT_TRANSLATION_PROPAGATION.

WEBLATE_DEFAULT_COMMITER_EMAIL
Configures DEFAULT_COMMITER_EMAIL.

WEBLATE_DEFAULT_COMMITER_NAME
Configures DEFAULT_COMMITER_NAME.

WEBLATE_DEFAULT_SHARED_TM
Configures DEFAULT_SHARED_TM.

WEBLATE_AKISMET_API_KEY
Configures the Akismet API key, see AKISMET_API_KEY.

WEBLATE_GPG_IDENTITY
Configures GPG signing of commits, see WEBLATE_GPG_IDENTITY.

Voir aussi : Signing Git commits with GnuPG

WEBLATE_URL_PREFIX
Configures URL prefix where Weblate is running, see URL_PREFIX.

WEBLATE_SILENCED_SYSTEM_CHECKS
Configures checks which you do not want to be displayed, see SILENCED_SYSTEM_CHECKS.

WEBLATE_CSP_SCRIPT_SRC

WEBLATE_CSP_IMG_SRC

WEBLATE_CSP_CONNECT_SRC

WEBLATE_CSP_STYLE_SRC

WEBLATE_CSP_FONT_SRC
Allows to customize Content-Security-Policy HTTP header.

Voir aussi : Content security policy, CSP_SCRIPT_SRC, CSP_IMG_SRC, CSP_CONNECT_SRC, CSP_STYLE_SRC, CSP_FONT_SRC

WEBLATE_LICENSE_FILTER
Configures LICENSE_FILTER.

WEBLATE_LICENSE_REQUIRED
Configures LICENSE_REQUIRED

WEBLATE_WEBSITE_REQUIRED
Configures WEBSITE_REQUIRED

WEBLATE_HIDE_VERSION
Configures HIDE_VERSION.

2.1. Instructions de configuration
The Weblate Manual, Version 4.16

WEBLATE_BASIC_LANGUAGES
Configure BASIC_LANGUAGES.

WEBLATE_DEFAULT_AUTO_WATCH
Configures DEFAULT_AUTO_WATCH.

WEBLATE_RATELIMIT_ATTEMPTS
WEBLATE_RATELIMIT_LOCKOUT
WEBLATE_RATELIMIT_WINDOW
Nouveau dans la version 4.6.
Configures rate limiter.

Indication : You can set configuration for any rate limiter scopes. To do that add WEBLATE_ prefix to any of setting described in Limite de requêtes.

Voir aussi :
Limite de requêtes, RATELIMIT_ATTEMPTS, RATELIMIT_WINDOW, RATELIMIT_LOCKOUT

WEBLATE_API_RATELIMIT_ANON
WEBLATE_API_RATELIMIT_USER
Nouveau dans la version 4.11.
Configures API rate limiting. Defaults to 100/day for anonymous and 5000/hour for authenticated users.
Voir aussi :
API rate limiting

WEBLATE_ENABLE_HOOKS
Nouveau dans la version 4.13.
Configures ENABLE_HOOKS.

WEBLATE_ENABLE_AVATARS
Nouveau dans la version 4.6.1.
Configures ENABLE_AVATARS.

WEBLATE_AVATAR_URL_PREFIX
Nouveau dans la version 4.15.
Configures AVATAR_URL_PREFIX.

WEBLATE_LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH
Nouveau dans la version 4.9.
Configure LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH.

WEBLATE_SSH_EXTRA_ARGS
Nouveau dans la version 4.9.
Configures SSH_EXTRA_ARGS.

WEBLATE_BORG_EXTRA_ARGS
Nouveau dans la version 4.9.
Configures BORG_EXTRA_ARGS.

WEBLATE_ENABLE_SHARING
Configures ENABLE_SHARING.

WEBLATE_EXTRA_HTML_HEAD
Nouveau dans la version 4.15.
Configures EXTRA_HTML_HEAD.
WEBLATE_PRIVATE_COMMIT_EMAIL_TEMPLATE
Nouveau dans la version 4.15.
Configures PRIVATE_COMMIT_EMAIL_TEMPLATE.

WEBLATE_PRIVATE_COMMIT_EMAIL_OPT_IN
Nouveau dans la version 4.15.
Configures PRIVATE_COMMIT_EMAIL_OPT_IN.

WEBLATE_CORS_ALLOWED_ORIGINS
Nouveau dans la version 4.16.
Allow CORS requests from given origins.
Exemple :

```bash
environment:
```

Automatic suggestion settings

Modifié dans la version 4.13 : Automatic suggestion services are now configured in the user interface, see Configuring automatic suggestions.

The existing environment variables are imported during the migration to Weblate 4.13, but changing them will not have any further effect.

Paramètres d’authentification

LDAP

WEBLATE_AUTH_LDAP_SERVER_URI
WEBLATE_AUTH_LDAP_USER_DN_TEMPLATE
WEBLATE_AUTH_LDAP_USER_ATTR_MAP
WEBLATE_AUTH_LDAP_BIND_DN
WEBLATE_AUTH_LDAP_BIND_PASSWORD
WEBLATE_AUTH_LDAP_BIND_PASSWORD_FILE
Path to the file containing the LDAP server bind password.
Voir aussi :
WEBLATE_AUTH_LDAP_BIND_PASSWORD

WEBLATE_AUTH_LDAP_CONNECTION_OPTION_REFERRALS
WEBLATE_AUTH_LDAP_USER_SEARCH
WEBLATE_AUTH_LDAP_USER_SEARCH_FILTER
WEBLATE_AUTH_LDAP_USER_SEARCH_UNION
WEBLATE_AUTH_LDAP_USER_SEARCH_UNION_DELIMITER
LDAP authentication configuration.
Example for direct bind :
**environment:**

```ini
WEBLATE_AUTH_LDAP_SERVER_URI: ldap://ldap.example.org
WEBLATE_AUTH_LDAP_USER_DN_TEMPLATE: uid=%(user)s,ou=People,dc=example,dc=net
# map weblate 'full_name' to ldap 'name' and weblate 'email' attribute to ldap 'mail' attribute.
# another example that can be used with OpenLDAP: 'full_name:cn,email:mail'
WEBLATE_AUTH_LDAP_USER_ATTR_MAP: full_name:name,email:mail
```

**Example for search and bind:**

```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
```

**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_UNION: ou=users,dc=example,dc=com|ou=otherusers,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)
```

**Voir aussi :**
*S'authentifier avec LDAP*

**GitHub**

```ini
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
```

Active *S'authentifier avec GitHub.*
## Instructions de configuration

### Bitbucket

- `WEBLATE_SOCIAL_AUTH_BITBUCKET_OAUTH2_KEY`
- `WEBLATE_SOCIAL_AUTH_BITBUCKET_OAUTH2_SECRET`
- `WEBLATE_SOCIAL_AUTH_BITBUCKET_KEY`
- `WEBLATE_SOCIAL_AUTH_BITBUCKET_SECRET`

Active: S'authentifier avec Bitbucket.

### Facebook

- `WEBLATE_SOCIAL_AUTH_FACEBOOK_KEY`
- `WEBLATE_SOCIAL_AUTH_FACEBOOK_SECRET`

Active: Facebook OAuth 2.

### Google

- `WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_KEY`
- `WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET`
- `WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_WHITELISTED_DOMAINS`
- `WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_WHITELISTED_EMAILS`

Active: Google OAuth 2.

### GitLab

- `WEBLATE_SOCIAL_AUTH_GITLAB_KEY`
- `WEBLATE_SOCIAL_AUTH_GITLAB_SECRET`
- `WEBLATE_SOCIAL_AUTH_GITLAB_API_URL`

Active: GitLab OAuth 2.

### Gitea

- `WEBLATE_SOCIAL_AUTH_GITEA_API_URL`
- `WEBLATE_SOCIAL_AUTH_GITEA_KEY`
- `WEBLATE_SOCIAL_AUTH_GITEA_SECRET`

Enables Gitea authentication.
Azure Active Directory

WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_KEY
WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_SECRET

Enables Azure Active Directory authentication, see Active Directory Microsoft Azure.

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 Active Directory Microsoft Azure.

Keycloak

WEBLATE_SOCIAL_AUTH_KEYCLOAK_KEY
WEBLATE_SOCIAL_AUTH_KEYCLOAK_SECRET
WEBLATE_SOCIAL_AUTH_KEYCLOAK_PUBLIC_KEY
WEBLATE_SOCIAL_AUTH_KEYCLOAK_ALGORITHM
WEBLATE_SOCIAL_AUTH_KEYCLOAK_AUTHORIZATION_URL
WEBLATE_SOCIAL_AUTH_KEYCLOAK_ACCESS_TOKEN_URL
WEBLATE_SOCIAL_AUTH_KEYCLOAK_TITLE
WEBLATE_SOCIAL_AUTH_KEYCLOAK_IMAGE

Enables Keycloak authentication, see documentation.

Fournisseurs Linux

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

WEBLATE_SOCIAL_AUTH_FEDORA
WEBLATE_SOCIAL_AUTH_OPENSUSE
WEBLATE_SOCIAL_AUTH_UBUNTU

Slack

WEBLATE_SOCIAL_AUTH_SLACK_KEY
SOCIAL_AUTH_SLACK_SECRET

Enables Slack authentication, see Slack.
OpenID Connect

Nouveau dans la version 4.13-1.

WEBLATE_SOCIAL_AUTH_OIDC_OIDC_ENDPOINT
WEBLATE_SOCIAL_AUTH_OIDC_KEY
WEBLATE_SOCIAL_AUTH_OIDC_SECRET
WEBLATE_SOCIAL_AUTH_OIDC_USERNAME_KEY

Configures generic OpenID Connect integration.

Voir aussi :
OIDC (OpenID Connect)

SAML

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

WEBLATE_SAML_IDP_ENTITY_ID
WEBLATE_SAML_IDP_URL
WEBLATE_SAML_IDP_X509CERT
WEBLATE_SAML_IDP_IMAGE
WEBLATE_SAML_IDP_TITLE

SAML Identity Provider settings, see S’authentifier avec SAML.

Other authentication settings

WEBLATE_NO_EMAIL_AUTH

Disables e-mail authentication when set to any value. See Désactiver l’authentification par mot de passe.

PostgreSQL database setup

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

Voir aussi :
Database setup for Weblate

POSTGRES_PASSWORD

Mot de passe PostgreSQL.

POSTGRES_PASSWORD_FILE

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

POSTGRES_USER

Nom d’utilisateur PostgreSQL.

POSTGRES_DATABASE

PostgreSQL database name.
POSTGRES_HOST
PostgreSQL server hostname or IP address. Defaults to database.

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

POSTGRES_SSL_MODE
Configure how PostgreSQL handles SSL in connection to the server, for possible choices see SSL Mode Descriptions.

POSTGRES_ALTER_ROLE
Configures name of role to alter during migrations, see Configuring Weblate to use PostgreSQL.

POSTGRES_CONN_MAX_AGE
Nouveau dans la version 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 connection to the database. Please adjust your database configuration prior enabling.
Exemple de configuration :

```environment:
  POSTGRES_CONN_MAX_AGE: 3600
```

Voir aussi :
CONN_MAX_AGE, Connexions persistantes

POSTGRES_DISABLE_SERVER_SIDE_CURSORS
Nouveau dans la version 4.9.1.
Disable server side cursors in the database. This is necessary in some pgbouncer setups.
Exemple de configuration :

```environment:
  POSTGRES_DISABLE_SERVER_SIDE_CURSORS: 1
```

Voir aussi :
DISABLE_SERVER_SIDE_CURSORS, Transactions groupées et curseurs côté serveur

Database backup settings

Voir aussi :
Données supprimées pour les sauvegardes

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.

Voir aussi :
Activer la mise en cache

REDIS_HOST
The Redis server hostname or IP address. Defaults to cache.
**REDIS_PORT**

The Redis server port. Defaults to 6379.

**REDIS_DB**

The Redis database number, defaults to 1.

**REDIS_PASSWORD**

The Redis server password, not used by default.

**REDIS_PASSWORD_FILE**

Path to the file containing the Redis server password.

Voir aussi:

**REDIS_PASSWORD**

**REDIS_TLS**

Enables using SSL for Redis connection.

**REDIS_VERIFY_SSL**

Can be used to disable SSL certificate verification for Redis connection.

### Email server setup

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

Example TLS configuration:

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

Example SSL configuration:

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

Voir aussi:

*Configuring outgoing e-mail*

**WEBLATE_EMAIL_HOST**

Mail server hostname or IP address.

Voir aussi:

**WEBLATE_EMAIL_PORT, WEBLATE_EMAIL_USE_SSL, WEBLATE_EMAIL_USE_TLS, EMAIL_HOST**

**WEBLATE_EMAIL_PORT**

Mail server port, defaults to 25.

Voir aussi:

**EMAIL_PORT**

**WEBLATE_EMAIL_HOST_USER**

Utilisateur authentifié par adresse courriel.

Voir aussi:

**EMAIL_HOST_USER**
WEBLATE_EMAIL_HOST_PASSWORD
Mot de passe d'authentification par adresse courriel.
Voir aussi :
EMAIL_HOST_PASSWORD

WEBLATE_EMAIL_HOST_PASSWORD_FILE
Path to the file containing the e-mail authentication password.
Voir aussi :
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.
Modifié dans la version 4.11 : The SSL/TLS support is automatically enabled based on the WEBLATE_EMAIL_PORT.
Voir aussi :
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.
Modifié dans la version 4.11 : The SSL/TLS support is automatically enabled based on the WEBLATE_EMAIL_PORT.
Voir aussi :
WEBLATE_EMAIL_PORT, WEBLATE_EMAIL_USE_SSL, EMAIL_USE_TLS

WEBLATE_EMAIL_BACKEND
Configures Django back-end to use for sending e-mails.
Voir aussi :
Configure e-mail sending, EMAIL_BACKEND

WEBLATE_AUTO_UPDATE
Configures if and how Weblate should update repositories.
Voir aussi :
AUTO_UPDATE

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

Site integration

WEBLATE_GET_HELP_URL
Configures GET_HELP_URL.

WEBLATE_STATUS_URL
Configures STATUS_URL.

WEBLATE_LEGAL_URL
Configures LEGAL_URL.

WEBLATE_PRIVACY_URL
Configures PRIVACY_URL.
Déclaration d’erreurs

It is recommended to collect errors from the installation systematically, see *Collecting error reports*.

To enable support for Rollbar, set the following:

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

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

To enable support for Sentry, set following:

**SENTRY_DSN**
Your Sentry DSN.

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

CDN de localisation

**WEBLATE_LOCALIZE_CDN_URL**

**WEBLATE_LOCALIZE_CDN_PATH**
Nouveau dans la version 4.2.1.
Configuration for *JavaScript localisation 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:

```plaintext
environment:
  WEBLATE_LOCALIZE_CDN_URL: https://cdn.example.com/
  WEBLATE_LOCALIZE_CDN_PATH: /app/data/l10n-cdn
```

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

Voir aussi :
weblate-cdn, LOCALIZE_CDN_URL, LOCALIZE_CDN_PATH

Changing enabled apps, checks, add-ons or autofixes

Nouveau dans la version 3.8-5.

The built-in configuration of enabled checks, add-ons or autofixes can be adjusted by the following variables:

**WEBLATE_ADD_APPS**

**WEBLATE_REMOVE_APPS**

**WEBLATE_ADD_CHECK**

**WEBLATE_REMOVE_CHECK**

**WEBLATE_ADD_AUTOFIX**

**WEBLATE_REMOVE_AUTOFIX**
WEBLATE_ADD_ADDONS

WEBLATE_REMOVE_ADDONS

Exemple :

<table>
<thead>
<tr>
<th>environment:</th>
</tr>
</thead>
<tbody>
<tr>
<td>WEBLATE_REMOVE_AUTOFIX: weblate.trans.autofixes.whitespace.</td>
</tr>
<tr>
<td>SameBookendingWhitespace</td>
</tr>
<tr>
<td>WEBLATE_ADD_ADDONS: customize.addons.MyAddon,customize.addons.OtherAddon</td>
</tr>
</tbody>
</table>

Voir aussi :

CHECK_LIST, AUTOFIX_LIST, WEBLATE_ADDONS, INSTALLED_APPS

Paramètres du conteneur

WEBLATE_WORKERS

Nouveau dans la version 4.6.1.

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

It is used to determine CELERY_MAIN_OPTIONS, CELERY_NOTIFY_OPTIONS, CELERY_MEMORY_OPTIONS, CELERY_TRANSLATE_OPTIONS, CELERY_BACKUP_OPTIONS, CELERY_BEAT_OPTIONS, and WEB_WORKERS. You can use these settings to fine-tune.

CELERY_MAIN_OPTIONS

CELERY_NOTIFY_OPTIONS

CELERY_MEMORY_OPTIONS

CELERY_TRANSLATE_OPTIONS

CELERY_BACKUP_OPTIONS

CELERY_BEAT_OPTIONS

These variables allow you to adjust Celery worker options. It can be useful to adjust concurrency (--concurrency 16) or use different pool implementation (--pool=gevent).

By default, the number of concurrent workers is based on WEBLATE_WORKERS.

Exemple :

<table>
<thead>
<tr>
<th>environment:</th>
</tr>
</thead>
<tbody>
<tr>
<td>CELERY_MAIN_OPTIONS: --concurrency 16</td>
</tr>
</tbody>
</table>

Voir aussi :

Celery worker options, Background tasks using Celery

WEB_WORKERS

Configure how many uWSGI workers should be executed.

It defaults to WEBLATE_WORKERS.

Exemple :

<table>
<thead>
<tr>
<th>environment:</th>
</tr>
</thead>
<tbody>
<tr>
<td>WEB_WORKERS: 32</td>
</tr>
</tbody>
</table>

WEBLATE_SERVICE

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

Following services are defined :
celery-beat
Celery task scheduler, only one instance should be running. This container is also responsible for the database structure migrations and it should be started prior others.

celery-backup
Celery worker for backups, only one instance should be running.

celery-celery
Generic Celery worker.

celery-memory
Mémoire de traduction Celery worker.

celery-notify
Notification de Celery worker.

celery-translate
Traduction automatique de Celery worker.

web
Serveur web.

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 and CACHE_DIR. 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.

Voir aussi:
Docker volumes documentation

Configuration beyond environment variables

Docker environment variables are intended to expose most configuration settings of relevance for Weblate installations.

If you find a setting that is not exposed as an environment variable, and you believe that it should be, feel free to ask for it to be exposed in a future version of Weblate.

If you need to modify a setting that is not exposed as a Docker environment variable, you can still do so, either from the data volume or extending the Docker image.

Voir aussi:
Personnaliser Weblate
Overriding settings from the data volume

You can create a file at `/app/data/settings-override.py`, i.e. at the root of the `data volume`, to extend or override settings defined through environment variables.

Overriding settings by extending the Docker image

To override settings at the Docker image level instead of from the data volume:

1. **Create a custom Python package.**

2. Add a module to your package that imports all settings from `weblate.settings_docker`. For example, within the example package structure defined at *Creating a Python module*, you could create a file at `weblate_customization/weblate_customization/settings.py` with the following initial code:

   ```python
   from weblate.settings_docker import *
   ```

3. Create a custom Dockerfile that inherits from the official Weblate Docker image, and then installs your package and points the `DJANGO_SETTINGS_MODULE` environment variable to your settings module:

   ```sh
   FROM weblate/weblate
   USER root
   COPY weblate_customization /usr/src/weblate_customization
   RUN pip install --no-cache-dir /usr/src/weblate_customization
   ENV DJANGO_SETTINGS_MODULE=weblate_customization.settings
   USER 1000
   ```

4. Instead of using the official Weblate Docker image, build a custom image from this Dockerfile file. There is no clean way to do this with `docker-compose.override.yml`. You could add `build: .` to the `webate` node in that file, but then your custom image will be tagged as `weblate/weblate` in your system, which could be problematic.

   So, instead of using the `docker-compose.yml` straight from the official repository, unmodified, and extending it through `docker-compose.override.yml`, you may want to make a copy of the official `docker-compose.yml` file, and edit your copy to replace `image: weblate/weblate` with `build: .`:

   See the Compose file build reference for details on building images from source when using `docker-compose`.

5. Extend your custom settings module to define or redefine settings.

   You can define settings before or after the import statement above to determine which settings take precedence. Settings defined before the import statement can be overridden by environment variables and setting overrides defined in the data volume. Setting defined after the import statement cannot be overridden.

   You can also go further. For example, you can reproduce some of the things that `weblate.docker_settings` does, such as exposing settings as environment variables, or allow overriding settings from Python files in the data volume.
Replacing logo and other static files

Nouveaux dans la version 3.8-5.

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

**Indication:** The files are copied to the corresponding location upon container startup, so a restart of Weblate is needed after changing the content of the volume.

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

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

```
weblate:
  volumes:
    - weblate-data:/app/data
    - ./weblate_customization/weblate_customization:/app/data/python/weblate_customization

  environment:
    WEBLATE_ADD_APPS: weblate_customization
```

Configuring PostgreSQL server

The PostgreSQL container uses default PostgreSQL configuration and it won’t effectively utilize your CPU cores or memory. It is recommended to customize the configuration to improve the performance.

The configuration can be adjusted as described in Database Configuration at https://hub.docker.com/_/postgres. The configuration matching your environment can be generated using https://pgtune.leopard.in.ua/.

Container internals

The container is using `supervisor` to start individual services. In case of Scaling horizontally, it only starts single service in a container.

To check the services status use:

```
docker-compose exec --user weblate weblate supervisorctl status
```

There are individual services for each Celery queue (see Background tasks using Celery for details). You can stop processing some tasks by stopping the appropriate worker:

```
docker-compose exec --user weblate weblate supervisorctl stop celery-translate
```
Installation sur Debian et Ubuntu

Exigences matérielles

Weblate should run on any contemporary hardware without problems, the following is the minimal configuration required to run Weblate on a single host (Weblate, database and webserver):

— 3 GB of RAM
— 2 CPU cores
— 1 GB of storage space

The more memory the better - it is used for caching on all levels (filesystem, database and Weblate).

Many concurrent users increases the amount of needed CPU cores. For hundreds of translation components at least 4 GB of RAM is recommended.

The typical database storage usage is around 300 MB per 1 million hosted words. Storage space needed for cloned repositories varies, but Weblate tries to keep their size minimal by doing shallow clones.

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

Installation

Exigences du système

Installlez les dépendances nécessaires à la construction des modules Python (voir Exigences logicielles):

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

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

```
apt install -y
  tesseract-ocr libtesseract-dev libleptonica-dev
  libldap2-dev libldap-common libsasl2-dev
  libxmlsec1-dev
```

Optionally install software for running production server, see Exécuter un serveur, Database setup for Weblate, Background tasks using Celery. Depending on size of your installation you might want to run these components on dedicated servers.

The local installation instructions:

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

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

# Caching backend: Redis
apt install -y redis-server

# Database server: PostgreSQL
apt install -y postgresql postgresql-contrib
```

(suite sur la page suivante)
# SMTP server

```bash
apt install -y exim4
```

## Modules Python

**Indication:** We’re using virtualenv to install Weblate in a separate environment from your system. If you are not familiar with it, check virtualenv User Guide.

1. Create the virtualenv for Weblate:
   ```bash
   virtualenv ~/weblate-env
   ```

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

3. Install Weblate including all optional dependencies:
   ```bash
   # Install Weblate with all optional dependencies
   pip install "Weblate[all]"
   ```
   Please check Optional dependencies for fine-tuning of optional dependencies.

**Note:** On some Linux distributions running Weblate fails with libffi error:

```bash
ffi_prep_closure(): bad user_data (it seems that the version of the libffi_
---library seen at runtime is different from the 'ffi.h' file seen at compile-
---time)
```

This is caused by incompatibility of binary packages distributed via PyPI with the distribution. To address this, you need to rebuild the package on your system:

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

## Configurer Weblate

**Note:** The following assumes the virtualenv used by Weblate is activated (by executing . ~/weblate-env/bin/activate). If not, specify the full path to the `weblate` command as ~/weblate-env/bin/weblate.


2. Adjust the values in the new `settings.py` file to your liking. You will need to provide at least the database credentials and Django secret key, but you will want more changes for production setup, see Ajuster la configuration.

3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check Database setup for Weblate for a production-ready setup):
   ```bash
   weblate migrate
   ```

4. Create an account for the administrator user and copy its password to the clipboard, and also save it for later use:
5. Collect the static files for your web server (see Exécuter un serveur and Serving static files):

   weblate collectstatic

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

   weblate compress

7. Start the Celery workers. This is not necessary for development purposes, but strongly recommended otherwise. Background tasks using Celery has more info:

   ~/weblate-env/lib/python3.9/site-packages/weblate/examples/celery start

8. Start the development server (Exécuter un serveur details a production setup):

   weblate runserver

**Après installation**

Congratulations, your Weblate server is now running and you can start using it.

— You can now access Weblate on http://localhost:8000/.
— Sign in with admin credentials obtained during installation or register with new users.
— You can now run Weblate commands using weblate command when Weblate virtualenv is active, see Commandes de gestion.
— You can stop the test server with Ctrl+C.
— Review potential issues with your installation either on /manage/performance/ URL (see Interface de gestion) or using weblate check --deploy, see Configuration de production.

**Ajouter une traduction**

1. Open the admin interface (http://localhost:8000/create/project/) and create the project you want to translate. See Configuration du projet for more details.
   All you need to specify here is the project name and its website.
2. Create a component which is the real object for translation - it points to the VCS repository, and selects which files to translate. See Configuration des composants for more details.
   The important fields here are: Nom du composant, Dépôt du code source, and Masque de fichier for finding translatable files. Weblate supports a wide range of formats including GNU gettext, Android string resources, Apple iOS strings, Java properties, Stringsdict format or Fluent format, see Formats de fichiers pris en charge for more details.
3. Once the above is completed (it can be lengthy process depending on the size of your VCS repository, and number of messages to translate), you can start translating.

**Installation sur SUSE et openSUSE**

**Exigences matérielles**

Weblate should run on any contemporary hardware without problems, the following is the minimal configuration required to run Weblate on a single host (Weblate, database and webserver):

— 3 GB of RAM
— 2 CPU cores
— 1 GB of storage space
The more memory the better - it is used for caching on all levels (filesystem, database and Weblate).

Many concurrent users increases the amount of needed CPU cores. For hundreds of translation components at least 4 GB of RAM is recommended.

The typical database storage usage is around 300 MB per 1 million hosted words. Storage space needed for cloned repositories varies, but Weblate tries to keep their size minimal by doing shallow clones.

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

## Installation

### Exigences du système

Installlez les dépendances nécessaires à la construction des modules Python (voir *Exigences logicielles*) :

```bash
zypper install \
    libxslt-devel libxml2-devel freetype-devel libjpeg-devel zlib-devel \
    libyaml-devel libffi-devel cairo-devel pango-devel \
    gobject-introspection-devel libacl-devel python3-pip python3-virtualenv \
    python3-devel git
```

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

```bash
zypper install tesseract-ocr tesseract-devel leptonica-devel
zypper install libldap2-devel libssl2-devel
zypper install libxmlsec1-devel
```

Optionally install software for running production server, see *Exécuter un serveur*, *Database setup for Weblate*, *Background tasks using Celery*. Depending on size of your installation you might want to run these components on dedicated servers.

The local installation instructions :

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

# Web server option 2: Apache with ``mod_wsgi``
zypper install apache2 apache2-mod_wsgi

# Caching backend: Redis
zypper install redis-server

# Database server: PostgreSQL
zypper install postgresql postgresql-contrib

# SMTP server
zypper install postfix
```
### Indication:

We're using virtualenv to install Weblate in a separate environment from your system. If you are not familiar with it, check virtualenv User Guide.

1. Create the virtualenv for Weblate:

   ```bash
   virtualenv ~/weblate-env
   ```

2. Activate the virtualenv for Weblate:

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

3. Install Weblate including all optional dependencies:

   ```bash
   # Install Weblate with all optional dependencies
   pip install "Weblate[all]"
   ```

   Please check Optional dependencies for fine-tuning of optional dependencies.

### Note:

On some Linux distributions running Weblate fails with libffi error:

```bash
ffi_prep_closure(): bad user_data (it seems that the version of the libffi_
       →library seen at runtime is different from the 'ffi.h' file seen at compile-
       →time)
```

This is caused by incompatibility of binary packages distributed via PyPI with the distribution. To address this, you need to rebuild the package on your system:

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

---

### Configurer Weblate

**Note:** The following assumes the virtualenv used by Weblate is activated (by executing . ~/weblate-env/bin/activate). If not, specify the full path to the `weblate` command as `~/weblate-env/bin/weblate`.


2. Adjust the values in the new `settings.py` file to your liking. You will need to provide at least the database credentials and Django secret key, but you will want more changes for production setup, see *Ajuster la configuration*.

3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check *Database setup for Weblate* for a production-ready setup):

   ```bash
   weblate migrate
   ```

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

   ```bash
   weblate createadmin
   ```

5. Collect the static files for your web server (see *Exécuter un serveur* and *Serving static files*):
6. Compress the JavaScript and CSS files (optional, see Compressing client assets):

   ```bash
   weblate compress
   ```

7. Start the Celery workers. This is not necessary for development purposes, but strongly recommended otherwise. Background tasks using Celery has more info:

   ```bash
   ~/weblate-env/lib/python3.9/site-packages/weblate/examples/celery start
   ```

8. Start the development server (Execute un serveur details a production setup):

   ```bash
   weblate runserver
   ```

### Après installation

Congratulations, your Weblate server is now running and you can start using it.

- You can now access Weblate on http://localhost:8000/.
- Sign in with admin credentials obtained during installation or register with new users.
- You can now run Weblate commands using `weblate` command when Weblate virtualenv is active, see Commandes de gestion.
- You can stop the test server with Ctrl+C.
- Review potential issues with your installation either on /manage/performance/ URL (see Interface de gestion) or using `weblate check --deploy`, see Configuration de production.

### Ajouter une traduction

1. Open the admin interface (http://localhost:8000/create/project/) and create the project you want to translate. See Configuration du projet for more details.
   All you need to specify here is the project name and its website.

2. Create a component which is the real object for translation - it points to the VCS repository, and selects which files to translate. See Configuration des composants for more details.
   The important fields here are: Nom du composant, Dépôt du code source, and Masque de fichier for finding translatable files. Weblate supports a wide range of formats including GNU gettext, Android string resources, Apple iOS strings, Java properties, Stringsdict format or Fluent format, see Formats de fichiers pris en charge for more details.

3. Once the above is completed (it can be a lengthy process depending on the size of your VCS repository, and number of messages to translate), you can start translating.

### Installing on RedHat, Fedora and CentOS

### Exigences matérielles

Weblate should run on any contemporary hardware without problems, the following is the minimal configuration required to run Weblate on a single host (Weblate, database and webserver):

- 3 GB of RAM
- 2 CPU cores
- 1 GB of storage space

The more memory the better - it is used for caching on all levels (filesystem, database and Weblate).

Many concurrent users increases the amount of needed CPU cores. For hundreds of translation components at least 4 GB of RAM is recommended.

The typical database storage usage is around 300 MB per 1 million hosted words. Storage space needed for cloned repositories varies, but Weblate tries to keep their size minimal by doing shallow clones.
**Installation**

**Exigences du système**

Installez les dépendances nécessaires à la construction des modules Python (voir *Exigences logicielles*):

```
#!/bin/bash

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
```

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

```
#!/bin/bash

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

dnf install
tessdata
```

Optionally install software for running production server, see *Exécuter un serveur*, Database setup for Weblate, *Background tasks using Celery*. Depending on size of your installation you might want to run these components on dedicated servers.

The local installation instructions:

```
#!/bin/bash

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

# Web server option 2: Apache with `mod_wsgi`
# Caching backend: Redis
# Database server: PostgreSQL
# SMTP server
```

**Modules Python**

**Indication**: We’re using virtualenv to install Weblate in a separate environment from your system. If you are not familiar with it, check virtualenv *User Guide*.

1. Create the virtualenv for Weblate:

   ```
   virtualenv ~/weblate-env
   ```

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

```
# Install Weblate with all optional dependencies
pip install "Weblate[all]"
```

Please check Optional dependencies for fine-tuning of optional dependencies.

Note: On some Linux distributions running Weblate fails with libffi error:

```
ffi_prep_closure(): bad user_data (it seems that the version of the libffi...

library seen at runtime is different from the 'ffi.h' file seen at compile-

time)
```

This is caused by incompatibility of binary packages distributed via PyPI with the distribution. To address this, you need to rebuild the package on your system:

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

###Configurer Weblate

Note: The following assumes the virtualenv used by Weblate is activated (by executing . ~/weblate-env/bin/activate). If not, specify the full path to the `weblate` command as ~/weblate-env/bin/weblate.


2. Adjust the values in the new settings.py file to your liking. You will need to provide at least the database credentials and Django secret key, but you will want more changes for production setup, see Ajuster la configuration.

3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check Database setup for Weblate for a production-ready setup):

```
weblate migrate
```

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

```
weblate createadmin
```

5. Collect the static files for your web server (see Exécuter un serveur and Serving static files):

```
weblate collectstatic
```

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

```
weblate compress
```

7. Start the Celery workers. This is not necessary for development purposes, but strongly recommended otherwise. Background tasks using Celery has more info:

```
~/weblate-env/lib/python3.9/site-packages/weblate/examples/celery start
```

8. Start the development server (Exécuter un serveur details a production setup):

### 2.1. Instructions de configuration
Après installation

Congratulations, your Weblate server is now running and you can start using it.

- You can now access Weblate on http://localhost:8000/.
- Sign in with admin credentials obtained during installation or register with new users.
- You can now run Weblate commands using {weblate} command when Weblate virtualenv is active, see {Commandes de gestion}.
- You can stop the test server with Ctrl+C.
- Review potential issues with your installation either on /manage/performance/ URL (see {Interface de gestion}) or using {weblate} check --deploy, see {Configuration de production}.

Ajouter une traduction

1. Open the admin interface (http://localhost:8000/create/project/) and create the project you want to translate. See {Configuration du projet} for more details.
   All you need to specify here is the project name and its website.
2. Create a component which is the real object for translation - it points to the VCS repository, and selects which files to translate. See {Configuration des composants} for more details.
   The important fields here are: Nom du composant, Dépôt du code source, and Masque de fichier for finding translatable files. Weblate supports a wide range of formats including GNU gettext, Android string resources, Apple iOS strings, Java properties, Stringsdict format or Fluent format, see {Formats de fichiers pris en charge} for more details.
3. Once the above is completed (it can be lengthy process depending on the size of your VCS repository, and number of messages to translate), you can start translating.

Installing on macOS

Exigences matérielles

Weblate should run on any contemporary hardware without problems, the following is the minimal configuration required to run Weblate on a single host (Weblate, database and webserver):

- 3 GB of RAM
- 2 CPU cores
- 1 GB of storage space

The more memory the better - it is used for caching on all levels (filesystem, database and Weblate).

Many concurrent users increases the amount of needed CPU cores. For hundreds of translation components at least 4 GB of RAM is recommended.

The typical database storage usage is around 300 MB per 1 million hosted words. Storage space needed for cloned repositories varies, but Weblate tries to keep their size minimal by doing shallow clones.

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

Exigences du système

Installez les dépendances nécessaires à la construction des modules Python (voir Exigences logicielles):

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

Make sure pip will be able to find the libffi version provided by homebrew — this will be needed during the installation build step.

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

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

```
brew install tesseract
```

Optionally install software for running production server, see Exécuter un serveur, Database setup for Weblate, Background tasks using Celery. Depending on size of your installation you might want to run these components on dedicated servers.

The local installation instructions:

```
# Web server option 1: NGINX and uWSGI
brew install nginx uwsgi

# Web server option 2: Apache with `mod_wsgi`
brew install httpd

# Caching backend: Redis
brew install redis

# Database server: PostgreSQL
brew install postgresql
```

Modules Python

**Indication:** We’re using virtualenv to install Weblate in a separate environment from your system. If you are not familiar with it, check virtualenv User Guide.

1. Create the virtualenv for Weblate:

```
virtualenv ~/weblate-env
```

2. Activate the virtualenv for Weblate:

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

3. Install Weblate including all optional dependencies:

```
# Install Weblate with all optional dependencies
pip install "Weblate[all]"
```

Please check Optional dependencies for fine-tuning of optional dependencies.

**Note:** On some Linux distributions running Weblate fails with libffi error:
ffi_prep_closure(): bad user_data (it seems that the version of the libffi library seen at runtime is different from the 'ffi.h' file seen at compile-time)

This is caused by incompatibility of binary packages distributed via PyPI with the distribution. To address this, you need to rebuild the package on your system:

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

### Configurer Weblate

**Note:** The following assumes the virtualenv used by Weblate is activated (by executing `. ~/weblate-env/bin/activate`). If not, specify the full path to the `weblate` command as `~/weblate-env/bin/weblate`.


2. Adjust the values in the new `settings.py` file to your liking. You will need to provide at least the database credentials and Django secret key, but you will want more changes for production setup, see `Ajuster la configuration`.

3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check `Database setup for Weblate` for a production-ready setup):

   ```bash
   weblate migrate
   ```

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

   ```bash
   weblate createadmin
   ```

5. Collect the static files for your web server (see `Exécuter un serveur` and `Serving static files`):

   ```bash
   weblate collectstatic
   ```

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

   ```bash
   weblate compress
   ```

7. Start the Celery workers. This is not necessary for development purposes, but strongly recommended otherwise. `Background tasks using Celery` has more info:

   ```bash
   ~/weblate-env/lib/python3.9/site-packages/weblate/examples/celery start
   ```

8. Start the development server (`Exécuter un serveur` details a production setup):

   ```bash
   weblate runserver
   ```
Après installation

Congratulations, your Weblate server is now running and you can start using it.
— You can now access Weblate on http://localhost:8000/.
— Sign in with admin credentials obtained during installation or register with new users.
— You can now run Weblate commands using `weblate` command when Weblate virtualenv is active, see Commandes de gestion.
— You can stop the test server with Ctrl+C.
— Review potential issues with your installation either on /manage/performance/ URL (see Interface de gestion) or using `weblate check --deploy`, see Configuration de production.

Ajouter une traduction

1. Open the admin interface (http://localhost:8000/create/project/) and create the project you want to translate. See Configuration du projet for more details.
   All you need to specify here is the project name and its website.
2. Create a component which is the real object for translation - it points to the VCS repository, and selects which files to translate. See Configuration des composants for more details.
   The important fields here are : Nom du composant, Dépôt du code source, and Masque de fichier for finding translatable files. Weblate supports a wide range of formats including GNU gettext, Android string resources, Apple iOS strings, Java properties, Stringsdict format or Fluent format, see Formats de fichiers pris en charge for more details.
3. Once the above is completed (it can be lengthy process depending on the size of your VCS repository, and number of messages to translate), you can start translating.

Installing from sources

1. Please follow the installation instructions for your system first up to installing Weblate:
   — Installation sur Debian et Ubuntu
   — Installation sur SUSE et openSUSE
   — Installing on RedHat, Fedora and CentOS
2. Grab the latest Weblate sources using Git (or download a tarball and unpack that):
   ```bash
git clone https://github.com/WeblateOrg/weblate.git weblate-src
```
   Alternatively you can use released archives. You can download them from our website <https://weblate.org/>.
   Those downloads are cryptographically signed, please see Verifying release signatures.
3. Install current Weblate code into the virtualenv:
   ```bash
   . ~/weblate-env/bin/activate
pip install -e weblate-src
   ``
4. Copy `weblate/settings_example.py` to `weblate/settings.py`.
5. Adjust the values in the new `settings.py` file to your liking. You will need to provide at least the database credentials and Django secret key, but you will want more changes for production setup, see Ajuster la configuration.
6. Create the database used by Weblate, see Database setup for Weblate.
7. Build Django tables, static files and initial data (see Filling up the database and Serving static files):
   ```bash
   weblate migrate
   weblate collectstatic
   weblate compress
   ```

Note : This step should be repeated whenever you update the repository.
Installing on OpenShift

With the OpenShift Weblate template you can get your personal Weblate instance up and running in seconds. All of Weblate’s dependencies are already included. PostgreSQL is set up as the default database and persistent volume claims are used.

You can find the template at <https://github.com/WeblateOrg/openshift/>.

Installation

The following examples assume you have a working OpenShift v3.x environment, with oc client tool installed. Please check the OpenShift documentation for instructions.

The template.yml is suited for running all components in OpenShift. There is also template-external-postgresql.yml which does not start a PostgreSQL server and allows you to configure external PostgreSQL server.

Web Console

Copy the raw content from template.yml and import them into your project, then use the Create button in the OpenShift web console to create your application. The web console will prompt you for the values for all of the parameters used by the template.

CLI

To upload the Weblate template to your current project’s template library, pass the template.yml file with the following command:

```bash
-n <PROJECT>
```

The template is now available for selection using the web console or the CLI.

Paramètres

The parameters that you can override are listed in the parameters section of the template. You can list them with the CLI by using the following command and specifying the file to be used:

```bash

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

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

```
   -p APPLICATION_NAME=weblate \\
   -p WEBLATE_VERSION=4.3.1-1 \\
   -p WEBLATE_SITE_DOMAIN=weblate.app-openshift.example.com \\
   -p POSTGRESQL_IMAGE=docker-registry.default.svc:5000/openshift/postgresql:9.6 \\
   -p REDIS_IMAGE=docker-registry.default.svc:5000/openshift/redis:3.2 \\
   | oc create -f
```

The Weblate instance should be available after successful migration and deployment at the specified WEBLATE_SITE_DOMAIN parameter.

After container setup, you can sign in as admin user with password provided in WEBLATE_ADMIN_PASSWORD, or a random password generated on first start if that was not set.

To reset admin password, restart the container with WEBLATE_ADMIN_PASSWORD set to new password in the respective Secret.

Éliminer

```
$ 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
```

Configuration

By processing the template a respective ConfigMap will be created and which can be used to customize the Weblate image. The ConfigMap is directly mounted as environment variables and triggers a new deployment every time it is changed. For further configuration options, see Docker environment variables for full list of environment variables.

Installing on Kubernetes

**Note**: This guide is looking for contributors experienced with Kubernetes to cover the setup in more details.

With the Kubernetes Helm chart you can get your personal Weblate instance up and running in seconds. All of Weblate’s dependencies are already included. PostgreSQL is set up as the default database and persistent volume claims are used.

You can find the chart at <https://github.com/WeblateOrg/helm/> and it can be displayed at <https://artifacthub.io/packages/helm/weblate/weblate>.  

---

2.1. Instructions de configuration
Installation

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

Configuration

For further configuration options, see Docker environment variables for full list of environment variables.

Depending on your setup and experience, choose an appropriate installation method for you:
- Installing using Docker, recommandé pour les configurations de production.
- Virtualenv installation, recommended for production setups:
  - Installation sur Debian et Ubuntu
  - Installation sur SUSE et openSUSE
  - Installing on RedHat, Fedora and CentOS
  - Installing on macOS
- Installing from sources, recommandé pour le développement.
- Installing on OpenShift
- Installing on Kubernetes

2.1.2 Exigences logicielles

Système d’exploitation

Weblate est connu pour fonctionner sous Linux, FreeBSD et macOS. D’autres systèmes de type Unix fonctionneront très probablement aussi.

Weblate is not supported on Windows. But it may still work and patches are happily accepted.

Autres services

Weblate is using other services for its operation. You will need at least following services running:
- PostgreSQL database server, see Database setup for Weblate.
- Redis server for cache and tasks queue, see Background tasks using Celery.
- SMTP server for outgoing e-mail, see Configuring outgoing e-mail.

Dépendances python

Weblate is written in Python and supports Python 3.6 or newer. You can install dependencies using pip or from your distribution packages, full list is available in requirements.txt.

Most notable dependencies:
- Django
  https://www.djangoproject.com/
- Celery
  https://docs.celeryq.dev/
- Boîte à outils de traduction
  https://toolkit.translatehouse.org/
- translation-finder
  https://github.com/WeblateOrg/translation-finder
- Python Social Auth
  https://python-social-auth.readthedocs.io/
- Django REST Framework
  https://www.django-rest-framework.org/
Optional dependencies

Les modules suivants sont nécessaires pour certaines fonctionnalités de Weblate. Vous pouvez les trouver tous dans requirements-optional.txt.

**Mercurial** (optional for Mercurial repositories support)
- https://www.mercurial-scm.org/

**phply** (optional for Chaînes de caractères PHP)
- https://github.com/viraptor/phply

**tesserocr** (optional for OCR in Visual context for strings)
- https://github.com/sirfz/tesserocr

**python-akismet** (optional for Protection contre le spam)
- https://github.com/Nekmo/python-akismet

**ruamel.yaml** (facultatif pour YAML files)
- https://pypi.org/project/ruamel.yaml/

**Zeep** (optional for Microsoft Terminology)
- https://docs.python-zeep.org/

**aeidon** (facultatif pour Subtitle files)
- https://pypi.org/project/aeidon/

**fluent.syntax** (optional for Fluent format)
- https://projectfluent.org/

**Indication :** When installing using pip, you can directly specify desired features when installing:

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

Or you can install Weblate with all optional features:

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

Or you can install Weblate without any optional features:

```
-pip install Weblate
```

Database backend dependencies

Weblate supports PostgreSQL, MySQL and MariaDB, see Database setup for Weblate and backends documentation for more details.

Autres exigences système requises

The following dependencies have to be installed on the system:

**Git**
- https://git-scm.com/

**Pango**, Cairo and related header files and GObject introspection data
- https://cairographics.org/, https://pango.gnome.org/, see Pango and Cairo

**git-review** (facultatif pour la prise en charge de Gerrit)
- https://pypi.org/project/git-review/

**git-svn** (facultatif pour la prise en charge de Subversion)
- https://git-scm.com/docs/git-svn

**tesseract** et ses données (facultatif pour la ROC des captures d’écran)
- https://github.com/tesseract-ocr/tesseract

**licensee** (facultatif pour la détection des licences lors de la création des composants)
- https://github.com/licensee/licensee

2.1. Instructions de configuration
Build-time dependencies

To build some of the Dépendances python you might need to install their dependencies. This depends on how you install them, so please consult individual packages for documentation. You won’t need those if using prebuilt Wheels while installing using pip or when you use distribution packages.

Pango and Cairo

Modifié dans la version 3.7.

Weblate uses Pango and Cairo for rendering bitmap widgets (see promotion) and rendering checks (see Gestion des polices). To properly install Python bindings for those you need to install system libraries first - you need both Cairo and Pango, which in turn need GLib. All those should be installed with development files and GObject introspection data.

2.1.3 Verifying release signatures

Weblate release are cryptographically signed by the releasing developer. Currently this is Michal Čihař. Fingerprint of his PGP key is:

```
63CB 1DF1 EF12 CF2A C0EE 5A32 9C27 B313 42B7 511D
```

and you can get more identification information from <https://keybase.io/nijel>.

You should verify that the signature matches the archive you have downloaded. This way you can be sure that you are using the same code that was released. You should also verify the date of the signature to make sure that you downloaded the latest version.

Each archive is accompanied with .asc files which contain the PGP signature for it. Once you have both of them in the same folder, you can verify the signature:

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

As you can see GPG complains that it does not know the public key. At this point you should do one of the following steps:

— Use wkd to download the key:

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

— Download the keyring from Michal’s server, then import it with:

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

— Download and import the key from one of the key servers:
This will improve the situation a bit - at this point you can verify that the signature from the given key is correct but you still can not trust the name used in the key:

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

The problem here is that anybody could issue the key with this name. You need to ensure that the key is actually owned by the mentioned person. The GNU Privacy Handbook covers this topic in the chapter Validating other keys on your public keyring. The most reliable method is to meet the developer in person and exchange key fingerprints, however you can also rely on the web of trust. This way you can trust the key transitively through signatures of others, who have met the developer in person.

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

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

Should the signature be invalid (the archive has been changed), you would get a clear error regardless of the fact that the key is trusted or not:

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

### 2.1.4 Permissions du système de fichiers

The Weblate process needs to be able to read and write to the directory where it keeps data - `DATA_DIR`. All files within this directory should be owned and writable by the user running all Weblate processes (typically WSGI and Celery, see Exécuter un serveur and Background tasks using Celery).

The default configuration places them in the same tree as the Weblate sources, however you might prefer to move these to a better location such as `:/var/lib/weblate`.

Weblate tries to create these directories automatically, but it will fail when it does not have permissions to do so.

You should also take care when running Commandes de gestion, as they should be ran under the same user as Weblate itself is running, otherwise permissions on some files might be wrong.
In the Docker container, all files in the `/app/data` volume have to be owned by the `weblate` user inside the container (UID 1000).

Voir aussi :
*Serving static files*

### 2.1.5 Database setup for Weblate

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

Voir aussi :
*Use a powerful database engine*, Bases de données, *Migrating from other databases to PostgreSQL*

**PostgreSQL**

PostgreSQL is usually the best choice for Django-based sites. It’s the reference database used for implementing Django database layer.

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

Voir aussi :
*Notes sur PostgreSQL*

#### Creating a database in PostgreSQL

It is usually a good idea to run Weblate in a separate database, and separate user account :

```
# If PostgreSQL was not installed before, set the main password
sudo -u postgres psql postgres -c "\password postgres"

# Create a database user called "weblate"
sudo -u postgres createuser --superuser --pwprompt weblate

# Create the database "weblate" owned by "weblate"
sudo -u postgres createdb -E UTF8 -O weblate weblate
```

**Indication** : If you don’t want to make the Weblate user a superuser in PostgreSQL, you can omit that. In that case you will have to perform some of the migration steps manually as a PostgreSQL superuser in schema Weblate will use :

```
CREATE EXTENSION IF NOT EXISTS pg_trgm WITH SCHEMA weblate;
CREATE EXTENSION IF NOT EXISTS btree_gin WITH SCHEMA weblate;
```
Configuring Weblate to use PostgreSQL

The settings.py snippet for PostgreSQL:

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

The database migration performs ALTER ROLE on database role used by Weblate. In most cases the name of the role matches username. In more complex setups the role name is different than username and you will get error about non-existing role during the database migration (psycopg2.errors.UndefinedObject: role "weblate@hostname" does not exist). This is known to happen with Azure Database for PostgreSQL, but it’s not limited to this environment. Please set ALTER_ROLE to change name of the role Weblate should alter during the database migration.

MySQL and MariaDB

**Indication**: Some Weblate features will perform better with PostgreSQL. This includes searching and translation memory, which both utilize full-text features in the database and PostgreSQL implementation is superior.

Weblate can be also used with MySQL or MariaDB, please see Notes sur MySQL and Notes MariaDB for caveats using Django with those. Because of the limitations it is recommended to use PostgreSQL for new installations.

Weblate requires MySQL at least 5.7.8 or MariaDB at least 10.2.7.

Following configuration is recommended for Weblate:

- Use the utf8mb4 charset to allow representation of higher Unicode planes (for example emojis).
- Configure the server with innodb_large_prefix to allow longer indices on text fields.
- Set the isolation level to READ COMMITTED.
- The SQL mode should be set to STRICT_TRANS_TABLES.

MySQL 8.x, MariaDB 10.5.x or newer have reasonable default configuration so that no server tweaking should be necessary and all what is needed can be configured on the client side.

Below is an example /etc/my.cnf.d/server.cnf for a server with 8 GB of RAM. These settings should be sufficient for most installs. MySQL and MariaDB have tunables that will increase the performance of your server that are considered not necessary unless you are planning on having large numbers of concurrent users accessing the system. See the various vendors documentation on those details.

It is absolutely critical to reduce issues when installing that the setting innodb_file_per_table is set properly and MySQL/MariaDB restarted before you start your Weblate install.
Configurer Weblate pour utiliser MySQL/MariaDB

The settings.py snippet for MySQL and MariaDB:

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

You should also create the weblate user account in MySQL or MariaDB before you begin the install. Use the commands below to achieve that:

```sql
GRANT ALL ON weblate.* to 'weblate'@'localhost' IDENTIFIED BY 'password';
FLUSH PRIVILEGES;
```
2.1.6 Autres configurations

Configuring outgoing e-mail

Weblate sends out e-mails on various occasions - for account activation and on various notifications configured by users. For this it needs access to an SMTP server.

The mail server setup is configured using these settings: EMAIL_HOST, EMAIL_HOST_PASSWORD, EMAIL_USE_TLS, EMAIL_USE_SSL, EMAIL_HOST_USER and EMAIL_PORT. Their names are quite self-explanatory, but you can find more info in the Django documentation.

**Indication**: In case you get error about not supported authentication (for example SMTP AUTH extension not supported by server), it is most likely caused by using insecure connection and server refuses to authenticate this way. Try enabling EMAIL_USE_TLS in such case.

**Voir aussi**:

Not receiving e-mails from Weblate, Configuration des courriels sortants dans le conteneur Docker

Running behind reverse proxy

Several features in Weblate rely on being able to get client IP address. This includes Limite de requêtes, Protection contre le spam or Journal d’audit.

In default configuration Weblate parses IP address from REMOTE_ADDR which is set by the WSGI handler.

In case you are running a reverse proxy, this field will most likely contain its address. You need to configure Weblate to trust additional HTTP headers and parse the IP address from these. This can not be enabled by default as it would allow IP address spoofing for installations not using a reverse proxy. Enabling IP_BEHIND_REVERSE_PROXY might be enough for the most usual setups, but you might need to adjust IP_PROXY_HEADER and IP_PROXY_OFFSET as well.

Another thing to take care of is the Host header. It should match to whatever is configured as SITE_DOMAIN. Additional configuration might be needed in your reverse proxy (for example use ProxyPreserveHost On for Apache or proxy_set_header Host $host; with nginx).

**Voir aussi**:

Protection contre le spam, Limite de requêtes, Journal d’audit, IP_BEHIND_REVERSE_PROXY, IP_PROXY_HEADER, IP_PROXY_OFFSET, SECURE_PROXY_SSL_HEADER

Proxy HTTP

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

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

**Voir aussi**:

Proxy Environment Variables
2.1.7 Ajuster la configuration

Voir aussi :

*Configuration d’exemple*

Copy `weblate/settings_example.py` to `weblate/settings.py` and adjust it to match your setup. You will probably want to adjust the following options: `ADMINS`

List of site administrators to receive notifications when something goes wrong, for example notifications on failed merges, or Django errors.

Voir aussi :

`ADMINS`, *Properly configure admins*

`ALLOWED_HOSTS`

You need to set this to list the hosts your site is supposed to serve. For example :

```python
ALLOWED_HOSTS = ['demo.weblate.org']
```

Alternatively you can include wildcard :

```python
ALLOWED_HOSTS = ['*']
```

Voir aussi :

`ALLOWED_HOSTS`, `WEBLATE_ALLOWED_HOSTS`, *Allowed hosts setup*

`SESSION_ENGINE`

Configure how your sessions will be stored. In case you keep the default database backend engine, you should schedule : `weblate clearsessions` to remove stale session data from the database.

If you are using Redis as cache (see *Activer la mise en cache*) it is recommended to use it for sessions as well :

```python
SESSION_ENGINE = 'django.contrib.sessions.backends.cache'
```

Voir aussi :

*Configuration du moteur de sessions*, `SESSION_ENGINE`

`DATABASES`

Connectivity to database server, please check Django’s documentation for more details.

Voir aussi :

*Database setup for Weblate*, `DATABASES`, *Bases de données*

`DEBUG`

Disable this for any production server. With debug mode enabled, Django will show backtraces in case of error to users, when you disable it, errors will be sent per e-mail to `ADMINS` (see above).

Debug mode also slows down Weblate, as Django stores much more info internally in this case.

Voir aussi :

`DEBUG`, *Disable debug mode*

`DEFAULT_FROM_EMAIL`

E-mail sender address for outgoing e-mail, for example registration e-mails.

Voir aussi :

`DEFAULT_FROM_EMAIL`

`SECRET_KEY`

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

Voir aussi :

`SECRET_KEY`

`SERVER_EMAIL`

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

Voir aussi :

`SERVER_EMAIL`
2.1.8 Filling up the database

After your configuration is ready, you can run `weblate migrate` to create the database structure. Now you should be able to create translation projects using the admin interface.

In case you want to run an installation non interactively, you can use `weblate migrate --noinput`, and then create an admin user using `createadmin` command.

Once you are done, you should also check the Performance report in the admin interface, which will give you hints of potential non optimal configuration on your site.

Voir aussi :

Configuration, Liste des privilèges et des rôles intégrés

2.1.9 Configuration de production

For a production setup you should carry out adjustments described in the following sections. The most critical settings will trigger a warning, which is indicated by an exclamation mark in the top bar if signed in as a superuser:

![Screenshot of Weblate dashboard](image)

Choose what languages you want in the preferences, to see overview of available translations for those languages in your watched projects.

It is also recommended to inspect checks triggered by Django (though you might not need to fix all of them):

```
weblate check --deploy
```

You can also review the very same checklist from the Interface de gestion.

Voir aussi :

Liste de contrôle de déploiement

**Disable debug mode**

Disable Django’s debug mode (`DEBUG`) by:

```
DEBUG = False
```

With debug mode on, Django stores all executed queries and shows users backtraces of errors, which is not desired in a production setup.

Voir aussi :

Ajuster la configuration
Properly configure admins

Set the correct admin addresses to the `ADMINS` setting to defining who will receive e-mails in case something goes wrong on the server, for example:

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

Voir aussi :

Ajuster la configuration

Set correct site domain

Adjust site name and domain in the admin interface, otherwise links in RSS or registration e-mails will not work. This is configured using `SITE_DOMAIN` which should contain site domain name.

Modifié dans la version 4.2 : Prior to the 4.2 release the Django sites framework was used instead, please see L'infrastructure des « sites ».

Voir aussi :

Allowed hosts setup, Correctly configure HTTPS SITE_DOMAIN, WEBLATE_SITE_DOMAIN, ENABLE_HTTPS

Correctly configure HTTPS

It is strongly recommended to run Weblate using the encrypted HTTPS protocol. After enabling it, you should set `ENABLE_HTTPS` in the settings:

```
ENABLE_HTTPS = True
```

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

Voir aussi :

ENABLE_HTTPS, Allowed hosts setup, Set correct site domain

Set properly `SECURE_HSTS_SECONDS`

If your site is served over SSL, you have to consider setting a value for `SECURE_HSTS_SECONDS` in the `settings.py` to enable HTTP Strict Transport Security. By default it's set to 0 as shown below.

```
SECURE_HSTS_SECONDS = 0
```

If set to a non-zero integer value, the `django.middleware.security.SecurityMiddleware` sets the Sécurité de transport HTTP stricte (HSTS) header on all responses that do not already have it.

Avertissement : Setting this incorrectly can irreversibly (for some time) break your site. Read the Sécurité de transport HTTP stricte (HSTS) documentation first.
Use a powerful database engine

— Please use PostgreSQL for a production environment, see Database setup for Weblate for more info.
— Use adjacent location for running the database server, otherwise the networking performance or reliability might ruin your Weblate experience.
— Check the database server performance or tweak its configuration, for example using PGTune.

Voir aussi :
Database setup for Weblate, Migrating from other databases to PostgreSQL, Ajuster la configuration, Bases de données

Activer la mise en cache

If possible, use Redis from Django by adjusting the CACHES configuration variable, for example:

```python
CACHES = {
    "default": {
        "BACKEND": "django_redis.cache.RedisCache",
        "LOCATION": "redis://127.0.0.1:6379/0",
        # If redis is running on same host as Weblate, you might want to use unix sockets instead:
        # 'LOCATION': 'unix:///var/run/redis/redis.sock?db=0',
        "OPTIONS": {
            "CLIENT_CLASS": "django_redis.client.DefaultClient",
            "PARSER_CLASS": "redis.connection.HiredisParser",
        },
    },
}
```

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

Voir aussi :
Cache Avatar, L’infrastructure de cache dans Django

Cache Avatar

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

```python
CACHES = {
    "default": {
        # Default caching backend setup, see above
        "BACKEND": "django_redis.cache.RedisCache",
        "LOCATION": "unix:///var/run/redis/redis.sock?db=0",
        "OPTIONS": {
            "CLIENT_CLASS": "django_redis.client.DefaultClient",
            "PARSER_CLASS": "redis.connection.HiredisParser",
        },
    },
    "avatar": {
        "BACKEND": "django.core.cache.backends.filebased.FileBasedCache",
        "LOCATION": os.path.join(DATA_DIR, "avatar-cache"),
        "TIMEOUT": 604800,
        "OPTIONS": {
            "MAX_ENTRIES": 1000,
        },
    },
}
```

(suite sur la page suivante)
Configure e-mail sending

Weblate needs to send out e-mails on several occasions, and these e-mails should have a correct sender address, please configure `SERVER_EMAIL` and `DEFAULT_FROM_EMAIL` to match your environment, for example:

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

**Note:** To disable sending e-mails by Weblate set `EMAIL_BACKEND` to `django.core.mail.backends.dummy.EmailBackend`.

This will disable all e-mail delivery including registration or password reset e-mails.

Allowed hosts setup

Django requires `ALLOWED_HOSTS` to hold a list of domain names your site is allowed to serve, leaving it empty will block any requests.

In case this is not configured to match your HTTP server, you will get errors like `Invalid HTTP_HOST header: '1.1.1.1'. You may need to add '1.1.1.1' to ALLOWED_HOSTS`.

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

Django secret key

The `SECRET_KEY` setting is used by Django to sign cookies, and you should really generate your own value rather than using the one from the example setup.

You can generate a new key using `weblate-generate-secret-key` shipped with Weblate.

**Voir aussi:**

`SECRET_KEY`
Running maintenance tasks

For optimal performance, it is good idea to run some maintenance tasks in the background. This is now automatically done by Background tasks using Celery and covers following tasks:

- Configuration health check (hourly).
- Committing pending changes (hourly), see Archivages lazy and commit_pending.
- Updating component alerts (daily).
- Update remote branches (nightly), see AUTO_UPDATE.
- Translation memory backup to JSON (daily), see dump_memory.
- Fulltext and database maintenance tasks (daily and weekly tasks), see cleanuptrans.

Modified in version 3.2: Since version 3.2, the default way of executing these tasks is using Celery and Weblate already comes with proper configuration, see Background tasks using Celery.

System locales and encoding

The system locales should be configured to UTF-8 capable ones. On most Linux distributions this is the default setting. In case it is not the case on your system, please change locales to UTF-8 variant.

For example by editing /etc/default/locale and setting there LANG="C.UTF-8".

In some cases the individual services have separate configuration for locales. This varies between distribution and web servers, so check documentation of your web server packages for that.

Apache on Ubuntu uses /etc/apache2/envvars:

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

Apache on CentOS uses /etc/sysconfig/httpd (or /opt/rh/httpd24/root/etc/sysconfig/httpd):

```
LANG='en_US.UTF-8'
```

Using custom certificate authority

Weblate does verify SSL certificates during HTTP requests. In case you are using custom certificate authority which is not trusted in default bundles, you will have to add its certificate as trusted.

The preferred approach is to do this at system level, please check your distro documentation for more details (for example on debian this can be done by placing the CA certificate into /usr/local/share/ca-certificates/ and running update-ca-certificates).

Once this is done, system tools will trust the certificate and this includes Git.

For Python code, you will need to configure requests to use system CA bundle instead of the one shipped with it. This can be achieved by placing following snippet to settings.py (the path is Debian specific):

```python
import os

os.environ["REQUESTS_CA_BUNDLE"] = "/etc/ssl/certs/ca-certificates.crt"
```
Compressing client assets

Weblate comes with a bunch of JavaScript and CSS files. For performance reasons it is good to compress them before sending to a client. In default configuration this is done on the fly at cost of little overhead. On big installations, it is recommended to enable offline compression mode. This needs to be done in the configuration and the compression has to be triggered on every Weblate upgrade.

The configuration switch is simple by enabling `django.conf.settings.COMPRESS_OFFLINE` and configuring `django.conf.settings.COMPRESS_OFFLINE_CONTEXT` (the latter is already included in the example configuration):

```python
COMPRESS_OFFLINE = True
```

On each deploy you need to compress the files to match current version:

```
weblate compress
```

**Indication**: The official Docker image has this feature already enabled.

**Voir aussi**:
- Common Deployment Scenarios
- Serving static files

### 2.1.10 Exécuter un serveur

**Indication**: In case you are not experienced with services described below, you might want to try [Installing using Docker](#).

You will need several services to run Weblate, the recommended setup consists of:

- Database server (see [Database setup for Weblate](#))
- Cache server (see [Activer la mise en cache](#))
- Frontend web server for static files and SSL termination (see [Serving static files](#))
- WSGI server for dynamic content (see [Sample configuration for NGINX and uWSGI](#))
- Celery for executing background tasks (see [Background tasks using Celery](#))

**Note**: There are some dependencies between the services, for example cache and database should be running when starting up Celery or uwsgi processes.

In most cases, you will run all services on single (virtual) server, but in case your installation is heavy loaded, you can split up the services. The only limitation on this is that Celery and Wsgi servers need access to `DATA_DIR`.

**Note**: The WSGI process has to be executed under the same user the Celery process, otherwise files in the `DATA_DIR` will be stored with mixed ownership, leading to runtime issues.

**Voir aussi** [Permissions du système de fichiers](#) et [Background tasks using Celery](#).
Running web server

Running Weblate is not different from running any other Django based program. Django is usually executed as uWSGI or fcgi (see examples for different web servers below).

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

```
weblate runserver
```

**Avertissement** : DO NOT USE THIS SERVER IN A PRODUCTION SETTING. It has not gone through security audits or performance tests. See also Django documentation on `runserver`.

**Indication** : The Django built-in server serves static files only with `DEBUG` enabled as it is intended for development only. For production use, please see wsgi setups in Sample configuration for NGINX and uWSGI, Sample configuration for Apache, Sample configuration for Apache and Gunicorn, and Serving static files.

Serving static files

Modifié dans la version 2.4 : Prior to version 2.4, Weblate didn’t properly use the Django static files framework and the setup was more complex.

Django needs to collect its static files in a single directory. To do so, execute `weblate collectstatic --noinput`. This will copy the static files into a directory specified by the `STATIC_ROOT` setting (this defaults to a static directory inside `DATA_DIR`).

It is recommended to serve static files directly from your web server, you should use that for the following paths:

```
/static/
```
- Serves static files for Weblate and the admin interface (from defined by `STATIC_ROOT`).

```
/media/
```
- Used for user media uploads (e.g. screenshots).

```
/favicon.ico
```
- Should be rewritten to rewrite a rule to serve `/static/favicon.ico`.

Voir aussi :

Sample configuration for NGINX and uWSGI, Sample configuration for Apache, Sample configuration for Apache and Gunicorn, Compressing client assets, Comment déployer Django, Déploiement des fichiers statiques

Content security policy

The default Weblate configuration enables `weblate.middleware.SecurityMiddleware` middleware which sets security related HTTP headers like `Content-Security-Policy` or `X-XSS-Protection`. These are by default set up to work with Weblate and its configuration, but this might need customization for your environment.

Voir aussi :

`CSP_SCRIPT_SRC`, `CSP_IMG_SRC`, `CSP_CONNECT_SRC`, `CSP_STYLE_SRC`, `CSP_FONT_SRC`
Sample configuration for NGINX and uWSGI

To run production webserver, use the wsgi wrapper installed with Weblate (in virtual env case it is installed as ~/weblate-env/lib/python3.9/site-packages/weblate/wsgi.py). Don’t forget to set the Python search path to your virtualenv as well (for example using virtualenv = /home/user/weblate-env in uWSGI).

The following configuration runs Weblate as uWSGI under the NGINX webserver.

Configuration for NGINX (also available as weblate/examples/weblate.nginx.conf):

```
# nginx configuration for Weblate
# You will want to change:
# - server_name
# - change /home/weblate/weblate-env to location where Weblate virtualenv is placed
# - change /home/weblate/data to match your DATA_DIR
# - change python3.9 to match your Python version
# - change weblate user to match your Weblate user

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

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

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

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

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

Configuration for uWSGI (also available as weblate/examples/weblate.uwsgi.ini):

```
# uWSGI configuration for Weblate
# You will want to change:
```

(suite sur la page suivante)
# - change /home/weblate/weblate-env to location where Weblate virtualenv is placed
# - change /home/weblate/data to match your DATA_DIR
# - change python3.9 to match your Python version
# - change weblate user to match your Weblate user
#
[uwsgi]
plugins   = python3
master    = true
protocol  = uwsgi
socket    = 127.0.0.1:8080
wsgi-file = /home/weblate/weblate-env/lib/python3.9/site-packages/weblate/wsgi.py

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

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

# Needed for OAuth/OpenID
buffer-size = 8192

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

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

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

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

# Avoid default 0000 umask
umask = 0022

# Run as weblate user
uid = weblate
gid = weblate

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

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

# Do not log some errors caused by client disconnects
ignore-sigpipe = true
ignore-write-errors = true
disable-write-exception = true

Voir aussi :

Déploiement de Django avec uWSGI
Sample configuration for Apache

It is recommended to use prefork MPM when using WSGI with Weblate.

The following configuration runs Weblate as WSGI, you need to have enabled `mod_wsgi` (available as `weblate/examples/apache.conf`):

```apache
# VirtualHost for Weblate
# You will want to change:
# - ServerAdmin and ServerName
# - change /home/weblate/weblate-env to location where Weblate virtualenv is placed
# - change /home/weblate/data to match your DATA_DIR
# - change python3.9 to match your Python version
# - change weblate user to match your Weblate user

<VirtualHost *:80>
  ServerAdmin admin@weblate.example.org
  ServerName weblate.example.org

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

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

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

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

  WSGIScriptAlias / /home/weblate/weblate-env/lib/python3.9/site-packages/weblate/wsgi.py process-group=weblate
  WSGIPassAuthorization On

  <Directory /home/weblate/weblate-env/lib/python3.9/site-packages/weblate/>
    <Files wsgi.py>
      Require all granted
    </Files>
  </Directory>
</VirtualHost>
```

**Note:** Weblate requires Python 3, so please make sure you are running Python 3 variant of the modwsgi. Usually it is available as a separate package, for example `libapache2-mod-wsgi-py3`.

**Voir aussi:**

*System locales and encoding, Django avec Apache et mod_wsgi*
Sample configuration for Apache and Gunicorn

The following configuration runs Weblate in Gunicorn and Apache 2.4 (available as weblate/examples/apache.gunicorn.conf):

```apache
# # VirtualHost for Weblate using gunicorn on localhost:8000
# # You will want to change:
# #  - ServerAdmin and ServerName
#  - change /home/weblate/weblate-env to location where Weblate virtualenv is placed
#  - change /home/weblate/data to match your DATA_DIR
#  - change python3.9 to match your Python version
#  - change weblate user to match your Weblate user
#
<VirtualHost *:443>
  ServerAdmin admin@weblate.example.org
  ServerName weblate.example.org

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

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

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

  SSLEngine on
  SSLCertificateFile /etc/apache2/ssl/https_cert.cert
  SSLCertificateKeyFile /etc/apache2/ssl/https_key.pem
  SSLProxyEngine On

  ProxyPass /favicon.ico !
  ProxyPass /static/ !
  ProxyPass /media/ !

  ProxyPass / http://localhost:8000/
  ProxyPassReverse / http://localhost:8000/
  ProxyPreserveHost On
</VirtualHost>
```

Voir aussi :

Déploiement de Django avec Gunicorn

2.1. Instructions de configuration
Running Weblate under path

Nouveaux dans la version 1.3.

It is recommended to use prefork MPM when using WSGI with Weblate.

A sample Apache configuration to serve Weblate under /weblate. Again using mod_wsgi (also available as weblate/examples/apache-path.conf):

```conf
# # VirtualHost for Weblate, running under /weblate path
# # You will want to change:
# # - ServerAdmin and ServerName
# - change /home/weblate/weblate-env to location where Weblate virtualenv is placed
# - change /home/weblate/data to match your DATA_DIR
# - change python3.9 to match your Python version
# - change weblate user to match your Weblate user
#
<VirtualHost *:80>
  ServerAdmin admin@weblate.example.org
  ServerName weblate.example.org

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

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

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

  # Path to your Weblate virtualenv
  WSGIDaemonProcess weblate python-home=/home/weblate/weblate-env user=weblate...
  request-timeout=600
  WSGIPassAuthorization On
</VirtualHost>
```

Additionally, you will have to adjust weblate/settings.py:

```python
URL_PREFIX = "\weblate"
```
2.1.11 Background tasks using Celery

Nouveau dans la version 3.2.

Weblate uses Celery to execute regular and background tasks. You are supposed to run a Celery service that will execute these. For example, it is responsible for handling following operations (this list is not complete):

- Receiving webhooks from external services (see Déclencheurs de notification).
- Running regular maintenance tasks such as backups, cleanups, daily add-ons, or updates (see Sauvegarder et déplacer Weblate, BACKGROUND_TASKS, Extensions).
- Exécution de Traduction automatique.
- Envoi de notifications condensées.
- Offloading expensive operations from the wsgi process.
- Validation des modifications en attente (voir Archivages lazy).

A typical setup using Redis as a backend looks like this:

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

Voir aussi :

Configuration de Redis broker dans Celery

You should also start the Celery worker to process the tasks and start scheduled tasks, this can be done directly on the command-line (which is mostly useful when debugging or developing):

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

Note : The Celery process has to be executed under the same user as the WSGI process, otherwise files in the DATA_DIR will be stored with mixed ownership, leading to runtime issues.

Voir aussi Permissions du système de fichiers et Exécuter un serveur.

Executing Celery tasks in the wsgi using eager mode

Note: This will have severe performance impact on the web interface, and will break features depending on regular trigger (for example committing pending changes, digest notifications, or backups).

For development, you might want to use eager configuration, which does process all tasks in place:

```bash
CELERY_TASK_ALWAYS_EAGER = True
CELERY_BROKER_URL = "memory://"
CELERY_TASK_EAGER_PROPAGATES = True
```

Running Celery as system service

Most likely you will want to run Celery as a daemon and that is covered by Daemonization. For the most common Linux setup using systemd, you can use the example files shipped in the examples folder listed below.

Systemd unit to be placed as /etc/systemd/system/celery-weblate.service:

```ini
[Unit]
Description=Celery Service (Weblate)
After=network.target
```

(suite sur la page suivante)
**[Service]**
Type=forking
User=weblate
Group=weblate
EnvironmentFile=/etc/default/celery-weblate
WorkingDirectory=/home/weblate
RuntimeDirectory=celsius
EnvironmentFilePreserve=restart
LogDirectory=celsius

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

Environment configuration to be placed as /etc/default/celery-weblate:

```
# Name of nodes to start
CELERYD_NODES="celery notify memory backup translate"

# Absolute or relative path to the 'celery' command:
CELERY_BIN="/home/weblate/weblate-env/bin/celery"

# App instance to use
# comment out this line if you don't use an app
CELERY_APP="weblate.utils"

# Extra command-line arguments to the worker,
# increase concurrency if you get weblate.E019
CELERYD_OPTS="--beat=celery --queues:celery=celery --prefetch-multiplier:celery=4 \

# Logging configuration
# - %n will be replaced with the first part of the nodename.
# - %I will be replaced with the current child process index
# and is important when using the prefork pool to avoid race conditions.
CELERYD_PID_FILE="/run/celery/weblate-%n.pid"
CELERYD_LOG_FILE="/var/log/celery/weblate-%n%I.log"
CELERYD_LOG_LEVEL="INFO"
```

Additional configuration to rotate Celery logs using **logrotate** to be placed as /etc/logrotate.d/celery:

```
/var/log/celery/*.log {
  weekly
  missingok
  rotate 12
 compress
  notifempty
}
```
Periodic tasks using Celery beat

Weblate comes with built-in setup for scheduled tasks. You can however define additional tasks in `settings.py`, for example see Archivages lazy.

The tasks are supposed to be executed by Celery beats daemon. In case it is not working properly, it might not be running or its database was corrupted. Check the Celery startup logs in such case to figure out root cause.

Monitoring Celery status

You can find current length of the Celery task queues in the Interface de gestion or you can use celery_queues on the command-line. In case the queue will get too long, you will also get configuration error in the admin interface.

**Avertissement :** The Celery errors are by default only logged into Celery log and are not visible to user. In case you want to have overview on such failures, it is recommended to configure Collecting error reports.

Voir aussi :

Surveiller Weblate, How can I check whether my Weblate is set up properly?, Configuration and defaults, Workers Guide, Daemonization, Monitoring and Management Guide, celery_queues

2.1.12 Surveiller Weblate

Weblate provides the /healthz/ URL to be used in simple health checks, for example using Kubernetes. The Docker container has built-in health check using this URL.

For monitoring metrics of Weblate you can use GET /api/metrics/ API endpoint.

Voir aussi :

How can I check whether my Weblate is set up properly?, Monitoring Celery status, Weblate plugin for Munin

2.1.13 Collecting error reports

Weblate, as any other software, can fail. In order to collect useful failure states we recommend to use third party services to collect such information. This is especially useful in case of failing Celery tasks, which would otherwise only report error to the logs and you won’t get notified on them. Weblate has support for the following services :

**Sentry**

Weblate has built-in support for Sentry. To use it, it’s enough to set `SENTRY_DSN` in the `settings.py`:

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

**Rollbar**

Weblate has built-in support for Rollbar. To use it, it’s enough to follow instructions for Rollbar notifier for Python. In short, you need to adjust `settings.py`:

```
# Add rollbar as last middleware:
MIDDLEWARE = [
    # ... other middleware classes ...  
    "rollbar.contrib.django.middleware.RollbarNotifierMiddleware",
]
```

(suite sur la page suivante)
# Configure client access

```json
ROLLBAR = {
    "access_token": "POST_SERVER_ITEM_ACCESS_TOKEN",
    "client_token": "POST_CLIENT_ITEM_ACCESS_TOKEN",
    "environment": "development" if DEBUG else "production",
    "branch": "main",
    "root": "/absolute/path/to/code/root",
}
```

Everything else is integrated automatically, you will now collect both server and client side errors.

## 2.1.14 Migrating Weblate to another server

Migrating Weblate to another server should be pretty easy, however it stores data in few locations which you should migrate carefully. The best approach is to stop Weblate for the migration.

### Migrer la base de données

Depending on your database backend, you might have several options to migrate the database. The most straightforward approach is to use database native tools, as they are usually the most effective (e.g. `mysqldump` or `pg_dump`). Alternatively you can use replication in case your database supports it.

**Voir aussi :**

Migrating between databases described in *Migrating from other databases to PostgreSQL*.

### Migrating VCS repositories

The VCS repositories stored under `DATA_DIR` need to be migrated as well. You can simply copy them or use `rsync` to do the migration more effectively.

### Autres notes

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

## 2.2 Déploiements Weblate

Weblate can be easily installed in your cloud. Please find detailed guide for your platform :

- *Installing using Docker*
- *Installing on OpenShift*
- *Installing on Kubernetes*
2.2.1 Third-party deployments for Weblate

**Note:** Following deployments are not developed or supported by Weblate team. Parts of the setup might vary from what is described in this documentation.

**Bitnami Weblate stack**

Bitnami provides a Weblate stack for many platforms at <https://bitnami.com/stack/weblate>.

**Voir aussi:**

Weblate packaged by Bitnami

**Paquet Cloudron Weblate**

Cloudron is a platform for self-hosting web applications. Weblate installed with Cloudron will be automatically kept up-to-date. The package is maintained by the Cloudron team at their Weblate package repo.

![Cloudron install](image)

**Weblate dans YunoHost**

The self-hosting project YunoHost provides a package for Weblate. Once you have your YunoHost installation, you may install Weblate as any other application. It will provide you with a fully working stack with backup and restoration, but you may still have to edit your settings file for specific usages.

You may use your administration interface, or this button (it will bring you to your server):

![Install with YunoHost](image)

It also is possible to use the command-line interface:

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

2.3 Mise à niveau de Weblate

2.3.1 Docker image upgrades

The official Docker image (see *Installing using Docker*) has all Weblate upgrade steps integrated. There are typically no manual steps needed besides pulling latest version.

**Voir aussi:**

*Upgrading the Docker container*
2.3.2 Generic upgrade instructions

Before upgrading, please check the current *Exigences logicielles* as they might have changed. Once all requirements are installed or updated, please adjust your `settings.py` to match changes in the configuration (consult `settings_example.py` for correct values).

Always check *Version specific instructions* before upgrade. In case you are skipping some versions, please follow instructions for all versions you are skipping in the upgrade. Sometimes it’s better to upgrade to some intermediate version to ensure a smooth migration. Upgrading across multiple releases should work, but is not as well tested as single version upgrades.

**Note**: It is recommended to perform a full database backup prior to upgrade so that you can roll back the database in case upgrade fails, see *Sauvegarder et déplacer Weblate*.

1. Stop wsgi and Celery processes. The upgrade can perform incompatible changes in the database, so it is always safer to avoid old processes running while upgrading.

2. Upgrade Weblate code.
   For pip installs it can be achieved by:
   ```bash
   pip install -U "Weblate[all]==version"
   ```
   Or, if you just want to get the latest released version:
   ```bash
   pip install -U "Weblate[all]"
   ```
   If you don’t want to install all of the optional dependencies do:
   ```bash
   pip install -U Weblate
   ```
   With Git checkout you need to fetch new source code and update your installation:
   ```bash
   cd weblate-src
git pull
   # Update Weblate inside your virtualenv
   .~/weblate-env/bin/pip install -e .
   # Install dependencies directly when not using virtualenv
   pip install --upgrade -r requirements.txt
   # Install optional dependencies directly when not using virtualenv
   pip install --upgrade -r requirements-optional.txt
   ```

3. New Weblate release might have new *Optional dependencies*, please check if they cover features you want.

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

5. Upgrade database structure:
   ```bash
   weblate migrate --noinput
   ```

6. Collect updated static files (see *Exécuter un serveur* and *Serving static files*):
   ```bash
   weblate collectstatic --noinput --clear
   ```

7. Compress JavaScript and CSS files (optional, see *Compressing client assets*):
   ```bash
   weblate compress
   ```

8. If you are running version from Git, you should also regenerate locale files every time you are upgrading. You can do this by invoking:
   ```bash
   weblate compilemessages
   ```

9. Verify that your setup is sane (see also *Configuration de production*):

214 Chapitre 2. Documentation pour l’administrateur
10. Restart Celery worker (see *Background tasks using Celery*).

### 2.3.3 Version specific instructions

#### Upgrade from 2.x

If you are upgrading from 2.x release, always first upgrade to 3.0.1 and then continue upgrading in the 3.x series. Upgrades skipping this step are not supported and will break.

**Voir aussi :**

Upgrade from 2.20 to 3.0 in Weblate 3.0 documentation

#### Upgrade from 3.x

If you are upgrading from 3.x release, always first upgrade to 4.0.4 or 4.1.1 and then continue upgrading in the 4.x series. Upgrades skipping this step are not supported and will break.

**Voir aussi :**

Upgrade from 3.11 to 4.0 in Weblate 4.0 documentation

#### Upgrade from 4.0 to 4.1

Please follow *Generic upgrade instructions* in order to perform update.

Notable configuration or dependencies changes:
- There are several changes in *settings_example.py*, most notable middleware changes, please adjust your settings accordingly.
- There are new file formats, you might want to include them in case you modified the `WEBLATE_FORMATS`.
- There are new quality checks, you might want to include them in case you modified the `CHECK_LIST`.
- There is change in `DEFAULT_THROTTLE_CLASSES` setting to allow reporting of rate limiting in the API.
- There are some new and updated requirements.
- There is a change in `INSTALLED_APPS`.
- The `MT_DEEPL_API_VERSION` setting has been removed in Version 4.7. The DeepL machine translation now uses the new `MT_DEEPL_API_URL` instead. You might need to adjust `MT_DEEPL_API_URL` to match your subscription.

**Voir aussi :**

*Generic upgrade instructions*

#### Upgrade from 4.1 to 4.2

Please follow *Generic upgrade instructions* in order to perform update.

Notable configuration or dependencies changes:
- Upgrade from 3.x releases is not longer supported, please upgrade to 4.0 or 4.1 first.
- There are some new and updated requirements.
- There are several changes in *settings_example.py*, most notable new middleware and changed application ordering.
- The keys for JSON based formats no longer include leading dot. The strings are adjusted during the database migration, but external components might need adjustment in case you rely on keys in exports or API.
- The Celery configuration was changed to no longer use memory queue. Please adjust your startup scripts and `CELERY_TASK_ROUTES` setting.
- The Weblate domain is now configured in the settings, see `SITE_DOMAIN` (or `WEBLATE_SITE_DOMAIN`). You will have to configure it before running Weblate.
— The username and email fields on user database now should be case insensitive unique. It was mistakenly not enforced with PostgreSQL.

**Voir aussi**:

*Generic upgrade instructions*

### Upgrade from 4.2 to 4.3

Please follow *Generic upgrade instructions* in order to perform update.

Notable configuration or dependencies changes:

— There are some changes in quality checks, you might want to include them in case you modified the `CHECK_LIST`.

— The source language attribute was moved from project to a component what is exposed in the API. You will need to update `Client Weblate` in case you are using it.

— The database migration to 4.3 might take long depending on number of strings you are translating (expect around one hour of migration time per 100,000 source strings).

— There is a change in `INSTALLED_APPS`.

— There is a new setting `SESSION_COOKIE_AGE_AUTHENTICATED` which complements `SESSION_COOKIE_AGE`.

— In case you were using `hub` or `lab` to integrate with GitHub or GitLab, you will need to reconfigure this, see `GITHUB_CREDENTIALS` and `GITLAB_CREDENTIALS`.

**Modifié dans la version 4.3.1**:

— The Celery configuration was changed to add `memory` queue. Please adjust your startup scripts and `CELERY_TASK_ROUTES` setting.

**Modifié dans la version 4.3.2**:

— The `post_update` method of add-ons now takes extra `skip_push` parameter.

**Voir aussi**:

*Generic upgrade instructions*

### Upgrade from 4.3 to 4.4

Please follow *Generic upgrade instructions* in order to perform update.

Notable configuration or dependencies changes:

— There is a change in `INSTALLED_APPS`, `weblate.configuration` has to be added there.

— Django 3.1 is now required.

— In case you are using MySQL or MariaDB, the minimal required versions have increased, see *MySQL and MariaDB*.

**Modifié dans la version 4.4.1**:

— Gettext monolingue now uses both `msgid` and `msgctxt` when present. This will change identification of translation strings in such files breaking links to Weblate extended data such as screenshots or review states. Please make sure you commit pending changes in such files prior upgrading and it is recommended to force loading of affected component using `loadpo`.

— Increased minimal required version of translate-toolkit to address several file format issues.

**Voir aussi**:

*Generic upgrade instructions*
Upgrade from 4.4 to 4.5

Please follow Generic upgrade instructions in order to perform update.

Notable configuration or dependencies changes:
- The migration might take considerable time if you had big glossaries.
- Glossaries are now stored as regular components.
- The glossary API is removed, use regular translation API to access glossaries.
- There is a change in INSTALLED_APPS: `weblate.metrics` should be added.

Modifié dans la version 4.5.1:
- There is a new dependency on the `pyahocorasick` module.

Voir aussi:
Generic upgrade instructions

Upgrade from 4.5 to 4.6

Please follow Generic upgrade instructions in order to perform update.

Notable configuration or dependencies changes:
- There are new file formats, you might want to include them in case you modified the `WEBLATE_FORMATS`.
- API for creating components now automatically uses `URLs internes de Weblate`, see `POST /api/projects/(string:project)/components/`.
- There is a change in dependencies and `PASSWORD_HASHERS` to prefer Argon2 for passwords hashing.

Voir aussi:
Generic upgrade instructions

Upgrade from 4.6 to 4.7

Please follow Generic upgrade instructions in order to perform update.

Notable configuration or dependencies changes:
- There are several changes in `settings_example.py`, most notable middleware changes (`MIDDLEWARE`), please adjust your settings accordingly.
- The `DeepL` machine translation now has a generic `MT_DEEPL_API_URL` setting to adapt to different subscription models more flexibly. The `MT_DEEPL_API_VERSION` setting is no longer used.
- Django 3.2 is now required.

Voir aussi:
Generic upgrade instructions

Upgrade from 4.7 to 4.8

Please follow Generic upgrade instructions in order to perform update.

There are no additional upgrade steps needed in this release.

Voir aussi:
Generic upgrade instructions
Upgrade from 4.8 to 4.9

Please follow Generic upgrade instructions in order to perform update.

— There is a change in storing metrics, the upgrade can take long time on larger sites.

Voir aussi :

Generic upgrade instructions

Upgrade from 4.9 to 4.10

Please follow Generic upgrade instructions in order to perform update.

— There is a change in per-project groups, the upgrade can take long time on sites with thousands of projects.
— Django 4.0 has made some incompatible changes, see Changements incompatibles avec les anciennes versions dans Django 4.0. Weblate still supports Django 3.2 for now, in case any of these are problematic. Most notable changes which might affect Weblate :
  — Dropped support for PostgreSQL 9.6, Django 4.0 supports PostgreSQL 10 and higher.
  — Format of CSRF_TRUSTED_ORIGINS was changed.
  — The Docker container now uses Django 4.0, see above for changes.

Voir aussi :

Generic upgrade instructions

Mise à niveau de la version 4.10 vers la version 4.11

Please follow Generic upgrade instructions in order to perform update.

— Weblate now requires Python 3.7 or newer.
— The implementation of Gestion du contrôle d’accès par projet has changed, removing the project prefix from the group names. This affects API users.
— Weblate now uses charset-normalizer instead of chardet module for character set detection.
— Changed in 4.11.1 : There is a change in REST_FRAMEWORK setting (removal of one of the backends in DEFAULT_AUTHENTICATION_CLASSES).

Voir aussi :

Generic upgrade instructions

Upgrade from 4.11 to 4.12

Please follow Generic upgrade instructions in order to perform update.

— There are no special steps required.

Voir aussi :

Generic upgrade instructions

Upgrade from 4.12 to 4.13

Please follow Generic upgrade instructions in order to perform update.

— The Définitions de langue are now automatically updated on upgrade, use UPDATE_LANGUAGES to disable that.
— Handling of context and location has been changed for Windows RC files, HTML files, IDML Format, and Fichiers texte file formats. In most cases the context is now shown as location.
— The machine translation services are now configured using the user interface, settings from the configuration file will be imported during the database migration.

Voir aussi :

Generic upgrade instructions
Upgrade from 4.13 to 4.14

Please follow Generic upgrade instructions in order to perform update.

— The Java formatting checks now match GNU gettext flags. The flags set in Weblate will be automatically migrated, but third-party scripts will need to use java-printf-format instead of java-format and java-format instead of java-messageformat.
— The jellyfish dependency has been replaced by rapidfuzz.
— Changed in 4.14.2: Deprecated insecure configuration of VCS service API keys via _TOKEN/_USERNAME configuration instead of _CREDENTIALS list. In Docker, please add matching _HOST directive. For example see WEBLATE_GITHUB_HOST and GITHUB_CREDENTIALS.

Voir aussi:
Generic upgrade instructions

Upgrade from 4.14 to 4.15

Please follow Generic upgrade instructions in order to perform update.

— Weblate now requires btree_gin extension in PostgreSQL. The migration process will install it if it has sufficient privileges. See Creating a database in PostgreSQL for manual setup.
— The Docker image no longer enables debug mode by default. In case you want it, enable it in the environment using WEBLATE_DEBUG.
— The database migration make take hours on larger instances due to recreating some of the indexes.
— Changed in 4.15.1: The default value for DEFAULT_PAGINATION_CLASS in rest framework settings was changed.

Voir aussi:
Generic upgrade instructions

Upgrade from 4.15 to 4.16

Please follow Generic upgrade instructions in order to perform update.

— Celery beat is now storing the tasks schedule in the database. CELERY_BEAT_SCHEDULER and INSTALLED_APPS need to be changed for that.
— The deprecated VCS setting for credentials is no longer supported, see Upgrade from 4.13 to 4.14.
— Upgrade of django-crispy-forms requires changes in INSTALLED_APPS.
— Integration of django-cors-headers requires changes in INSTALLED_APPS and MIDDLEWARE.

Voir aussi:
Generic upgrade instructions

2.3.4 Upgrading from Python 2 to Python 3

Weblate no longer supports Python older than 3.6. In case you are still running on older version, please perform migration to Python 3 first on existing version and upgrade later. See Upgrading from Python 2 to Python 3 in the Weblate 3.11.1 documentation.
2.3.5 Migrating from other databases to PostgreSQL

If you are running Weblate on other database than PostgreSQL, you should consider migrating to PostgreSQL as Weblate performs best with it. The following steps will guide you in migrating your data between the databases. Please remember to stop both web and Celery servers prior to the migration, otherwise you might end up with inconsistent data.

Creating a database in PostgreSQL

It is usually a good idea to run Weblate in a separate database, and separate user account:

```
# If PostgreSQL was not installed before, set the main password
sudo -u postgres psql postgres -c "\password postgres"

# Create a database user called "weblate"
sudo -u postgres createdb -D -P weblate

# Create the database "weblate" owned by "weblate"
sudo -u postgres createdb -E UTF8 -O weblate weblate
```

Migrating using Django JSON dumps

The simplest approach for migration is to utilize Django JSON dumps. This works well for smaller installations. On bigger sites you might want to use pgloader instead, see Migrating to PostgreSQL using pgloader.

1. Add PostgreSQL as additional database connection to the settings.py:

```
DATABASES = {
    "default": {
        # Database engine
        "ENGINE": "django.db.backends.mysql",
        # Database name
        "NAME": "weblate",
        # Database user
        "USER": "weblate",
        # Database password
        "PASSWORD": "password",
        # Set to empty string for localhost
        "HOST": "database.example.com",
        # Set to empty string for default
        "PORT": "",
        # Additional database options
        "OPTIONS": {
            # In case of using an older MySQL server, which has MyISAM as a default storage
            # 'init_command': 'SET storage_engine=INNODB',
            # Uncomment for MySQL older than 5.7:
            # 'init_command': "SET sql_mode='STRICT_TRANS_TABLES'",
            # If your server supports it, see the Unicode issues above
            # 'charset': "utf8mb4",
            # Change connection timeout in case you get MySQL gone away error:
            # 'connect_timeout': 28800,
        },
    },
    "postgresql": {
        # Database engine
        "ENGINE": "django.db.backends.postgresql",
        # Database name
        "NAME": "weblate",
    },
}
```

(suite sur la page suivante)
# Database user
"USER": "weblate",
# Database password
"PASSWORD": "password",
# Set to empty string for localhost
"HOST": "database.example.com",
# Set to empty string for default
"PORT": ",";
}

2. Run migrations and drop any data inserted into the tables:

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

3. Dump legacy database and import to PostgreSQL

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

4. Adjust DATABASES to use just PostgreSQL database as default, remove legacy connection.

Weblate should be now ready to run from the PostgreSQL database.

## Migrating to PostgreSQL using pgloader

The pgloader is a generic migration tool to migrate data to PostgreSQL. You can use it to migrate Weblate database.

1. Adjust your settings.py to use PostgreSQL as a database.

2. Migrate the schema in the PostgreSQL database:

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

3. Run the pgloader to transfer the data. The following script can be used to migrate the database, but you might want to learn more about pgloader to understand what it does and tweak it to match your setup:

```
LOAD DATABASE FROM mysql://weblate:password@localhost/weblate INTO postgresql://weblate:password@localhost/weblate
WITH include no drop, truncate, create no tables, create no indexes, no
→foreign keys, disable triggers, reset sequences, data only
ALTER SCHEMA 'weblate' RENAME TO 'public'
```

2.3. Mise à niveau de Weblate
2.3.6 Migrating from Pootle

As Weblate was originally written as replacement from Pootle, it is supported to migrate user accounts from Pootle. You can dump the users from Pootle and import them using `importusers`.

2.4 Sauvegarder et déplacer Weblate

2.4.1 Project level backups


**Avertissement** : Restoring backups is only supported when using PostgreSQL or MariaDB 10.5+ as a database.

The project backups all translation content from Weblate (project, components, translations, string comments, suggestions or checks). It is suitable for transferring a project to another Weblate instance.

You can perform a project backup in Manage → Backups. The backup can be restored when creating a project (see Adding translation projects and components).

The backups currently do not include access control information and history.

The comments and suggestions are backed up with an username of user who did create them. Upon import it is assigned to a matching user. If there is no user with such username, it is assigned to anonymous user.

The generated backups are kept on the server as configured by `PROJECT_BACKUP_KEEP_DAYS` and `PROJECT_BACKUP_KEEP_COUNT` (it defaults to keep at most 3 backups for 30 days).

2.4.2 Sauvegarde automatique avec BorgBackup

Nouveau dans la version 3.9.

Weblate est conçu pour supporter la création de services de sauvegarde avec BorgBackup. Borg crée des sauvegardes chiffrées pouvant être stockées en sécurité dans le cloud. Ces sauvegardes peuvent être contrôlées depuis l’onglet Backups de l'interface de gestion.

Modifié dans la version 4.4.1 : Les bases de données PostgreSQL et MySQL/MariaDB sont incluses dans les sauvegardes automatisées.

Les sauvegardes avec Borg sont incrémentales et Weblate est configuré pour conserver les sauvegardes suivantes :

- Sauvegardes quotidiennes des 14 derniers jours
- Sauvegardes hebdomadaires sur 8 semaines
- Sauvegardes mensuelles des 6 derniers mois
2.4. Sauvegarder et déplacer Weblate
Clé de chiffrement Borg

BorgBackup crée des sauvegardes chiffrées et, sans phrase de passe, vous ne serez pas en mesure de les restaurer. La phrase de passe est générée à l’ajout d’un nouveau service de sauvegarde. Vous devriez la copier et la conserver en lieu sûr.

Si jamais vous utilisez Espace de sauvegarde provisionné par Weblate, merci de sauvegarder aussi votre clé SSH privée – elle est utilisée pour accéder à vos sauvegardes.

Voir aussi :
borg init

Personnalisation de la sauvegarde

— La sauvegarde de la base de données peut être configurée avec DEFAULT_ACCESS_CONTROL.
— La création de la sauvegarde peut être personnalisée en utilisant BORG_EXTRA_ARGS.

2.4.3 Espace de sauvegarde provisionné par Weblate

La manière la plus simple de sauvegarder votre instance Weblate est de s’offrir le service de sauvegarde de weblate.org.
La procédure d’activation peut être effectuée en quelques étapes :
2. Entrez la clé obtenue dans l’interface de gestion, voir Intégration de l’assistance.
3. Weblate se connecte au service cloud pour obtenir les informations d’accès pour les sauvegardes.

Indication : L’étape manuelle d’activation est nécessaire pour votre sécurité. Sans votre consentement, aucune donnée n’est envoyée au dépôt de sauvegarde obtenu par la procédure d’inscription.

2.4.4 Utiliser un espace de stockage personnalisé

Vous pouvez aussi utiliser votre propre espace de stockage pour vos sauvegardes. SSH peut être utilisé pour stocker les sauvegardes à distance ; dans ce cas, BorgBackup doit être installé sur le serveur cible.

Voir aussi :
General dans la documentation de Borg

Système de fichiers local

Il est recommandé de spécifier le chemin absolu de la sauvegarde locale, par exemple /chemin/vers/la/sauvegarde. L’utilisateur qui fait tourner Weblate doit avoir les permissions d’écriture sur le répertoire. Voir Permissions du système de fichiers. Si le répertoire n’existe pas, Weblate essayera de le créer, mais devra disposer des permissions nécessaires pour ce faire.

Indication : When running Weblate in Docker, please ensure the backup location is exposed as a volume from the Weblate container. Otherwise the backups will be discarded by Docker upon restarting the container it is in.

One option is to place backups into an existing volume, for example /app/data/borgbackup. This is an existing volume in the container.

Vous pouvez aussi ajouter un nouveau conteneur pour les sauvegardes dans le fichier Docker Compose et utiliser par exemple /borgbackup:
services:
  weblate:
    volumes:
    - /home/weblate/data:/app/data
    - /home/weblate/borgbackup:/borgbackup

Le répertoire où seront enregistrées les sauvegardes doit avoir pour propriétaire UID 1000, sinon Weblate ne sera pas en mesure d’y écrire les sauvegardes.

Sauvegardes à distance

Pour créer des sauvegardes distantes, vous devrez installer BorgBackup sur un autre serveur accessible par votre déploiement Weblate via SSH en utilisant la clé SSH de Weblate :

1. Préparez un serveur où seront stockées vos sauvegardes.
2. Installez le serveur SSH dessus (vous l’auriez par défaut avec la plupart des distributions Linux).
3. Installez BorgBackup sur ce serveur ; la plupart des distributions Linux disposent des paquets nécessaires (voir Installation).
5. Ajoutez la clé SSH de Weblate à l’utilisateur pour que Weblate puisse se connecter en SSH au serveur sans mot de passe (voir Weblate SSH key).
6. Configure the backup location in Weblate as user@host:/path/to/backups or ssh://

   user@host:port/path/to/backups.

Indication : Espace de sauvegarde provisionné par Weblate offre des sauvegardes distantes automatiques sans effort.

Voir aussi :

Weblate SSH key, General

2.4.5 Restaurer une sauvegarde depuis BorgBackup

1. Restaurer l’accès au répertoire de sauvegarde et préparer la phrase de passe.
2. Lister toutes les sauvegardes existantes sur le serveur avec borg list REPOSITORY.
3. Restore the desired backup to the current directory using borg extract REPOSITORY::ARCHIVE.
4. Restaurer la base de données depuis un dump SQL place dans le répertoire backup dans le répertoire de données de Weblate. Voir Données supprimées pour les sauvegardes.
5. Copy the Weblate configuration (backups/settings.py, see Données supprimées pour les sauvegardes) to the correct location, see Ajuster la configuration.
   Lorsque vous utilisez un conteneur Docker, le fichier de paramètres est déjà inclus dans le conteneur et vous devez restaurer les variables d'environnement d'origine. Le fichier environment.yml peut vous y aider (voir Données supprimées pour les sauvegardes).
6. Copy the whole restored data dir to the location configured by DATA_DIR.
   Lorsque vous utilisez le conteneur Docker, placez les données dans le volume de données, voir Docker container volumes.
   Assurez-vous que le propriétaire et les permissions des fichiers sont corrects, voir Permissions du système de fichiers.

The Borg session might look like this :
2.4.6 Sauvegarde manuelle

Depending on what you want to save, backup the type of data Weblate stores in each respective place.

**Indication :** 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 `WEB\_BLATE\_SILENCED_SYSTEM_CHECKS` for Docker.

```shell
SILENCED_SYSTEM_CHECKS.append("weblate.I028")
```

Base de données

La destination de stockage utilisée dépend de la configuration de votre base de données.

**Indication :** The database is the most important storage. Set up regular backups of your database. Without the database, all the translations are gone.

Sauvegarde native de base de données

The recommended approach is to save a dump of the database using database-native tools such as `pg_dump` or `mysqldump`. It usually performs better than Django backup, and it restores complete tables with all their data.

You can restore this backup in a newer Weblate release, it will perform all the necessary migrations when running `migrate`. Please consult [*Mise à niveau de Weblate*](https://docs.weblate.org/fr/latest/admin/migrate.html) on more detailed info on how to upgrade between versions.

Sauvegarde de base de données 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*](https://docs.weblate.org/fr/latest/admin/management-shell.html)):

```python
weblate shell
>>> from weblate.auth.models import User
>>> User.objects.get(username='anonymous').delete()
```
Fichiers

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.

Données supprimées pour les sauvegardes

Modifié dans la version 4.7: The environment dump was added as `environment.yml` to help in restoring in the Docker environments.

Stocké dans `DATA_DIR/backups`.

Weblate dépose diverses données ici, et vous pouvez inclure ces fichiers pour des sauvegardes plus complètes. Les fichiers sont mis à jour quotidiennement (nécessite un serveur Celery beats fonctionnel, voir Background tasks using Celery). Actuellement, ceci inclut:

— Les paramètres Weblate comme `settings.py` (il existe aussi une version étendue dans `settings-expanded.py`).
— La sauvegarde de la base de données PostgreSQL comme `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
```

Dépôts des contrôles de version

Stockés dans `DATA_DIR/vcs`.

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

Clés SSH et PGP

Stocké dans `DATA_DIR/ssh` et `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.

Fichiers téléversés par les utilisateurs

Stocké dans `DATA_DIR/media`.

You should back up all user uploaded files (e.g. Visual context for strings).
Tâches Celery

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

Voir aussi :

*Background tasks using Celery*

Sauvegarde manuelle en ligne de commande

Using a cron job, you can set up a Bash command to be executed on a daily basis, for example:

```
$ XZ_OPT="-9" tar -Jcf ~/backup/weblate-backup-$\{date -u +"%Y-%m-%d_%H%M%S\}\.xz ...
```

The string between the quotes after `XZ_OPT` allows you to choose your xz options, for instance the amount of memory used for compression; see https://linux.die.net/man/1/xz

You can adjust the list of folders and files to your needs. To avoid saving the translation memory (in backups folder), you can use:

```
$ XZ_OPT="-9" tar -Jcf ~/backup/weblate-backup-$\{date -u +"%Y-%m-%d_%H%M%S\}\.xz ...
```

2.4.7 Restaurer une sauvegarde manuelle

1. Restaurer toutes les données que vous avez sauvegardées.
2. Mettre à jour tous les répertoires en utilisant `updategit`.

```
weblate updategit --all
```

2.4.8 Déplacer une installation Weblate

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

Voir aussi :

*Upgrading from Python 2 to Python 3, Migrating from other databases to PostgreSQL*

2.5 Authentification

2.5.1 Enregistrement utilisateur

La configuration par défaut est d’utiliser python-social-auth, un formulaire web pour gérer les inscriptions de nouveaux utilisateurs. Après confirmation de son e-mail, on peut contribuer et s’authentifier en utilisant des services tiers.

Vous pouvez aussi désactiver les inscriptions de nouveaux utilisateurs en utilisant `REGISTRATION_OPEN`.

Le nombre de tentatives d’authentification est sujet à **Limite de requêtes**.
2.5.2 Backends d'authentification

La solution intégrée de Django est utilisée pour l'authentification, et inclue différents services sociaux pour le faire. Son utilisation signifie que vous pouvez importer la base de données d'utilisateurs d'autres projets basés sur Django (voir *Migrating from Pootle*).

De plus Django peut aussi être configuré pour utiliser d'autre méthodes d'authentification.

**Voir aussi :**

*Paramètres d'authentification* décrit comment configurer l'authentification sur l'image Docker officielle.

2.5.3 Authentification sociale

Grâce à *Welcome to Python Social Auth's documentation!*, Weblate supporte l’authentification par différents services tiers comme GitLab, Ubuntu, Fedora, etc.

Merci de consulter leur documentation pour des instructions de configuration générique dans *Django Framework*.

**Note** : Par défaut, Weblate sur des services tiers d'authentification pour confirmer les adresses e-mail. Si certains services de votre choix ne le supportent pas, merci d'imposer la confirmation de l'email en configurant `FORCE_EMAIL_VALIDATION`. Par exemple :

```
SOCIAL_AUTH_OPENSUSE_FORCE_EMAIL_VALIDATION = True
```

**Voir aussi :**

*Pipeline*

Activier d'autres backends est plutôt simple, il suffit d'ajouter une ligne à `AUTHENTICATION_BACKENDS` et les clefs nécessaires à une méthode d'authentification donnée. Notez que certains backends ne fournissent pas d'adresse e-mail utilisateur par défaut, vous devez la demander explicitement, sinon Weblate ne sera pas en mesure de créditer les contributions des utilisateurs comme il se doit.

**Indication** : La plupart des « backends » d'authentification nécessitent HTTPS. Une fois que HTTPS est activé sur votre serveur web, veuillez configurer Weblate pour qu'il l'annonce correctement en utilisant `ENABLE_HTTPS`, ou `WEBLATE_ENABLE_HTTPS` dans un conteneur Docker.

**Voir aussi :**

*Python Social Auth backend*

**Authentification OpenID**

Pour les services reposant sur OpenID il n'y a généralement qu'à les activer. La section suivante activer l'authentification avec OpenID pour OpenSUSE, Fedora et Ubuntu :

```
# Authentication configuration
AUTHENTICATION_BACKENDS = {
    "social_core.backends.email.EmailAuth",
    "social_core.backends.suse.OpenSUSEOpenId",
    "social_core.backends.ubuntu.UbuntuOpenId",
    "social_core.backends.fedora.FedoraOpenId",
    "weblate.accounts.auth.WeblateUserBackend",
}
```

**Voir aussi :**

*OpenID*
S’authentifier avec GitHub

Vous devez créer une application OAuth sur Github et confier à Weblate tous ses petits secrets :

```python
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.github.GithubOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

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

Le compte GitHub doit avoir une callback URL configurée tel que : https://example.com/accounts/complete/github/.


**Note** : La callback URL fournie par Weblate pendant l’authentification inclue le domaine configuré. En cas d’erreurs sur l’URL, vous devriez pouvoir arranger cela. Voir Set correct site domain.

Voir aussi :
GitHub

S’authentifier avec Bitbucket

Vous devez créer une application sur Bitbucket et confier à Weblate tous ses petits secrets :

```python
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.bitbucket.BitbucketOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_BITBUCKET_OAUTH2_KEY = "Bitbucket Client ID"
SOCIAL_AUTH_BITBUCKET_OAUTH2_SECRET = "Bitbucket Client Secret"
SOCIAL_AUTH_BITBUCKET_OAUTH2_VERIFIED_EMAILS_ONLY = True
```

**Note** : La callback URL fournie par Weblate pendant l’authentification inclue le domaine configuré. En cas d’erreurs sur l’URL, vous devriez pouvoir arranger cela. Voir Set correct site domain.

Voir aussi :
Bitbucket
Google OAuth 2


L’URL de redirection est https://WEBLATE SERVER/accounts/complete/google-oauth2/

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (  
    "social_core.backends.google.GoogleOAuth2",  
    "social_core.backends.email.EmailAuth",  
    "weblate.accounts.auth.WeblateUserBackend",  
)

# Social auth backends setup
SOCIAL_AUTH_GOOGLE_OAUTH2_KEY = "Client ID"
SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET = "Client secret"
```

**Note** : La callback URL fournie par Weblate pendant l’authentification inclue le domaine configuré. En cas d’erreurs sur l’URL, vous devriez pouvoir arranger cela. Voir *Set correct site domain*.

Voir aussi :

Google

Facebook OAuth 2

Comme d’habitude avec les services OAuth 2, vous devez enregistrer une application chez Facebook. Une fois que c’est fait, vous pouvez configurer Weblate pour l’utiliser :

L’URL de redirection est https://WEBLATE SERVER/accounts/complete/facebook/

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (  
    "social_core.backends.facebook.FacebookOAuth2",  
    "social_core.backends.email.EmailAuth",  
    "weblate.accounts.auth.WeblateUserBackend",  
)

# Social auth backends setup
SOCIAL_AUTH_FACEBOOK_KEY = "key"
SOCIAL_AUTH_FACEBOOK_SECRET = "secret"
SOCIAL_AUTH_FACEBOOK_SCOPE = ["email", "public_profile"]
```

**Note** : La callback URL fournie par Weblate pendant l’authentification inclue le domaine configuré. En cas d’erreurs sur l’URL, vous devriez pouvoir arranger cela. Voir *Set correct site domain*.

Voir aussi :

Facebook
**GitLab OAuth 2**

Pour utiliser GitLab OAuth 2, vous devez enregistrer une application sur <https://gitlab.com/profile/applications>.

L’URL de redirection est `https://WEBLATE SERVER/accounts/complete/gitlab/` and ensure you mark the `read_user` scope.

```python
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.gitlab.GitLabOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_GITLAB_KEY = "Application ID"
SOCIAL_AUTH_GITLAB_SECRET = "Secret"
SOCIAL_AUTH_GITLAB_SCOPE = ["read_user"]

# If you are using your own GitLab
# SOCIAL_AUTH_GITLAB_API_URL = 'https://gitlab.example.com/'
```

**Note** : La callback URL fournie par Weblate pendant l’authentification inclut le domaine configuré. En cas d’erreurs sur l’URL, vous devriez pouvoir arranger cela. Voir [Set correct site domain](#).

Voir aussi :

- [GitLab](#)

**Active Directory Microsoft Azure**

Weblate peut être configuré pour utiliser des tenants communs ou spécifiques à l’authentification.


```python
# Azure AD common
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.azuread.AzureADOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# OAuth2 keys
SOCIAL_AUTH_AZUREAD_OAUTH2_KEY = ""
SOCIAL_AUTH_AZUREAD_OAUTH2_SECRET = ""
```

```python
# Azure AD Tenant
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.azuread_tenant.AzureADTenantOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)
```

(suite sur la page suivante)
# OAuth2 keys

```
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_KEY = ""
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_SECRET = ""
# Tenant ID
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_TENANT_ID = ""
```

**Note**: La callback URL fournie par Weblate pendant l'authentification inclue le domaine configuré. En cas d'erreurs sur l'URL, vous devriez pouvoir arranger cela. Voir *Set correct site domain*.

Voir aussi :
Microsoft Azure Active Directory

**Slack**

Pour utiliser OAuth 2 avec Slack, vous devez enregistrer une application sur <https://api.slack.com/apps>.

L'URL de redirection est `https://WEBLATE SERVER/accounts/complete/slack/`.

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

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

**Note**: La callback URL fournie par Weblate pendant l'authentification inclue le domaine configuré. En cas d'erreurs sur l'URL, vous devriez pouvoir arranger cela. Voir *Set correct site domain*.

Voir aussi :
Slack

**Overriding authentication method names and icons**

Vous pouvez remplacer le nom affiché et l'icône de la méthode d'authentification en utilisant des paramètres comme `SOCIAL_AUTH_<NAME>_IMAGE` et `SOCIAL_AUTH_<NAME>_TITLE`. Par exemple le remplacement du nom pour la méthode d'authentification utilisant Auth0 ressemblerait à ceci :

```
SOCIAL_AUTH_AUTH0_IMAGE = "custom.svg"
SOCIAL_AUTH_AUTH0_TITLE = "Custom auth"
```
Désactiver l’authentification par mot de passe

L’authentification par e-mail et mot de passe peut être désactivée en supprimant `social_core.backends.email.EmailAuth` de `AUTHENTICATION_BACKENDS`. Conservez toujours `weblate.accounts.auth.WeblateUserBackend` là, c’est nécessaire pour les fonctionnalités essentielles de Weblate.

Désactiver l’authentification par adresse e-mail désactivera toutes les fonctionnalités liées aux e-mails – l’invitation d’utilisateur ou la fonctionnalité de réinitialisation de mot de passe.

**Astuce** : Vous pouvez toujours utiliser l’authentification par e-mail et mot de passe sur l’interface d’admin pour les utilisateurs créés manuellement. Allez simplement sur `/admin/login/`.

Par exemple, l’authentification utilisant seulement openSUSE OpenID peut être réalisée comme suit :

```python
AUTHENTICATION_BACKENDS = (
    "social_core.backends.suse.OpenSUSEOpenId",
    "weblate.accounts.auth.WeblateUserBackend",
)
```

### 2.5.4 S’authentifier par mot de passe

Le fichier par défaut `settings.py` vient avec un ensemble raisonnable de `AUTH_PASSWORD_VALIDATORS` :
- Les mots de passe ne peuvent être similaires aux autres informations personnelles.
- Les mots de passe doivent contenir au moins 10 caractères.
- Les mots de passe ne peuvent être communément employés.
- Les mots de passe ne peuvent être entièrement numériques.
- Les mots de passe ne peuvent consister en un seul caractère ou que des espaces.
- Les mots de passe ne peuvent contenir un mot de passe utilisé dans le passé.

Vous pouvez personnaliser ces paramètres pour les faire correspondre à votre politique de mots de passe.

De plus vous pouvez aussi installer `django-zxcvbn-password` qui donne une estimation plutôt réaliste de la complexité d’un mot de passe et permet de les rejeter en dessous d’un certain seuil.

### 2.5.5 S’authentifier avec SAML

Nouveau dans la version 4.1.1.

Merci de suivre les instructions d’authentification sociale spécifiques à python pour la configuration. Différences notable :
- Weblate supporte single IDP qui doit être appelé `weblate` dans `SOCIAL_AUTH_SAML_ENABLED_IDPS`.
- L’URL SAML de métadonnées XML est `/accounts/metadata/saml/`.

Exemple de configuration :

```python
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.email.EmailAuth",
    "social_core.backends.saml.SAMLAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_SAML_SP_ENTITY_ID = f"https://[SITE_DOMAIN]/accounts/metadata/saml/"
SOCIAL_AUTH_SAML_SP_PUBLIC_CERT = "-----BEGIN CERTIFICATE-----"
```

(suite sur la page suivante)
The default configuration extracts user details from following attributes, configure your IDP to provide them:

<table>
<thead>
<tr>
<th>Attribut</th>
<th>Reference de l’URI SAML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nom complet</td>
<td>urn:oid:2.5.4.3</td>
</tr>
<tr>
<td>Prénom</td>
<td>urn:oid:2.5.4.42</td>
</tr>
<tr>
<td>Nom de famille</td>
<td>urn:oid:2.5.4.4</td>
</tr>
<tr>
<td>Adresse courriel</td>
<td>urn:oid:0.9.2342.19200300.100.1.3</td>
</tr>
<tr>
<td>Nom d’utilisateur</td>
<td>urn:oid:0.9.2342.19200300.100.1.1</td>
</tr>
</tbody>
</table>

**Indication** : L’exemple ci-dessus et l’image Docker définissent un IDP appelé weblate. Vous devrez peut-être configurer cette chaîne en tant que *Relay* dans votre IDP (fournisseur d’identité).

**Voir aussi** :

*Configuring SAML in Docker, SAML*

### 2.5.6 S’authentifier avec LDAP

C’est mieux de s’authentifier avec LDAP en utilisant le paquet *django-auth-ldap*. Vous pouvez l’installer comme d’habitude :

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

# Using apt-get
apt-get install python-django-auth-ldap
```
**Indication :** Ce package est inclus dans le conteneur Docker, voir *Installing using Docker.*

**Note :** Des incompatibilités dans le module Python LDAP 3.1.0 peuvent vous empêcher d'utiliser cette version. Si vous obtenez l'erreur `AttributeError: “module” object has no attribute “_trace_level”`, le retour à la version 3.0.0 de python-ldap pourrait résoudre le problème.

Une fois que le paquet est installé, vous pouvez le relier à l'authentification Django :

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

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

# DN to use for authentication
AUTH_LDAP_USER_DN_TEMPLATE = "cn=%(user)s,o=Example"

# Depending on your LDAP server, you might use a different DN
# like:
# AUTH_LDAP_USER_DN_TEMPLATE = 'ou=users,dc=example,dc=com'

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

# Hide the registration form
REGISTRATION_OPEN = False
```

**Note :** Vous devriez supprimer 'social_core.backends.email.EmailAuth' du paramètre `AUTHENTICATION_BACKENDS`, sinon les utilisateurs seront en mesure de créer un mot de passe sur Weblate et de s'authentifier avec. Garder 'weblate.accounts.auth.WeblateUserBackend' est encore nécessaire pour gérer les permissions et permettre l'utilisation anonyme. Cela permet aussi de s'identifier avec un compte admin local, si vous en avez créé un (p. ex. en utilisant `createadmin`).
En utilisant le mot de passe bind

Si vous ne pouvez pas utiliser la liaison directe pour l'authentification, vous devrez utiliser la recherche, et fournir un utilisateur à lier pour la recherche. Par exemple :

```python
import ldap
from django_auth_ldap.config import LDAPSearch

AUTH_LDAP_BIND_DN = ""
AUTH_LDAP_BIND_PASSWORD = ""
AUTH_LDAP_USER_SEARCH = LDAPSearch(
    "ou-users,dc=example,dc=com", ldap.SCOPE_SUBTREE, "\(uid=%(user)s\)"
)
```

Intégration avec Active Directory

```python
import ldap
from django_auth_ldap.config import LDAPSearch, NestedActiveDirectoryGroupType

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

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

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

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

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

Voir aussi :

Django Authentication Using LDAP, Authentication

2.5.7 S'authentifier avec CAS

On peut s'authentifier avec CAS en utilisant un paquet comme django-cas-ng.

La première étape est de révéler le champ e-mail de l'utilisateur via CAS. Cela doit être configuré sur le serveur de CAS, et demande d'utiliser au moins CAS v2 vu que CAS v1 ne supporte pas du tout les attributs.

L'étape suivante est de mettre à jour Weblate pour utiliser votre server CAS et les attributs.

Pour installer django-cas-ng :

```bash
pip install django-cas-ng
```
Une fois que le paquet est installé vous pouvez relier au système d’authentification Django en modifiant le fichier `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")
```

Au final, un signal peut être utilisé pour assigner le champ e-mail à l’objet utilisateur. Pour que cela fonctionne, vous devez importer le signal depuis le paquet `django-cas-ng` et connecter votre code à ce signal. Faire cette manipulation depuis les fichiers de configuration peut être problématique, alors on conseille de faire différemment :

- Dans la méthode `django.apps.AppConfig.ready()` de la configuration de l’application
- Dans le fichier `urls.py` du projet (quand il n’existe pas de modèle)

```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()
```

Voir aussi :

Django CAS NG

2.5.8 Configurer l’authentification Django avec des services tiers

Généralement n’importe quel module d’authentification Django devrait fonctionner avec Weblate. Il suffit de suivre les instructions du module et de se rappeler de conserver le backend utilisateur de Weblate.

Voir aussi :

*Authentifier avec LDAP, Authentifier avec CAS*

Typiquement l’installation va consister à l’ajout d’un backend d’authentification à `AUTHENTICATION_BACKENDS` et l’installation d’une app d’authentification (s’il y en une) dans `INSTALLED_APPS` :

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

INSTALLED_APPS += (  
    # Install authentication app here
)  
```
2.6 Contrôle d’accès

Weblate est doté d’un système de privilèges précis permettant d’attribuer des autorisations d’accès à l’ensemble de l’instance ou à une partie limitée de celle-ci.

Modifié dans la version 3.0 : Avant Weblate 3.0, le système de privilèges était uniquement basé sur celui de Django, mais il est maintenant spécifiquement construit pour Weblate. Si vous utilisez une version plus ancienne, veuillez consulter la documentation pour la version spécifique que vous utilisez.

2.6.1 Contrôle d’accès simple

Si vous n’administrez pas toute l’installation de Weblate et que vous avez juste accès à la gestion de certains projets (comme sur Weblate hébergé), vos options de gestion de contrôle d’accès sont limitées aux paramètres suivants. Si vous n’avez pas besoin d’une configuration complexe, ceux-ci sont suffisants pour vous.

Contrôle d’accès au projet

Note : Cette fonctionnalité n’est pas disponible pour les projets utilisant le plan « projet Libre » hébergé sur Hosted Weblate.

Vous pouvez limiter les accès d’un utilisateur à des projets individuels en sélectionnant un Contrôle d’accès différent. Les options disponibles sont :

Public
Visible publiquement, traduisible par les tous les utilisateurs connectés.

Protégé
Visible publiquement, mais traduisible uniquement pour les utilisateurs sélectionnés.

Privé
Visible et traduisible uniquement pour les utilisateurs sélectionnés.

Personnalisé
Les fonctionnalités Gestion d’utilisateurs seront désactivées ; par défaut, les utilisateurs ne peuvent faire aucune action sur le projet. Vous devrez paramétrer toutes les autorisations en utilisant Contrôle d’accès personnalisé.

Les Contrôles d’accès peuvent être changés dans l’onglet Accès dans la configuration (Gestion → Paramètres) de chaque projet.
La valeur par défaut peut être modifiée avec `DEFAULT_ACCESS_CONTROL`.

**Note** : Même pour les projets Privés, certaines informations sur votre projet seront exposées : les résumés de statistiques et de langage pour toute la session incluront les décomptes pour tous les projets malgré le paramètre de contrôle d'accès. Votre nom de projet et d'autres informations ne peuvent pas être révélés par ce moyen.

**Note** : Les sets de permissions disponibles par défaut pour les utilisateurs dans les projets Public, Protégé, et Privé peuvent être redéfinis via l'instance administrateur de Weblate en utilisant paramètres personnalisés.

Voir aussi :
*Contrôle d'accès*

**Gestion du contrôle d'accès par projet**

Les utilisateurs ayant le privilège *Gérer l'accès au projet* (voir privilèges) peuvent gérer les utilisateurs dans les projets en les ajoutant aux équipes. La collection initiale d'équipes est fournie par Weblate, mais des équipes supplémentaires peuvent être définies pour fournir un contrôle d'accès plus fin. Vous pouvez limiter les équipes aux langues et leur attribuer des rôles d'accès désignés (voir privilèges).

Les équipes suivantes sont créées automatiquement pour chaque projet :

Pour les projets Public, Protégé et Privé :
- **Administration**
  - Inclut toutes les autorisations disponibles pour le projet.
- **Révision (seulement si réviser le processus est activé)**
  - Peut approuver les traductions lors de la révision.

Pour les projets Protégés et Privés uniquement :
- **Traduire**
  - Peut traduire le projet et téléverser des traductions effectuées hors ligne.
- **Sources**
  - Peut modifier des chaînes source (si autorisé par les paramètres du projet) et les informations de chaînes source.
- **Langues**
  - Peut gérer les langues traduites (ajouter ou supprimer des traductions).
- **Glossaire**
  - Peut gérer le glossaire (ajouter ou supprimer des entrées, ou téléverser).
- **Mémoire**
  - Peut gérer la mémoire de traduction.
- **Captures d'écran**
  - Peut gérer les captures d'écran (les ajouter ou les supprimer, et les associer à des chaînes de caractères source).
- **Traduction automatique**
  - Peut utiliser la traduction automatique.
- **Système de contrôle de versions**
  - Peut gérer le système de contrôle des versions et accéder au dépôt exporté.
- **Facturation**
  - Peut accéder aux informations et paramètres de facturation (voir *Facturation*).
Ces fonctionnalités sont disponibles dans la page *Access control*, qui est accessible depuis le menu *Manage ➔ Users* du projet.

**Administrateurs de l’équipe**

Nouveau dans la version 4.15.

Chaque équipe peut disposer d’un administrateur qui peut ajouter et supprimer des utilisateurs dans/de l’équipe. Ceci est utile si vous souhaitez mettre en place des équipes auto-gérées.
**Invitation de nouvel utilisateur**

En plus d'ajouter un utilisateur existant au projet, il est possible d'en inviter de nouveaux. Tout nouvel utilisateur sera créé immédiatement, mais son compte restera inactif jusqu'à ce qu'il se connecte à l'aide d'un lien contenu dans l'invitation envoyée par e-mail. Il n'est pas nécessaire d'avoir des privilèges à l'échelle du site pour cela, une autorisation de gestion d'accès sur le périmètre du projet (par exemple une appartenance au groupe *Administration*) serait suffisante.

**Indication** : Si l'utilisateur invité a manqué la validité de l'invitation, il peut définir son mot de passe à l'aide de l'adresse courriel invitée dans le formulaire de réinitialisation du mot de passe puisque le compte est déjà créé.

Nouveau dans la version 3.11 : Il est possible de renvoyer le courriel d'invitation aux utilisateurs (cet envoi annule toutes les invitations envoyées précédemment).

Le même type d'invitations sont disponibles partout sur le site depuis *management interface* dans l'onglet *Users*.

**Les utilisateurs bloqués**

Nouveau dans la version 4.7.

Dans certains cas, des utilisateurs ont un mauvais comportement. Vous avez la possibilité de les empêcher de contribuer. L'utilisateur bloqué pourra toujours voir le projet s'il a les autorisations, mais il ne pourra plus contribuer.

**Gestion des droits par projet**

Réglez vos projets sur *Protégé* ou *Privé*, et gérez les utilisateurs par projet dans l'interface Webate.

Par défaut, cela empêche Webate d'accorder les accès fournis par les *Users* et les *Viewers default teams* en raison de la configuration propre de ces équipes. Cela ne vous empêche pas d'accorder des permissions à ces projets à l'échelle du site en modifiant les équipes par défaut, créer un nouveau, ou créer des paramètres personnalisés supplémentaires pour un composant individuel comme décrit dans *Contrôle d'accès personnalisé* ci-dessous.

L'un des principaux avantages de la gestion des permissions via l'interface utilisateur de Webate est que vous pouvez la déléguer à d'autres utilisateurs sans leur donner le privilège de super-utilisateur. Pour ce faire, ajoutez-les à l'équipe *Administration* du projet.

**2.6.2 Contrôle d'accès personnalisé**

**Note** : Cette fonctionnalité n'est pas disponible pour les projets utilisant le plan « projet Libre » hébergé sur Hosted Webate.

Le système de permission est basé sur les équipes et les rôles, où les rôles définissent un ensemble de permissions, et les équipes les lient aux utilisateurs et aux traductions, consultez *Utilisateurs, rôles, équipes et droits* pour plus de détails.

Les fonctionnalités les plus puissantes du système de contrôle d'accès de Webate ne sont pour l'instant disponibles qu'à travers l'interface d'administration *Django*. Vous pouvez l'utiliser pour gérer les permissions de n'importe quel projet. Vous n'avez pas nécessairement besoin de passer en mode *Custom contrôle d'accès* pour l'utiliser. Cependant, vous devez avoir les privilèges de super-utilisateur pour l'utiliser.

Si vous n'êtes pas intéressé par les détails de l'implémentation et que vous voulez juste créer une configuration suffisamment simple, basée sur les valeurs par défaut, ou que vous n'avez pas accès à l'ensemble de l'installation de Webate (comme sur *Hosted Webate*), veuillez vous référer à la section *Contrôle d'accès simple*. 

242 Chapitre 2. Documentation pour l'administrateur
Configurations communes

Cette section contient un aperçu de certaines configurations courantes susceptibles de vous intéresser.

Gestion des droits à l’échelle du site

Pour gérer les autorisations d’une instance entière en une seule fois, ajoutez des utilisateurs aux équipes appropriées équipes par défaut :

— Users (ceci est fait par défaut par la attribution automatique des équipes).
— Reviewers (si vous utilisez review workflow avec des réviseurs d édiés).
— Gestionnaires (si vous souhaitez déléguer la plupart des opérations de gestion à une autre personne).

Vous devez garder tous les projets configurés en tant que Public (voir Contrôle d’accès au projet), sinon les autorisations pour tout le site fournies par l’appartenance aux équipes Users et Reviewers n’auront aucun effet.

Vous pouvez également accorder des permissions supplémentaires de votre choix aux équipes par défaut. Par exemple, vous pouvez donner une permission de gérer les captures d’écran à tous les Utilisateurs.

Vous pouvez également définir de nouvelles équipes personnalisées. Si vous souhaitez continuer à gérer les permissions de ces équipes à l’échelle du site, choisissez une valeur appropriée pour le paramètre Sélection du projet (par exemple Tous les projets ou Tous les projets publics).

Autorisations personnalisées pour les langues, les composants ou les projets

Vous pouvez créer vos propres équipes dédiées pour gérer les autorisations pour des objets distincts tels que des langages, des composants et des projets. Bien que ces équipes ne puissent accorder que des privilèges supplémentaires, vous ne pouvez révoquer aucune autorisation accordée par des équipes à l’échelle du site ou par projet en ajoutant un autre équipe personnalisée.

Exemple :

Si vous souhaitez (peu importe la raison) autoriser la traduction vers une langue spécifique (par exemple « tchèque ») à un groupe restreint de traducteurs fiables tout en gardant publiques les traductions vers d’ autres langues, vous devrez :

1. Retirer le droit de traduction en Tchèque pour tous les utilisateurs. Dans la configuration par défaut, cela peut être fait en modifiant Users default team.

<table>
<thead>
<tr>
<th>Tableau 1 – Groupe Utilisateurs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sélection de la langue</td>
</tr>
<tr>
<td>Langues</td>
</tr>
</tbody>
</table>

2. Ajouter une équipe spécifique pour les traducteurs en Tchèque.

<table>
<thead>
<tr>
<th>Tableau 2 – Groupe Traducteurs tchèques</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rôles</td>
</tr>
<tr>
<td>Sélection de projets</td>
</tr>
<tr>
<td>Sélection de la langue</td>
</tr>
<tr>
<td>Langues</td>
</tr>
</tbody>
</table>

3. Ajoutez les utilisateurs auxquels vous souhaitez accorder les autorisations dans cette équipe.

Comme vous pouvez le voir, la gestion des autorisations de cette façon est puissante, mais peut être un travail assez fastidieux. Vous ne pouvez pas le déléguer à un autre utilisateur, sauf en accordant des autorisations de super-utilisateur.
Utilisateurs, rôles, équipes et droits

Les modèles d’authentification se composent de plusieurs éléments :

**Droit**
Droit individuel. Les droits ne peuvent pas être attribués à un utilisateur. Ils doivent être attribués via des rôles.

**Rôle**
Un rôle définit un ensemble de droits. Ils peuvent être utilisés à plusieurs endroits pour faciliter l’administration.

**Utilisateur**
Un utilisateur peut être membre de plusieurs équipes.

**Groupe**
Un groupe relie les rôles, les utilisateurs et les éléments d’authentification (projets, langues et listes de composants).

Note : Une équipe peut n’avoir aucun rôle assigné, dans ce cas l’accès à la navigation du projet par n’importe qui est supposé (voir ci-dessous).

Accès pour parcourir un projet

Un utilisateur doit être membre d’une équipe liée au projet ou de l’un des composants. Il suffit d’être membre, aucune permission spécifique n’est nécessaire pour naviguer dans un projet (ceci est utilisé dans l’équipe par défaut Lecteurs, voir List of teams).

Accès pour parcourir un composant

Un utilisateur peut accéder à des composants non restreints dès qu’il est en mesure d’accéder au projet des composants (et disposer de toutes les permissions qui lui ont été accordées pour le projet). Si l’option Accès restreint est activée, l’accès au composant nécessite des autorisations explicites pour le composant (ou une liste de composants dans laquelle se trouve le composant).
Portée des équipes

L’étendue de l’autorisation attribuée par les rôles dans les équipes est appliquée par les règles suivantes :
— Si l’équipe spécifie une :guilabel: Component list, toutes les autorisations accordées aux membres de cette équipe sont accordées pour tous les composants des listes de composants attachées à l’équipe, et un accès sans autorisation supplémentaire est donné pour tous les projets auxquels appartiennent ces composants. :guilabel: Components et :guilabel: Projects sont ignorés.
— Si l’équipe spécifie un Composants, toutes les permissions données aux membres de cette équipe sont accordées pour tous les composants attachés à l’équipe et un accès sans permissions supplémentaires est accordé pour tous les projets dans lesquels ces composants se trouvent. Projets sont ignorés.
— Sinon, si l’équipe spécifie des Projets, soit en les listant directement, soit en ayant Sélection de projets défini sur une valeur comme Tous les projets publics, toutes ces permissions sont appliquées à tous les projets, ce qui accorde effectivement les mêmes permissions d’accès à tous les projets composants non restreints.
— The restrictions imposed by a team’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.

Indication : Utilisez Sélection de la langue ou Sélection de projet pour automatiser l’inclusion de toutes les langues ou projets.

Exemple : 

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

<table>
<thead>
<tr>
<th>Rôles</th>
<th>Réviser les chaînes, Gérer le dépôt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Composants</td>
<td>foo/bar</td>
</tr>
<tr>
<td>Langues</td>
<td>Espagnol</td>
</tr>
</tbody>
</table>

Members of that team will have following permissions (assuming the default role settings) :
— Accès général (navigation) à l’ensemble du projet foo incluant les deux composants :guilabel: foo/bar et "foo/baz". 
— Vérifier la traduction espagnole des chaînes de :guilabel: foo/bar (pas ailleurs).
— Gérer le système de contrôle des versions pour le dépôt :guilabel: foo/bar complet, par exemple valider les changements en attente effectués par les traducteurs pour toutes les langues.

Automatic team assignments

On the bottom of the Group editing page in the Django admin interface, you can specify Automatic team assignments, which is a list of regular expressions used to automatically assign newly created users to a team based on their e-mail addresses. This assignment only happens upon account creation.

L’utilisation la plus courante de la fonctionnalité est d’affecter tous les nouveaux utilisateurs à une équipe par défaut. Pour faire cela, vous voudrez probablement garder la valeur par défaut (\^[\^.*]$) dans le champ de l’expression rationnelle. Une autre utilisation de cette option peut être de donner, par défaut, des privilèges supplémentaires aux employés de votre entreprise. En supposant qu’ils utilisent tous des adresses e-mail avec votre nom de domaine, cela peut être accompli en utilisant une expression rationnelle comme \^[\^.*@mycompany.com].

Note : Automatic team assignment to Users and Viewers is always recreated when upgrading from one Weblate version to another. If you want to turn it off, set the regular expression to ^$ (which won’t match anything).

Note : As for now, there is no way to bulk-add already existing users to some team via the user interface. For that, you may resort to using the REST API.
# Default teams and roles

Après l’installation, un ensemble d’équipes par défaut est créé (voir List of teams).

Ces rôles et équipes sont créés lors de l’installation. Les rôles intégrés sont toujours à jour avec la base de données lorsqu’ils sont mis à jour. Vous ne pouvez pas les changer. Si vous voulez définir un nouveau rôle, vous devez le définir manuellement.

## Liste des privilèges et des rôles intégrés

<table>
<thead>
<tr>
<th>Portée</th>
<th>Droits</th>
<th>Rôles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facturation (voir Facturation)</td>
<td>Consulter la facturation</td>
<td>Administration, Facturation</td>
</tr>
<tr>
<td>Modifications</td>
<td>Télécharger les modifications</td>
<td>Administration</td>
</tr>
<tr>
<td>Commentaires</td>
<td>Publier un commentaire</td>
<td>Administration, Modifier la source</td>
</tr>
<tr>
<td></td>
<td>Supprimer le commentaire</td>
<td>Administration</td>
</tr>
<tr>
<td></td>
<td>Fermer le commentaire</td>
<td>Administration, Vérifier les chaînes</td>
</tr>
<tr>
<td>Composant</td>
<td>Modifier les paramètres du composant</td>
<td>Administration</td>
</tr>
<tr>
<td></td>
<td>Composant verrouillé, traductions impossibles</td>
<td>Administration</td>
</tr>
<tr>
<td>Glossaire</td>
<td>Ajouter une entrée de glossaire</td>
<td>Administration, Gérer le glossaire</td>
</tr>
<tr>
<td></td>
<td>Modifier une entrée de glossaire</td>
<td>Administration, Gérer le glossaire</td>
</tr>
<tr>
<td></td>
<td>Supprimer une entrée de glossaire</td>
<td>Administration, Gérer le glossaire</td>
</tr>
<tr>
<td></td>
<td>Télérer des entrées de glossaires</td>
<td>Administration, Gérer le glossaire</td>
</tr>
<tr>
<td>Suggestions automatiques</td>
<td>Utiliser les suggestions automatiques</td>
<td>Administrateur, Modifier la source</td>
</tr>
<tr>
<td>Mémoire de traduction</td>
<td>Modifier le mémoire de traduction</td>
<td>Administration, Gérer le mémoire</td>
</tr>
<tr>
<td></td>
<td>Supprimer le mémoire de traduction</td>
<td>Administration, Gérer le mémoire</td>
</tr>
<tr>
<td>Projets</td>
<td>Modifier les paramètres du projet</td>
<td>Administration</td>
</tr>
<tr>
<td></td>
<td>Gérer l’accès au projet</td>
<td>Administration</td>
</tr>
<tr>
<td>Rapports</td>
<td>Télécharger des rapports</td>
<td>Administration</td>
</tr>
<tr>
<td>Captures d’écran</td>
<td>Ajouter une capture d’écran</td>
<td>Administration, Gérer les captures</td>
</tr>
<tr>
<td></td>
<td>Modifier la capture d’écran</td>
<td>Administration, Gérer les captures</td>
</tr>
<tr>
<td></td>
<td>Supprimer la capture d’écran</td>
<td>Administration, Gérer les captures</td>
</tr>
<tr>
<td>Chaînes sources</td>
<td>Modifier des informations de chaîne supplémentaire</td>
<td>Administrateur, Modifier la source</td>
</tr>
<tr>
<td>Chaînes</td>
<td>Ajouter une nouvelle chaîne</td>
<td>Administrateur</td>
</tr>
<tr>
<td></td>
<td>Supprimer une chaîne</td>
<td>Administrateur</td>
</tr>
<tr>
<td></td>
<td>Ignorer les vérifications en échec</td>
<td>Administrateur, Modifier la source</td>
</tr>
<tr>
<td></td>
<td>Modifier les chaînes</td>
<td>Administrateur, Modifier la source</td>
</tr>
<tr>
<td></td>
<td>Réviser les chaînes</td>
<td>Administrateur, Vérifier les chaînes</td>
</tr>
<tr>
<td></td>
<td>Modifier une chaîne lorsque les suggestions sont appliquées</td>
<td>Administrateur, Vérifier les chaînes</td>
</tr>
<tr>
<td></td>
<td>Modifier les chaînes sources</td>
<td>Administrateur, Modifier la source</td>
</tr>
<tr>
<td>Suggestions</td>
<td>Accepter une suggestion</td>
<td>Administrateur, Modifier la source</td>
</tr>
<tr>
<td></td>
<td>Ajouter une suggestion</td>
<td>Administrateur, Modifier la source</td>
</tr>
<tr>
<td></td>
<td>Supprimer une suggestion</td>
<td>Administrateur, Utilisateur expérimentaire</td>
</tr>
<tr>
<td></td>
<td>Voter pour une suggestion</td>
<td>Administrateur, Modifier la source</td>
</tr>
<tr>
<td>Traductions</td>
<td>Ajouter une langue à traduire</td>
<td>Administrateur, Utilisateur expérimentaire</td>
</tr>
<tr>
<td></td>
<td>Réaliser une traduction automatique</td>
<td>Administrateur, Traduction automatique</td>
</tr>
<tr>
<td></td>
<td>Supprimer une traduction existante</td>
<td>Administrateur, Gérer les langues</td>
</tr>
<tr>
<td></td>
<td>Télécharger le fichier de traduction</td>
<td>Administrateur, Modifier la source</td>
</tr>
<tr>
<td></td>
<td>Ajouter plusieurs langues à traduire</td>
<td>Administrateur, Gérer les langues</td>
</tr>
<tr>
<td>Téléversements</td>
<td>Définir l’auteur de la traduction téléversée</td>
<td>Administrateur</td>
</tr>
<tr>
<td></td>
<td>Remplacer les traductions existantes par un téléversement</td>
<td>Administrateur, Modifier la source</td>
</tr>
<tr>
<td></td>
<td>Téléverser les traductions</td>
<td>Administrateur, Modifier la source</td>
</tr>
<tr>
<td>Système de contrôle de versions</td>
<td>Accéder au dépôt interne</td>
<td>Administrateur, Accès au dépôt, Gérer le dépôt</td>
</tr>
<tr>
<td></td>
<td>Commiter les modifications dans le dépôt interne</td>
<td>Administrateur, Gérer le dépôt</td>
</tr>
<tr>
<td></td>
<td>Pousser les modifications depuis le dépôt local</td>
<td>Administrateur, Gérer le dépôt</td>
</tr>
<tr>
<td></td>
<td>Réinitialiser les modifications dans le dépôt interne</td>
<td>Administrateur, Gérer le dépôt</td>
</tr>
</tbody>
</table>
### Privilèges pour l'ensemble du site

<table>
<thead>
<tr>
<th>Portée</th>
<th>Droits</th>
<th>Rôles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voire l'emplacement du dépôt en amont</td>
<td><code>Administration</code>, Accès au dépôt, Gérer le dépôt</td>
<td><code>Administration</code></td>
</tr>
<tr>
<td>Mettre à jour le dépôt interne</td>
<td></td>
<td><code>Gérer le dépôt</code></td>
</tr>
<tr>
<td>Utiliser l'interface de gestion</td>
<td></td>
<td><code>Gérer le dépôt</code></td>
</tr>
<tr>
<td>Créer de nouveaux projets</td>
<td></td>
<td><code>Gérer le dépôt</code></td>
</tr>
<tr>
<td>Ajouter des définitions de langue</td>
<td></td>
<td><code>Gérer le dépôt</code></td>
</tr>
<tr>
<td>Gérer les définitions de langue</td>
<td></td>
<td><code>Gérer le dépôt</code></td>
</tr>
<tr>
<td>Gérer les équipes</td>
<td></td>
<td><code>Gérer le dépôt</code></td>
</tr>
<tr>
<td>Gérer les utilisateurs</td>
<td></td>
<td><code>Gérer le dépôt</code></td>
</tr>
<tr>
<td>Gérer les rôles</td>
<td></td>
<td><code>Gérer le dépôt</code></td>
</tr>
<tr>
<td>Gérer les annonces</td>
<td></td>
<td><code>Gérer le dépôt</code></td>
</tr>
<tr>
<td>Gérer le mémoire de traduction</td>
<td></td>
<td><code>Gérer la traduction automatique</code></td>
</tr>
<tr>
<td>Gérer la traduction automatique</td>
<td></td>
<td><code>Gérer la traduction automatique</code></td>
</tr>
<tr>
<td>Gérer la liste des composants</td>
<td></td>
<td><code>Gérer la traduction automatique</code></td>
</tr>
</tbody>
</table>

**Note** : Les privilèges du site ne sont accordés à aucun rôle par défaut. Ils sont puissants et assez proches du statut de super-utilisateur. La plupart d’entre eux affectent tous les projets de votre installation Weblate.

### List of teams

Les équipes suivantes sont créées lors de l'installation (ou après l'exécution `setupgroups`) et vous êtes libre de les modifier. La migration les recrêera toutefois si vous les supprimez ou les renommez.

**Invités**

Définit les droits pour les utilisateurs non authentifiés.

This team only contains anonymous users (see `ANONYMOUS_USER_NAME`).

You can remove roles from this team to limit permissions for non-authenticated users.

Rôles par défaut : Ajouter une suggestion, Accès au dépôt

**Lecteurs**

This role ensures visibility of public projects for all users. By default, all users are members of this team.

Par défaut, automatic team assignment rend tous les nouveaux comptes membres de cette équipe lorsqu’ils la rejoignent.

Rôles par défaut : aucun

**Utilisateurs**

Default team for all users.

Par défaut, automatic team assignment rend tous les nouveaux comptes membres de cette équipe lorsqu’ils la rejoignent.

Rôles par défaut : Utilisateur expérimenté

**Réviseur**

Groupe pour les réviseurs (voir Flux de travail de traduction).

Rôles par défaut : Réviser les chaînes

**Gestionnaires**

Groupe pour les administrateurs.

Rôles par défaut : Administration

**Avertissement** : Never remove the predefined Weblate teams and users as this can lead to unexpected problems! If you have no use for them, you can removing all their privileges instead.
2.6.3 Restrictions d’accès supplémentaires

Si vous voulez utiliser votre installation de Weblate d’une manière moins publique, c’est à dire autoriser les nouveaux utilisateurs sur invitation uniquement, cela peut être fait en configurant Weblate de manière à ce que seuls les utilisateurs connus y aient accès. Pour ce faire, vous pouvez définir `REGISTRATION_OPEN` à `False` pour empêcher l’enregistrement de tout nouvel utilisateur, et définir `REQUIRE_LOGIN` à `/.*` pour exiger une connexion pour accéder à toutes les pages du site. C’est essentiellement la manière de verrouiller votre installation de Weblate.

**Indication** : Vous pouvez utiliser la fonction intégrée *Invitation de nouvel utilisateur* pour ajouter de nouveaux utilisateurs.

---

2.7 Projets de traduction

2.7.1 Organisation de traduction

Weblate organises translatable VCS content of project/components into a tree-like structure.

— The bottom level object is *Configuration du projet*, which should hold all translations belonging together (for example translation of an application in several versions and/or accompanying documentation).

— On the level above, *Configuration des composants*, which is actually the component to translate, you define the VCS repository to use, and the mask of files to translate.

— Above *Configuration des composants* there are individual translations, handled automatically by Weblate as translation files (which match *Masque de fichier* defined in *Configuration des composants*) appear in the VCS repository.

Weblate supports a wide range of translation formats (both bilingual and monolingual ones) supported by Translate Toolkit, see *Formats de fichiers pris en charge*.

**Note** : You can share cloned VCS repositories using *URLs internes de Weblate*. Using this feature is highly recommended when you have many components sharing the same VCS. It improves performance and decreases required disk space.

---

2.7.2 Adding translation projects and components

Modifié dans la version 3.2 : An interface for adding projects and components is included, and you no longer have to use *L’interface d’administration Django*.

Modifié dans la version 3.4 : The process of adding components is now multi staged, with automated discovery of most parameters.

Based on your permissions, new translation projects and components can be created. It is always permitted for users with the *Add new projects* permission, and if your instance uses billing (e.g. like [https://hosted.weblate.org/](https://hosted.weblate.org/) see *Facturation*), you can also create those based on your plans allowance from the user account that manages billing.

You can view your current billing plan on a separate page :
The project creation can be initiated from there, or using the menu in the navigation bar, filling in basic info about the translation project to complete addition of it:

**Project name**: WeblateOrg

**Display name**:

**URL slug**: weblateorg

**Project website**: https://weblate.org/

**Translation instructions**: https://weblate.org/contribute/

You can use Markdown and mention users by @username.

**Billing**: Weblate Test (Basic plan)

---

2.7. Projets de traduction
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:

**Depuis le contrôle de version**
- Creates component from remote version control repository.

**Depuis un composant existant**
- Creates additional component to existing one by choosing different files.

**Branche supplémentaire**
- Creates additional component to existing one, just for different branch.

**Téléverser les fichiers de traduction**
- 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 API REST de Weblate.

**Traduire le document**
- Upload single document or translation file and translate that.

**Démarrer de zéro**
- Create blank translation project and add strings manually.

Once you have existing translation components, you can also easily add new ones for additional files or branches using same repository.

First you need to fill in name and repository location:
On the next page, you are presented with a list of discovered translatable resources:

As a last step, you review the translation component info and fill in optional details:
Chapitre 2. Documentation pour l’administrateur
Voir aussi :
L’interface d’administration Django, Configuration du projet, Configuration des composants

2.7.3 Configuration du projet

Create a translation project and then add a new component for translation in it. The project is like a shelf, in which real translations are stacked. All components in the same project share suggestions and their dictionary; the translations are also automatically propagated through all components in a single project (unless turned off in the component configuration), see Mémoire de traduction.

Voir aussi :
/devel/integration

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

Nom du projet

Verbose project name, used to display the project name.

Voir aussi :

PROJECT_NAME_RESTRICT_RE

Abrégé de l’URL

Project name suitable for URLs.

Site Web du projet

URL where translators can find more info about the project.

This is a required parameter unless turned off by WEBSITE_REQUIRED.

Voir aussi :

PROJECT_WEB_RESTRICT_RE

Directives de traduction

Text describing localization process in the project, and any other information useful for translators. Markdown can be used for text formatting or inserting links.

Définir l’en-tête « Language-Team »

Whether Weblate should manage the Language-Team header (this is a GNU gettext only feature right now).
Utiliser un mémoire de traduction partagé

Whether to use shared translation memory, see Mémoire de traduction partagé for more details.

The default value can be changed by DEFAULT_SHARED_TM.

Contribute au mémoire de traduction partagé

Whether to contribute to shared translation memory, see Mémoire de traduction partagé for more details.

The default value can be changed by DEFAULT_SHARED_TM.

Contrôle d’accès

Configure per project access control, see Contrôle d’accès au projet for more details.

La valeur par défaut peut être modifiée avec DEFAULT_ACCESS_CONTROL.

Activer les révisions

Enable review workflow for translations, see Dedicated reviewers.

Activer la révision des chaînes sources

Enable review workflow for source strings, see Relectures des chaînes sources.

Voir aussi :
report-source, Commentaires

Activer les points d’ancrage

Whether unauthenticated Déclencheurs de notification are to be used for this repository.

Voir aussi :
Fichier de langue intermédiaire, Quality gateway for the source strings, Formats monolinguës et bilingues, Définitions de langue

Alias de langue

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

**Indication** : 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.

Voir aussi :
Adding new translations, Code langue, Parsing language codes
2.7.4 Configuration des composants

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.

Voir aussi : 
/devel/integration

You can find some examples of typical configurations in the Formats de fichiers pris en charge.

Note : 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 :

Nom du composant

Verbose component name, used to display the component name.

Identifiant du composant

Component name suitable for URLs.

Projet du composant

Configuration du projet where the component belongs.

Système de contrôle de version

VCS to use, see Intégration avec le système de contrôle de versions for details.

Voir aussi :

Pushing changes from Weblate

Dépôt du code source

VCS repository used to pull changes.

Voir aussi :

See Accessing repositories for more details on specifying URLs.

Indication : This can either be a real VCS URL or weblate://project/component indicating that the repository should be shared with another component. See URLs internes de Weblate for more details.
URL pour l’envoi du dépôt

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.

Voir aussi :

See Accessing repositories for more details on how to specify a repository URL and Pushing changes from Weblate for more details on pushing changes from Weblate.

Explorateur de dépôt

URL of repository browser used to display source files (location of used messages). When empty, no such links will be generated. You can use Balisage de modèle.

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 Balisage de modèle) : https://github.com/WeblateOrg/hello/blob/{{branch}}/{{filename|parentdir}}#L{{line}}

URL de dépôt exportée

URL where changes made by Weblate are exported. This is important when Traduction en continu is not used, or when there is a need to manually merge changes. You can use Exportateur Git to automate this for Git repositories.

Branche du dépôt

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.

Pousser la branche

Branch for pushing changes, leave empty to use Branche du dépôt.

For linked repositories, this is not used and setting from linked component applies.

Note : This is currently only supported for Git, GitLab and GitHub, it is ignored for other VCS integrations.

Voir aussi :

Pushing changes from Weblate
Masque de fichier

Mask of files to translate, including path. It should include one « * » replacing language code (see Définitions de langue for info on how this is processed). In case your repository contains more than one translation file (e.g. more gettext domains), you need to create a component for each of them.

For example po/*.po or locale/*/LC_MESSAGES/django.po.

In case your filename contains special characters such as [, ], these need to be escaped as [[]] or []].

Voir aussi :

Formats monolingues et bilingues, What does mean « There are more files for the single language (en) » ?

Fichier de langue de base mono-langue

Base file containing string definitions for Composants monolingues.

Voir aussi :

Formats monolingues et bilingues, What does mean « There are more files for the single language (en) » ?

Modifier le fichier de base

Whether to allow editing the base file for Composants monolingues.

Fichier de langue intermédiaire

Intermediate language file for Composants monolingues. In most cases this is a translation file provided by developers and is used when creating actual source strings.

When set, the source strings are based on this file, but all other languages are based on Fichier de langue de base mono-langue. In case the string is not translated into the source language, translating to other languages is prohibited. This provides Quality gateway for the source strings.

Voir aussi :

Quality gateway for the source strings, Formats monolingues et bilingues, What does mean « There are more files for the single language (en) » ?

Modèle pour les nouvelles traductions

Base file used to generate new translations, e.g. .pot file with gettext.

Indication : In many monolingual formats Weblate starts with empty file by default. Use this in case you want to have all strings present with empty value when creating new translation.

Voir aussi :

adding-translation, Adding new translations, Ajouter une nouvelle traduction, Formats monolingues et bilingues, What does mean « There are more files for the single language (en) » ?
Format de fichier

Translation file format, see also *Formats de fichiers pris en charge.*

Adresse pour signaler une anomalie de chaîne source

Email address used for reporting upstream bugs. This address will also receive notification about any source string comments made in Weblate.

Permettre la propagation de la traduction

You can turn off propagation of translations to this component from other components within same project. This really depends on what you are translating, sometimes it’s desirable to have make use of a translation more than once. It’s usually a good idea to turn this off for monolingual translations, unless you are using the same IDs across the whole project.

Default value can be changed by `DEFAULT_TRANSLATION_PROPAGATION`.

Voir aussi :

*Keeping translations same across components*

Autoriser les suggestions

Whether translation suggestions are accepted for this component.

Vote pour la suggestion

Turns on vote casting for suggestions, see *Vote pour la suggestion.*

Accepter automatiquement les suggestions

Automatically accept voted suggestions, see *Vote pour la suggestion.*

Drapeaux de traduction

Customization of quality checks and other Weblate behavior, see *Customizing behavior using flags.*

Vérifications forcées

List of checks which can not be ignored, see *Exécution des contrôles.*

Note : Enforcing the check does not automatically enable it, you still should enabled it using *Customizing behavior using flags* in *Drapeaux de traduction* or *Additional info on source strings.*
Licence associée à cette traduction

License of the translation (does not need to be the same as the source code license).

Accord de contribution

Accord que l'utilisateur doit approuver avant de pouvoir traduire ce composant.

Ajouter une nouvelle traduction

How to handle requests for creation of new languages. Available options:

Contacter les mainteneurs

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.

Pointer vers l'URL des directives de traduction

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).

Créer un fichier de nouvelle langue

User can select language and Weblate automatically creates the file for it and translation can begin.

Désactiver l'ajout de nouvelles traductions

There will be no option for user to start new translation.

Indication : The project admins can add new translations even if it is disabled here when it is possible (either Modèle pour les nouvelles traductions or the file format supports starting from an empty file).

Voir aussi :

adding-translation, Adding new translations

Gérer les chaînes

Nouveau dans la version 4.5.

Configures whether users in Weblate will be allowed to add new strings and remove existing ones. Adjust this to match your localization workflow - how the new strings are supposed to be introduced.

For bilingual formats, the strings are typically extracted from the source code (for example by using xgettext) and adding new strings in Weblate should be disabled (they would be discarded next time you update the translation files). In Weblate you can manage strings for every translation and it does not enforce the strings in all translations to be consistent.

For monolingual formats, the strings are managed only on source language and are automatically added or removed in the translations. The strings appear in the translation files once they are translated.

Voir aussi :

Formats monolingues et bilingues, adding-new-strings, POST /api/translations/(string:project)/(string:component)/(string:language)/units/
Style de code-langue

Personnaliser le code de langue utilisé pour générer le nom de fichier des traductions créées par Weblate.

Voir aussi :
- Adding new translations
- Code langue
- Parsing language codes

Style de fusion

Vous pouvez configurer comment les mises à jour du dépôt amont sont gérées. La mise en œuvre réelle dépend de
VCS, voir Intégration avec le système de contrôle de versions.

Refonder

Rebases Weblate commits on top of upstream repository on update. This provides clean history without extra
merge commits.

Rebasing can cause you trouble in case of complicated merges, so carefully consider whether or not you want
to enable them.

You might need to enable force pushing by choosing Git avec force push as Système de contrôle de version,
especially when pushing to a different branch.

Fusionner

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.

Fusionner sans « fast-forward »

Upstream repository changes are merged into Weblate one with doing a merge commit every time (even when
fast-forward would be possible). Every Weblate change will appear as a merge commit in Weblate repository.

Default value can be changed by DEFAULT_MERGE_STYLE.

Commit, add, delete, merge, add-on, and merge request messages

Message used when committing a translation, see Balisage de modèle.

Les valeurs par défaut peuvent être changées par DEFAULT_ADD_MESSAGE, DEFAULT_ADDON_MESSAGE,
DEFAULT_COMMIT_MESSAGE, DEFAULT_DELETE_MESSAGE, DEFAULT_MERGE_MESSAGE, DE-
FAULT_PULL_MESSAGE.

Pousser lors du commit

Whether committed changes should be automatically pushed to the upstream repository. When enabled, the push is
initiated once Weblate commits changes to its underlying repository (see Archivages lazy). To actually enable pushing
Repository push URL has to be configured as well.

Âge des modifications à committer

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.

Indication : There are other situations where pending changes might be committed, see Archivages lazy.
**Verrouiller en cas d’erreur**

Locks the component (and linked components, see *URLs internes de Weblate*) upon the first failed push or merge into its upstream repository, or pull from it. This avoids adding another conflicts, which would have to be resolved manually.

The component will be automatically unlocked once there are no repository errors left.

**Langue source**

Language used for source strings. Change this if you are translating from something else than English.

**Indication** : 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 *Fichier de langue intermédiaire*.

**Filtre sur la langue**

Regular expression used to filter the translation when scanning for file mask. It can be used to limit the list of languages managed by Weblate.

**Note** : You need to list language codes as they appear in the filename.

Some examples of filtering :

<table>
<thead>
<tr>
<th>Description du filtre</th>
<th>Expression rationnelle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selected languages only</td>
<td>^(cs</td>
</tr>
<tr>
<td>Exclure des langues</td>
<td>^([^!(lt</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>

**Expression rationnelle des variantes**

Regular expression used to determine the variants of a string, see variants.

**Note** : Most of the fields can be edited by project owners or administrators, in the Weblate interface.

**Voir aussi** :

*Does Weblate support other VCSes than Git and Mercurial?*, alerts
**Priorité**

Les composants prioritaires sont proposés en premier à la traduction.

Modifié dans la version 4.15 : This now also affects ordering of matched glossary terms.

**Accès restreint**

By default the component is visible to anybody who has access to the project, even if the person cannot perform any changes in the component. This makes it easier to keep translation consistency within the project.

Restricting access at a component, or component-list level takes over access permission to a component, regardless of project-level permissions. You will have to grant access to it explicitly. This can be done through granting access to a new user group and putting users in it, or using the default custom or private access control groups.

The default value can be changed in `DEFAULT_RESTRICTED_COMPONENT`.

**Indication** : This applies to project admins as well — please make sure you will not loose access to the component after toggling the status.

**Partager dans les projets**

You can choose additional projects where the component will be visible. Useful for shared libraries which you use in several projects.

**Note** : Sharing a component doesn’t change its access control. It only makes it visible when browsing other projects. Users still need access to the actual component to browse or translate it.

**Utiliser comme glossaire**

Nouveau dans la version 4.5.

Allows using this component as a glossary. You can configure how it will be listed using `Couleur du glossaire`.

The glossary will be accessible in all projects defined by `Partager dans les projets`.

It is recommended to enable `Gérer les chaînes` on glossaries in order to allow adding new words to them.

**Voir aussi** :

`Glossaire`

**Couleur du glossaire**

Display color for a glossary used when showing word matches.
2.7.5 Balisage de modèle

Weblate uses simple markup language in several places where text rendering is needed. It is based on Le langage de gabarit de Django, so it can be quite powerful.

Currently it is used in:

— Commit message formatting, see Configuration des composants
— Several add-ons
  — Découverte du composant
  — Générateur de statistiques
  — Exécution de scripts à partir du greffon

There following variables are available in the component templates:

```plaintext
{{ language_code }}  # Code langue
{{ language_name }}  # Nom de la langue
{{ component_name }}  # Nom du composant
{{ component_slug }}  # Identifiant du composant
{{ project_name }}    # Nom du projet
{{ project_slug }}    # Identifiant du projet
{{ url }}            # URL de traduction
{{ filename }}       # Nom du fichier de traduction
{{ stats }}          # Translation stats, this has further attributes, examples below.
  {{ stats.all }}     # Total strings count
  {{ stats.fuzzy }}   # Count of strings needing review
  {{ stats.fuzzy_percent }}  # Percent of strings needing review
  {{ stats.translated }}  # Nombre de chaînes traduites
  {{ stats.translated_percent }}  # Pourcentage des chaînes traduites
  {{ stats.allchecks }}  # Number of strings with failing checks
  {{ stats.allchecks_percent }}  # Percent of strings with failing checks
{{ author }}          # Author of current commit, available only in the commit scope.
{{ addon_name }}      # Name of currently executed add-on, available only in the add-on commit message.
```

The following variables are available in the repository browser or editor templates:

```plaintext
{{branch}}           # branche actuelle
{{line}}             # line in file
```
filename, you can also strip leading parts using the parentdir filter, for example `{{filename|parentdir}}`.

You can combine them with filters:

```plaintext
{{ component|title }}
```

You can use conditions:

```plaintext
{% if stats.translated_percent > 80 %}Well translated!{% endif %}
```

There is additional tag available for replacing characters:

```plaintext
{% replace component "-" " " %}
```

You can combine it with filters:

```plaintext
{% replace component|capfirst "-" " " %}
```

There are also additional filter to manipulate with filenames:

```plaintext
Directory of a file: `{{ filename|dirname }}`
File without extension: `{{ filename|stripext }}`
File in parent dir: `{{ filename|parentdir }}`
It can be used multiple times: `{{ filename|parentdir|parentdir }}`
```

…and other Django template features.

## 2.7.6 Vitesse d’importation

Fetching VCS repository and importing translations to Weblate can be a lengthy process, depending on size of your translations. Here are some tips:

**Optimiser la configuration**

The default configuration is useful for testing and debugging Weblate, while for a production setup, you should do some adjustments. Many of them have quite a big impact on performance. Please check *Configuration de production* for more details, especially:

- Configure Celery for executing background tasks (see *Background tasks using Celery*).
- Activer la mise en cache
- *Use a powerful database engine*
- Disable debug mode

**Check resource limits**

If you are importing huge translations or repositories, you might be hit by resource limitations of your server.

- Check the amount of free memory, having translation files cached by the operating system will greatly improve performance.
- Disk operations might be bottleneck if there is a lot of strings to process—the disk is pushed by both Weblate and the database.
- Additional CPU cores might help improve performance of background tasks (see *Background tasks using Celery*).
Disable unneeded checks

Some quality checks can be quite expensive, and if not needed, can save you some time during import if omitted. See `CHECK_LIST` for info on configuration.

### 2.7.7 Automatic creation of components

In case your project has dozen of translation files (e.g. for different gettext domains, or parts of Android apps), you might want to import them automatically. This can either be achieved from the command-line by using `import_project` or `import_json`, or by installing the `Découverte du composant` add-on.

To use the add-on, you first need to create a component for one translation file (choose the one that is the least likely to be renamed or removed in future), and install the add-on on this component.

For the management commands, you need to create a project which will contain all components and then run `import_project` or `import_json`.

Voir aussi :

Commandes de gestion, Découverte du composant

### 2.8 Définitions de langue

To present different translations properly, info about language name, text direction, plural definitions and language code is needed.

#### 2.8.1 Définitions de langue intégrées

Definitions for about 600 languages are included in Weblate and the list is extended in every release. Whenever Weblate is upgraded (more specifically whenever `weblate migrate` is executed, see *Generic upgrade instructions*) the database of languages is updated to include all language definitions shipped in Weblate.

This feature can be disable using `UPDATE_LANGUAGES`. You can also enforce updating the database to match Weblate built-in data using `setuplang`.

Voir aussi :

Extending built-in language definitions, Current language definitions

#### 2.8.2 Parsing language codes

While parsing translations, Weblate attempts to map language code (usually the ISO 639-1 one) from the `Masque de fichier` to any existing language object.

You can further adjust this mapping at project level by `Alias de langue`.

If no exact match can be found, an attempt will be made to best fit it into an existing language. Following steps are tried:

- Recherches insensibles à la casse.
- Normalisation des tirets bas et des tirets.
- Looking up built-in language aliases.
- Recherche par nom de langue.
- Ignoring the default country code for a given language—choosing `cs` instead of `cs_CZ`.

Should that also fail, a new language definition will be created using the defaults (left to right text direction, one plural). The automatically created language with code `xx_XX` will be named as `xx_XX (generated)`. You might want to change this in the admin interface later, (see *Changing language definitions*) and report it to the issue tracker (see *Contribuer à Weblate*), so that the proper definition can be added to the upcoming Weblate release.

2.8. Définitions de langue 265
Indication : In case you see something unwanted as a language, you might want to adjust *Filte sur la langue* to ignore such file when parsing translations.

Voir aussi :
*Code langue, Adding new translations*

2.8.3 Changing language definitions

You can change language definitions in the languages interface (\languages/ URL).

While editing, make sure all fields are correct (especially plurals and text direction), otherwise translators will be unable to properly edit those translations.

2.8.4 Code langue ambigus et macrolangues

In many cases it is not a good idea to use macrolanguage code for a translation. The typical problematic case might be Kurdish language, which might be written in Arabic or Latin script, depending on actual variant. To get correct behavior in Weblate, it is recommended to use individual language codes only and avoid macrolanguages.

Voir aussi :
*Macrolanguages definition, List of macrolanguages*

2.8.5 Définitions de langue

Each language consists of following fields :

**Code langue**

Code identifying the language. Weblate prefers two letter codes as defined by ISO 639-1, but uses ISO 639-2 or ISO 639-3 codes for languages that do not have two letter code. It can also support extended codes as defined by BCP 47.

Voir aussi :
*Parsing language codes, Adding new translations*

**Nom de la langue**

Visible name of the language. The language names included in Weblate are also being localized depending on user interface language.

**Orientation du texte**

Determines whether language is written right to left or left to right. This property is autodetected correctly for most of the languages.
Nombre pluriel

Number of plurals used in the language.

Forme plurielle

Gettext compatible plural formula used to determine which plural form is used for given count.

Voir aussi :

pluriels, GNU gettext utilitaires : Formes plurielles, Règles du pluriel en langue par le Consortium Unicode

Nombre de locuteurs

Number of worldwide speakers of this language.

2.8.6 Adding new translations

Modifié dans la version 2.18 : In versions prior to 2.18 the behaviour of adding new translations was file format specific.

Weblate can automatically start new translation for all of the file formats.

Some formats expect to start with an empty file and only translated strings to be included (for example Android string resources), while others expect to have all keys present (for example GNU gettext). The document-based formats (for example OpenDocument Format) start with a copy of the source document and all strings marked as needing editing. In some situations this really doesn’t depend on the format, but rather on the framework you use to handle the translation (for example with JSON files).

When you specify Modèle pour les nouvelles traductions in Configuration des composants, Weblate will use this file to start new translations. Any exiting translations will be removed from the file when doing so.

When Modèle pour les nouvelles traductions is empty and the file format supports it, an empty file is created where new strings will be added once they are translated.

The Style de code-langue allows you to customize language code used in generated filenames:

Par défaut, basé sur le format de fichier

Dependent on file format, for most of them POSIX is used.

Style POSIX utilisant le tiret du dessous en tant que séparateur

Typically used by gettext and related tools, produces language codes like pt_BR.

Style POSIX utilisant le tiret du bas comme séparateur et incluant le code pays

POSIX style language code including the country code even when not necessary (for example cs_CZ).

Style BCP utilisant un trait d’union comme séparateur

Typically used on web platforms, produces language codes like pt—BR.

Style BCP utilisant un trait d’union comme séparateur et incluant le code pays

BCP style language code including the country code even when not necessary (for example cs–CZ).

Style BCP utilisant un trait d’union comme séparateur, anciens codes de langue

Uses legacy codes for Chinese and BCP style notation.

Style BCP utilisant un trait d’union comme séparateur et en minuscules

BCP style notation, all in lower case (for example cs–cz).

Style des métadonnées de l’App Store

Style suitable for uploading metadata to Apple App Store.

Style des métadonnées Google Play

Style suitable for uploading metadata to Google Play Store.

Style Android

Only used in Android apps, produces language codes like pt-rBR.
**Style Linux**

Locales as used by Linux, uses legacy codes for Chinese and POSIX style notation. Additionally, any mappings defined in *Alias de langue* are applied in reverse.

**Note** : Weblate recognizes any of these when parsing translation files, the above settings only influences how new files are created.

**Voir aussi** :

*Code langue, Alias de langue, Parsing language codes*

### 2.9 Traduction en continu

There is infrastructure in place so that your translation closely follows development. This way translators can work on translations the entire time, instead of working through huge amount of new text just prior to release.

**Voir aussi** :

/devel/integration describes basic ways to integrate your development with Weblate.

This is the process :

1. Developers make changes and push them to the VCS repository.
2. Optionally the translation files are updated (this depends on the file format, see *Why does Weblate still show old translation strings when I've updated the template?*).
3. Weblate pulls changes from the VCS repository, see *Mise à jour des dépôts*.
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 *Archivages lazy*) and pushes them back if it has permissions to do so (see *Pushing changes from Weblate*).
2.9.1 Mise à jour des dépôts

You should set up some way of updating backend repositories from their source.

— Use Déclencheurs de notification to integrate with most of common code hosting services:
  — Automatically receiving changes from GitHub
  — Automatically receiving changes from GitLab
  — Automatically receiving changes from Bitbucket
  — Automatically receiving changes from Pagure
  — Automatically receiving changes from Azure Repos
  — Automatically receiving changes from Gitea Repos
— Manually trigger update either in the repository management or using API REST de Weblate or Client Weblate
— Enable AUTO_UPDATE to automatically update all components on your Weblate instance
— Execute `updategit` (with selection of project or --all to update all)

Whenever Weblate updates the repository, the post-update addons will be triggered, see Extensions.

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 API REST de Weblate 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:

```bash
# Lock Weblate translation
wlc lock
# Push changes from Weblate to upstream repository
wlc push
# Pull changes from upstream repository to your local copy
git pull
# Update translation files, this example is for Django
./manage.py makemessages --keep-pot -a
git commit -m 'Locale updates' -- locale
# Push changes to upstream repository
git push
# Tell Weblate to pull changes (not needed if Weblate follows your repo automatically)
wlc pull
# Unlock translations
wlc unlock
```

If you have multiple components sharing same repository, you need to lock them all separately:

```bash
wlc lock foo/bar
wlc lock foo/baz
wlc lock foo/baj
```

**Note:** The example uses Client 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 API REST de Weblate.

**Voir aussi:**

Client Weblate

**Automatically receiving changes from GitHub**

Weblate comes with native support for GitHub.

If you are using Hosted Weblate, the recommended approach is to install the Weblate app, that way you will get the correct setup without having to set much up. It can also be used for pushing changes back.

To receive notifications on every push to a GitHub repository, add the Weblate Webhook in the repository settings (Webhooks) as shown on the image below:
For the payload URL, append `/hooks/github/` to your Weblate URL, for example for the Hosted Weblate service, this is `https://hosted.weblate.org/hooks/github/`.

You can leave other values at default settings (Weblate can handle both content types and consumes just the `push` event).

Voir aussi :

`POST /hooks/github/`, `Accessing repositories from Hosted Weblate`

**Automatically receiving changes from Bitbucket**

Weblate has support for Bitbucket webhooks, add a webhook which triggers upon repository push, with destination to `/hooks/bitbucket/` URL on your Weblate installation (for example `https://hosted.weblate.org/hooks/bitbucket/`).
Voir aussi :

POST /hooks/bitbucket/, Accessing repositories from Hosted Weblate

Automatically receiving changes from GitLab

Weblate has support for GitLab hooks, add a project webhook with destination to /hooks/gitlab/ URL on your Weblate installation (for example https://hosted.weblate.org/hooks/gitlab/).

Voir aussi :

POST /hooks/gitlab/, Accessing repositories from Hosted Weblate

Automatically receiving changes from Pagure

Nouveau dans la version 3.3.

Weblate has support for Pagure hooks, add a webhook with destination to /hooks/pagure/ URL on your Weblate installation (for example https://hosted.weblate.org/hooks/pagure/). This can be done in Activate Web-hooks under Project options.
Voir aussi :

POST /hooks/pagure/, Accessing repositories from Hosted Weblate

Automatically receiving changes from Azure Repos

Nouveau dans la version 3.8.

Weblate has support for Azure Repos web hooks, add a webhook for Code pushed event with destination to /hooks/azure/ URL on your Weblate installation (for example https://hosted.weblate.org/hooks/azure/). This can be done in Service hooks under Project settings.

Voir aussi :

Web hooks in Azure DevOps manual, POST /hooks/azure/, Accessing repositories from Hosted Weblate
Automatically receiving changes from Gitea Repos

Nouveau dans la version 3.9.

Weblate has support for Gitea webhooks, add a Gitea Webhook for Push events event with destination to /hooks/gitea/ URL on your Weblate installation (for example https://hosted.weblate.org/hooks/gitea/). This can be done in Webhooks under repository Settings.

Voir aussi :
Webhooks in Gitea manual, POST /hooks/gitea/, Accessing repositories from Hosted Weblate

Automatically receiving changes from Gitee Repos

Nouveau dans la version 3.9.

Weblate has support for Gitee webhooks, add a WebHook for Push event with destination to /hooks/gitee/ URL on your Weblate installation (for example https://hosted.weblate.org/hooks/gitee/). This can be done in WebHooks under repository Management.

Voir aussi :
Webhooks in Gitee manual, POST /hooks/gitee/, Accessing repositories from Hosted Weblate

Automatically updating repositories nightly

Weblate automatically fetches remote repositories nightly to improve performance when merging changes later. You can optionally turn this into doing nightly merges as well, by enabling AUTO_UPDATE.

2.9.2 Pushing changes from Weblate

Each translation component can have a push URL set up (see URL pour l’envoi du dépôt), and in that case Weblate will be able to push change to the remote repository. Weblate can be also be configured to automatically push changes on every commit (this is default, see Pousser lors du commit). If you do not want changes to be pushed automatically, you can do that manually under Repository maintenance or using API via wlc push.

The push options differ based on the Intégration avec le système de contrôle de versions used, more details are found in that chapter.

In case you do not want direct pushes by Weblate, there is support for Requêtes de fusion GitHub, Requêtes de fusion GitLab, Gitea pull requests, Requêtes de fusion Pagure pull requests or Gerrit reviews, you can activate these by choosing GitHub, GitLab, Gitea, Gerrit or Pagure as Système de contrôle de version in Configuration des composants.

Overall, following options are available with Git, GitHub and GitLab :
<table>
<thead>
<tr>
<th>Configuration désirée</th>
<th>Système de contrôle de version</th>
<th>URL pour l’envoi du dépôt</th>
<th>Pousser la branche</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pas de push</td>
<td>Git</td>
<td>vide</td>
<td>vide</td>
</tr>
<tr>
<td>Pousser directement</td>
<td>Git</td>
<td>URL SSH</td>
<td>vide</td>
</tr>
<tr>
<td>Pousser dans une branche séparée</td>
<td>Git</td>
<td>URL SSH</td>
<td>Nom de la branche</td>
</tr>
<tr>
<td>GitHub pull request from fork</td>
<td>Requêtes de fusion GitHub</td>
<td>vide</td>
<td>vide</td>
</tr>
<tr>
<td>GitHub pull request from branch</td>
<td>Requêtes de fusion GitHub</td>
<td>SSH URL</td>
<td>Nom de la branche</td>
</tr>
<tr>
<td>GitLab merge request from fork</td>
<td>Requêtes de fusion GitLab</td>
<td>vide</td>
<td>vide</td>
</tr>
<tr>
<td>GitLab merge request from branch</td>
<td>Requêtes de fusion GitLab</td>
<td>SSH URL</td>
<td>Nom de la branche</td>
</tr>
<tr>
<td>Gitea merge request from fork</td>
<td>Gitea pull requests</td>
<td>vide</td>
<td>vide</td>
</tr>
<tr>
<td>Gitea merge request from branch</td>
<td>Gitea pull requests</td>
<td>SSH URL</td>
<td>Nom de la branche</td>
</tr>
<tr>
<td>Pagure merge request from fork</td>
<td>Requêtes de fusion Pagure</td>
<td>vide</td>
<td>vide</td>
</tr>
<tr>
<td>Pagure merge request from branch</td>
<td>Requêtes de fusion Pagure</td>
<td>SSH URL</td>
<td>Nom de la branche</td>
</tr>
</tbody>
</table>

**Note :** You can also enable automatic pushing of changes after Weblate commits, this can be done in **Pousser lors du commit**.

**Voir aussi :**

See *Accessing repositories* for setting up SSH keys, and *Archivages lazy* for info about when Weblate decides to commit changes.

**Branches protégées**

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 :

---

1. Can be empty in case *Dépôt du code source supports pushing.*
2.9.3 Interacting with others

Weblate makes it easy to interact with others using its API.

Voir aussi :

* API REST de Weblate

2.9.4 Archivages lazy

The behaviour of Weblate is to group commits from the same author into one commit if possible. This greatly reduces the number of commits, however you might need to explicitly tell it to do the commits in case you want to get the VCS repository in sync, e.g. for merge (this is by default allowed for the Managers group, see Liste des privilèges et des rôles intégrés).

The changes in this mode are committed once any of the following conditions are fulfilled:
- Somebody else changes an already changed string.
- A merge from upstream occurs.
- An explicit commit is requested.
- A file download is requested.
- Change is older than period defined as Âge des modifications à commiter on Configuration des composants.

**Indication**: Commits are created for every component. So in case you have many components you will still see lot of commits. You might utilize Squasher les commits 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. This can be done using Periodic Tasks in L’interface d’administration Django. First create desired Interval (for example 120 seconds). Then add new periodic task and choose weblate.trans.tasks.commit_pending as Task with {"hours": 0} as Keyword Arguments and desired interval.
2.9.5 Processing repository with scripts

The way to customize how Weblate interacts with the repository is Extensions. Consult Exécution de scripts à partir du greffon for info on how to execute external scripts through add-ons.

2.9.6 Keeping translations same across components

Once you have multiple translation components, you might want to ensure that the same strings have same translation. This can be achieved at several levels.

Propagation des traductions

With Permettre la propagation de la traduction enabled (what is the default, see Configuration des composants), all new translations are automatically done in all components with matching strings. Such translations are properly credited to currently translating user in all components.

Note: The translation propagation requires the key to be match for monolingual translation formats, so keep that in mind when creating translation keys.

Contrôle de cohérence

The Incohérence check fires whenever the strings are different. You can utilize this to review such differences manually and choose the right translation.

Traduction automatique

Automatic translation based on different components can be way to synchronize the translations across components. You can either trigger it manually (see Traduction automatique) or make it run automatically on repository update using add-on (see Traduction automatique).

2.10 Licence des traductions

You can specify which license translations are contributed under. This is especially important to do if translations are open to the public, to stipulate what they can be used for.

You should specify Configuration des composants license info. You should avoid requiring a contributor license agreement, though it is possible.

2.10.1 Informations sur la licence

Upon specifying license info (license name and URL), this info is shown in the translation info section of the respective Configuration des composants.

Usually this is best place to post licensing info if no explicit consent is required. If your project or translation is not libre you most probably need prior consent.
2.10.2 Accord de contribution

Si vous spécifiez un accord de licence de contributeur, seuls les utilisateurs qui l'ont accepté pourront contribuer. C'est une étape clairement visible lors de l'accès à la traduction :

La entrée du texte est mise en forme en paragraphes et les liens externes peuvent être inclus. Le markup HTML ne peut pas être utilisé.

2.10.3 Licences utilisateur

Tout utilisateur peut examiner toutes les licences de traduction de tous les projets publics sur l’instance depuis leur profil :

Les licences

Veuillez prendre note des informations de licence, car cela spécifie comment les traductions peuvent être utilisées.

En vous inscrivant, vous confirmez que vous utilisez votre nom et adresse e-mail dans les commits, et que vous fournissez votre contribution sous le licence définie par chaque projet de localisation.

Vous avez accepté les suivants en tant que contributeur :
- WeblateOrg/language names

Licences pour les traductions individuelles

GNU General Public License v3.0 ou ultérieur (Cette licence n'est pas applicable)
- WeblateOrg/WeblateOrg
- WeblateOrg/Django.js
- WeblateOrg/Django
- WeblateOrg/language names

MIT License (Cette licence n'est pas applicable)
- WeblateOrg/Android
2.11 Processus de traduction

2.11.1 Vote pour la suggestion

Everyone can add suggestions by default, to be accepted by signed in users. Suggestion voting can be used to make use of a string when more than one signed-in user agrees, by setting up the *Configuration des composants* with *Suggestion voting* to turn on voting, and *Autoaccept suggestions* to set a threshold for accepted suggestions (this includes a vote from the user making the suggestion if it is cast).

**Note:** Once automatic acceptance is set up, normal users lose the privilege to directly save translations or accept suggestions. This can be overridden with the *Edit string when suggestions are enforced* permission.

You can combine these with *access control* into one of the following setups:

- Users suggest and vote for suggestions and a limited group controls what is accepted. - Turn on voting. - Turn off automatic acceptance. - Don't let users save translations.
- Users suggest and vote for suggestions with automatic acceptance once the defined number of them agree. - Turn on voting. - Set the desired number of votes for automatic acceptance.
- Optional voting for suggestions. (Can optionally be used by users when they are unsure about a translation by making multiple suggestions.) - Only turn on voting.

2.11.2 Additional info on source strings

Enhance the translation process by adding additional info to the strings including explanations, string priorities, check flags and visual context. Some of that info may be extracted from the translation files and some may be added by editing the additional string info:
Access this directly from the translation interface by clicking the « Edit » icon next to Screenshot context or Flags.
2.11. Processus de traduction
**Priorité de chaînes**

Nouveau dans la version 2.0.

String priority can be changed to offer higher priority strings for translation earlier by using the `priority` flag.

**Indication** : This can be used to order the flow of translation in a logical manner.

**Voir aussi** :

*Quality checks*

**Drapeaux de traduction**

Nouveau dans la version 2.4.

Modifié dans la version 3.3 : Previously called *Quality checks flags*, it no longer configures only checks.

Customization of quality checks and other Weblate behavior, see *Customizing behavior using flags*.

The string flags are also inherited from the *Drapeaux de traduction* at *Configuration des composants* and flags from the translation file (see *Formats de fichiers pris en charge*).

**Voir aussi** :

*Quality checks, Customizing behavior using flags*

**Explication**

Modifié dans la version 4.1 : In previous versions this has been called *Extra context*.

Use the explanation to clarify scope or usage of the translation. You can use Markdown to include links and other markup.

**Visual context for strings**

Nouveau dans la version 2.9.

You can upload a screenshot showing a given source string in use within your program. This helps translators understand where it is used, and how it should be translated.

The uploaded screenshot is shown in the translation context sidebar :
In addition to Additional info on source strings, screenshots have a separate management interface under the Tools menu. Upload screenshots, assign them to source strings manually, or use optical character recognition to do so.

Once a screenshot is uploaded, this interface handles management and source string association:
Chapitre 2. Documentation pour l’administrateur
2.12 Contrôles de qualité et corrections

2.12.1 Personnaliser les réparations automatiques

Vous pouvez aussi implémenter vos propres réparations automatiques en plus de celles standards et les inclure dans \code{AUTOFIX_LIST}.

Les réparations automatiques sont puissantes, mais peuvent aussi causer des dommages ; soyez prudent quand vous en écrivez.

Par exemple, la réparation automatique suivante remplacerait chaque occurrence de la chaîne \texttt{foo} dans une traduction par \texttt{bar} :

```python
from django.utils.translation import gettext_lazy as _
from weblate.trans.autofixes.base import AutoFix

class ReplaceFooWithBar(AutoFix):
    """Replace foo with bar."""

    name = _("Foobar")

    def fix_single_target(self, target, source, unit):
        if "foo" in target:
            return target.replace("foo", "bar"), True
        return target, False
```

Pour installer des contrôles personnalisés, fournissez un chemin d'accès complet à la classe Python dans le fichier \code{AUTOFIX_LIST}, voir Custom quality checks, add-ons and auto-fixes.

2.12.2 Customizing behavior using flags

You can fine-tune the Weblate behavior by using flags. This can be done on the source string level (see Additional info on source strings), or in the Configuration des composants (Drapeaux de traduction). Some file formats also allow to specify flags directly in the format (see Formats de fichiers pris en charge).

Les drapeaux sont séparés par des virgules, les paramètres sont séparés par deux points. Vous pouvez utiliser des guillemets pour inclure des espaces ou des caractères spéciaux dans la chaîne de caractères. Par exemple :

```
placeholders:"special:value":"other value", regex:.*
```

Both single and double quotes are accepted, special characters are being escaped using backslash :

```
placeholders:"quoted \"string\":'single \'quoted\'
```

Voici une liste des drapeaux actuellement acceptés :

- **\texttt{rst-text}**
  Traiter le texte comme document reStructuredText, affecte Traduction inchangée.

- **\texttt{dos-eol}**
  Utilise les marqueurs de fin de ligne DOS au lieu de ceux d'Unix (\texttt{\r\n} au lieu de \texttt{\n}).

- **\texttt{read-only}**
  La chaîne est en lecture seule et ne devrait pas être modifiée dans Weblate, voir Chaînes en lecture seule.
priority:N
Priorité de la chaîne. Les chaînes de priorité plus élevée sont présentées en premier lieu pour la traduction. La priorité par défaut est de 100, plus une chaîne est prioritaire, plus elle est proposée tôt pour la traduction.

max-length:N
Limite la longueur maximale d’une chaîne à N caractères, voir Taille maximum de la traduction.

xml-text
Traitement du texte comme document XML, affecte Syntaxe XML et Balisage XML.

font-family:NAME
Définir la famille de polices pour les contrôles de rendu, voir Gestion des polices.

font-weight:WEIGHT
Définir l’ampleur des polices pour les contrôles de rendu, voir Gestion des polices.

font-size:SIZE
Définir la taille des polices pour les contrôles de rendu, voir Gestion des polices.

font-spacing:SPACING
Définir l’espacement des caractères pour les contrôles de rendu, voir Gestion des polices.

icu-flags:FLAGS
Définir des flags pour customiser le comportement de l’ICU MessageFormat qualité check.

icu-tag-prefix:PREFIX
Définir un préfixe requis pour les balises de type XML pour l’ICU MessageFormat qualité check.

placeholders:NAME:NAME2:...
Chaînes de caractères de remplacement attendues dans la traduction, voir Balises de remplacement.

replacements:FROM:TO:FROM2:TO2...
Remplacements à effectuer lors de la vérification des paramètres du texte résultant (par exemple dans Taille maximale de la traduction ou Taille maximum de la traduction). Le cas d’utilisation typique est l’expansion des caractères à placer pour s’assurer que le texte s’adapte même aux grandes valeurs, par exemple : replacements: %s : "John Doe".

variants:SOURCE
Marquage de cette chaîne en tant que variant de la chaîne avec la source correspondante. Voir variants.

regex:REGEX
Expression rationnelle correspondant à la traduction, voir Expression rationnelle.

forbidden
Indique une traduction interdite dans un glossaire, voir Forbidden translations.

strict-same
Faire en sorte que les « Traductions inchangées » évitent d’utiliser la liste noire de mots intégrés, voir Traduction inchangée.

check-glossary
Activation de la Non conforme au glossaire qualité check.

angularjs-format
Autorisation de l’Interpolation AngularJS qualité check.

c-format
Autorisation du Format C qualité check.

c-sharp-format
Autorisation du Format C# qualité check.

es-format
Autorisation du Modèle de littéraux ECMAScript qualité check.

i18next-interpolation
Autorisation de l’Interpolation i18next qualité check.

icu-message-format
Autorisation de l’ICU MessageFormat qualité check.

java_printf-format
Autorisation du Format Java qualité check.
java-format
Enable the MessageFormat Java quality check.

javascript-format
Enable the Format JavaScript quality check.

lua-format
Enable the Format Lua quality check.

object-pascal-format
Enable the Format Pascal objet quality check.

percent-placeholders
Enable the Balises de remplacement par caractères pour cent quality check.

perl-format
Enable the Format Perl quality check.

php-format
Enable the Format PHP quality check.

python-brace-format
Enable the Format d'accolade Python quality check.

python-format
Enable the Format Python quality check.

qt-format
Enable the Format Qt quality check.

qt-plural-format
Enable the Forme plurielle Qt quality check.

ruby-format
Enable the Format Ruby quality check.

scheme-format
Enable the Format Scheme quality check.

vue-format
Enable the Formatage Vue I18n quality check.

md-text
Treat text as a Markdown document. Enable Liens Markdown, Références Markdown, and Syntaxe Markdown quality checks.

case-insensitive
Adjust checks behavior to be case-insensitive. Currently affects only Balises de remplacement quality check.

safe-html
Enable the HTML non sůur quality check.

text
The string should consist of only a URL. Enable the URL quality check.

ignore-all-checks
Ignorer toutes les vérifications de qualité.

ignore-bbcode
Skip the Balisage BBCode quality check.

ignore-duplicate
Skip the Répétition de mots quality check.

ignore-check-glossary
Skip the Non conforme au glossoire quality check.

ignore-double-space
Skip the Double espace quality check.

ignore-angularjs-format
Skip the Chaîne d’interpolation AngularJS quality check.

ignore-c-format
Skip the Format C quality check.
ignore-c-sharp-format
   Skip the Format C# quality check.

ignore-es-format
   Skip the Modèle de littéraux ECMAScript quality check.

ignore-i18next-interpolation
   Skip the Interpolation i18next quality check.

ignore-icu-message-format
   Skip the ICU MessageFormat quality check.

ignore-java-format
   Skip the MessageFormat Java quality check.

ignore-java-printf-format
   Skip the Format Java quality check.

ignore-javascript-format
   Skip the Format JavaScript quality check.

ignore-lua-format
   Skip the Format Lua quality check.

ignore-object-pascal-format
   Skip the Format Pascal objet quality check.

ignore-percent-placeholders
   Skip the Balises de remplacement par caractères pour cent quality check.

ignore-perl-format
   Skip the Format Perl quality check.

ignore-php-format
   Skip the Format PHP quality check.

ignore-python-brace-format
   Skip the Format d’accolade Python quality check.

ignore-python-format
   Skip the Format Python quality check.

ignore-qt-format
   Skip the Format Qt quality check.

ignore-qt-plural-format
   Skip the Forme plurielle Qt quality check.

ignore-ruby-format
   Skip the Format Ruby quality check.

ignore-scheme-format
   Skip the Format Scheme quality check.

ignore-vue-format
   Skip the Formatage Vue I18n quality check.

ignore-translated
   Skip the A déjà été traduit quality check.

ignore-inconsistent
   Skip the Incohérence quality check.

ignore-kashida
   Skip the Présence d’un caractère kashida quality check.

ignore-md-link
   Skip the Liens Markdown quality check.

ignore-md-reflink
   Skip the Références Markdown quality check.

ignore-md-syntact
   Skip the Syntaxe Markdown quality check.

ignore-max-length
   Skip the Taille maximum de la traduction quality check.
ignore-max-size
Skip the Taille maximale de la traduction quality check.

ignore-escaped-newline
Skip the Pas de correspondance \n quality check.

ignore-end-colon
Skip the Incohérence de caractère deux-points quality check.

ignore-end-ellipsis
Skip the Incohérence de points de suspension quality check.

ignore-end-exclamation
Skip the Incohérence de point d’exclamation quality check.

ignore-end-stop
Skip the Incohérence de point final quality check.

ignore-end-question
Skip the Incohérence de point d’interrogation quality check.

ignore-end-semicolon
Skip the Incohérence de point-virgule quality check.

ignore-newline-count
Skip the Incohérence dans les sauts de ligne quality check.

ignore-plurals
Skip the Pluriels manquants quality check.

ignore-placeholders
Skip the Balises de remplacement quality check.

ignore-punctuation-spacing
Skip the Espacement de ponctuation quality check.

ignore-regex
Skip the Expression rationnelle quality check.

ignore-same-plurals
Skip the Pluriel identique quality check.

ignore-begin-newline
Skip the Nouvelle ligne au début quality check.

ignore-begin-space
Skip the Espaces au début quality check.

ignore-end-newline
Skip the Saut de ligne à la fin quality check.

ignore-end-space
Skip the Espace à la fin quality check.

ignore-same
Skip the Traduction inchangée quality check.

ignore-safe-html
Skip the HTML non sûr quality check.

ignore-url
Skip the URL quality check.

ignore-xml-tags
Skip the Balisage XML quality check.

ignore-xml-invalid
Skip the Syntaxe XML quality check.

ignore-zero-width-space
Skip the Espace sans chasse quality check.

ignore-ellipsis
Skip the Points de suspension quality check.

ignore-icu-message-format-syntax
Skip the Syntaxe ICU MessageFormat quality check.
**ignore-long-untranslated**  
Skip the *Ancienne chaîne non traduite* quality check.

**ignore-multiple-failures**  
Skip the *Plusieurs vérifications en échec* quality check.

**ignore-unnamed-format**  
Skip the *Multiples variables non nommées* quality check.

**ignore-optional-plural**  
Skip the *Non pluralisé* quality check.

---

**Note** : En général, la règle est nommée `ignore-*` pour tout contrôle, en utilisant son identifiant, de sorte que vous pouvez l’utiliser, même pour vos contrôles personnalisés.

Ces drapeaux sont compris à la fois dans les paramètres des *Configuration des composants*, par chaîne source et dans le fichier de traduction lui-même (par exemple dans GNU gettext).

---

### 2.12.3 Exécution des contrôles

Nouveau dans la version 3.11.

You can configure a list of checks which can not be ignored by setting *Vérifications forcées* in *Configuration des composants*. Each listed check can not be dismissed in the user interface and any string failing this check is marked as *Needs editing* (see *États de traduction*).

**Note** : Turning on check enforcing doesn’t enable it automatically. The check can be turned on by adding the corresponding flag to string or component flags.

**Voir aussi** :

*Additional info on source strings, Drapeaux de traduction* 

---

### 2.12.4 Gestion des polices

Nouveau dans la version 3.7.

**Indication** : Les polices téléchargées dans Webate sont utilisées uniquement pour la vérification *Taille maximale de la traduction*, elles n’ont pas d’effet sur l’interface utilisateur de Webate.

La vérification check-max-size’est utilisée pour calculer les dimensions du texte rendu, la police doit être chargée dans Webate et sélectionnée à l’aide d’une marque de traduction (voir :ref:`custom-checks`).

L’outil de gestion des polices Webate dans *Police de caractères* du menu *Gérer* de votre projet de traduction fournit une interface pour télécharger et gérer les polices. Vous pouvez téléverser des polices TrueType et OpenType, configurer des groupes de polices et les utiliser dans la vérification.

Les groupes de polices vous permettent de définir différentes polices pour différentes langues, ce qui est généralement nécessaire pour les langues non latines :
Les groupes de polices sont identifiés par leur nom, qui ne peut pas contenir d’espace ni de caractères spéciaux, afin qu’il puisse être facilement utilisé dans la définition du contrôle :
La famille de caractères et le style sont automatiquement reconnus après leur téléchargement :

Vous pouvez charger plusieurs polices dans Weblate :

<table>
<thead>
<tr>
<th>Group name</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>

Identifiez le nom de la polices que vous souhaitez utiliser dans Weblate.

Les polices sont utilisées à la place des polices par défaut lorsqu’elles sont disponibles.

Vous pouvez également utiliser des polices personnalisées.

-powered by Weblate 4.16-
Pour utiliser les polices afin de vérifier la longueur des chaînes, passez-lui les drapeaux appropriés (voir *Customizing behavior using flags*). Vous aurez probablement besoin des drapeaux suivants :

- **max-size:500**
  Définir la largeur maximale en pixels.
- **font-family:ubuntu**
  Définir le groupe de polices à utiliser en spécifiant son identifiant.
- **font-size:22**
  Définir la taille de la police en pixels.

### 2.12.5 Rédiger ses propres contrôles

A wide range of quality checks are built-in, (see *Quality checks*), though they might not cover everything you want to check. The list of performed checks can be adjusted using `CHECK_LIST`, and you can also add custom checks.

1. Subclass the `weblate.checks.Check`
2. Définir certains attributs.
3. Implement either the `check` (if you want to deal with plurals in your code) or the `check_single` method (which does it for you).

Quelques exemples :

To install custom checks, provide a fully-qualified path to the Python class in the `CHECK_LIST`, see Custom quality checks, add-ons and auto-fixes.

### Checking translation text does not contain « foo »

This is a pretty simple check which just checks whether the translation is missing the string « foo ».

```python
# Copyright © Michal Čihař <michal@weblate.org>
#
# SPDX-License-Identifier: GPL-3.0-or-later

"""Simple quality check example."""

from django.utils.translation import gettext_lazy as _
```

2.12. Contrôles de qualité et corrections 293
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

Checking that Czech translation text plurals differ

Vérifie à l’aide des infos de la langue que les deux formes plurielles en Tchèque ne sont pas les mêmes.

# Copyright © Michal Čihař <michal@weblate.org>
# # SPDX-License-Identifier: GPL-3.0-or-later
"""Quality check example for Czech plurals."""

from django.utils.translation import gettext_lazy as _

from weblate.checks.base import TargetCheck

class PluralCzechCheck(TargetCheck):
    # Used as identifier for check, should be unique
    # Has to be shorter than 50 characters
    check_id = "foo"

    # Short name used to display failing check
    name = _("Foo check")

    # Description for failing check
    description = _("Your translation is foo")

    # Real check code
    def check_target_unit(self, sources, targets, unit):
        if self.is_language(unit, ("cs",)):
            return targets[1] == targets[2]
        return False

    def check_single(self, source, target, unit):
        """We don't check target strings here."""
        return False
2.13 Configuring automatic suggestions

Modifié dans la version 4.13 : Prior to Weblate 4.13, the services were configured in the Configuration.

The support for several machine translation and translation memory services is built-in. Each service can be turned on by the administrator for whole site or at the project settings:
## Chapitre 2. Documentation pour l'administrateur

### Available automatic suggestion services

<table>
<thead>
<tr>
<th>Service</th>
<th>Install</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS</td>
<td></td>
</tr>
<tr>
<td>Amagama</td>
<td></td>
</tr>
<tr>
<td>Apertium APy</td>
<td></td>
</tr>
<tr>
<td>Baidu</td>
<td></td>
</tr>
<tr>
<td>DeepL</td>
<td></td>
</tr>
<tr>
<td>Glosbe</td>
<td></td>
</tr>
<tr>
<td>Google Translate</td>
<td></td>
</tr>
<tr>
<td>Google Translate API v3</td>
<td></td>
</tr>
<tr>
<td>IBM</td>
<td></td>
</tr>
<tr>
<td>LibreTranslate</td>
<td></td>
</tr>
<tr>
<td>Microsoft Terminology</td>
<td></td>
</tr>
<tr>
<td>Microsoft Translator</td>
<td></td>
</tr>
<tr>
<td>ModernMT</td>
<td></td>
</tr>
<tr>
<td>MyMemory</td>
<td></td>
</tr>
<tr>
<td>Netease Sight</td>
<td></td>
</tr>
<tr>
<td>SAP Translation Hub</td>
<td></td>
</tr>
<tr>
<td>Weblate</td>
<td></td>
</tr>
<tr>
<td>Weblate Translation Memory</td>
<td></td>
</tr>
<tr>
<td>Yandex</td>
<td></td>
</tr>
<tr>
<td>Youdao Zhiyun</td>
<td></td>
</tr>
<tr>
<td>tmserver</td>
<td></td>
</tr>
</tbody>
</table>

Some services will ask for additional configuration during installation.
Note: They come subject to their terms of use, so ensure you are allowed to use them how you want.

The services translate from the source language as configured at Configuration des composants, see Langue source.

Voir aussi:
Suggestions automatiques

2.13.1 Amagama

Service ID
amagama

Configuration
This service has no configuration.

Special installation of inserver run by the authors of Virtaal.

Voir aussi:
Installing amaGama, Amagama, amaGama Mémoire des traductions

2.13.2 Apertium APy

Service ID
apertium-apy

Configuration

<table>
<thead>
<tr>
<th>url</th>
<th>URL de l’API</th>
</tr>
</thead>
</table>

A libre software machine translation platform providing translations to a limited set of languages.

The recommended way to use Apertium is to run your own Apertium-APy server.

Voir aussi:
Apertium website, Apertium APy documentation

2.13.3 AWS

Nouveau dans la version 3.1.

Service ID
aws

Configuration

<table>
<thead>
<tr>
<th>key</th>
<th>ID de la clé d’accès</th>
</tr>
</thead>
<tbody>
<tr>
<td>secret</td>
<td>Clé secrète d’API</td>
</tr>
<tr>
<td>region</td>
<td>Nom de la région</td>
</tr>
</tbody>
</table>

Amazon Translate is a neural machine translation service for translating text to and from English across a breadth of supported languages.

Voir aussi:
Amazon Translate Documentation

2.13. Configuring automatic suggestions 297
2.13.4 Baidu

Nouveau dans la version 3.2.

**Service ID**

*baidu*

**Configuration**

<table>
<thead>
<tr>
<th>key</th>
<th>ID client</th>
</tr>
</thead>
<tbody>
<tr>
<td>secret</td>
<td>Secret client</td>
</tr>
</tbody>
</table>

Machine translation service provided by Baidu.

This service uses an API and you need to obtain an ID and API key from Baidu to use it.

**Voir aussi :**

Baidu Translate API

2.13.5 DeepL

Nouveau dans la version 2.20.

**Service ID**

*deepl*

**Configuration**

<table>
<thead>
<tr>
<th>url</th>
<th>URL de l'API</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>Clé d'API</td>
</tr>
</tbody>
</table>

DeepL is paid service providing good machine translation for a few languages. You need to purchase DeepL API subscription or you can use legacy DeepL Pro (classic) plan.

API URL to use with the DeepL service. At the time of writing, there is the v1 API as well as a free and a paid version of the v2 API.

**https://api.deepl.com/v2/ (default in Weblate)**

Is meant for API usage on the paid plan, and the subscription is usage-based.

**https://api-free.deepl.com/v2/**

Is meant for API usage on the free plan, and the subscription is usage-based.

**https://api.deepl.com/v1/**

Is meant for CAT tools and is usable with a per-user subscription.

Previously Weblate was classified as a CAT tool by DeepL, so it was supposed to use the v1 API, but now is supposed to use the v2 API. Therefore it defaults to v2, and you can change it to v1 in case you have an existing CAT subscription and want Weblate to use that.

The easiest way to find out which one to use is to open an URL like the following in your browser:


Replace the XXX with your auth_key. If you receive a JSON object which contains « Bonjour », you have the correct URL; if not, try the other three.

Weblate supports DeepL formality, it will choose matching one based on the language (for example, there is de@formal and de@informal).

**Voir aussi :**

DeepL website, DeepL pricing, DeepL API documentation
2.13.6 Glosbe

Service ID
glosbe

Configuration
This service has no configuration.

Free API dictionary and translation memory for almost every living language.
The API is gratis to use, but usage of the translations is subject to the license of the used data source. There is a limit of calls that may be done from one IP in a set period of time, to prevent abuse.

Voir aussi :
Glosbe website

2.13.7 Google Traduction

Service ID
google-translate

Configuration

<table>
<thead>
<tr>
<th>key</th>
<th>Clé d'API</th>
</tr>
</thead>
</table>

Machine translation service provided by Google.

This service uses the Google Translation API, and you need to obtain an API key and turn on billing in the Google API console.

Voir aussi :
Google translate documentation

2.13.8 Google Translate API v3

Service ID
google-translate-api-v3

Configuration

<table>
<thead>
<tr>
<th>credentials</th>
<th>Informations du compte pour le service Google Translate</th>
</tr>
</thead>
<tbody>
<tr>
<td>project</td>
<td>Projet Google Translate</td>
</tr>
<tr>
<td>location</td>
<td>Emplacement de Google Traduction</td>
</tr>
</tbody>
</table>

Machine translation service provided by Google Cloud services.

Voir aussi :
Google translate documentation, Authenticate to Cloud services using client libraries, Creating Google Translate project, Google Cloud App Engine locations
2.13.9 LibreTranslate

Nouveau dans la version 4.7.1.

Service ID
libretranslate

Configuration

<table>
<thead>
<tr>
<th>url</th>
<th>URL de l’API</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>Clé d’API</td>
</tr>
</tbody>
</table>

LibreTranslate is a free and open-source service for machine translations. The public instance requires an API key, but LibreTranslate can be self-hosted and there are several mirrors available to use the API for free.

https://libretranslate.com/ (official public instance)
Requires an API key to use outside of the website.

Voir aussi :
LibreTranslate website, LibreTranslate repository, LibreTranslate mirrors

2.13.10 Microsoft Terminology

Nouveau dans la version 2.19.

Service ID
microsoft-terminology

Configuration
This service has no configuration.

The Microsoft Terminology Service API allows you to programmatically access the terminology, definitions and user interface (UI) strings available in the Language Portal through a web service.

Voir aussi :
Microsoft Terminology Service API

2.13.11 Microsoft Traduction

Nouveau dans la version 2.10.

Service ID
microsoft-translator

Configuration
Machine translation service provided by Microsoft in Azure portal as a one of Cognitive Services.

Weblate implements Translator API V3.

**Translator Text API V2**

The key you use with Translator API V2 can be used with API 3.

**Translator Text API V3**

You need to register at Azure portal and use the key you obtain there. With new Azure keys, you also need to set region to locale of your service.

**Indication** : For Azure China, please use your endpoint from the Azure Portal.

**Voir aussi**:

Cognitive Services - Text Translation API, Microsoft Azure Portal, Base URLs, « Authenticating with a Multi-service resource » « Authenticating with an access token » section
2.13.12 ModernMT

Nouveau dans la version 4.2.

Service ID
modernmt

Configuration

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>URL de l’API</td>
</tr>
<tr>
<td>key</td>
<td>Clé d’API</td>
</tr>
</tbody>
</table>

Voir aussi :
ModernMT API,

2.13.13 MyMemory

Service ID
mymemory

Configuration

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>email</td>
<td>Courriel de contact</td>
</tr>
<tr>
<td>username</td>
<td>Nom d’utilisateur</td>
</tr>
<tr>
<td>key</td>
<td>Clé d’API</td>
</tr>
</tbody>
</table>

Huge translation memory with machine translation.
Free, anonymous usage is currently limited to 100 requests/day, or to 1000 requests/day when you provide a contact e-mail address in email. You can also ask them for more.

Voir aussi :
MyMemory website

2.13.14 NetEase Sight

Nouveau dans la version 3.3.

Service ID
netease-sight

Configuration

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>ID client</td>
</tr>
<tr>
<td>secret</td>
<td>Secret client</td>
</tr>
</tbody>
</table>

Service de traduction automatique fourni par NetEase.
This service uses an API, and you need to obtain key and secret from NetEase.

Voir aussi :
NetEase Sight Translation Platform
2.13.15 SAP Translation Hub

Service ID

sap-translation-hub

Configuration

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>url</strong></td>
<td>URL de l’API</td>
</tr>
<tr>
<td><strong>key</strong></td>
<td>Clé d’API</td>
</tr>
<tr>
<td><strong>user-name</strong></td>
<td>Nom d’utilisateur SAP</td>
</tr>
<tr>
<td><strong>password</strong></td>
<td>Mot de passe SAP</td>
</tr>
<tr>
<td><strong>enable</strong></td>
<td>Activer la traduction automatisée</td>
</tr>
<tr>
<td><strong>domain</strong></td>
<td>Domaine de la traduction</td>
</tr>
</tbody>
</table>

Machine translation service provided by SAP.

You need to have a SAP account (and the SAP Translation Hub enabled in the SAP Cloud Platform) to use this service.

You can also configure whether to also use machine translation services, in addition to the term database.

**Note**: To access the Sandbox API, you need to set url and key.

To access the productive API, you need to set url, username and password.

**Voir aussi**:

SAP Translation Hub API, Building the Base URL of SAP Translation Hub

2.13.16 tmserver

Service ID

tmserver

Configuration

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>url</strong></td>
<td>URL de l’API</td>
</tr>
</tbody>
</table>

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 anuGama server, which is an enhanced version of tmserver.

1. First you will want to import some data to the translation memory:

   build_tmdb -d /var/lib/tm/db -s en -t cs locale/cs/LC_MESSAGES/django.po
   build_tmdb -d /var/lib/tm/db -s en -t de locale/de/LC_MESSAGES/django.po
   build_tmdb -d /var/lib/tm/db -s en -t fr locale/fr/LC_MESSAGES/django.po

2. Start tmserver to listen to your requests:

   tmserver -d /var/lib/tm/db

3. Configure Weblate to talk to it, the default URL is http://localhost:8888/tmserver/.

2.13. Configuring automatic suggestions
Voir aussi :
tmserver Installing amaGama, Amagama, Amagama Translation Memory

### 2.13.17 IBM Watson Language Translator

**Service ID**
- ibm

**Configuration**

<table>
<thead>
<tr>
<th>url</th>
<th>URL de l’API</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>Clé d’API</td>
</tr>
</tbody>
</table>

IBM Watson Language Translator translates text from one language to another. The service offers multiple domain-specific models.

Voir aussi :
- Watson Language Translator, IBM Cloud API Docs

### 2.13.18 Weblate

**Service ID**
- weblate

**Configuration**

*This service has no configuration.*

Weblate machine translation service can provide translations for strings that are already translated inside Weblate. It looks for exact matches in the existing strings.

### 2.13.19 Weblate Translation Memory

Nouveau dans la version 2.20.

**Service ID**
- weblate-translation-memory

**Configuration**

*This service has no configuration.*

Use *Mémoire de traduction* as a machine translation service. Any string that has been translated in past (or uploaded to the translation memory) can be translated in this way.

### 2.13.20 Yandex

**Service ID**
- yandex

**Configuration**

| key | Clé d’API |

Machine translation service provided by Yandex.

This service uses a Translation API, and you need to obtain an API key from Yandex.

Voir aussi :
- Yandex Translate API, Powered by Yandex.Translate
2.13.21 Youdao Zhiyun

Nouveau dans la version 3.2.

Service ID

youdao-zhiyun

Configuration

<table>
<thead>
<tr>
<th>key</th>
<th>ID client</th>
</tr>
</thead>
<tbody>
<tr>
<td>secret</td>
<td>Secret client</td>
</tr>
</tbody>
</table>

Machine translation service provided by Youdao.

This service uses an API, and you need to obtain an ID and an API key from Youdao.

Voir aussi:

Youdao Zhiyun Natural Language Translation Service

2.13.22 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 © Michal Čihař <michal@weblate.org>
# SPDX-License-Identifier: GPL-3.0-or-later

/*! * 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: int = 75, ):
        """Return tuple with translations.""
        for t in dictionary.translate(text):
            yield {"text": t, "quality": 100, "service": self.name, "source": text}
```

You can list your own class in `WEBLATE_MACHINERY` and Weblate will start using that.
2.14 Extensions

Nouveau dans la version 2.19.

Les greffons permettent de personnaliser et d'automatiser le flux de traduction. Les administrateurs peuvent ajouter et gérer des greffons à partir du menu Gestion \ Greffons de chaque composant de traduction.

**Indication :** Vous pouvez également configurer les greffons en utilisant `API`, `DEFAULT_ADDONS`, ou `install_addon`. 

---

306 Chapitre 2. Documentation pour l'administrateur
### 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</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automatic translation</td>
<td>Automatically translates strings using machine translation or other components.</td>
<td><img src="#" alt="Install" /></td>
</tr>
<tr>
<td>Add missing languages</td>
<td>Ensures a consistent set of languages is used for all components within a project.</td>
<td><img src="#" alt="Install" /></td>
</tr>
<tr>
<td>Component discovery</td>
<td>Automatically adds or removes project components based on file changes in the version control system.</td>
<td><img src="#" alt="Install" /></td>
</tr>
<tr>
<td>Bulk edit</td>
<td>Bulk edit flags, labels, or states of strings.</td>
<td><img src="#" alt="Install" /></td>
</tr>
<tr>
<td>Statistics generator</td>
<td>Generates a file containing detailed info about the translation status.</td>
<td><img src="#" alt="Install" /></td>
</tr>
<tr>
<td>Prefill translation with source</td>
<td>Fills in translation strings with source string.</td>
<td><img src="#" alt="Install" /></td>
</tr>
<tr>
<td>Pseudolocale generation</td>
<td>Generates a translation by adding prefix and suffix to source strings automatically.</td>
<td><img src="#" alt="Install" /></td>
</tr>
<tr>
<td>Contributors in comment</td>
<td>Updates the comment part of the PO file header to include contributor names and years of contributions.</td>
<td><img src="#" alt="Install" /></td>
</tr>
<tr>
<td>Customize gettext output</td>
<td>Allows customization of gettext output behavior, for example line wrapping.</td>
<td><img src="#" alt="Install" /></td>
</tr>
<tr>
<td>Generate MO files</td>
<td>Automatically generates a MO file for every changed PO file.</td>
<td><img src="#" alt="Install" /></td>
</tr>
<tr>
<td>Update PO files to match POT (msgmerge)</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="#" alt="Install" /></td>
</tr>
<tr>
<td>Squash Git commits</td>
<td>Squash Git commits prior to pushing changes.</td>
<td><img src="#" alt="Install" /></td>
</tr>
<tr>
<td>Stale comment removal</td>
<td>Set a timeframe for removal of comments.</td>
<td><img src="#" alt="Install" /></td>
</tr>
<tr>
<td>Stale suggestion removal</td>
<td>Set a timeframe for removal of suggestions.</td>
<td><img src="#" alt="Install" /></td>
</tr>
</tbody>
</table>

Some add-ons will ask for additional configuration during installation.
2.14.1 Greffons intégrés

Traduction automatique

Nouveau dans la version 3.9.

ID du module
weblate.autotranslate.autotranslate

Configuration

<table>
<thead>
<tr>
<th>mode</th>
<th>Mode de traduction automatique</th>
<th>Choix disponibles:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>suggest – Ajouter comme suggestion</td>
</tr>
<tr>
<td></td>
<td></td>
<td>translate – Ajouter comme traduction</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fuzzy – Ajouter comme « À éditer »</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>filter_type</th>
<th>Filtre de recherche</th>
<th>Veuillez noter que traduire toutes les chaînes écrasera les traductions existantes.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Choix disponibles:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>all – Toutes les chaînes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>nottranslated – Chaînes non traduites</td>
</tr>
<tr>
<td></td>
<td></td>
<td>todo – Chaînes inachevées</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fuzzy – Chaînes marquées pour édition</td>
</tr>
<tr>
<td></td>
<td></td>
<td>check:inconsistent – Échec de la vérification : Incohérence</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>auto</th>
<th>Source des traductions automatiques</th>
<th>Choix disponibles:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>others – Autres composants de traduction</td>
</tr>
<tr>
<td></td>
<td></td>
<td>mt – Traduction automatique</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>components</th>
<th>Moteurs de traduction automatisée</th>
<th>Précisez l’identifiant du composant à utiliser comme source, laissez vide pour utiliser tous les composants du projet actuel.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>threshold</td>
<td>Seuil de score</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Déclencheurs

Mise à jour des composants, quotidienne

Traduit automatiquement les chaînes en utilisant des services de traduction automatique ou les traductions dans d’autres composants.

Il est déclenché :

— Quand de nouvelles chaînes apparaissent dans un composant.
— Une fois par mois pour chaque composant, ceci peut être configuré en utilisant background_tasks.

Voir aussi :

Traduction automatique, Keeping translations same across components
JavaScript localisation CDN

Nouveau dans la version 4.2.

**ID du module**
weblate.cdn.cdnjs

**Configuration**

<table>
<thead>
<tr>
<th></th>
<th>Seuil de traduction</th>
<th>Seuil d'inclusion des traductions.</th>
</tr>
</thead>
<tbody>
<tr>
<td>threshold</td>
<td></td>
<td></td>
</tr>
<tr>
<td>css_sel</td>
<td>Sélecteur CSS</td>
<td>Sélecteur CSS pour détecter les éléments traduisibles.</td>
</tr>
<tr>
<td>cookie_name</td>
<td>Nom du cookie de langue</td>
<td>Nom du cookie contenant la langue préférée.</td>
</tr>
<tr>
<td>files</td>
<td>Extraire des chaînes de caractères depuis des fichiers HTML</td>
<td>Liste des noms de fichiers dans le dépôt actuel ou des URL distantes à analyser pour les chaînes traduisibles.</td>
</tr>
</tbody>
</table>

**Déclencheurs**

quotidien, après le commit du dépôt, après la mise-à-jour du dépôt

Publie les traductions dans le réseau de diffusion de contenu (CDN) pour utilisation dans la localisation JavaScript ou HTML.

Peut être utilisé pour localiser des pages HTML statiques ou pour charger la localisation dans le code JavaScript.

Génère une URL unique pour votre composant que vous pouvez inclure dans les documents HTML pour les localiser. Voir weblate-cdn pour plus de détails.

**Voir aussi :**

cdn-addon-config, weblate-cdn, cdn-addon-extract, cdn-addon-html

Supprimer les chaînes constituées d’espaces

Nouveau dans la version 4.4.

**ID du module**
weblate.cleanup.blank

**Configuration**

*Ce module ne possède aucun paramètre.*

**Déclencheurs**

après le commit du dépôt, après la mise-à-jour du dépôt

Supprime les chaînes sans traduction des fichiers de traduction.

À utiliser pour ne pas avoir de chaînes vides dans les fichiers de traduction (par exemple si votre bibliothèque de localisation les affiche comme manquantes au lieu de revenir au texte d'origine).

**Voir aussi :**

*Does Weblate update translation files besides translations?*
Nettoyer les fichiers de traduction

ID du module
weblate.cleanup.generic

Configuration
Ce module ne possède aucun paramètre.

Déclencheurs
avant le commit du dépôt, après la mise-à-jour du dépôt

Mettre à jour tous les fichiers de traduction pour qu'ils correspondent au fichier mono-langue de base. Pour la plupart des formats de fichier, cela signifie supprimer les clés de traduction désuètes.

Voir aussi :
Does Weblate update translation files besides translations?

Ajouter les langues manquantes

ID du module
weblate.consistency.languages

Configuration
Ce module ne possède aucun paramètre.

Déclencheurs
quotidien, après l'ajout du dépôt

S'assure qu'un ensemble cohérent de langues est utilisé pour tous les composants d'un projet.

Les langues manquantes sont vérifiées toutes les 24 heures, et lorsque de nouvelles langues sont ajoutées dans Weblate. Contrairement à l'habitude, ce module s'applique à l'ensemble du projet.

Indication : Traduisez automatiquement les chaînes de caractères nouvellement ajoutées avec Traduction automa-

tique.

Découverte du composant

ID du module
weblate.discovery.discovery

Configuration
### Extensions

2.14. Extensions

Déclencheurs

mise à jour du dépôt de post

Ajoute ou supprime automatiquement des composants du projet en fonction des fichiers modifiés dans le système de contrôle de version.

Déclenchée à chaque mise à jour du système de contrôle des versions, et par ailleurs similaire à la commande de gestion `import_project`. De cette façon, vous pouvez suivre plusieurs composants de traduction dans un seul système de contrôle des versions.

La mise en correspondance se fait à l'aide d'expressions régulières permettant une configuration complexe, mais nécessitant certaines connaissances. Quelques exemples de cas d'utilisation courants sont disponibles dans la section d'aide du greffon.

Une fois que vous aurez cliqué sur Enregistrer, un aperçu des composants correspondants sera présenté, à partir duquel vous pourrez vérifier si la configuration correspond réellement à vos besoins :
312 Chapitre 2. Documentation pour l'administrateur
**Indication :** L’extension de la découverte des composants utilise les *URLs internes de Weblate*. C’est un moyen pratique pour partager la configuration du système de contrôle des versions entre plusieurs composants. Les composants liés utilisent le dépôt local du composant principal établit en remplissant le champ *Dépôt du code source* (dans *Gérer* → *Paramètres* → *Système de contrôle des versions*) avec la valeur `weblate://project/main-component` pour chaque composant respectif. Cela fait gagner du temps lors de la configuration et permet également d’économiser les ressources systèmes.

**Voir aussi :**

*Balisage de modèle*

**Modification en masse**

Nouveau dans la version 3.11.

**ID du module**

`weblate.flags.bulk`

**Configuration**

<table>
<thead>
<tr>
<th>q</th>
<th>Requête</th>
<th>Choix disponibles :</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>État à paramétrer</td>
<td>-1 – Ne pas modifier</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10 – À vérifier</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20 – Traduit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>30 – Approuvé</td>
</tr>
<tr>
<td>add_flags</td>
<td>Drapeaux de traduction à ajouter</td>
<td></td>
</tr>
<tr>
<td>re-move_flag</td>
<td>Drapeaux de traduction à supprimer</td>
<td></td>
</tr>
<tr>
<td>add_label</td>
<td>Libellés à ajouter</td>
<td></td>
</tr>
<tr>
<td>re-move_label</td>
<td>Libellés à supprimer</td>
<td></td>
</tr>
</tbody>
</table>

**Déclencheurs**

mise à jour de composant

Modifier en masse les drapeaux, les libellés ou les statuts des chaînes.

Automatise l’étiquetage en commençant la requête par `NOT has:label` et ajoute les étiquettes souhaitées jusqu’à ce que toutes les chaînes soient correctement étiquetées. Toute autre opération liée aux métadonnées Weblate peut également être automatisée.

**Exemples :**

**TABLEAU 5 – Étiqueter automatiquement les nouvelles chaînes de caractères**

<table>
<thead>
<tr>
<th>Requête de recherche</th>
<th>Libellés à ajouter</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>NOT has:label</code></td>
<td><code>récent</code></td>
</tr>
</tbody>
</table>

**TABLEAU 6 – Marquer toutes les chaines du journal des modifications de Fichiers de métadonnées de l’App Store en lecture seule**

<table>
<thead>
<tr>
<th>Requête de recherche</th>
<th>Drapeaux de traduction à ajouter</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>language:en AND key:changelogs/</code></td>
<td><code>read-only</code></td>
</tr>
</tbody>
</table>

**Voir aussi :**

*Modification en masse, Customizing behavior using flags, labels*
Marquer les traductions inchangées comme « À vérifier »

Nouveau dans la version 3.1.

ID du module
weblate.flags.same_edit

Configuration
Ce module ne possède aucun paramètre.

Déclencheurs
unité après la création

Chaque nouvelle chaîne de traduction importée depuis le système de contrôle des versions et identique à la chaîne source sera marquée comme « À vérifier » dans Weblate. Ce paramètre est particulièrement utile pour les formats de fichier contenant une copie des chaînes sources dans les chaînes à traduire.

Indication : Il se peut que vous vouliez également renforcer la vérification de Traduction inchangée en ajoutant l’indicateur strict-same à Drapeaux de traduction.

Voir aussi :
États de traduction

Marquer les nouvelles chaînes sources comme « À vérifier »

ID du module
weblate.flags.source_edit

Configuration
Ce module ne possède aucun paramètre.

Déclencheurs
unité après la création

Quand une nouvelle chaîne source est importée du système de contrôle de versions, elle est marquée dans Weblate comme « À vérifier ». Vous pouvez ainsi filtrer et modifier facilement les chaînes sources rédigées par les développeurs.

Voir aussi :
États de traduction

Marquer les nouvelles traductions comme « À vérifier »

ID du module
weblate.flags.target_edit

Configuration
Ce module ne possède aucun paramètre.

Déclencheurs
unité après la création

Lorsqu’une nouvelle chaîne à traduire est importée dans Weblate depuis le système de contrôle de versions, elle est marquée comme « À vérifier ». Vous pouvez ainsi filtrer et modifier facilement les traductions créées par les développeurs.

Voir aussi :
États de traduction
Générateur de statistiques

ID du module
weblate.generate.generate

Configuration

<table>
<thead>
<tr>
<th>filename</th>
<th>Nom du fichier généré</th>
</tr>
</thead>
<tbody>
<tr>
<td>template</td>
<td>Contenu du fichier généré</td>
</tr>
</tbody>
</table>

Déclencheurs
repository pre-commit

Génère un fichier avec des informations détaillées sur les statuts de la traduction.

Vous pouvez utiliser un modèle Django à la fois dans le nom de fichier et dans le contenu, voir *Balisage de modèle* pour une description détaillée des balises.

Par exemple, générer un fichier de résumé pour chaque traduction :

Nom du fichier généré
locale/{{ language_code }}.json

Contenu

```
{
  "language": "{{ language_code }}",
  "strings": "{{ stats.all }}",
  "translated": "{{ stats.translated }}",
  "last_changed": "{{ stats.last_changed }}",
  "last_author": "{{ stats.last_author }}",
}
```

Voir aussi :
*Balasge de modèle*

**Pré-remplir la traduction avec la source**

Nouveau dans la version 4.11.

ID du module
weblate.generate.prefill

Configuration

Ce module ne possède aucun paramètre.

Déclencheurs

Mise à jour des composants, quotidienne

Alimente les chaînes de traduction avec la chaîne source.

Toutes les chaînes non traduites dans le composant seront renseignées avec la chaîne source, et marquées comme devant être modifiées. Utilisez ceci lorsque vous ne pouvez pas avoir de chaînes vides dans les fichiers de traduction.

**Génération d'une pseudo-traduction**

Nouveau dans la version 4.5.

ID du module
weblate.generate.pseudolocale
Configuration

<table>
<thead>
<tr>
<th>source</th>
<th>Chaînes sources</th>
</tr>
</thead>
<tbody>
<tr>
<td>target</td>
<td>Traduction cible</td>
</tr>
<tr>
<td></td>
<td>Toutes les chaînes de cette traduction vont être remplacées</td>
</tr>
<tr>
<td>prefix</td>
<td>Préfixe de chaîne fixe</td>
</tr>
<tr>
<td>var_prefix</td>
<td>Préfixe de chaîne variable</td>
</tr>
<tr>
<td>suffix</td>
<td>Suffixe de chaîne fixe</td>
</tr>
<tr>
<td>var_suffix</td>
<td>Suffixe de chaîne variable</td>
</tr>
<tr>
<td>var_multiplier</td>
<td>Répétitions de la partie variable</td>
</tr>
<tr>
<td></td>
<td>Nombre de répétitions de la partie variable en fonction de la longueur de la chaîne source.</td>
</tr>
<tr>
<td>inclue_readonly</td>
<td>Inclure les chaînes en lecture seule</td>
</tr>
</tbody>
</table>

Déclencheurs
Mise à jour des composants, quotidienne

Génère une pseudo-traduction en ajoutant automatiquement un préfixe et un suffixe aux chaînes sources.

Les pseudo-traductions sont utiles pour trouver les chaînes de caractères qui ne sont pas préparées pour la traduction. Pour ce faire, toutes les chaînes sources sont modifiées afin de faciliter la détection des chaînes non-modifiées lors de l'exécution de l'application en utilisant la pseudo-traduction.

Il est également possible de trouver les chaînes de caractères dont les équivalents traduits pourraient ne pas correspondre à la mise en page.

L'utilisation de parties variables rend possible la recherche de chaînes de caractères qui pourrait ne pas tenir dans l'interface utilisateur après traduction - elle étend le texte en fonction de la taille de la chaîne source. Les parties variables sont répétées aussi longtemps que la longueur du texte multiplié par le coefficient. Par exemple, Hello world avec le suffixe variable _ et un coefficient de 1 devient Hello world__________ - le suffixe est répété une fois pour chaque caractère dans la chaîne source.

Les chaînes seront générées d’après le motif suivant :

*Fixed string prefix  Variable string prefix  Source string  Variable string suffix  Fixed string suffix*

**Indication** : Vous pouvez utiliser des langues réelles pour les tests, mais il existe des pseudo-traductions dédiées disponibles dans Weblate — *en_XA* et *ar_XB*.

**Indication** : Vous pouvez utiliser cette extension pour commencer une traduction vers une nouvelle locale d’un langage existant ou similaire. Une fois que vous avez ajouté la traduction au composant, suivez les instructions l’extension. *Exemple :* Si vous avez *fr* et vous voulez commencer une traduction en *fr_CA*, vous pouvez simplement définir *fr* comme une source, *fr_CA* comme une cible, et laisser vide le préfixe et le suffixe.

Désinstallez le greffon une fois la nouvelle traduction remplie pour empêcher Weblate de modifier les traductions effectuées après la copie.
Contributeurs dans le commentaire

**ID du module**
webate.gettext.authors

**Configuration**
Ce module ne possède aucun paramètre.

**Déclencheurs**
repository pre-commit

Ajoute le nom des contributeurs et les années de contribution dans le commentaire d’en-tête du fichier PO.

L’entête du fichier PO ressemble généralement à ceci :

```
# Pavel Borecki <pavel@example.com>, 2018, 2019.
# Filip Hron <filip@example.com>, 2018, 2019.
# anonymous <noreply@weblate.org>, 2019.
```

**Mettre à jour la variable ALL_LINGUAS dans le fichier « configure »**

**ID du module**
webate.gettext.configure

**Configuration**
Ce module ne possède aucun paramètre.

**Déclencheurs**
repository post-add, daily

Mettre à jour la variable ALL_LINGUAS dans les fichiers configure, configure.in ou configure.ac, lorsqu’une nouvelle traduction est ajoutée.

**Personnaliser la sortie gettext**

**ID du module**
webate.gettext.customize

**Configuration**

| width | Retour | Par défaut, gettext retourne à la ligne à 77 caractères et pour les nouvelles lignes. Avec le paramètre «–no-wrap», le retour à la ligne n’est réalisé que pour les nouvelles lignes. Choix disponibles:
| à la ligne | des lignes longues | 77 – Wrap lines at 77 characters and at newlines (xgettext default) | 65535 – Only wrap lines at newlines (like “xgettext –no-wrap”) | –1 – Aucun retour à la ligne |

**Déclencheurs**
stockage après chargement

Permet de personnaliser la sortie de gettext, par exemple le retour à la ligne automatique.

Il offre les options suivantes :

- Retour à la ligne à 77 caractères et pour les nouvelles lignes
- Retour à la ligne automatique uniquement pour les nouvelles lignes
- Aucun retour à la ligne automatique

**Note** : Par défaut, gettext saute à la ligne à 77 caractères et pour les nouvelles lignes. Avec le paramètre `--no-wrap` le saut de ligne n’a lieu que pour les nouvelles lignes.
Mettre à jour le fichier LINGUAS

ID du module
weblate.gettext.linguas

Configuration
Ce module ne possède aucun paramètre.

Déclencheurs
repository post-add, daily

Met à jour le fichier LINGUAS lors de l’ajout d’une nouvelle traduction.

Générer des fichiers MO

ID du module
weblate.gettext.mo

Configuration

| path | Chemin du fichier MO généré | Si non précisé, l'emplacement utilisé sera identique à celui du fichier PO. |

Déclencheurs
repository pre-commit

Génère automatiquement un fichier MO pour chaque fichier PO modifié.

L'emplacement du fichier MO généré peut être personnalisé et le champ pour cela utilise Balisage de modèle.

Mettre à jour les fichiers PO afin qu’ils correspondent au POT (msgmerge)

ID du module
weblate.gettext.msgmerge

Configuration

| previous | Conserver les msgids précédents des chaînes traduites |
| no_location | Supprimer l'emplacement des chaînes traduites |
| fuzzy | Utiliser les correspondances approximatives |

Déclencheurs
mise à jour du dépôt de post

Met à jour tous les fichiers PO (tels que configuré par Masque de fichier) pour correspondre au fichier POT (tel que configuré par Modèle pour les nouvelles traductions) en utilisant msgmerge.

Triggered whenever new changes are pulled from the upstream repository. Most msgmerge command-line options can be set up through the add-on configuration.

Voir aussi :

Does Weblate update translation files besides translations?
Squasher les commits Git

**ID du module**

weblate.git.squash

**Configuration**

<table>
<thead>
<tr>
<th>squash</th>
<th>Squash du commit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Choix disponibles :</td>
</tr>
<tr>
<td></td>
<td>all – Tous les archivages en un seul</td>
</tr>
<tr>
<td></td>
<td>language – Per language</td>
</tr>
<tr>
<td></td>
<td>file – Par fichier</td>
</tr>
<tr>
<td></td>
<td>author – Par auteur</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>append_trailers</th>
<th>Ajout de champs d'en-têtes aux messages de commits squashed</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>commit_message</th>
<th>Message de commit</th>
</tr>
</thead>
</table>
|                 | Ce message de commit sera utilisé à la place des messages de commit combinés des commits squashed.

**Déclencheurs**

repository post-commit

Squasher les commits Git avant de pousser les modifications.

Les commits de Git peuvent être écrasés avant de pousser les changements dans l'un des modes suivants :
- Tous les commits en un seul
- Par langue
- Par fichier
- Par auteur

Les messages d'archivage originaux sont préservés mais perdront la paternité de l'archivage, sauf si vous sélectionnez Par auteur ou si vous personnalisez le message d'archivage pour l'inclure.

Les messages d'archivage originaux peuvent être écrasés par un message d'archivage personnalisé.

Les Trailers (lignes d'archivage comme Co-authored-by : ...) peuvent éventuellement être supprimés des messages d'archivage originaux et ajoutés à la fin du message d'archivage écrasé. Ceci génère également un crédit Co-authored-by : pour chaque traducteur.

Personnaliser la sortie JSON

**ID du module**

weblate.json.customize

**Configuration**

<table>
<thead>
<tr>
<th>sort_keys</th>
<th>Trier les clés JSON</th>
</tr>
</thead>
<tbody>
<tr>
<td>indent</td>
<td>Indentation JSON</td>
</tr>
<tr>
<td>style</td>
<td>Style</td>
</tr>
<tr>
<td></td>
<td>JSON</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Déclencheurs**

stockage après chargement

Permet de paramétrer le comportement de la sortie de JSON, par exemple l'indentation ou le tri.
Formater le fichier de propriétés Java

ID du module
weblate.properties.sort

Configuration
Ce module ne possède aucun paramètre.

Déclencheurs
repository pre-commit

Formate et ordonne le fichier de propriétés Java.
— Consolidation des nouvelles lignes en celles d'Unix.
— Formatage en majuscules des séquences d'échappements d'Unicode (si elles sont présentes).
— Supprime les lignes vides et les commentaires.
— Trie les chaînes par les clés.
— Supprime les chaînes en double.

Purge automatique des commentaires

Nouveau dans la version 3.7.

ID du module
weblate.removal.comments

Configuration

<table>
<thead>
<tr>
<th>age</th>
<th>Nombre de jours à conserver</th>
</tr>
</thead>
</table>

Déclencheurs
quotidien

Définir le délai au bout duquel les commentaires sont supprimés.

Cela peut être utile pour supprimer d'anciens commentaires qui pourraient être devenus obsolètes. Utilisez-les avec précaution, car le fait qu'un commentaire devienne ancien ne signifie pas qu'il a perdu de son importance.

Purge automatique des suggestions

Nouveau dans la version 3.7.

ID du module
weblate.removal.suggestions

Configuration

<table>
<thead>
<tr>
<th>age</th>
<th>Nombre de jours à conserver</th>
</tr>
</thead>
</table>
| vote: Seuil de votes | Seuil pour la suppression d'une suggestion. Ce champ est sans effet lorsque le vote est désactivé.

Déclencheurs
quotidien

Définir le délai au bout duquel les suggestions sont supprimées.

Peut être très utile dans le cadre du vote par suggestion (voir Peer review) pour supprimer les suggestions qui ne reçoivent pas assez de votes positifs dans un délai donné.
Mettre à jour les fichiers RESX

Nouveau dans la version 3.9.

**ID du module**
weblate.resx.update

**Configuration**

*Ce module ne possède aucun paramètre.*

**Déclencheurs**
mise à jour du dépôt de post

Mettre à jour tous les fichiers de traduction pour qu’ils correspondent au fichier de base amont monolingue. Les chaînes inutilisées sont supprimées et de nouvelles chaînes sont ajoutées en copiant la chaîne source.

**Indication :** Utilisez *Nettoyer les fichiers de traduction* si vous voulez seulement supprimer les clés de traduction périmées.

**Voir aussi :**

*Does Weblate update translation files besides translations?*

Personnaliser la sortie XML

Nouveau dans la version 4.15.

**ID du module**
weblate.xml.customize

**Configuration**

<table>
<thead>
<tr>
<th>closing_tags</th>
<th>Inclure la balise de fermeture pour les balises XML vides</th>
</tr>
</thead>
</table>

**Déclencheurs**

stockage après chargement

Allows adjusting XML output behavior, for example closing tags instead of self- closing tags for empty tags.

Personnaliser la sortie YAML

Nouveau dans la version 3.10.2.

**ID du module**
weblate.yaml.customize

**Configuration**

<table>
<thead>
<tr>
<th>indent</th>
<th>Indentation YAML</th>
</tr>
</thead>
</table>
| width  | Retour à la ligne des lignes longues | Choix disponibles :
| 80     | Retourne à la ligne à 80 caractères |
| 100    | Retourne à la ligne à 100 caractères |
| 120    | Retourne à la ligne à 120 caractères |
| 180    | Retourne à la ligne à 180 caractères |
| 65535  | Aucun retour à la ligne |

<table>
<thead>
<tr>
<th>line_break</th>
<th>Retour à la ligne automatique</th>
</tr>
</thead>
<tbody>
<tr>
<td>dos</td>
<td>DOS (\r\n)</td>
</tr>
<tr>
<td>unix</td>
<td>UNIX (\n)</td>
</tr>
<tr>
<td>mac</td>
<td>MAC (\r)</td>
</tr>
</tbody>
</table>

**Déclencheurs**

stockage après chargement

Permet de paramétrer le comportement de la sortie YAML, par exemple la longueur des lignes ou le saut de ligne.
2.14.2 Personnalisation de la liste des greffons

The list of add-ons is configured by `WEBLATE_ADDONS`. To add another add-on, simply include the absolute class name in this setting.

2.14.3 Écrire un greffon

You can write your own add-ons too, create a subclass of `weblate.addons.base.BaseAddon` to define the add-on metadata, and then implement a callback to do the processing.

Voir aussi :

Développement de greffons

2.14.4 Exécution de scripts à partir du greffon

Les modules peuvent également être utilisés pour exécuter des scripts externes. Avant, cette fonction était intégrée dans Weblate, mais maintenant vous devez écrire du code pour envelopper votre script avec un module.

```python
# Copyright © Michal Čihař <michal@weblate.org>
#
# SPDX-License-Identifier: GPL-3.0-or-later

"""Example pre commit script."""

from django.utils.translation import gettext_lazy as _

from weblate.addons.events import EVENT_PRE_COMMIT
from weblate.addons.scripts import BaseScriptAddon

class ExamplePreAddon(BaseScriptAddon):
    # Event used to trigger the script
    events = (EVENT_PRE_COMMIT,)
    # Name of the add-on, 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"
```

Pour les instructions d’installation, consultez Custom quality checks, add-ons and auto-fixes.

Le script est exécuté avec le répertoire courant placé à la racine du dépôt de système de contrôle des versions pour un composant donné.

En outre, les variables d’environnement suivantes sont disponibles :

**WL_VCS**

Système de contrôle de version utilisé.

**WL_REPO**

URL du dépôt en amont.
WL_PATH
Chemin absolu vers le dépôt du système de contrôle des versions.

WL_BRANCH
Nouveau dans la version 2.11.
Branche du dépôt configurée dans le composant actuel.

WL_FILEMASK
Masque de fichier pour le composant actuel.

WL_TEMPLATE
Nom de fichier du modèle pour les traductions monolingues (peut être vide).

WL_NEW_BASE
Nom de fichier du fichier utilisé pour créer de nouvelles traductions (peut être vide).

WL_FILE_FORMAT
Format de fichier utilisé dans le composant actuel.

WL_LANGUAGE
Langue de la traduction actuellement traitée (non disponible pour les crochets de niveau composant).

WL_PREVIOUS_HEAD
HEAD précédent après la mise à jour (disponible uniquement après l'exécution du crochet post update).

WL_COMPONENT_SLUG
Nouveau dans la version 3.9.
Identifiant du composant utilisé pour construire l'URL.

WL_PROJECT_SLUG
Nouveau dans la version 3.9.
Identifiant du projet utilisé pour construire l'URL.

WL_COMPONENT_NAME
Nouveau dans la version 3.9.
Nom du composant.

WL_PROJECT_NAME
Nouveau dans la version 3.9.
Nom du projet.

WL_COMPONENT_URL
Nouveau dans la version 3.9.
URL du composant.

WL_ENGAGE_URL
Nouveau dans la version 3.9.
URL d'engagement du projet.

Voir aussi :
Configuration des composants
Post-update repository processing

Can be used to update translation files when the VCS upstream source changes. To achieve this, please remember Weblate only sees files committed to the VCS, so you need to commit changes as a part of the script.

Par exemple, avec Gulp, vous pouvez le faire en utilisant le code suivant :

```bash
#!/bin/sh
gulp --gulpfile gulp-i18n-extract.js
git commit -m 'Update source strings' src/languages/en.lang.json
```

Traitement des traductions avant archivage

Utilisez le script d’archivage pour apporter automatiquement des modifications à la traduction avant qu’elle soit archivée dans le dépôt.

Il est transmis en tant que paramètre unique composé du nom de fichier d’une traduction en cours.

2.15 Mémoire de traduction

Nouveau dans la version 2.20.

Weblate comes with a built-in translation memory consisting of the following :
- Manually imported translation memory (see Interface utilisateur).
- 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, Suggestions automatiques view while translating.
- Automatically, by translating strings using Traduction automatique, or Traduction automatique add-on.

For installation tips, see Weblate Translation Memory, which is turned on by default.

2.15.1 Translation memory scopes

Nouveau dans la version 3.2 : In earlier versions translation memory could be only loaded from a file corresponding to the current imported translation memory scope.

The translation memory scopes are there to allow both privacy and sharing of translations, to suit the desired behavior.

Imported translation memory

Importing arbitrary translation memory data using the import_memory command makes memory content available to all users and projects.

Per user translation memory

Stores all user translations automatically in the personal translation memory of each respective user.
Per project translation memory

All translations within a project are automatically stored in a project translation memory only available for this project.

Mémoire de traduction partagé

All translation within projects with shared translation memory turned on are stored in a shared translation memory available to all projects.

Please consider carefully whether to turn this feature on for shared Weblate installations, as it can have severe implications:

- The translations can be used by anybody else.
- This might lead to disclosing secret information.

2.15.2 Managing translation memory

Interface utilisateur

Nouveau dans la version 3.2.

In the basic user interface you can manage per user and per project translation memories. It can be used to download, wipe or import translation memory.

Indication: Translation memory in JSON can be imported into Weblate, TMX is provided for interoperability with other tools.

Voir aussi :

Schéma de mémoire des traductions Weblate

```
<table>
<thead>
<tr>
<th>Translation memory status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of your entries</td>
</tr>
<tr>
<td>Total number of entries</td>
</tr>
</tbody>
</table>
```

Powered by Weblate 4.3.6  About Weblate  Legal  Contact  Documentation  Donate to Weblate
Interface de gestion

There are several management commands to manipulate the translation memory content. These operate on the translation memory as whole, unfiltered by scopes (unless requested by parameters):

- **dump_memory**
  - Exports the memory into JSON

- **import_memory**
  - Imports TMX or JSON files into the translation memory

2.16 Configuration

Tous les paramètres sont stockés dans *settings.py* (comme habituellement avec Django).

Note : Après avoir changé l’un de ces paramètres, vous devez redémarrer Weblate - à la fois le processus WSGI que Celery.
S’il est exécuté en tant que *mod_wsgi*, vous devez redémarrer Apache pour recharger la configuration.

Voir aussi :

Veuillez également consulter la documentation de Django pour les paramètres configurant spécialement Django.

2.16.1 AKISMET_API_KEY

Weblate peut utiliser Akismet pour vérifier que les suggestions anonymes proposées ne soient pas des spams. Consultez [akismet.com](http://akismet.com) pour acheter une clé API et pour l’associer à un site.

2.16.2 ANONYMOUS_USER_NAME

Le nom des utilisateurs qui ne sont pas connectés.

Voir aussi :

*Contrôle d’accès*

2.16.3 AUDITLOG_EXPIRY

Nouveau dans la version 3.6.
Le nombre de jours durant lesquels Weblate conserve les journaux d’audit, qui contiennent des informations sur l’activité du compte.
La valeur par défaut est de 180 jours.
2.16.4 AUTH_LOCK_ATTEMPTS


Le nombre maximal de tentatives d'authentification échouées avant que la limitation de débit ne soit appliquée.

Ce paramètre est actuellement appliqué dans les emplacements suivants :

— Lors de la connexion. Supprime le mot de passe du compte, empêchant l’utilisateur de se connecter sans demander un nouveau mot de passe.
— Lors de la réinitialisation du mot de passe. Empêche l’envoi de nouveaux e-mails évitant ainsi de spammer les utilisateurs avec trop de tentatives de réinitialisation de mot de passe.

La valeur par défaut est 10.

Voir aussi :

Limite de requêtes

2.16.5 AUTO_UPDATE

Nouveau dans la version 3.2.

Modifié dans la version 3.11 : L’option originale on/off a été modifiée pour différencier les chaînes de caractères acceptées.

Met quotidiennement à jour tous les dépôts.

Indication : Utilise si vous n’utilisez pas les Déclencheurs de notification pour mettre automatiquement à jour les dépôts de Weblate.

Note : L’option on/off existe en plus de l'utilisation d'une chaîne de caractères pour la rétrocompatibilité.

Les options sont :

"none"                Pas de mises à jour quotidiennes.
"remote" et False     Met à jour uniquement les dépôts distants.
"full" et True        Met à jour les dépôts distants et fusionne la copie de travail.

Note : Ce paramètre exige que Background tasks using Celery fonctionne, et prendra effet après son redémarrage.

2.16.6 AVATAR_URL_PREFIX

Prefix for constructing avatar URLs as : ${AVATAR_URL_PREFIX}/avatar/${MAIL_HASH}?${PARAMS}. The following services are known to work :

Gravatar (default), as per https://gravatar.com/
   AVATAR_URL_PREFIX = 'https://www.gravatar.com/

Libravatar, as per https://www.libravatar.org/
   AVATAR_URL_PREFIX = 'https://www.libravatar.org/

Voir aussi :

Cache Avatar, ENABLE_AVATARS, Avatars
2.16.7 **AUTH_TOKEN_VALID**


How long the authentication token and temporary password from password reset e-mails is valid for. Set in number of seconds, defaulting to 172800 (2 days).

2.16.8 **AUTH_PASSWORD_DAYS**

Nouveau dans la version 2.15.

How many days will Weblate reject reusing previously used password for an user.

The checking is based on the audit log, `AUDITLOG_EXPIRY` needs to be at least same as this.

**Note** : Password changes made prior to Weblate 2.15 will not be accounted for in this policy.

La valeur par défaut est de 180 jours.

2.16.9 **AUTOFIX_LIST**

Liste des correctifs automatiques à appliquer lors de l'enregistrement d’une chaîne.

**Note** : Provide a fully-qualified path to the Python class that implementing the autofixer interface.

Corrections disponibles :

- `weblate.trans.autofixes.whitespace.SameBookendingWhitespace`
  Matches whitespace at the start and end of the string to the source.

- `weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis`
  Remplace les 3 points (…) par des points de suspension (…) si la chaîne source y recourt.

- `weblate.trans.autofixes.chars.RemoveZeroSpace`
  Removes zero-width space characters if the source does not contain any.

- `weblate.trans.autofixes.chars.RemoveControlChars`
  Removes control characters if the source does not contain any.

- `weblate.trans.autofixes.html.BleachHTML`
  Removes unsafe HTML markup from strings flagged as `safe-html` (see *HTML non sûr*).

You can select which ones to use :

```python
AUTOFIX_LIST = {
    "weblate.trans.autofixes.whitespace.SameBookendingWhitespace",
    "weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis",
}
```

**Voir aussi** :

*Automatic fixups, Personnaliser les réparations automatiques*
2.16.10 BACKGROUND_TASKS

Nouveau dans la version 4.5.2.

Defines how often lengthy maintenance tasks should be triggered for a component.

Right now this controls:

- Traduction automatique add-on
- Contrôles de qualité et corrections recalculation

Choix possibles:

- monthly (this is the default)
- weekly
- daily
- never

Note : Increasing the frequency is not recommended when Weblate contains thousands of components.

2.16.11 BASIC_LANGUAGES

Nouveau dans la version 4.4.

List of languages to offer users for starting new translation. When not specified built-in list is used which includes all commonly used languages, but without country specific variants.

This only limits non privileged users to add unwanted languages. The project admins are still presented with full selection of languages defined in Weblate.

Note : This does not define new languages for Weblate, it only filters existing ones in the database.

Exemple :

```
BASIC_LANGUAGES = \{"cs", "it", "ja", "en"\}
```

Voir aussi :

Définitions de langue

2.16.12 BORG_EXTRA_ARGS

Nouveau dans la version 4.9.

You can pass additional arguments to `borg create` when built-in backups are triggered.

Exemple :

```
BORG_EXTRA_ARGS = ["--exclude", "vcs/"]
```

Voir aussi :

Sauvegarder et déplacer Weblate, borg create
2.16.13 CACHE_DIR

Nouveau dans la version 4.16.
Directory where Weblate stores cache files. Defaults to cache subfolder in DATA_DIR.
Change this to local or temporary filesystem if DATA_DIR is on a network filesystem.
The Docker container uses a separate volume for this, see Docker container volumes.

2.16.14 CSP_SCRIPT_SRC, CSP_IMG_SRC, CSP_CONNECT_SRC, CSP_STYLE_SRC, CSP_FONT_SRC

Customize Content-Security-Policy header for Weblate. The header is automatically generated based on enabled integrations with third-party services (Matomo, Google Analytics, Sentry, …).
All these default to empty list.
Exemple :

```
# Enable Cloudflare Javascript optimizations
CSP_SCRIPT_SRC = ["ajax.cloudflare.com"]
```

Voir aussi :
Content security policy, Content Security Policy (CSP)

2.16.15 CHECK_LIST

Liste des vérifications de qualité à effectuer sur une traduction.

Note : Provide a fully-qualified path to the Python class implementing the check interface.

Adjust the list of checks to include ones relevant to you.
All built-in Quality checks are turned on by default, from where you can change these settings. By default they are commented out in Configuration d'exemple so that default values are used. New checks then carried out for each new Weblate version.

You can turn off all checks :

```
CHECK_LIST = ()
```

You can turn on only a few :

```
CHECK_LIST = ( 
    "weblate.checks.chars.BeginNewlineCheck",
    "weblate.checks.chars.EndNewlineCheck",
    "weblate.checks.chars.MaxLengthCheck",
)
```

Note : Changing this setting only affects newly changed translations, existing checks will still be stored in the database. To also apply changes to the stored translations, run updatechecks.

Voir aussi :
Quality checks, Customizing behavior using flags
2.16.16 COMMENT_CLEANUP_DAYS

Nouveau dans la version 3.6.
Delete comments after a given number of days. Defaults to `None`, meaning no deletion at all.

2.16.17 COMMIT_PENDING_HOURS

Nouveau dans la version 2.10.
Number of hours between committing pending changes by way of the background task.

Voir aussi :
* Configuration des composants, Âge des modifications à commiter, Running maintenance tasks, commit_pending

2.16.18 CONTACT_FORM

Nouveau dans la version 4.6.
Configures how e-mail from the contact form is being sent. Choose a configuration that matches your mail server configuration.

"reply-to"
The sender is used in as Reply-To, this is the default behaviour.

"from"
The sender is used in as From. Your mail server needs to allow sending such e-mails.

2.16.19 DATA_DIR

The folder Weblate stores all data in. It contains links to VCS repositories, a fulltext index and various configuration files for external tools.

The following subdirectories usually exist :

home  
Home directory used for invoking scripts.

ssh  
SSH keys and configuration.

static  
Default location for static Django files, specified by `STATIC_ROOT`. See *Serving static files*. The Docker container uses a separate volume for this, see *Docker container volumes*.

media  
Default location for Django media files, specified by `MEDIA_ROOT`. Contains uploaded screenshots, see *Visual context for strings*.

vcs  
Dépôts de contrôle de version pour les traductions.

backups  
Daily backup data, please check *Données supprimées pour les sauvegardes* for details.

fonts:
User-uploaded fonts, see *Gestion des polices*.

cache  
Various caches, can be placed elsewhere using `CACHE_DIR`. The Docker container uses a separate volume for this, see *Docker container volumes*.

Note: This directory has to be writable by Weblate. Running it as uWSGI means the `www-data` user should have write access to it.

The easiest way to achieve this is to make the user the owner of the directory :
Defaults to /home/weblate/data, but it is expected to be configured.

Voir aussi :

Permissions du système de fichiers, Sauvegarder et déplacer Weblate, CACHE_DIR

2.16.20 DATABASE_BACKUP

Nouveau dans la version 3.1.

Whether the database backups should be stored as plain text, compressed or skipped. The authorized values are :

- "plain"
- "compressed"
- "none"

Voir aussi :

Sauvegarder et déplacer Weblate

2.16.21 DEFAULT_ACCESS_CONTROL

Nouveau dans la version 3.3.

The default access control setting for new projects :

0  
   Public

1  
   Protégé

100  
   Privé

200  
   Personnalisé

Use Custom if you are managing ACL manually, which means not relying on the internal Weblate management.

Voir aussi :

Contrôle d'accès au projet, Contrôle d'accès

2.16.22 DEFAULT_AUTO_WATCH

Nouveau dans la version 4.5.

Configures whether Automatically watch projects on contribution should be turned on for new users. Defaults to True.

Voir aussi :

Notifications
2.16.23 DEFAULT_RESTRICTED_COMPONENT

Nouveau dans la version 4.1.
The default value for component restriction.
Voir aussi :
Accès restreint, Portée des équipes

2.16.24 DEFAULT_ADD_MESSAGE, DEFAULT_ADDON_MESSAGE, DEFAULT_COMMIT_MESSAGE, DEFAULT_DELETE_MESSAGE, DEFAULT_MERGE_MESSAGE

Default commit messages for different operations, please check Configuration des composants for details.
Voir aussi :
Balisage de modèle, Configuration des composants, Commit, add, delete, merge, add-on, and merge request messages

2.16.25 DEFAULT_ADDONS

Default add-ons to install on every created component.

Note : This setting affects only newly created components.

Exemple :

```yaml
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",
    },
}
```

Voir aussi :
install_addon, Extensions, WEBLATE_ADDONS

2.16.26 DEFAULT_COMMITER_EMAIL

Nouveau dans la version 2.4.
Committer e-mail address defaulting to noreply@weblate.org.
Voir aussi :
DEFAULT_COMMITER_NAME
2.16.27 DEFAULT_COMMITER_NAME

Nouveau dans la version 2.4.
Le nom de l’auteur par défaut est Weblate.
Voir aussi :

DEFAULT_COMMITER_EMAIL

2.16.28 DEFAULT_LANGUAGE

Nouveau dans la version 4.3.2.
Default source language to use for example in Langue source.
Defaults to en. The matching language object needs to exist in the database.
Voir aussi :

Définitions de langue, Langue source

2.16.29 DEFAULT_MERGE_STYLE

Nouveau dans la version 3.4.
Merge style for any new components.
— rebase - default
— merge
Voir aussi :

Configuration des composants, Style de fusion

2.16.30 DEFAULT_SHARED_TM

Nouveau dans la version 3.2.
Configures default value of Utiliser un mémoire de traduction partagé and Contribue au mémoire de traduction partagé.

2.16.31 DEFAULT_TRANSLATION_PROPAGATION

Nouveau dans la version 2.5.
Default setting for translation propagation, defaults to True.
Voir aussi :

Configuration des composants, Permettre la propagation de la traduction
2.16.32 DEFAULT_PULL_MESSAGE

Configures the default title and message for pull requests.

2.16.33 ENABLE_AVATARS

Whether to turn on Gravatar-based avatars for users. By default this is on.
Avatars are fetched and cached on the server, lowering the risk of leaking private info, speeding up the user experience.

Voir aussi:
Cache Avatar, AVATAR_URL_PREFIX, Avatars

2.16.34 ENABLE_HOOKS

Whether to enable anonymous remote hooks.

Voir aussi:
Déclencheurs de notification

2.16.35 ENABLE_HTTPS

Whether to send links to Weblate as HTTPS or HTTP. This setting affects sent e-mails and generated absolute URLs.
In the default configuration this is also used for several Django settings related to HTTPS - it enables secure cookies, toggles HSTS or enables redirection to HTTPS URL.
The HTTPS redirection might be problematic in some cases and you might hit issue with infinite redirection in case you are using a reverse proxy doing SSL termination which does not correctly pass protocol headers to Django. Please tweak your reverse proxy configuration to emit X-Forwards-Proto or Forwarded headers or configure SECURE_PROXY_SSL_HEADER to let Django correctly detect the SSL status.

Voir aussi:
SESSION_COOKIE_SECURE, CSRF_COOKIE_SECURE, SECURE_SSL_REDIRECT, SECURE_PROXY_SSL_HEADER Set correct site domain

2.16.36 ENABLE_SHARING

Turn on/off the Share menu so users can share translation progress on social networks.

2.16.37 EXTRA_HTML_HEAD

Nouveau dans la version 4.15.
Insert additional markup into HTML header. Can be used for verification of site ownership, for example:

```
EXTRA_HTML_HEAD = '<link href="https://fosstodon.org/@weblate" rel="me">'
```

Avertissement: No sanitization is performed on the string, it is inserted as is into the HTML header.
2.16.38 GET_HELP_URL

Nouveau dans la version 4.5.2.
URL where support for your Weblate instance can be found.

2.16.39 GITEA_CREDENTIALS

Nouveau dans la version 4.12.
List for credentials for Gitea servers.

```
GITEA_CREDENTIALS = {
    "try.gitea.io": {
        "username": "weblate",
        "token": "your-api-token",
    },
    "gitea.example.com": {
        "username": "weblate",
        "token": "another-api-token",
    },
}
```

Voir aussi :
*Gitea pull requests*, Creating a Gitea personal access token

2.16.40 GITLAB_CREDENTIALS

Nouveau dans la version 4.3.
List for credentials for GitLab servers.

```
GITLAB_CREDENTIALS = {
    "gitlab.com": {
        "username": "weblate",
        "token": "your-api-token",
    },
    "gitlab.example.com": {
        "username": "weblate",
        "token": "another-api-token",
    },
}
```

Voir aussi :
*Requêtes de fusion GitLab*, GitLab : Personal access token

2.16.41 GITHUB_CREDENTIALS

Nouveau dans la version 4.3.
List for credentials for GitHub servers.

```
GITHUB_CREDENTIALS = {
    "api.github.com": {
        "username": "weblate",
        "token": "your-api-token",
    },
    "github.example.com": {
}
```

(suite sur la page suivante)
Voir aussi :

Requêtes de fusion GitHub, Creating a GitHub personal access token

2.16.42 BITBUCKETSERVER_CREDENTIALS

Nouveau dans la version 4.16.
List for credentials for Bitbucket servers.

```json
BITBUCKETSERVER_CREDENTIALS = {
    "git.self-hosted.com": {
        "username": "weblate",
        "token": "http-access-token",
    },
}
```

Voir aussi :

Bitbucket Server pull requests, Bitbucket : HTTP access token

2.16.43 GOOGLE_ANALYTICS_ID

Google Analytics ID to turn on monitoring of Weblate using Google Analytics.

2.16.44 HIDE_REPO_CREDENTIALS

Hide repository credentials from the web interface. In case you have repository URL with user and password, Weblate will hide it when related info is shown to users.

For example instead of https://user:password@git.example.com/repo.git it will show just https://git.example.com/repo.git. It tries to clean up VCS error messages too in a similar manner.

Note : This is turned on by default.

2.16.45 HIDE_VERSION

Nouveau dans la version 4.3.1.
Hides version information from unauthenticated users. This also makes all documentation links point to latest version instead of the documentation matching currently installed version.

Hiding version is recommended security practice in some corporations, but it doesn’t prevent attacker to figure out version by probing the behavior.

Note : Cette fonction est désactivée par défaut.
2.16.46 INTERLEDGER_PAYMENT_POINTERS

Nouveau dans la version 4.12.1.
List of Interledger Payment Pointers (ILPs) for Web Monetization.
If multiple are specified, probabilistic revenue sharing is achieved by selecting one randomly.
Please check <https://webmonetization.org/> for more details.

**Indication** : The default value lets users fund Weblate itself.

2.16.47 IP_BEHIND_REVERSE_PROXY

Indicates whether Weblate is running behind a reverse proxy.
If set to True, Weblate gets IP address from a header defined by IP_PROXY_HEADER.

**Avertissement** : Ensure you are actually using a reverse proxy and that it sets this header, otherwise users will be able to fake the IP address.

**Note** : This is not on by default.

**Voir aussi**:
Running behind reverse proxy, Limite de requêtes, IP_PROXY_HEADER, IP_PROXY_OFFSET

2.16.48 IP_PROXY_HEADER

Indicates which header Weblate should obtain the IP address from when IP_BEHIND_REVERSE_PROXY is turned on.
Defaults to HTTP_X_FORWARDED_FOR.

**Voir aussi**:
Running behind reverse proxy, Limite de requêtes, SECURE_PROXY_SSL_HEADER, IP_BEHIND_REVERSE_PROXY, IP_PROXY_OFFSET

2.16.49 IP_PROXY_OFFSET

Indicates which part of IP_PROXY_HEADER is used as client IP address.
Depending on your setup, this header might consist of several IP addresses, (for example X-Forwarded-For: a, b, client-ip) and you can configure which address from the header is used as client IP address here.

**Avertissement** : Setting this affects the security of your installation, you should only configure it to use trusted proxies for determining IP address.
Defaults to 0.

Voir aussi:

* Running behind reverse proxy, Limite de requêtes, SECURE_PROXY_SSL_HEADER, IP_BEHIND_REVERSE_PROXY, IP_PROXY_HEADER

### 2.16.50 LEGAL_TOS_DATE

Nouveau dans la version 4.15.

**Note:** You need *Mentions légales* installed to make this work.

Date of last update of terms of service documents. Whenever the date changes, users are required to agree with the terms of service.

```python
from datetime import date
LEGAL_TOS_DATE = date(2022, 2, 2)
```

### 2.16.51 LEGAL_URL

Nouveau dans la version 3.5.

URL where your Weblate instance shows its legal documents.

**Indication:** Useful if you host your legal documents outside Weblate for embedding them inside Weblate, please check *Mentions légales* for details.

Exemple :

```python
LEGAL_URL = "https://weblate.org/terms/"
```

Voir aussi:

* PRIVACY_URL

### 2.16.52 LICENSE_EXTRA

Additional licenses to include in the license choices.

**Note:** Each license definition should be tuple of its short name, a long name and an URL.

Par exemple :

```python
LICENSE_EXTRA = [
    ("AGPL-3.0", "GNU Affero General Public License v3.0", 
    "https://www.gnu.org/licenses/agpl-3.0-standalone.html"),
]
```
2.16.53 LICENSE_FILTER

Modifié dans la version 4.3 : Setting this to blank value now disables license alert.
Filter list of licenses to show. This also disables the license alert when set to empty.

Note : This filter uses the short license names.

Par exemple :

```python
LICENSE_FILTER = {'AGPL-3.0', 'GPL-3.0-or-later'}
```

Following disables the license alert :

```python
LICENSE_FILTER = set()
```

Voir aussi :
alerts

2.16.54 LICENSE_REQUIRED

Defines whether the license attribute in Configuration des composants is required.

Note : This is off by default.

2.16.55 LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH

Whether the length of a given translation should be limited. The restriction is the length of the source string × 10 characters.

Indication : Set this to False to allow longer translations (up to 10,000 characters) irrespective of source string length.

Note : Defaults to True.

2.16.56 LOCALIZE_CDN_URL et LOCALIZE_CDN_PATH

These settings configure the JavaScript localisation CDN add-on. LOCALIZE_CDN_URL defines root URL where the localization CDN is available and LOCALIZE_CDN_PATH defines path where Weblate should store generated files which will be served at the LOCALIZE_CDN_URL.


Voir aussi :
JavaScript localisation CDN
2.16.57 LOGIN_REQUIRED_URLS

A list of URLs you want to require signing in. (Besides the standard rules built into Weblate).

**Indication** : This allows you to password protect a whole installation using:

```
LOGIN_REQUIRED_URLS = (r"/(.*)$",)
REST_FRAMEWORK["DEFAULT_PERMISSION_CLASSES"] = [
    "rest_framework.permissions.IsAuthenticated"
]
```

**Indication** : Il est également souhaitable de verrouiller l’accès à l’API, comme le montre l’exemple ci-dessus.

**Voir aussi** :

REQUIRE_LOGIN

2.16.58 LOGIN_REQUIRED_URLS_EXCEPTIONS

List of exceptions for LOGIN_REQUIRED_URLS. If not specified, users are allowed to access the sign in page.

Some of exceptions you might want to include:

```
LOGIN_REQUIRED_URLS_EXCEPTIONS = {
    r"/accounts/(.*)$", # Required for sign in
    r"/static/(.*)$", # Required for development mode
    r"/widgets/(.*)$", # Allowing public access to widgets
    r"/data/(.*)$", # Allowing public access to data exports
    r"/hooks/(.*)$", # Allowing public access to notification hooks
    r"/api/(.*)$", # Allowing access to API
    r"/js/i18n/$", # JavaScript localization
}
```

2.16.59 MATOMO_SITE_ID

ID of a site in Matomo (formerly Piwik) you want to track.

**Note** : This integration does not support the Matomo Tag Manager.

**Voir aussi** :

MATOMO_URL

2.16.60 MATOMO_URL

Full URL (including trailing slash) of a Matomo (formerly Piwik) installation you want to use to track Weblate use. Please check <https://matomo.org/> for more details.

**Indication** : This integration does not support the Matomo Tag Manager.

Par exemple:
**2.16.61 NEARBY_MESSAGES**

How many strings to show around the currently translated string. This is just a default value, users can adjust this in *Profil utilisateur*.

**2.16.62 DEFAULT_PAGE_LIMIT**

Nouveau dans la version 4.7.
Default number of elements to display when pagination is active.

**2.16.63 PAGURE_CREDENTIALS**

Nouveau dans la version 4.3.2.
List for credentials for Pagure servers.

```python
PAGURE_CREDENTIALS = {
    "pagure.io": {
        "username": "weblate",
        "token": "your-api-token",
    },
    "pagure.example.com": {
        "username": "weblate",
        "token": "another-api-token",
    },
}
```

**2.16.64 PRIVACY_URL**

Nouveau dans la version 4.8.1.
URL where your Weblate instance shows its privacy policy.

**Indication** : Useful if you host your legal documents outside Weblate for embedding them inside Weblate, please check *Mentions légales* for details.

**Exemple** :

```
PRIVACY_URL = "https://weblate.org/terms/"
```

**Voir aussi** :

*Requêtes de fusion Pagure*, Pagure API

*LEGAL_URL*
2.16.65 PRIVATE_COMMIT_EMAIL_OPT_IN

Nouveau dans la version 4.15.
Configures whether the private commit e-mail is opt-in or opt-out (by default it is opt-out).

Voir aussi :
Profil, PRIVATE_COMMIT_EMAIL_TEMPLATE

2.16.66 PRIVATE_COMMIT_EMAIL_TEMPLATE

Nouveau dans la version 4.15.
Template to generate private commit e-mail for an user. Defaults to "\{username\}@users.noreply.\{site_domain\}".
Set to blank string to disable.

Note : Using different commit e-mail is opt-in for users unless configured by PRIVATE_COMMIT_EMAIL_OPT_IN. Users can configure commit e-mail in the Profil.

2.16.67 PROJECT_BACKUP_KEEP_COUNT

Defines how many backups per project are kept on the server. It defaults to 3.

Voir aussi :
Project level backups

2.16.68 PROJECT_BACKUP_KEEP_DAYS

Defines how long the project backups will be kept on the server. Defaults to 30 days.

Voir aussi :
Project level backups

2.16.69 PROJECT_NAME_RESTRICT_RE

Nouveau dans la version 4.15.
Defines a regular expression to restrict project naming. Any matching names will be rejected.

Voir aussi :
Nom du projet
2.16.70 PROJECT_WEB_RESTRICT_RE

Nouveau dans la version 4.15.
Defines a regular expression to restrict project websites. Any matching URLs will be rejected.

Voir aussi :
Site Web du projet

2.16.71 RATELIMIT_ATTEMPTS

Nouveau dans la version 3.2.
Maximum number of authentication attempts before rate limiting is applied.
Defaults to 5.

Voir aussi :
Limite de requêtes, RATELIMIT_WINDOW, RATELIMIT_LOCKOUT

2.16.72 RATELIMIT_WINDOW

Nouveau dans la version 3.2.
How long authentication is accepted after rate limiting applies.
An amount of seconds defaulting to 300 (5 minutes).

Voir aussi :
Limite de requêtes, RATELIMIT_ATTEMPTS, RATELIMIT_LOCKOUT

2.16.73 RATELIMIT_LOCKOUT

Nouveau dans la version 3.2.
How long authentication is locked after rate limiting applies.
An amount of seconds defaulting to 600 (10 minutes).

Voir aussi :
Limite de requêtes, RATELIMIT_ATTEMPTS, RATELIMIT_WINDOW

2.16.74 REGISTRATION_ALLOW_BACKENDS

Nouveau dans la version 4.1.
List of authentication backends to allow registration from. This only limits new registrations, users can still authenticate and add authentication using all configured authentication backends.
It is recommended to keep REGISTRATION_OPEN enabled while limiting registration backends, otherwise users will be able to register, but Weblate will not show links to register in the user interface.
Exemple :

REGISTRATION_ALLOW_BACKENDS = ["azuread-oauth2", "azuread-tenant-oauth2"]
**Indication** : The backend names match names used in URL for authentication.

**Voir aussi** :

REGISTRATION_OPEN, Authentification

### 2.16.75 REGISTRATION_Captcha

A value of either `True` or `False` indicating whether registration of new accounts is protected by CAPTCHA. This setting is optional, and a default of `True` will be assumed if it is not supplied.

If turned on, a CAPTCHA is added to all pages where a user enters their e-mail address:
- New account registration.
- Récupération du mot de passe.
- Adding e-mail to an account.
- Contact form for users that are not signed in.

### 2.16.76 REGISTRATION_EMAIL_MATCH

Nouveau dans la version 2.17.

Allows you to filter which e-mail addresses can register.

Defaults to .* which allows any e-mail address to be registered.

You can use it to restrict registration to a single e-mail domain:

```
REGISTRATION_EMAIL_MATCH = r'^.*@weblate\.org$'
```

### 2.16.77 REGISTRATION_OPEN

Whether registration of new accounts is currently permitted. This optional setting can remain the default `True`, or changed to `False`.

This setting affects built-in authentication by e-mail address or through the Python Social Auth (you can whitelist certain back-ends using REGISTRATION_ALLOW_BACKENDS).

**Note** : If using third-party authentication methods such as S'authentifier avec LDAP, it just hides the registration form, but new users might still be able to sign in and create accounts.

**Voir aussi** :

REGISTRATION_ALLOW_BACKENDS, REGISTRATION_EMAIL_MATCH, Authentification

### 2.16.78 REGISTRATION_REBIND

Nouveau dans la version 4.16.

Allow rebinding authentication backends for existing users. Turn this on when migrating between authentication providers.

**Note** : Disabled by default to not allow adding other authentication backends to existing account. Rebinding can lead to account compromise when using more third-party authentication backends.
2.16.79 REPOSITORY_ALERT_THRESHOLD

Nouveau dans la version 4.0.2.
Threshold for triggering an alert for outdated repositories, or ones that contain too many changes. Defaults to 25.

Voir aussi :
alerts

2.16.80 REQUIRE_LOGIN

Nouveau dans la version 4.1.
This enables LOGIN_REQUIRED_URLS and configures REST framework to require authentication for all API endpoints.

Note : This is implemented in the Configuration d’exemple. For Docker, use WEBLATE_REQUIRE_LOGIN.

2.16.81 SENTRY_DSN

Nouveau dans la version 3.9.
Sentry DSN to use for Collecting error reports.

Voir aussi :
Django integration for Sentry

2.16.82 SESSION_COOKIE_AGE_AUTHENTICATED

Nouveau dans la version 4.3.
Set session expiry for authenticated users. This complements SESSION_COOKIE_AGE which is used for unauthenticated users.

Voir aussi :
SESSION_COOKIE_AGE

2.16.83 SIMPLIFY_LANGUAGES

Use simple language codes for default language/country combinations. For example an fr_FR translation will use the fr language code. This is usually the desired behavior, as it simplifies listing languages for these default combinations.

Turn this off if you want to different translations for each variant.
2.16.84 SITE_DOMAIN

Configures site domain. This is necessary to produce correct absolute links in many scopes (for example activation e-mails, notifications or RSS feeds).

In case Weblate is running on non-standard port, include it here as well.

Exemples :

```bash
# Production site with domain name
SITE_DOMAIN = "weblate.example.com"
# Local development with IP address and port
SITE_DOMAIN = "127.0.0.1:8000"
```

Note: This setting should only contain the domain name. For configuring protocol, (enabling and enforcing HTTPS) use ENABLE_HTTPS and for changing URL, use URL_PREFIX.

Indication: On a Docker container, the site domain is configured through WEBLATE_ALLOWED_HOSTS.

Voir aussi: Set correct site domain, Allowed hosts setup, Correctly configure HTTPS WEBLATE_SITE_DOMAIN, ENABLE_HTTPS

2.16.85 SITE_TITLE

Site title to be used for the website and sent e-mails.

2.16.86 SPECIAL_CHARS

Additional characters to include in the visual keyboard, Clavier visuel.

La valeur par défaut est :

```
SPECIAL_CHARS = ("\t", "\n", "\u00a0", "_")
```

2.16.87 SINGLE_PROJECT

Nouveau dans la version 3.8.

Redirects users directly to a project or component instead of showing the dashboard. You can either set it to True and in this case it only works in case there is actually only single project in Weblate. Alternatively set the project slug, and it will redirect unconditionally to this project.

Modifié dans la version 3.11: The setting now also accepts a project slug, to force displaying that single project.

Exemple :

```
SINGLE_PROJECT = "test"
```
2.16.88 SSH_EXTRA_ARGS

Nouveau dans la version 4.9.

Permet d'ajouter des paramètres personnalisés lors de la commande SSH. Cela est utile lors de la connexion à des serveurs utilisant des encodages non standard.

Par exemple, lors de la connexion à un serveur pour lequel Weblate ne peut pas négocier avec l'host legacy: no matching key exchange method found. Their offer: diffie-hellman-group1-sha1, vous pouvez l'activer en utilisant :

```
SSH_EXTRA_ARGS = "-oKexAlgorithms=+diffie-hellman-group1-sha1"
```

**Indication** : La chaîne est évaluée par le shell, assurez-vous de la formater correctement avec des espaces et des caractères spéciaux.

**Voir aussi** :

OpenSSH Legacy Options

2.16.89 STATUS_URL

Le URL où votre instance de Weblate se connecte pour publier son état.

2.16.90 SUGGESTION_CLEANUP_DAYS

Nouveau dans la version 3.2.1.

Supprime automatiquement les suggestions après un délai défini. Par défaut, None, ce qui signifie qu'aucune suppression ne se produira.

2.16.91 UPDATE_LANGUAGES

Nouveau dans la version 4.3.2.

Contrôle si les bases de données de langues doivent être mises à jour lors de la migration de la base de données et est activée par défaut. Ce paramètre n'affecte pas la commande `setuplang`.

**Avertissement** : Les affichages des langues pourraient être incorrects avec cela. Les définitions de langues Weblate s'étendent sur le temps et ne pourront pas afficher le code de langue pour les langues définies.

**Voir aussi** :

Définitions de langue intégrées

2.16.92 URL_PREFIX

Cette option vous permet de mettre en route Weblate sous un chemin différent (sinon, elle se dépend de la configuration de votre serveur web). Note :

Pour utiliser cette option, vous devez également configurer votre serveur pour supprimer ce préfixe. Par exemple, avec WSGI, cela peut être réalisé en définissant `WSGIScriptAlias`.

**Indication** : Le préfixe doit commencer par un `/`.

**Exemple** :
URL_PREFIX = "/translations"

**Note:** This setting does not work with Django’s built-in server, you would have to adjust urls.py to contain this prefix.

### 2.16.93 VCS_API_DELAY

Nouveau dans la version 4.15.1.

Configures minimal delay in seconds between third-party API calls in Requêtes de fusion GitHub, Requêtes de fusion GitLab, Gitea pull requests, and Requêtes de fusion Pagure.

This rate-limits API calls from Weblate to these services to avoid overloading them.

If you are being limited by secondary rate limiter at GitHub, increasing this might help.

La valeur par défaut est 10.

### 2.16.94 VCS_BACKENDS

Configuration of available VCS backends.

**Note:** Weblate tries to use all supported back-ends you have the tools for.

**Indication:** You can limit choices or add custom VCS back-ends by using this.

VCS_BACKENDS = ("weblate.vcs.git.GitRepository",)

**Voir aussi:**

* Intégration avec le système de contrôle de versions

### 2.16.95 VCS_CLONE_DEPTH

Nouveau dans la version 3.10.2.

Configures how deep cloning of repositories Weblate should do.

**Note:** Currently this is only supported in Git. By default Weblate does shallow clones of the repositories to make cloning faster and save disk space. Depending on your usage (for example when using custom Extensions), you might want to increase the depth or turn off shallow clones completely by setting this to 0.

**Indication:** In case you get fatal: protocol error: expected old/new/ref, got 'shallow <commit hash>' error when pushing from Weblate, turn off shallow clones completely by setting :

VCS_CLONE_DEPTH = 0
2.16.96 WEBLATE_ADDONS

List of add-ons available for use. To use them, they have to be enabled for a given translation component. By default this includes all built-in add-ons, when extending the list you will probably want to keep existing ones enabled, for example:

```python
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.xml.XMLCustomizeAddon",
    "weblate.addons.properties.PropertiesSortAddon",
    "weblate.addons.git.GitSquashAddon",
    "weblate.addons.removal.RemoveComments",
    "weblate.addons.removal.RemoveSuggestions",
    "weblate.addons.resx.ResxUpdateAddon",
    "weblate.addons.autotranslate.AutoTranslateAddon",
    "weblate.addons.yaml.YAMLCustomizeAddon",
    "weblate.addons.cdn.CDNJSAddon",
    # Add-on you want to include
    "weblate.addons.example.ExampleAddon",
)
```

**Note:** Removing the add-on from the list does not uninstall it from the components. Weblate will crash in that case. Please uninstall add-on from all components prior to removing it from this list.

**Voir aussi:**

*Extensions, DEFAULT_ADDONS*

2.16.97 WEBLATE_EXPORTERS

Nouveau dans la version 4.2.

List of available exporters offering downloading translations or glossaries in various file formats.

**Voir aussi:**

*Formats de fichiers pris en charge*
2.16.98 WEBLATE_FORMATS

Nouveau dans la version 3.0.
List of file formats available for use.

Note : The default list already has the common formats.

Voir aussi :
Formats de fichiers pris en charge

2.16.99 WEBLATE_MACHINERY

Nouveau dans la version 4.13.
List of machinery services available for use.

Voir aussi :
Configuring automatic suggestions

2.16.100 WEBLATE_GPG_IDENTITY

Nouveau dans la version 3.1.
Identity used by Weblate to sign Git commits, for example :

WEBLATE_GPG_IDENTITY = "Weblate <weblate@example.com>"

The Weblate GPG keyring is searched for a matching key (home/.gnupg under DATA_DIR). If not found, a key is generated, please check Signing Git commits with GnuPG for more details.

Voir aussi :
Signing Git commits with GnuPG

2.16.101 WEBSITE_REQUIRED

Defines whether Site Web du projet has to be specified when creating a project. Turned on by default as that suits public server setups.

2.17 Configuration d’exemple

The following example is shipped as weblate/settings_example.py with Weblate :

```python
# Copyright © Michal Čihař <michal@weblate.org>
#
# SPDX-License-Identifier: GPL-3.0-or-later

import os
import platform
from logging.handlers import SysLogHandler

# Title of site to use
SITE_TITLE = "Weblate"
```

(suite sur la page suivante)
# Site domain
SITE_DOMAIN = ""

# Whether site uses https
ENABLE_HTTPS = False

# Django settings for Weblate project.

DEBUG = True

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

MANAGERS = ADMINS

DATABASES = {
    "default": {
        # Use "postgresql" or "mysql".
        "ENGINE": "django.db.backends.postgresql",
        # Database name.
        "NAME": "weblate",
        # Database user.
        "USER": "weblate",
        # Name of role to alter to set parameters in PostgreSQL,
        # use in case role name is different than user used for authentication.
        # "ALTER_ROLE": "weblate",
        # Database password.
        "PASSWORD": "",
        # Set to empty string for localhost.
        "HOST": "127.0.0.1",
        # Set to empty string for default.
        "PORT": "",
        # Customizations for databases.
        "OPTIONS": {
            # In case of using an older MySQL server,
            # which has MyISAM as a default storage
            # "init_command": "SET storage_engine=INNODB",
            # Uncomment for MySQL older than 5.7:
            # "init_command": "SET sql_mode='STRICT_TRANS_TABLES'",
            # Set emoji capable charset for MySQL:
            # "charset": "utf8mb4",
            # Change connection timeout in case you get MySQL gone away error:
            # "connect_timeout": 28800,
        },
        # Persistent connections
        "CONN_MAX_AGE": 0,
        # Disable server-side cursors, might be needed with pgbouncer
        "DISABLE_SERVER_SIDE_CURSORS": False,
    },
}

# Data directory, you can use following for the development purposes:
# os.path.join(os.path.dirname(__file__)), "data"
DATA_DIR = "/home/weblate/data"
CACHE_DIR = "/home/weblate/cache"

# 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", "Benaruxska"),
    ("bg", "Bългарски"),
    ("br", "Brezhoneg"),
    ("ca", "Català"),
    ("cs", "Čeština"),
    ("da", "Dansk"),
    ("de", "Deutsch"),
    ("en", "English"),
    ("el", "Ελληνικά"),
    ("en-gb", "English (United Kingdom)"),
    ("es", "Español"),
    ("fi", "Suomi"),
    ("fr", "Français"),
    ("gl", "Galego"),
    ("he", "עברית"),
    ("hu", "Magyár"),
    ("hr", "Hrvatski"),
    ("id", "Indonesia"),
    ("is", "Íslenska"),
    ("it", "Italiano"),
    ("ja", "日本"),
    ("kab", "Taqbaylit"),
    ("kk", "Қазақ тілі"),
    ("ko", "한국어"),
    ("nb", "Norsk bokmål"),
    ("nl", "Nederlands"),
    ("pl", "Polski"),
    ("pt", "Português"),
    ("pt-br", "Português brasileiro"),
    ("ro", "Română"),
    ("ru", "Русский"),
    ("sk", "Slovenčina"),
    ("sl", "Slovenščina"),
    ("sq", "Shqip"),
    ("sr", "Српски"),
    ("sr-latn", "Srpski"),
    ("sv", "Svenska"),
    ("th", "ไทย"),
    ("tr", "Türkçe"),
    ("uk", "Українська"),
    ("zh-hans", "中文"),
    ("zh-hant", "中文"),
)

SITE_ID = 1

# If you set this to False, Django will make some optimizations so as not
# to load the internationalization machinery.
USE_I18N = True
# If you set this to False, Django will not format dates, numbers and 
# calendars according to the current locale.
USE_L10N = True

# If you set this to False, Django will not use timezone-aware datetimes.
USE_TZ = True

# Type of automatic primary key, introduced in Django 3.2
DEFAULT_AUTO_FIELD = "django.db.models.AutoField"

# URL prefix to use, please see documentation for more details
URL_PREFIX = ""

# Absolute filesystem path to the directory that will hold user-uploaded files.
MEDIA_ROOT = os.path.join(DATA_DIR, "media")

# URL that handles the media served from MEDIA_ROOT. Make sure to use a 
# trailing slash.
MEDIA_URL = f"{URL_PREFIX}/media/

# Absolute path to the directory static files should be collected to.
# Don't put anything in this directory yourself; store your static files 
# in apps' "static/" subdirectories and in STATICFILES_DIRS.
STATIC_ROOT = os.path.join(DATA_DIR, "static")

# URL prefix for static files.
STATIC_URL = f"{URL_PREFIX}/static/

# Additional locations of static files
STATICFILES_DIRS = ( 
    # Put strings here, like "/home/html/static" or "C:/www/django/static". 
    # Always use forward slashes, even on Windows. 
    # Don't forget to use absolute paths, not relative paths. 
)

# List of finder classes that know how to find static files in 
# various locations.
STATICFILES_FINDERS = ( 
    "django.contrib.staticfiles.finders.FileSystemFinder", 
    "django.contrib.staticfiles.finders.AppDirectoriesFinder", 
    "compressor.finders.CompressorFinder", 
)

# Make this unique, and don't share it with anybody. 
# You can generate it using weblate-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.request", 
            "django.template.context_processors.csrf", 
            "django.contrib.messages.context_processors.messages", 
            "weblate.trans.context_processors.weblate_context", 
        ],
    },


# GitHub username and token for sending pull requests.
# Please see the documentation for more details.
GITHUB_CREDENTIALS = {}

# GitLab username and token for sending merge requests.
# Please see the documentation for more details.
GITLAB_CREDENTIALS = {}

# Bitbucket username and token for sending merge requests.
# Please see the documentation for more details.
BITBUCKETSERVER_CREDENTIALS = {}

# Authentication configuration
AUTHENTICATION_BACKENDS = [
    "social_core.backends.email.EmailAuth",
    # "social_core.backends.google.GoogleOAuth2",
    # "social_core.backends.github.GithubOAuth2",
    # "social_core.backends.suse.OpenSUSETokenAuth",
    # "social_core.backends.ubuntu.UbuntuOpenId",
    # "social_core.backends.fedora.FedoraOpenId",
    # "social_core.backends.facebook.FacebookOAAuth2",
    "weblate.accounts.auth.WeblateUserBackend",
]

# Custom user model
AUTH_USER_MODEL = "weblate_auth.User"

# Social auth backends setup
SOCIAL_AUTH_GITHUB_KEY = 
SOCIAL_AUTH_GITHUB_SECRET = 
SOCIAL_AUTH_GITHUB_SCOPE = ["user:email"]

SOCIAL_AUTH_GITHUB_ORG_KEY = 
SOCIAL_AUTH_GITHUB_ORG_SECRET = 
SOCIAL_AUTH_GITHUB_ORG_NAME = 

SOCIAL_AUTH_GITHUB_TEAM_KEY = 
SOCIAL_AUTH_GITHUB_TEAM_SECRET = 
SOCIAL_AUTH_GITHUB_TEAM_ID = 

SOCIAL_AUTH_BITBUCKET_OAUTH2_KEY = 
SOCIAL_AUTH_BITBUCKET_OAUTH2_SECRET = 
SOCIAL_AUTH_BITBUCKET_OAUTH2_VERIFIED_EMAILS_ONLY = True

SOCIAL_AUTH_FACEBOOK_KEY = 
SOCIAL_AUTH_FACEBOOK_SECRET = 
SOCIAL_AUTH_FACEBOOK_SCOPE = ["email", "public_profile"]
SOCIAL_AUTH_FACEBOOK_PROFILE_EXTRA_PARAMS = {"fields": "id, name, email"}

SOCIAL_AUTH_GOOGLE_OAUTH2_KEY = 
SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET = 

# Social auth settings
SOCIAL_AUTH_PIPELINE = (suite sur la page suivante)
"social_core.pipeline.social_auth.social_details",
"social_core.pipeline.social_auth.social_uid",
"social_core.pipeline.social_auth.social_user",
"weblate.accounts.pipeline.user.get_username",
"weblate.accounts.pipeline.require_email",
"social_core.pipeline.mail.mail_validation",
"weblate.accounts.pipeline.store_params",
"weblate.accounts.pipeline.verify_open",
"social_core.pipeline.user.create_user",
"social_core.pipeline.social_auth.associate_user",
"social_core.pipeline.social_auth.associate_by_email",
"social_core.pipeline.social_auth.auth_allowed",
"social_core.pipeline.social_auth.load_extra_data",
"weblate.accounts.pipeline.store_email",
"weblate.accounts.pipeline.notify_connect",
"weblate.accounts.pipeline.password_reset",
"weblate.accounts.pipeline.ensure_valid",
"weblate.accounts.pipeline.remove_account",
"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",
"social_core.pipeline.disconnect.disconnect",
"weblate.accounts.pipeline.cleanup_next",
"weblate.accounts.pipeline.notify_disconnect",
"weblate.accounts.pipeline.reauthenticate",
"social_core.pipeline.disconnect.disconnect",
"weblate.accounts.pipeline.cleanup_next",

"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",
"social_core.pipeline.disconnect.disconnect",
"weblate.accounts.pipeline.cleanup_next",

SOCIAL_AUTH_CONNECTION_PIPELINE = [
"social_core.pipeline.disconnect.disconnect",
"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",
"social_core.pipeline.disconnect.disconnect",
"weblate.accounts.pipeline.cleanup_next",

# Custom authentication strategy
SOCIAL_AUTH_STRATEGY = "weblate.accounts.strategy.WeblateStrategy"

# Raise exceptions so that we can handle them later
SOCIAL_AUTH_RAISE_EXCEPTIONS = True

SOCIAL_AUTH_EMAIL_VALIDATION_FUNCTION = "weblate.accounts.pipeline.send_validation"
SOCIAL_AUTH_EMAIL_VALIDATION_URL = f"{URL_PREFIX}/accounts/email-sent/
SOCIAL_AUTH_LOGIN_ERROR_URL = f"{URL_PREFIX}/accounts/login/
SOCIAL_AUTH_EMAIL_FORM_URL = f"{URL_PREFIX}/accounts/email/
SOCIAL_AUTH_NEW_ASSOCIATION_REDIRECT_URL = f"{URL_PREFIX}/accounts/profile/#account

SOCIAL_AUTH_PROTECTED_USER_FIELDS = ("email",)
SOCIAL_AUTH_SLUGIFY_USERNAMES = True
SOCIAL_AUTH_SLUGIFY_FUNCTION = "weblate.accounts.pipeline.slugify_username"

# Password validation configuration
AUTH_PASSWORD_VALIDATORS = [
{
"NAME": "django.contrib.auth.password_validation.
UserAttributeSimilarityValidator" # noqa: E501, pylint: disable=line-too-long
},
{
"NAME": "django.contrib.auth.password_validation.MinimumLengthValidator",
"OPTIONS": {'min_length': 10},
},

(suite sur la page suivante)
# Optional password strength validation by django-zxcvbn-password
# {
#   "NAME": "zxcvbn_password.ZXCVBNValidator",
#   "OPTIONS": {
#     "min_score": 3,
#     "user_attributes": ["username", "email", "full_name"]
#   }
# },

# Password hashing (prefer Argon)
PASSWORD_HASHERS = [
    "django.contrib.auth.hashers.Argon2PasswordHasher",
    "django.contrib.auth.hashers.PBKDF2PasswordHasher",
    "django.contrib.auth.hashers.PBKDF2SHA1PasswordHasher",
    "django.contrib.auth.hashers.BCryptSHA256PasswordHasher",
]

# Allow new user registrations
REGISTRATION_OPEN = True

# Shortcut for login required setting
REQUIRE_LOGIN = False

# Middleware
MIDDLEWARE = [
    "weblate.middleware.RedirectMiddleware",
    "weblate.middleware.ProxyMiddleware",
    "corsheaders.middleware.CorsMiddleware",
    "django.middleware.security.SecurityMiddleware",
    "django.contrib.sessions.middleware.SessionMiddleware",
    "django.middleware.csrf.CsrfViewMiddleware",
    "weblate.accounts.middleware.AuthenticationMiddleware",
    "django.contrib.messages.middleware.MessageMiddleware",
    "weblate.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",
    "weblateformats",
    "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",
"crispy_bootstrap3",
"compressor",
"rest_framework",
"rest_framework.authtoken",
"django_filters",
"django_celery_beat",
"corsheaders",
]

# Custom exception reporter to include some details
DEFAULT_EXCEPTION_REPORTER_FILTER = "weblate.trans.debug.
\rightarrow 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_\rightarrow LOCAL2)
        handler.close()
    except OSError:
        HAVE_SYSLOG = True
    else:
        HAVE_SYSLOG = False

DEFAULT_LOG = "console" if DEBUG or not HAVE_SYSLOG else "syslog"
DEFAULT_LOGLEVEL = "DEBUG" if DEBUG else "INFO"

# A sample logging configuration. The only tangible logging
# performed by this configuration is to send an email to
# the site admins on every HTTP 500 error when DEBUG=False.
# See http://docs.djangoproject.com/en/stable/topics/logging for
# more details on how to customize your logging configuration.

```json
LOGGING = {
    "version": 1,
    "disable_existing_loggers": True,
    "filters": {
        "require_debug_false": {"()": "django.utils.log.RequireDebugFalse"},
    },
    "formatters": {
        "syslog": {
            "format": "weblate\[(process)d\]: (%levelname)s %message$s",
        },
        "simple": {"format": "(%asctime)s (%levelname)s (%process)s %message$s"},
        "logfile": {"format": "(%asctime)s (%levelname)s (%message)s"},
        "django.server": {
            "()": "django.utils.log.ServerFormatter",
            "format": "[%(server_time)s] %(message)s",
        },
    },
    "handlers": {
        "mail_admins": {
            "level": "ERROR",
            "filters": ["require_debug_false"],
            "class": "django.utils.log.AdminEmailHandler",
            "include_html": True,
        },
        "console": {
            "level": "DEBUG",
            "class": "logging.StreamHandler",
            "formatter": "simple",
        },
        "django.server": {
            "level": "INFO",
            "class": "logging.StreamHandler",
            "formatter": "django.server",
        },
        "syslog": {
            "level": "DEBUG",
            "class": "logging.handlers.SysLogHandler",
            "formatter": "syslog",
            "address": "/dev/log",
            "facility": SysLogHandler.LOG_LOCAL2,
        },
    },
    "loggers": {
        "django.request": {
            "handlers": ["mail_admins", DEFAULT_LOG],
            "level": "ERROR",
            "propagate": True,
        },
        "django.server": {
            "handlers": ["django.server"],
            "level": "INFO",
            "propagate": False,
        },
    }
}
```

# Logging to a file

```json
# "logfile": {
#     "level": "DEBUG",
#     "class": "logging.handlers.RotatingFileHandler",
#     "filename": "/var/log/weblate/weblate.log",
#     "maxBytes": 100000,
#     "backupCount": 3,
#     "formatter": "logfile",
# }
```

2.17. Configuration d'exemple
# Logging database queries
# "django.db.backends": {
#  "handlers": [DEFAULT_LOG],
#  "level": "DEBUG",
# },
# "weblate": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},

# Logging VCS operations
"weblate.vcs": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},

# Python Social Auth
"social": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},

# Django Authentication Using LDAP
"django_auth_ldap": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},

# SAML IdP
"djangosaml2idp": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},

# Remove syslog setup if it's not present
if not HAVE_SYSLOG:
    del LOGGING["handlers"]["syslog"]

# List of machine translations
MT_SERVICES = [
    # "weblate.machinery.apertium.ApertiumAPYTranslation",
    # "weblate.machinery.baidu.BaiduTranslation",
    # "weblate.machinery.deepi.DeepILTranslation",
    # "weblate.machinery.glosbe.GlosbeTranslation",
    # "weblate.machinery.google.GoogleTranslation",
    # "weblate.machinery.googlev3.GoogleV3Translation",
    # "weblate.machinery.libretranslate.LibreTranslateTranslation",
    # "weblate.machinery.microsoft.MicrosoftCognitiveTranslation",
    # "weblate.machinery.microsoftterminology.MicrosoftTerminologyService",
    # "weblate.machinery.modernmt.ModernMTTranslation",
    # "weblate.machinery.mymemory.MyMemoryTranslation",
    # "weblate.machinery.netease.NeteaseSightTranslation",
    # "weblate.machinery.tmserver.AmagamaTranslation",
    # "weblate.machinery.tmserver.TMServerTranslation",
    # "weblate.machinery.yandex.YandexTranslation",
    # "weblate.machinery.saptranslataionhub.SAPTranslationHub",
    # "weblate.machinery.youdao.YoudaoTranslation",
    # "weblate.machinery.weblatetm.WeblateTranslation",
    "weblate.memory.machine.WeblateMemory",
]

# Machine translation API keys

# URL of the Apertium APy server
MT_APERTIUM_API = None

# DeepL API key
MT_DEEPL_KEY = None

# LibreTranslate
MT_LIBRETRANSLATE_API_URL = None
MT_LIBRETRANSLATE_KEY = None

# Microsoft Cognitive Services Translator API, register at
# https://portal.azure.com/
MT_MICROSOFT_COGNITIVE_KEY = None
MT_MICROSOFT_REGION = None
# ModernMT
MT_MODERNMT_KEY = None

# MyMemory identification email, see
# https://mymemory.translated.net/doc/spec.php
MT_MYMEMORY_EMAIL = None

# Optional MyMemory credentials to access private translation memory
MT_MYMEMORY_USER = None
MT_MYMEMORY_KEY = None

# Google API key for Google Translate API v2
MT_GOOGLE_KEY = None

# Google Translate API3 credentials and project id
MT_GOOGLE_CREDENTIALS = None
MT_GOOGLE_PROJECT = None

# Baidu app key and secret
MT_BAIDU_ID = None
MT_BAIDU_SECRET = None

# Youdao Zhiyun app key and secret
MT_YOUDAO_ID = None
MT_YOUDAO_SECRET = None

# NetEase Sight (Jianwai) app key and secret
MT_NETEASE_KEY = None
MT_NETEASE_SECRET = None

# API key for Yandex Translate API
MT_YANDEX_KEY = None

# tmserver URL
MT_TMSERVER = None

# SAP Translation Hub
MT_SAP_BASE_URL = None
MT_SAP_SANDBOX_APIKEY = None
MT_SAP_USERNAME = None
MT_SAP_PASSWORD = None
MT_SAP_USE_MT = True

# Use HTTPS when creating redirect URLs for social authentication, see
# documentation for more details:
# html#processing-redirects-and-urlopen
SOCIAL_AUTH_REDIRECT_IS_HTTPS = ENABLE_HTTPS

# Make CSRF cookie HttpOnly, see documentation for more details:
# https://docs.djangoproject.com/en/1.11/ref/settings/#csrf-cookie-httponly
CSRF_COOKIE_HTTPONLY = True
CSRF_COOKIE_SECURE = ENABLE_HTTPS

# Store CSRF token in session
CSRF_USE_SESSIONS = True

# Customize CSRF failure view
CSRF_FAILURE_VIEW = "weblate.trans.views.error.csrf_failure"
SESSION_COOKIE_SECURE = ENABLE_HTTPS
SESSION_COOKIE_HTTPONLY = True

# SSL redirect
SECURE_SSL_REDIRECT = ENABLE_HTTPS
SECURE_SSL_HOST = SITE_DOMAIN
# Sent refererrer only for same origin links
SECURE_REFERRER_POLICY = "same-origin"
# SSL redirect URL exemption list
SECURE_REDIRECT_EXEMPT = (r"healthz/$",)  # Allowing HTTP access to health check
# Session cookie age (in seconds)
SESSION_COOKIE_AGE = 1000
SESSION_COOKIE_AGE_AUTHENTICATED = 1209600
SESSION_COOKIE_SAMESITE = "Lax"
# Increase allowed upload size
DATA_UPLOAD_MAX_MEMORY_SIZE = 50000000
# Allow more fields for case with a lot of subscriptions in profile
DATA_UPLOAD_MAX_NUMBER_FIELDS = 2000

# Apply session cookie settings to language cookie as well
LANGUAGE_COOKIE_SECURE = SESSION_COOKIE_SECURE
LANGUAGE_COOKIE_HTTPONLY = SESSION_COOKIE_HTTPONLY
LANGUAGE_COOKIE_AGE = SESSION_COOKIE_AGE_AUTHENTICATED * 10
LANGUAGE_COOKIE_SAMESITE = SESSION_COOKIE_SAMESITE

# Some security headers
SECURE_BROWSER_XSS_FILTER = True
X_FRAME_OPTIONS = "DENY"
SECURE_CONTENT_TYPE_NOSNIFF = True

# Optionally enable HSTS
SECURE_HSTS_SECONDS = 31536000 if ENABLE_HTTPS else 0
SECURE_HSTS_PRELOAD = ENABLE_HTTPS
SECURE_HSTS_INCLUDE_SUBDOMAINS = ENABLE_HTTPS

# HTTPS detection behind reverse proxy
SECURE_PROXY_SSL_HEADER = None

# URL of login
LOGIN_URL = f"{URL_PREFIX}/accounts/login/"

# URL of logout
LOGOUT_URL = f"{URL_PREFIX}/accounts/logout/"

# Default location for login
LOGIN_REDIRECT_URL = f"{URL_PREFIX}/"

# Anonymous user name
ANONYMOUS_USER_NAME = "anonymous"

# Reverse proxy settings
IP_PROXY_HEADER = "HTTP_X_FORWARDED_FOR"
IP_BEHIND_REVERSE_PROXY = False
IP_PROXY_OFFSET = 0

# Sending HTML in mails
EMAIL_SEND_HTML = True

# Subject of emails includes site title
EMAIL_SUBJECT_PREFIX = f"[{SITE_TITLE}] "

# Enable remote hooks
ENABLE_HOOKS = True

# By default the length of a given translation is limited to the length of
# the source string * 10 characters. Set this option to False to allow longer
# translations (up to 10,000 characters)
LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH = True

# Use simple language codes for default language/country combinations
SIMPLIFY_LANGUAGES = True

# Render forms using bootstrap
CRISPY_ALLOWED_TEMPLATE_PACKS = "bootstrap3"
CRISPY_TEMPLATE_PACK = "bootstrap3"

# List of quality checks
# CHECK_LIST = (  
#   "weblate.checks.same.SameCheck",
#   "weblate.checks.chars.BeginNewlineCheck",
#   "weblate.checks.chars.EndNewlineCheck",
#   "weblate.checks.chars.BeginSpaceCheck",
#   "weblate.checks.chars.EndSpaceCheck",
#   "weblate.checks.chars.DoubleSpaceCheck",
#   "weblate.checks.chars.EndStopCheck",
#   "weblate.checks.chars.EndColonCheck",
#   "weblate.checks.chars.EndQuestionCheck",
#   "weblate.checks.chars.EndExclamationCheck",
#   "weblate.checks.chars.EndEllipsisCheck",
#   "weblate.checks.chars.EndSemicolonCheck",
#   "weblate.checks.chars.MaxLengthCheck",
#   "weblate.checks.chars.KashidaCheck",
#   "weblate.checks.chars.PunctuationSpacingCheck",
#   "weblate.checks.format.PythonFormatCheck",
#   "weblate.checks.format.PythonBraceFormatCheck",
#   "weblate.checks.format.PHPFormatCheck",
#   "weblate.checks.format.CFormatCheck",
#   "weblate.checks.format.PerlFormatCheck",
#   "weblate.checks.format.JavaScriptFormatCheck",
#   "weblate.checks.format.LuaFormatCheck",
#   "weblate.checks.format.ObjectPascalFormatCheck",
#   "weblate.checks.format.SchemeFormatCheck",
#   "weblate.checks.format.CSharpFormatCheck",
#   "weblate.checks.format.JavaFormatCheck",
#   "weblate.checks.format.JsonMessageFormatCheck",
#   "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.chars.OutOfOrderCheck",
#   "weblate.checks.chars.EscapedNewlineCountingCheck",
#   "weblate.checks.chars.NewLineCountCheck",
#   "weblate.checks.markup.BBCodeCheck",
#   "weblate.checks.markup.XMLValidityCheck",
#   "weblate.checks.markup.XMLTagsCheck",
#   "weblate.checks.markup.MarkdownRefLinkCheck",  
)
Les éléments suivants sont inclus dans la configuration :

- "weblate.checks.markup.MarkdownLinkCheck",
- "weblate.checks.markup.MarkdownSyntaxCheck",
- "weblate.checks.markup.URLCheck",
- "weblate.checks.markup.SafeHTMLCheck",
- "weblate.checks.placeholders.PlaceholderCheck",
- "weblate.checks.placeholders.RegexCheck",
- "weblate.checks.duplicate.DuplicateCheck",
- "weblate.checks.source.OptionalPluralCheck",
- "weblate.checks.source.EllipsisCheck",
- "weblate.checks.source.MoreFailingCheck",
- "weblate.checks.source.LongUntranslatedCheck",
- "weblate.checks.format.MultipleUnnamedFormatsCheck",
- "weblate.checks.glossary.GlossaryCheck",

# List of automatic fixups

USER_FIX_LIST = (
    "weblate.trans.autofixes.whitespace.ReplaceBookendingWhitespaces",
    "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.xml.XMLCustomizeAddon",
    "weblate.addons.properties.PropertiesSortAddon",
    "weblate.addons.git.GitSquashAddon",
    "weblate.addons.removal.RemoveComments",
    "weblate.addons.removal.RemoveSuggestions",
    "weblate.addons.xml.XMLUpdateAddon",
    "weblate.addons.yaml.YAMLCustomizeAddon",
    "weblate.addons.cdn.CDNJSAddon",
)

# E-mail address that error messages come from.

SERVER_EMAIL = "noreply@example.com"

# Default email address to use for various automated correspondence from
# the site managers. Used for registration emails.

DEFAULT_FROM_EMAIL = "noreply@example.com"
# List of URLs your site is supposed to serve
ALLOWED_HOSTS = ['*']

# Configuration for caching
CACHES = {
    "default": {
        "BACKEND": "django_redis.cache.RedisCache",
        "LOCATION": "redis://127.0.0.1:6379/1",
        # If redis is running on same host as Weblate, you might want to use unix sockets instead:
        # "LOCATION": "unix:///var/run/redis/redis.sock?db=1",
        "OPTIONS": {
            "CLIENT_CLASS": "django_redis.client.DefaultClient",
            "PARSER_CLASS": "redis.connection.HiredisParser",
            # If you set password here, adjust CELERY_BROKER_URL as well
            "PASSWORD": None,
            "CONNECTION_POOL_KWARGS": {},
        },
        "KEY_PREFIX": "weblate",
        "TIMEOUT": 3600,
    },
    "avatar": {
        "BACKEND": "django.core.cache.backends.filebased.FileBasedCache",
        "LOCATION": os.path.join(DATA_DIR, "avatar-cache"),
        "TIMEOUT": 86400,
        "OPTIONS": {"MAX_ENTRIES": 1000},
    },
}

# Store sessions in cache
SESSION_ENGINE = "django.contrib.sessions.backends.cache"
# Store messages in session
MESSAGE_STORAGE = "django.contrib.messages.storage.session.SessionStorage"

# REST framework settings for API
REST_FRAMEWORK = {
    # Use Django's standard 'django.contrib.auth' permissions,
    # or allow read-only access for unauthenticated users.
    "DEFAULT_PERMISSION_CLASSES": [
        # Require authentication for login required sites
        "rest_framework.permissions.IsAuthenticated"
        if REQUIRE_LOGIN
        else "rest_framework.permissions.IsAuthenticatedOrReadOnly"
    ],
    "DEFAULT_AUTHENTICATION_CLASSES": (
        "rest_framework.authentication.TokenAuthentication",
        "weblate.api.authentication.BearerAuthentication",
        "rest_framework.authentication.SessionAuthentication",
    ),
    "DEFAULT_THROTTLE_CLASSES": (
        "weblate.api.throttling.UserRateThrottle",
        "weblate.api.throttling.AnonRateThrottle",
    ),
    "DEFAULT_THROTTLE_RATES": {"anon": "100/day", "user": "5000/hour"},
    "DEFAULT_PAGINATION_CLASS": "weblate.api.pagination.StandardPagination",
    "PAGE_SIZE": 50,
    "VIEW_DESCRIPTION_FUNCTION": "weblate.api.views.get_view_description",
    "UNAUTHENTICATED_USER": "weblate.auth.models.get_anonymous",
}

# Fonts CDN URL
FONT CDs URL = None

# Django compressor offline mode
COMPRESS_OFFLINE = False
COMPRESS_OFFLINE_CONTEXT = [
    (
        "fonts_cdn_url": FONTS_CDN_URL,
        "STATIC_URL": STATIC_URL,
        "LANGUAGE_BIDI": False,
    ),
    (
        "fonts_cdn_url": FONTS_CDN_URL,
        "STATIC_URL": STATIC_URL,
        "LANGUAGE_BIDI": True,
    ),
]

# Require login for all URLs
if REQUIRE LOGIN:
    LOGIN_REQUIRED_URLS = (r"/(.*)$",)
# In such case you want to include some of the exceptions
LOGIN_REQUIRED_URLS_EXCEPTIONS = [
    rf"{URL_PREFIX}/accounts/(.*)$", # Required for login
    rf"{URL_PREFIX}/admin/login/(.*)$", # Required for admin login
    rf"{URL_PREFIX}/static/(.*)$", # Required for development mode
    rf"{URL_PREFIX}/widgets/(.*)$", # Allowing public access to widgets
    rf"{URL_PREFIX}/data/(.*)$", # Allowing public access to data exports
    rf"{URL_PREFIX}/hooks/(.*)$", # Allowing public access to notification hooks
    rf"{URL_PREFIX}/healthz/$", # Allowing public access to health check
    rf"{URL_PREFIX}/api/(.*)$", # Allowing access to API
    rf"{URL_PREFIX}/js/i18n/$", # JavaScript localization
    rf"{URL_PREFIX}/contact/$", # Optional for contact form
    rf"{URL_PREFIX}/legal/(.*)$", # Optional for legal app
    rf"{URL_PREFIX}/avatar/(.*)$", # Optional for avatars
]

# Silence some of the Django system checks
SILENCED_SYSTEM_CHECKS = [
    # We have modified django.contrib.auth.middleware.AuthenticationMiddleware
    # as weblate.accounts.middleware.AuthenticationMiddleware
    "admin.E408",
]

# Celery worker configuration for testing
# CELERY_TASK_ALWAYS_EAGER = True
# CELERY_BROKER_URL = "memory://"
# CELERY_TASK_EAGER_PROPAGATES = True
CELERY_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_MAX_MEMORY_PER_CHILD = 200000
CELERY_BEAT_SCHEDULER = "django_celery_beat.schedulers:DatabaseScheduler"
CELERY_TASK_ROUTES = {
    "weblate.trans.tasks.auto_translate*": {"queue": "translate"},
    "weblate.accounts.tasks.notify_*": {"queue": "notify"},
    "weblate.accounts.tasks.send_mails": {"queue": "notify"},
    "weblate.utilis.tasks.settings_backup": {"queue": "backup"},
    "weblate.utilis.tasks.database_backup": {"queue": "backup"},
    "weblate.wladmin.tasks.backup": {"queue": "backup"},
    "weblate.wladmin.tasks.backup_service": {"queue": "backup"},
    "weblate.memory.tasks.*": {"queue": "memory"},
}
# CORS allowed origins
CORS_ALLOWED_ORIGINS = []
CORS_URLS_REGEX = r'^/api/.*$

# Enable plain database backups
DATABASE_BACKUP = "plain"

# Enable auto updating
AUTO_UPDATE = False

# PGP commits signing
WEBLATE_GPG_IDENTITY = None

# Third party services integration
MATOMO_SITE_ID = None
MATOMO_URL = None
GOOGLE_ANALYTICS_ID = None
SENTRY_DSN = None
SENTRY_ENVIRONMENT = SITE_DOMAIN
AKISMET_API_KEY = None

## 2.18 Commandes de gestion

**Note :** Running management commands under a different user than the one running your webserver can result in files getting wrong permissions, please check *Permissions du système de fichiers* for more details.

You will find basic management commands (available as `./manage.py` in the Django sources, or as an extended set in a script called `weblate` installable atop Weblate).

### 2.18.1 Invoking management commands

As mentioned before, invocation depends on how you installed Weblate.

If using virtualenv for Weblate, you can either specify the full path to `weblate`, or activate the virtualenv prior to invoking it:

```bash
# Direct invocation
~/weblate-env/bin/weblate

# Activating virtualenv adds it to search path
. ~/weblate-env/bin/activate
weblate
```

If you are using source code directly (either from a tarball or Git checkout), the management script is `./manage.py` available in the Weblate sources. To run it:

```
python ./manage.py list_versions
```

If you’ve installed Weblate using the pip installer, or by using the `./setup.py` script, the `weblate` is installed to your path (or virtualenv path), from where you can use it to control Weblate:

```
weblate list_versions
```

For the Docker image, the script is installed like above, and you can run it using `docker exec`:
The Weblate Manual, Version 4.16

```bash
docker exec --user weblate <container> weblate list_versions
```

For **docker-compose** the process is similar, you just have to use **docker-compose exec**:

```bash
docker-compose exec --user weblate weblate list_versions
```

In case you need to pass it a file, you can temporary add a volume:

```bash
docker-compose exec --user weblate /tmp:/tmp weblate importusers /tmp/←users.json
```

Voir aussi :

* Installing using Docker, Installation sur Debian et Ubuntu, Installation sur SUSE et openSUSE, Installing on RedHat, Fedora and CentOS, Installing from sources

### 2.18.2 add_suggestions

```bash
weblate add_suggestions <project> <component> <language> <file>
```

Nouveau dans la version 2.5.

Imports a translation from the file to use as a suggestion for the given translation. It skips duplicated translations; only different ones are added.

```bash
--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).

Exemple :

```bash
weblate --author michal@cihar.com add_suggestions weblate application cs /tmp/←suggestions-cs.po
```

### 2.18.3 auto_translate

```bash
weblate auto_translate <project> <component> <language>
```

Nouveau dans la version 2.5.

Modifié dans la version 4.6 : Ajouter un paramètre pour le mode de traduction.

Performs automatic translation based on other component translations.

```bash
--source PROJECT/COMPONENT
```

Specifies the component to use as source available for translation. If not specified all components in the project are used.

```bash
--user USERNAME
```

Specify username listed as author of the translations. « Anonymous user » is used if not specified.

```bash
--overwrite
```

Whether to overwrite existing translations.

```bash
--inconsistent
```

Whether to overwrite existing translations that are inconsistent (see Incohérence).

```bash
--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.

Exemple :
```
weblate auto_translate --user nijel --inconsistent --source weblate/application --weblate website cs
```

Voir aussi :

Traduction automatique

2.18.4 celery_queues

**weblate celery_queues**

Nouveau dans la version 3.7.
Displays length of Celery task queues.

2.18.5 checkgit

**weblate checkgit <project|project/component>**

Prints current state of the back-end Git repository.
You can either define which project or component to update (for example weblate/application), or use --all to update all existing components.

2.18.6 commitgit

**weblate commitgit <project|project/component>**

Commits any possible pending changes to the back-end Git repository.
You can either define which project or component to update (for example weblate/application), or use --all to update all existing components, or use --file-format to filter based on the file format.

2.18.7 commit_pending

**weblate commit_pending <project|project/component>**

Commits pending changes older than a given age.
You can either define which project or component to update (for example weblate/application), or use --all to update all existing components.

--age HOURS
Age in hours for committing. If not specified the value configured in Configuration des composants is used.
Note : 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 *Configuration des composants*.

Voir aussi :

*Running maintenance tasks, COMMIT_PENDING_HOURS*

### 2.18.8 cleanuptrans

**weblate cleanuptrans**

Cleans up orphaned checks and translation suggestions. There is normally no need to run this manually, as the cleanups happen automatically in the background.

Voir aussi :

*Running maintenance tasks*

### 2.18.9 cleanup_ssh_keys

**weblate cleanup_ssh_keys**

Nouveau dans la version 4.9.1.

Performs cleanup of stored SSH host keys :

- Removes deprecated RSA keys for GitHub which might cause issues connecting to GitHub.
- Removes duplicate entries in host keys.

Voir aussi :

*SSH repositories*

### 2.18.10 createadmin

**weblate createadmin**

Creates an admin account with a random password, unless it is specified.

---password PASSWORD

Provides a password on the command-line, to not generate a random one.

---no-password

Do not set password, this can be useful with `--update`.

---username USERNAME

Use the given name instead of admin.

---email USER@EXAMPLE.COM

Specify the admin e-mail address.

---name

Specify the admin name (visible).

---update

Update the existing user (you can use this to change passwords).

Modifié dans la version 2.9 : Added parameters `--username,--email,--name and --update`. 

Chapitre 2. Documentation pour l'administrateur
2.18.11 dump_memory

**weblate dump_memory**

Nouveau dans la version 2.20.
Export a JSON file containing Weblate Translation Memory content.

Voir aussi :
* Mémoire de traduction, Schéma de mémoire des traductions Weblate*

2.18.12 dumpuserdata

**weblate dumpuserdata <file.json>**

Dumps userdata to a file for later use by *importuserdata*.

**Indication** : This comes in handy when migrating or merging Weblate instances.

2.18.13 import_demo

**weblate import_demo**

Nouveau dans la version 4.1.
Creates a demo project with components based on <https://github.com/WeblateOrg/demo>. Make sure the celery tasks are running before running this command.
This can be useful when developing Weblate.

2.18.14 import_json

**weblate import_json <json-file>**

Nouveau dans la version 2.7.
Batch import of components based on JSON data.
The imported JSON file structure pretty much corresponds to the component object (see GET /api/components/(string:project)/(string:component)/). You have to include the name and filename 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.
```

Modifié dans la version 2.9 : The parameters --ignore and --update are there to deal with already imported components.
Example of JSON file :

---

2.18. Commandes de gestion
import_memory

2.18.15 import_memory

weblate import_memory <file>

Nouveau dans la version 2.20.
Imports a TMX or JSON file into the Weblate translation memory.

--language-map LANGMAP
   Allows mapping languages in the TMX to the Weblate translation memory. The language codes are mapped
   after normalization usually done by Weblate.
   This can be useful in case your TMX file locales happen not to match what you use in Weblate.

Voir aussi :

Mémoire de traduction, Schéma de mémoire des traductions Weblate

2.18.16 import_project

weblate import_project <project> <gitrepo> <branch> <filemask>

Modifié dans la version 3.0 : The import_project command is now based on the Découverte du composant 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

L'expression régulière doit contenir des groupes nommés component et language. Par exemple : (?P<language>[^/]*)/(?P<component>[-/]*).po
The import matches existing components based on files and adds the ones that do not exist. It does not change already existing ones.

```
--name-template TEMPLATE
  Customize the name of a component using Django template syntax.
  For example: Documentation: {{ component }}
```

```
--base-file-template TEMPLATE
  Customize the base file for monolingual translations.
  For example: {{ component }}/res/values/string.xml
```

```
--new-base-template TEMPLATE
  Customize the base file for addition of new translations.
  For example: {{ component }}/ts/en.ts
```

```
--file-format FORMAT
  You can also specify the file format to use (see Formats de fichiers pris en charge), the default is auto-detection.
```

```
--language-regex REGEX
  You can specify language filtering (see Configuration des composants) with this parameter. It has to be a valid regular expression.
```

```
--main-component
  You can specify which component will be chosen as the main one—the one actually containing the VCS repository.
```

```
--license NAME
  Specify the overall, project or component translation license.
```

```
--license-url URL
  Specify the URL where the translation license is to be found.
```

```
--vcs NAME
  In case you need to specify which version control system to use, you can do it here. The default version control is Git.
```

To give you some examples, let’s try importing two projects.

First The Debian Handbook translations, where each language has separate a folder with the translations of each chapter:

```
weblate import_project
  debian-handbook
  git://anonscm.debian.org/debian-handbook/debian-handbook.git
  squeeze/master
  */**.po
```

Then the Tanaguru tool, where the file format needs be specified, along with the base file template, and how all components and translations are located in single folder:

```
weblate import_project
  --file-format=properties
  --base-file-template=web-app/tgol-web-app/src/main/resources/i18n/%s-I18N
  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:
Filtering only translations in a chosen language:

```bash
./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:

```bash
$ 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:

```bash
$ 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'
```

Voir aussi :

More detailed examples can be found in the starting chapter, alternatively you might want to use `import_json`.

### 2.18.17 importuserdata

```bash
weblate importuserdata <file.json>
```

Imports user data from a file created by `dumpuserdata`.

### 2.18.18 importusers

```bash
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:

```bash
weblate dumpdata auth.User > users.json
```
2.18.19 install_addon

Nouveau dans la version 3.2.

**weblate install_addon --addon ADDON <project|project/component>**

Installs an add-on to a set of components.

---addon ADDON
   Name of the add-on to install. For example `weblate.gettext.customize`.

---configuration CONFIG
   JSON encoded configuration of an add-on.

---update
   Mettre à jour la configuration existante du module complémentaire.

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 *Personnaliser la sortie gettext* for all components:

```bash
weblate install_addon --addon weblate.gettext.customize --config '{"width": -1}' --update --all
```

**Voir aussi:**

*Extensions*

2.18.20 list_languages

**weblate list_languages <locale>**

Lists supported languages in MediaWiki markup - language codes, English names and localized names.

This is used to generate [https://wiki.l10n.cz/Slovn%C3%ADk_s_n%C3%A1zvy_jazyk%C5%AF](https://wiki.l10n.cz/Slovn%C3%ADk_s_n%C3%A1zvy_jazyk%C5%AF).

2.18.21 list_translators

**weblate list_translators <project|project/component>**

Lists translators by contributed language for the given project:

```
[French]
Jean Dupont <jean.dupont@example.com>

[English]
John Doe <jd@example.com>
```

---language-code
   List names by language code instead of language name.

You can either define which project or component to use (for example `weblate/application`), or use `--all` to list translators from all existing components.
2.18.22 list_versions

weblate list_versions

Lists all Weblate dependencies and their versions.

2.18.23 loadpo

weblate loadpo <project|project/component>

Reloads translations from disk (for example in case you have done some updates in the VCS repository).

--force
   Force update, even if the files should be up-to-date.

--lang LANGUAGE
   Limit processing to a single language.

You can either define which project or component to update (for example weblate/application), or use --all to update all existing components.

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.

2.18.24 lock_translation

weblate lock_translation <project|project/component>

Prevents further translation of a component.

Indication: 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.

Voir aussi: unlock_translation

2.18.25 move_language

weblate move_language source target

Nouveau dans la version 3.0.

Allows you to merge language content. This is useful when updating to a new version which contains aliases for previously unknown languages that have been created with the (generated) suffix. It moves all content from the source language to the target one.

Exemple:

weblate move_language cze cs

After moving the content, you should check whether there is anything left (this is subject to race conditions when somebody updates the repository meanwhile) and remove the (generated) language.
2.18.26 pushgit

`weblate pushgit <project|project/component>`

Pushes committed changes to the upstream VCS repository.

**--force-commit**

Force commits any pending changes, prior to pushing.

You can either define which project or component to update (for example `weblate/application`), or use **--all** to update all existing components.

**Note**: Weblate pushes changes automatically if *Pousser lors du commit* in *Configuration des composants* is turned on, which is the default.

2.18.27 unlock_translation

`weblate unlock_translation <project|project/component>`

Unlocks a given component, making it available for translation.

**Indication**: 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.

**voir aussi**:

`lock_translation`

2.18.28 setupgroups

`weblate setupgroups`

Configure les groupes par défaut et ajoute optionnellement tous les utilisateurs à ce groupe.

**--no-privs-update**

Désactive la mise à jour automatique des groupes existants (ajoute uniquement les nouveaux).

**--no-projects-update**

Empêche les mises à jour automatiques des groupes pour les projets existants. Cela permet d’ajouter les nouveaux groupes ajoutés aux projets existants, voir *Contrôle d’accès au projet*.

Voir aussi:

*Liste des privilèges et des rôles intégrés*
2.18.29 setuplang

`weblate setuplang`

Updates list of defined languages in Weblate.

`--no-update`

Turns off automatic updates of existing languages (only adds new ones).

2.18.30 updatechecks

`weblate updatechecks <project|project/component>`

Updates all checks for all strings.

**Indication:** Useful for upgrades which do major changes to checks.

You can either define which project or component to update (for example `weblate/application`), or use `--all` to update all existing components.

2.18.31 updategit

`weblate updategit <project|project/component>`

Fetches remote VCS repositories and updates the internal cache.

You can either define which project or component to update (for example `weblate/application`), or use `--all` to update all existing components.

**Note:** Usually it is better to configure hooks in the repository to trigger Déclencheurs de notification, instead of regular polling by `updategit`.

2.19 Annonces

Modifié dans la version 4.0 : Dans les versions précédentes cette fonctionnalité s’appelait whiteboard messages.

Fournissez des informations à vos traducteurs en publiant des annonces sur l’ensemble du site, par projet, composant ou langue.

Afficher la finalité, les échéances, le statut ou préciser les objectifs de la traduction.

Les utilisateurs seront notifiés des publications pour chaque projet suivi (a moins de se désinscrire).

Cela peut être utile pour différentes raisons, comme la publication de la finalité du site web ou spécifier les objectifs de traduction.

Les publications peuvent être postées à chaque niveau dans le menu :guilabel:`Manage`, :guilabel:`Post announcement`:
Cela peut aussi être ajouté en utilisant l’interface d’admin :
Les publications sont alors affichées en se basant sur leur contexte spécifique :

Aucun contexte spécifié
  Montré sur le tableau de bord (page d’accueil).

Projet spécifié
  Montré dans le projet, en incluant tous ses composants et traductions.

Composant spécifié
  Montré pour un composant donné et toutes ses traductions.

Langue spécifiée
  Montré sur la vue d’ensemble de la langue et toutes les traductions dans cette langue.

Voici à quoi cela ressemble sur la page d’ensemble de la langue :
2.20 Liste des composants

Spécifie plusieurs listes de composants à afficher comme options sur le tableau de bord de l’utilisateur, parmi lesquelles les utilisateurs peuvent choisir une vue par défaut. Voir Tableau de bord pour en savoir plus.

Modifié dans la version 2.20 : Un statut sera présenté pour chaque liste de composants présentée sur le tableau de bord.

Les noms et le contenu des listes de composants peuvent être spécifiés dans l’interface d’administration, dans la section Component lists. Chaque liste de composants doit avoir un nom qui est affiché à l’utilisateur et une abréviation le représentant dans l’adresse de la page.

Modifié dans la version 2.13 : Modifie les paramètres du tableau de bord pour les utilisateurs anonymes depuis l’interface d’administration, en altérant le tableau de bord présenté aux utilisateurs non-authentifiés.

2.20.1 Listes automatiques de composants

Nouveau dans la version 2.13.

Ajoute automatiquement des composants à la liste en fonction de leur abréviation en créant des règles Affectation automatique de la liste des composants.

— Utile pour maintenir les listes de composants pour de grosses installations, ou si vous voulez avoir une liste de composants avec tous les composants de votre installation de Weblate.

**Indication :** Crée une liste de composants contenant tous les composants de votre installation de Weblate.

1. Define Automatic component list assignment with ^.*$ as regular expression in both the project and the component fields, as shown on this image :
2.21 Modules Weblate optionnels

Plusieurs modules optionnels sont disponibles pour votre installation.
2.21.1 Exportateur Git

Nouveau dans la version 2.10.
Provides you read-only access to the underlying Git repository using HTTP(S).

Installation

1. Add `weblate.gitexport` to installed apps in `settings.py`:

```python
INSTALLED_APPS += ("weblate.gitexport",)
```

2. Export existing repositories by migrating your database after installation:

`weblate migrate`

Utilisation

The module automatically hooks into Weblate and sets the exported repository URL in the `Configuration des composants`. The repositories are accessible under the `/git/ part of the Weblate URL, for example `https://example.org/git/weblate/main/`.

Repositories for publicly available projects can be cloned without authentication:

```
git clone 'https://example.org/git/weblate/main/'
```

Access to browse the repositories with restricted access (with `Private access control` or when `REQUIRE_LOGIN` is enabled) requires an API token which can be obtained in your `user profile`:

```
git clone 'https://user:KEY@example.org/git/weblate/main/'
```

Indication : By default members or Users group and anonymous user have access to the repositories for public projects via Access repository and Power user roles.

2.21.2 Facturation

Nouveau dans la version 2.4.
This is used on Hosted Weblate to define billing plans, track invoices and usage limits.

Installation

1. Add `weblate.billing` to installed apps in `settings.py`:

```python
INSTALLED_APPS += ("weblate.billing",)
```

2. Run the database migration to optionally install additional database structures for the module:

`weblate migrate`
Utilisation

After installation you can control billing in the admin interface. Users with billing enabled will get new Billing tab in their Profil utilisateur.

The billing module additionally allows project admins to create new projects and components without being superusers (see Adding translation projects and components). This is possible when following conditions are met:

- The billing is in its configured limits (any overusage results in blocking of project/component creation) and paid (if its price is non zero)
- The user is admin of existing project with billing or user is owner of billing (the latter is necessary when creating new billing for users to be able to import new projects).

Upon project creation user is able to choose which billing should be charged for the project in case he has access to more of them.

2.21.3 Mentions légales

Nouveau dans la version 2.15.

This is used on Hosted Weblate to provide required legal documents. It comes provided with blank documents, and you are expected to fill out the following templates in the documents:

- **legal/documents/tos.html**: Terms of service document
- **legal/documents/privacy.html**: Privacy policy document
- **legal/documents/summary.html**: Short overview of the terms of service and privacy policy

On changing the terms of service documents, please adjust `LEGAL_TOS_DATE` so that users are forced to agree with the updated documents.

**Note**: Legal documents for the Hosted Weblate service are available in this Git repository <https://github.com/WeblateOrg/wllegal/tree/main/wllegal/templates/legal/documents>.

Most likely these will not be directly usable to you, but might come in handy as a starting point if adjusted to meet your needs.

Installation

1. Add `weblate.legal` to installed apps in `settings.py`:

```
INSTALLED_APPS += ("weblate.legal",)

# Optional:
SOCIAL_AUTH_PIPELINE += ("weblate.legal.pipeline.tos_confirm"),

MIDDLEWARE += [ "weblate.legal.middleware.RequireTOSMiddleware", ]
```

2. Run the database migration to optionally install additional database structures for the module:

```
weblate migrate
```

3. Edit the legal documents in the `weblate/legal/templates/legal/` folder to match your service.
Utilisation

After installation and editing, the legal documents are shown in the Weblate UI.

2.21.4 Avatars

Avatars are downloaded and cached server-side to reduce information leaks to the sites serving them by default. The built-in support for fetching avatars from e-mails addresses configured for it can be turned off using `ENABLE_AVATARS`.

Weblate currently supports:
- Gravatar
- Libravatar

Voir aussi:
- Cache Avatar, `AVATAR_URL_PREFIX`, `ENABLE_AVATARS`

2.21.5 Protection contre le spam

You can protect against spamming by users by using the Akismet service.

1. Install the `akismet` Python module (this is already included in the official Docker image).
2. Obtain the Akismet API key.
3. Store it as `AKISMET_API_KEY` or `WEBLATE_AKISMET_API_KEY` in Docker.

Following content is sent to Akismet for checking:
- Suggestions d’utilisateurs non authentifiés
- Project and component descriptions and links

Note: This (among other things) relies on IP address of the client, please see `Running behind reverse proxy` for properly configuring that.

Voir aussi:
- `Running behind reverse proxy`, `AKISMET_API_KEY`, `WEBLATE_AKISMET_API_KEY`

2.21.6 Signing Git commits with GnuPG

Nouveau dans la version 3.1.

All commits can be signed by the GnuPG key of the Weblate instance.

1. Turn on `WEBLATE_GPG_IDENTITY`. (Weblate will generate a GnuPG key when needed and will use it to sign all translation commits.)

Cette fonctionnalité nécessite l'installation de GnuPG 2.1 ou plus récent.

You can find the key in the `DATA_DIR` and the public key is shown on the « About » page:
2. Alternativment vous pouvez également importer des clés existantes dans Weblate, en définissant `HOME=$DATA_DIR/home` lorsque vous appelez `gpg`.

**Voir aussi :**

`WEBLATE_GPG.IDENTITY`

### 2.21.7 Limite de requêtes

Modifié dans la version 3.2 : La limitation des requêtes accepte maintenant une configuration plus fine-grained.

Modifié dans la version 4.6 : La limite de requêtes n’est plus applicable pour les super utilisateurs.

Certaines opérations dans Weblate sont limitées. Les operations limitées sont le nombre maximum d'essais dans `RATELIMIT_ATTEMPTS` secondes. Le nombre d'essais est ensuite bloqué pour `RATELIMIT_LOCKOUT` secondes. Il y a aussi des configurations spécifiques pour les scopes, par exemple `RATELIMIT_CONTACT_ATTEMPTS` ou `RATELIMIT_TRANSLATE_ATTEMPTS`. Le tableau suivant est une liste complète des scopes disponibles.

Le tableau suivant est une liste complète des scopes disponibles.

<table>
<thead>
<tr>
<th>Nombre d'essais dans <code>RATELIMIT_ATTEMPTS</code></th>
<th>Secondes</th>
<th><code>RATELIMIT_LOCKOUT</code> secondes</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>60</td>
<td>180</td>
</tr>
<tr>
<td>10</td>
<td>120</td>
<td>360</td>
</tr>
<tr>
<td>20</td>
<td>240</td>
<td>720</td>
</tr>
<tr>
<td>50</td>
<td>600</td>
<td>1800</td>
</tr>
</tbody>
</table>

Le tableau suivant est une liste complète des scopes disponibles.

Le tableau suivant est une liste complète des scopes disponibles.

<table>
<thead>
<tr>
<th>Scope</th>
<th>DESCRIPTION</th>
<th>LIMITATION ATTEMPTS</th>
<th>LIMITATION LOCKOUT</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>TRANSLATE</code></td>
<td>Traduction</td>
<td>20</td>
<td>60</td>
</tr>
<tr>
<td><code>CONTACT</code></td>
<td>Contact</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td><code>TRANSLATE_CONTACT</code></td>
<td>Traduction et contact</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td><code>EDIT</code></td>
<td>Edition</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td><code>EDIT_CONTACT</code></td>
<td>Edition et contact</td>
<td>5</td>
<td>15</td>
</tr>
</tbody>
</table>

Le tableau suivant est une liste complète des scopes disponibles.
<table>
<thead>
<tr>
<th>Nom</th>
<th>Portée</th>
<th>Tentatives autorisées</th>
<th>Fenêtre de débit</th>
<th>Période de verrouillage</th>
</tr>
</thead>
<tbody>
<tr>
<td>S’inscrire</td>
<td>REGISTRATION</td>
<td>5</td>
<td>300</td>
<td>600</td>
</tr>
<tr>
<td>Sending message to admins</td>
<td>MESSAGE</td>
<td>2</td>
<td>300</td>
<td>600</td>
</tr>
<tr>
<td>Authentification par mot de passe à la connexion</td>
<td>LOGIN</td>
<td>5</td>
<td>300</td>
<td>600</td>
</tr>
<tr>
<td>Recherche à l’échelle du site</td>
<td>SEARCH</td>
<td>6</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>Traduction</td>
<td>TRANSLATE</td>
<td>30</td>
<td>60</td>
<td>600</td>
</tr>
<tr>
<td>Adding to glossary</td>
<td>GLOSSARY</td>
<td>30</td>
<td>60</td>
<td>600</td>
</tr>
<tr>
<td>Starting translation into a new language</td>
<td>LANGUAGE</td>
<td>2</td>
<td>300</td>
<td>600</td>
</tr>
<tr>
<td>Creating new project</td>
<td>PROJECT</td>
<td>5</td>
<td>600</td>
<td>600</td>
</tr>
</tbody>
</table>

If a user fails to sign in `AUTH_LOCK_ATTEMPTS` times, password authentication will be turned off on the account until having gone through the process of having its password reset.

The settings can be also applied in the Docker container by adding `WEBLATE_` prefix to the setting name, for example `RATELIMIT_ATTEMPTS` becomes `WEBLATE_RATELIMIT_ATTEMPTS`.

The API has separate rate limiting settings, see `API rate limiting`.

Voir aussi :

Limite de requêtes, Running behind reverse proxy, API rate limiting

### 2.21.8 Fedora Messaging integration

Fedora Messaging is AMQP-based publisher for all changes happening in Weblate. You can hook additional services on changes happening in Weblate using this.

The Fedora Messaging integration is available as a separate Python module `weblate-fedora-messaging`. Please see <https://github.com/WeblateOrg/fedora_messaging/> for setup instructions.

### 2.22 Personnaliser 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.

**Avertissement** : 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.

Voir aussi :

Contribuer à Weblate
2.22.1 Creating a Python module

If you are not familiar with Python, you might want to look into Python For Beginners, explaining the basics and pointing to further tutorials.

To write a file with custom Python code (called a module), a place to store it is needed, either in the system path (usually something like `/usr/lib/python3.9/site-packages/`) or in the Weblate directory, which is also added to the interpreter search path.

Nouveau dans la version 3.8-5: When using Docker, you can place Python modules in `/app/data/python/` (see Docker container volumes), so they can be loaded by Weblate, for example from a settings override file.

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",
       packages=['weblate_customization'],
   )
   ```

3. Create a folder for the Python module (also called `weblate_customization`) for the customization code.
4. Within it, create a `__init__.py` file to make sure Python can import the module.
5. This package can now be installed using `pip install -e`. More info to be found in Editable installs.
6. Once installed, the module can be used in the Weblate configuration (for example `weblate_customization.checks.FooCheck`).

Your package structure should look like this:

```
weblate_customization
  ├── setup.py
  └── weblate_customization
      ├── __init__.py
      ├── addons.py
      └── checks.py
```

You can find an example of customizing Weblate at <https://github.com/WeblateOrg/customize-example>, it covers all the topics described below.

2.22.2 Changer le logo

1. Create a simple Django app containing the static files you want to overwrite (see Creating a Python module). Branding appears in the following files:

   - **icons/weblate.svg**
     Logo shown in the navigation bar.
   - **logo-*.png**
     Web icons depending on screen resolution and web-browser.
   - **favicon.ico**
     Web icon used by legacy browsers.
Avatars for bots or anonymous users. Some web-browsers use these as shortcut icons.

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.

Voir aussi :

Gestion des fichiers statiques (par ex. images, JavaScript, CSS), Serving static files

### 2.22.3 Custom quality checks, add-ons and auto-fixes

Pour installer votre code pour Personnaliser les réparations automatiques, Rédiger ses propres contrôles ou Écrire un greffon dans Weblate :

1. Place the files into your Python module containing the Weblate customization (see Creating a Python module).
2. Add its fully-qualified path to the Python class in the dedicated settings (`WEBLATE_ADDONS`, `CHECK_LIST` or `AUTOFIX_LIST`):

```python
# Checks
CHECK_LIST += ("weblate_customization.checks.FooCheck",)

# Autofixes
AUTOFIX_LIST += ("weblate_customization.autofix.FooFixer",)

# Add-ons
WEBLATE_ADDONS += ("weblate_customization.addons.ExamplePreAddon",)
```

Voir aussi :

Personnaliser les réparations automatiques, Rédiger ses propres contrôles, Écrire un greffon, Exécution de scripts à partir du greffon

### 2.23 Interface de gestion

L’interface de gestion offre des paramètres d’administration sous l’URL `/manage/`. Elle est disponible pour les utilisateurs connectés avec des privilèges d’administrateur, et est accessible en utilisant l’icône en forme de clé située en haut à droite:
Elle comprend un aperçu de base de votre Weblate :
— Statut de l’assistance, voir Getting support for Weblate
— Sauvegardes, voir Sauvegarder et déplacer Weblate
— Mémoire de traduction partagée, voir Mémoire de traduction
— Rapport de performance pour examiner l’état de Weblate et la longueur des files d’attente Celery
— Gestion des clés SSH, voir SSH repositories
— Vue d’ensemble des alertes pour tous les composants, voir alerts

2.23.1 L’interface d’administration Django

_Avertissement_ : Use with caution as this is a low level interface. You should not need it in most cases as most things are comfortably approachable through Weblate UI or API.

Ici vous pouvez gérer les objets enregistrés dans la base de données, comme les utilisateurs, traductions et autres paramètres :
### Webtrans administration

#### Site administration

- Reports
  - Weblate support status
  - Status of repositories
  - SSL keys
  - Performance report
- Translations memory

#### Accounts

- Audit log entries
- User profiles
- Verified e-mails

#### Auth tokens

- Tokens

#### Authentication

- Groups
- Roles
- Users

#### Billing plan

- Billing plans
- Customer billings
- Invoices

#### Points

- Point groups
- Points

#### Legal

- TOS agreements

#### Projects roles

- Projects roles
  - Contributors
  - Owners
  - Users

#### Screenshot

- Screenshots

#### Translation memory

- Translation memory entries

#### Weblate configuration

- Settings

#### Weblate languages

- Languages

#### Website translations

- Enhancements
  - Component data
  - Components
  - Contributor agreements
  - Projects

### Recent actions

- My actions
- None available
Dans la section *Reports*, vous pouvez vérifier le statut de votre site, le modifier pour la *Configuration de production*, où gérer les clefs SSH utilisées pour accéder aux *Accessing repositories*.

Gérer les objets de base de données sous n’importe quelle section. La plus intéressante est probablement *Weblate translations*, où vous pouvez gérer les projets à traduire, voir *Configuration du projet* et *Configuration des composants*.

guilabel : *Weblate languages* contient les définitions de langage, davantage expliquées dans *Définitions de langue*.

**Ajouter un projet**

Ajouter un projet sert de conteneur pour tous les composants. Généralement vous créez un projet pour un logiciel ou un livre (voir *Configuration du projet* pour plus d’infos sur les paramètres individuels) :

---

The Weblate Manual, Version 4.16

Dans la section *Reports*, vous pouvez vérifier le statut de votre site, le modifier pour la *Configuration de production*, où gérer les clefs SSH utilisées pour accéder aux *Accessing repositories*.

Gérer les objets de base de données sous n’importe quelle section. La plus intéressante est probablement *Weblate translations*, où vous pouvez gérer les projets à traduire, voir *Configuration du projet* et *Configuration des composants*.

guilabel : *Weblate languages* contient les définitions de langage, davantage expliquées dans *Définitions de langue*.

**Ajouter un projet**

Ajouter un projet sert de conteneur pour tous les composants. Généralement vous créez un projet pour un logiciel ou un livre (voir *Configuration du projet* pour plus d’infos sur les paramètres individuels) :

---
Voir aussi :

*Configuration du projet*

2.23. Interface de gestion
Composants bilingues

Une fois que vous avez ajouté un projet, des composants de traductions peuvent y être ajoutés. Voir *Configuration des composants* pour des infos concernant les paramètres individuels :
2.23. Interface de gestion
Voir aussi :

*Configuration des composants, Formats monolingues et bilingues*

**Composants monolingues**

Pour faciliter leur traduction, fournissez un fichier modèle contenant une représentation de message IDs du langage source (généralement l'anglais). Voir *Configuration des composants* pour en savoir plus sur les paramètres individuels :
## 2.23. Interface de gestion

<table>
<thead>
<tr>
<th>Tableau</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Interface de gestion</strong></td>
<td>Fonctionnalités d'administration des traductions.</td>
</tr>
<tr>
<td><strong>Langages</strong></td>
<td>Liste des langages gérés.</td>
</tr>
<tr>
<td><strong>Gestionnaire</strong></td>
<td>Interface pour gérer les projets.</td>
</tr>
<tr>
<td><strong>Affichage de traductions</strong></td>
<td>Fonctionnalités pour visualiser les traductions.</td>
</tr>
<tr>
<td><strong>Tâches</strong></td>
<td>Liste des tâches en cours et à venir.</td>
</tr>
</tbody>
</table>
Voir aussi :
*Configuration des composants, Formats monolingues et bilingues*

## 2.24 Getting support for Weblate

Weblate is copylefted libre software with community support. Subscribers receive priority support at no extra charge. Prepaid help packages are available for everyone. You can find more info about current support offerings at [https://weblate.org/support/](https://weblate.org/support/).

### 2.24.1 Intégration de l’assistance

Nouveau dans la version 3.8.

Purchased support packages can optionally be integrated into your Weblate subscription management interface, from where you will find a link to it. Basic instance details about your installation are also reported back to Weblate this way.

---

### 2.24.2 Data submitted to the Weblate

- URL where your Weblate instance is configured
- Your site title
- The Weblate version you are running
- Tallies of some objects in your Weblate database (projects, components, languages, source strings and users)
- The public SSH key of your instance

Additionally, when *Découverte Weblate* is turned on :
- List of public projects (name, URL and website)

No other data is submitted.
2.24.3 Intégration de services

— See if your support package is still valid
— Espace de sauvegarde provisionné par Weblate
— Découverte Weblate

Indication : Purchased support packages are already activated upon purchase, and can be used without integrating them.

2.24.4 Découverte Weblate

Nouveau dans la version 4.5.2.

Note : Cette fonctionnalité est actuellement en version bêta.

Discover Weblate is an opt-in service that makes it easier for users to find Weblate servers and communities. Users can browse registered services on <https://weblate.org/discover/>, and find there projects to contribute.

Getting listed

Indication : Participating in Discover Weblate makes Weblate submit some information about your server, please see Data submitted to the Weblate.

To list your server with an active support subscription (see Intégration de l’assistance) in Discover Weblate all you need to do is turn this on in the management panel:
Listing your server without a support subscription in Discover Weblate:

1. Register yourself at <https://weblate.org/user/>
2. Register your Weblate server in the discovery database at <https://weblate.org/subscription/discovery/>
3. Confirm the service activation in your Weblate and turn on the discovery listing in your Weblate management page using Enable discovery button:
Customizing listing

You can customize the listing by providing a text and image (570 x 260 pixels) at <https://weblate.org/user/>.

2.25 Documents juridiques

**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.
2.25.1 Licensing compliance

Weblate comes with REUSE 3.0 compliant license specification.

2.25.2 ITAR and other export controls

Weblate can be run within your own datacenter or virtual private cloud. As such, it can be used to store ITAR or other export-controlled information, however, end users are responsible for ensuring such compliance.

The Hosted Weblate service has not been audited for compliance with ITAR or other export controls, and does not currently offer the ability to restrict translations access by country.

2.25.3 US encryption controls

Weblate does not contain any cryptographic code, but might be subject export controls as it uses third party components utilizing cryptography for authentication, data-integrity and confidentiality.

Most likely Weblate would be classified as ECCN 5D002 or 5D992 and, as publicly available libre software, it should not be subject to EAR (see Encryption items NOT Subject to the EAR).

Software components used by Weblate (listing only components related to cryptographic function):

Python
- See https://wiki.python.org/moin/PythonSoftwareFoundationLicenseFaq#Is_Python_subject_to_export_laws.3F

GnuPG
- Utilisé en option par Weblate

Git
- Utilisé en option par Weblate

curl
- Used by Git

OpenSSL
- Used by Python and cURL

The strength of encryption keys depends on the configuration of Weblate and the third party components it interacts with, but in any decent setup it will include all export restricted cryptographic functions:
- In excess of 56 bits for a symmetric algorithm
- Factorisation of integers in excess of 512 bits for an asymmetric algorithm
- Computation of discrete logarithms in a multiplicative group of a finite field of size greater than 512 bits for an asymmetric algorithm
- Discrete logarithms in a group different than above in excess of 112 bits for an asymmetric algorithm

Weblate doesn’t have any cryptographic activation feature, but it can be configured in a way where no cryptography code would be involved. The cryptographic features include:
- Accessing remote servers using secure protocols (HTTPS)
- Generating signatures for code commits (PGP)

Voir aussi:

Export Controls (EAR) on Open Source Software
3.1 Contribuer à Weblate

There are dozens of ways to improve Weblate. You can choose the one you feel comfortable with, be it coding, graphics design, documentation, sponsorship, or an idea:

- Reporting issues in Weblate
- Commencer à contribuer au code de Weblate
- Contributing to Weblate modules
- Translating Weblate
- Contribute to Weblate documentation
- Weblate discussions
- Funding Weblate development

3.1.1 Translating Weblate

Weblate is continually being translated using Weblate itself. Feel free to take your part in the effort of making Weblate available in as many human languages as possible. It brings Weblate closer to its users!

If you find a possible mistake in the source string, you can mark it with a comment in the Weblate editor. This way, it can be discussed and corrected. If you’re certain, you can also click on the link in the Source string location section and submit a PR with your correction.

3.1.2 Contribute to Weblate documentation

You are welcome to improve the documentation page of your choice. Do it easily by clicking the Edit on GitHub button in the top-right corner of the page.

Please respect these guidelines while writing:

1. Don’t remove part of the documentation if it’s valid.
2. Use clear and easily-understandable language. You are writing tech docs, not a poem. Not all docs readers are native speakers, be thoughtful.
3. Don’t be afraid to ask if you are not certain. If you have to ask about some feature while editing, don’t change its docs before you have the answer. This means: You change or ask. Don’t do both at the same time.
4. Verify your changes by performing described actions while following the docs.
5. Send PR with changes in small chunks to make it easier and quicker to review and merge.
6. If you want to rewrite and change the structure of a big article, do it in two steps:
   1. Rewrite
   2. Once the rewrite is reviewed, polished, and merged, change the structure of the paragraphs in another PR.

**Indication:** You can translate the docs.

### 3.1.3 Extending built-in language definitions

The language definitions are in the `weblate-language-data` repository.
You are welcome to add missing language definitions to `languages.csv`, other files are generated from that file.

### 3.1.4 Weblate discussions

If you have an idea and not sure if it's suitable for an issue, don't worry. You can join the community in GitHub discussions.

### 3.1.5 Funding Weblate development

You can boost Weblate's development on the donate page. Funds collected there are used to enable gratis hosting for libre software projects and further development of Weblate. Please check the donate page for options, such as funding goals and the rewards you get as a proud funder.

**Supporters who have funded Weblate**

List of Weblate supporters:
- Yashiro Cc
- Cheng-Chia Tseng
- Timon Reinhard
- Cassidy James
- Loic Dachary
- Marozed
- https://freedombox.org/
- GNU Solidario (GNU Health)
- BallotReady
- Richard Nepithal
- MyExpenses.Mobi
- Michael Totschnig

Do you want to be in the list? Please see options on the Donate to Weblate.

### 3.2 Commencer à contribuer au code de Weblate

Understand the Weblate source code by going through Code source de Weblate, Interface de Weblate and Weblate internals.
3.2.1 Starting with the codebase

Familiarize yourself with the Weblate codebase, by having a go at the bugs labelled good first issue.

You are welcome to start working on these issues without asking. Just announce that in the issue, so that it’s clear that somebody is working on that issue.

3.2.2 Running Weblate locally

The most comfortable approach to get started with Weblate development is to follow Installing from sources. It will get you a virtualenv with editable Weblate sources.

1. Clone the Weblate source code:

   ```
git clone https://github.com/WeblateOrg/weblate.git
cd weblate
   ```

2. Create a virtualenv:

   ```
virtualenv .venv
.venv/bin/activate
   ```

3. Install Weblate (for this you need some system dependencies, see Installing from sources):

   ```
pip install -e .
   ```

3. Install all dependencies useful for development:

   ```
pip install -r requirements-dev.txt
   ```

4. Start a development server:

   ```
weblate runserver
   ```

5. Depending on your configuration, you might also want to start Celery workers:

   ```
./weblate/examples/celery start
   ```

6. To run a test (see Local testing for more details):

   ```
./scripts/test-database.sh
./manage.py test
   ```

Voir aussi :

Installing from sources

3.2.3 Running Weblate locally in Docker

If you have Docker and docker-compose installed, you can spin up the development environment by simply running:

```
./rundev.sh
```

It will create a development Docker image and start it. Weblate is running on http://127.0.0.1:8080/ and you can sign in as the user admin using admin as the password. The new installation is empty, so you might want to continue with Adding translation projects and components.

The Dockerfile and docker-compose.yml for this are located in the dev-docker directory.

The script also accepts some parameters, to execute tests, run it with the test parameter and then specify any test parameters, for example running only tests in the weblate.machine module:
Note: Be careful that your Docker containers are up and running before running the tests. You can check that by running the `docker ps` command.

To display the logs:

```
./rundev.sh logs
```

To stop the background containers, run:

```
./rundev.sh stop
```

Running the script without arguments will re-create the Docker container and restart it.

Note: This is not a suitable setup for production, as it includes several hacks which are insecure, but they make development easier.

### 3.2.4 Bootstrapping your devel instance

You might want to use `import_demo` to create demo translations and `createadmin` to make an admin user.

### 3.2.5 Coding Weblate with PyCharm

PyCharm is a known IDE for Python, here are some guidelines to help you set up your Weblate project in it.

Considering you have just cloned the GitHub repository to a folder, just open it with PyCharm. Once the IDE is open, the first step is to specify the interpreter you want to use:

You can either choose to let PyCharm create the virtualenv for you, or select an already existing one:
Don’t forget to install the dependencies once the interpreter is set: Either through the console (the console from the IDE will directly use your virtualenv by default), or through the interface when you get a warning about missing dependencies.

The second step is to set the right info to use Django natively inside PyCharm: The idea is to be able to immediately trigger the unit tests in the IDE. For that you need to specify the root path of the Django project and the path to its settings:
Be careful, the *Django project root* is the actual root of the repository, not the Weblate sub-directory. About the settings, you could use the `weblate/settings_test.py` from the repository, but you could create your own setting and set it there.

The last step is to run the server and to put breakpoints in the code to be able to debug it. This is done by creating a new *Django Server* configuration:

![Django Server configuration](image)
Indication: Be careful with the property called No reload: It prevents the server from being reloaded live if you modify files. This allows the existing debugger breakpoints to persist, when they normally would be discarded upon reloading the server.

3.3 Code source de Weblate

Weblate est développé sur GitHub. Vous êtes bienvenu pour bifurquer le code et ouvrir des demandes de tirage. Les correctifs sous toute autre forme sont également les bienvenus.

Voir aussi:
Check out Weblate internals to see how Weblate looks from inside.

3.3.1 Coding guidelines

Any code for Weblate should be written with Security by Design Principles in mind.

Any code should come with documentation explaining the behavior. Don’t forget documenting methods, complex code blocks, or user visible features.

Any new code should utilize PEP 484 type hints. We’re not checking this in our CI yet as existing code does not yet include them.
3.3.2 Coding standard and linting the code

The code should follow PEP-8 coding guidelines and should be formatted using black code formatter.

To check the code quality, you can use flake8, the recommended plugins are listed in .pre-commit-config.yml and its configuration is placed in setup.cfg.

The easiest approach to enforce all this is to install pre-commit. The repository contains configuration for it to verify the committed files are sane. After installing it (it is already included in the requirements-lint.txt) turn it on by running pre-commit install in Weblate checkout. This way all your changes will be automatically checked.

You can also trigger check manually, to check all files run:

```
pre-commit run --all
```

3.4 Déboguer Weblate

Bugs can behave as application crashes or as various misbehavior. You are welcome to collect info on any such issue and submit it to the issue tracker.

3.4.1 Mode de débogage

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.

Voir aussi :

Disable debug mode, Properly configure admins, Collecting error reports

3.4.2 Logs de Weblate

Weblate can produce detailed logs of what is going on in the background. In the default configuration it uses syslog and that makes the log appear either in /var/log/messages or /var/log/syslog (depending on your syslog daemon configuration).

The Celery process (see Background tasks using Celery) usually produces its own logs as well. The example systemwide 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.

Voir aussi :

Configuration d’exemple contient la configuration LOGGING.
3.4.3 Not processing background tasks

A lot of things are done in the background by Celery workers. If things like sending out e-mails or component removal does not work, there might be a related issue.

Choses à vérifier dans ce cas :
— Check that the Celery process is running, see Background tasks using Celery
— Check the Celery queue status, either in Interface de gestion, or using celery_queues
— Look in the Celery logs for errors (see Logs de Weblate)

3.4.4 Not receiving e-mails from Weblate

You can verify whether outgoing e-mail is working correctly by using the sendtestemail management command (see Invoking management commands for instructions on how to invoke it in different environments) or by using Interface de gestion under the Tools tab.

These send e-mails directly, so this verifies that your SMTP configuration is correct (see Configuring outgoing e-mail). Most of the e-mails from Weblate are however sent in the background and there might be some issues with Celery involved as well, please see Not processing background tasks for debugging that.

3.4.5 Analyzing application crashes

In case the application crashes, it is useful to collect as much info about the crash as possible. This can be achieved by using third-party services which can collect such info automatically. You can find info on how to set this up in Collecting error reports.

3.4.6 Échecs silencieux

Lots of tasks are offloaded to Celery for background processing. Failures are not shown in the user interface, but appear in the Celery logs. Configuring Collecting error reports helps you to notice such failures easier.

3.4.7 Problèmes de performance

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 dogsnow along with Collecting error reports and get pinpointed and detailed tracebacks in the error collection tool.

In case the slow performance is linked to the database, you can also enable logging of all database queries using following configuration after enabling DEBUG:

```
LOGGING["loggers"]['django.db.backends'] = {"handlers": ["console"], "level": "DEBUG"}
```
3.5 Weblate internals

Note: This chapter will give you basic overview of Weblate internals.

Weblate derives most of its code structure from, and is based on Django.

3.5.1 Directory structure

Quick overview of directory structure of Weblate main repository:

docs
  Source code for this documentation, which can be built using Sphinx.

dev-docker
  Docker code to run development server, see Running Weblate locally in Docker.

weblate
  Source code of Weblate as a Django application, see Weblate internals.

weblate/static
  Client files (CSS, Javascript and images), see Interface de Weblate.

3.5.2 Modules

Weblate consists of several Django applications (some optional, see Modules Weblate optionnels):

accounts
  User account, profiles and notifications.

addons
  Add-ons to tweak Weblate behavior, see Extensions.

api
  API based on Django REST framework.

auth
  Authentication and permissions.

billing
  The optional Facturation module.

checks
  Translation string Quality checks module.

fonts
  Font rendering checks module.

formats
  File format abstraction layer based on translate-toolkit.

gitexport
  The optional Exportateur Git module.

lang
  Module defining language and plural models.

legal
  The optional Mentions légales module.

machinery
  Integration of machine translation services.

memory
Mémoire de traduction intégré, voir *Mémoire de traduction*.

screenshots

Screenshots management and OCR module.

trans

Main module handling translations.

utils

Various helper utilities.

vcs

Version control system abstraction.

wladmin

Django admin interface customization.

## 3.6 Développement de greffons

Extensions are a way to customize localization workflow in Weblate.

```python
class weblate.addons.base.BaseAddon(storage=None)
    Base class for Weblate add-ons.

classmethod can_install(component, user)
    Check whether add-on is compatible with given component.

classmethod get_add_form(user, component, **kwargs)
    Return configuration form for adding new add-on.

classmethod get_settings_form(user, **kwargs)
    Return configuration form for this add-on.

post_add(translation)
    Crochet déclenché lors de l’ajout d’une nouvelle traduction.

post_commit(component)
    Crochet déclenché après que des modifications aient été archivées dans le dépôt.

post_push(component)
    Crochet déclenché après que le dépôt est poussé en amont.

post_update(component, previous_head : str, skip_push : bool)
    Hook triggered after repository is updated from upstream.

    Paramètres
    — previous_head(str) – HEAD of the repository prior to update, can be blank on initial clone.
    — skip_push(bool) – Whether the add-on operation should skip pushing changes upstream. Usually you can pass this to underlying methods as commit_and_push or commit_pending.

pre_commit(translation, author)
    Crochet déclenché avant que des modifications aient été archivées dans le dépôt.

pre_push(component)
    Hook triggered before repository is pushed upstream.

pre_update(component)
    Hook triggered before repository is updated from upstream.
```
The Weblate Manual, Version 4.16

save_state()
Save add-on state information.

store_post_load(translation, store)
Crochet déclenché après l'analyse d'un fichier.
It receives an instance of a file format class as a argument.
This is useful to modify file format class parameters, for example adjust how the file will be saved.

unit_pre_create(unit)
Hook triggered before new unit is created.

Here is an example add-on:

```python
# Copyright © Michal Čihař <michal@weblate.org>
#
# SPDX-License-Identifier: GPL-3.0-or-later

from django.utils.translation import gettext_lazy as _

from weblate.addons.base import BaseAddon
from weblate.addons.events import EVENT_PRE_COMMIT

class ExampleAddon(BaseAddon):
    # Filter for compatible components, every key is
    # matched against property of component
    compat = {"file_format": ["po", "po-mono"]}
    # List of events add-on should receive
    events = (EVENT_PRE_COMMIT,)
    # Add-on unique identifier
    name = "weblate.example.example"
    # Verbose name shown in the user interface
    verbose = _("Example add-on")
    # Detailed add-on description
    description = _("This add-on does nothing it is just an example.")

    # Callback to implement custom behavior
    def pre_commit(self, translation, author):
        return
```

3.7 Interface de Weblate
The frontend is currently built using Bootstrap, jQuery and few third party libraries.

3.7.1 Navigateurs pris en charge

Weblate supports the latest, stable releases of all major browsers and platforms.

Alternative browsers which use the latest version of WebKit, Blink, or Gecko, whether directly or via the platform’s web view API, are not explicitly supported. However, Weblate should (in most cases) display and function correctly in these browsers as well.

Older browsers might work, but some features might be limited.
3.7.2 Gestion des dépendances

The yarn package manager is used to update third party libraries. The configuration lives in `scripts/yarn` and there is a wrapper script `scripts/yarn-update` to upgrade the libraries, build them and copy to correct locations in `weblate/static/vendor`, where all third party frontend code is located. The Weblate specific code should be placed directly in `weblate/static` or feature specific subdirectories (for example `weblate/static/editor`).

Adding new third-party library typically consists of:

```bash
# Add a yarn package
yarn --cwd scripts/yarn add PACKAGE
# Edit the script to copy package to the static folder
edit scripts/yarn-update
# Run the update script
./scripts/yarn-update
# Add files to git
git add .
```

3.7.3 Coding style

Weblate relies on Prettier for the code formatting for both JavaScript and CSS files.

We also use ESLint to check the JavaScript code.

3.7.4 Traduction

Should you need any user visible text in the frontend code, it should be localizable. In most cases all you need is to wrap your text inside `gettext` function, but there are more complex features available:

```javascript
// Add a yarn package
yarn --cwd scripts/yarn add PACKAGE
// Edit the script to copy package to the static folder
edit scripts/yarn-update
// Run the update script
./scripts/yarn-update
// Add files to git
git add .
```

3.7.5 Icônes

Weblate currently uses material design icons. In case you are looking for new symbol, check Material Design Icons or Material Design Resources.

Additionally, there is `scripts/optimize-svg` to reduce size of the SVG as most of the icons are embedded inside the HTML to allow styling of the paths.
3.8 Reporting issues in Weblate

Le système de suivi de Weblate est hébergé sur GitHub.

Feel welcome to report any issues you have, or suggest improvement for Weblate there. There are various templates prepared to comfortably guide you through the issue report.

If what you have found is a security issue in Weblate, please consult the Problèmes de sécurité section below.

If you are not sure about your bug report or feature request, you can try Weblate discussions.

3.8.1 Problèmes de sécurité

In order to give the community time to respond and upgrade, you are strongly urged to report all security issues privately. HackerOne is used to handle security issues, and can be reported directly at HackerOne. Once you submit it there, community has limited but enough time to solve the incident.

Alternatively, report to security@weblate.org, which ends up on HackerOne as well.

If you don’t want to use HackerOne, for whatever reason, you can send the report by e-mail to michal@weblate.org. You can choose to encrypt it using this PGP key 3CB 1DF1 EF12 CF2A C0EE 5A32 9C27 B313 42B7 511D. You can also get the PGP key from Keybase.

Note: Weblate depends on third-party components for many things. In case you find a vulnerability affecting one of those components in general, please report it directly to the respective project.

Some of these are:
- Django
- Django REST framework
- Python Social Auth

3.9 Weblate testsuite and continuous integration

Testsuites exist for most of the current code, increase coverage by adding testcases for any new functionality, and verify that it works.

3.9.1 Intégration continue

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.sh:

```bash
# Copyright © Michal Čihař <michal@weblate.org>
#
# SPDX-License-Identifier: GPL-3.0-or-later
```

(suite sur la page suivante)
The simple way to configure test database from environment

```bash
# Database backend to use postgresql / mysql / mariadb
export CI_DATABASE="$\{1:-postgresql\}"
```

```bash
# Database server configuration
export CI_DB_USER=weblate
export CI_DB_PASSWORD=weblate
export CI_DB_HOST=127.0.0.1
```

```bash
# Django settings module to use
export DJANGO_SETTINGS_MODULE=weblate.settings_test
```

The simple execution can look like:

```bash
. scripts/test-database.sh
./ci/run-migrate
./ci/run-test
./ci/run-docs
```

### 3.9.2 Local testing

To run a testsuite locally, use:

```
DJANGO_SETTINGS_MODULE=weblate.settings_test ./manage.py test
```

**Indication**: You will need a database (PostgreSQL) server to be used for tests. By default Django creates separate database to run tests with test_prefix, so in case your settings is configured to use weblate, the tests will use test_weblate database. See [Database setup for Weblate](#) for setup instructions.

The `weblate/settings_test.py` is used in CI environment as well (see [Intégration continue](#)) and can be tuned using environment variables:

```bash
# Copyright © Michal Čihař <michal@weblate.org>
#
# SPDX-License-Identifier: GPL-3.0-or-later
#
# Simple way to configure test database from environment
# shellcheck shell=sh
#
# Database backend to use postgresql / mysql / mariadb
export CI_DATABASE="$\{1:-postgresql\}"
```

```bash
# Database server configuration
export CI_DB_USER=weblate
export CI_DB_PASSWORD=weblate
export CI_DB_HOST=127.0.0.1
```

```bash
# 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:
You can also specify individual tests to run:

```
DJANGO_SETTINGS_MODULE=weblate.settings_test ./manage.py test weblate.gitexport
```

**Indication**: The tests can also be executed inside developer docker container, see *Running Weblate locally in Docker*.

**Voir aussi**:

See *Les tests dans Django* for more info on running and writing tests for Django.

### 3.10 Schémas de données

Weblate utilise JSON Schema pour définir la disposition des fichiers JSON externes.

#### 3.10.1 Schéma de mémoire des traductions Weblate

<table>
<thead>
<tr>
<th>type</th>
<th>tableau</th>
<th>Elément de mémoire de traduction</th>
<th>objet</th>
</tr>
</thead>
<tbody>
<tr>
<td>items</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **category**
  - Catégorie de la chaîne
  - 1 = global, 2 = partagé, 10000000 et plus = spécifique à un projet, 20000000 et plus = spécifique à un utilisateur
  - type
  - exemple
  - minimum
  - par défaut

- **origine**
  - Origine de la chaîne
  - Nom de fichier ou nom du composant
  - type
  - exemple
  - par défaut

- **source**
  - Chaîne source
  - type
  - exemple
  - minLength
  - par défaut

- **source_language**
  - Langue source
  - ISO 639-1 / ISO 639-2 / IETF BCP 47
  - type
  - exemple
  - motif
  - par défaut

- **cible**
  - Chaîne cible
  - type
  - exemple
  - minLength
  - par défaut

suite sur la page suivante
Tableau 1 – suite de la page précédente

<table>
<thead>
<tr>
<th>cible_language</th>
<th>Langue cible</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISO 639-1 / ISO 639-2 / IETF BCP 47</td>
<td>chaîne de caractères</td>
</tr>
<tr>
<td>exemples</td>
<td>cs</td>
</tr>
<tr>
<td>motif</td>
<td>^[^ ]*$</td>
</tr>
<tr>
<td>par défaut</td>
<td></td>
</tr>
</tbody>
</table>

additionalProperties | Faux |

définitions

Voir aussi :

Mémoire de traduction, dump_memory, import_memory

3.10.2 Export des données utilisateur Weblate

https://weblate.org/schemas/weblate-userdata.schema.json

type objet

— de base

<table>
<thead>
<tr>
<th>nom_utilisateur</th>
<th>Nom d’utilisateur</th>
</tr>
</thead>
<tbody>
<tr>
<td>type chaîne de caractères</td>
<td></td>
</tr>
<tr>
<td>exemples</td>
<td>admin</td>
</tr>
<tr>
<td>par défaut</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>full_name</th>
<th>Nom complet</th>
</tr>
</thead>
<tbody>
<tr>
<td>type chaîne de caractères</td>
<td></td>
</tr>
<tr>
<td>exemples</td>
<td>Administrateur Weblate</td>
</tr>
<tr>
<td>par défaut</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>email</th>
<th>Adresse courriel</th>
</tr>
</thead>
<tbody>
<tr>
<td>type chaîne de caractères</td>
<td></td>
</tr>
<tr>
<td>exemples</td>
<td><a href="mailto:noreply@example.com">noreply@example.com</a></td>
</tr>
<tr>
<td>par défaut</td>
<td></td>
</tr>
<tr>
<td>format</td>
<td>courriel</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>date_joined</th>
<th>Date d’adhésion</th>
</tr>
</thead>
<tbody>
<tr>
<td>type chaîne de caractères</td>
<td></td>
</tr>
<tr>
<td>exemples</td>
<td>2019-11-18T18:53:54.862Z</td>
</tr>
<tr>
<td>par défaut</td>
<td></td>
</tr>
<tr>
<td>format</td>
<td>date-time</td>
</tr>
</tbody>
</table>

— profile

additionalProperties | Faux |

<table>
<thead>
<tr>
<th>langue</th>
<th>Langue</th>
</tr>
</thead>
<tbody>
<tr>
<td>type chaîne de caractères</td>
<td></td>
</tr>
<tr>
<td>exemples</td>
<td>cs</td>
</tr>
<tr>
<td>motif</td>
<td>^[^ ]*$</td>
</tr>
<tr>
<td>par défaut</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>suggéré</th>
<th>Nombre de chaînes suggérées</th>
</tr>
</thead>
<tbody>
<tr>
<td>type entier</td>
<td></td>
</tr>
<tr>
<td>exemples</td>
<td>1</td>
</tr>
<tr>
<td>par défaut</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>traduit</th>
<th>Nombre de chaînes traduites</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

suite sur la page suivante
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>entier</td>
<td>Nombre de captures d'écran téléversées</td>
</tr>
<tr>
<td>0</td>
<td>par défaut</td>
</tr>
<tr>
<td>chaîne de caractères</td>
<td>Lien vers l'éditeur</td>
</tr>
<tr>
<td>chaîne de caractères</td>
<td>Mode pour l'éditeur de traductions</td>
</tr>
<tr>
<td>chaîne de caractères</td>
<td>Mode Zen</td>
</tr>
<tr>
<td>chaîne de caractères</td>
<td>Caractères spéciaux</td>
</tr>
<tr>
<td>chaîne de caractères</td>
<td>Affichage du tableau de bord par défaut</td>
</tr>
<tr>
<td>chaîne de caractères</td>
<td>Liste de composants par défaut</td>
</tr>
<tr>
<td>tableau</td>
<td>Langues traduites</td>
</tr>
<tr>
<td>tableau</td>
<td>Langues secondaires</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>booléen</td>
<td>Masquer les traductions terminées sur le tableau de bord</td>
</tr>
<tr>
<td>Vrai</td>
<td>par défaut</td>
</tr>
<tr>
<td>Vrai</td>
<td>Afficher les traductions secondaires en mode Zen</td>
</tr>
<tr>
<td>Vrai</td>
<td>Masquer la source lorsqu'une traduction secondaire existe</td>
</tr>
<tr>
<td>Faux</td>
<td>par défaut</td>
</tr>
<tr>
<td>Vrai</td>
<td>Masquer les traductions terminées sur le tableau de bord</td>
</tr>
<tr>
<td>Vrai</td>
<td>24 exemples</td>
</tr>
<tr>
<td>1</td>
<td>1 exemples</td>
</tr>
<tr>
<td>0</td>
<td>0 exemples</td>
</tr>
<tr>
<td>0</td>
<td>0 exemples</td>
</tr>
<tr>
<td>^.*$</td>
<td>chaîne de caractères</td>
</tr>
<tr>
<td>^.*$</td>
<td>chaîne de caractères</td>
</tr>
<tr>
<td>^.*$</td>
<td>chaîne de caractères</td>
</tr>
<tr>
<td>^.*$</td>
<td>chaîne de caractères</td>
</tr>
<tr>
<td>nul</td>
<td>Liste de composants par défaut</td>
</tr>
<tr>
<td>nul</td>
<td>Affichage du tableau de bord par défaut</td>
</tr>
<tr>
<td>cs</td>
<td>Langues traduites</td>
</tr>
<tr>
<td>^.*$</td>
<td>chaîne de caractères</td>
</tr>
<tr>
<td>^.*$</td>
<td>chaîne de caractères</td>
</tr>
<tr>
<td>^.*$</td>
<td>chaîne de caractères</td>
</tr>
<tr>
<td>^.*$</td>
<td>chaîne de caractères</td>
</tr>
</tbody>
</table>
### Tableau 2 – suite de la page précédente

<table>
<thead>
<tr>
<th>items</th>
<th>Code langue</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>chaîne de caractères</td>
</tr>
<tr>
<td>exemples</td>
<td>sk</td>
</tr>
<tr>
<td>motif</td>
<td>^.*$</td>
</tr>
<tr>
<td>par défaut</td>
<td></td>
</tr>
</tbody>
</table>

#### — surveillés

**Projets surveillés**

<table>
<thead>
<tr>
<th>type</th>
<th>tableau</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>items</th>
<th>Identifiant du projet</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>chaîne de caractères</td>
</tr>
<tr>
<td>exemples</td>
<td>weblate</td>
</tr>
<tr>
<td>motif</td>
<td>^.*$</td>
</tr>
<tr>
<td>par défaut</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>additionalProperties</th>
<th>Faux</th>
</tr>
</thead>
</table>

#### — audilog

**Journal d’audit**

<table>
<thead>
<tr>
<th>type</th>
<th>tableau</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>items</th>
<th>Éléments</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>objet</td>
</tr>
<tr>
<td>properties</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>— address</th>
<th>Adresse IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>chaîne de caractères</td>
</tr>
<tr>
<td>exemples</td>
<td>127.0.0.1</td>
</tr>
<tr>
<td>motif</td>
<td>^.*$</td>
</tr>
<tr>
<td>par défaut</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>— user_agent</th>
<th>User agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>chaîne de caractères</td>
</tr>
<tr>
<td>exemples</td>
<td>PC / Linux / Firefox 70.0</td>
</tr>
<tr>
<td>motif</td>
<td>^.*$</td>
</tr>
<tr>
<td>par défaut</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>— horodatage</th>
<th>Horodatage</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>chaîne de caractères</td>
</tr>
<tr>
<td>exemples</td>
<td>2019-11-18T18:58:30.845Z</td>
</tr>
<tr>
<td>par défaut</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>— activité</th>
<th>Activité</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>chaîne de caractères</td>
</tr>
<tr>
<td>exemples</td>
<td>connexion</td>
</tr>
<tr>
<td>motif</td>
<td>^.*$</td>
</tr>
<tr>
<td>par défaut</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>additionalProperties</th>
<th>Faux</th>
</tr>
</thead>
</table>

### Voir aussi :

*Profil utilisateur, dumpuserdata*
3.11 Publication de Weblate

3.11.1 Calendrier de publication

Weblate has two month release cycle for releases (x.y). These are usually followed by a bunch of bugfix releases to fix issues which slip into them (x.y.z).

The change in the major version indicates that the upgrade process can not skip this version - you always have to upgrade to x.0 before upgrading to higher x.y releases.

Voir aussi :
Mise à niveau de Weblate

3.11.2 Release planning

The features for upcoming releases are collected using GitHub milestones, you can see our roadmap at <https://github.com/WeblateOrg/weblate/milestones>.

3.11.3 Release process

Choses à vérifier avant publication :
1. Vérifier les langues nouvellement traduites à l'aide de ./scripts/list-translated-languages.
2. Définir la version finale à l'aide de ./scripts/prepare-release.
3. Make sure screenshots are up to date make -j 12 -C docs update-screenshots.
4. Merge any possibly pending translations wlc push; git remote update; git merge origin/weblate

Effectuer la publication :
5. Créer une version ./scripts/create-release --tag (voir ci-dessous pour les exigences).

Étapes manuelles après publication :
7. Fermer le jalon GitHub.
8. Une fois l'image Docker testée, ajouter un libellé, puis pousser la nouvelle version.
9. Mettre à jour le graphique Helm vers la nouvelle version.
10. Inclure la nouvelle version dans .github/workflows/migrations.yml pour la couvrir dans les tests de migration.
12. Incrémenter le numéro de version dans le dépôt à l'aide de ./scripts/set-version.
13. Check that readthedocs.org did build all translations of the documentation using ./scripts/rtd-projects.

Pour créer des libellés à l'aide du script ./scripts/create-release, vous avez besoin de éléments suivants :
— GnuPG avec la clé privée utilisée pour signer la version
— Accès pour pousser sur le dépôt Git Weblate (pour pousser les balises)
— L'outil :command:`hub` configuré pour accéder et créer des communiés sur le dépôt Weblate
— Accès SSH au serveur de téléchagement Weblate (les téléchargements du site Web y sont copiés)
3.12 Sécurité et confidentialité

Astuces : Chez Weblate, la sécurité maintient un environnement qui respecte la vie privée de nos utilisateurs.

Development of Weblate adheres to the Best Practices of the Linux Foundation’s Core Infrastructure Initiative.

Voir aussi :

Problèmes de sécurité

3.12.1 Security updates

Only the latest release is guaranteed to receive security updates.

3.12.2 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.

Indication : There might be vulnerabilities in third-party libraries which do not affect Weblate, so those are not addressed by releasing bugfix versions of Weblate.

3.12.3 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).

Voir aussi :

Intégration continue

3.13 Contributing to Weblate modules

Besides the main repository, Weblate consists of several Python modules. All these follow same structure and this documentation covers them all.

For example, this covers :

— wlc, Python client library, see Client Weblate
— translation-finder, used to discover translatable files in the repository
— language-data, définitions de langage pour Weblate, voir Définitions de langue
3.13.1 Coding guidelines

Any code for Weblate should be written with Security by Design Principles in mind. Any code should come with documentation explaining the behavior. Don’t forget documenting methods, complex code blocks, or user visible features. Any new code should utilize PEP 484 type hints. We’re not checking this in our CI yet as existing code does not yet include them.

3.13.2 Running tests

The tests are executed using **py.test**. First you need to install test requirements:

```bash
pip install -r requirements-test.txt
```

You can then execute the testsuite in the repository checkout:

```bash
py.test
```

**Voir aussi:**

The CI integration is very similar to *Weblate testsuite and continuous integration*.

3.13.3 Coding standard and linting the code

The code should follow PEP-8 coding guidelines and should be formatted using **black** code formatter.

To check the code quality, you can use **flake8**, the recommended plugins are listed in .pre-commit-config.yaml and its configuration is placed in setup.cfg.

The easiest approach to enforce all this is to install **pre-commit**. The repository contains configuration for it to verify the committed files are sane. After installing it (it is already included in the requirements-lint.txt) turn it on by running **pre-commit install** in Weblate checkout. This way all your changes will be automatically checked.

You can also trigger check manually, to check all files run:

```bash
pre-commit run --all
```

**Voir aussi:**

*Code source de Weblate*

3.14 À propos de Weblate

3.14.1 Objectifs du projet

Web-based continuous localization tool with tight Integration avec le système de contrôle de versions supporting a wide range of file formats, making it easy for translators to contribute.
3.14.2 Nom du projet

« Weblate » est un mot valise formé à partir des mots « web » et « translate ».

3.14.3 Site Web du projet

The landing page is [https://weblate.org](https://weblate.org) and there is a cloud-hosted service at [https://hosted.weblate.org](https://hosted.weblate.org). The documentation can be read at [https://docs.weblate.org](https://docs.weblate.org).

3.14.4 Logos du projet


3.14.5 Direction

This project is maintained by Michal Čihař, who can be reached at michal@weblate.org.

3.14.6 Auteurs

Weblate a été lancé par Michal Čihař. Depuis sa création en 2012, des milliers de personnes y ont contribué.

3.15 Licence

More detailed licensing information is available in the Weblate source code and follows REUSE 3.0 specification.

Copyright © Michal Čihař michal@weblate.org

Ce programme est un logiciel libre : vous pouvez le redistribuer ou le modifier suivant les termes de la GNU General Public License telle que publiée par la Free Software Foundation, soit la version 3 de la licence, soit (à votre gré) toute version ultérieure.

Ce programme est distribué dans l’espoir qu’il sera utile, mais SANS AUCUNE GARANTIE ; sans même la garantie tacite de QUALITÉ MARCHANDE ou d’ADÉQUATION à UN BUT PARTICULIER. Consultez la GNU General Public License pour plus de détails.

Vous devez avoir reçu une copie de la GNU General Public License en même temps que ce programme ; si ce n’est pas le cas, consultez <https://www.gnu.org/licenses/>.
CHAPITRE 4

Historique des modifications

4.1 Weblate 4.16

Released on March 1st 2023.

- Format string checks now also detects duplicated formats.
- Improved search performance for some specially formatted strings.
- Celery beat is now storing the tasks schedule in the database.
- Added support for IBM Watson Language Translator.
- Dropped support for VCS integration settings deprecated in 4.14.
- Added support for Bitbucket Server pull requests.
- Improved conflicts handling in gettext PO files.
- Added support for defining strings state when adding via API.
- Added support for configuring CORS allowed origins.
- Added plurals support to automatic suggestions.

All changes in detail.

4.2 Weblate 4.15.2

 Released on January 25th 2023.

- Enabled gotext JSON and i18next v4 formats in the default configuration.
- Fixed crash on uploading corrupted files.
- Show stale directories in Git repository status.

All changes in detail.
4.3 Weblate 4.15.1

Released on January 19th 2023.

— Fixed suggestions from automatic translation.
— Fixed add-on page crash in some corner cases.
— Fixed untranslating template for new translations in some cases.
— Documented licensing using REUSE 3.0.
— Fixed users pagination on team management.
— Improved performance of project creation and saving.
— Added support for gotext JSON files.
— Added support for i18next v4 files.
— Pagination in the API is now customizable.

All changes in detail.

4.4 Weblate 4.15

Released on December 16th 2022.

— Added support for browsing changes for a individual string.
— Fixed plurals handling in automatic translation from other components.
— Added keyboard shortcut Alt+Enter to submit string as a suggestion.
— Added support for placeables in the Fluent format.
— Improved performance of translation memory.
— Autogenerate repoweb browsing links for well known code hosting services.
— Improved performance of several views.
— Improved listing of strings with plurals.
— Added support for adding custom markup to HTML head.
— Fixed generation of MO files in the add-on to include only translated files.
— Fixed rendering of regular expression flags.
— Improved placeholders check behavior with plurals.
— Added support for translation files naming suitable for Google Play.
— Added support for labels in API.
— Added support for choosing different e-mail for commits than for notifications.
— The Docker image no longer enables debug mode by default.
— Order glossary terms based on the glossary component priority.
— Added team administrators who can add or remove members of the team.
— Added a popup confirmation before deleting users.
— Added add-on to customize XML output.

All changes in detail.

4.5 Weblate 4.14.2

Released on November 5th 2022.

— Added support for removing entries from translation memory.
— Improved analysis on the duplicate language alert.
— Improved accuracy of the consecutive duplicated words check.
— Improved scaling of sending many notifications.
— Improved string state handling for subtitle translation.
— Deprecated insecure configuration of VCS service API keys via _TOKEN/_USERNAME configuration instead of _CREDENTIALS list.
— Fixed processing of some uploaded CSV files.
— Improved whitespace changes handling in diff display.
— Added automatic suggestions management link to management pages.
— Track comment removal/resolving in history.
— Fixed restoring project backups with linked components.
— Fixed captcha entering on unsuccessful registration.
— Improved language support in DeepL.
— Improved webhooks compatibility with authenticated repositories.
— Added support for Python 3.11.

All changes in detail.

4.6 Weblate 4.14.1

Released on September 15th 2022.
— Fixed generating project backups in some situations.
— Improved error reporting on file upload.
— Fetch all user verified e-mails from GitHub during authentication.
— Avoid matching glossary terms on context or keys.
— Added notifications for string removals.
— Improved management of untranslatable terms in glossary.
— List number of team members on team management page.
— Add group management interface.
— Always show review stats when reviews are enabled.
— Added searching support in units API.
— Fixed progress bar display for read-only strings in the review workflow.
— Improved Burmese punctuation check.
— Fixed garbage collecting of metrics data.

All changes in detail.

4.7 Weblate 4.14

Released on August 22nd 2022.
— Track add-on changes in a history.
— Fixed parsing translation from Windows RC, HTML and text files.
— Extended language code style configuration options.
— Added support for plurals updated in the recent CLDR releases.
— Reduced memory usage while updating components with a lot of translations.
— Added support for translation domain in SAP Translation Hub.
— Allow absolute links in source string locations.
— Improved operation behind some reverse proxies.
— Extension de l’API pour couvrir la mémoire de traduction.
— Improved document translation workflow.
— Improved reliability of HTML and text files translation.
— Added support for project level backups.
— Improved performance and memory usage of translation memory lookups.

All changes in detail.

4.8 Weblate 4.13.1

Released on July 1st 2022.
— Fixed tracking suggestions in history.
— Fixed parsing reverse proxy info from Cloudflare.
— Make parse error lock a component from translating.
— Fixed configuring intermediate file in the discovery add-on.
— Fixed DeepL translations behavior with placeholders.
— Fixed untranslating strings via API.
— Added support for removing user from a group via API.
— Fixed audit log for user invitation e-mails.
— Fixed flag names for Java formatting strings.

All changes in detail.

### 4.9 Weblate 4.13

Released on June 15th 2022.
— Changed behavior of updating language names.
— Added pagination to projects listing.
— API for creating new units now returns information about newly created unit.
— La découverte de composants supporte maintenant la configuration d’un langage intermédiaire.
— Added fixed encoding variants to CSV formats.
— Changed handling of context and location for some formats to better fit underlying implementation.
— Added support for ResourceDictionary format.
— Improved progress bar colors for color blind.
— Fixed variants cleanup on string removal.
— Compatibility with Django 4.1.
— Added support for storing escaped XML elements in XLIFF.
— Improved formatting of placeholder check errors.
— Redirect /well-known/change-password to /accounts/password/.
— Machine translation services are now configurable per project.
— Added separate permission for resolving comments and grant it to the Review strings role.
— Added support for storing alternative translations in the CSV file.
— The placeholders check can now be case-insensitive as well.

All changes in detail.

### 4.10 Weblate 4.12.2

Released on May 11th 2022.
— Fixed rebuilding project translation memory for some components.
— Correction du tri des composants par chaînes non traduites.
— Fixed possible loss of translations while adding new language.
— Ensure Weblate SSH key is generated during migrations.

All changes in detail.

### 4.11 Weblate 4.12.1

Released on April 29th 2022.
— Fixed pull request message title.
— Improved syntax error handling in Fluent format.
— Fixed avatar display in notification e-mails.
— Add support for web monetization.
— Correction de la suppression des chaînes source obsolètes lors de la suppression des traductions.

All changes in detail.
4.12 Weblate 4.12

Released on April 20th 2022.

- Added support for Amharic in *Incohérence de point final*.
- Added support for Burmese in *Incohérence de point d’interrogation*.
- Extended options of the *Génération d’une pseudo-traduction* add-on.
- Added *ignore-all-checks* flag to ignore all quality checks on a string.
- Avoid *Génération d’une pseudo-traduction* add-on to trigger failing checks.
- Added support for *Gitea pull requests*.
- Added Linux style language code to *Style de code-langue*.
- Added support for rebuilding project translation memory.
- Improved API for creating components from a file.
- Add copy and clone buttons to other translations.
- Make merge request message configurable at component level.
- Improved maximal length restriction behavior with XML tags.
- Fixed loading Fluent files with additional comments.

All changes in detail.

4.13 Weblate 4.11.2

Released on March 4th 2022.

- Fixed corrupted MO files in the binary release.

All changes in detail.

4.14 Weblate 4.11.1

Released on March 4th 2022.

- Fixed missing sanitizing of arguments to Git and Mercurial - CVE-2022-23915, see GHSA-3872-f48p-pxqj for more details.
- Fixed loading fuzzy strings from CSV files.
- Added support for creating teams using the API.
- Fixed user mention suggestions display.
- The project tokens access can now be customized.

All changes in detail.

4.15 Weblate 4.11

Released on February 25th 2022.

- Fixes stored XSS - CVE-2022-24710, see GHSA-6jp6-9rf9-gc66 for more details.
- Fixed add-on installation using API.
- Renamed *Strings needing action* to *Unfinished strings*.
- Fixed false positives from *Syntaxe ICU MessageFormat*.
- 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.
- Improved error handling on string removal.
- Weblate now requires Python 3.7 or newer.
- Fixed some write operations with project token authentication.
- Correction du suivi de l’état des chaînes lorsque celles-ci changent dans le référentiel.
- Suivre les modifications des chaînes de caractères du dépôt.
- Sticky header on translations listing to improve navigation.
Correction des chaînes non traduites dans Java properties.
Fixed Git operation with non-ascii branch names.
Nouveau module Pré-remplir la traduction avec la source.
Ajouté Fusion sans avance rapide Style de fusion.
Correction : déclenchement du module complémentaire Traduction automatique sur les chaînes nouvellement ajoutées.
Improved punctuation checks for Burmese.
Added support for defining custom teams at project level to grant users access, see Gestion du contrôle d’accès par projet.
Added documentation links to alerts.
Docker container automatically enables TLS/SSL for outgoing e-mail when needed.
Added support for searching for resolved comments.
Added support for borgbackup 1.2.
Fixed applying of Automatically translated label.

All changes in detail.

4.16 Weblate 4.10.1

Version publiée le 22 décembre 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.
— Improved analysis on the failed push alert.
— Fixed manually editing page when browsing changes.
— Amélioration de la précision de Présence d’un caractère kashida.
— The Weblate Docker container now uses Python 3.10.

All changes in detail.

4.17 Weblate 4.10

Version publiée le 16 décembre 2021.
— Added support for formality and placeholders with DeepL.
— Bulk edit and search and replace are now available on project and language level.
— Added filtering to search and replace.
— Fixed : « Perform automatic translation » privilege is no longer part of the Languages group.
— « Perform automatic translation » is in the Administration and the new Automatic translation group.
— Fixed generating XLSX files with special chars.
— Added ability to the GitHub authentication backend to check if the user belongs to a specific GitHub organization or team.
— Improved feedback on invalid parameters passed to API.
— Added support for project scoped access tokens for API.
— Fixed string removal in some cases.
— Correction de la traduction des chaînes nouvellement ajoutées.
— Label automatically translated strings to ease their filtering.

Toutes les modifications en détail.
4.18 Weblate 4.9.1

Version publiée le 19 novembre 2021.
— Fixed upload of monolingual files after changing template.
— Improved handling of whitespace in flags.
— Add support for filtering in download API.
— Correction de l'affichage des statistiques lors de l'ajout de nouvelles traductions.
— Mitigate issues with GitHub SSH key change.
Tous les changements en détail.

4.19 Weblate 4.9

— Provide more details for events in history.
— Improved rendering of history.
— Improved performance of the translation pages.
— Added support for restricting translation file downloads.
— The safe-html can now understand Markdown when used with md-text.
— The max-length tag now ignores XML markup when used with xml-text.
— Fixed dimensions of rendered texts in Taille maximale de la traduction.
— Lowered app store title length to 30 to assist with upcoming Google policy changes.
— Added support for customizing SSH invocation via SSH_EXTRA_ARGS.
— Added support for customizing Borg invocation via BORG_EXTRA_ARGS.
— Improved gettext PO merging in case of conflicts.
— Improved glossary highlighting.
— Improved safe-html flag behavior with XML checks.
— Fixed commit messages for linked components.
Tous les changements en détail.

4.20 Weblate 4.8.1

Version publiée le 10 septembre 2021.
— Fixed user removal in Django admin interface.
— Document add-on parameters in greater detail.
— Fixed JavaScript error in glossary.
— Add limit to number of matches in consistency check.
— Amélioration de la gestion des caractères de remplacement dans les traductions automatiques.
— Fixed creating add-ons using API.
— Added PRIVACY_URL setting to add privacy policy link to the footer.
— Hide member e-mail addresses from project admins.
— Improved gettext PO merging in case of conflicts.
— Improved glossary highlighting.
— Improved safe-html flag behavior with XML checks.
— Fixed commit messages for linked components.
Tous les changements en détail.
4.21 Weblate 4.8

— Added support for Apple stringsdict format.
— The exact search operator is now case-sensitive with PostgreSQL.
— Fixed saving glossary explanations in some cases.
— Améliorations de la documentation.
— Amélioration des performances.
— Improved squash add-on compatibility with Gerrit.
— Fixed adding strings to monolingual glossary components.
— Improved performance in handling variants.
— Fixed squash add-on sometimes skipping parsing upstream changes.
— Preserve file extension for downloads.
— Added support for the Fluent format.
— Added support for using tabs to indent JSON formats.
Tous les changements en détail.

4.22 Weblate 4.7.2

Version publiée le 15 juillet 2021.
— Support more language aliases to be configured on a project.
— Correction de la validation des chaînes de recherche dans l’API.
— Fixed Git exporter URLs after a domain change.
— Fixed cleanup add-on for Windows RC files.
— Fixed possible crash in XLIFF updating.
Tous les changements en détail.

4.23 Weblate 4.7.1

— Improved popup for adding terms to glossary.
— Ajout du support du service de traduction LibreTranslate.
— Added rate limiting on creating new projects.
— Improved performance of file updates.
Tous les changements en détail.

4.24 Weblate 4.7

Version publiée le 17 juin 2021.
— Improved configuration health check.
— Added support for object-pascal-format used in gettext PO, see Formatting Pascal Get.
— Renamed Nearby keys to Similar keys to better describe the purpose.
— Added support for mi18n lang files.
— Improved SAML authentication integration.
— Fixed Gerrit integration to better handle corner cases.
— Weblate now requires Django 3.2.
— Fixed inviting users when e-mail authentication is disabled.
— Amélioration des définitions des langues.
— Added support for blocking users from contributing to a project.
— Fixed automatic creation of glossary languages.
— Extended documentation about add-ons.
— Performance improvements for components with linked repositories.
— Added support for free DeepL API.
— The user management no longer needs Django admin interface.
Tous les changements en détail.

### 4.25 Weblate 4.6.2

Version publiée le 8 mai 2021.
— Fixed crash after moving shared component between projects.
— Fixed adding new strings to empty properties files.
— Fixed copy icon alignment in RTL languages.
— Extended string statistics on the Info tab.
— Fixed handling of translation files ignored in Git.
— Improved metrics performance.
— Fixed possible bug in saving glossaries.
— Fixed consistency check behavior on languages with different plural rules.
Tous les changements en détail.

### 4.26 Weblate 4.6.1

— Remove obsolete spam protection code.
— Improve source plural check accuracy.
— Update list of user interface languages in Docker.
— Improved error messages when creating pull requests.
— Fixed creating pull requests on Pagure.
— Fixed triggering automatically installed add-ons.
— Fixed possible caching issues on upgrade.
— Fixed adding new units to monolingual translations using upload.
Tous les changements en détail.

### 4.27 Weblate 4.6

— The auto_translate management command has now a parameter for specifying translation mode.
— Added support for Fichiers texte.
— Added trends and metrics for all objects.
— Added support for directly copying text from secondary languages.
— Added date filtering when browsing changes.
— Amélioration des graphiques d’activité.
— Sender for contact form e-mails can now be configured.
— Improved parameters validation in component creation API.
— The rate limiting no longer applies to superusers.
— Improved automatic translation add-on performance and reliability.
— The rate limiting now can be customized in the Docker container.
— API for creating components now automatically uses URLs internes de Weblate.
— Simplified state indication while listing strings.
— Password hashing now uses Argon2 by default.
— Barres de progression simplifiées indiquant l’état de la traduction.
— Renamed Ajouter les langues manquantes to clarify the purpose.
— Correction de la sauvegarde de l’état des chaînes de caractères en XLIFF.
— Ajout d’une recherche dans toute langue.
— Initial support for Scaling horizontally the Docker deployment.
Tous les changements en détail.
4.28 Weblate 4.5.3

Version publiée le 1 avril 2021.
— Fixed metrics collection.
— Fixed possible crash when adding strings.
— Improved search query examples.
— Fixed possible loss of newly added strings on replace upload.

4.29 Weblate 4.5.2

Version publiée le 26 mars 2021.
— Date configurable pour la traduction automatique.
— Added Lua format check.
— Ignore format strings in the Répétition de mots check.
— Allow uploading screenshot from a translate page.
— Added forced file synchronization to the repository maintenance.
— Fixed automatic suggestions for languages with a longer code.
— Improved performance when adding new strings.
— Several bug fixes in quality checks.
— Several performance improvements.
— Ajout de l'intégration avec Découverte Weblate.
— Fixed checks behavior with read-only strings.

Tous les changements en détail.

4.30 Weblate 4.5.1

Version publiée le 5 mars 2021.
— Fixed editing of glossary flags in some corner cases.
— Extend metrics usage to improve performance of several pages.
— Store correct source language in TMX files.
— Better handling for uploads of monolingual PO using API.
— Improved alerts behavior on glossary components.
— Improved Markdown link checks.
— Indicate glossary and source language in breadcrumbs.
— Pagination des composants pour les grands projets.
— Improved performance of translation, component or project removal.
— Improved bulk edit performance.
— Fixed preserving « Needs editing » and « Approved » states for ODF files.
— Amélioration de l’interface pour la personnalisation des téléchargements de fichiers de traduction

Tous les changements en détail.

4.31 Weblate 4.5

Version publiée le 19 février 2021.
— Prise en charge de “lua-format” utilisé dans les fichiers PO gettext.
— Prise en charge du partage de composants entre projets.
— Correction du comportement du contrôle de variables multiples non nommées avec des drapeaux de format multiples.
— Dropped mailing list field on the project in favor of generic instructions for translators.
— Added pseudolocale generation add-on.
— Added support for TermBase eXchange files.
— Prise en charge de la définition manuelle de variantes de chaîne à l’aide d’un indicateur.
— Amélioration des performances des contrôles de cohérence.
— Improved performance of translation memory for long strings.
— Prise en charge de la recherche dans les explications.
— Strings can now be added and removed in bilingual formats as well.
— Extend list of supported languages in Amazon Translate machine translation.
— Automatically enable Java MessageFormat checks for Java Properties.
— Added a new upload method to add new strings to a translation.
— Added a simple interface to browse translation.
— Glossaries are now stored as regular components.
— Dropped specific API for glossaries as component API is used now.
— Added simplified interface to toggle some of the flags.
— Added support for non-translatable or forbidden terms in the glossary.
— Added support for defining terminology in a glossary.
— Moved text direction toggle to get more space for the visual keyboard.
— Ajout d'une option permettant de surveiller automatiquement les projets auxquels l'utilisateur contribue.
— Added check whether translation matches the glossary.
— Added support for customizing navigation text color.

Tous les changements en détail.

### 4.32 Weblate 4.4.2

Version publiée le 14 janvier 2021.

— Correction des corruptions lors de l'utilisation d'un fichier MO distribué unique.

### 4.33 Weblate 4.4.1

Version publiée le 13 janvier 2021.

— Correction de l'annulation des changements sur les pluriels.
— Correction de l'affichage de l'aide des paramètres des projets.
— Amélioration de l'administration des utilisateurs.
— Amélioration du traitement du contexte dans les fichiers PO unilingues.
— Fixed cleanup add-on behavior with HTML, ODF, IDML and Windows RC formats.
— Correction de l'analyse des emplacements dans les fichiers CSV.
— Utilisation de la compression du contenu pour le téléchargement des fichiers.
— Amélioration de l'expérience utilisation lors de l'importation de fichiers ZIP.
— Amélioration de détection de type de fichier pour les téléchargements.
— Correction pour éviter la duplication de demande de tirage sur Pagure.
— Amélioration des performances lors de l'affichage des traductions fantômes.
— Réimplémentation de l'éditeur de traduction pour utiliser les zones de texte natives du navigateur.
— Fixed cleanup add-on breaking adding new strings.
— Added API for add-ons.

Tous les changements en détail.

### 4.34 Weblate 4.4


— Amélioration de la validation lors de la création d’un composant.
— Weblate nécessite désormais Django 3.1.
— Prise en charge de la personnalisation de l’apparence de l’interface de gestion.
— Correction de la gestion de l’état de lecture seule lors de la modification en masse.
— Amélioration de l’intégration avec CodeMirror.
— Added add-on to remove blank strings from translation files.
— L’éditeur CodeMirror est maintenant utilisé pour les traductions.
— Mise en évidence de la syntaxe dans l'éditeur de traduction pour XML, HTML, Markdown et reStructured-Text.
— Mettre en surbrillance les placeables dans l'éditeur de traduction.
— Amélioration de la prise en charge des codes linguistiques non standard.
— Ajout d'une alerte lors de l'utilisation de codes de langue ambigu.
— L'utilisateur se voit maintenant présenter une liste filtrée de langues lors de l'ajout d'une nouvelle traduction.
— Capacités de recherche étendues pour les changements dans l'historique.
— Improved billing detail pages and Libre hosting workflow.
— API des statistiques de traduction étendue.
— Amélioration de l'onglet « Autres langues » lors de la traduction.
— Ajout d'une API pour les tâches.
— Amélioration des performances du téléversement de fichier.
— Amélioration de l'affichage des caractères spéciaux définis par l'utilisateur.
— Amélioration des performances de la traduction automatique.
— Plusieurs améliorations mineures de l'interface utilisateur.
— Amélioration du nom des fichiers ZIP téléchargés.
— Ajout d'une option pour recevoir des notifications sur les projets non surveillés.

Tous les changements en détail.

### 4.35  Weblate 4.3.2


— Fixed crash on certain component file masks.
— Amélioration de la précision de la vérification des mots consécutifs dupliqués.
— Ajout de la prise en charge des demandes de fusion Pagure.
— Amélioration des messages d'erreur pour les inscriptions échouées.
— Annulation du changement qui affichait les commentaires du développeur en Markdown.
— Simplification de la configuration des dépôts Git avec une branche par défaut différente de « master ».
— Les dépôts internes nouvellement créés utilisent maintenant « main » comme branche par défaut.
— Réduction du taux de faux positifs pour les traductions inchangées lors de la traduction de reStructuredText.
— Correction de problèmes d'affichage de CodeMirror dans certaines situations.
— Renommage du groupe Modèle en « Sources » pour clarifier sa signification.
— Correction des demandes de tirage GitLab sur les dépôts ayant des chemins d'accès longs.

Tous les changements en détail.

### 4.36  Weblate 4.3.1


— Amélioration des performances de la traduction automatique.
— Correction de l'expiration de la session pour les utilisateurs authentifiés.
— Prise en charge du masquage des informations de version.
— Amélioration de la compatibilité des crochets avec Bitbucket Server.
— Amélioration des performances pour la mise à jour des mémoires de traduction.
— Réduction de l'utilisation mémoire.
— Amélioration des performances de la vue matricielle.
— Ajout d'une confirmation avant de retirer un utilisateur d'un projet.

Tous les changements en détail.
4.37 Weblate 4.3

— Inclusion des statistiques utilisateur dans l’API.
— Correction de l’ordre des composants sur les pages paginées.
— Définition des langues sources pour un glossaire.
— Réécriture de la prise en charge des demandes de tirage de GitHub et GitLab.
— Correction des statistiques après la suppression de suggestions.
— Extension des profils d’utilisateur publics.
— Correction de la configuration des contrôles de qualité forçés.
— Amélioration de la documentation à propos des sauvegardes intégrées.
— Déplacement de l’attribut de langue source d’un projet vers un composant.
— Ajout du contrôle du formatage Vue 118n.
— Prise en charge des expressions rationnelles pour les contrôles de qualité des balises de remplacement.
— Amélioration de l’aspect du mode matriciel.
— Traduction automatique est maintenant appelé Suggestions automatiques.
— Ajout de la prise en charge de l’interaction avec plusieurs instances GitLab ou GitHub.
— Extension de l’API pour couvrir les mises à jour de projets et d’unités, les suppressions, ainsi que le glossaire.
— L’API d’unité prend désormais en charge les chaînes plurielles.
— Les composants peuvent maintenant être créés en téléversant des fichiers ZIP ou des documents.
— Consolidation des codes de réponse de l’API.
— Prise en charge du Markdown dans l’accord de contribution.
— Amélioration du suivi des chaînes sources.
— Amélioration de la compatibilité avec les formats JSON, YAML et CSV.
— Ajout de la prise en charge des suppressions de chaînes.
— Amélioration des performances des téléchargements de fichiers.
— Amélioration de la vue de gestion des dépôts.
— Activation automatique du format java pour Android.
— Ajout de la prise en charge des captures d’écran traduites.
— Ajout de la prise en charge de Python 3.9.
— Correction de la traduction des fichiers HTML sous certaines conditions.

Tous les changements en détail.

4.38 Weblate 4.2.2

— Correction de la correspondance des chaînes source pour les formats JSON.
— Correction de la redirection de la connexion pour certaines configurations d’authentification.
— Correction de l’autentification LDAP avec synchronisation de groupe.
— Correction d’un dysfonctionnement dans la notification d’avancement des traductions automatiques.
— Correction des écrasements des archivages Git lorsque les « trailers » sont activés.
— Correction de la création de composants de système de contrôle des versions local à l’aide de l’API.

4.39 Weblate 4.2.1

— Correction de la sauvegarde des pluriels pour certaines langues dans les ressources Android.
— Fixed crash in the cleanup add-on for some XLIFF files.
— Ajout de la possibilité de configurer le CDN de localisation dans l’image Docker.
4.40 Weblate 4.2

— Amélioration des pages utilisateurs et ajout d’une liste d’utilisateurs.
— Suppression de la prise en charge des migrations depuis les versions 3.x, migrez en passant par les versions 4.1 ou 4.0.
— Ajout de l’exportation dans plusieurs formats monolingues.
— Amélioration des graphiques d’activité.
— Le nombre de chaînes à proximité affiché peut être configuré.
— Prise en charge du verrouillage des composants qui rencontrent des erreurs de dépôt.
— Simplification de la navigation principale (remplacement des boutons par des icônes).
— Amélioration de la gestion des codes de langues dans l’intégration de Google Translate.
— The Git squash add-on can generate Co-authored-by: trailers.
— Amélioration de l’analyseur de requêtes de recherche.
— Amélioration du retour d’information utilisateur pour les contrôles de format des chaînes.
— Amélioration des performances des modifications en masse.
— Ajout de redirections de compatibilité après renommage d’un projet ou d’un composant.
— Ajout de notifications pour les approbations de chaînes, les verrouillages de composants et les changements de licence.
— Ajout de la prise en charge de ModernMT.
— Ajout d’une fonction permettant d’éviter l’écrasement des traductions approuvées lors du téléversement de fichiers.
— Suppression de la prise en charge de certaines redirections d’URL de compatibilité.
— Ajout d’une vérification pour les ECMA Script template literals.
— Ajout d’une option pour surveiller un composant.
— Suppression du point final dans les clés d’unité JSON.
— Suppression de la file d’attente séparée de Celery pour le mémoire de traduction.
— Permet de traduire tous les composants d’une langue à la fois.
— Ajout d’une option permettant la configuration des en-têtes HTTP Content-Security-Policy.
— Ajout de la prise en charge de l’aliasing des langues au niveau du projet.
— New add-on to help with HTML or JavaScript localization, see JavaScript localisation CDN.
— Le domaine Weblate est maintenant configuré dans les paramètres, voir SITE_DOMAIN.
— Prise en charge de la recherche par composant et par projet.

4.41 Weblate 4.1.1

— Fixed changing autofix or add-ons configuration in Docker.
— Fixed possible crash in « About » page.
— Improved installation of byte-compiled locale files.
— Fixed adding words to glossary.
— Fixed keyboard shortcuts for machinery.
— Removed debugging output causing discarding log events in some setups.
— Fixed lock indication on project listing.
— Fixed listing GPG keys in some setups.
— Added option for which DeepL API version to use.
— Added support for acting as SAML Service Provider, see S’authentifier avec SAML.
4.42 Weblate 4.1


- Added support for creating new translations with included country code.
- Added support for searching source strings with screenshot.
- Extended info available in the stats insights.
- Improved search editing on « Translate » pages.
- Improve handling of concurrent repository updates.
- Include source language in project creation form.
- Include changes count in credits.
- Fixed UI language selection in some cases.
- Allow to whitelist registration methods with registrations closed.
- Improved lookup of related terms in glossary.
- Improved translation memory matches.
- Group same machinery results.
- Add direct link to edit screenshot from translate page.
- Improved removal confirmation dialog.
- Include templates in ZIP download.
- Add support for Markdown and notification configuration in announcements.
- Extended details in check listings.
- Consistently use dismissed as state of dismissed checks.
- Add support for configuring default add-ons to enable.
- Fixed editor keyboard shortcut to dismiss checks.
- Improved machine translation of strings with placeholders.
- Show ghost translation for user languages to ease starting them.
- Improved language code parsing.
- Show translations in user language first in the list.
- Renamed shapings to more generic name variants.
- Added new quality checks : Multiples variables non nommées, Ancienne chaîne non traduite, Répétition de mots.
- Reintroduced support for wiping translation memory.
- Fixed option to ignore source checks.
- Added support for configuring different branch for pushing changes.
- API now reports rate limiting status in the HTTP headers.
- Added support for Google Translate V3 API (Advanced).
- Added ability to restrict access on component level.
- Added support for whitespace and other special chars in translation flags, see Customizing behavior using flags.
- Always show rendered text check if enabled.
- API now supports filtering of changes.
- Added support for sharing glossaries between projects.

4.43 Weblate 4.0.4

Version publiée le 7 mai 2020.

- Fixed testsuite execution on some Python 3.8 environments.
- Typo fixes in the documentation.
- Fixed creating components using API in some cases.
- Fixed JavaScript errors breaking mobile navigation.
- Fixed crash on displaying some checks.
- Fixed screenshots listing.
- Fixed monthly digest notifications.
- Fixed intermediate translation behavior with units non existing in translation.
4.44 Weblate 4.0.3

Version publiée le 2 mai 2020.
— Fixed possible crash in reports.
— User mentions in comments are now case insensitive.
— Fixed PostgreSQL migration for non superusers.
— Fixed changing the repository URL while creating component.
— Fixed crash when upstream repository is gone.

4.45 Weblate 4.0.2

— Improved performance of translation stats.
— Improved performance of changing labels.
— Improved bulk edit performance.
— Improved translation memory performance.
— Fixed possible crash on component deletion.
— Fixed displaying of translation changes in some corner cases.
— Improved warning about too long celery queue.
— Fixed possible false positives in the consistency check.
— Fixed deadlock when changing linked component repository.
— Included edit distance in changes listing and CSV and reports.
— Avoid false positives of punctuation spacing check for Canadian French.
— Fixed XLIFF export with placeholders.
— Fixed false positive with zero width check.
— Improved reporting of configuration errors.
— Fixed bilingual source upload.
— Automatically detect supported languages for DeepL machine translation.
— Fixed progress bar display in some corner cases.
— Fixed some checks triggering on non translated strings.

4.46 Weblate 4.0.1

— Fixed package installation from PyPI.

4.47 Weblate 4.0

— 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.
— Afficher les suggestions échouant aux contrôles.
— Record source string changes in history.
— Upgraded Microsoft Translator to version 3 API.
— Reimplemented translation memory backend.
— Added support for several is: lookups in Recherche.
— Allow to make Traduction inchangée avoid internal blacklist.
— Improved comments extraction from monolingual po files.
— Renamed whiteboard messages to announcements.
— Fixed occasional problems with registration mails.
— Improved LINGUAS update add-on to handle more syntax variants.
— Fixed editing monolingual XLIFF source file.
— Added support for exact matching in Recherche.
— Extended API to cover screenshots, users, groups, component lists and extended creating projects.
— Added support for source upload on bilingual translations.
— Added support for intermediate language from developers.
— Added support for source strings review.
— Extended download options for platform wide translation memory.

4.48 Weblate 3.x

4.48.1 Weblate 3.11.3

— Fixed searching for fields with certain priority.
— Fixed predefined query for recently added strings.
— Fixed searching returning duplicate matches.
— Fixed notifications rendering in Gmail.
— Fixed reverting changes from the history.
— Added links to events in digest notifications.
— Fixed email for account removal confirmation.
— Added support for Slack authentication in Docker container.
— Avoid sending notifications for not subscribed languages.
— Include Celery queues in performance overview.
— Fixed documentation links for add-ons.
— Reduced false negatives for unchanged translation check.
— Raised bleach dependency to address CVE-2020-6802.
— Fixed listing project level changes in history.
— Fixed stats invalidation in some corner cases.
— Fixed searching for certain string states.
— Improved format string checks behavior on missing percent.
— Fixed authentication using some third party providers.

4.48.2 Weblate 3.11.2

— Fixed rendering of suggestions.
— Fixed some strings wrongly reported as having no words.

4.48.3 Weblate 3.11.1

— Documented Celery setup changes.
— Improved filename validation on component creation.
— Fixed minimal versions of some dependencies.
— Fixed adding groups with certain Django versions.
— Fixed manual pushing to upstream repository.
— Improved glossary matching.
4.48.4 Weblate 3.11

Version publiée le 17 février 2020.
— Allow using VCS push URL during component creation via API.
— Rendered width check now shows image with the render.
— Fixed links in notifications e-mails.
— Improved look of plaintext e-mails.
— Display ignored checks and allow to make them active again.
— Display nearby keys on monolingual translations.
— Prise en charge du regroupement des formes de chaînes.
— 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.
— Ajout de la modification en masse des drapeaux.
— Added support for labels.
— Added bulk edit add-on.
— Added option for Exécution des contrôles.
— Increased default validity of confirmation links.
— Improved Matomo integration.
— Fixed A déjà été traduit to correctly handle source string change.
— Extended automatic updates configuration by AUTO_UPDATE.
— LINGUAS add-ons now do full sync of translations in Weblate.

4.48.5 Weblate 3.10.3

Version publiée le 18 janvier 2020.
— Support for translate-toolkit 2.5.0.

4.48.6 Weblate 3.10.2

Version publiée le 18 janvier 2020.
— Add lock indication to projects.
— Fixed CSS bug causing flickering in some web browsers.
— Fixed searching on systems with non-English locales.
— Improved repository matching for GitHub and Bitbucket hooks.
— Fixed data migration on some Python 2.7 installations.
— Allow configuration of Git shallow cloning.
— Improved background notification processing.
— Fixed broken form submission when navigating back in web browser.
— New add-on to configure YAML formatting.
— Fixed same plurals check to not fire on single plural form languages.
— Fixed regex search on some fields.
4.48.7 Weblate 3.10.1

— Extended API with translation creation.
— Fixed several corner cases in data migrations.
— Compatibility with Django 3.0.
— Amélioration des performances lors du nettoyage des données.
— Added support for customizable security.txt.
— Improved breadcrumbs in changelog.
— Improved translations listing on dashboard.
— Improved HTTP responses for webhooks.
— Added support for GitLab merge requests in Docker container.

4.48.8 Weblate 3.10

— Improved application user interface.
— Added doublespace check.
— Fixed creating new languages.
— Avoid sending auditlog notifications to deleted e-mails.
— Added support for read-only strings.
— Added support for Markdown in comments.
— Allow placing translation instruction text in project info.
— Add copy to clipboard for secondary languages.
— Improved support for Mercurial.
— Improved Git repository fetching performance.
— Add search lookup for age of string.
— Show source language for all translations.
— Show context for nearby strings.
— Added support for notifications on repository operations.
— Improved translation listings.
— Extended search capabilities.
— Added support for automatic translation strings marked for editing.
— Avoid sending duplicate notifications for linked component alerts.
— Improve default merge request message.
— Better indicate string state in Zen mode.
— Added support for more languages in Yandex Translate.
— Improved look of notification e-mails.
— Provide choice for translation license.

4.48.9 Weblate 3.9.1

— Remove some unneeded files from backups.
— Fixed potential crash in reports.
— Fixed cross database migration failure.
— Added support for force pushing Git repositories.
— Reduced risk of registration token invalidation.
— Fixed account removal hitting rate limiter.
— Added search based on priority.
— Fixed possible crash on adding strings to JSON file.
— Safe HTML check and fixup now honor source string markup.
— Avoid sending notifications to invited and deleted users.
— Fix SSL connection to redis in Celery in Docker container.
4.48.10 Weblate 3.9

Version publiée le 15 octobre 2019.

- Include Weblate metadata in downloaded files.
- Improved UI for failing checks.
- Indicate missing strings in format checks.
- Separate check for French punctuation spacing.
- Add support for fixing some of quality checks errors.
- Add separate permission to create new projects.
- Extend stats for char counts.
- Improve support for Java style language codes.
- Added new generic check for placeholders.
- Added support for WebExtension JSON place holders.
- Added support for flat XML format.
- Extended API with project, component and translation removal and creation.
- Added support for Gitea and Gitee webhooks.
- Added new custom regex based check.
- Allow to configure contributing to shared translation memory.
- Added ZIP download for more translation files.
- Make XLIFF standard compliant parsing of maxwidth and font.
- Added new check and fixer for safe HTML markup for translating web applications.
- Add component alert on unsupported configuration.
- Added automatic translation add-on to bootstrap translations.
- Extend automatic translation to add suggestions.
- Display add-on parameters on overview.
- Sentry is now supported through modern Sentry SDK instead of Raven.
- Changed example settings to be better fit for production environment.
- Added automated backups using BorgBackup.
- Split cleanup add-on for RESX to avoid unwanted file updates.
- Added advanced search capabilities.
- Allow users to download their own reports.
- Added localization guide to help configuring components.
- Added support for GitLab merge requests.
- Improved display of repository status.
- Perform automated translation in the background.

4.48.11 Weblate 3.8


- Added support for simplified creating of similar components.
- Added support for parsing translation flags from the XML based file formats.
- Log exceptions into Celery log.
- Improve performance of repository scoped add-ons.
- Improved look of notification e-mails.
- Fixed password reset behavior.
- Improved performance on most of translation pages.
- Fixed listing of languages not known to Weblate.
- Add support for cloning add-ons to discovered components.
- Add support for replacing file content with uploaded.
- Add support for translating non VCS based content.
- Added OpenGraph widget image to use on social networks.
- Added support for animated screenshots.
- Improved handling of monolingual XLIFF files.
- Avoid sending multiple notifications for single event.
- Add support for filtering changes.
- Extended predefined periods for reporting.
- Added webhook support for Azure Repos.
— New opt-in notifications on pending suggestions or untranslated strings.
— Add one click unsubscribe link to notification e-mails.
— Fixed false positives with Has been translated check.
— New management interface for admins.
— String priority can now be specified using flags.
— Added language management views.
— Add checks for Qt library and Ruby format strings.
— Added configuration to better fit single project installations.
— Notify about new string on source string change on monolingual translations.
— Added separate view for translation memory with search capability.

4.48.12 Weblate 3.7.1

— Modifications de la documentation.
— Fixed some requirements constraints.
— Updated language database.
— Mises à jour de localisation.
— Various user interface tweaks.
— Improved handling of unsupported but discovered translation files.
— More verbosely report missing file format requirements.

4.48.13 Weblate 3.7

— Added separate Celery queue for notifications.
— Use consistent look with application for API browsing.
— Include approved stats in the reports.
— Report progress when updating translation component.
— Allow to abort running background component update.
— Extend template language for filename manipulations.
— Use templates for editor link and repository browser URL.
— Indicate max length and current characters count when editing translation.
— Improved handling of abbreviations in unchanged translation check.
— Refreshed landing page for new contributors.
— Add support for configuring msgmerge add-on.
— Delay opening SMTP connection when sending notifications.
— Improved error logging.
— Allow custom location in MO generating add-on.
— Added add-ons to cleanup old suggestions or comments.
— Added option to enable horizontal mode in the Zen editor.
— Improved import performance with many linked components.
— Fixed examples installation in some cases.
— Improved rendering of alerts in changes.
— Added new horizontal stats widget.
— Improved format strings check on plurals.
— Added font management tool.
— New check for rendered text dimensions.
— Added support for subtitle formats.
— Include overall completion stats for languages.
— Added reporting at project and global scope.
— Improved user interface when showing translation status.
— New Weblate logo and color scheme.
— New look of bitmap badges.
4.48.14 Weblate 3.6.1

— 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 projectson project listing.
— Allow update to recover from missing VCS repository.

4.48.15 Weblate 3.6

— Add support for downloading user data.
— Add-ons are now automatically triggered upon installation.
— Improved instructions for resolving merge conflicts.
— Cleanup add-on is now compatible with app store metadata translations.
— Configurable language code syntax when adding new translations.
— Warn about using Python 2 with planned termination of support in April 2020.
— Extract special characters from the source string for visual keyboard.
— Extended contributor stats to reflect both source and target counts.
— Admins and consistency add-ons can now add translations even if disabled for users.
— Fixed description of toggle disabling Language-Team header manipulation.
— Notify users mentioned in comments.
— Removed file format autodetection from component setup.
— Fixed generating MO file for monolingual PO files.
— Added digest notifications.
— Added support for muting component notifications.
— Added notifications for new alerts, whiteboard messages or components.
— Notifications for administered projects can now be configured.
— Improved handling of three letter language codes.

4.48.16 Weblate 3.5.1

— Fixed Celery systemd unit example.
— Fixed notifications from HTTP repositories with login.
— Fixed race condition in editing source string for monolingual translations.
— Include output of failed add-on execution in the logs.
— Improved validation of choices for adding new language.
— Allow to edit file format in component settings.
— Update installation instructions to prefer Python 3.
— Performance and consistency improvements for loading translations.
— Make Microsoft Terminology service compatible with current Zeep releases.
— Mises à jour de localisation.
4.48.17 **Weblate 3.5**


- Amélioration du performance de la mémoire de traduction intégrée.
- Ajout d’une interface pour gérer la mémoire de traduction globale.
- Amélioration de l’alerte sur le statut non valide d’un composant.
- Ajout d’une interface pour gérer les messages blanc.
- Ajout d’un onglet pour configurer les messages de commit.
- Réduction du nombre de commits lors de la mise à jour de la repository.
- Correction de la perte de données métadat lors du déplacement d’un composant entre des projets.
- Amélioration de la navigation dans le mode Zen.
- Ajout de plusieurs nouvelles vérifications de qualité (markdown related et URL).
- Ajout d’un support pour les fichiers de metadata des app store.
- Ajout d’un support pour basculer entre GitHub ou Gerrit整合.
- Ajout d’une vérification pour les lettres Kashida.
- Ajout d’une option pour raser les commits en fonction de l’auteur.
- Amélioration de la performance du chargement des fichiers de traduction.
- Nouveau onglet pour raser les commits avant le push.
- Amélioration de l’affichage des changements de traduction.
- Changement de l’affichage de la style de merge de par défaut à rebase et de devenant configurable.
- Gestion des sous-groupes privés dans le code de langue.
- Amélioration de la performance des mises à jour de l’index de textes complets.
- Extension de l’API de chargement de fichiers pour prendre en charge de nouveaux paramètres.

4.48.18 **Weblate 3.4**


- Ajout du support pour les placeholders de XLIFF.
- Celery peut maintenant utiliser plusieurs queues de tâches.
- Ajout d’un support pour le renommage et le déplacement des projets et composants.
- Incorporation des comptages de caractères dans les rapports.
- Ajout de guidance pour l’ajout de composants de traduction automatique avec détection automatique des fichiers de traduction.
- Ajout de messages de merge commit personnalisable pour Git.
- Ajout d’une indication visuelle des alertes de composant dans la navigation.
- Amélioration de la performance de chargement des fichiers de traduction.
- Nouveau onglet pour raser les commits prior à push.
- Amélioration de l’affichage des changements de traduction.
- Changement du style de merge défaut à rebase et devenant configurable.
- Gestion des sous-groupes privés dans le code de langue.
- Amélioration de la performance des mises à jour de l’index de texte.
- Extension de l’API de chargement de fichiers pour prendre en charge de nouveaux paramètres.

4.48.19 **Weblate 3.3**

Version publiée le 30 novembre 2018.

- Ajout du support pour le déplacement des composants et des projets.
- Amélioration de la performance pour certaines traductions monolithiques.
- Ajout de messages d’alerte pour les problèmes de traduction.
- Exposition de la resname de XLIFF de string comme contexte lorsqu’il est disponible.
- Ajout d’un support pour les états de XLIFF.
- Ajout de vérification pour les fichiers non écritables dans le DATA_DIR.
- Amélioration de l’export CSV pour les changements.
- Ajout de support pour les métadat dans DATA_DIR.
- Amélioration de l’export CSV pour les changements.
4.48.20  Weblate 3.2.2

Version publiée le 20 octobre 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.

4.48.21  Weblate 3.2.1

Version publiée le 10 octobre 2018.
- Document dependency on backports.csv on Python 2.7.
- Fix running tests under root.
- Improved error handling in gitexport module.
- Fixed progress reporting for newly added languages.
- Correctly report Celery worker errors to Sentry.
- Fixed creating new translations with Qt Linguist.
- Fixed occasional fulltext index update failures.
- Improved validation when creating new components.
- Added support for cleanup of old suggestions.

4.48.22  Weblate 3.2

Version publiée le 6 octobre 2018.
- Add install_addon management command for automated add-on installation.
- Allow more fine grained ratelimit settings.
- Added support for export and import of Excel files.
- Improve component cleanup in case of multiple component discovery add-ons.
- Rewritten Microsoft Terminology machine translation backend.
- Weblate now uses Celery to offload some processing.
- Improved search capabilities and added regular expression search.
- Added support for Youdao Zhiyun API machine translation.
- Added support for Baidu API machine translation.
- Integrated maintenance and cleanup tasks using Celery.
- Improved performance of loading translations by almost 25%.
- Removed support for merging headers on upload.
- Removed support for custom commit messages.
- Configurable editing mode (zen/full).
- Added support for error reporting to Sentry.
- Added support for automated daily update of repositories.
- Added support for creating projects and components by users.
- Built-in translation memory now automatically stores translations done.
- Users and projects can import their existing translation memories.
- Better management of related strings for screenshots.
- Added support for checking Java MessageFormat.

See 3.2 milestone on GitHub for detailed list of addressed issues.
4.48.23 Weblate 3.1.1

Version publiée le 27 juillet 2018.
— Fix testsuite failure on some setups.

4.48.24 Weblate 3.1

Version publiée le 27 juillet 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.
— Nouveau module complémentaire permettant de marquer les traductions inchangées comme devant être modifiées.
— Add support for jumping to specific location while translating.
— Downloaded translations can now be customized.
— Improved calculation of string similarity in translation memory matches.
— Added support by signing Git commits by GnuPG.

4.48.25 Weblate 3.0.1

Version publiée le 10 juin 2018.
— Fixed possible migration issue from 2.20.
— Mises à jour de localisation.
— Removed obsolete hook examples.
— Improved caching documentation.
— Fixed displaying of admin documentation.
— Improved handling of long language names.

4.48.26 Weblate 3.0

Version publiée le 1er juin 2018.
— Rewritten access control.
— Several code cleanups that lead to moved and renamed modules.
— New add-on for automatic component discovery.
— The import_project management command has now slightly different parameters.
— Added basic support for Windows RC files.
— New add-on to store contributor names in PO file headers.
— The per component hook scripts are removed, use add-ons instead.
— Add support for collecting contributor agreements.
— Access control changes are now tracked in history.
— New add-on to ensure all components in a project have same translations.
— Support for more variables in commit message templates.
— Add support for providing additional textual context.
4.49 Weblate 2.x

4.49.1 Weblate 2.20

Version publiée le 4 avril 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.
— Ajout du mémoire de traduction intégré.
— Added component lists overview to dashboard and per component list overview pages.
— Added support for DeepL machine translation service.
— Machine translation results are now cached inside Weblate.
— Prise en charge de la réorganisation des modifications archivées.

4.49.2 Weblate 2.19.1

Version publiée le 20 février 2018.
— Fixed migration issue on upgrade from 2.18.
— Improved file upload API validation.

4.49.3 Weblate 2.19

Version publiée le 15 février 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.
— Amélioration du service de traduction automatique.
— Added support for SAP Translation Hub service.
— Added support for Microsoft Terminology service.
— Removed support for advertisement in notification e-mails.
— Improved translation progress reporting at language level.
— Improved support for different plural formulas.
— Added support for Subversion repositories not using stdlayout.
— Added add-ons to customize translation workflows.
4.49.4 Weblate 2.18

— Extended contributor stats.
— Improved configuration of special characters virtual keyboard.
— Added support for DTD file format.
— Changed keyboard shortcuts to less likely collide with browser/system ones.
— Improved support for approved flag in XLIFF files.
— Added support for not wrapping long strings in gettext PO files.
— Added button to copy permalink for current translation.
— Dropped support for Django 1.10 and added support for Django 2.0.
— Removed locking of translations while translating.
— Added support for adding new strings to monolingual translations.
— Added support for translation workflows with dedicated reviewers.

4.49.5 Weblate 2.17.1

— Fixed running testsuite in some specific situations.
— Locales updates.

4.49.6 Weblate 2.17

— 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 noweditable.
— Added optional spam protection for suggestions using Akismet.

4.49.7 Weblate 2.16

— Various performance improvements.
— Added support for nested JSON format.
— Added support for WebExtension JSON format.
— Fixed git exporter authentication.
— Improved CSV import in certain situations.
— Improved look of Other translations widget.
— The max-length checks is now enforcing length of text in form.
— Make the commit_pending age configurable per component.
— Various user interface cleanups.
— Fixed component/project/site wide search for translations.
4.49.8 Weblate 2.15

— Show more related translations in other translations.
— Add option to see translations of current string to other languages.
— Use 4 plural forms for Lithuanian by default.
— Fixed upload for monolingual files of different format.
— Improved error messages on failed authentication.
— Keep page state when removing word from glossary.
— Added direct link to edit secondary language translation.
— Added Perl format quality check.
— Added support for rejecting reused passwords.
— Extended toolbar for editing RTL languages.

4.49.9 Weblate 2.14.1

— Fixed possible error when paginating search results.
— Fixed migrations from older versions in some corner cases.
— Fixed possible CSRF on project watch and unwatch.
— The password reset no longer authenticates user.
— Fixed possible CAPTCHA bypass on forgotten password.

4.49.10 Weblate 2.14

— Add glossary entries using AJAX.
— The logout now uses POST to avoid CSRF.
— The API key token reset now uses POST to avoid CSRF.
— Weblate sets Content-Security-Policy by default.
— The local editor URL is validated to avoid self-XSS.
— The password is now validated against common flaws by default.
— Notify users about important activity with their account such as password change.
— The CSV exports now escape potential formulas.
— Various minor improvements in security.
— The authentication attempts are now rate limited.
— Suggestion content is stored in the history.
— Store important account activity in audit log.
— Ask for password confirmation when removing account or adding new associations.
— Show time when suggestion has been made.
— There is new quality check for trailing semicolon.
— Ensure that search links can be shared.
— Included source string information and screenshots in the API.
— Allow to overwrite translations through API upload.
4.49.11 Weblate 2.13.1

   — Fixed listing of managed projects in profile.
   — Fixed migration issue where some permissions were missing.
   — Fixed listing of current file format in translation download.
   — Return HTTP 404 when trying to access project where user lacks privileges.

4.49.12 Weblate 2.13

   — Fixed quality checks on translation templates.
   — Added quality check to trigger on losing translation.
   — Add option to view pending suggestions from user.
   — Add option to automatically build component lists.
   — Default dashboard for unauthenticated users can be configured.
   — Add option to browse 25 random strings for review.
   — History now indicates string change.
   — Better error reporting when adding new translation.
   — Added per language search within project.
   — Group ACLs can now be limited to certain permissions.
   — The per project ACLs are now implemented using Group ACL.
   — Added more fine grained privileges control.
   — Various minor UI improvements.

4.49.13 Weblate 2.12

   — Improved admin interface for groups.
   — Added support for Yandex Translate API.
   — Improved speed of site wide search.
   — Added project and component wide search.
   — Added project and component wide search and replace.
   — Improved rendering of inconsistent translations.
   — Added support for opening source files in local editor.
   — Added support for configuring visual keyboard with special characters.
   — Improved screenshot management with OCR support for matching source strings.
   — Default commit message now includes translation information and URL.
   — Added support for Joomla translation format.
   — Improved reliability of import across file formats.

4.49.14 Weblate 2.11

   — Include language detailed information on language page.
   — Mercurial backend improvements.
   — Added option to specify translation component priority.
   — More consistent usage of Group ACL even with less used permissions.
   — Added WL_BRANCH variable to hook scripts.
   — Improved developer documentation.
   — Better compatibility with various Git versions in Git exporter add-on.
   — Included per project and component stats.
   — Added language code mapping for better support of Microsoft Translate API.
   — Moved fulltext cleanup to background job to make translation removal faster.
   — Fixed displaying of plural source for languages with single plural form.
— Improved error handling in import_project.
— Various performance improvements.

4.49.15 Weblate 2.10.1

Version publiée le 20 janvier 2017.
— Do not leak account existence on password reset form (CVE-2017-5537).

4.49.16 Weblate 2.10

Version publiée le 15 décembre 2016.
— Added quality check to check whether plurals are translated differently.
— Fixed GitHub hooks for repositories with authentication.
— Added optional Git exporter module.
— Support for Microsoft Cognitive Services Translator API.
— Simplified project and component user interface.
— Added automatic fix to remove control characters.
— Added per language overview to project.
— Added support for CSV export.
— Added CSV download for stats.
— Added matrix view for quick overview of all translations.
— Added basic API for changes and strings.
— Added support for Apertium APy server for machine translations.

4.49.17 Weblate 2.9

Version publiée le 4 novembre 2016.
— Extended parameters for createadmin management command.
— Extended import_json to be able to handle with existing components.
— Added support for YAML files.
— Project owners can now configure translation component and project details.
— Use « Watched » instead of « Subscribed » projects.
— Projects can be watched directly from project page.
— Added multi language status widget.
— Highlight secondary language if not showing source.
— Record suggestion deletion in history.
— Improved UX of languages selection in profile.
— Fixed showing whiteboard messages for component.
— Keep preferences tab selected after saving.
— Show source string comment more prominently.
— Automatically install Gettext PO merge driver for Git repositories.
— Added search and replace feature.
— Added support for uploading visual context (screenshots) for translations.
4.49.18 Weblate 2.8

Version publiée le 31 août 2016.
- Améliorations de la documentation.
- Mise à jour des traductions.
- Updated bundled JavaScript libraries.
- Added list_translators management command.
- Django 1.8 is no longer supported.
- Fixed compatibility with Django 1.10.
- Added Subversion support.
- Separated XML validity check from XML mismatched tags.
- Fixed API to honor HIDE_REPO_CREDENTIALS settings.
- Show source change in Zen mode.
- Alt+PageUp/PageDown/Home/End now works in Zen mode as well.
- Add tooltip showing exact time of changes.
- Add option to select filters and search from translation page.
- Added UI for translation removal.
- Improved behavior when inserting placeables.
- Fixed auto locking issues in Zen mode.

4.49.19 Weblate 2.7

Version publiée le 10 juillet 2016.
- Removed Google web translate machine translation.
- Improved commit message when adding translation.
- Fixed Google Translate API for Hebrew language.
- Compatibility with Mercurial 3.8.
- Added import_json management command.
- Correct ordering of listed translations.
- Show full suggestion text, not only a diff.
- Extend API (detailed repository status, statistics, …).
- Testsuite no longer requires network access to test repositories.

4.49.20 Weblate 2.6

Version publiée le 28 avril 2016.
- Fixed validation of components with language filter.
- Improved support for XLIFF files.
- Fixed machine translation for non English sources.
- Added REST API.
- Django 1.10 compatibility.
- Added categories to whiteboard messages.

4.49.21 Weblate 2.5

Version publiée le 10 mars 2016.
- Fixed automatic translation for project owners.
- Improved performance of commit and push operations.
- New management command to add suggestions from command-line.
- Added support for merging comments on file upload.
- Added support for some GNU extensions to C printf format.
- Améliorations de la documentation.
- 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.
— Correction pour les recherches sur l’ensemble du site des contrôles défaillants.
— Added option to specify source language.
— Improved support for XLIFF files.
— Extended list of options for import_project.
— Improved targeting for whiteboard messages.
— Support for automatic translation across projects.
— Optimized fulltext search index.
— Added management command for auto translation.
— Added placeables highlighting.
— Added keyboard shortcuts for placeables, checks and machine translations.
— Improved translation locking.
— Added quality check for AngularJS interpolation.
— Added extensive group based ACLs.
— Clarified terminology on strings needing edit (formerly fuzzy).
— Clarified terminology on strings needing action and untranslated strings.
— Support for Python 3.
— Dropped support for Django 1.7.
— Dropped dependency on msginit for creating new gettext PO files.
— Added configurable dashboard views.
— Improved notifications on parse errors.
— Added option to import components with duplicate name to import_project.
— Improved support for translating PHP files.
— Added XLIFF export for dictionary.
— Added XLIFF and gettext PO export for all translations.
— Améliorations de la documentation.
— Added support for configurable automatic group assignments.
— Improved adding of new translations.

### 4.49.22 Weblate 2.4

Version publiée le 20 septembre 2015.
— Improved support for PHP files.
— Ability to add ACL to anonymous user.
— Improved configurability of import_project command.
— Added CSV dump of history.
— Avoid copy/paste errors with whitespace characters.
— Added support for Bitbucket webhooks.
— Tighter control on fuzzy strings on translation upload.
— Several URLs have changed, you might have to update your bookmarks.
— Hook scripts are executed with VCS root as current directory.
— Hook scripts are executed with environment variables describing current component.
— Add management command to optimize fulltext index.
— Added support for error reporting to Rollbar.
— Projects now can have multiple owners.
— Project owners can manage themselves.
— Added support for javascript-format used in gettext PO.
— Support for adding new translations in XLIFF.
— Improved file format autodetection.
— Extended keyboard shortcuts.
— Improved dictionary matching for several languages.
— Improved layout of most of pages.
— Support for adding words to dictionary while translating.
— Added support for filtering languages to be managed by Weblate.
— Added support for translating and importing CSV files.
— Rewritten handling of static files.
— Direct login/registration links to third-party service if that's the only one.
— Commit pending changes on account removal.
— Add management command to change site name.
— Add option to configure default committer.
— Add hook after adding new translation.
— Add option to specify multiple files to add to commit.

4.49.23 Weblate 2.3

Version publiée le 22 mai 2015.
— Dropped support for Django 1.6 and South migrations.
— Support for adding new translations when using Java Property files.
— Allow to accept suggestion without editing.
— Improved support for Google OAuth 2.0.
— Added support for Microsoft .resx files.
— Tuned default robots.txt to disallow big crawling of translations.
— Simplified workflow for accepting suggestions.
— Added project owners who always receive important notifications.
— Allow to disable editing of monolingual template.
— More detailed repository status view.
— Direct link for editing template when changing translation.
— Allow to add more permissions to project owners.
— Allow to show secondary language in Zen mode.
— Support for hiding source string in favor of secondary language.

4.49.24 Weblate 2.2

Version publiée le 19 février 2015.
— Amélioration des performances.
— Fulltext search on location and comments fields.
— New SVG/JavaScript-based activity charts.
— Support for Django 1.8.
— Support for deleting comments.
— Added own SVG badge.
— Added support for Google Analytics.
— Improved handling of translation filenames.
— Added support for monolingual JSON translations.
— Record component locking in a history.
— Support for editing source (template) language for monolingual translations.
— Added basic support for Gerrit.

4.49.25 Weblate 2.1

Version publiée le 5 décembre 2014.
— Added support for Mercurial repositories.
— Replaced Glyphicon font by Awesome.
— Added icons for social authentication services.
— Better consistency of button colors and icons.
— Améliorations de la documentation.
— Diverses corrections de bugs.
— Automatic hiding of columns in translation listing for small screens.
— Changed configuration of filesystem paths.
— Improved SSH keys handling and storage.
— Improved repository locking.
— Customizable quality checks per source string.
— Allow to hide completed translations from dashboard.

4.49.26 Weblate 2.0

Version publiée le 6 novembre 2014.
— New responsive UI using Bootstrap.
— Rewritten VCS backend.
— Améliorations de la documentation.
— Added whiteboard for site wide messages.
— Configurable strings priority.
— Added support for JSON file format.
— Fixed generating mo files in certain cases.
— Added support for GitLab notifications.
— Added support for disabling translation suggestions.
— Django 1.7 support.
— ACL projects now have user management.
— Extended search possibilities.
— Give more hints to translators about plurals.
— Fixed Git repository locking.
— Compatibility with older Git versions.
— Improved ACL support.
— Added buttons for per language quotes and other special characters.
— Support for exporting stats as JSONP.

4.50 Weblate 1.x

4.50.1 Weblate 1.9

Version publiée le 6 mai 2014.
— Django 1.6 compatibility.
— No longer maintained compatibility with Django 1.4.
— Management commands for locking/unlocking translations.
— Improved support for QtTS files.
— Users can now delete their account.
— Avatars can be disabled.
— Merged first and last name attributes.
— Avatars are now fetched and cached server side.
— Added support for shields.io badge.

4.50.2 Weblate 1.8

Version publiée le 7 novembre 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.
— Améliorations de la documentation.
— Improved source strings review.
— Searching across all strings.
— Better tracking of source strings.
— Captcha protection for registration.

4.50.3 Weblate 1.7

Version publiée le 7 octobre 2013.
— Please check manual for upgrade instructions.
— Support for checking Python brace format string.
— Per component customization of quality checks.
— Detailed per translation stats.
— Changed way of linking suggestions, checks and comments to strings.
— Users can now add text to commit message.
— Support for subscribing on new language requests.
— Support for adding new translations.
— Widgets and charts are now rendered using Pillow instead of Pango + Cairo.
— Add status badge widget.
— Dropped invalid text direction check.
— Changes in dictionary are now logged in history.
— Performance improvements for translation view.

4.50.4 Weblate 1.6

— Nicer error handling on registration.
— Browsing of changes.
— Fixed sorting of machine translation suggestions.
— Improved support for MyMemory machine translation.
— Added support for Amagama machine translation.
— Various optimizations on frequently used pages.
— Highlights searched phrase in search results.
— Support for automatic fixups while saving the message.
— Tracking of translation history and option to revert it.
— Added support for Google Translate API.
— Added support for managing SSH host keys.
— Various form validation improvements.
— Various quality checks improvements.
— Performance improvements for import.
— Added support for voting on suggestions.
— Cleanup of admin interface.

4.50.5 Weblate 1.5

Version publiée le 16 avril 2013.
— Please check manual for upgrade instructions.
— Added public user pages.
— Better naming of plural forms.
— Added support for TBX export of glossary.
— Added support for Bitbucket notifications.
— Activity charts are now available for each translation, language or user.
— Extended options of import_project admin command.
— Compatible with Django 1.5.
— Avatars are now shown using libravatar.
— Added possibility to pretty print JSON export.
— Various performance improvements.
— Indicate failing checks or fuzzy strings in progress bars for projects or languages as well.
— Added support for custom pre-commit hooks and committing additional files.
— Rewritten search for better performance and user experience.
— New interface for machine translations.
— Added support for monolingual po files.
— Extend amount of cached metadata to improve speed of various searches.
— Now shows word counts as well.

4.50.6 Weblate 1.4

Version publiée le 23 janvier 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 tkit loader.
— Added sitemaps to allow easier access by crawlers.
— Provide direct links to string in notification e-mails or feeds.
— Various improvements to admin interface.
— Provide hints for production setup in admin interface.
— Added per language widgets and engage page.
— Improved translation locking handling.
— Show code snippets for widgets in more variants.
— Indicate failing checks or fuzzy strings in progress bars.
— More options for formatting commit message.
— Fixed error handling with machine translation services.
— Improved automatic translation locking behaviour.
— Support for showing changes from previous source string.
— Added support for substring search.
— Various quality checks improvements.
— Support for per project ACL.
— Basic code coverage by unit tests.

4.50.7 Weblate 1.3

Version publiée le 16 novembre 2012.
— Compatibility with PostgreSQL database backend.
— Removes languages removed in upstream git repository.
— Improved quality checks processing.
— Added new checks (BBCode, XML markup and newlines).
— Support for optional rebasing instead of merge.
— Possibility to relocate Weblate (for example to run it under /weblate path).
— Support for manually choosing file type in case autodetection fails.
— Better support for Android resources.
— Support for generating SSH key from web interface.
— More visible data exports.
— New buttons to enter some special characters.
— Support for exporting dictionary.
— Support for locking down whole Weblate installation.
— Checks for source strings and support for source strings review.
— Support for user comments for both translations and source strings.
— Better changes log tracking.
— Changes can now be monitored using RSS.
— Improved support for RTL languages.
4.50.8 Weblate 1.2

Version publiée le 14 août 2012.
— Weblate now uses South for database migration, please check upgrade instructions if you are upgrading.
— Fixed minor issues with linked git repos.
— New introduction page for engaging people with translating using Weblate.
— Added widgets which can be used for promoting translation projects.
— Added option to reset repository to origin (for privileged users).
— Project or component can now be locked for translations.
— Possibility to disable some translations.
— Configurable options for adding new translations.
— Configuration of git commits per project.
— Simple antispam protection.
— Better layout of main page.
— Support for automatically pushing changes on every commit.
— Support for e-mail notifications of translators.
— List only used languages in preferences.
— Improved handling of not known languages when importing project.
— Support for locking translation by translator.
— Optionally maintain Language-Team header in po file.
— Include some statistics in about page.
— Supports (and requires) django-registration 0.8.
— Mise en cache du nombre de chaines avec des contrôles défaillants.
— Checking of requirements during setup.
— Améliorations de la documentation.

4.50.9 Weblate 1.1

Version publiée le 4 juillet 2012.
— Improved several translations.
— Better validation while creating component.
— Added support for shared git repositories across components.
— Do not necessary commit on every attempt to pull remote repo.
— Added support for offloading indexing.

4.50.10 Weblate 1.0

Version publiée le 10 mai 2012.
— Improved validation while adding/saving component.
— Experimental support for Android component files (needs patched ttkit).
— Updates from hooks are run in background.
— Improved installation instructions.
— Improved navigation in dictionary.

4.51 Weblate 0.x

4.51.1 Weblate 0.9

Version publiée le 18 avril 2012.
— Fixed import of unknown languages.
— Improved listing of nearby messages.
— Improved several checks.
— Modifications de la documentation.
— Added definition for several more languages.
— Various code cleanups.
— Améliorations de la documentation.
— Changed file layout.
— Update helper scripts to Django 1.4.
— Improved navigation while translating.
— Better handling of po file renames.
— Better validation while creating component.
— Integrated full setup into syncdb.
— Added list of recent changes to all translation pages.
— Check for untranslated strings ignores format string only messages.

4.51.2 Weblate 0.8

Version publiée le 3 avril 2012.
— Replaced own full text search with Whoosh.
— Various fixes and improvements to checks.
— New command updatechecks.
— Lot of translation updates.
— Added dictionary for storing most frequently used terms.
— Added /admin/report/ for overview of repositories status.
— Machine translation services no longer block page loading.
— Management interface now contains also useful actions to update data.
— Records log of changes made by users.
— Ability to postpone commit to Git to generate less commits from single user.
— Possibility to browse failing checks.
— Traduction automatique à l'aide de chaînes déjà traduites.
— New about page showing used versions.
— Django 1.4 compatibility.
— Ability to push changes to remote repo from web interface.
— Added review of translations done by others.

4.51.3 Weblate 0.7

Version publiée le 16 février 2012.
— Direct support for GitHub notifications.
— Added support for cleaning up orphaned checks and translations.
— Displays nearby strings while translating.
— Displays similar strings while translating.
— Improved searching for string.

4.51.4 Weblate 0.6

Version publiée le 14 février 2012.
— Added various checks for translated messages.
— Tunable access control.
— Improved handling of translations with new lines.
— Added client side sorting of tables.
— Please check upgrading instructions in case you are upgrading.
4.51.5 Weblate 0.5

Version publiée le 12 février 2012.

— Support for machine translation using following online services:
  — Apertium
  — Microsoft Traduction
  — MyMemory
— Several new translations.
— Improved merging of upstream changes.
— Better handle concurrent git pull and translation.
— Propagating works for fuzzy changes as well.
— Propagating works also for file upload.
— Fixed file downloads while using FastCGI (and possibly others).

4.51.6 Weblate 0.4

Version publiée le 8 février 2012.

— Added usage guide to documentation.
— Fixed API hooks not to require CSRF protection.

4.51.7 Weblate 0.3

Version publiée le 8 février 2012.

— Better display of source for plural translations.
— New documentation in Sphinx format.
— Displays secondary languages while translating.
— Improved error page to give list of existing projects.
— New per language stats.

4.51.8 Weblate 0.2

Version publiée le 7 février 2012.

— Improved validation of several forms.
— Warn users on profile upgrade.
— Remember URL for login.
— Naming of text areas while entering plural forms.
— Automatic expanding of translation area.

4.51.9 Weblate 0.1

Version publiée le 6 février 2012.

— Première version.
Index des modules Python

W
wlc, 147
wlc.config, 147
wlc.main, 148
<table>
<thead>
<tr>
<th>Route</th>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/</td>
<td>ANY</td>
<td></td>
</tr>
<tr>
<td>GET /api/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GET /api/addons</td>
<td>GET</td>
<td></td>
</tr>
<tr>
<td>GET /api/addons/(int:id)</td>
<td>PUT</td>
<td></td>
</tr>
<tr>
<td>DELETE /api/addons/(int:id)</td>
<td>PATCH</td>
<td></td>
</tr>
<tr>
<td>GET /api/changes</td>
<td>GET</td>
<td></td>
</tr>
<tr>
<td>GET /api/changes/(int:id)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GET /api/component-lists</td>
<td>POST</td>
<td></td>
</tr>
<tr>
<td>POST /api/component-lists/(str:slug)/</td>
<td>PUT</td>
<td></td>
</tr>
<tr>
<td>DELETE /api/component-lists/(str:slug)/</td>
<td>DELETE</td>
<td></td>
</tr>
<tr>
<td>PATCH /api/component-lists/(str:slug)/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GET /api/components</td>
<td>GET</td>
<td></td>
</tr>
<tr>
<td>GET /api/components/(string:project)/(string:component)/</td>
<td>POST</td>
<td></td>
</tr>
<tr>
<td>GET /api/components/(string:project)/(string:component)/changes/</td>
<td>PUT</td>
<td></td>
</tr>
<tr>
<td>GET /api/components/(string:project)/(string:component)/file/</td>
<td>DELETE</td>
<td></td>
</tr>
<tr>
<td>GET /api/components/(string:project)/(string:component)/links/</td>
<td>PATCH</td>
<td></td>
</tr>
<tr>
<td>GET /api/components/(string:project)/(string:component)/new_template/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GET /api/components/(string:project)/(string:component)/repository/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GET /api/components/(string:project)/(string:component)/screenshots/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GET /api/components/(string:project)/(string:component)/statistics/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GET /api/components/(string:project)/(string:component)/translations/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GET /api/groups</td>
<td>GET</td>
<td></td>
</tr>
<tr>
<td>GET /api/groups/(int:id)</td>
<td>POST</td>
<td></td>
</tr>
<tr>
<td>POST /api/groups/(int:id)/componentlists/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>POST /api/groups/(int:id)/components/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>POST /api/groups/(int:id)/file/id/components/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>POST /api/groups/(int:id)/changes/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>POST /api/groups/(int:id)/componentlists/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>POST /api/groups/(int:id)/components/</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table de routage HTTP

POST /api/groups/(int:id)/languages/, PUT /api/projects/(string:project)/,
DELETE /api/projects/(string:project)/, PATCH /api/projects/(string:project)/,
DELETE /api/groups/(int:id)/componentlists/(int:component_list_id),
DELETE /api/groups/(int:id)/components/(int:component_id),
DELETE /api/groups/(int:id)/languages/(string:language_code),
DELETE /api/groups/(int:id)/projects/(int:project_id),
DELETE /api/groups/(int:id)/roles/, PUT /api/groups/(int:id)/,
DELETE /api/groups/(int:id)/, /api/roles,
GET /api/roles/, POST /api/roles/, PUT /api/roles/(int:id)/,
DELETE /api/roles/(int:id)/, PATCH /api/roles/(int:id)/,
GET /api/screenshots/, POST /api/screenshots/, GET /api/screenshots/(int:id)/,
POST /api/screenshots/(int:id)/file/, POST /api/screenshots/(int:id)/units/,
DELETE /api/screenshots/(int:id)/units/(int:unit_id), /api/tasks,
GET /api/tasks/, GET /api/tasks/(str:uuid)/,
GET /api/translations/, GET /api/translations/(string:project)/(string:component)/(string:language)/,
GET /api/translations/(string:project)/(string:component)/(string:language)/changes/,
GET /api/translations/(string:project)/(string:component)/(string:language)/file/,
GET /api/translations/(string:project)/(string:component)/(string:language)/repository/,
GET /api/translations/(string:project)/(string:component)/(string:language)/statistics/,
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)/repository/,
POST /api/translations/(string:project)/(string:component)/(string:language)/units/
GET /api/memory/, DELETE /api/memory/(int:memory_object_id)/,
GET /api/metrics/, GET /api/projects/, GET /api/projects/(string:project)/,
GET /api/projects/(string:project)/changes/, GET /api/projects/(string:project)/components/
GET /api/projects/(string:project)/languages/, GET /api/projects/(string:project)/repository/,
GET /api/projects/(string:project)/statistics/
DELETE /api/translations/(string:project)/(string:component)/(string:language)/,
POST /hooks/gitlab/, 138
POST /hooks/pagure/, 138

/api/units
GET /api/units/, 131
GET /api/units/(int:id)/, 131
PUT /api/units/(int:id)/, 132
DELETE /api/units/(int:id)/, 132
PATCH /api/units/(int:id)/, 132

/api/users
GET /api/users/, 104
GET /api/users/(str:username)/, 104
GET /api/users/(str:username)/notifications/, 106
GET /api/users/(str:username)/notifications/(int:subscription_id)/,
106
GET /api/users/(str:username)/statistics/, 105
POST /api/users/, 104
POST /api/users/(str:username)/groups/, 105
POST /api/users/(str:username)/notifications/, 106
PUT /api/users/(str:username)/, 105
PUT /api/users/(str:username)/notifications/(int:subscription_id)/,
106
DELETE /api/users/(str:username)/, 105
DELETE /api/users/(str:username)/groups/, 105
DELETE /api/users/(str:username)/notifications/(int:subscription_id)/,
106
PATCH /api/users/(str:username)/, 105
PATCH /api/users/(str:username)/notifications/(int:subscription_id)/,
106

/exports
GET /exports/rss/, 141
GET /exports/rss/(string:project)/, 141
GET /exports/rss/(string:project)/(string:component)/,
141
GET /exports/rss/(string:project)/(string:component)/(string:language)/,
141
GET /exports/rss/language/(string:language)/,
141
GET /exports/stats/(string:project)/(string:component)/,
140

/hooks
GET /hooks/update/(string:project)/,
138
GET /hooks/update/(string:project)/(string:component)/,
138
POST /hooks/azure/, 139
POST /hooks/bitbucket/, 138
POST /hooks/gitea/, 139
POST /hooks/gitee/, 139
POST /hooks/github/, 138
Symboles

.XML resource file
  file format, 86
--add
  option de ligne de commande
    auto_translate, 368
--addon
  option de ligne de commande install_addon, 375
--age
  option de ligne de commande commit_pending, 369
--author
  option de ligne de commande add_suggestions, 368
--author-email
  option de ligne de commande wlc, 145
--author-name
  option de ligne de commande wlc, 145
--base-file-template
  option de ligne de commande import_project, 373
--check
  option de ligne de commande importusers, 374
--config
  option de ligne de commande wlc, 143
--config-section
  option de ligne de commande wlc, 143
--configuration
  option de ligne de commande install_addon, 375
--convert
  option de ligne de commande wlc, 144
--email
  option de ligne de commande createadmin, 370
--file-format
  option de ligne de commande import_project, 373
--force
  option de ligne de commande loadpo, 376
--force-commit
  option de ligne de commande push-git, 377
--format
  option de ligne de commande wlc, 143
--fuzzy
  option de ligne de commande wlc, 145
--ignore
  option de ligne de commande import_json, 371
--inconsistent
  option de ligne de commande auto_translate, 368
--input
  option de ligne de commande wlc, 144
--key
  option de ligne de commande wlc, 143
--lang
  option de ligne de commande loadpo, 376
--language-code
  option de ligne de commande list_translators, 375
--language-map
  option de ligne de commande import_memory, 372
--language-regex
  option de ligne de commande import_project, 373
--license
  option de ligne de commande import_project, 373
--license-url
  option de ligne de commande import_project, 373
--main-component
  option de ligne de commande import_json, 371
--method
  option de ligne de commande import_project, 373
--mode
  option de ligne de commande loadpo, 376
autotranslate, 369
--mt
   option de ligne de commande
   autotranslate, 368
--name
   option de ligne de commande cre-
   atedadmin, 370
--name-template
   option de ligne de commande im-
   port_project, 373
--new-base-template
   option de ligne de commande im-
   port_project, 373
--no-password
   option de ligne de commande cre-
   atedadmin, 370
--no-privs-update
   option de ligne de commande setup-
   groups, 377
--no-projects-update
   option de ligne de commande setup-
   groups, 377
--no-update
   option de ligne de commande setu-
   plang, 378
--output
   option de ligne de commande wlc, 144
--overwrite
   option de ligne de commande autotranslate, 368
   option de ligne de commande wlc, 144
--password
   option de ligne de commande cre-
   atedadmin, 370
--project
   option de ligne de commande im-
   port_json, 371
--source
   option de ligne de commande autotranslate, 368
--threshold
   option de ligne de commande autotranslate, 369
--update
   option de ligne de commande cre-
   atedadmin, 370
   option de ligne de commande im-
   port_json, 371
   option de ligne de commande ins-
   tall_addon, 375
--url
   option de ligne de commande wlc, 143
--user
   option de ligne de commande autotranslate, 368
--username
   option de ligne de commande cre-
   atedadmin, 370
--vcs
   option de ligne de commande im-
   port_project, 373

A
add_suggestions
   weblate admin command, 368
ADMIN
   setting, 196
AKISMET_API_KEY
   setting, 326
ALLOWED_HOSTS
   setting, 196
Android
   file format, 80
ANONYMOUS_USER_NAME
   setting, 326
API
   setting, 146
APPLStrings
   file format, 81
ARB
   file format, 84
AUDITLOG_EXPIRY
   setting, 326
AUTH_LOCK_ATTEMPTS
   setting, 326
AUTH_TOKEN_VALID
   setting, 327
toauto_translate
   weblate admin command, 368
AUTO_UPDATE
   setting, 327
AUTO_FIX_LIST
   setting, 328
AVATAR_URL_PREFIX
   setting, 327

B
BACKGROUND_TASKS
   setting, 328
BaseAddon (classe dans weblate.addons.base), 413
BASIC_LANGUAGES
   setting, 329
bilingual
   translation, 71
BITBUCKETSERVER_CREDENTIALS
   setting, 337
BORG_EXTRA_ARGS
   setting, 329

C
CACHE_DIR
   setting, 329
can_install() (méthode de la classe weblate.addons.base.BaseAddon), 413
CELELY_BACKUP_OPTIONS, 154, 170
CELELY_BEAT_OPTIONS, 154, 170
CELELY_MAIN_OPTIONS, 154, 170
CELERY_MEMORY_OPTIONS, 154, 170
CELERY_NOTIFY_OPTIONS, 154, 170
celery_queues
    weblate admin command, 369
CELERY_TRANSLATE_OPTIONS, 154, 170
changes
    option de ligne de commande wlc, 144
CHECK_LIST
    setting, 330
cHECKgit
    weblate admin command, 369
cleanup
    option de ligne de commande wlc, 144
cleanup_ssh_keys
    weblate admin command, 370
cleanuptrans
    weblate admin command, 370
Comma separated values
    file format, 86
Command (classe dans wlc.main), 148
COMMENT_CLEANUP_DAYS
    setting, 330
commit
    option de ligne de commande wlc, 143
commit_pending
    weblate admin command, 369
COMMIT_PENDING_HOURS
    setting, 331
commitgit
    weblate admin command, 369
configure() (méthode weblate.addons.base.BaseAddon), 413
CONTACT_FORM
    setting, 331
createadmin
    weblate admin command, 370
CSP_CONNECT_SRC
    setting, 330
CSP_FONT_SRC
    setting, 330
CSP_IMG_SRC
    setting, 330
CSP_SCRIPT_SRC
    setting, 330
CSP_STYLE_SRC
    setting, 330
CSV
    file format, 86
D
    daily() (méthode weblate.addons.base.BaseAddon), 413
DATA_DIR
    setting, 331
DATABASE_BACKUP
    setting, 332
DATABASES
    setting, 196
DEBUG
    setting, 196
DEFAULT_ACCESS_CONTROL
    setting, 332
DEFAULT_ADD_MESSAGE
    setting, 333
DEFAULT_ADDON_MESSAGE
    setting, 333
DEFAULT_ADDONS
    setting, 333
DEFAULT_AUTO_WATCH
    setting, 332
DEFAULT_COMMIT_MESSAGE
    setting, 333
DEFAULT_COMMITER_EMAIL
    setting, 333
DEFAULT_COMMITER_NAME
    setting, 333
DEFAULT_DELETE_MESSAGE
    setting, 333
DEFAULT_FROM_EMAIL
    setting, 196
DEFAULT_LANGUAGE
    setting, 334
DEFAULT_MERGE_MESSAGE
    setting, 333
DEFAULT_MERGE_STYLE
    setting, 334
DEFAULT_PAGE_LIMIT
    setting, 342
DEFAULT_PULL_MESSAGE
    setting, 334
DEFAULT_RESTRICTED_COMPONENT
    setting, 332
DEFAULT_SHARED_TM
    setting, 334
DEFAULT_TRANSLATION_PROPAGATION
    setting, 334
download
    option de ligne de commande wlc, 144
DTD
    file format, 88
dump_memory
    weblate admin command, 371
dumpuserdata
    weblate admin command, 371
E
    ENABLE_AVATARS
        setting, 335
    ENABLE_HOOKS
        setting, 335
    ENABLE_HTTPS
        setting, 335
    ENABLE_SHARING
        setting, 335
    EXTRA_HTML_HEAD
        setting, 335
file format

- .XML resource file
- Android
- Apple strings
- ARB
- Comma separated values
- CSV
- DTD
- gettext
- go-i18n
- gotext
- GWT properties
- i18next
- INI translations
- Java properties
- Joomla translations
- JSON
- mi18n lang
- PHP strings
- PO
- Qt
- RC
- ResourceDictionary
- RESX
- Ruby YAML
- Ruby YAML Ain't Markup Language
- string resources
- TS
- WPF
- XLIFF
- XML
- YAML
- YAML Ain't Markup Language

get() (méthode wlc.Weblate)

get_add_form() (méthode de la classe weblate.addons.base.BaseAddon)

GET_HELP_URL

get_settings_form() (méthode weblate.addons.base.BaseAddon)

ggettext

file format
Index 473

M
main() (dans le module wlc.main), 148
MATOMO_SITE_ID
setting, 341
MATOMO_URL
setting, 341
mi18n lang
file format, 77
module
wlc, 147
wlc.config, 147
wlc.main, 148
monolingual
translation, 71
move_language
weblate admin command, 376

N
NEARBY_MESSAGES
setting, 342

O
option de ligne de commande
add_suggestions
--author, 368
option de ligne de commande
auto_translate
--add, 368
--inconsistent, 368
--mode, 369
--mt, 368
--overwrite, 368
--source, 368
--threshold, 369
--user, 368
option de ligne de commande commit_pending
--age, 369
option de ligne de commande createadmin
--email, 370
--name, 370
--no-password, 370
--password, 370
--update, 370
--username, 370
option de ligne de commande import_json
--ignore, 371
--main-component, 371
--project, 371
--update, 371
option de ligne de commande import_memory
--language-map, 372
option de ligne de commande import_project
--base-file-template, 373
--file-format, 373
--language-regex, 373
--license, 373
--license-url, 373
--main-component, 373
--name-template, 373
--new-base-template, 373
--vcs, 373
option de ligne de commande importusers
--check, 374
option de ligne de commande install_addon
--addon, 375
--configuration, 375
--update, 375
option de ligne de commande list_translators
--language-code, 375
option de ligne de commande loadpo
--force, 376
--lang, 376
option de ligne de commande pushgit

Index
---force-commit, 377
option de ligne de commande setup-
groups
---no-privs-update, 377
---no-projects-update, 377
option de ligne de commande setuplang
---no-update, 378
option de ligne de commande wlc
---author-email, 145
---author-name, 145
---config, 143
---config-section, 143
---convert, 144
---format, 143
---fuzzy, 145
---input, 144
---key, 143
---method, 144
---output, 144
---overwrite, 144
---url, 143
changes, 144
cleanup, 144
commit, 143
download, 144
list-components, 143
list-languages, 143
list-projects, 143
list-translations, 143
lock, 144
lock-status, 144
ls, 143
pull, 144
push, 144
repo, 144
reset, 144
show, 143
stats, 144
unlock, 144
upload, 144
version, 143

P
PAGURE_CREDENTIALS
setting, 342
PHP strings
file format, 81
PIWIK_SITE_ID
setting, 341
PIWIK_URL
setting, 341
PO
file format, 74
post() (méthode wlc.Weblate), 147
post_add() (méthode wlc.addons.base.BaseAddon), 413
post_commit() (méthode wlc.addons.base.BaseAddon), 413
post_push() (méthode wlc.addons.base.BaseAddon), 413

PAGURE_CREDENTIALS
setting, 342
PHP strings
file format, 81
PIWIK_SITE_ID
setting, 341
PIWIK_URL
setting, 341
PO
file format, 74
post() (méthode wlc.Weblate), 147
post_add() (méthode wlc.addons.base.BaseAddon), 413
post_commit() (méthode wlc.addons.base.BaseAddon), 413
PRIVACY_URL, 342
PRIVATE_COMMIT_EMAIL_OPT_IN, 342
PRIVATE_COMMIT_EMAIL_TEMPLATE, 343
PROJECT_BACKUP_KEEP_COUNT, 343
PROJECT_BACKUP_KEEP_DAYS, 343
PROJECT_NAME_RESTRICT_RE, 343
PROJECT_WEB_RESTRICT_RE, 343
RATELIMIT_ATTEMPTS, 344
RATELIMIT_LOCKOUT, 344
RATELIMIT_WINDOW, 344
REGISTRATION_ALLOW_BACKENDS, 344
REGISTRATION_EMAIL_MATCH, 345
REGISTRATION_OPEN, 345
REGISTRATION_REBIND, 345
REPOSITORY_ALERT_THRESHOLD, 345
RATELIMIT_ATTEMPTS, 344
RATELIMIT_LOCKOUT, 344
RATELIMIT_WINDOW, 344
SECRET_KEY, 196
SERVER_EMAIL, 196
SESSION_COOKIE_AGE_AUTHENTICATED, 346
SESSION_ENGINE, 196
SIMPLIFY_LANGUAGES, 346
SINGLE_PROJECT, 347
SITE_DOMAIN, 346
SITE_TITLE, 347
SPECIAL_CHARS, 347
SSH_EXTRA_ARGS, 347
STATUS_URL, 348
SUGGESTION_CLEANUP_DAYS, 348
UPDATE_LANGUAGES, 348
VCS_API_DELAY, 349
VCS_BACKENDS, 349
VCS_CLONE_DEPTH, 349
WEBLATE_ADDONS, 349
WEBLATE_EXPORTERS, 350
WEBLATE_FORMATS, 350
WEBLATE_GPG_IDENTITY, 351
WEBLATE_MACHINERY, 351
POSTGRES_ALTER_ROLE, 154, 170
POSTGRES_BEAT_OPTIONS, 154, 170
POSTGRES_MAIN_OPTIONS, 154, 170
POSTGRES_MEMORY_OPTIONS, 154, 170
POSTGRES_NOTIFY_OPTIONS, 154, 170
POSTGRES_TRANSLATE_OPTIONS, 154, 170
POSTGRES_DISABLE_SERVER_SIDE_CURSORS, 166
POSTGRES_ALTER_ROLE, 166
POSTGRES_CONN_MAX_AGE, 166
POSTGRES_DATABASE, 165
POSTGRES_DISABLE_SERVER_SIDE_CURSORS, 166
POSTGRES_HOST, 165
POSTGRES_PASSWORD, 165
POSTGRES_PASSWORD_FILE, 165
POSTGRES_PORT, 166
POSTGRES_SSL_MODE, 166
POSTGRES_USER, 165
WEBLATE_NO_EMAIL_AUTH, 165
WEBLATE_PAGURE_HOST, 158
WEBLATE_PAGURE_TOKEN, 158
WEBLATE_PAGURE_USERNAME, 158
WEBLATE_PRIVACY_URL, 168
WEBLATE_PRIVATE_COMMIT_EMAIL_OPT_IN, 161
WEBLATE_PRIVATE_COMMIT_EMAIL_TEMPLATE, 160
WEBLATE_RATELIMIT_ATTEMPTS, 160, 387
WEBLATE_RATELIMIT_LOCKOUT, 160
WEBLATE_RATELIMIT_WINDOW, 160
WEBLATE_REGISTRATION_ALLOW_BACKENDS, 156
WEBLATE_REGISTRATION_OPEN, 156
WEBLATE_REGISTRATION_REBIND, 156
WEBLATE_REMOVE_ADDONS, 170
WEBLATE_REMOVE_APPS, 169
WEBLATE_REMOVE_AUTOFIX, 169
WEBLATE_REMOVE_CHECK, 169
WEBLATE_REMOVE_LOGIN_REQUIRED_URLS_EXCEPTIONS, 158
WEBLATE_REQUIRE_LOGIN, 157, 346
WEBLATE_SAML_IDP_ENTITY_ID, 165
WEBLATE_SAML_IDP_IMAGE, 165
WEBLATE_SAML_IDP_TITLE, 165
WEBLATE_SAML_IDP_URL, 165
WEBLATE_SAML_IDP_X509CERT, 165
WEBLATE_SECURE_PROXY_SSL_HEADER, 157
WEBLATE_SERVER_EMAIL, 155
WEBLATE_SERVICE, 154, 170
WEBLATE_SILENCED_SYSTEM_CHECKS, 159, 226
WEBLATE_SIMPLIFY_LANGUAGES, 158
WEBLATE_SITE_DOMAIN, 155, 198, 215, 347
WEBLATE_SITE_TITLE, 155
WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_KEY, 163
WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_SECRET, 163
WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_WHITELISTED_DOMAINS, 163
WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_WHITELISTED_EMAILS, 163
WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_KEY, 163
WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_SECRET, 163
WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_TENANT_ID, 163
WEBLATE_SOCIAL_AUTH_BITBUCKET_KEY, 163
WEBLATE_SOCIAL_AUTH_BITBUCKET_OAUTH2_KEY, 163
WEBLATE_SOCIAL_AUTH_BITBUCKET_OAUTH2_SECRET, 163
WEBLATE_SOCIAL_AUTH_FACEBOOK_KEY, 163
WEBLATE_SOCIAL_AUTH_FACEBOOK_SECRET, 163
WEBLATE_SOCIAL_AUTH_GITHUB_KEY, 162
WEBLATE_SOCIAL_AUTH_GITHUB_ORG_KEY, 162
WEBLATE_SOCIAL_AUTH_GITHUB_ORG_NAME, 162
WEBLATE_SOCIAL_AUTH_GITHUB_ORG_SECRET, 162
WEBLATE_SOCIAL_AUTH_GITHUB_SECRET, 162
WEBLATE_SOCIAL_AUTH_GITHUB_TEAM_ID, 162
WEBLATE_SOCIAL_AUTH_GITHUB_TEAM_KEY, 162
WEBLATE_SOCIAL_AUTH_GITHUB_TEAM_SECRET, 162
WEBLATE_SOCIAL_AUTH_GITLAB_API_URL, 163
WEBLATE_SOCIAL_AUTH_GITLAB_KEY, 163
WEBLATE_SOCIAL_AUTH_GITLAB_SECRET, 163
WEBLATE_SOCIAL_AUTH_GITLAB_API_TOKEN, 163
WEBLATE_SOCIAL_AUTH_GITLAB_OAUTH2_TOKEN, 163
WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_KEY, 163
WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET, 163
WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_WHITELISTED_DOMAINS, 163
WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_WHITELISTED_EMAILS, 163
WEBLATE_SOCIAL_AUTH_KEYCLOAK_ACCESS_TOKEN_URL, 164
WEBLATE_SOCIAL_AUTH_KEYCLOAK_KEY, 164
WEBLATE_SOCIAL_AUTH_KEYCLOAK_SECRET, 164
WEBLATE_SOCIAL_AUTH_KEYCLOAK_TITLE, 164
WEBLATE_SOCIAL_AUTH_OIDC_KEY, 165
WEBLATE_SOCIAL_AUTH_OIDC_OIDC_ENDPOINT, 165
WEBLATE_SOCIAL_AUTH_OIDC_SECRET, 165
WEBLATE_SOCIAL_AUTH_OIDC_USERNAME_KEY, 165
WEBLATE_SOCIAL_AUTH_OIDC_OIDC_ENDPOINT, 165
WEBLATE_SOCIAL_AUTH_OIDC_SECRET, 165
WEBLATE_SOCIAL_AUTH_OIDC_USERNAME_KEY, 165
WEBLATE_SOCIAL_AUTH_OIDC_OIDC_ENDPOINT, 165
WEBLATE_SOCIAL_AUTH_OIDC_SECRET, 165
WEBLATE_SOCIAL_AUTH_OIDC_USERNAME_KEY, 165
The Weblate Manual, Version 4.16

WEBLATE_SOCIAL_AUTH_OPENSUSE, 164
WEBLATE_SOCIAL_AUTH_SLACK_KEY, 164
WEBLATE_SOCIAL_AUTH_UBUNTU, 164
WEBLATE_SSL_EXTRA_ARGS, 160
WEBLATE_STATUS_URL, 168
WEBLATE_TIME_ZONE, 156
WEBLATE_URL_PREFIX, 159
WEBLATE_WEBSITE_REQUIRED, 159
WEBLATE_WORKERS, 154, 170
WL_BRANCH, 323
WL_COMPONENT_NAME, 323
WL_COMPONENT_SLUG, 323
WL_COMPONENT_URL, 323
WL_ENGAGE_URL, 323
WL_FILE_FORMAT, 323
WL_FILEMASK, 323
WL_LANGUAGE, 323
WL_NEW_BASE, 323
WL_PATH, 322
WL_PREVIOUS_HEAD, 323
WL_PROJECT_NAME, 323
WL_PROJECT_SLUG, 323
WL_REPO, 322
WL_TEMPLATE, 323
WL_VCS, 322
VCS_API_DELAY
setting, 349
VCS_BACKENDS
setting, 349
VCS_CLONE_DEPTH
setting, 349
version
option de ligne de commande wlc, 143

W
WEB_WORKERS, 154, 170
Weblate (classe dans wlc), 147
weblate admin command
add_suggestions, 368
auto_translate, 368
celery_queues, 369
checkgit, 369
cleanup_ssh_keys, 370
cleanuptrans, 370
commit_pending, 369
commitgit, 369
createdadmin, 370
dump_memory, 371
dumpuserdata, 371
import_demo, 371
import_json, 371
import_memory, 372
import_project, 372
importuserdata, 374
importusers, 374
install_addon, 375
list_languages, 375
list_translators, 375
list_versions, 376
loadpo, 376
lock_translation, 376
move_language, 376
pushgit, 377
setupgroups, 377
updatelang, 378
unlock_translation, 377
updatechecks, 378
updatelog, 378
WEBLATE_ADDONS
setting, 349
WEBLATE_ADMIN_EMAIL, 154, 155
WEBLATE_ADMIN_NAME, 154, 155
WEBLATE_ADMIN_PASSWORD, 150, 154, 155
WEBLATE_ADMIN_PASSWORD_FILE, 155
WEBLATE_AKISMET_API_KEY, 385
WEBLATE_ALLOWED_HOSTS, 196, 200, 347
WEBLATE_API_RATELIMIT_ANON, 103
WEBLATE_API_RATELIMIT_USER, 103
WEBLATE_AUTH_LDAP_BIND_PASSWORD, 161
WEBLATE_DEBUG, 219
WEBLATE_EMAIL_HOST_PASSWORD, 168
WEBLATE_EMAIL_PORT, 167, 168
WEBLATE_EMAIL_USE_SSL, 167, 168
WEBLATE_EMAIL_USE_TLS, 167, 168
WEBLATE_ENABLE_HTTPS, 229
WEBLATE_EXPORTERS
setting, 350
WEBLATE_FORMATS
setting, 350
WEBLATE_GITHUB_HOST, 219
WEBLATE_GPG_IDENTITY
setting, 351
WEBLATE_LOCALIZE_CDN_PATH, 169
WEBLATE_MACHINERY
setting, 351
WEBLATE_RATELIMIT_ATTEMPTS, 387
WEBLATE_REQUIRE_LOGIN, 346
WEBLATE_SECURE_PROXY_SSL_HEADER, 157
WEBLATE_SERVICE, 154
WEBLATE_SILENCED_SYSTEM_CHECKS, 226
WEBLATE_SITE_DOMAIN, 198, 215, 347
WEBLATE_WORKERS, 154, 170
WeblateConfig (classe dans wlc.config), 147
WeblateException, 147
WEBSITE_REQUIRED
setting, 351
wlc, 141
module, 147
wlc.config
module, 147
wlc.main
module, 148
WPFF
file format, 86
X
  XLIFF
    file format. 75
  XML
    file format. 89

Y
  YAML
    file format. 87
  YAML Ain't Markup Language
    file format. 87