The Weblate Manual

Выпуск 4.3

Michal Čihař

окт. 15, 2020

Пользовательская документация

1 Пользовательская документация **1**
1.1 Основы Weblate . . . 1
1.2 Регистрация и профиль пользователя . . . 1
1.3 Перевод с помощью Weblate . . . 10
1.4 Скачивание и загрузка переводов . . . 21
1.5 Проверки и исправления . . . 23
1.6 Поиск . . . 39
1.7 Руководство разработчика приложений . . . 43
1.8 Рабочие процессы перевода . . . 64
1.9 Часто задаваемые вопросы . . . 67
1.10 Поддерживаемые форматы файлов . . . 75
1.11 Интеграция с системой контроля версий . . . 94
1.12 REST API Weblate'a . . . 101
1.13 Клиент Weblate . . . 146
1.14 Python API Weblate'a . . . 151

2 Документация для администратора **153**
2.1 Инструкции по настройке . . . 153
2.2 Развёртывание Weblate . . . 209
2.3 Обновление Weblate . . . 210
2.4 Резервное копирование и перенос Weblate . . . 215
2.5 Аутентификация . . . 221
2.6 Управление доступом . . . 230
2.7 Проекты перевода . . . 238
2.8 Определения языка . . . 255
2.9 Непрерывная локализация . . . 256
2.10 Лицензирование переводов . . . 266
2.11 Процесс перевода . . . 268
2.12 Проверки и исправления . . . 274
2.13 Машинный перевод . . . 282
2.14 Надстройки . . . 289
2.15 Память переводов . . . 298
2.16 Конфигурация . . . 300
2.17 Пример конфигурации . . . 327
2.18 Команды управления . . . 342
2.19 Объявления . . . 353
2.20 Списки компонентов . . . 355
2.21 Дополнительные модули Weblate . . . 356
2.22 Настройка Weblate . . . 361
2.23 Интерфейс управления . . . 363
2.24 Getting support for Weblate . . . 370

2.25 Правовые документы . . . 371

3 Документация для участников **373**
3.1 Внесение вклада в Weblate . . . 373
3.2 Starting contributing code to Weblate . . . 374
3.3 Исходный код Weblate . . . 378
3.4 Отладка Weblate . . . 379
3.5 Weblate internals . . . 380
3.6 Weblate frontend . . . 381
3.7 Сообщение о проблемах в Weblate'е . . . 382
3.8 Weblate testsuite and continuous integration . . . 382
3.9 Схемы данных . . . 384
3.10 Releasing Weblate . . . 388
3.11 О Weblate . . . 389
3.12 Лицензия . . . 389

4 История изменений **390**
4.1 Weblate 4.3 . . . 390
4.2 Weblate 4.2.2 . . . 391
4.3 Weblate 4.2.1 . . . 391
4.4 Weblate 4.2 . . . 391
4.5 Weblate 4.1.1 . . . 392
4.6 Weblate 4.1 . . . 393
4.7 Weblate 4.0.4 . . . 394
4.8 Weblate 4.0.3 . . . 394
4.9 Weblate 4.0.2 . . . 394
4.10 Weblate 4.0.1 . . . 395
4.11 Weblate 4.0 . . . 395
4.12 Серия Weblate 3.x . . . 396
4.13 Серия Weblate 2.x . . . 408
4.14 Серия Weblate 1.x . . . 419
4.15 Серия Weblate 0.x . . . 423

Содержание модулей Python **427**

HTTP Routing Table **428**

Алфавитный указатель **431**

ГЛАВА 1

Пользовательская документация

1.1 Основы Weblate

1.1.1 Структура проекта

Переводы в Weblate организованы по проектам и компонентам. Каждый проект может содержать несколько компонентов, а те — переводы на отдельные языки. Компоненту соответствует один переводимый файл (например, *GNU gettext* или *Строковые ресурсы Android*). Проекты служат для организации компонентов в логические наборы (например, для группировки всех переводов, используемых в одном приложении).

Внутри каждого проекта переводы общих строк по умолчанию распространяются на другие его компоненты. Это облегчает бремя повторного и много-версионного перевода. При отключении этого распространения в *настройках компонента* для результирующих переводов, кажущихся непоследовательными, всё равно будут генерироваться ошибки.

1.2 Регистрация и профиль пользователя

1.2.1 Регистрация

По умолчанию любой может просматривать проекты, смотреть или предлагать переводы. Право сохранять изменения имеют только зарегистрированные пользователи и они упоминаются как авторы каждого сделанного ими перевода.

Вы можете зарегистрироваться, выполнив несколько простых шагов:

1. Заполните форму регистрации своими учётными данными.
2. Активируйте регистрацию, пройдя по ссылке в полученном вами электронном письме.
3. Дополнительно настройте свой профиль, выбрав, какие языки вы знаете.

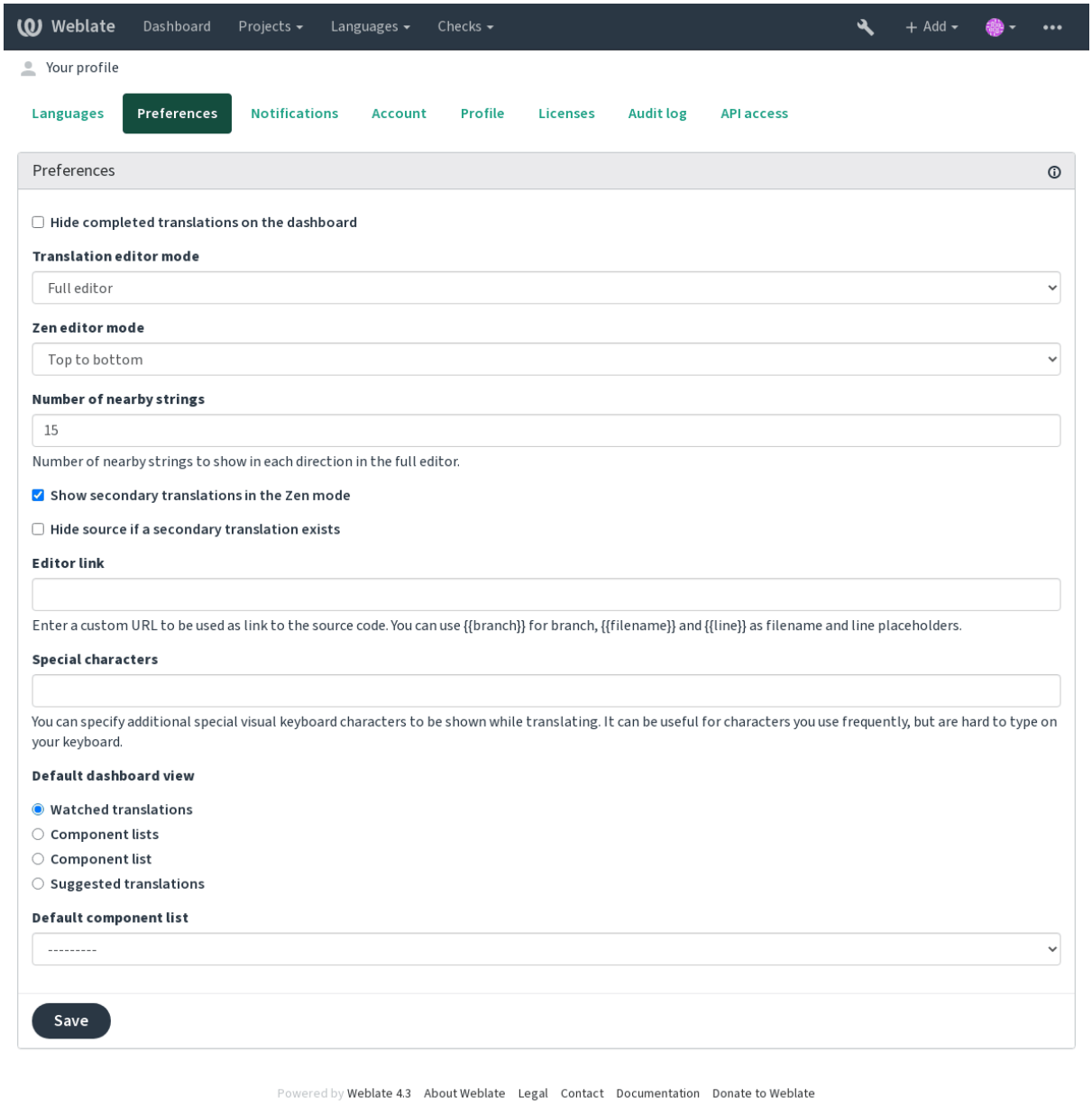
1.2.2 Панель управления

При входе в систему вы увидите сводку по проектам и их компонентам, а также информацию о завершённости их перевода.

Добавлено в версии 2.5.

По умолчанию показываются компоненты отслеживаемых вами проектов со ссылками на предпочитаемые вами языки.

Подсказка: При помощи вкладок навигации вы можете переключиться на другие представления.



Меню содержит следующие параметры:

- Ссылка *Проекты > Просмотр всех проектов* в главном меню показывает статус перевода каждого проекта на экземпляре Weblate.
- Выборе языка в главном меню *Языки* покажет статус перевода всех проектов, отфильтрованных по одному из ваших основных языков.

- На вкладке *Отслеживаемые переводы* панели управления показывается статус перевода только тех проектов, которые вы отслеживаете, отфильтрованных по вашим основным языкам.

Кроме того, в выпадающем списке может отображаться любое количество *списков компонентов*, наборов компонентов проекта, предварительно настроенных администратором Weblate, смотрите раздел *Списки компонентов*.

Вы можете настроить свой личный вид панели управления по умолчанию на вкладке *Настройки* параметров вашего профиля пользователя.

Примечание: Когда Weblate настроен для одного проекта при помощи параметра `SINGLE_PROJECT` в файле `settings.py` (смотрите раздел *Конфигурация*), панель управления показываться не будет, а пользователь вместо этого будет перенаправлен на один этот проект или компонент.

1.2.3 Профиль пользователя

Доступ к профилю пользователя осуществляется щелчком по иконке вашего пользователя в верхней правой части верхнего меню и выбором пункта *Параметры*.

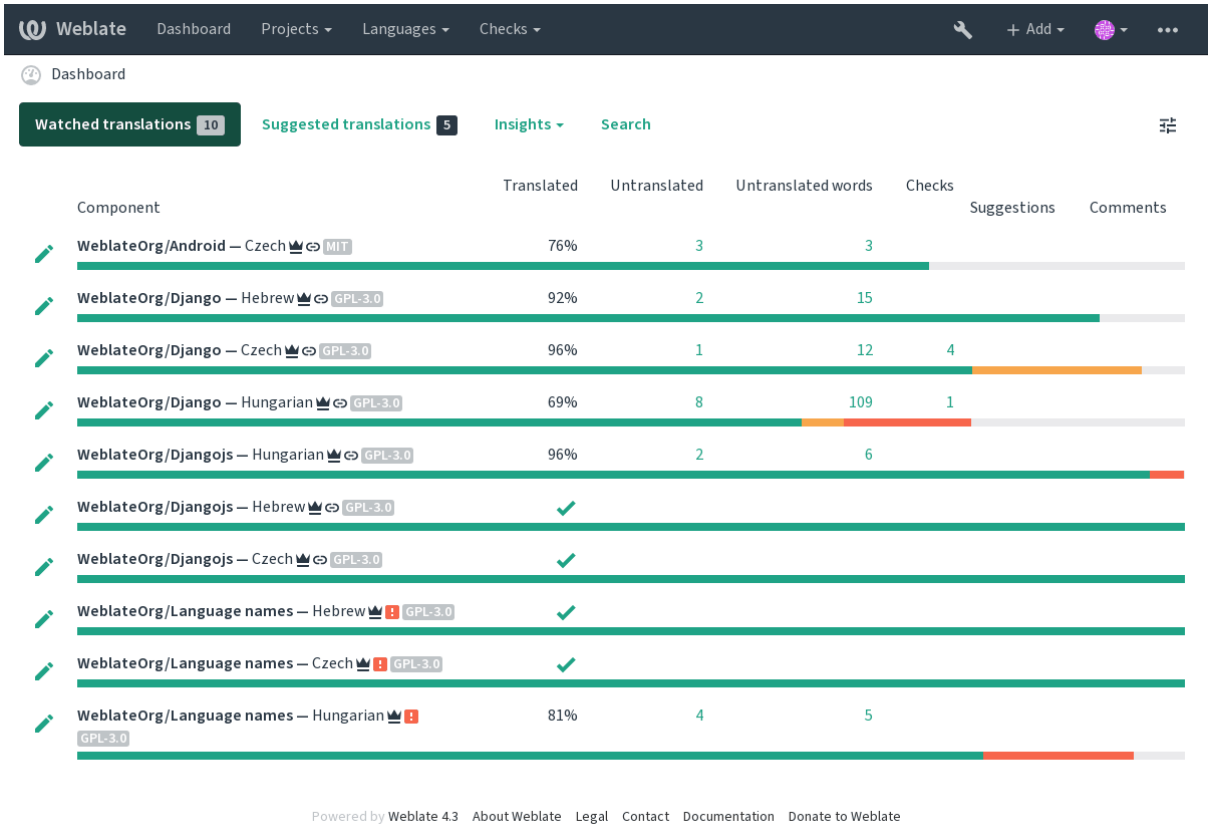
Профиль пользователя содержит ваши предпочтения. Имя и адрес электронной почты используются в коммитах системы контроля версий, поэтому держите эту информацию точной.

Примечание: Все пункты настроек, связанные с выбором языка, предлагают выбор только из языков, переводимых в настоящее время.

Подсказка: Чтобы сделать доступными и другие языки, на которые вы хотите переводить, запросите их добавление или добавьте их, нажав на соответствующую кнопку.

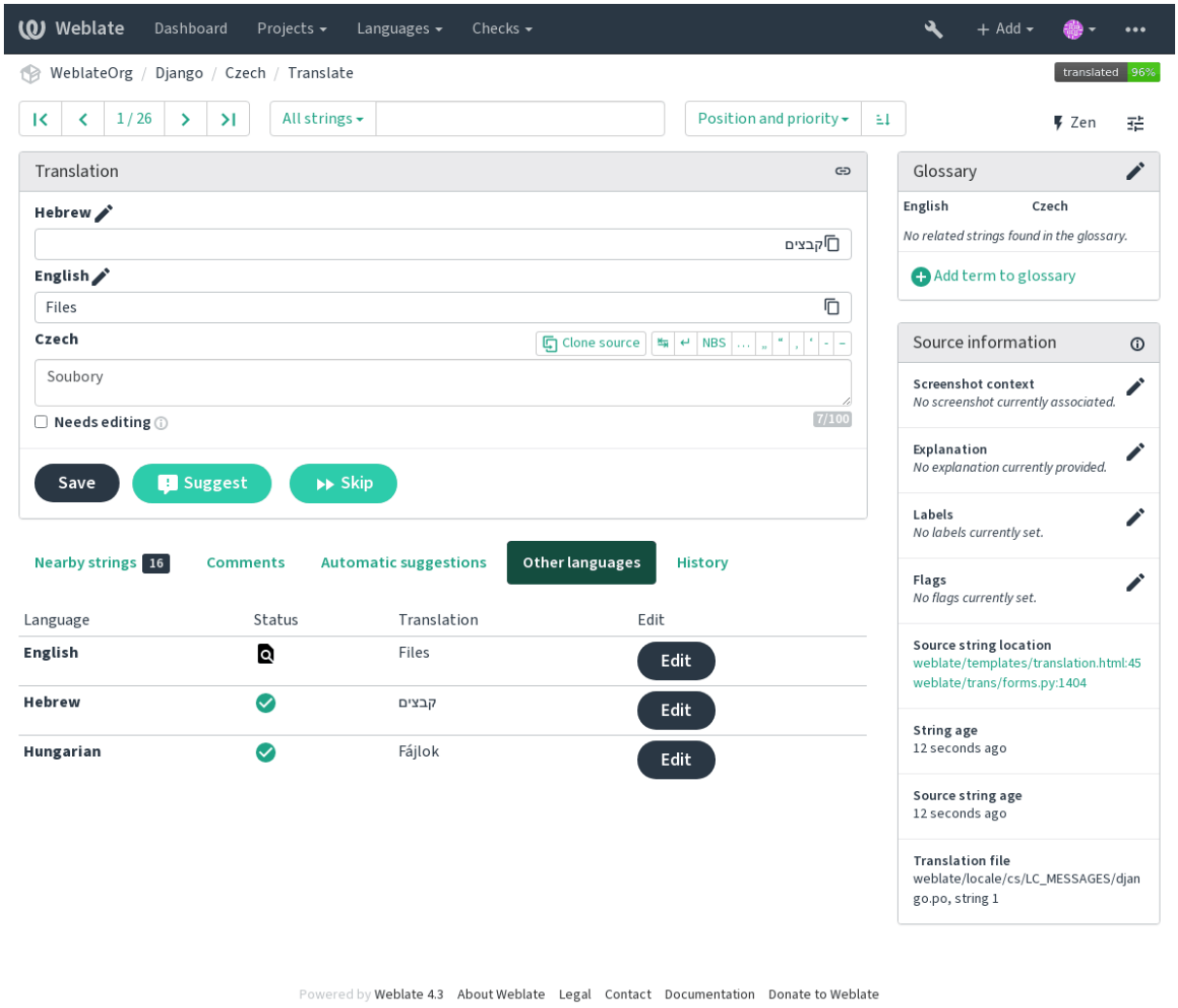
Языки перевода

Выберите, на какие языки вы предпочитаете переводить, и они будут показаны на главной странице отслеживаемых проектов, так что доступ ко всем переводам на каждый из этих языков для вас упростится.



Вспомогательные языки

Вы можете определить вспомогательные языки, которые будут вам показываться при переводе для ориентира. Пример показан на следующем рисунке, на котором вспомогательным языком выбран иврит:



Вид панели управления по умолчанию

На вкладке *Настройки* можно выбрать, какой из доступных видов панели управления будет показан по умолчанию. При выборе пункта *Список компонентов* необходимо будет выбрать из раскрывающегося списка *Список компонентов по умолчанию*, какой именно список компонентов будет отображаться.

См.также:

Списки компонентов

Общедоступный профиль

Все поля на этой странице являются необязательными и в любое время могут быть удалены, и заполнив их, вы даёте нам согласие на показ этих данных везде, где появляется ваш профиль пользователя.

Аватар может показываться для каждого пользователя (это зависит от параметра `ENABLE_AVATARS`). Эти изображения получаются от сервиса https://gravatar.com/.

Ссылка на редактор

Ссылка на исходный код показывается в web-браузере, настроенном по умолчанию в разделе *Конфигурация компонента*.

Подсказка: Установив *Ссылку на редактор*, вы будете использовать свой локальный редактор для открытия исходного кода файла из системы контроля версий с переведёнными строками. Вы можете использовать *шаблонную разметку*.

Обычно хорошим выбором является что-то вроде `editor://open/?file={{filename}}&line={{line}}`.

См.также:

Более подробную информацию о регистрации пользовательских URL-протоколов для редактора вы можете найти в документации к Nette (на английском языке).

1.2.4 Уведомления

На вкладке *Уведомления* можно подписаться на различные уведомления. Уведомления по выбранным событиям для отслеживаемых или управляемых проектов будут высылаться вам по электронной почте.

Некоторые уведомления отправляются только для событий, связанных с вашим языком (например, уведомление о новых строках для перевода), в то время как другие срабатывают на уровне компонента (например, уведомление об ошибках слияния). Эти две группы уведомлений в настройках визуально разделены.

Вы можете включать или отключать уведомления для отслеживаемых и управляемых проектов, а в дальнейшем их можно настроить (или полностью отключить) для каждого проекта и компонента. Посетите страницу компонента и выберите соответствующий пункт в меню *Отслеживание*.

Примечание: Вы не будете получать уведомления о собственных действиях.

Weblate Dashboard Projects Languages Checks + Add

Your profile

Languages Preferences Notifications Account Profile Licenses Audit log API access

Watched projects

Watched projects

Search...

Available:

WeblateOrg

Chosen:

WeblateOrg

You can receive notifications for watched projects and they are shown on the dashboard by default.

Add all projects you want to translate to see them as watched projects on the dashboard.

Save

Notification settings

Watched projects Managed projects

Component wide notifications

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

Repository failure	Do not notify
Repository operation	Do not notify
Component locking	Do not notify
Changed license	Do not notify
Parse error	Do not notify
Comment on own translation	Instant notification
Mentioned in comment	Instant notification
New language	Do not notify
New translation component	Do not notify
New announcement	Instant notification
New alert	Do not notify

Translation notifications

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

New string	Do not notify
New contributor	Do not notify
New suggestion	Do not notify
New comment	Do not notify
Changed string	Do not notify
Translated string	Do not notify
Approved string	Do not notify
Pending suggestions	Do not notify
Strings needing action	Do not notify

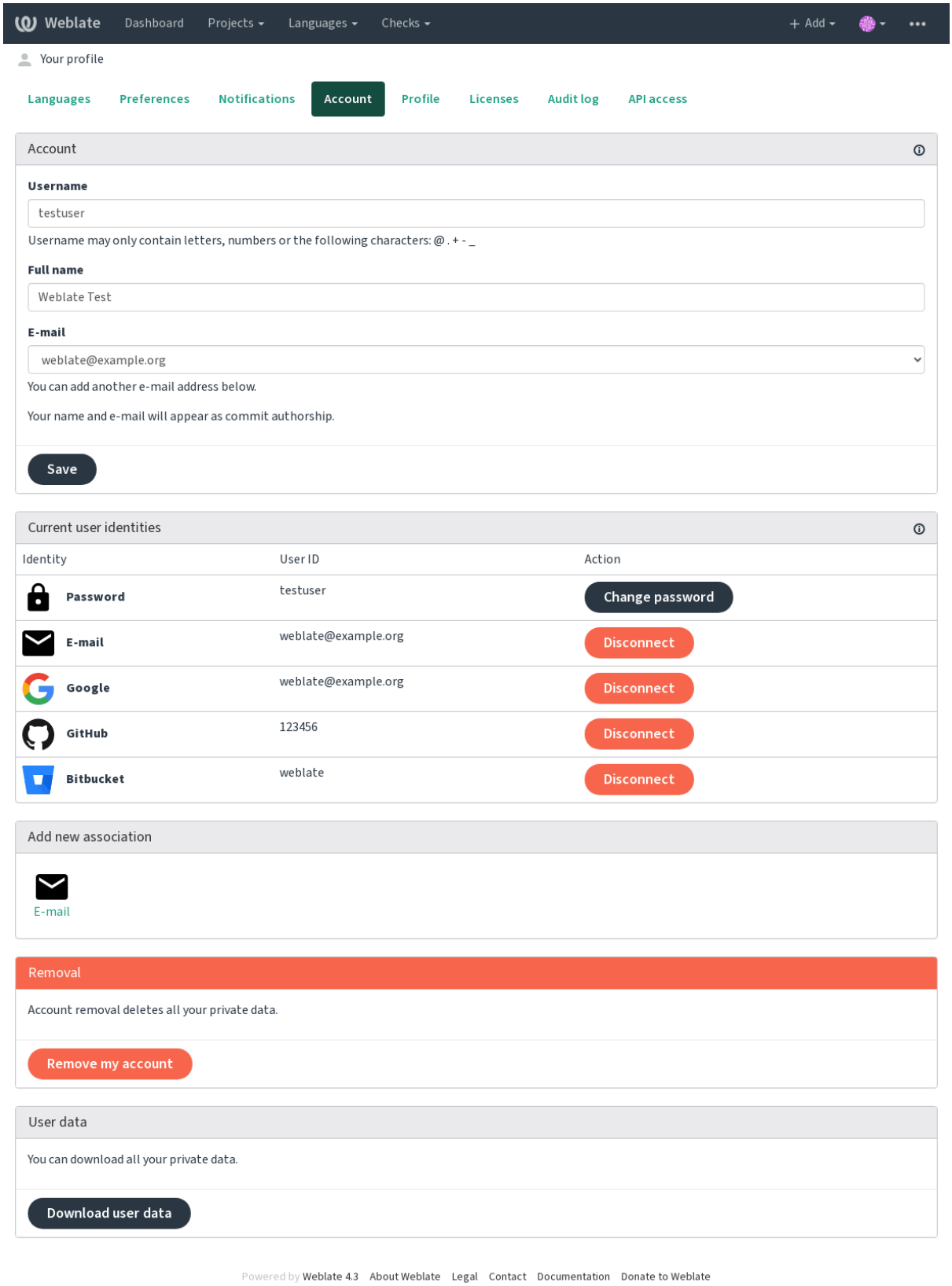
Save

Powered by Weblate 4.3 About Weblate Legal Contact Documentation Donate to Weblate

1.2.5 Учётная запись

Вкладка *Учётная запись* позволяет вам настроить основные данные учётной записи, подключить различные сервисы, которые вы можете использовать для входа в Weblate, полностью удалить свою учётную запись или скачать свои данные пользователя (смотрите раздел *Файл экспорта данных пользователя Weblate'а*).

Примечание: Список сервисов зависит от вашей конфигурации Weblate и может включать такие популярные сайты, как GitLab, GitHub, Google, Facebook или Bitbucket, а также других провайдеров OAuth 2.0.


Weblate
Dashboard
Projects
Languages
Checks
Add
Your profile
Languages
Preferences
Notifications
Account
Profile
Licenses
Audit log
API access
Account
Username
testuser
Username may only contain letters, numbers or the following characters: @ . + - _
Full name
Weblate Test
E-mail
weblate@example.org
You can add another e-mail address below.
Your name and e-mail will appear as commit authorship.
Save
Current user identities
Identity
User ID
Action
Password
testuser
Change password
E-mail
weblate@example.org
Disconnect
Google
weblate@example.org
Disconnect
GitHub
123456
Disconnect
Bitbucket
weblate
Disconnect
Add new association
E-mail
Removal
Account removal deletes all your private data.
Remove my account
User data
You can download all your private data.
Download user data
Powered by Weblate 4.3
About Weblate
Legal
Contact
Documentation
Donate to Weblate

1.2.6 Журнал аудита

Журнал аудита отслеживает действия, произведённые с вашей учётной записи. Для каждого важного действия с вашей учётной записью в нём регистрируются IP-адрес и браузер. Критические действия вдобавок инициируют уведомление на основной адрес электронной почты.

См.также:

Работа за обратным прокси

1.3 Перевод с помощью Weblate

Благодарим вас за проявленный интерес к переводу с помощью Weblate. Проекты могут быть настроены либо на прямой перевод, либо на принятие предложений, внесённых пользователями без учётных записей.

В целом, существует два режима перевода:

- Проект принимает прямые переводы
- Проект принимает только предложения, которые автоматически подтверждаются по достижении определённого числа голосов

Для получения более подробной информации о процессе перевода, пожалуйста, ознакомьтесь с разделом *Рабочие процессы перевода*.

Варианты видимости проекта перевода:

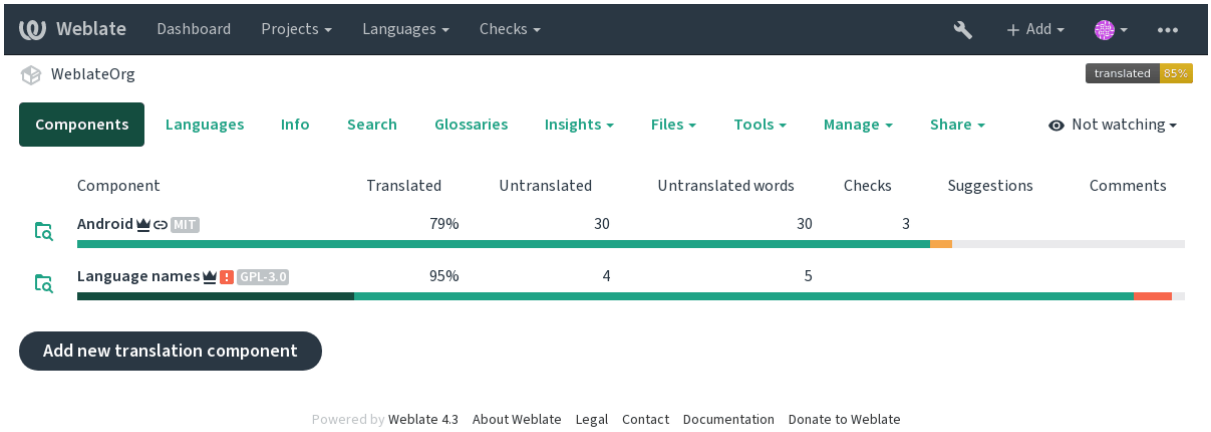
- Видимый публично, в который внести свой вклад может любой желающий
- Видимый только для определённой группы переводчиков

См.также:

Управление доступом, *Рабочие процессы перевода*

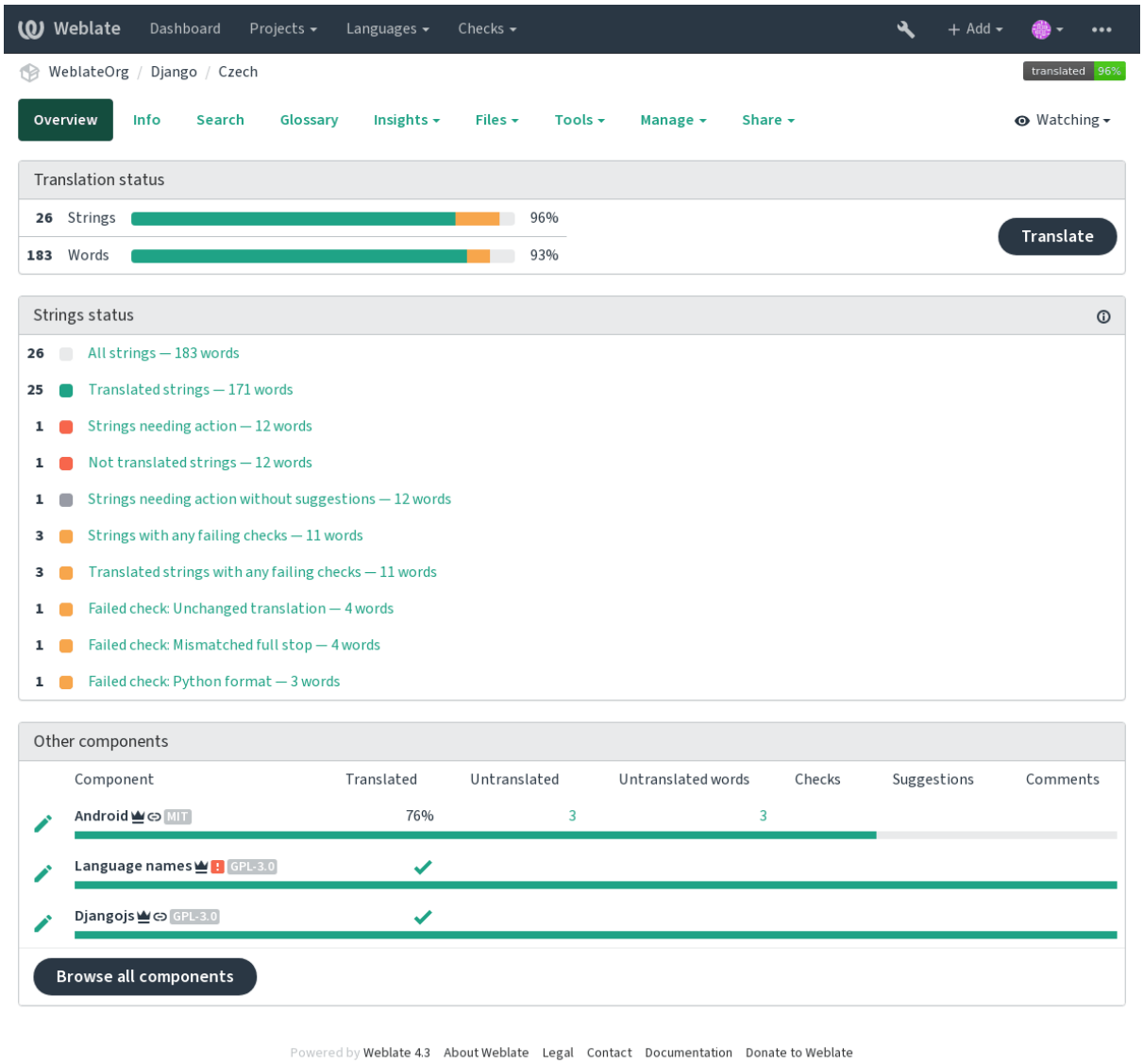
1.3.1 Проекты перевода

В проектах переводов содержатся относящиеся к ним компоненты, связанные с одним и тем же программным обеспечением, книгой или проектов.



1.3.2 Ссылки на перевод

В компоненте находится набор ссылок, которые ведут к реальному переводу. Далее перевод подразделяется на отдельные проверки, такие как *Непереведённые строки* или *Строки, требующие действий*. Если весь проект переведён без ошибок, у него всё равно доступна ссылка *Все строки*. В качестве альтернативы вы можете воспользоваться полем поиска для поиска определённой строки или термина.



1.3.3 Предложения

Примечание: Фактические разрешения могут отличаться в зависимости от конфигурации Weblate.

Анонимные пользователи могут только (если это разрешено) вносить предложения. При этом, в случае возникновения в переводе неопределённости, он всё равно будет доступен для зарегистрированных пользователей, что подтолкнёт другого переводчика его отрецензировать.

Предложения ежедневно сканируются с целью удаления дубликатов или предложений, совпадающих с текущим переводом.

1.3.4 Комментарии

Комментарии могут быть размещены ко двум объектам — к исходной строке или к переводу. Выберите тот, который соответствует теме, которую вы хотите обсудить. Комментарии к исходной строке хороши для того, чтобы дать обратную связь по изначальной строке, например, что её нужно перефразировать или что она вводит в заблуждение.

В комментариях вы можете использовать синтаксис Markdown и упоминать других пользователей, используя синтаксис `@упоминаний`.

1.3.5 Варианты

Варианты используются для группировки вариантов строки различной длины. Пользовательский интерфейс может использовать различные строки в зависимости от размеров экрана или окна.

См.также:

Варианты строк

1.3.6 Метки

Метки используются для категоризации строк в проекте. Их можно использовать для дальнейшей настройки рабочего процесса локализации, например, для определения категорий строк.

См.также:

Метки строк

1.3.7 Перевод

На странице перевода отображаются исходная строка и область редактирования для перевода. Если переводится строка со множественным числом, то показываются несколько исходных строк и областей редактирования, каждая из которых описывает свою форму множественного числа.

Все специальные пробельные символы подчёркиваются красным цветом и обозначаются серыми видимыми символами-заменителями. Более одного последовательного пробела также подчёркиваются красным цветом, чтобы предупредить переводчика о потенциальной проблеме форматирования.

На этой странице также могут быть показаны различные кусочки дополнительной информации, большая часть которой берётся из исходного кода проекта (например, контекст, комментарии разработчика или место, где используется сообщение). Если вы в своих предпочтениях выбрали дополнительные языки, над исходной строкой будет показан перевод на эти языки (смотрите раздел *Вспомогательные языки*).

Под переводом будут показаны все предложения, внесённые другими пользователями, которые вы, в свою очередь, можете принять, принять с изменениями или удалить.

Формы множественного числа

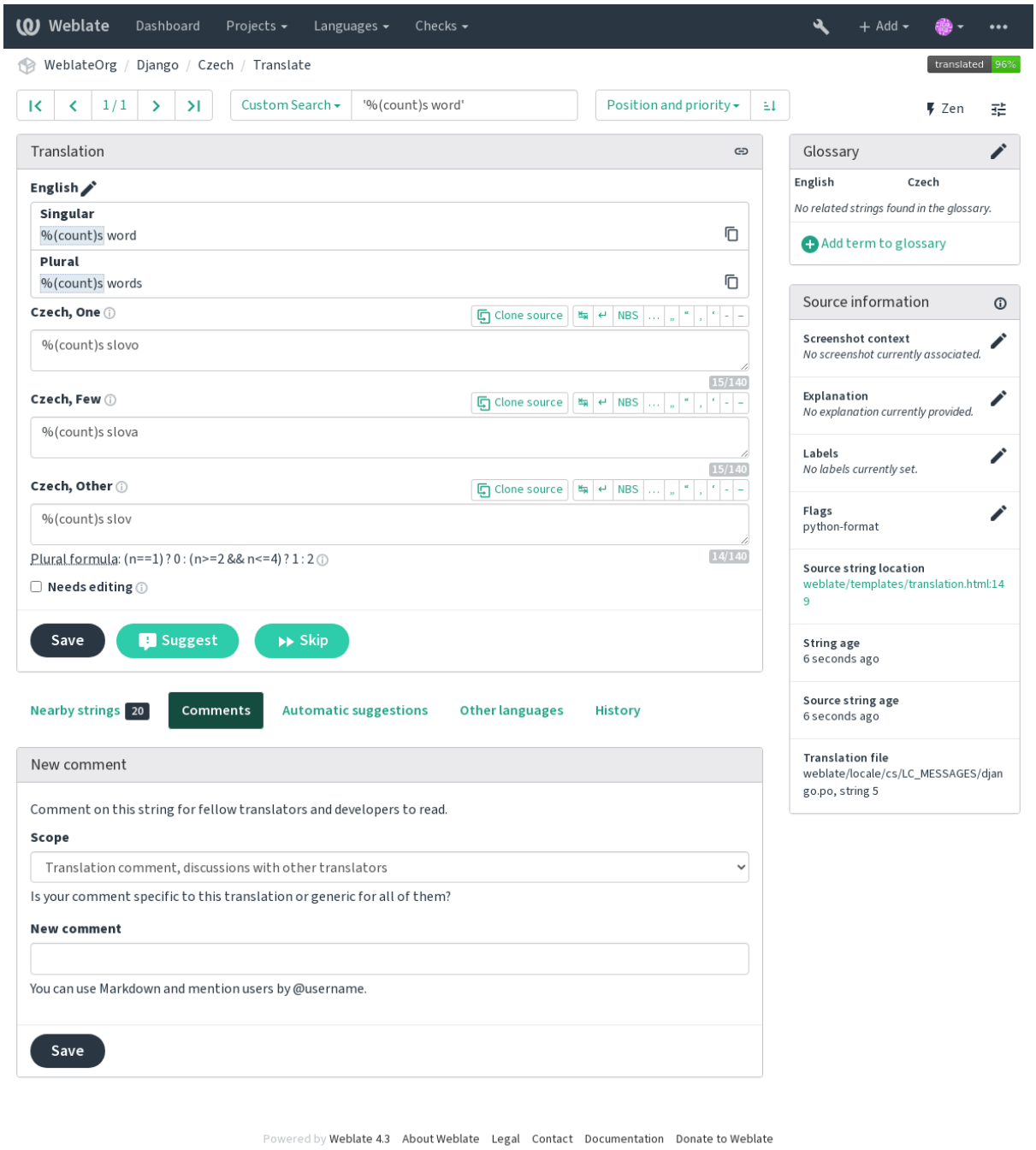
Слова, изменяющие свою форму в связи с их числовым обозначением, называются формами множественного числа. Каждый язык имеет своё определение множественного числа. В английском языке, например, существует одна форма множественного числа. В единичном определении, к примеру «автомобиля» — «car» — неявно подразумевается одна машина, а во множественном определении «автомобили» — «cars» — подразумеваются уже две или более машины, либо же понятие автомобилей как существительное. Такие языки, как, например, чешский, арабский или русский, имеют больше форм множественного числа, а их правила для этих форм отличаются от правил для английского языка.

Weblate полностью поддерживает каждую из этих форм, на каждом соответствующем языке, путём отдельного перевода каждой формы множественного числа. Количество полей и способ их использования в переводе

зависит от настроенной формулы множественного числа. Weblate показывает только основную информацию, но вы можете найти более подробное описание в языковых правилах для форм множественного числа от Консорциума Юникода.

См.также:

Формула множественного числа



Горячие клавиши

Изменено в версии 2.18: Сочетания клавиш в версии 2.18 были обновлены, чтобы снизить вероятность пересечения с умолчательными настройками браузера или системы.

Во время перевода можно использовать следующие сочетания клавиш:

Alt+Home Переход к первому переводу в текущем поиске.

Alt+End Переход к последнему переводу в текущем поиске.

Alt+PageUp Переход к предыдущему переводу в текущем поиске.

Alt+PageDown Переход к последующему переводу в текущем поиске.

Alt+Enter, Ctrl+Enter, или Cmd+Enter Сохранение текущего перевода.

Ctrl+Shift+Enter или Cmd+Shift+Enter Снятие с перевода пометки «на правку» и его сохранение.

Ctrl+E или Cmd+E Фокусировка на редакторе перевода.

Ctrl+U или Cmd+U Фокусировка на редакторе комментариев.

Ctrl+M или Cmd+M Показ вкладки автоматизированного перевода.

Ctrl+<ЧИСЛО> или Cmd+<ЧИСЛО> Копирование из исходной строки кусочка под соответствующим номером.

Ctrl+M <ЧИСЛО> или Cmd+M <ЧИСЛО> Копирование в текущий перевод машинного перевода с заданным номером.

Ctrl+I <ЧИСЛО> или Cmd+I <ЧИСЛО> Игнорирование одного элемента в списке проваленных проверок.

Ctrl+J или Cmd+J Показ вкладки *Соседние строки*.

Ctrl+S или Cmd+S Focuses search field.

Ctrl+O или Cmd+O Копирование исходной строки.

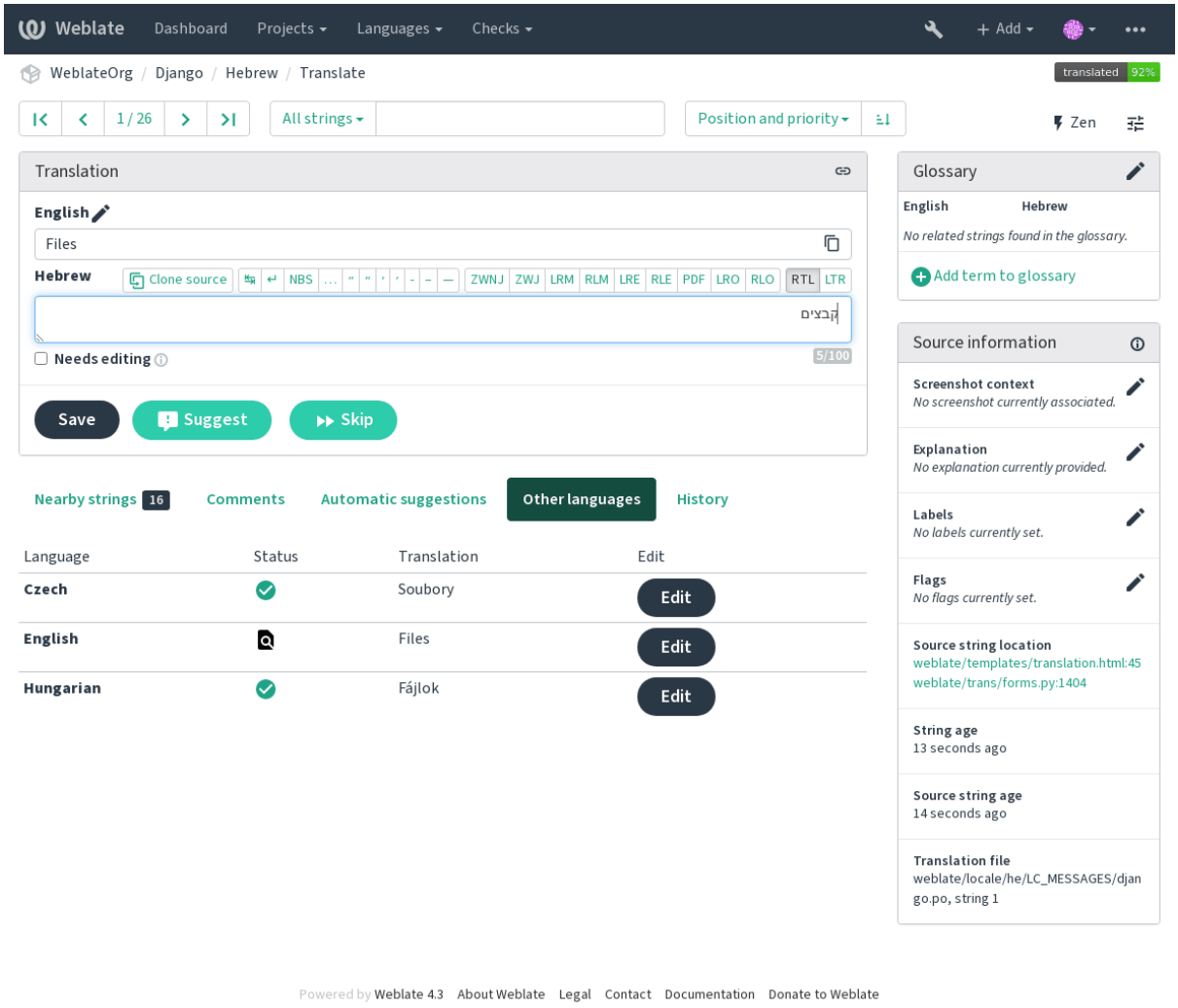
Ctrl+Y или Cmd+Y Переключение флага *На правку*.

Визуальная клавиатура

Чуть выше поля перевода показывается небольшая визуальная клавиатура. Она может быть полезна для ввода символов, которые обычно не встречаются в тексте или которые трудно набирать.

Показываемые символы разделены на три категории:

- Пользовательские символы, определённые в его *профиле*
- Предоставляемые Weblate символы для конкретного языка (например, кавычки или символы, специфичные для текста справа-налево)
- Символы, настроенные через параметр `SPECIAL_CHARS`



Контекст перевода

Это контекстное описание содержит информацию, относящуюся к текущей строке.

Атрибуты строки Такие вещи, как идентификатор сообщения, его контекст (`msgctxt`) или местоположение в исходном коде.

Снимки экрана В Weblate могут быть загружены снимки экрана, чтобы лучше информировать переводчиков о том, где и как используется строка, смотрите раздел *Визуальный контекст для строк*.

Соседние строки Отображает соседние сообщения из файла перевода. Обычно они также используются в аналогичном контексте и оказываются полезными для поддержания согласованности перевода.

Другие вхождения В случае, если сообщение появляется в нескольких местах (например, в нескольких компонентах), эта вкладка показывает их все, если в них найдены противоречия (смотрите раздел *Противоречия*). Вы можете выбрать, какую из строк использовать.

Память переводов Показывает похожие строки, переведённые в прошлом, смотрите раздел Memory Management.

Глоссарий Отображает термины из глоссария проекта, используемые в текущем сообщении.

Недавние изменения Список людей, которые недавно изменили это сообщение, используя Weblate.

Проект Информация о проекте, например, инструкции для переводчиков или информация о репозитории его системы контроля версий.

Также, если формат перевода поддерживает такую возможность, вы можете перейти по предоставленным ссылкам к соответствующему исходному коду, содержащему исходную строку.

История переводов

Каждое изменение по умолчанию (если это не отключено в настройках компонента) сохраняется в базе данных и может быть отменено. Также при необходимости через нижележащую систему управления версиями можно отменить всё, что угодно.

Длина переведённой строки

Weblate может ограничивать длину перевода несколькими способами, чтобы гарантировать, что переведённая строка не будет слишком длинной:

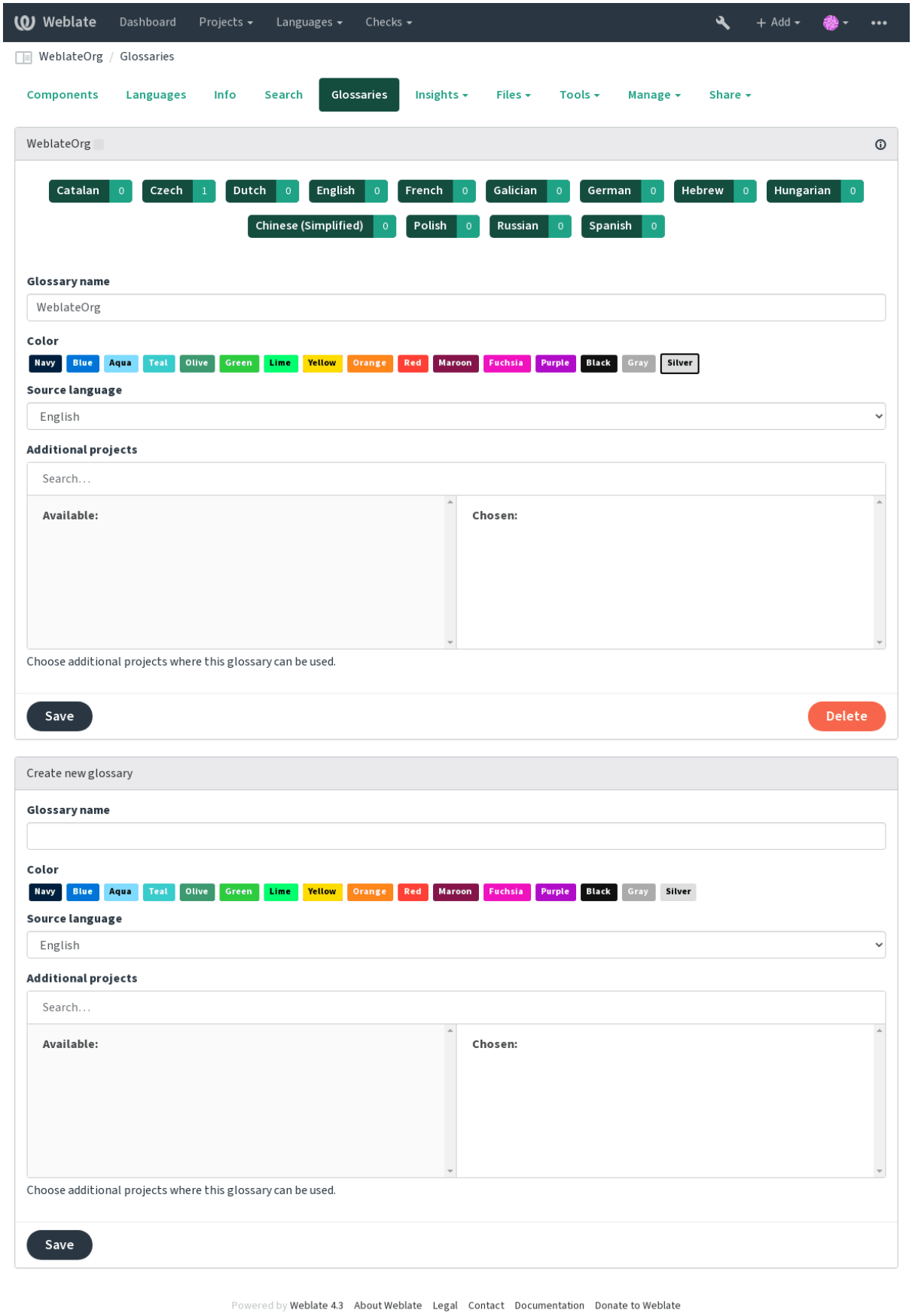
- Ограничение по умолчанию разрешает переводу быть в десять раз длиннее исходной строки. Его можно отключить с помощью параметра `LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH`. Если вы упёрлись в это ограничение, оно может быть вызвано тем, что одноязычный перевод был сконфигурирован как двуязычный, что заставило Weblate думать, что исходной строкой вместо настоящей исходной строки является ключ перевода. Дополнительную информацию смотрите в разделе *Двуязычные и одноязычные форматы*.
- Максимальная длина в символах, определяемая в файле перевода или флагом, смотрите раздел *Максимальная длина перевода*.
- Максимальный отрисовываемый размер текста в пикселях, определяемый флагами, смотрите раздел *Максимальный размер перевода*.

1.3.8 Глоссарий

К каждому проекту к любому языку может быть прикреплён глоссарий, хранящий сокращения для используемой терминологии. Таким образом облегчается поддержание согласованности перевода. Термины из переводимой в данный момент строки могут быть отображены в нижних вкладках.

Управление глоссариями

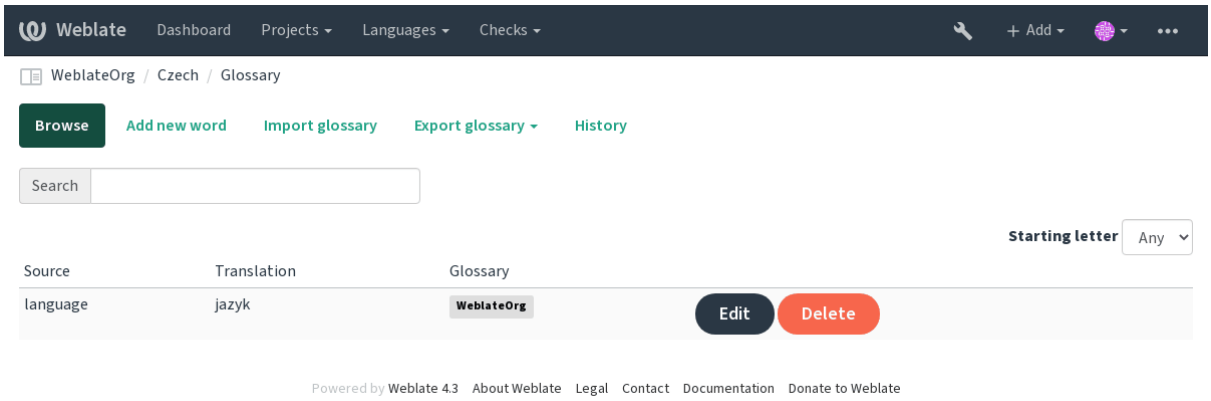
На вкладке *Глоссарии* любого проекта вы можете редактировать существующие глоссарии.



Пустой глоссарий для проекта создаётся автоматически при создании проекта. Глоссарии используются совместно всеми компонентами одного и того же проекта, кроме того, вы можете использовать их совместно с

другими проектами. Это можно сделать только для проектов, которые вы можете администрировать.

В этом списке можно выбрать, каким глоссарием управлять (показываются все языки, используемые в текущем проекте). Проследовав по одной из языковых ссылок, вы попадёте на страницу, на которой можно редактировать, импортировать или экспортировать выбранный глоссарий, либо просматривать его историю редактирования:



1.3.9 Автоматические предложения

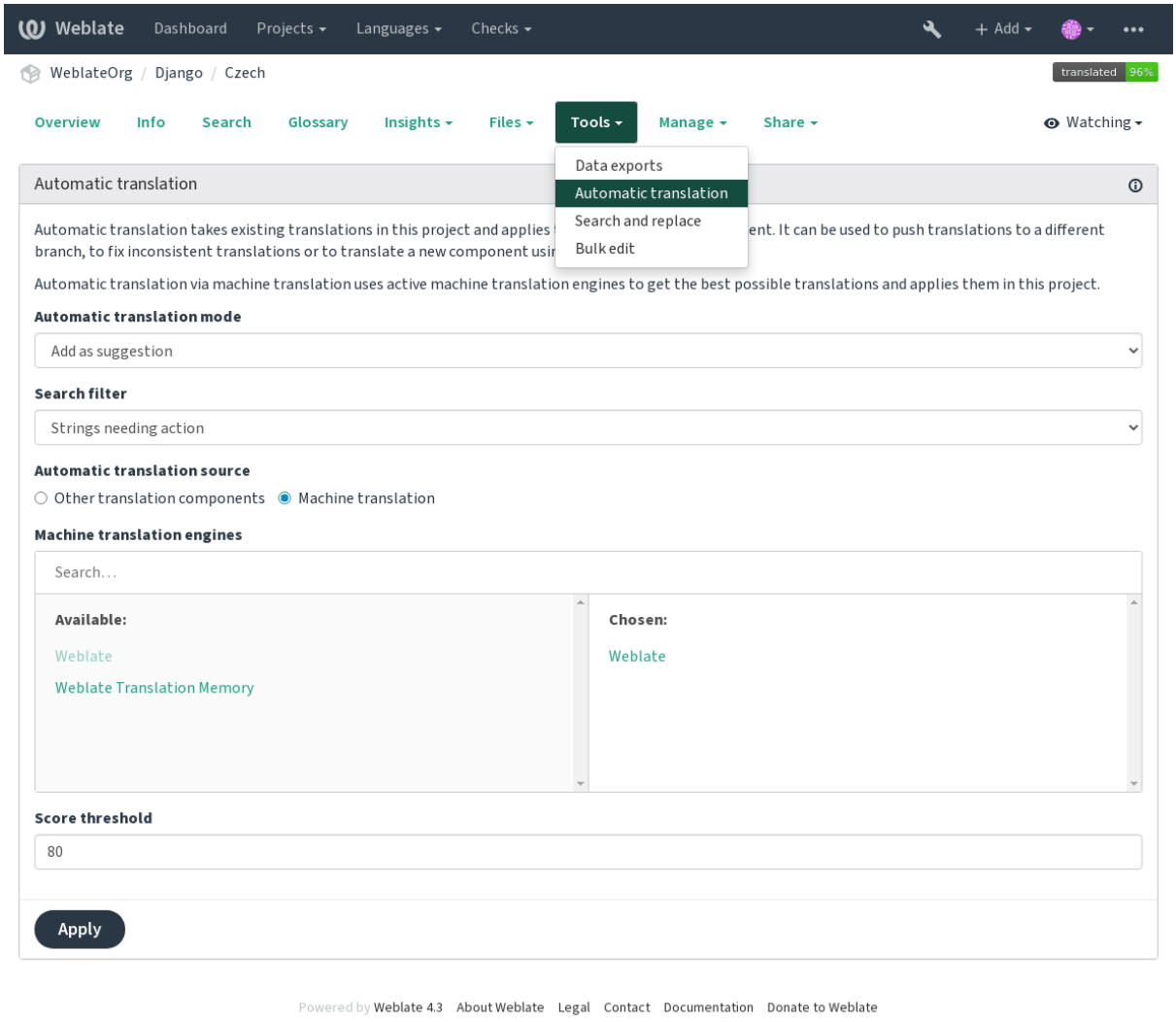
Based on configuration and your translated language, Weblate provides you suggestions from several machine translation tools and *Память переводов*. All machine translations are available in a single tab of each translation page.

См.также:

Список поддерживаемых инструментов можно найти в разделе *Машинный перевод*.

1.3.10 Автоматический перевод

Вы можете использовать автоматический перевод для начального перевода на основе внешних источников. Этот инструмент называется *Автоматический перевод* и доступен в меню *Инструменты* после выбора компонента и языка:



Возможны два режима работы этого инструмента:

- Использование в качестве источника для переводов других компонентов Weblate.
- Использование выбранных сервисов машинного перевода с переводами выше определённого порога качества.

Вы также можете выбрать, какие строки должны быть переведены автоматически.

> **Предупреждение:** Имейте в виду, что если использовать широкие фильтры, вроде фильтра *Все строки*, это приведёт к перезаписи существующих переводов.

Инструмент полезен в ряде ситуаций, например, при объединении переводов различных компонентов (к примеру, веб-сайта и приложения) или при начальном переводе нового компонента с использованием существующих переводов (памяти переводов).

См.также:

Поддержание единого перевода в разных компонентах

1.3.11 Ограничение частоты запросов

Чтобы избежать злоупотребления интерфейсом, существует ограничение на частоту запросов, применяемое к некоторым операциям, таким как поиск, отправка контактной формы или перевод. В случае, если вы натолкнулись на это ограничение, вы будете на определённый период заблокированы, по истечению которого вы сможете заново выполнить операцию.

Ограничения по умолчанию описаны в руководстве по администрированию в разделе *Ограничение частоты запросов*, но они могут быть изменены в конфигурации.

1.3.12 Массовая правка

Массовая правка позволяет выполнять операцию одновременно с несколькими строками. Вы определяете критерии поиска строк и операцию для выполнения, и все найденные строки обновляются. Поддерживаются следующие операции:

- Изменение состояния строки (например, для одобрения всех строк, находящихся на рецензировании)
- Корректировка флагов перевода (смотрите раздел *Настройка поведения*)
- Корректировка меток строк (смотрите раздел *Метки строк*)

Подсказка: Этот инструмент называется *Массовая правка* и доступен он из меню *Инструменты* проекта, компонента или перевода.

См.также:

Надстройка «Массовая правка»

1.4 Скачивание и загрузка переводов

Вы можете экспортировать файлы из перевода, внести в них изменения и снова их импортировать. Это позволяет поработать над переводом в автономном режиме, а затем влить изменения обратно в существующий перевод. Это работает, даже если существующий перевод за это время был изменён.

Примечание: Доступные варианты могут быть ограничены *списками доступа*.

1.4.1 Скачивание переводов

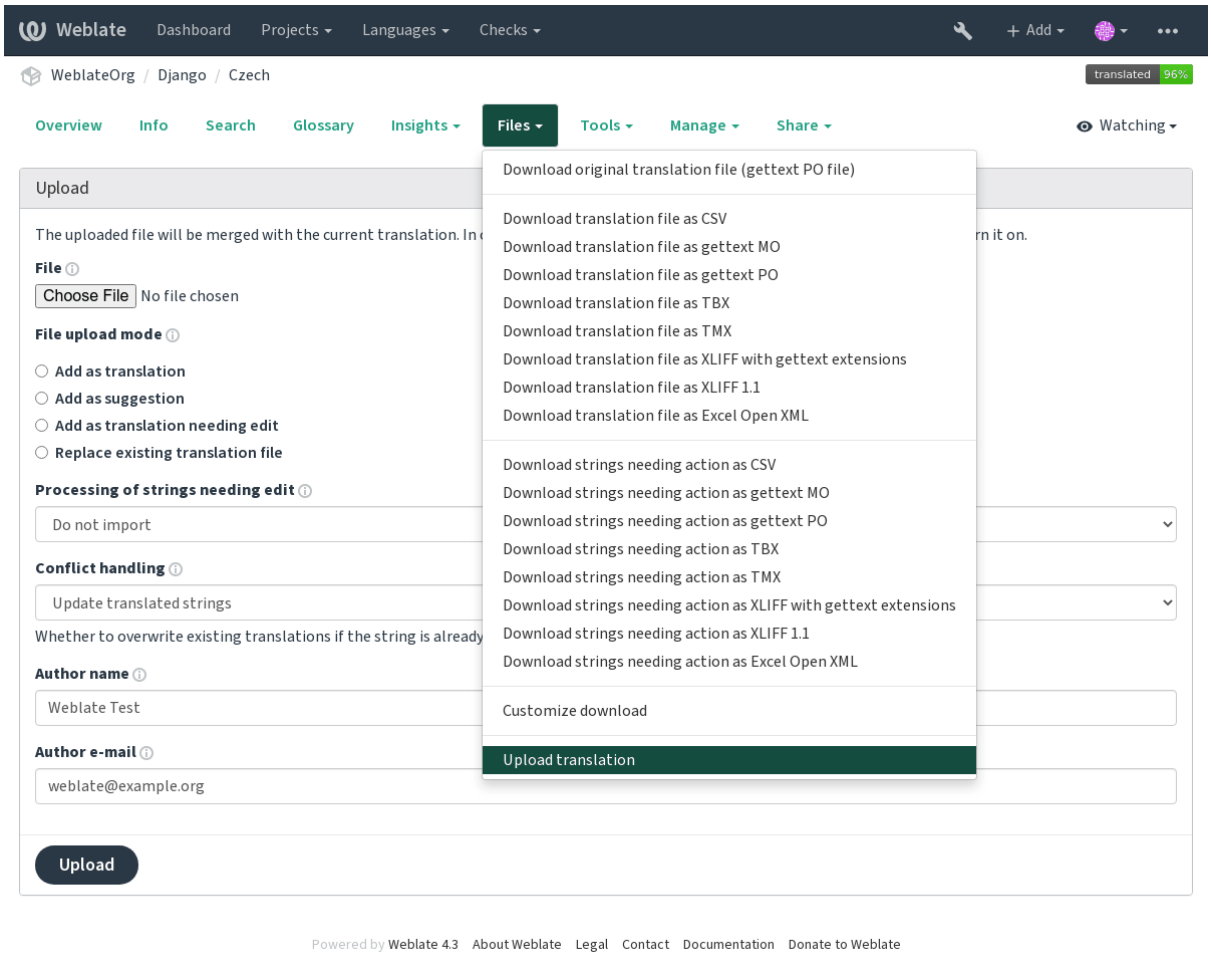
Из панели управления проекта или компонента переводимые файлы можно скачать через пункт *Скачать оригинальный файл перевода* меню *Файлы*, который создаёт копию оригинального файла в том виде, в котором он хранится в вышестоящей системе контроля версий.

Также вы можете скачать перевод, преобразованный в один из широко распространённых форматов локализации. Преобразованные файлы будут обогащены данными, предоставленными Weblate, такими как дополнительный контекст, комментарии или флаги.

В меню *Файлы* доступно несколько форматов файлов, в том числе скомпилированные файлы для использования в выбранном вами приложении (например, файлы `.mo` для GNU Gettext).

1.4.2 Загрузка переводов

После того, как вы внесли в перевод свои изменения, воспользуйтесь пунктом *Загрузить перевод* меню *Файлы*.



Поддерживаемые форматы файлов

Можно загрузить любой файл в поддерживаемом формате, но всё же рекомендуется использовать тот же самый формат файла, который используется для перевода, в противном случае некоторые возможности могут быть перенесены неправильно.

См.также:

Поддерживаемые форматы файлов

Загруженный файл объединяется с существующим для обновления перевода, по умолчанию перезаписывая существующие записи (это можно отключить или включить в диалоге загрузки).

Способы импортирования

При загрузке файлов перевода вы можете выбирать из следующих вариантов:

Добавить как перевод (`translate`) Импортированные переводы добавляются как переводы. Это наиболее распространённое использование и поведение по умолчанию.

Добавить как предложение (`suggest`) Импортированные переводы добавляются как предложения, делайте это тогда, когда хотите, чтобы загруженные строки были отрецензированы.

Добавить перевод как «На правку» (`fuzzy`) Импортированные переводы добавляются как переводы, требующие правки. Это может быть полезно, когда вы хотите, чтобы переводы не только использовались, но также были и отрецензированы.

Заменить существующий файл перевода (`replace`) Существующий файл заменяется новым содержимым. Это может привести к потере существующих переводов, используйте этот вариант с осторожностью.

Обновить исходные строки (`source`) Обновляет исходные строки в двуязычном файле перевода. Это поведение похоже на то, что делает надстройка *Обновлять PO-файлы до соответствия POT (msgmerge)*.

См.также:

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

Разрешение конфликтов

Определяет, что делать с загружаемыми строками, которые уже переведены.

Строки, отмеченные как «На правку»

Также существует параметр, контролирующий то, что делать со строками в импортируемом файле, нуждающимися в редактировании . Такие строки можно обработать одним из трёх следующих способов: «Не импортировать», «Импортировать строки, как отмеченные „На правку“» или «Импортировать как переведённое».

Переопределение авторства

С разрешениями администратора вы также можете указать авторство загружаемого файла. Это может быть полезно в случае, если вы получили файл другим способом и хотите объединить переводы из него с существующими с правильным указанием фактического автора.

1.5 Проверки и исправления

Проверки качества помогают выявить распространённые ошибки переводчиков, что гарантирует хорошее состояние перевода. В случае ложно-положительных срабатываний проверки могут быть проигнорированы.

После отправки перевода с неудачной проверкой, она сразу же показывается пользователю:

Weblate Dashboard Projects Languages Checks + Add

WeblateOrg / Django / Czech / Translate translated 96%

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

1 / 1 Custom Search '%(count)s word' Position Zen

Translation

English

Singular
%(count)s word

Plural
%(count)s words

Czech, One Clone source NBS

0/140

Czech, Few Clone source NBS
několik slov

12/140

Czech, Other Clone source NBS
%(count)s slov

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

14/140

Needs editing

Save Suggest Skip

Nearby strings 20 Comments Automatic suggestions Other languages History

New comment

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

Scope

Translation comment, discussions with other translators

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

New comment

You can use Markdown and mention users by @username.

Save

Things to check

Python format 1
Following format strings are missing: %(count)s
Dismiss
Dismiss for all languages

Missing plurals 2
Some plural forms are not translated
Dismiss
Dismiss for all languages

Glossary

English Czech

No related strings found in the glossary.

Add term to glossary

Source information

Screenshot context
No screenshot currently associated.

Explanation
No explanation currently provided.

Labels
No labels currently set.

Flags
python-format

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

String age
10 seconds ago

Source string age
11 seconds ago

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

Powered by Weblate 4.3 About Weblate Legal Contact Documentation Donate to Weblate

1.5.1 Автоматические исправления

В дополнение к *проверкам качества*, Weblate может автоматически исправлять некоторые распространённые ошибки в переведённых строках. Используйте эту возможность с осторожностью, чтобы не допустить добавления новых ошибок.

См.также:

`AUTOFIX_LIST`

1.5.2 Проверки качества

Weblate использует широкий спектр проверок качества строк. В следующем разделе они все описываются более подробно. Также существуют проверки, специфичные для конкретных языков. Если какая-то проверка срабатывает ошибочно, пожалуйста, отправьте сообщение об ошибке.

См.также:

`CHECK_LIST`, *Настройка поведения*

1.5.3 Проверки перевода

Выполняется при каждом изменении перевода, помогая переводчикам поддерживать высокое качество перевода.

BBcode-разметка

BBcode в переводе не соответствует источнику

BBCode представляет собой простую разметку, полезную для, например, выделения важных частей сообщения жирным шрифтом или курсивом.

Эта проверка гарантирует, что они также будут присутствовать в переводе.

Примечание: На данный момент метод обнаружения тегов BBcode достаточно прост, поэтому данная проверка может привести к ложным срабатываниям.

Повторяющиеся слова

Одно и то же слово повторяется в тексте дважды подряд:

Добавлено в версии 4.1.

Проверяет, что в переводе не встречаются последовательные дубликаты слова. Обычно это указывает на ошибку в переводе.

Подсказка: Эта проверка включает в себя правила, специфичные для конкретных языков, чтобы избежать ложных срабатываний. В случае ложного срабатывания этой проверки в вашем случае, сообщите нам об этом. Смотрите раздел *Сообщение о проблемах в Weblate'е*.

Двойной пробел

Перевод содержит двойной пробел

Проверяет наличие в переводе двойного пробела, чтобы избежать ложных срабатываний на других проверках, связанных с пробелами.

Проверка возвращает ложь, если в исходной строке найден двойной пробел, что означает, что двойной пробел был внесён в неё преднамеренно.

Форматированные строки

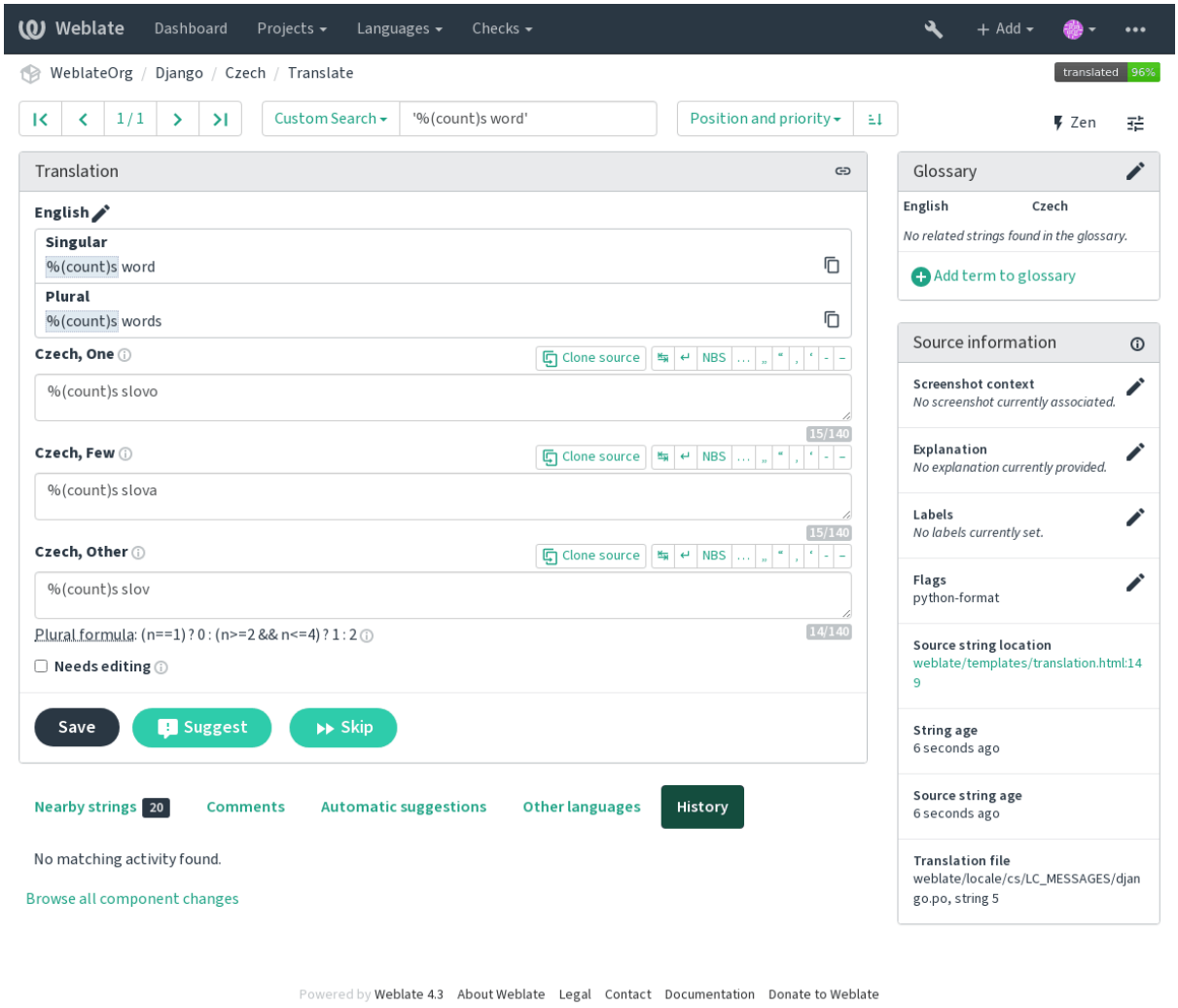
Проверяет, что форматирование в исходной строке и в переводе одинаково. Упущение в переводе строк формата обычно вызывает серьёзные проблемы, поэтому форматирование в переводе обычно должно соответствовать форматированию исходной строки.

Weblate поддерживает проверку строк формата на нескольких языках. Проверка не включается автоматически, только если строка соответствующим образом помечена (например, флагом *c-format* для C-формата). Gettext добавляет эту разметку автоматически, но для других форматов файлов или если ваши PO-файлы не сгенерированы программой **`xgettext`** вам, вероятно, придётся добавить её вручную.

Её можно добавлять как индивидуально в каждую единицу перевода (смотрите раздел *Дополнительная информация об исходных строках*), так и в *конфигураци компонента*. Определение формата для компонента целиком проще, но может привести к ложным срабатываниям, если строка не интерпретируется как строка формата, но при этом внутри неё используется синтаксис строки формата.

Подсказка: В случае, если конкретная проверка формата в Weblate отсутствует, вы можете использовать общую проверку *Символы-заполнители*.

Помимо проверки, эти флаги также включают подсветку строк формата, чтобы их было легко вставлять в перевод:



Интерполяционная строка AngularJS

Интерполяционные строки AngularJS не сооответствуют источнику

Именованная строка формата	`Your balance is {{amount}} {{ currency }}`
Флаг для включения	*angularjs-format*

См.также:

AngularJS: API: $interpolate

C-формат

Строка в формате C не соответствует источнику

Простая строка формата	`There are %d apples`
Позиционная строка формата	`Your balance is %1$d %2$s`
Флаг для включения	*c-format*

См.также:

Строки формата C, Синтаксис строки форматирования функции printf C

C#-формат

Строка в формате C# не соответствует источнику

Позиционная строка формата	`There are {0} apples`
Флаг для включения	*c-sharp-format*

См.также:

Строки формата C#

Шаблонные литералы ECMAScript

Шаблонные литералы ECMAScript не соответствует источнику

Интерполяция	`There are ${number} apples`
Флаг для включения	*es-format*

См.также:

Шаблонные литералы

Интерполяция i18next

Интерполяция i18next не соответствует источнику

Добавлено в версии 4.0.

Интерполяция	`There are {{number}} apples`
Вложение	`There are $t(number) apples`
Флаг для включения	*i18next-interpolation*

См.также:

Интерполяция i18next

Java-формат

Строка в формате Java не соответствует источнику

Простая строка формата	`There are %d apples`
Позиционная строка формата	`Your balance is %1$d %2$s`
Флаг для включения	*java-format*

См.также:

Строки в формате Java

Java MessageFormat

Строка в формате Java MessageFormat не соответствует источнику

Позиционная строка формата	`There are {0} apples`
Флаг для включения	Флаг *java-messageformat* включает эту проверку безусловно
	Флаг *auto-java-messageformat* включает проверку только если в исходной строке содержится строка формата

См.также:

Java MessageFormat

JavaScript-формат

Строка в формате JavaScript не соответствует источнику

Простая строка формата	`There are %d apples`
Флаг для включения	*javascript-format*

См.также:

Строки форматирования JavaScript

Символы-заполнители в процентной нотации

Символы-заполнители в процентной нотации не соответствуют источнику

Добавлено в версии 4.0.

Простая строка формата	`There are %number% apples`
Флаг для включения	*percent-placeholders*

Perl-формат

Строка в формате Perl'a не соответствует источнику

Простая строка формата	`There are %d apples`
Позиционная строка формата	`Your balance is %1$d %2$s`
Флаг для включения	*perl-format*

См.также:

Функция sprintf Perl'a, Строки формата Perl'a

PHP- формат

Строка в формате PHP не соответствует источнику

Простая строка формата	`There are %d apples`
Позиционная строка формата	`Your balance is %1$d %2$s`
Флаг для включения	*php-format*

См.также:

Документация по функции sprintf PHP, Строки формата PHP

Python-формат: фигурные скобки

Форматирование фигурных скобок в Python не соответствует источнику

Простая строка формата	`There are {} apples`
Именованная строка формата	`Your balance is {amount} {currency}`
Флаг для включения	*python-brace-format*

См.также:

Формат фигурных скобок в Python'е, строки формата Python

Python-формат

Строка в формате Python'а не соответствует источнику

Простая строка формата	`There are %d apples`
Именованная строка формата	`Your balance is %(amount) %(currency)`
Флаг для включения	*python-format*

См.также:

Форматирование строк в Python'е, Python'ьи строки формата

Qt-формат

Строка в формате Qt не соответствует источнику

Позиционная строка формата	`There are %1 apples`
Флаг для включения	*qt-format*

См.также:

Qt QString::arg()

Формат множественного числа Qt

Строка в формате множественного числа Qt не соответствует источнику

Строка в формате множественного числа	`There are %Ln apple(s)`
Флаг для включения	*qt-plural-format*

См.также:

Руководство Qt по интернационализации

Ruby-формат

Строка формата Ruby не соответствует источнику

Простая строка формата	`There are %d apples`
Позиционная строка формата	`Your balance is %1$f %2$s`
Именованная строка формата	`Your balance is %+.2<amount>f %<currency>s`
Именованная шаблонная строка	`Your balance is %{amount} %{currency}`
Флаг для включения	*ruby-format*

См.также:

Ruby Kernel#sprintf

Форматирование Vue I18n

Форматирование Vue I18n не соответствует источнику

Именованная строка формата	`There are {count} apples`
Форматирование локализации Rails	`There are %{count} apples`
Связанные сообщения локали	`@:message.dio @:message.the_world!`
Флаг для включения	*vue-format*

См.также:

Vue I18n Formatting, Vue I18n Linked locale messages

Уже переведено

Эта строка уже переводилась

Означает, что строка уже переведена. Это может произойти, если переводы откатывались в системе контроля версий или иным образом терялись.

Противоречия

Эта строка имеет более одного перевода в этом проекте, либо же в некоторых компонентах она не переведена.

Weblate проверяет переводы одинаковых строк по всем переводам в проекте, чтобы помочь вам поддерживать переводы согласованными.

Проверка срабатывает, когда в проекте обнаруживаются различные переводы одной и той же строки. Также это может привести к несогласованности в отображаемых проверках. Другие переводы этой строки можно найти на вкладке *Другие вхождения*.

Примечание: Также эта проверка срабатывает в том случае, если строка в одном компоненте переведена, а в другом — нет. Она может использоваться, чтобы по-быстрому вручную обработать строки, не переведённые в некоторых компонентах; для этого надо просто на вкладке *Другие вхождения* нажать на кнопку *Использовать этот перевод*, отображаемую напротив каждой строки.

Для автоматизации перевода новых добавленных строк, которые уже переведены в другом компоненте, вы можете использовать надстройку *Автоматический перевод*.

См.также:

Поддержание единого перевода в разных компонентах

Используется кашида

Декоративные буквы кашиды не должны использоваться

Добавлено в версии 3.5.

Декоративные буквы кашида не должны использоваться в переводе. Также они известны как татвиль.

См.также:

Кашида в Википедии

Ссылки Markdown

Ссылки Markdown-разметки не соответствуют источнику

Добавлено в версии 3.5.

Ссылки Markdown-разметки не соответствуют источнику.

См.также:

Ссылки Markdown

Сноски Markdown

Ссылки на сноски Markdown не соответствуют источнику

Добавлено в версии 3.5.

Ссылки на сноски Markdown не соответствуют источнику.

См.также:

Ссылки Markdown

Синтаксис Markdown

Синтаксис Markdown-разметки не соответствует источнику

Добавлено в версии 3.5.

Синтаксис Markdown-разметки не соответствует источнику

См.также:

Строчные элементы Markdown

Максимальная длина перевода

Перевод не должен превышать заданную длину

Проверяет, что переводы имеют приемлемую длину, чтобы укладываться в доступное пространство. Эта проверка проверяет только длину символов перевода.

В отличие от других проверок, флаг для этой проверки должен быть установлен в виде пары `ключ:значение`, например, `max-length:100`.

Подсказка: Эта проверка смотрит на количество символов, что может быть не самой лучшей метрикой при использовании для отображения текста пропорциональных шрифтов. Проверка *Максимальный размер перевода* проверяет фактическую отрисовку текста.

Также для расширения позиций-заполнителей перед проверкой строки может быть полезен флаг `replacements:`.

Максимальный размер перевода

Отрисованный текст перевода не должен превышать заданный размер

Добавлено в версии 3.7.

Отрисованный текст перевода не должен превышать заданный размер. Эта проверка отрисовывает текст с переносом по строкам и проверяет, вписывается ли он в заданные границы.

Для этой проверки требуется один или два параметра — максимальная ширина и максимальное количество строк. В случае, если количество строк не указано, считается, что текст должен уместиться в одну строку.

Также директивами `font-*` вы можете настроить используемый шрифт (смотрите раздел *Настройка поведения*), например, следующие флаги перевода говорят, что текст, отрисовываемый шрифтом ubuntu размера 22, должен вписаться в две строки и 500 пикселей:

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

Подсказка: Возможно, вы захотите установить директивы `font-*` в *настройках компонента*, чтобы для всех строк компонента был настроен один и тот же шрифт. В случае, если вам нужно будет настроить их для отдельной строки, вы сможете переопределить их значения для этой строки.

Также для расширения позиций-заполнителей перед проверкой строки может быть полезен флаг `replacements:`.

См.также:

Управление шрифтами, *Настройка поведения*, *Максимальная длина перевода*

Несовпадающие \n

Количество \n в переводе не соответствует источнику

Экранированные символы новой строки обычно важны при форматировании вывода программы. Проверка срабатывает, если количество символов `\n` в переводе не совпадает с их количеством в исходной строке.

Несовпадающие двоеточия

Источник и перевод не заканчиваются оба двоеточием

Проверяет, что двоеточия в исходной строке и в переводе одинаковы. Также проверяется наличие двоеточий для тех языков, в которых они не используются (китайский или японский).

См.также:

Двоеточие в Википедии

Несовпадающие многоточия

Источник и перевод не заканчиваются оба многоточием

Проверяет, что конечные многоточия в исходной строке и в переводе одинаковы. Эта проверка проверяет только настоящие многоточия (символы …), но не три подряд идущих точки (`...`).

Как правило, на печати многоточие отрисовывается лучше, чем три подряд стоящих точки и лучше звучит при преобразовании текста в речь.

См.также:

Многоточие в Википедии

Несовпадающие восклицательные знаки

Источник и перевод не заканчиваются оба восклицательным знаком

Проверяет, что восклицательные знаки в исходной строке и в переводе одинаковы. Также проверяется наличие восклицательных знаков для тех языков, в которых они не используются (китайский, японский, корейский, армянский, лимбу, бирманский или нко).

См.также:

Восклицательный знак в Википедии

Несовпадающие точки

Источник и перевод не заканчиваются оба точкой

Проверяет, что точки в исходной строке и в переводе одинаковы. Также проверяется наличие точек для тех языков, в которых они не используются (китайский, японский, деванагари или урду).

См.также:

Точка в Википедии

Несовпадающие вопросительные знаки

Источник и перевод не заканчиваются оба вопросительным знаком

Проверяет, что вопросительные знаки в исходной строке и в переводе одинаковы. Также проверяется наличие вопросительных знаков для тех языков, в которых они не используются (армянский, арабский, китайский, корейский, японский, эфиопский, ваи или коптский).

См.также:

Вопросительный знак в Википедии

Несовпадающие точки с запятой

Источник и перевод не заканчиваются оба точкой с запятой

Проверяет, что точки с запятой в конце предложений в исходной строке и в переводе одинаковы. Эта проверка может быть полезна, например, для сохранения форматирования в записях файлов `.desktop`.

См.также:

Точка с запятой в Википедии

Несовпадающие переводы строк

Количество символов новой строки в переводе не соответствует источнику

Символы новой строки обычно важны при форматировании вывода программы. Проверка срабатывает, если количество литералов``n`` в переводе не совпадает с их количеством в исходной строке.

Нет форм множественного числа

Некоторые формы множественного числа не переведены

Проверяет, что были переведены все формы множественного числа исходной строки. Сведения о том, как используется каждая форма множественного числа, можно найти в определении строки.

Отсутствие перевода в форме множественного числа в некоторых случаях приводит к тому, что при использовании формы множественного числа ничего не будет отображаться.

Символы-заполнители

В переводе пропущены некоторые символы-заполнители:

Добавлено в версии 3.9.

Изменено в версии 4.3: В качестве заполнителя вы можете использовать регулярное выражение.

В переводе пропущены некоторые символы-заполнители. Эти символы либо извлекаются из файла перевода, либо определяются вручную с помощью флага ``placeholders`; если их несколько, они разделяются двоеточием, а строки с пробелами могут быть заключены в кавычки:

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

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

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

См.также:

Настройка поведения

Пробелы и пунктуация

Отсутствует неразрывный пробел перед знаком двойной пунктуации

Добавлено в версии 3.9.

Проверяет наличие неразрывного пробела перед знаком двойной пунктуации (восклицательным знаком, вопросительным знаком, точкой с запятой или двоеточием). Это правило используется только в некоторых избранных языках, вроде французского или бретонского, где по типографским правилам обязателен пробел перед знаком двойной пунктуации.

См.также:

Пробелы во французском и английском языках в (англоязычной) Википедии

Регулярное выражение

Перевод не соответствует регулярному выражению:

Добавлено в версии 3.9.

Перевод не соответствует регулярному выражению. Это выражение либо извлекается из файла перевода, либо определяется вручную с помощью флага `regex`:

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

Одинаковые формы множественного числа

Некоторые формы множественного числа переведены одинаково

Проверка срабатывает, если некоторые формы множественного числа в переводе переведены одинаково. В большинстве языков они должны быть разными.

Пустая строка в начале

Источник и перевод не начинаются оба с новой строки

Обычно символы новой строки в исходных текстах появляются по уважительной причине, их пропуск или добавление могут привести к проблемам с форматированием при использовании переведённого текста.

См.также:

Пустая строка в конце

Пробелы в начале

Источник и перевод не начинаются оба одинаковым числом пробелов

Пробелы в начале строки обычно используются для набивания отступов в интерфейсе, поэтому их важно сохранить.

Пустая строка в конце

Источник и перевод не заканчиваются оба новой строкой

Обычно символы новой строки в исходных текстах появляются по уважительной причине, их пропуск или добавление могут привести к проблемам с форматированием при использовании переведённого текста.

См.также:

Пустая строка в начале

Пробелы в конце

Источник и перевод не заканчиваются оба пробелом

Проверяет, что количество конечных пробелов в исходной строке и в переводе одинаково.

Конечные пробелы обычно используется для добавления отступа до соседних элементов, поэтому их удаление может поломать макет.

Перевод без изменения

Источник и перевод тождественны

Возникает, если исходная строка и соответствующая строка перевода идентичны, по крайней мере в одной из форм множественного числа. Некоторые строки, обычно встречающиеся во всех языках, игнорируются, а различная разметка удаляется. Это уменьшает количество ложных срабатываний.

Эта проверка может помочь найти ошибочно не переведённые строки.

Поведение этой проверки по умолчанию заключается в исключении из проверки слов из встроенного чёрного списка. Это такие слова, которые зачастую не переводятся. Это полезно для того, чтобы избежать ложных срабатываний на коротких строках, которые состоят только из одного слова, одинакового в нескольких языках. Этот чёрный список можно отключить, добавив в строку или в компонент флаг `strict-same`.

См.также:

Конфигурация компонента, *Настройка поведения*

Небезопасный HTML-код

Перевод использует небезопасную HTML-разметку

Добавлено в версии 3.9.

Перевод использует небезопасную HTML-разметку. Эта проверка должна быть включена с помощью флага `safe-html` (смотрите раздел *Настройка поведения*). Также имеется соответствующий автоматический исправляльщик, который может автоматически почистить разметку.

См.также:

Проверка HTML осуществляется с помощью разработанной Mozilla библиотеки **`Bleach <https://bleach.readthedocs.io/`_**.

URL-адреса

Перевод не содержит URL

Добавлено в версии 3.5.

Перевод не содержит URL-адрес. Эта проверка срабатывает только в том случае, если единица перевода помечена как содержащая URL-адрес. В этом случае перевод должен быть корректным URL-адресом.

XML-разметка

XML-теги в переводе не соответствуют источнику

Обычно это означает, что полученный результат будет выглядеть по-другому. В большинстве случаев такое изменение из-за перевода не является желаемым результатом, но иногда именно это и надо.

Проверяет, что XML-теги в исходной строке и в переводе одинаковы.

Синтаксис XML

Перевод не является корректным XML

Добавлено в версии 2.8.

Разметка XML некорректна.

Пробел нулевой ширины

Перевод содержит лишние символы пробела нулевой ширины

Символы пробела нулевой ширины (<U+200B>) используются для разбиения сообщений по словам (переноса по словам).

Поскольку обычно они вставляются по ошибке, эта проверка срабатывает на их присутствие в переводе. В некоторых программах при использовании этого символа могут возникнуть проблемы.

См.также:

Пробел нулевой ширины в (англоязычной) Википедии

1.5.4 Проверки для исходных строк

Проверки для исходных строк могут помочь разработчикам улучшить качество исходных строк.

Многоточие

В этой строке используются три точки (…) вместо символа многоточия (…)

Эта проверка срабатывает, когда в строке используются три подряд стоящих точки (`...`), когда вместо них должен использоваться символ многоточия (…).

Использование юникодного символа в большинстве случаев является лучшим подходом; он лучше выглядит при отрисовке текста и может лучше звучать при преобразовании текста в речь.

См.также:

Многоточие в Википедии

Давно не переводилась

Строка давно не переводилась

Добавлено в версии 4.1.

Если строка долгое время не переводилась, это может указывать на проблему в исходной строке, затрудняющую её перевод.

Множество неудачных проверок

Переводы на несколько языков имеют неудачные проверки

Во многих переводах этой строки имеются неудачные проверки качества. Обычно это свидетельствует о том, что нужно что-то сделать для улучшения исходной строки.

Часто неудачная проверка может быть вызвана пропущенной точкой в конце предложения или подобными мелкими проблемами, которые переводчики склонны исправлять в переводе, в то время как лучше было бы их исправить в исходной строке.

Несколько переменных без названий

Строка содержит несколько неименованных переменных, что не позволяет переводчикам их переупорядочивать

Добавлено в версии 4.1.

Строка содержит несколько неименованных переменных, что не позволяет переводчикам их переупорядочивать.

Рассмотрите возможность использования именованных переменных вместо неименованных, чтобы позволить переводчикам менять их порядок в переводимом тексте.

Формы единственного числа

Строка содержит множественное число, но не использует формы множественного числа

Строка содержит множественное число, но не использует формы множественного числа. В случае, если ваша система перевода его поддерживает, вы должны использовать вариант, учитывающий множественное число.

Например, при использовании Python'а и Gettext'а это может выглядеть так:

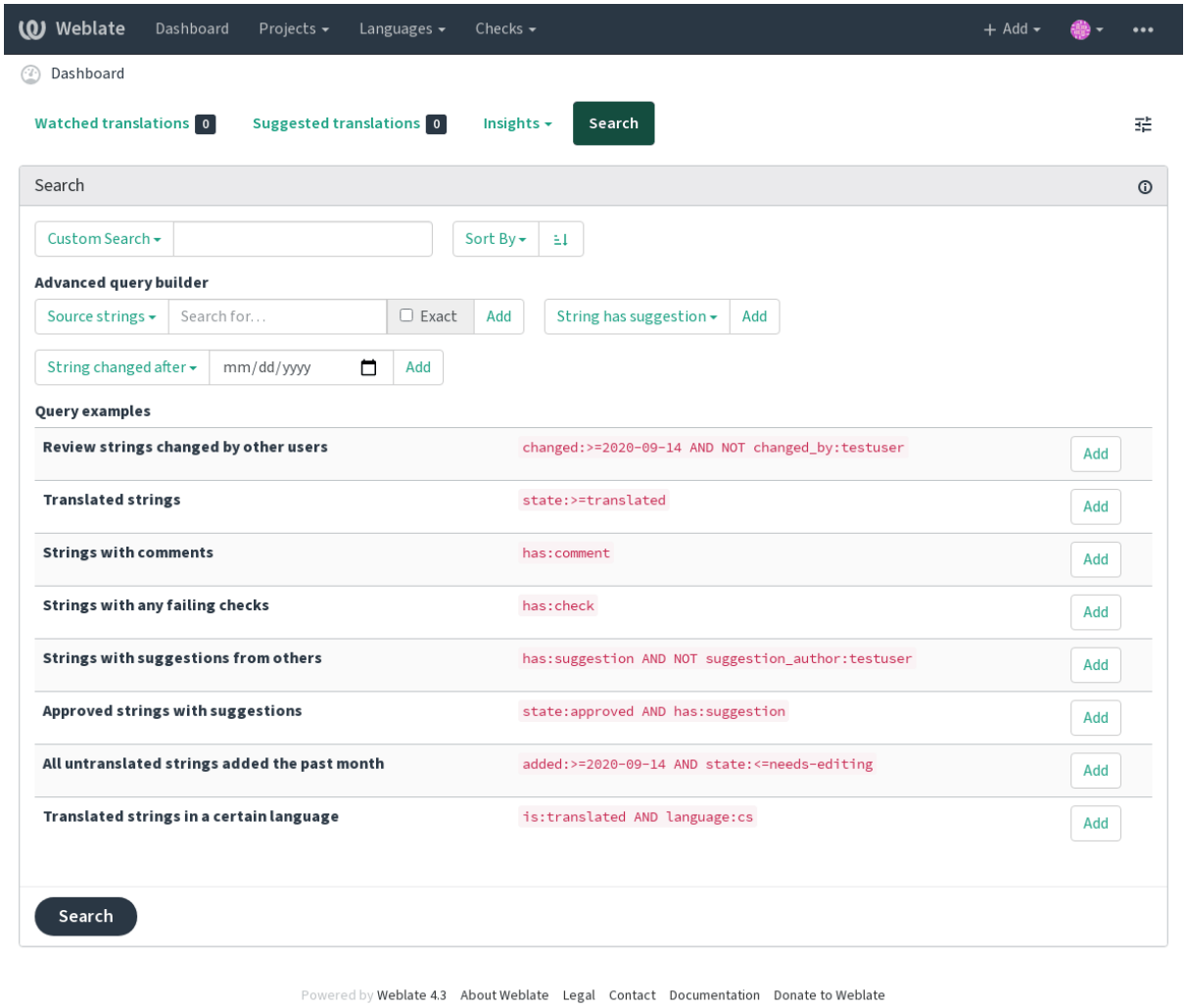
```
from gettext import ngettext

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

1.6 Поиск

Добавлено в версии 3.9.

Для поиска нужных строк можно использовать расширенные поисковые запросы, включающие в себя логические операции, круглые скобки или поиск по конкретным полям.

Если поле не определено, поиск проводится по полям *Источник*, *Перевод* и *Контекст*.



1.6.1 Простой поиск

Любая фраза, введённая в поле поиска, разбивается на слова. При поиске показываются строки, содержащие любое из этих слов. Чтобы найти точную фразу, поместите "фразу для поиска" в кавычки (работают как одинарные ('), так и двойные (") кавычки): `"это строка, заключённая в кавычки"` или `'другая закавыченная строка'`.

1.6.2 Поля

`source:ТЕКСТ` Регистронезависимый поиск по исходной строке.

`target:ТЕКСТ` Регистронезависимый поиск по целевой строке.

`context:ТЕКСТ` Регистронезависимый поиск по строке контекста.

`key:ТЕКСТ` Регистронезависимый поиск по строке ключа.

`note:ТЕКСТ` Регистронезависимый поиск по строке комментария.

`location:ТЕКСТ` Регистронезависимый поиск по строке местоположения.

`priority:ЧИСЛО` Приоритет строки.

`added:ДАТА_И_ВРЕМЯ` Метка времени, когда строка была добавлена в Weblate.

state:ТЕКСТ Поиск по состоянию (`approved`, `translated`, `needs-editing`, `empty`, `read-only`), поддерживает *Операторы поиска по полям*.

pending:ЛОГИЧЕСКОЕ_ЗНАЧЕНИЕ Строка, ожидающая сброса в систему контроля версий.

has:ТЕКСТ Поиск строк, имеющих атрибуты — `plural`, `context`, `suggestion`, `comment`, `check`, `dismissed-check`, `translation`, `variant`, `screenshot` (работает только для исходных строк).

is:ТЕКСТ Поиск по состояниям строки (`pending`, `translated`, `untranslated`).

language:ТЕКСТ Целевой язык строки.

component:ТЕКСТ Плашка компонента, смотрите раздел *Плашка компонента*.

project:ТЕКСТ Project slug, see *Плашка проекта*.

changed_by:ТЕКСТ Строка была изменена автором с указанным именем пользователя.

changed:ДАТА_И_ВРЕМЯ Дата изменения строки, поддерживает *Операторы поиска по полям*.

check:ТЕКСТ Строка имеет неудачные проверки.

dismissed_check:ТЕКСТ Строка имеет скрытые проверки.

comment:ТЕКСТ Поиск в пользовательских комментариях.

comment_author:ТЕКСТ Фильтр по авторским комментариям.

suggestion:ТЕКСТ Поиск в предложениях.

suggestion_author:ТЕКСТ Фильтр по автору предложения.

1.6.3 Логические операторы

Вы можете комбинировать поисковые запросы, используя для формирования сложных запросов операторы `AND`, `OR` и `NOT` и круглые скобки. Например: `state:translated AND (source:hello OR source:bar)`

1.6.4 Операторы поиска по полям

Вы можете указать операторы, диапазоны или частичный поиск по дате или числам:

state:>=translated Состояние `translated` или лучше (`approved`).

changed:2019 Изменено в 2019 году.

changed:[2019-03-01 to 2019-04-01] Изменено между двумя указанными датами.

1.6.5 Оператор точного поиска

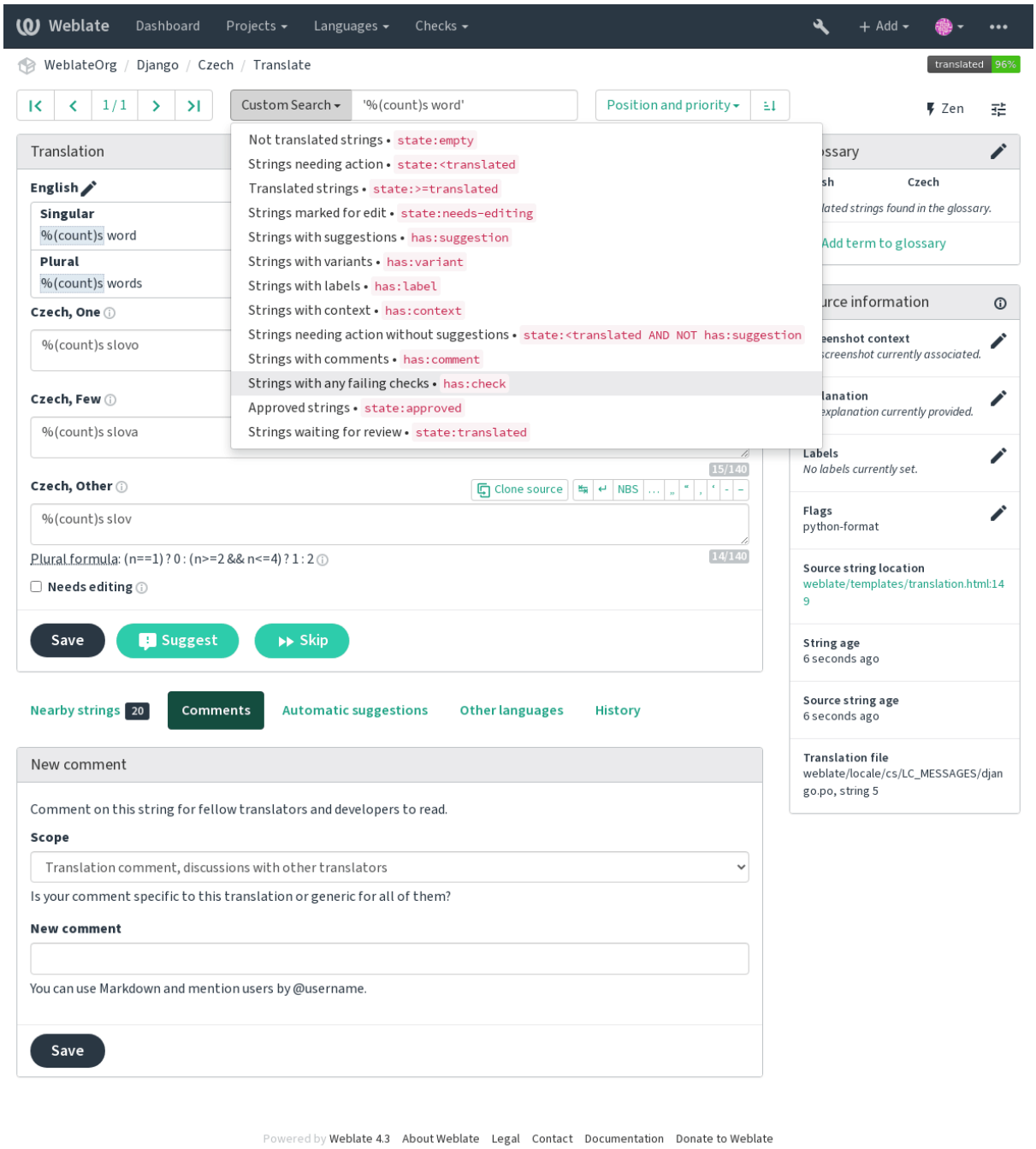
С помощью оператора `=` вы можете сделать запрос на точное совпадение различных строковых полей. Например, для нахождения всех исходных строк, точно равных строке `hello world`, используйте запрос `source:="hello world"`. При поиске односложных выражений кавычки можно опустить. Например, для нахождения всех исходных строк, точно равных строке `hello`, вы можете использовать запрос `source:=hello`.

1.6.6 Регулярные выражения

Везде, где принимается текст, вы также можете указать регулярное выражение в виде `r"выражение"`. Например, для поиска по исходным строкам, содержащим любую цифру от 2 до 5, используйте запрос `source:r"[2-5]"`

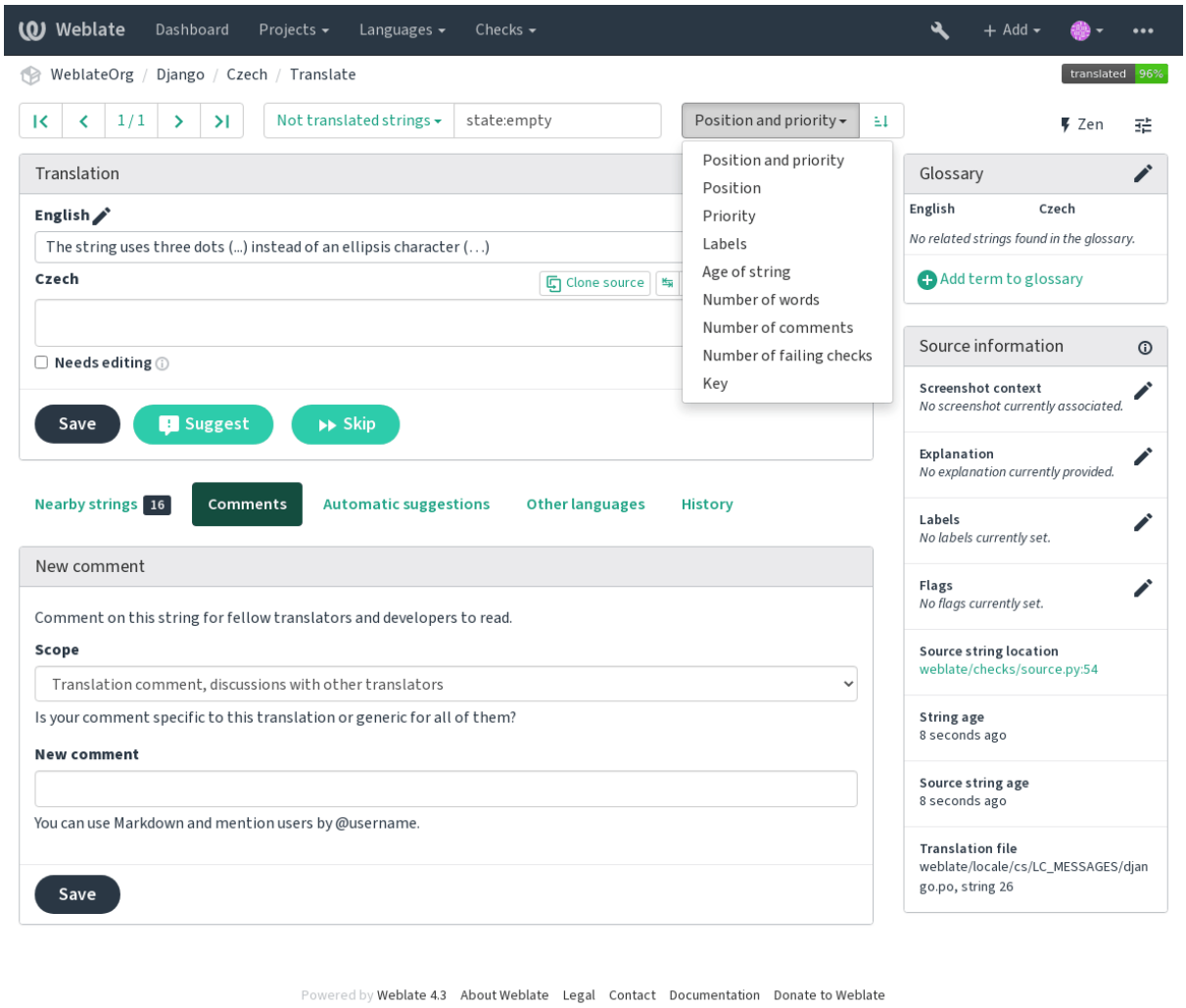
1.6.7 Предопределённые запросы

На странице поиска можно выбрать один из предопределённых запросов, которые позволяют вам быстро получить доступ к наиболее часто встречающимся поисковым запросам:



1.6.8 Упорядочивание результатов

Существует множество вариантов сортировки строк в соответствии с вашими потребностями:



1.7 Руководство разработчика приложений

Использование Weblate — это процесс, который приближает пользователей к вам путём приближения вас к вашим переводчикам. Вы сами решаете, какой объём его возможностей вы хотите использовать.

1.7.1 Знакомство с интернационализацией

У вас есть проект и вы хотите перевести его на несколько языков? Это руководство вам в этом поможет. В нём приведены несколько типичных ситуаций, но большинство примеров достаточно общи и могут быть применены также и к другим сценариям.

Перед тем, как переводить какое-либо программное обеспечение, вы должны осознать, что языки в мире сильно отличаются друг от друга, так что вы не должны делать никаких предположений, основанных на вашем опыте. Если вы попытаетесь собрать предложение из отдельно переведённых сегментов, в большинстве языков оно будет выглядеть очень странно. Также вы должны правильно обрабатывать множественные формы, поскольку во многих языках они строятся по сложным правилам, а используемый вами фреймворк интернационализации должен их поддерживать.

И последнее, но не менее важное: иногда в переведённую строку может понадобиться добавить некий контекст. Представьте себе, что переводчик получил для перевода строку `Sun`. Без контекста большинство людей перевели бы её как «Солнце», подразумевая, что речь идёт о нашей ближайшей звезде, но на самом деле она могла бы быть аббревиатурой для воскресенья (**Sun**day).

Выбор фреймворка интернационализации

Выбирайте тот фреймворк, который является стандартом для вашей платформы, постарайтесь не изобретать заново колесо, создавая свой собственный фреймворк для работы с локализациями. Weblate поддерживает большинство широко используемых фреймворков, подробнее смотрите раздел *Поддерживаемые форматы файлов* (особенно его подраздел *Возможности типов переводов*).

Наши личные рекомендации для некоторых платформ приведены в следующей таблице. Эти рекомендации основаны на нашем опыте, но он не всеобъемлющ, поэтому, когда делаете выбор, всегда учитывайте своё окружение.

Платформа	Рекомендуемый формат
Android	*Строковые ресурсы Android*
iOS	*Строки Apple iOS*
Qt	*Файлы .ts Qt Linguist*
Python	*GNU gettext*
PHP	*GNU gettext*[1]
C/C++	*GNU gettext*
C#	*.XML-файлы ресурсов*
Perl	*GNU gettext*
Ruby	*Файлы Ruby YAML*
Расширения браузеров	*JSON-файлы WebExtension*
Java	*XLIFF*[2]
JavaScript	*JSON -файлы i18next*[3]

Более подробно рабочий процесс для некоторых форматов описан в следующих главах:

- *Перевод программ с помощью GNU Gettext*
- *Перевод документации с помощью Sphinx*
- *Перевод HTML и JavaScript с помощью сети доставки содержимого Weblate*

Интеграция с Weblate

Получение из Weblate обновлений переводов

Для получения обновлённых строк из Weblate вы можете просто затянуть изменения из нижележащего репозитория (либо из файловой системы, либо его можно сделать доступным через *Экспортер Git*). До этого вы можете захотеть зафиксировать любые отложенные изменения (смотрите раздел *Отложенные коммиты*). Это можно сделать либо через пользовательский интерфейс (в разделе *Обслуживание репозитория*), либо из командной строки при помощи *клиента Weblate*.

Эти действия можно автоматизировать, если вы предоставите Weblate доступ на отправку изменений в ваш репозиторий и настроите в *настройках компонента* *URL для отправки*.

См.также:

Непрерывная локализация

[1] Родная поддержка Gettext в PHP содержит множество ошибок и часто отсутствует в сборках под Windows, так что вместо неё мы рекомендуем использовать стороннюю библиотеку motranslator.

[2] Также, если множественные числа вам не нужны, вы можете использовать *Файлы свойств Java*.

[3] Также, если множественные числа вам не нужны, вы можете использовать простой *JSON-файлы*.

Отправка в Weblate изменений строк

Для отправки недавно обновлённые строки в Weblate, просто позвольте ему извлечь их из вышестоящего репозитория. Это можно сделать либо через пользовательский интерфейс (в разделе *Обслуживание репозитория*), либо из командной строки при помощи *клиента Weblate*.

Это действие можно автоматизировать путём установки в вашем репозитории веб-обработчика, запускающего Weblate по появлению нового коммита, подробнее смотрите в разделе *Обновление репозиториев*.

См.также:

Непрерывная локализация

1.7.2 Перевод программ с помощью GNU Gettext

GNU Gettext является одним из наиболее широко используемых инструментов интернационализации свободного программного обеспечения. Он предоставляет простой, но гибкий способ локализации программного обеспечения. Он имеет отличную поддержку форм множественного числа, может добавлять к переводимым строкам дополнительный контекст и на его основе построено довольно большое количество инструментов. И конечно, он имеет отличную поддержку в Weblate (смотрите описание формата файла *GNU gettext*).

Примечание: Если вы собираетесь использовать его в несвободном программном обеспечении, пожалуйста, сначала ознакомьтесь с его лицензией, она может оказаться для вас непригодной.

GNU Gettext можно использовать со множеством различных языков (C, Python, PHP, Ruby, JavaScript и со многими другими), и обычно во фреймворках пользовательского интерфейса уже есть некоторая его поддержка. Стандартным его использованием является вызов функции *gettext()*, которая часто сокращается до своего псевдонима *_()*, чтобы сделать код более простым и лёгким для чтения.

Дополнительно он предоставляет вызовы *pgettext()*, позволяющий предоставлять дополнительный контекст для переводчиков, и *ngettext()*, который умеет обрабатывать формы множественного числа по правилам целевого языка.

Как широко распространённый инструмент, для него существует множество обёрток, значительно упрощающих его использование; вместо описанного ниже ручного вызова Gettext, вы, возможно, захотите попробовать одну из них, например, intltool.

Пример программы

Простая программа на C, использующая Gettext, может выглядеть следующим образом:

```
#include <libintl.h>
#include <locale.h>
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
    int count = 1;
    setlocale(LC_ALL, "");
    bindtextdomain("hello", "/usr/share/locale");
    textdomain("hello");
    printf(
        ngettext(
            "Orangutan has %d banana.\n",
            "Orangutan has %d bananas.\n",
            count
        ),
```

(continues on next page)

(продолжение с предыдущей страницы)

```
        count
    );
    printf("%s\n", gettext("Thank you for using Weblate."));
    exit(0);
}
```

Извлечение строк для перевода

После расстановки по коду вызовов gettext, вы можете воспользоваться программой **xgettext** для извлечения из кода сообщений и записи их в .pot-файлы:

```
$ xgettext main.c -o po/hello.pot
```

Примечание: Существуют альтернативные программы для извлечения строк из кода, например pybabel.

После этого вызова создастся шаблонный файл, который можно использовать для начала новых переводов (используя программу **msginit**) или обновления существующих после изменения кода (для этого можно использовать программу **msgmerge**). В результате получается простой структурированный текстовый файл:

```
# SOME DESCRIPTIVE TITLE.
# Copyright (C) YEAR THE PACKAGE'S COPYRIGHT HOLDER
# This file is distributed under the same license as the PACKAGE package.
# FIRST AUTHOR <EMAIL@ADDRESS>, YEAR.
#
#, fuzzy
msgid ""
msgstr ""
"Project-Id-Version: PACKAGE VERSION\n"
"Report-Msgid-Bugs-To: \n"
"POT-Creation-Date: 2015-10-23 11:02+0200\n"
"PO-Revision-Date: YEAR-MO-DA HO:MI+ZONE\n"
"Last-Translator: FULL NAME <EMAIL@ADDRESS>\n"
"Language-Team: LANGUAGE <LL@li.org>\n"
"Language: \n"
"MIME-Version: 1.0\n"
"Content-Type: text/plain; charset=CHARSET\n"
"Content-Transfer-Encoding: 8bit\n"
"Plural-Forms: nplurals=INTEGER; plural=EXPRESSION;\n"

#: main.c:14
#, c-format
msgid "Orangutan has %d banana.\n"
msgid_plural "Orangutan has %d bananas.\n"
msgstr[0] ""
msgstr[1] ""

#: main.c:20
msgid "Thank you for using Weblate."
msgstr ""
```

Каждая запись с `msgid` в начале определяет переводимую строку текста, специальная пустая запись в начале файла — это заголовок файла, содержащий метаданные о переводе.

Начало нового перевода

Имея шаблон, мы можем начать наш первый перевод:

```
$ msginit -i po/hello.pot -l cs --no-translator -o po/cs.po
Created cs.po.
```

Только что созданный файл `cs.po` уже содержит некоторую информацию. Самое главное заключается в том, что в нём правильно определены формы множественного числа для выбранного языка, и вы можете видеть, что, отражая этот факт, количество форм множественного числа изменилось:

```
# Czech translations for PACKAGE package.
# Copyright (C) 2015 THE PACKAGE'S COPYRIGHT HOLDER
# This file is distributed under the same license as the PACKAGE package.
# Automatically generated, 2015.
#
msgid ""
msgstr ""
"Project-Id-Version: PACKAGE VERSION\n"
"Report-Msgid-Bugs-To: \n"
"POT-Creation-Date: 2015-10-23 11:02+0200\n"
"PO-Revision-Date: 2015-10-23 11:02+0200\n"
"Last-Translator: Automatically generated\n"
"Language-Team: none\n"
"Language: cs\n"
"MIME-Version: 1.0\n"
"Content-Type: text/plain; charset=ASCII\n"
"Content-Transfer-Encoding: 8bit\n"
"Plural-Forms: nplurals=3; plural=(n==1) ? 0 : (n>=2 && n<=4) ? 1 : 2;\n"

#: main.c:14
#, c-format
msgid "Orangutan has %d banana.\n"
msgid_plural "Orangutan has %d bananas.\n"
msgstr[0] ""
msgstr[1] ""
msgstr[2] ""

#: main.c:20
msgid "Thank you for using Weblate."
msgstr ""
```

Этот файл компилируется в оптимизированную двоичную форму — файл .mo, используемый функциями GNU Gettext времени выполнения.

Обновление строк

После добавления в вашу программу ещё нескольких строк или после изменения старых, вы снова выполняете команду **xgettext**, которая пересоздаёт файл шаблона:

```
$ xgettext main.c -o po/hello.pot
```

Затем вы можете обновить отдельные файлы перевода, чтобы они соответствовали вновь созданным шаблонам (обновление, в том числе, переупорядочивает строки для соответствия их новому шаблону):

```
$ msgmerge --previous --update po/cs.po po/hello.pot
```

Импорт в Weblate

Для импорта такого перевода в Weblate, всё, что вам нужно — это определить следующие поля при создании компонента (подробное описание полей смотрите в разделе в *Конфигурация компонента*):

Поле	Значение
Репозиторий исходного кода	URL-адрес репозитория системы контроля версий с вашим проектом
Маска файла	`po/*.po`
Шаблон для новых переводов	`po/hello.pot`
Формат файла	Выберите *gettext PO-файл*
Новый язык	Выберите *Создать новый язык*

Вот и всё, теперь вы готовы начать переводить свою программу!

См.также:

Пример использования Gettext со многими языками можно найти в проекте Weblate Hello на GitHub'e: <https://github.com/WeblateOrg/hello>.

1.7.3 Перевод документации с помощью Sphinx

Sphinx — это инструмент для создания красивой документации. Он использует простой синтаксис reStructuredText и может генерировать вывод во многих форматах. Если вы ищете пример, то эта документация также собирается с его помощью. Очень полезным спутником при использовании Sphinx является сервис Read the Docs, который бесплатно будет собирать и публиковать вашу документацию.

Я не буду заострять внимание на написание самой документации, если вы нуждаетесь в консультации, просто следуйте инструкциям на сайте Sphinx'a. После того, как ваша документация будет написана, её перевод будет довольно простым делом, поскольку Sphinx уже поддерживает для этого всё необходимое, что хорошо описано в его документации по интернационализации. Это буквально вопрос настройки нескольких параметров и вызова инструмента `sphinx-intl`.

Если вы пользуетесь сервисом Read the Docs, вы запустить сборку переведённой документации на Read the Docs. Их документация по локализации покрывает практически всё, что вам нужно знать — создание ещё одного проекта, установку его языка и связывание его с основным проектом в качестве перевода.

Теперь всё, что вам нужно — это перевести содержимое документации. Sphinx генерирует PO-файл для каждого каталога или файла верхнего уровня, что может привести к тому, что у вас появится довольно большое количество файлов для перевода (в зависимости от настройки параметра `gettext_compact`). Вы можете импортировать в Weblate в качестве начального компонента файл `index.po`, а затем настроить надстройку *Обнаружение компонента* для автоматического обнаружения всех остальных файлов.

Таблица 1: Конфигурация компонента

Название компонента	`Документация`
Маска файла	`docs/locales/*/LC_MESSAGES/index.po`
Шаблон для новых переводов	`docs/locales/index.pot`
Формат файла	*PO-файл gettext*
Флаги перевода	`rst-text`

Таблица 2: Конфигурация обнаружения компонентов

Регулярное выражение для сопоставления с файлами перевода	`docs/locales/(?P<language>[^/.]*)/LC_MESSAGES/(?P<component>[^/]*)\.po`
Настроить название компонента	`Документация: {{ component|title }}`
Задать базовый файл для нового перевода	`docs/locales/{{ component }}.pot`

Подсказка: Предпочитаете, чтобы Sphinx генерировал только один PO-файл? Существует неординарный способ этого добиться (используемый документацией самого Weblate), работающий за счёт переопределения способа, которым Sphinx получает gettext-домен документа. Добавьте в свою конфигурацию Sphinx в файле `conf.py` следующий фрагмент кода:

```
import sphinx.transforms.i18n
import sphinx.util.i18n

# Hacky way to have all localized content in single domain
sphinx.transforms.i18n.docname_to_domain = (
    sphinx.util.i18n.docname_to_domain
) = lambda docname, compact: "docs"
```

В будущих выпусках Sphinx'а это поведение может быть поддержано напрямую, смотрите замечание <https://github.com/sphinx-doc/sphinx/issues/784>.

См.также:

Документация по модулю Python Odorik собрана при помощи Sphinx и Read the Docs и переведена с помощью Weblate.

1.7.4 Перевод HTML и JavaScript с помощью сети доставки содержимого Weblate

Начиная с версии Weblate 4.2, локализацию можно экспортировать в CDN, используя надстройку *CDN локализации JavaScript'а*.

Примечание: Эта возможность настроена на Hosted Weblate. На вашей установке она требует дополнительной настройки, смотрите описание параметров `LOCALIZE_CDN_URL` и `LOCALIZE_CDN_PATH`.

Upon installation into your component it will push committed translations (see *Отложенные коммиты*) to the CDN and these can be used in your web pages to localize them.

Создание компонента

First, you need to create a monolingual component which will hold your strings, see *Добавление проектов и компонентов перевода* for generic instructions on that.

In case you have existing repository to start with (for example the one containing HTML files), create an empty JSON file in the repository for the source language (see *Исходный язык*), for example `locales/en.json`. The content should be `{}` to indicate an empty object. Once you have that, the repository can be imported into Weblate and you can start with an addon configuration.

Подсказка: In case you have existing translations, you can place them into the language JSON files and those will be used in Weblate.

For those who do not want to use existing repository (or do not have one), choose *Start from scratch* when creating component and choose *JSON file* as a file format (it is okay to choose any monolingual format at this point).

Настройка надстройки Weblate CDN

The *CDN локализации JavaScript'a* addon provides few configuration options.

Порог перевода Translations translated above this threshold will be included in the CDN.

Селектор CSS Configures which strings from the HTML documents are translatable, see *Извлечение строк для Weblate CDN* and *Локализация HTML с помощью Weblate CDN*.

Имя языковой куки Name of cookie which contains user selected language. Used in the JavaScript snippet for *Локализация HTML с помощью Weblate CDN*.

Извлечение строк из HTML-файлов List of files in the repository or URLs where Weblate will look for translatable strings and offer them for a translation, see *Извлечение строк для Weblate CDN*.

Извлечение строк для Weblate CDN

Строки перевода должны присутствовать в Weblate. Вы можете заводить их вручную, использовать для их создания API или перечислить файлы или URL-адреса в параметре *Извлечение строк из HTML-файлов* и Weblate извлечёт их автоматически. Файлы должны присутствовать в репозитории или содержать удалённые URL-адреса, которые Weblate будет регулярно проверять и анализировать.

В конфигурации по умолчанию для параметра *Селектор CSS* извлекаются элементы с CSS-классом `l10n`, к примеру, из следующих фрагментов извлекаются две строки:

```
<section class="content">
    <div class="row">
        <div class="wrap">
            <h1 class="section-title min-m l10n">Maintenance in progress</h1>
            <div class="page-desc">
                <p class="l10n">We're sorry, but this site is currently down for␣
↪maintenance.</p>
            </div>
        </div>
    </div>
</section>
```

Если вы не хотите изменять существующий код, вы также можете использовать `` * `` в качестве селектора для обработки всех элементов.

Примечание: На текущий момент извлекается только текст элементов. Эта надстройка не поддерживает локализацию атрибутов элементов или элементов с дочерними элементами.

Локализация HTML с помощью Weblate CDN

Для локализации HTML-документа вам необходимо загрузить скрипт `weblate.js`:

```
<script src="https://weblate-cdn.com/a5ba5dc29f39498aa734528a54b50d0a/weblate.js"␣
↪async></script>
```

При загрузке автоматически будут найдены все соответствующие переводимые элементы (на основе настройки из параметра *Селектор CSS*) и их текст будет заменён переводом.

Язык пользователя определяется из настроенной куки, а при её отсутствии используется один из предпочтительных языков браузера пользователя.

The *Language cookie name* can be useful for integration with other applications (for example choose `django_language` when using Django).

Локализация JavaScript

Переводы на отдельные языки представляются в виде двуязычных файлов JSON в сети доставки содержимого. Чтобы их получить, вы можете использовать следующий код:

```
fetch(("https://weblate-cdn.com/a5ba5dc29f39498aa734528a54b50d0a/cs.json")
  .then(response => response.json())
  .then(data => console.log(data));
```

The actual localization logic needs to be implemented in this case.

1.7.5 Предупреждения компонента перевода

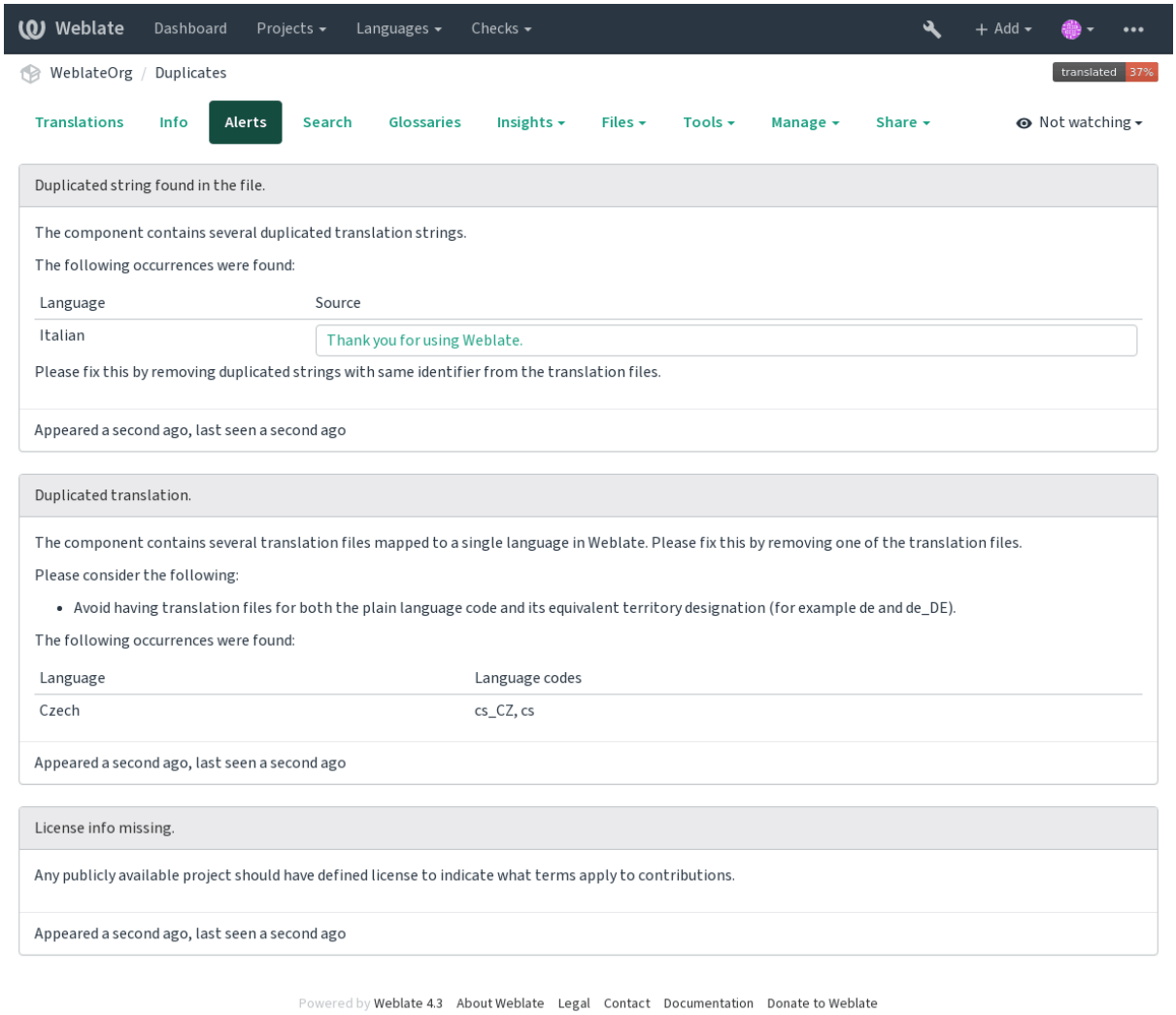
Показывает ошибки в конфигурации Weblate или в проекте перевода для любого компонента перевода. Также предлагает руководство по решению найденных проблем.

В настоящее время проверяются следующие вещи:

- Дублирование исходных строк в файлах перевода
- Дублирование языков в переводах
- Сбои при объединении или при обновлении исходного репозитория
- Неиспользованный новый базовый файл в настройках компонента
- Синтаксические ошибки в файлах перевода
- Дублирование маски файлов, используемой для связанных компонентов
- Сломанные URL-адреса
- Missing licenses

Предупреждения выводятся на соответствующей странице компонента в разделе *Предупреждения*. Если они отсутствуют, компонент очищает все текущие проверки. Предупреждения не могут быть проигнорированы, но исчезнут, как только нижележащая проблема будет устранена.

Компонент с дубликатами исходных строк и языков выглядит вот так:



См.также:

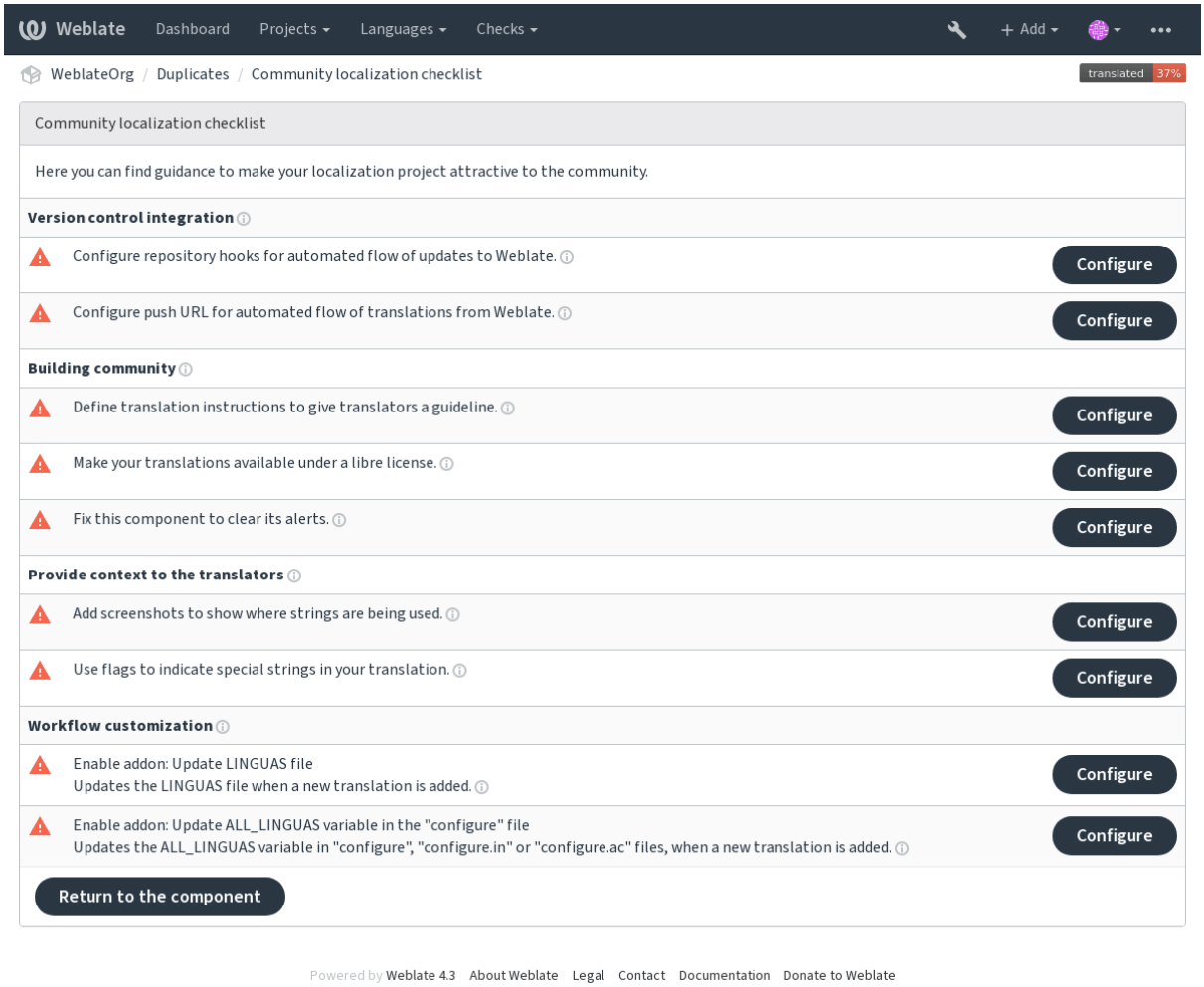
Использование пользовательского центра сертификации

1.7.6 Создание сообщества переводчиков

Контрольный список локализации сообщества

Добавлено в версии 3.9.

Контрольный список локализации сообщества, который можно найти в меню каждого компонента, может дать вам рекомендации, выполнив которые, вы упростите сообществу переводчиков процесс локализации вашей программы.



1.7.7 Управление переводами

Добавление новых переводов

Новые строки становятся доступными для перевода когда они появляются в базовом файле, называемом *Шаблон для новых переводов* (смотрите раздел *Конфигурация компонента*). Если ваш формат файла не требует такого файла, как и происходит в большинстве рабочих процессах одноязычных переводов, вы можете начать с пустых файлов).

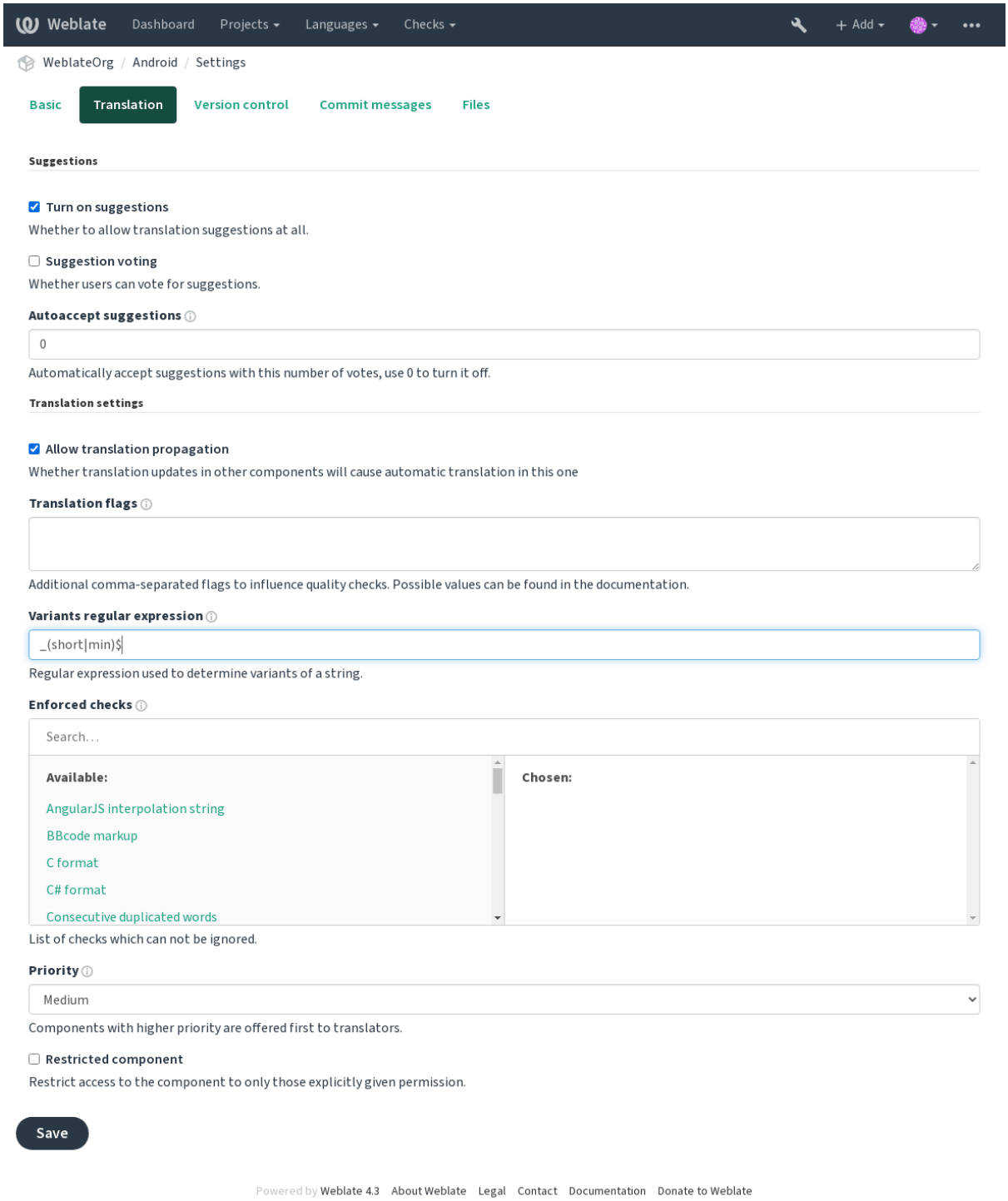
Новые языки могут быть добавлены сразу же, как пользователь их запросит в Weblate, либо же администраторам проекта будет отправлено уведомление, чтобы они одобрили запрос и добавили новый язык вручную. Это можно сделать с помощью кнопки *Начать новый перевод компонента*.

Примечание: Администраторы проектов всегда могут начать перевод напрямую в Weblate.

Языковые файлы, вручную добавленные в систему контроля версий, добавляются в компонент, когда Weblate обновляет репозиторий. Для получения информации о настройках обновления репозитория смотрите раздел *Обновление репозиториев*.

Варианты строк

Варианты полезны для группировки нескольких строк, чтобы переводчики могли видеть все варианты строки в одном месте. Вы можете задать регулярное выражение для группировки строк в *настройках компонента*:

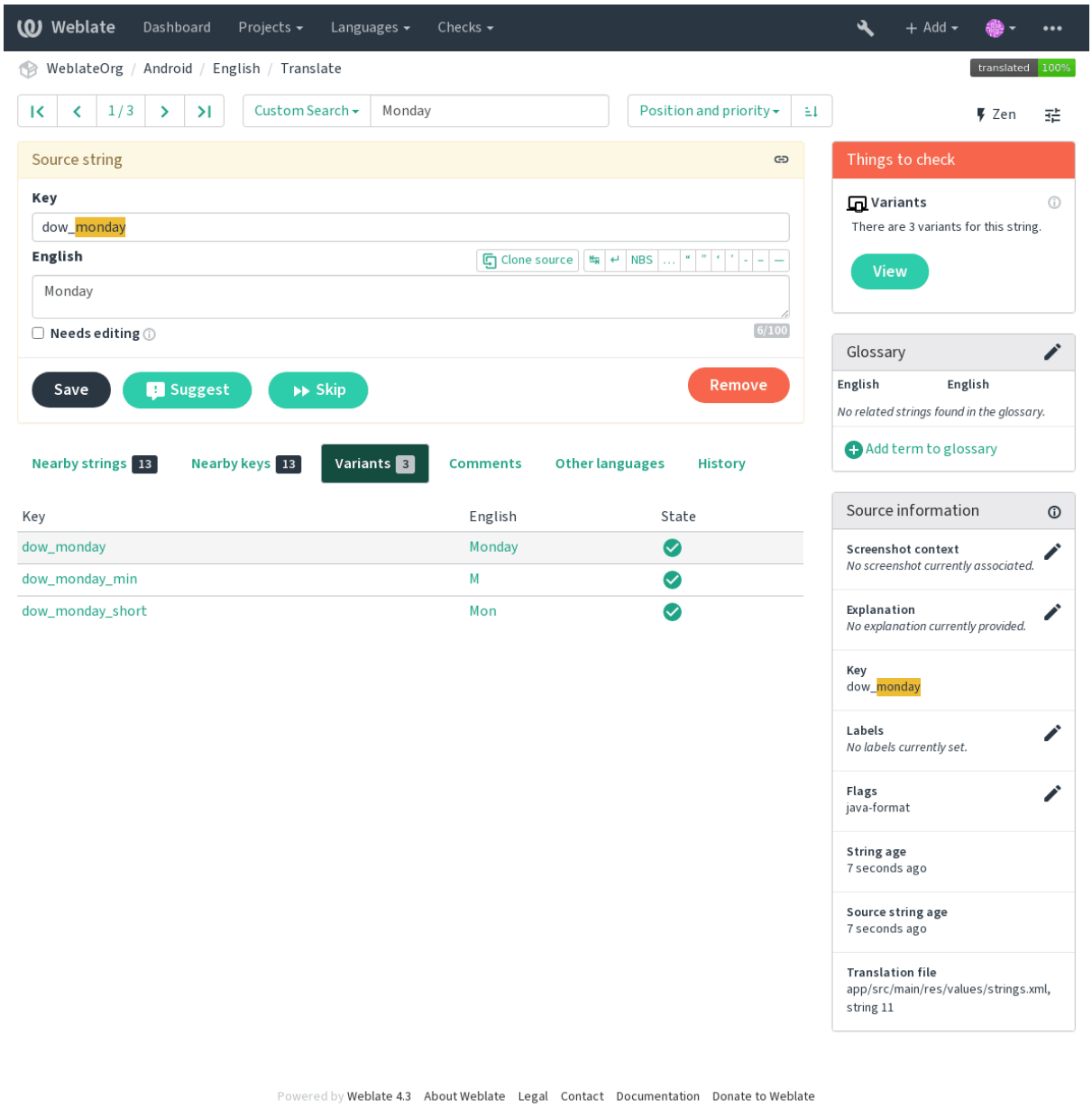


Это выражение сопоставляется с *Ключ*ом для генерации корневого ключа варианта. После этого все найденные строки становятся частью одной группы вариантов, включая перевод, точно совпадающий с корневым ключом, даже если он не подпадает под регулярное выражение.

В следующей таблице приведены некоторые примеры использования:

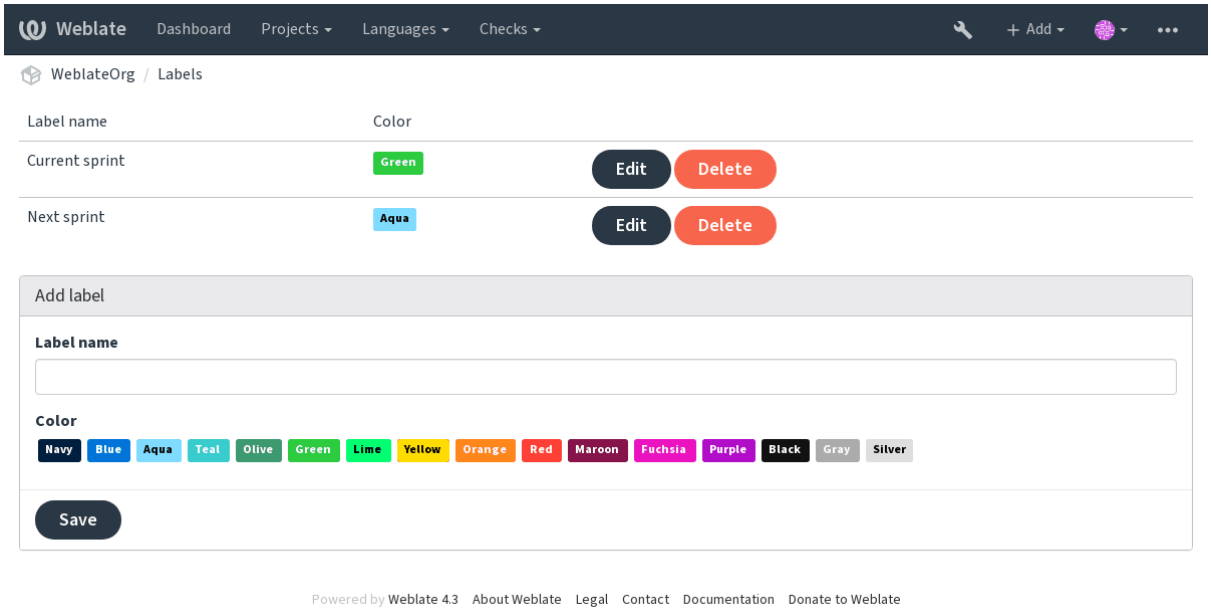
Пример использования	Регулярное выражение варианта	Соответствующие ключи перевода
Идентификация по суффиксу	`(Short|Min)$`	`monthShort`, `monthMin`, `month`
Идентификация по части строки	`#[SML]`	`dial#S.key`, `dial#M.key`, `dial.key`

Позже при переводе строки с одним вариантом группируются вместе:



Метки строк

Разбивают переводимые строки компонента на категории по тексту и цвету, настроенным в конфигурации проекта.



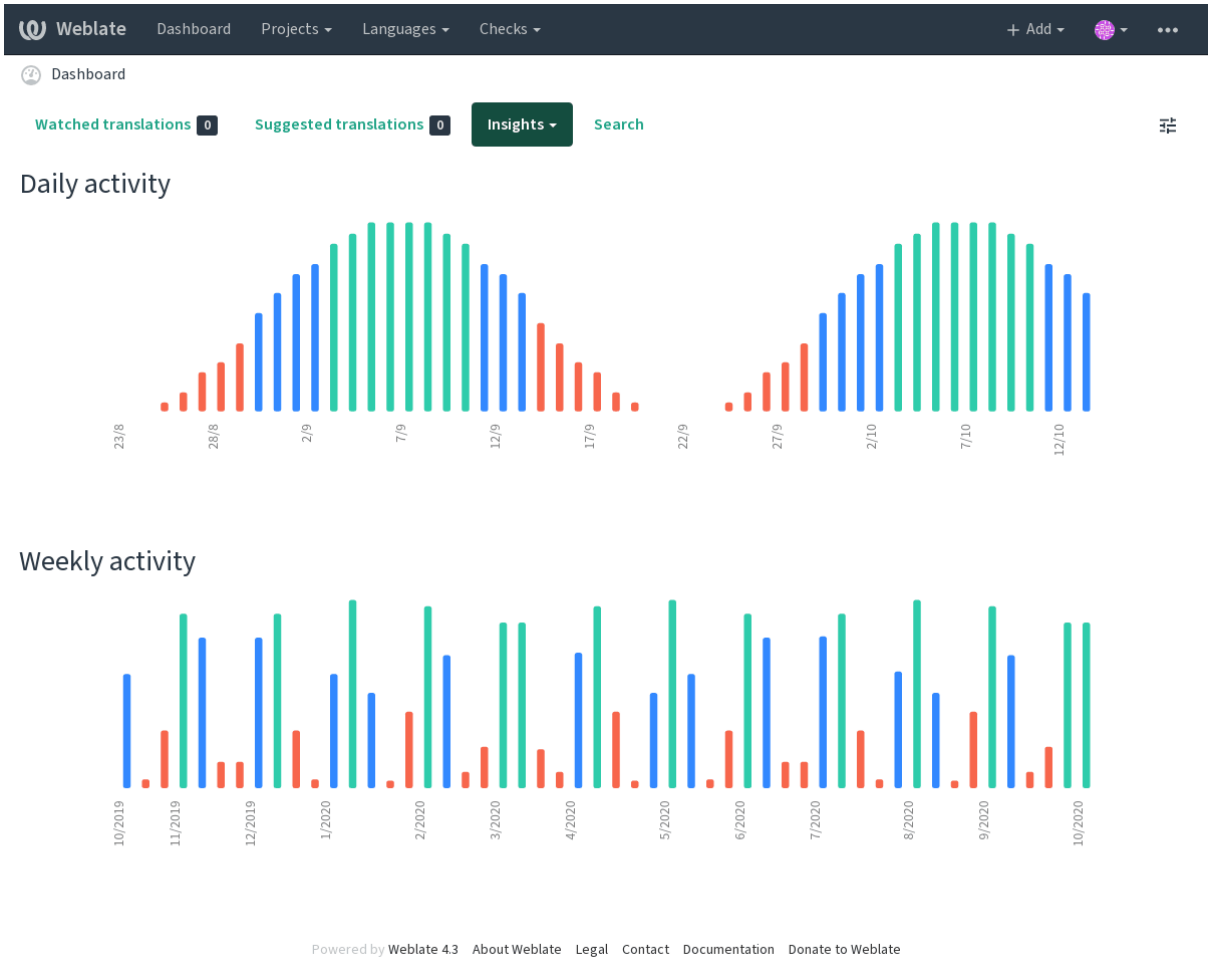
Подсказка: Метки могут быть назначены единицам перевода в окне *Дополнительная информация об исходных строках* путём массовой правки строк или с помощью надстройки *Массовая правка*.

1.7.8 Рецензирование строк

Отчёты о деятельности

Отчёты о деятельности показывают изменения переводов для проектов, компонентов или отдельных пользователей.

Отчёты о деятельности для проекта или компонента доступны из его панели управления, на вкладке *Аналитика*, в подпункте *Деятельность*.



Другие отчеты доступны на вкладке *Аналитика* в подпункте *Отчёты по переводам*.

Активность зарегистрированного в данный момент пользователя можно увидеть, нажав на пункт *Профиль* в меню пользователя из правого верхнего угла.

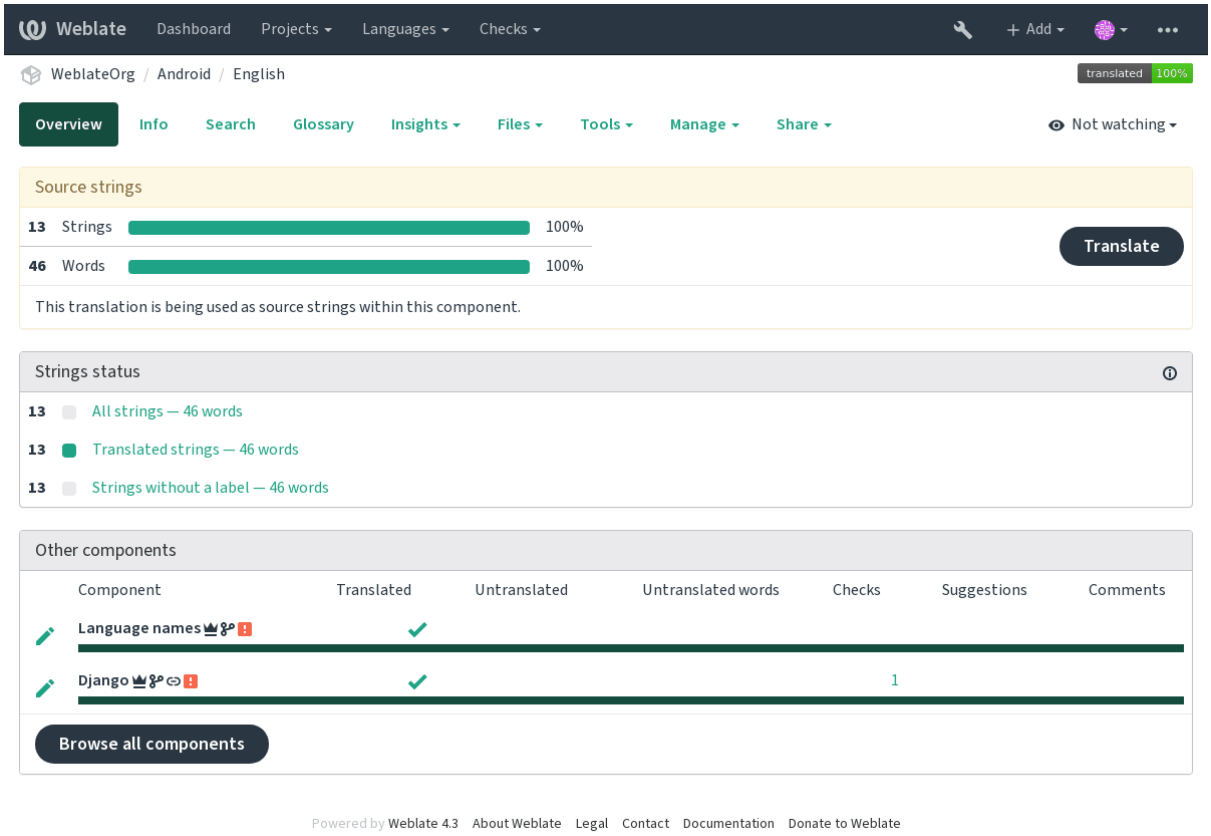
Проверки исходных строк

В Weblate существует множество *проверок* и некоторые из них направлены на улучшение качества исходных строк. Множество неудачных проверок косвенно намекают на то, что исходные строки нужно сделать более дружественными к их переводу. Все типы неудачных проверок исходных строк отображаются на вкладке *Источник* каждого компонента.

Проверки переведённых строк

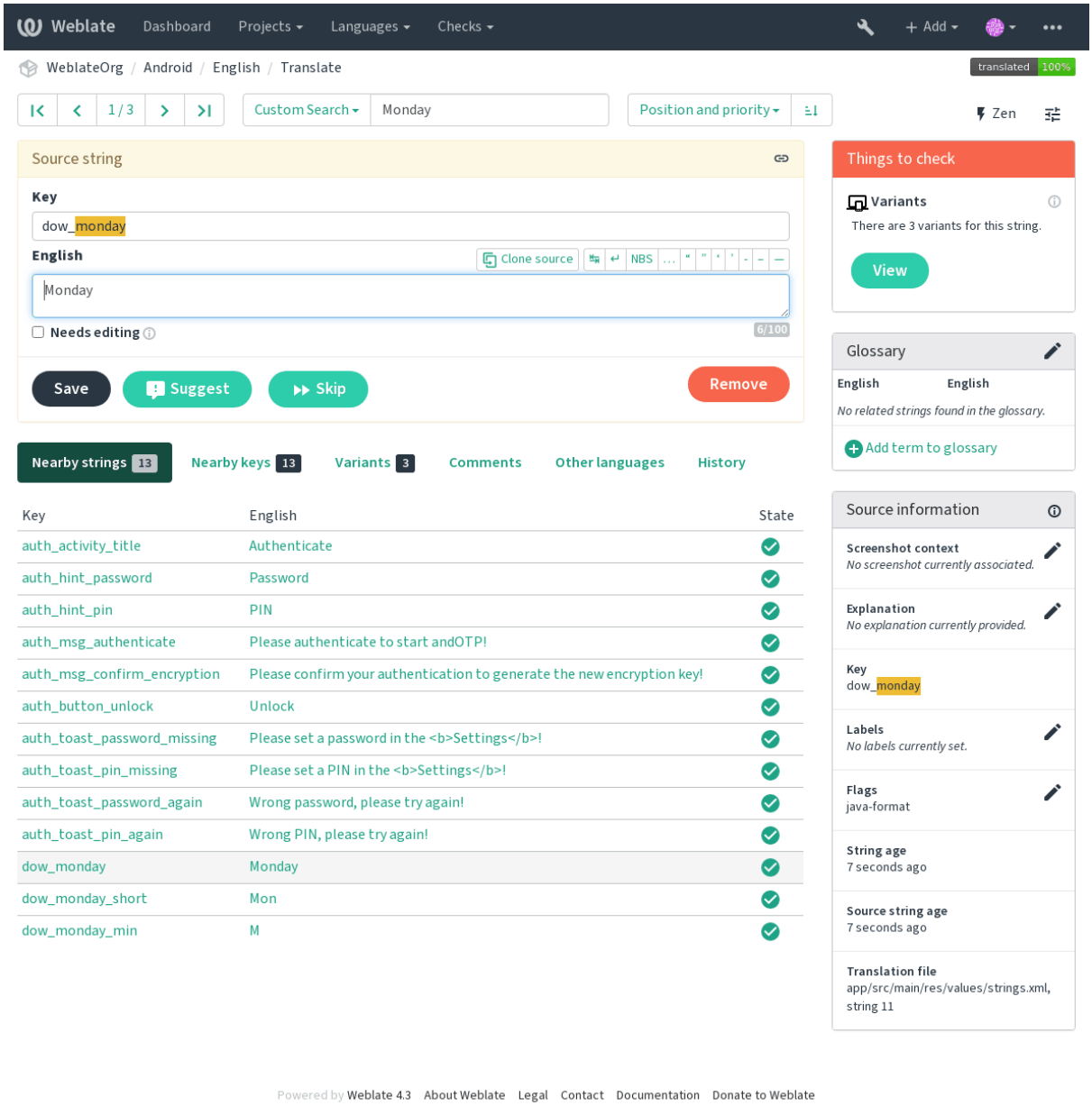
Ошибочные неудачные проверки переведённых строк говорят о том, что проблема находится в исходной строке. Иногда переводчики вместо того, чтобы сообщать о ошибках в исходных строках, просто исправляют их в переводе; типичный такой пример — пропущенная точка в конце предложения.

Просмотр всех неудачных проверок может дать ценную обратную связь для улучшения исходных строк проекта. Чтобы упростить рецензирование исходных строк, Weblate автоматически создаёт фиктивный перевод на исходный язык и показывает в нём проверки для исходных строк:



Одной из наиболее интересных проверок здесь является проверка *Множество неудачных проверок* — она возникает всякий раз, когда возникает неудачная проверка в нескольких переводах указанной строки. Обычно эта строка и оказывается тем, что вы искали — поскольку на ней у переводчиков возникли проблемы с её правильным переводом.

По каждому языку даётся подробный список:

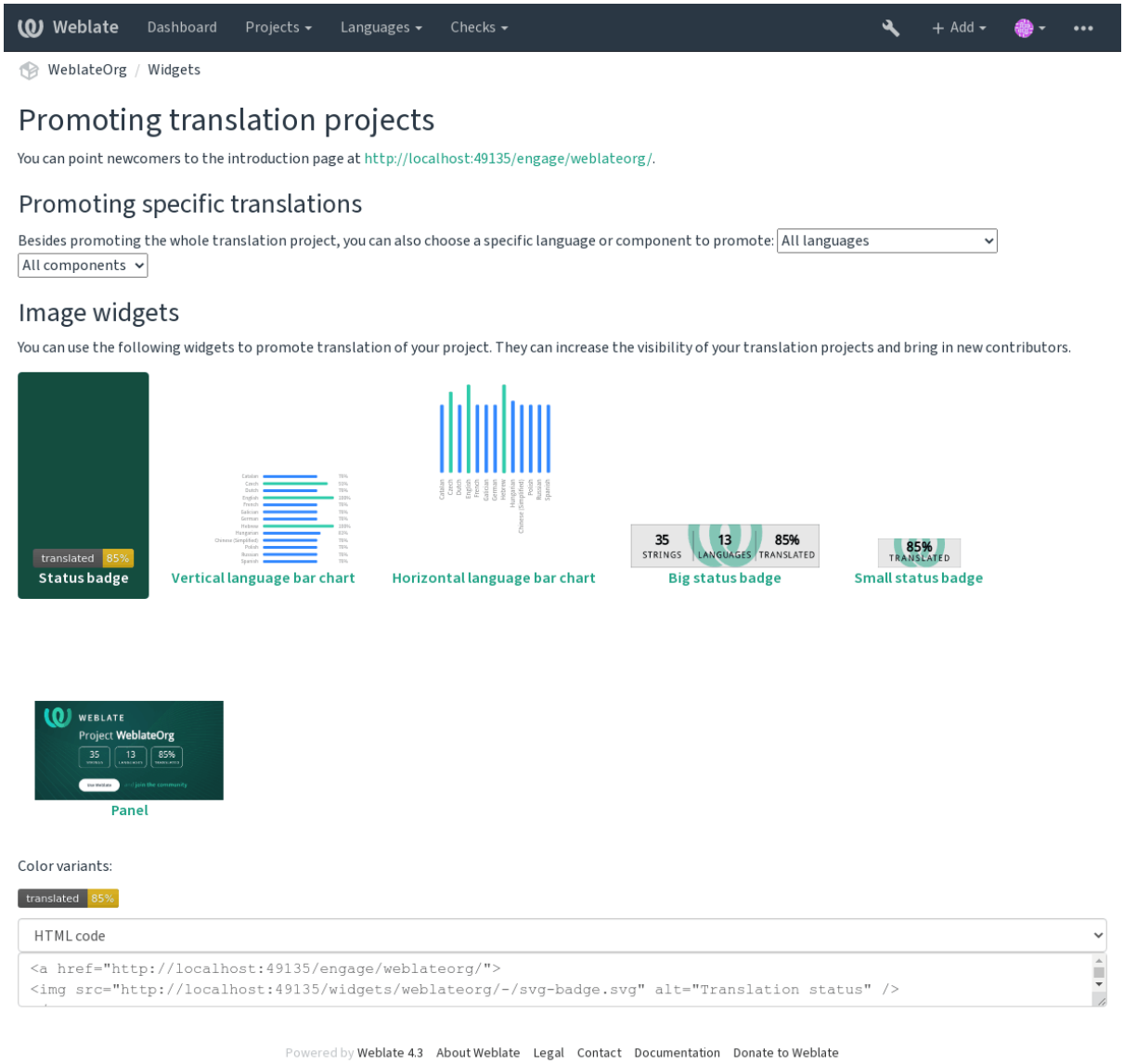


Комментарии к строкам

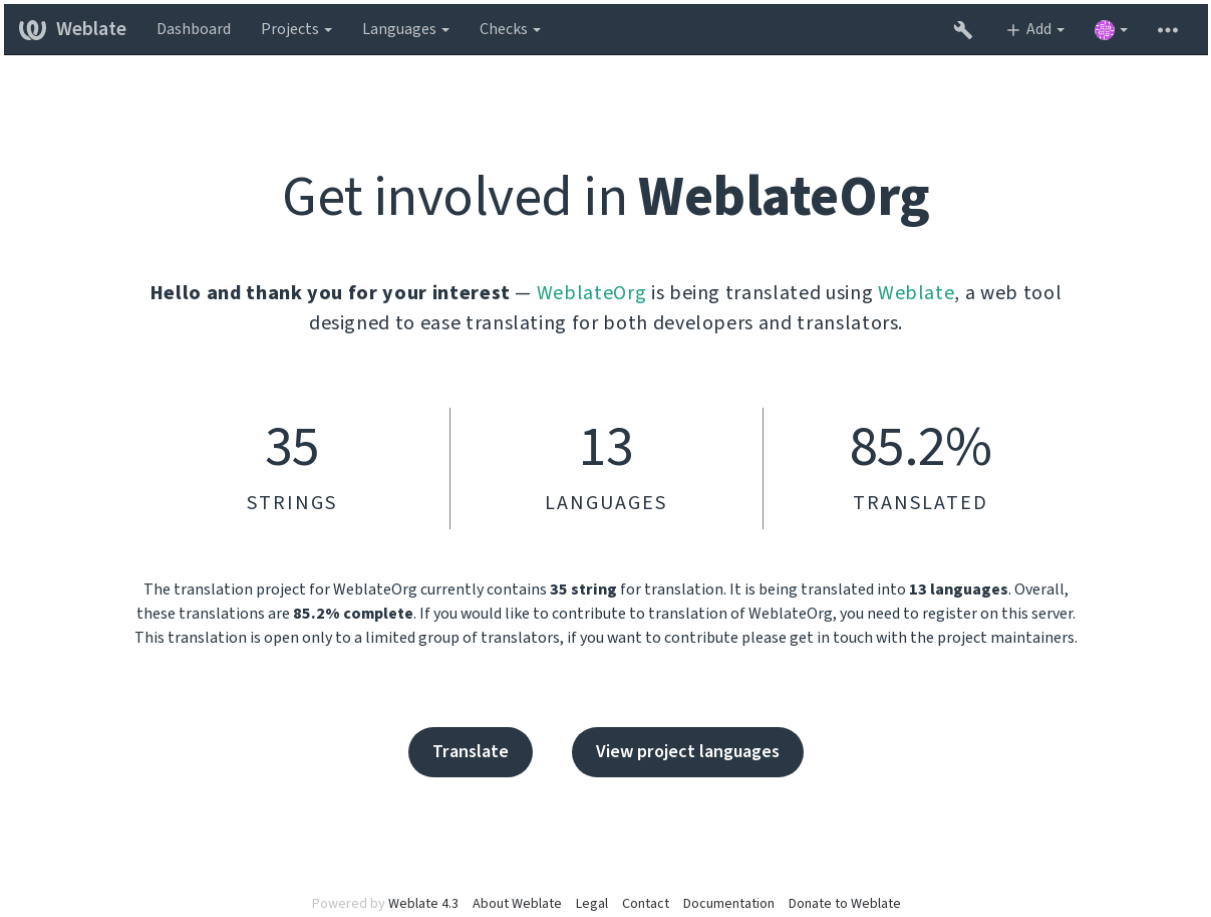
Переводчики могут комментировать как перевод, так и исходные строки. В *конфигурации* каждого компонента можно настроить отправку таких комментариев на электронный адрес; как правило, наилучшим подходом является указание там адреса списка рассылки для разработчиков. Таким образом, вы сможете следить за возникновением проблем в переводе, вовремя о них узнавать и быстро их устранять.

1.7.9 Продвижение перевода

Weblate предоставляет вам виджеты, которые вы можете разместить на своем сайте или в других источниках для продвижения проекта перевода. Ещё он предоставляет красивую страницу приветствия для новых участников, на которой они могут получить основную информацию о переводе. Кроме того, вы можете поделиться информацией о переводе через Facebook или Twitter. Все эти возможности можно найти на вкладке *Поделиться*:

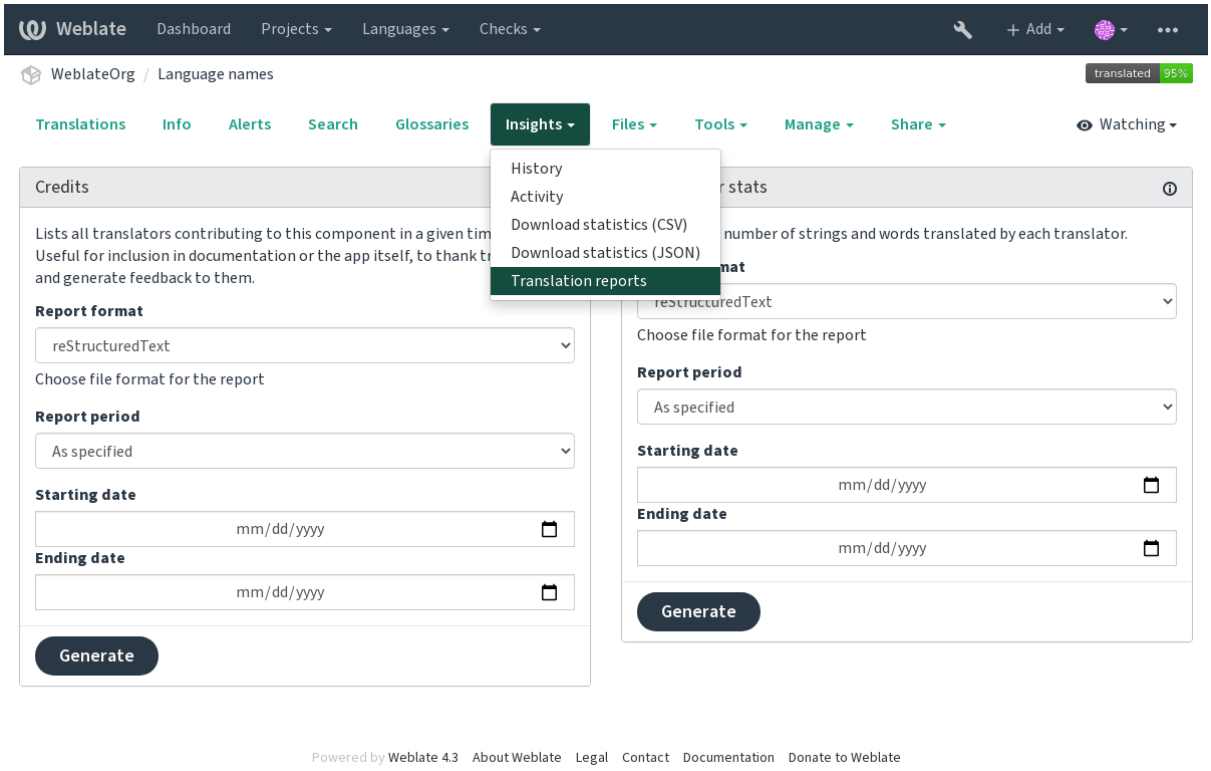


Все эти значки снабжаются ссылкой на простую страницу, которая объясняет пользователям, как переводить с помощью Weblate:



1.7.10 Отчёт о прогрессе перевода

Функции отчётности дают вам представление о том, как продвигался перевод за определённый период. В отчёте содержится сводка по переводам того или иного компонента за определённый период времени. Инструмент для создания отчётов находится в меню *Аналитика* любого компонента перевода, проекта или на панели управления:



На этой странице доступно несколько инструментов создания отчётов и все они могут производить вывод в HTML, reStructuredText или JSON. Первые два формата подходят для встраивания статистики в существующую документацию, в то время как JSON полезен для дальнейшей обработки данных.

Списки переводчиков

Генерирует документ, пригодный для выражения благодарности переводчикам — отсортированный по языку и содержащий список всех участников, внёсших вклад в перевод на данный язык:

```
* Czech

    * Michal Čihař <michal@cihar.com> (10)
    * John Doe <john@example.com> (5)

* Dutch

    * Jane Doe <jane@example.com> (42)
```

Будет отрисован как:

- Czech
  - Michal Čihař <michal@cihar.com> (10)
  - John Doe <john@example.com> (5)
- Нидерландский
  - Jae Doe <jane@example.com> (42)

Подсказка: Число в скобках указывает количество вкладов за указанный период.

Статистика участника

Генерирует количество переведённых слов и строк по имени переводчика:

```
========================================= ========================================␣
↪======================== ======================== ========================␣
↪======================== ======================== ========================␣
↪======================== ======================== ========================␣
↪======================== ======================== ========================␣
↪======================== ======================== ========================␣
↪======================== ======================== ========================␣
↪======================== ========================
Name                                      Email                                    ␣
↪Count total              Source words total       Source chars total      ␣
↪Target words total       Target chars total       Count new               ␣
↪Source words new         Source chars new         Target words new        ␣
↪Target chars new         Count approved           Source words approved   ␣
↪Source chars approved    Target words approved    Target chars approved    Count␣
↪edited             Source words edited      Source chars edited      Target␣
↪words edited      Target chars edited
========================================= ========================================␣
↪======================== ======================== ========================␣
↪======================== ======================== ========================␣
↪======================== ======================== ========================␣
↪======================== ======================== ========================␣
↪======================== ======================== ========================␣
↪======================== ======================== ========================␣
↪======================== ========================
Michal Čihař                              michal@cihar.com                         ␣
↪                      1                         3                       24       ␣
↪                3                        21                       1              ␣
↪          3                       24                       3                    ␣
↪   21                       0                        0                        0␣
↪                      0                        0                        0      ␣
↪                 0                        0                        0            ␣
↪           0
Allan Nordhøy                             allan@example.com                        ␣
↪                      2                         5                       25       ␣
↪                4                        28                       2              ␣
↪          3                       24                       3                    ␣
↪   21                       0                        0                        0␣
↪                      0                        0                        0      ␣
↪                 0                        0                        0            ␣
↪           0
========================================= ========================================␣
↪======================== ======================== ========================␣
↪======================== ======================== ========================␣
↪======================== ======================== ========================␣
↪======================== ======================== ========================␣
↪======================== ======================== ========================␣
↪======================== ======================== ========================␣
↪======================== ========================
```

И будет отрисован в виде:

Имя	Email	Count total	Source words total	Target chars total	Target words total	Count new	Source words new	Source chars new	Target words new	Target chars new	Count approved	Source words approved	Source chars approved	Target words approved	Target chars approved	Count edited	Source words edited	Source chars edited	Target words edited	Target chars edited
Michal Čihař	michal@cihar.com	1	3	24	3	21	1	3	24	3	21	0	0	0	0	0	0	0	0	0	0
Allan Nordhøy	allan@example.com	2	5	25	4	28	2	3	24	3	21	0	0	0	0	0	0	0	0	0	0

Он может быть полезен, если вы платите переводчикам в зависимости от объёма выполненной ими работы, он даёт вам различную статистику по работе переводчиков.

Вся статистика доступна в трёх вариантах:

Итоги Общее количество отредактированных строк.

Новые Недавно переведённые строки, у которых раньше перевода не было.

Одобренные Считает количество одобренных строк в процессе рецензирования (смотрите раздел *Выделенные рецензенты*).

Отредактированные Отредактированные строки, которые ранее уже были переведены.

Для каждого из этих вариантов доступны следующие метрики:

Количество Количество строк.

Правки Количество правок в строке, измеряемое в расстоянии Дамерау — Левенштейна.

Исходных слов Количество слов в исходной строке.

Исходных символов Количество символов в исходной строке.

Целевых слов Количество слов в переведённой строке.

Целевых символов Количество символов в переведённой строке.

1.8 Рабочие процессы перевода

Weblate поддерживает несколько рабочих процессов перевода.

Ниже приведён не полный список способов настройки Weblate. Вы можете основывать другие рабочие процессы на перечисленных здесь самых обычных примерах.

1.8.1 Доступ к переводу

Управление доступом рассматривается в рабочих процессах не слишком подробно, поскольку каждый параметр контроля доступа может быть применён к любому рабочему процессу. Пожалуйста, ознакомьтесь с этой документацией для получения информации о том, как управлять доступом к переводам.

В следующих главах *любой пользователь* означает пользователя, имеющего доступ к переводу. Это может быть любой аутентифицированный пользователь, если проект является публичным, или пользователь, имеющий в проекте разрешение *Перевод*.

1.8.2 Состояния перевода

Каждая переведённая строка может находиться в одном из следующих состояний:

Не переведёно Перевод пуст, он может как храниться в файле, так и не храниться в нём, в зависимости от формата файла.

На правку Перевод нуждается в редактировании, обычно это результат изменения исходной строки. Перевод хранится в файле, в зависимости от формата файла он может быть помечен как нуждающийся в редактировании (например, получить флаг неточного перевода).

Ожидает рецензирования Перевод выполнен, но ещё не проверен. Он хранится в файле как готовый перевод.

Одобрено Перевод был одобрен в процессе рецензирования. Он больше не может быть изменён переводчиками — только рецензентами. Переводчики могут только добавлять к нему предложения.

Предложения Предложения хранятся только в Weblate, а не в файле перевода.

1.8.3 Прямой перевод

Это самая обычная настройка для небольших команд, любой может переводить напрямую. Также она действует по умолчанию в Weblate.

- *Любой пользователь* может редактировать переводы.
- Предложения являются необязательным способом предложения изменений, когда переводчики не уверены в этом изменении.

Параметр	Значение	Примечание
Включить рецензии	выкл	Настраивается на уровне проекта.
Включить предложения	вкл	Полезны для пользователей в качестве способа предложения изменений, когда они в них не уверены.
Голосование по предложению	выкл	
Автоматическое принятие предложений	0	
Группа переводчиков	*Пользователи*	Или *Перевод*, если включено *Управление доступом*.
Группа рецензентов	Н/Д	Не используется.

1.8.4 Коллегиальное рецензирование

В этом рабочем процессе любой может добавлять предложения и чтобы они стали переводом, они должны получить одобрение от другого участника (или участников).

- *Любой пользователь* может добавлять предложения.
- *Любой пользователь* может голосовать за предложения.
- Предложения становятся переводами, когда они получают заранее определённое количество голосов.

Параметр	Значение	Примечание
Включить рецензии	выкл	Настраивается на уровне проекта.
Включить предложения	вкл	
Голосование по предложению	выкл	
Автоматическое принятие предложений	1	Вы можете установить более высокое значение, чтобы потребовать больше рецензий от коллег.
Группа переводчиков	*Пользователи*	Или *Перевод*, если включено *Управление доступом*.
Группа рецензентов	Н/Д	Не используется, рецензентами являются все переводчики.

1.8.5 Выделенные рецензенты

Добавлено в версии 2.18: Соответствующий рабочий процесс рецензирования поддерживается с версии Weblate 2.18.

В рабочем процессе с выделенными рецензентами у вас есть две группы пользователей, одна из которых может отправлять переводы, а другая — их проверять, чтобы убедиться в том, что переводы согласованы и хорошего качества.

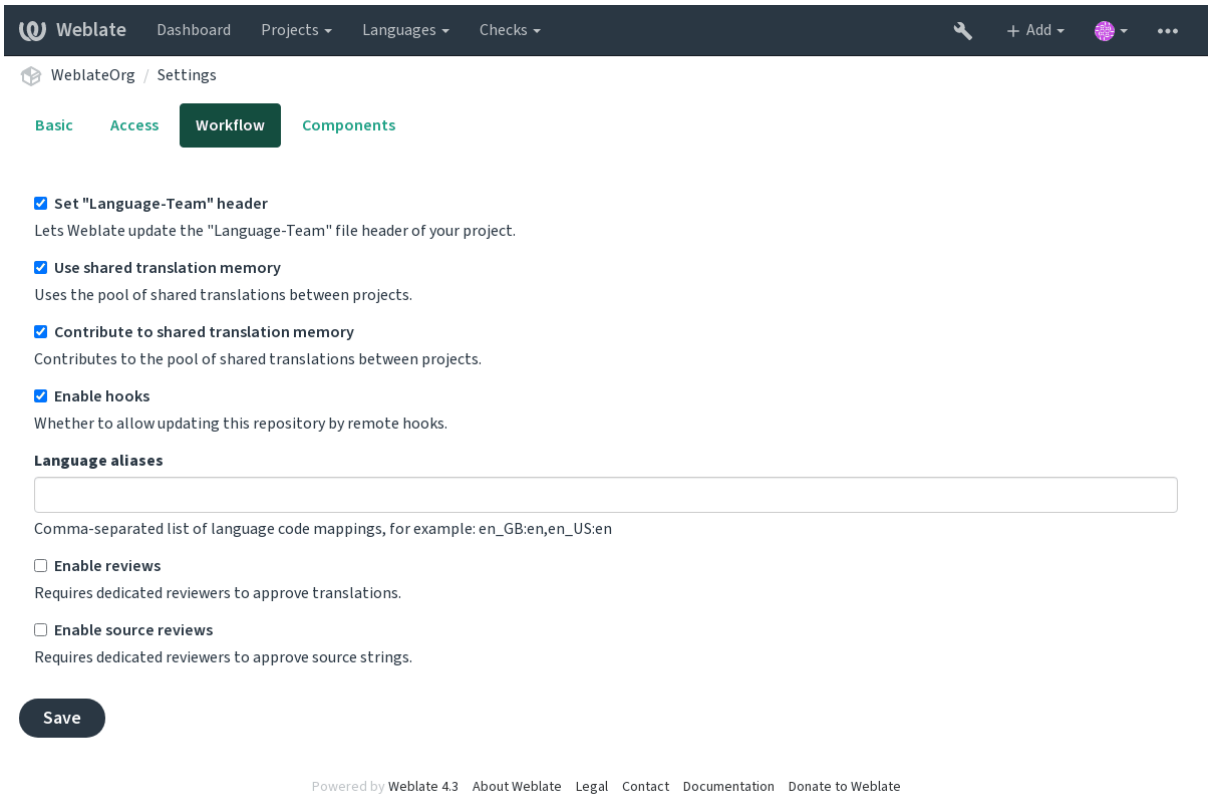
- *Любой пользователь* может редактировать не одобренные переводы.
- *Рецензент* может одобрять строки или снимать с них одобрение.

- *Рецензент* может редактировать все переводы (включая одобренные).
- Также для внесения предложений на изменение одобренных строк могут использовать предложения.

Параметр	Значение	Примечание
Включить рецензии	вкл	Настраивается на уровне проекта.
Включить предложения	выкл	Полезны для пользователей в качестве способа предложения изменений, когда они в них не уверены.
Голосование по предложению	выкл	
Автоматическое принятие предложений	0	
Группа переводчиков	*Пользователи*	Или *Перевод*, если включено *Управление доступом*.
Группа рецензентов	*Рецензенты*	Или *Рецензировать*, если включено *Управление доступом*.

1.8.6 Включение рецензирования

Рецензирование можно включить в конфигурации проекта, на странице *Рабочий процесс* настроек проекта (находится в меню *Управлять* → *Параметры*):

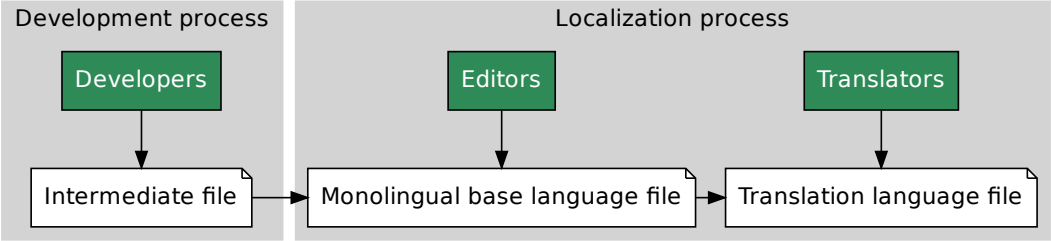


Примечание: В зависимости от конфигурации Weblate этот параметр может быть недоступен. Например, на Hosted Weblate он недоступен для проектов, размещённых бесплатно.

1.8.7 Шлюз качества для исходных строк

Во многих случаях строки на исходном языке поступают от разработчиков, потому что они пишут код и предоставляют исходные строки. Однако зачастую разработчики не являются носителями исходного языка и не обеспечивают желаемого качества исходных строк. Помочь решить эту проблему может промежуточный перевод — это дополнительный шлюз качества для строк между разработчиками, переводчиками и пользователями.

Если установлен *Файл промежуточного языка*, он будет использоваться в качестве источника исходных строк, но его можно будет отредактировать, чтобы отполировать предложения на исходном языке. Когда строка на исходном языке будет готова, она также станет доступна переводчикам для перевода на остальные языки.



См.также:

Файл промежуточного языка, *Одноязычный базовый файл*, *Двуязычные и одноязычные форматы*

1.8.8 Рецензирование исходных строк

При включённом параметре *Включить рецензии исходных строк* процесс рецензирования может быть применён также к исходным строкам. После включения этого параметра пользователи смогут сообщать о проблемах в исходных строках. Реальный процесс зависит от того, используете ли вы двуязычные или одноязычные форматы.

Для одноязычных форматов рецензирование исходной строки ничем не отличается от обычного *рецензирования* — когда сообщается о проблеме в исходной строке, она помечается флагом *На правку*.

Двуязычные форматы не позволяют напрямую редактировать исходные строки (они, как правило, извлекаются непосредственно из исходного кода). В этом случае к строкам, передаваемым переводчиками, прикрепляется метка *Исходная строка требует рецензирования*. Вы должны просмотреть такие строки и либо отредактировать их в исходном коде, либо удалить эту метку.

См.также:

Двуязычные и одноязычные форматы, *Выделенные рецензенты*, *Метки строк*

1.9 Часто задаваемые вопросы

1.9.1 Конфигурация

Как создать автоматизированный рабочий процесс?

Weblate может выполнить все действия, связанные с переводом, в полуавтоматическом режиме. Если вы дадите ему доступ на отправку изменений в ваш репозиторий, переводы смогут в нём появляться без запроса подтверждения, если только не возникнут конфликты слияния.

1. Настройте свой Git-репозиторий так, чтобы он сообщал Weblate, когда происходят какие-либо изменения; для получения информации о том, как это сделать, смотрите раздел *Обработчики уведомлений*.
2. В *конфигурации вашего компонента* установите URL для отправки, это позволит Weblate'у отправлять изменения в ваш репозиторий.
3. В *конфигурации вашего проекта* включите отправку-при-коммите, это заставит Weblate отправлять изменения в ваш репозиторий всякий раз, когда они происходят в Weblate.

См.также:

Непрерывная локализация, *Предотвращение конфликтов слияния*

Как получить доступ к репозиториям по SSH?

Информацию о настройке SSH-ключей смотрите в разделе *Доступ к репозиториям*.

Как исправить в переводах конфликты слияния?

Время от времени, когда файл перевода изменяется одновременно и в Weblate, и в вышестоящем репозитории, происходят конфликты слияния. Обычно этого можно избежать, влив переводы из Weblate до внесения изменений в файлы перевода (например, перед запуском msgmerge). Просто скажите Weblate закоммитить все отложенные переводы (вы можете сделать это в подпункте *Обслуживание репозитория* меню *Управлять*) и влейте их в репозиторий (если не включена автоматическая отправка).

Если же вы уже нарвались на конфликт слияния, то самым простым способом решения всех конфликтов локально на вашей рабочей машине будет добавление Weblate в качестве удалённого репозитория, объединения изменений из него с изменениями из вышестоящего репозитория с исправлением возникших конфликтов. После того, как вы отправите изменения обратно, Weblate сможет использовать объединённую версию без каких-либо других специальных действий.

Примечание: В зависимости от ваших настроек, доступ к репозиторию Weblate может потребовать аутентификации. При использовании встроенного в Weblate *экспортера в Git* вы аутентифицируетесь с помощью своего имени пользователя и ключа API.

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

# Update weblate remote:
git remote update weblate

# Merge Weblate changes:
git merge weblate/master

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

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

(continues on next page)

(продолжение с предыдущей страницы)

```
# Open Weblate for translation:
wlc unlock
```

Если вы используете в Weblate несколько ветвей, то же самое вы можете сделать со всеми ними:

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

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

# Merge master branch:
git checkout master
git merge weblates-second/master
... # Resolve conflicts
git commit

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

В случае использования файлов PO gettext'а существует способ полуавтоматического исправления конфликтов:

Извлеките и сохраните локальный клон Git-репозитория Weblate. Также возьмите второй свежий локальный клон вышестоящего Git-репозитория (то есть, вам нужно две копии вышестоящего Git-репозитория: нетронутая и рабочая):

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

# Update Weblate remote:
git remote update weblate

# Merge Weblate changes:
git merge weblate/master

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

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

См.также:

Как экспортировать используемый Weblate репозиторий Git?, *Непрерывная локализация*, *Предотвращение конфликтов слияния*

Как мне сразу перевести несколько веток?

Weblate поддерживает одновременную отправку изменений перевода внутри одного *проекта*. В каждом *компоненте*, в котором она включена (это поведение по умолчанию), внесённое изменение автоматически вносится и в прочие компоненты. Таким образом, переводы остаются синхронизированными, даже если сами ветки уже достаточно сильно разошлись, и простое объединение изменений перевода между ними невозможно.

После вливания изменений из Weblate, вам, возможно, придётся объединить эти ветви (в зависимости от вашего рабочего процесса разработки) для устранения различий:

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

См.также:

Поддержание единого перевода в разных компонентах

Как переводить многоплатформенные проекты?

Weblate поддерживает широкий диапазон форматов файлов (смотрите раздел *Поддерживаемые форматы файлов*) и самым простым подходом является использование родного формата для каждой платформы.

После того, как вы добавите все файлы перевода для платформ в качестве компонентов в один проект (смотрите раздел *Добавление проектов и компонентов перевода*), вы сможете использовать функцию распространения перевода (включена по умолчанию и может быть отключена в *настройках компонента*) для перевода строк одновременно для всех платформ.

См.также:

Поддержание единого перевода в разных компонентах

Как экспортировать используемый Weblate репозиторий Git?

Репозиторий ничем таким не выделяется, он живёт в каталоге `DATA_DIR` и называется `vcs/<проект>/<компонент>/`. Если у вас есть SSH-доступ к этой машине, вы можете работать с репозиторием напрямую.

Для анонимного доступа вы можете запустить Git-сервер и позволить ему открыть репозиторий во внешний мир.

В качестве альтернативы вы можете использовать *Экспортер Git* изнутри Weblate для автоматизации этого процесса.

Какие есть способы для отправки изменений в вышестоящий репозиторий?

В значительной степени это зависит от вашей установки, в этом плане Weblate достаточно гибок. Вот несколько примеров используемых с Weblate'ом рабочих процессов:

- Weblate автоматически отправляет и вливает изменения (смотрите раздел *Как создать автоматизированный рабочий процесс?*).
- Вы вручную приказываете Weblate отправить изменения (ему нужны права доступа на отправку изменений в вышестоящий репозиторий).
- Кто-то вручную вливает изменения из git-репозитория Weblate в вышестоящий репозиторий.
- Кто-то переписывает созданную Weblate историю (например, устраняя коммиты слияния), объединяет изменения и приказывает Weblate'у сбросить содержимое в вышестоящий репозиторий.

Конечно, вы можете смешивать все эти рабочие процессы по своему усмотрению.

Как мне ограничить Weblate доступом только к переводам, не раскрывая ему при этом исходный код?

Для отделения переводов от исходного кода вы можете использовать подмодули git, при этом вы по-прежнему держите их под контролем версий.

1. Создайте репозиторий с вашими файлами перевода.
2. Добавьте его в качестве подмодуля к своему коду:

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

3. Свяжите Weblate с этим репозиторием, ему больше не нужен доступ к репозиторию, содержащему ваш исходный код.
4. Вы можете обновлять основной репозиторий переводами из Weblate, выполняя команду:

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

Для получения более подробной информации обратитесь к документации по подмодулям git.

Как мне проверить, правильно ли настроен мой Weblate?

Weblate включает в себя набор проверок конфигурации, которые можно увидеть в интерфейсе администратора, просто перейдите по ссылке *Отчёт о производительности* в интерфейсе администратора, или напрямую откройте адрес `/manage/performance/`.

Почему все коммиты от имени Weblate имеют электронную почту <noreply@weblate.org>?

Это умолчательное имя коммитящего, настроенное при создании компонента перевода. Вы в любое время можете изменить его в административной зоне сайта.

В любом случае, в качестве пользователя, сделавший перевод всё равно корректно записывается автор коммита (если это поддерживает нижележащая система контроля версий).

См.также:

Конфигурация компонента

1.9.2 Использование

Как мне отрецензировать переводы других переводчиков?

- Вы можете подписаться на любые изменения на вкладке *Уведомления*, а затем проверять вклады других переводчиков по мере их поступления по электронной почте.
- В нижней части представления перевода имеется инструмент рецензирования, с помощью которого можно просматривать переводы, выполненные другими пользователями с заданной даты.

Как мне предоставить обратную связь по исходной строке?

На контекстных вкладках, расположенных ниже перевода, вы можете использовать вкладку *Комментарии* для предоставления обратной связи по исходной строке или для её обсуждения с другими переводчиками.

Как я могу использовать при переводе уже существующие переводы?

- Используйте функциональность импорта для загрузки компендиума в качестве переводов, предложений или переводов, нуждающихся в рецензировании. Это лучший подход для одноразового перевода с использованием компендиума или аналогичной базы данных переводов.
- Вы можете настроить *tmserver* со всеми имеющимися у вас базами данных и позволить Weblate'у его использовать. Это хороший вариант, если вы хотите использовать его при переводе несколько раз.
- Другим вариантом является перенос всех связанных проектов на один экземпляр Weblate, что позволит автоматически подбирать переводы из других проектов.

См.также:

Машинный перевод, *Автоматические предложения*

Обновляет ли Weblate помимо самих переводов также и файлы перевода?

Weblate пытается свести к минимуму изменения в файлах перевода. Для некоторых форматов файлов это, к сожалению, может привести к переформатированию файла. Если вы хотите, чтобы файл был отформатирован по-вашему, пожалуйста, используйте для этого перехватчик pre-commit.

Для одноязычных файлов (смотрите раздел *Поддерживаемые форматы файлов*) Weblate может добавлять новые строки перевода, отсутствующие как в *шаблоне*, так и в реальных переводах. Однако он не выполняет автоматическую очистку устаревших строк, так как это может привести к неожиданным результатам. Если вы хотите её выполнять, пожалуйста, установите перехватчик pre-commit, который будет производить очистку в соответствии с вашими требованиями.

Weblate также не будет пытаться каким-либо образом обновлять двуязычные файлы, так что, если вам нужны файлы `po`, обновляемые из файлов `pot`, вам нужно делать это самостоятельно.

См.также:

Обработка репозитория скриптами

Откуда берутся определения языка и как я могу добавить свои собственные определения?

Базовый набор определений языка входит в состав Weblate и Translate-toolkit. Он охватывает более 150 языков и включает информацию о множественных формах или направлении текста.

Вы вольны определять в административном интерфейсе свои собственные языки, для этого вам просто необходимо предоставить о них необходимую информацию.

Может ли Weblate подсветить изменения в строке «на правку»?

Weblate поддерживает такую подсветку, однако, чтобы показать различия, ему нужны данные.

К примеру, для PO-файлов Gettext при обновлении PO-файлов необходимо в вызов команды **`msgmerge`** передать параметр `--previous`:

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

Для одноязычных переводов Weblate может найти предыдущую строку по её идентификатору, поэтому различия он показывает автоматически.

Почему Weblate до сих пор показывает старые строки перевода, хотя я уже обновил шаблон?

Weblate не пытается манипулировать файлами перевода никаким другим способом, кроме как разрешения переводчикам их переводить. Поэтому он также не обновляет переводимые файлы при изменении шаблона или исходного кода. Вам просто нужно сделать это вручную и внести изменения в репозиторий, после чего Weblate автоматически подхватит изменения.

Примечание: Обычно перед обновлением файлов перевода полезно влить изменения, сделанные в Weblate, в основной репозиторий, так как в противном случае вы, вероятно, столкнётесь с некоторыми конфликтами при слиянии.

Например, если вы используете PO-файлы gettext, вы можете обновить файлы перевода с помощью инструмента **msgmerge**:

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

В случае, если вы хотите выполнять такое обновление автоматически, вы можете установить надстройку *Обновлять PO-файлы до соответствия POT (msgmerge)*.

1.9.3 Решение проблем

Запросы иногда завершаются с ошибкой «слишком много открытых файлов» (“too many open files”)

Такое иногда случается, когда ваш Git-репозиторий слишком разрастается и у вас их очень много. Сжатие Git-репозиториев улучшит эту ситуацию.

Простейший способ это сделать — запустить следующий скрипт:

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

См.также:

DATA_DIR

При доступе к сайту я получаю ошибку «Плохой запрос (400)» (“Bad Request (400)”)

Скорее всего это вызвано неправильной настройкой параметра *ALLOWED_HOSTS*. Он должен содержать все имена хостов, с которых вы хотите получать доступ к вашему Weblate’у. Например:

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

См.также:

Настройка разрешённых хостов

Что означает сообщение «Есть больше файлов для одного языка (en)»?

Обычно оно возникает, когда у вас есть файл перевода для исходного языка. Weblate отслеживает исходные строки и резервирует для этого язык оригинала. Дополнительный файл для того же самого языка не обрабатывается.

- В случае необходимости перевода на исходный язык в настройках компонента необходимо изменить параметр *Исходный язык*.
- Если файл перевода для исходного языка не нужен, пожалуйста, удалите его из репозитория.
- Если же файл перевода для исходного языка необходим, но должен игнорироваться Weblate, пожалуйста, настройте для его исключения *языковой фильтр*.

1.9.4 Возможности

Поддерживает ли Weblate другие системы контроля версий, кроме Git и Mercurial?

В настоящее время Weblate не имеет родной поддержки ни для чего, кроме *Git* (с расширенной поддержкой *GitHub*, *Gerrit* и *Subversion*) и *Mercurial*, но вполне возможно дописать поддержку и других систем контроля версий.

Также для доступа к другим системам контроля версий вы можете использовать *Удалённые помощники Git* в Git'е .

Кроме того, Weblate поддерживает операции, не требующие системы контроля версий, смотрите раздел *Локальные файлы*.

Примечание: Для родной поддержки других систем контроля версий Weblate требует использования распределённых СКВ, и, вероятно, может быть настроен на работу с любой системой, помимо Git или Mercurial, но кто-то должен эту поддержку реализовать.

См.также:

Интеграция с системой контроля версий

Как Weblate сообщает о том, кто какую строку перевёл?

Каждое изменение, сделанное через Weblate, фиксируется в системе контроля версий под именем переводчика. Таким образом, каждое изменение имеет надлежащее авторство, и вы можете отследить его с помощью стандартных инструментов СКВ, которые вы используете для кода.

Кроме того, если формат файла перевода это поддерживает, он обновляет заголовки файлов и включает в них имя переводчика.

См.также:

`list_translators`, *Отчёт о прогрессе перевода*

Почему Weblate принудительно показывает все PO-файлы в одном дереве?

Weblate был спроектирован таким образом, что каждый PO-файл представляется одним компонентом. Это выгодно переводчиками, поскольку они знают, что они переводят на самом деле. Если вы считаете, что ваш проект должен переводиться как один компонент, подумайте об объединении этих po-файлов. Это облегчит жизнь даже переводчикам, не использующим Weblate.

Примечание: В случае большого спроса на эту функцию она может быть реализована в будущих версиях.

Почему Weblate использует такие коды языков, как sr_Latn или zh_Hant?

Это коды языков, определяемые **RFC 4646**, чтобы лучше показать, что это и в самом деле разные языки, вместо ранее ошибочно используемых модификаторов (для вариантов `@latin`) или кодов стран (для китайского языка).

Weblate по-прежнему понимает старые коды языков и сопоставляет их с текущими — например, `sr@latin` будет обрабатываться как `sr_Latn`, а `zh@CN` — как `zh_Hans`.

1.10 Поддерживаемые форматы файлов

Weblate поддерживает большинство форматов перевода, понимаемых translate-toolkit, однако понимание любого из них чем-нибудь да отличаются, поэтому могут возникнуть некоторые проблемы с форматами, которые были протестированы недостаточно тщательно.

См.также:

Форматы файлов, относящиеся к переводу

Примечание: При выборе формата файла для вашего приложения лучше придерживаться какого-нибудь устоявшегося формата для используемого вами инструментария/платформы. Таким образом, ваши переводчики в дополнение смогут использовать любые инструменты, к которым они привыкли, и вероятность внесения ими своего вклада в ваш проект будет выше.

1.10.1 Двуязычные и одноязычные форматы

Поддерживаются как одноязычные, так и двуязычные форматы. Двуязычные форматы хранят в одном файле два языка — исходный язык и перевод (типичными примерами являются *GNU gettext*, *XLIFF* или *Строки Apple iOS*). С другой стороны, одноязычные форматы идентифицируют строку по идентификатору, и каждый языковой файл содержит только сопоставление этих форматов с любым заданным языком (обычно это *Строковые ресурсы Android*). Некоторые форматы файлов могут использоваться и так, и эдак, подробнее смотрите ниже.

Для правильного использования одноязычных файлов Weblate требует доступа к файлу, содержащему полный список строк для перевода и их «перевод» на исходный язык — этот файл внутри Weblate называется *одноязычным базовым файлом*, хотя в вашей парадигме наименование этой сущности может отличаться.

Кроме того, этот рабочий процесс может быть расширен с помощью *файла промежуточного языка*, позволяющего включить в перевод строки, предоставленные разработчиками, но не использованными «как есть» в окончательных строках.

1.10.2 Автоматическое обнаружение

Weblate умеет автоматически обнаруживать некоторые широко распространённые форматы файлов, но это обнаружение может навредить вашей производительности и ограничено возможностями, специфичными для данного формата файла (к примеру, автоматическим добавлением новых переводов).

1.10.3 Возможности типов переводов

Возможности всех поддерживаемых форматов:

Формат	Языковость[1]	Множественные формы[2]	Комментарии[3]	Контекст[4]	Расположение[5]	Флаги[8]	Дополнительные состояния[6]
GNU gettext	двуязычный	да	да	да	да	да[9]	требуется правка
Одноязычный gettext	одноязычный	да	да	да	да	да[9]	требуется правка
XLIFF	оба	да	да	да	да	да[10]	требуется правка, одобрено
Файлы свойств Java	оба	нет	да	нет	нет	нет	
Файлы свойств GWT	одноязычный	да	да	нет	нет	нет	
Переводы Joomla	одноязычный	нет	да	нет	да	нет	
Файлы .ts Qt Linguist	оба	да	да	нет	да	да[10]	требуется правка
Строковые ресурсы Android	одноязычный	да	да[7]	нет	нет	да[10]	
Строки Apple iOS	двуязычный	нет	да	нет	нет	нет	
Строки PHP	одноязычный	нет[11]	да	нет	нет	нет	
JSON-файлы	одноязычный	нет	нет	нет	нет	нет	
JSON -файлы i18next	одноязычный	да	нет	нет	нет	нет	
JSON-файлы go-i18n	одноязычный	да	нет	нет	нет	нет	
Файл ARB	одноязычный	да	да	нет	нет	нет	
JSON-файлы WebExtension	одноязычный	да	да	нет	нет	нет	
.XML-файлы ресурсов	одноязычный	нет	да	нет	нет	да[10]	
Файлы CSV	одноязычный	нет	да	да	да	нет	требуется правка

continues on next page

Таблица 3 – продолжение с предыдущей страницы

Формат	Языковость[1]	Множественные формы[2]	Комментарии[3]	Контекст[4]	Расположение[5]	Флаги[8]	Дополнительные состояния[6]
Файлы YAML	одноязычный	нет	да	нет	нет	нет	
Файлы Ruby YAML	одноязычный	да	да	нет	нет	нет	
Файлы DTD	одноязычный	нет	нет	нет	нет	нет	
Flat XML	одноязычный	нет	нет	нет	нет	да[10]	
Файлы ресурсов Windows	одноязычный	нет	да	нет	нет	нет	
Excel Open XML	одноязычный	нет	да	да	да	нет	требуется правка
Файлы метаданных магазина приложений	одноязычный	нет	нет	нет	нет	нет	
Файлы субтитров	одноязычный	нет	нет	нет	да	нет	
Файлы HTML	одноязычный	нет	нет	нет	нет	нет	
Формат OpenDocument	одноязычный	нет	нет	нет	нет	нет	
Формат IDML	одноязычный	нет	нет	нет	нет	нет	
Переводы в INI-файлах	одноязычный	нет	нет	нет	нет	нет	
Переводы в INI-файлах Inno Setup	одноязычный	нет	нет	нет	нет	нет	

[1] Смотрите раздел *Двуязычные и одноязычные форматы*

[2] Множественные формы необходимы для правильной локализации строк, содержащих переменную количества.

[3] Комментарии могут быть использованы для указания дополнительной информации о строке для перевода.

[4] Контекст используется для различения одинаковых строк, используемых в разных областях (например, *Sun* может означать как сокращённое название дня «воскресенье» — «Sunday» — так и название нашей ближайшей звезды).

[5] Расположение строки в исходном коде может помочь опытным переводчикам понять предназначение этой строки.

[8] Смотрите раздел *Настройка поведения*

[6] Дополнительные состояния, поддерживаемые форматом файла в дополнение к состояниям «Не переведено» и «Переведено».

[9] В качестве флагов используются комментарии типа gettext.

[10] Для всех форматов на основе XML флаги извлекаются из нестандартного атрибута `weblate-flags`. Кроме того, флаг `max-length:N` поддерживается через атрибут `maxwidth`, как определено в стандарте XLIFF, смотрите раздел *Указание флагов перевода*.

[7] Комментарий XML, размещённый перед элементом `<string>`, трактуется как комментарий разработчика.

[11] Множественные формы поддерживаются только для Laravel, который использует для их определения внутристрочный синтаксис, смотрите документ Локализация в Laravel.

1.10.4 GNU gettext

Наиболее широко используемый формат для перевода свободного программного обеспечения. Этот формат Weblate поддержал самым первым, и он до сих пор имеет самую лучшую поддержку.

Контекстная информация, хранящаяся в файле, поддерживается настройкой его заголовков или его привязкой к соответствующим исходным файлам.

Двуязычный PO-файл gettext обычно выглядит следующим образом:

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

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

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

Типичная *настройка компонента* в Weblate	
Маска файла	`po/*.po`
Одноязычный базовый файл	*Пусто*
Шаблон для новых переводов	`po/messages.pot`
Формат файла	*PO-файл gettext*

См.также:

Перевод программ с помощью GNU Gettext, *Перевод документации с помощью Sphinx*, Gettext в Википедии, PO-файлы, *Обновление переменной ALL_LINGUAS в файле «configure»*, *Настройка вывода gettext*, *Обновление файла LINGUAS*, *Генерирование MO-файлов*, *Обновлять PO-файлы до соответствия POT (msgmerge)*

Одноязычный gettext

Некоторые проекты решают использовать gettext в качестве одноязычного формата — в их исходном коде присутствуют только идентификаторы и строка должна быть переведена на все языки, в том числе и на английский. Такая конфигурация поддерживается, хотя для этого и надо явно выбирать этот формат файлов при импорте компонентов в Weblate.

Одноязычный PO-файл gettext обычно выглядит следующим образом:

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

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

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

А базовый языковой файл — вот таким образом:

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

(continues on next page)

(продолжение с предыдущей страницы)

```
msgstr "Monday"

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

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

Типичная *настройка компонента* в Weblate	
Маска файла	`po/*.po`
Одноязычный базовый файл	`po/en.po`
Шаблон для новых переводов	`po/messages.pot`
Формат файла	*PO-файл gettext (одноязычный)*

1.10.5 XLIFF

Формат на основе XML, созданный для стандартизации файлов перевода, но в конечном итоге ставший просто ещё одним из многочисленных стандартов в этой области.

Формат файла XML для обмена локализациями (XLIFF) обычно используется как двуязычный, но Weblate поддерживает его использование и в качестве одноязычного.

См.также:

Спецификация *формата файла XML для обмена локализациями (XLIFF)*

Состояния перевода

Изменено в версии 3.3: До выпуска 3.3 Weblate игнорировал атрибут состояния.

Атрибут состояния `state` в файле частично обрабатывается и отображается в Weblate в состояние «На правку» (следующие состояния используются для пометки строки на правку, если присутствует цель: `new`, `needs-translation`, `needs-adaptation`, `needs-l10n`). Если атрибут `state` отсутствует, то строка считается переведённой, если существует элемент `<target>`.

Если строка перевода имеет атрибут `approved="yes"`, она также будет импортирована в Weblate в состоянии «Одобрено», все остальные строки будут импортированы в состоянии «Ожидает рецензирования» (что соответствует спецификации XLIFF).

При сохранении Weblate добавляет эти атрибуты только если они действительно необходимы:

- Атрибут `state` добавляется только в том случае, если строка помечена на правку.
- Атрибут `approved` добавляется только в том случае, если строка была отрецензирована.
- В остальных случаях эти атрибуты не добавляются, но если они присутствуют, они обновляются.

Это означает, что при использовании формата XLIFF настоятельно рекомендуется включить в Weblate процесс рецензирования, чтобы видеть и изменять состояние одобрения строк.

Смотрите раздел *Выделенные рецензенты*.

Аналогично, при импорте таких файлов (в форме загрузки) в выпадающем списке *Обработка строк, отмеченных «На правку»* необходимо выбрать пункт *Импортировать как переведённое*.

Пробелы и переводы строк в файлах XLIFF

Как правило, тип или количество пробельных символов в XML-форматах не различаются. Если вы хотите их сохранить, вы должны добавить к строке флаг `xml:space="preserve"`.

Например:

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

Указание флагов перевода

С помощью атрибута `weblate-flags` вы можете указать дополнительные флаги перевода (смотрите раздел *Настройка поведения*). Также Weblate понимает атрибуты `maxwidth` и `font` из спецификации XLIFF:

```
<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>
```

Атрибут `font` разбирается в семейство шрифтов, размер и вес шрифта. Приведённый выше пример показывает использование всех трёх параметров, хотя к указанию обязательно только семейство шрифтов. Любые пробельные символы в наименовании семейства шрифтов преобразуются в подчёркивания, так что семейство `Source Sans Pro` становится семейством `Source_Sans_Pro`, поэтому при именовании группы шрифтов (смотрите раздел *Управление шрифтами*) имейте это в виду.

Типичная *настройка компонента* в Weblate для двуязычного XLIFF	
Маска файла	`localizations/*.xliff`
Одноязычный базовый файл	*Пусто*
Шаблон для новых переводов	`localizations/en-US.xliff`
Формат файла	*Файл перевода XLIFF*

Типичная *настройка компонента* в Weblate для одноязычного XLIFF	
Маска файла	`localizations/*.xliff`
Одноязычный базовый файл	`localizations/en-US.xliff`
Шаблон для новых переводов	`localizations/en-US.xliff`
Формат файла	*Файл перевода XLIFF*

См.также:

XLIFF в Википедии, XLIFF, Атрибут font в XLIFF 1.2, Атрибут maxwidth в XLIFF 1.2

1.10.6 Файлы свойств Java

Родной формат Java для переводов.

Файлы свойств Java обычно используются в качестве одноязычных переводов.

Weblate поддерживает варианты этого формата в кодировках ISO-8859-1, UTF-8 и UTF-16. Все они поддерживают сохранение всех символов юникода, просто в разных кодировках. В формате ISO-8859-1 используются юникодные экранирующие последовательности (например, `zkou\u0161ka`), остальные кодируют символы непосредственно в UTF-8 или UTF-16.

Примечание: Загрузка экранирующих последовательностей также работает в режиме UTF-8, поэтому, пожалуйста, будьте внимательны при выборе правильного набора кодировок в соответствии с потребностями вашего приложения.

Типичная *настройка компонента* в Weblate	
Маска файла	`src/app/Bundle_*.properties`
Одноязычный базовый файл	`src/app/Bundle.properties`
Шаблон для новых переводов	*Пусто*
Формат файла	*Свойства Java (ISO 8859-1)*

См.также:

Свойства Java в Википедии, Файлы свойств Mozilla и Java, *Форматирование файла свойств Java*, *Очистка файлов перевода*

1.10.7 Файлы свойств GWT

Родной формат GWT для переводов.

Файлы свойств GWT обычно используются в качестве одноязычных переводов.

Типичная *настройка компонента* в Weblate	
Маска файла	`src/app/Bundle_*.properties`
Одноязычный базовый файл	`src/app/Bundle.properties`
Шаблон для новых переводов	*Пусто*
Формат файла	*Свойства GWT*

См.также:

Руководство по локализации GWT Файлы свойств Mozilla и Java, *Форматирование файла свойств Java*, *Очистка файлов перевода*

1.10.8 Переводы в INI-файлах

Добавлено в версии 4.1.

Формат INI-файла для переводов.

Переводы в INI-файлах обычно используются в качестве одноязычных переводов.

Типичная *настройка компонента* в Weblate	
Маска файла	`language/*.ini`
Одноязычный базовый файл	`language/en.ini`
Шаблон для новых переводов	*Пусто*
Формат файла	*INI-файл*

См.также:

INI-файлы, *Переводы Joomla*, *Переводы в INI-файлах Inno Setup*

1.10.9 Переводы в INI-файлах Inno Setup

Добавлено в версии 4.1.

Формат INI-файлов Inno Setup для переводов.

Переводы в INI-файлах Inno Setup обычно используются в качестве одноязычных переводов.

Примечание: Единственным заметным отличием от *обычных переводов в INI-файлах* является поддержка заполнителей `%n` и `%t` для символов переноса строки и табуляции.

Типичная *настройка компонента* в Weblate	
Маска файла	`language/*.islu`
Одноязычный базовый файл	`language/en.islu`
Шаблон для новых переводов	*Пусто*
Формат файла	*INI-файл Inno Setup*

Примечание: На текущий момент поддерживаются только юникодные файлы (`.islu`), файлы в кодировке ANSI (`.isl`) в настоящее время не поддерживаются.

См.также:

INI-файлы, *Переводы Joomla*, *Переводы в INI-файлах*

1.10.10 Переводы Joomla

Добавлено в версии 2.12.

Родной формат Joomla для переводов.

Переводы Joomla обычно используются в качестве одноязычных переводов.

Типичная *настройка компонента* в Weblate	
Маска файла	`language/*/com_foobar.ini`
Одноязычный базовый файл	`language/en-GB/com_foobar.ini`
Шаблон для новых переводов	*Пусто*
Формат файла	*Языковой файл Joomla*

См.также:

Спецификация языковых файлов Joomla, Файлы свойств Mozilla и Java, *Переводы в INI-файлах*, *Переводы в INI-файлах Inno Setup*

1.10.11 Файлы .ts Qt Linguist

Формат перевода, используемый в приложениях на основе Qt.

Файлы Qt Linguist используются и как двуязычные, так и одноязычные переводы.

Типичная *настройка компонента* в Weblate при использовании формата как двуязычного	
Маска файла	`i18n/app.*.ts`
Одноязычный базовый файл	*Пусто*
Шаблон для новых переводов	`i18n/app.ru.ts`
Формат файла	*Файл перевода Qt Linguist*

Типичная *настройка компонента* в Weblate при использовании формата как одноязычного	
Маска файла	`i18n/app.*.ts`
Одноязычный базовый файл	`i18n/app.en.ts`
Шаблон для новых переводов	`i18n/app.en.ts`
Формат файла	*Файл перевода Qt Linguist*

См.также:

Руководство по Qt Linguist, Qt .ts, *Двуязычные и одноязычные форматы*

1.10.12 Строковые ресурсы Android

Специфичный для Android формат файла для перевода приложений.

Строковые ресурсы Android являются одноязычными, *Одноязычный базовый файл* хранится в другом месте, нежели остальные строки — в файле `res/values/strings.xml`.

Типичная *настройка компонента* в Weblate	
Маска файла	`res/values-*/strings.xml`
Одноязычный базовый файл	`res/values/strings.xml`
Шаблон для новых переводов	*Пусто*
Формат файла	*Строковый ресурс Android*

См.также:

Документация по строковым ресурсам Android, Строковые ресурсы Android

---

Примечание: Структуры строкового массива Android, *string-array*, в настоящее время не поддерживаются. Чтобы это обойти, вы можете разбить свои строковые массивы на части:

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

станет:

```
<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>
```

Элемент *string-array*, указывающий на элементы *string*, должен храниться в другом файле и не должен быть доступен для перевода.

Помочь в подготовке существующих файлов strings.xml и переводов вам может вот этот скрипт: https://gist.github.com/paour/11291062

1.10.13 Строки Apple iOS

Специфичный для Apple формат файла для перевода приложений, используемый для переводов приложений как для iOS, так и для iPhone'a/iPad'a.

Строки Apple iOS обычно используются в качестве двуязычных переводов.

Типичная *настройка компонента* в Weblate	
Маска файла	`Resources/*.lproj/Localizable.strings`
Одноязычный базовый файл	`Resources/en.lproj/Localizable.strings` or `Resources/Base.lproj/Localizable.strings`
Шаблон для новых переводов	*Пусто*
Формат файла	*Строки iOS (UTF-8)*

См.также:

Документация Apple по «строковым файлам», Строки Mac OSX

1.10.14 Строки PHP

Переводы PHP обычно одноязычные, поэтому рекомендуется указывать базовый файл со строками на (как это чаще всего бывает) английском языке.

Пример файла:

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

Типичная *настройка компонента* в Weblate	
Маска файла	`lang/*/texts.php`
Одноязычный базовый файл	`lang/en/texts.php`
Шаблон для новых переводов	`lang/en/texts.php`
Формат файла	*Строки PHP*

Строки Laravel PHP

Изменено в версии 4.1.

Файлы локализации Laravel PHP также поддерживают множественные формы:

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

См.также:

PHP, Локализация в Laravel

1.10.15 JSON-файлы

Добавлено в версии 2.0.

Изменено в версии 2.16: Начиная с Weblate версии 2.16 и translate-toolkit версии не ниже 2.2.4 также поддерживаются JSON-файлы с вложенной структурой.

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

Формат JSON используется в основном для перевода приложений, реализованных на JavaScript.

В настоящее время Weblate поддерживает несколько вариантов переводов в JSON:

- Simple key / value files, used for example by *vue-i18n* or *react-intl*.
- Файлы с вложенными ключами.
- *JSON -файлы i18next*
- *JSON-файлы go-i18n*
- *JSON-файлы WebExtension*
- *Файл ARB*

Переводы JSON обычно одноязычные, поэтому рекомендуется указывать базовый файл со строками на (как это чаще всего бывает) английском языке.

Пример файла:

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

Также поддерживается вложенная структура файлов (условия поддержки смотрите выше), такой файл может выглядеть следующим образом:

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

Подсказка: The *JSON file* and *JSON nested structure file* can both handle same type of files. The only difference between them is when adding new strings. The nested variant tries to parse the key and insert the new string into the matching structure.

Типичная *настройка компонента* в Weblate	
Маска файла	`langs/translation-*.json`
Одноязычный базовый файл	`langs/translation-en.json`
Шаблон для новых переводов	*Пусто*
Формат файла	*Файл вложенной структуры JSON*

См.также:

JSON, *Настройка вывода в JSON*, *Очистка файлов перевода*,

1.10.16 JSON -файлы i18next

Изменено в версии 2.17: Начиная с Weblate версии 2.17 и translate-toolkit версии не ниже 2.2.5 также поддерживаются JSON-файлы i18next с множественными формами.

i18next — это фреймворк интернационализации, написанный на и для JavaScript. Weblate поддерживает его файлы локализации с такими возможностями, как формы множественного числа.

Переводы i18next обычно одноязычные, поэтому рекомендуется указывать базовый файл со строками на (как это чаще всего бывает) английском языке.

Примечание: Weblate поддерживает формат JSON-файла i18next версии v3. Версии v2 и v1 в основном совместимы, за исключением того, как они работают с формами множественного числа.

Пример файла:

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

Типичная *настройка компонента* в Weblate	
Маска файла	`langs/*.json`
Одноязычный базовый файл	`langs/en.json`
Шаблон для новых переводов	*Пусто*
Формат файла	*JSON-файл i18next*

См.также:

JSON, Формат JSON i18next, *Настройка вывода в JSON*, *Очистка файлов перевода*

1.10.17 JSON-файлы go-i18n

Добавлено в версии 4.1.

Переводы go-i18n обычно одноязычные, поэтому рекомендуется указывать базовый файл со строками на (как это чаще всего бывает) английском языке.

Примечание: Weblate поддерживает формат JSON-файла go-i18n версии v1, для плоских JSON-файлов используйте формат *JSON-файлы*. Формат v2 с хэшами в настоящее время не поддерживается.

Типичная *настройка компонента* в Weblate	
Маска файла	`langs/*.json`
Одноязычный базовый файл	`langs/en.json`
Шаблон для новых переводов	*Пусто*
Формат файла	*JSON -файл go-i18n*

См.также:

JSON, go-i18n, *Настройка вывода в JSON*, *Очистка файлов перевода*,

1.10.18 Файл ARB

Добавлено в версии 4.1.

Переводы ARB обычно одноязычные, поэтому рекомендуется указывать базовый файл со строками на (как это чаще всего бывает) английском языке.

Типичная *настройка компонента* в Weblate	
Маска файла	`lib/l10n/intl_*.arb`
Одноязычный базовый файл	`lib/l10n/intl_en.arb`
Шаблон для новых переводов	*Пусто*
Формат файла	*Файл ARB*

См.также:

JSON, Спецификация пакета ресурсов приложения, Интернационализация приложений Flutter, *Настройка вывода в JSON*, *Очистка файлов перевода*

1.10.19 JSON-файлы WebExtension

Добавлено в версии 2.16: Поддерживаются с версии Weblate 2.16 и translate-toolkit версии не ниже 2.2.4.

Формат файла, используемый при переводе расширений для Mozilla Firefox или Google Chromium.

Примечание: Хотя этот формат называется JSON, его спецификация позволяет включать комментарии, которые не являются частью спецификации JSON. В настоящее время Weblate не поддерживает файл с комментариями.

Пример файла:

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

Типичная *настройка компонента* в Weblate	
Маска файла	`_locales/*/messages.json`
Одноязычный базовый файл	`_locales/en/messages.json`
Шаблон для новых переводов	*Пусто*
Формат файла	*JSON-файл WebExtension*

См.также:

JSON, Google chrome.i18n, Интернационализация расширений Mozilla

1.10.20 .XML-файлы ресурсов

Добавлено в версии 2.3.

В .XML-файле ресурса (.resx) применяется формат одноязычного XML-файла, используемый в приложениях Microsoft .NET. Он взаимозаменяем с форматом .resw, который использует синтаксис, идентичный синтаксису .resx.

Типичная *настройка компонента* в Weblate	
Маска файла	`Resources/Language.*.resx`
Одноязычный базовый файл	`Resources/Language.resx`
Шаблон для новых переводов	*Пусто*
Формат файла	*Файл ресурсов .NET*

См.также:

Файлы ресурсов .NET (.resx), *Очистка файлов перевода*

1.10.21 Файлы CSV

Добавлено в версии 2.4.

Файлы CSV могут содержать простой список исходных строк и переводов. Weblate поддерживает следующие файлы:

- Файлы с заголовком, определяющим поля (источник, перевод, местоположение, …). Это рекомендуемый подход, так как он наименее подвержен ошибкам.
- Файлы с двумя полями — источником и переводом (в этом порядке), для этого в качестве формата файла выберите *Простой CSV файл*
- Файлы с полями, определёнными translate-toolkit: location, source, target, ID, fuzzy, context, translator_comments, developer_comments

Предупреждение: Формат CSV в настоящее время автоматически определяет диалект файла CSV. В некоторых случаях автоматическое определение может не сработать, и вы получите смешанные результаты. Особенно это касается файлов CSV с символами новых строк внутри значений. В качестве обходного пути рекомендуется не использовать кавычки.

Пример файла:

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

Типичная *настройка компонента* в Weblate	
Маска файла	`locale/*.csv`
Одноязычный базовый файл	*Пусто*
Шаблон для новых переводов	`locale/en.csv`
Формат файла	*Файл CSV*

См.также:

CSV

1.10.22 Файлы YAML

Добавлено в версии 2.9.

Простые файлы YAML со строковыми ключами и значениями. Также Weblate извлекает строки из списков и словарей.

Пример файла YAML:

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

Типичная *настройка компонента* в Weblate	
Маска файла	`translations/messages.*.yml`
Одноязычный базовый файл	`translations/messages.en.yml`
Шаблон для новых переводов	*Пусто*
Формат файла	*Файл YAML*

См.также:

YAML, *Файлы Ruby YAML*

1.10.23 Файлы Ruby YAML

Добавлено в версии 2.9.

Файлы YAML локализации Ruby с языком в качестве корневого узла.

Пример файла YAML локализации Ruby:

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

Типичная *настройка компонента* в Weblate	
Маска файла	`translations/messages.*.yml`
Одноязычный базовый файл	`translations/messages.en.yml`
Шаблон для новых переводов	*Пусто*
Формат файла	*Файл Ruby YAML*

См.также:

YAML, *Файлы YAML*

1.10.24 Файлы DTD

Добавлено в версии 2.18.

Пример DTD-файла:

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

Типичная *настройка компонента* в Weblate	
Маска файла	`locale/*.dtd`
Одноязычный базовый файл	`locale/en.dtd`
Шаблон для новых переводов	*Пусто*
Формат файла	*Файл DTD*

См.также:

Формат DTD Mozilla

1.10.25 Плоские XML-файлы

Добавлено в версии 3.9.

Пример плоского XML-файла:

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

Типичная *настройка компонента* в Weblate	
Маска файла	`locale/*.xml`
Одноязычный базовый файл	`locale/en.xml`
Шаблон для новых переводов	*Пусто*
Формат файла	*Плоский XML-файл*

См.также:

Плоские XML-файлы

1.10.26 Файлы ресурсов Windows

Изменено в версии 4.1: Поддержка файлов ресурсов Windows была переписана.

Примечание: Поддержка этого формата в настоящее время находится в бета-режиме, приветствуются обратная связь от тестирования.

Пример файла ресурсов Windows:

```
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
```

Типичная *настройка компонента* в Weblate	
Маска файла	`lang/*.rc`
Одноязычный базовый файл	`lang/en-US.rc`
Шаблон для новых переводов	`lang/en-US.rc`
Формат файла	*Файл RC*

См.также:

Файлы ресурсов Windows

1.10.27 Файлы метаданных магазина приложений

Добавлено в версии 3.5.

Метаданные, используемые для публикации приложений в различных магазинах приложений, могут быть переведены. В настоящее время совместимы следующие инструменты:

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

Метаданные состоят из нескольких текстовых файлов, которые Weblate представляет для перевода в виде отдельных строк.

Типичная *настройка компонента* в Weblate	
Маска файла	`fastlane/android/metadata/*`
Одноязычный базовый файл	`fastlane/android/metadata/en-US`
Шаблон для новых переводов	`fastlane/android/metadata/en-US`
Формат файла	*Файлы метаданных магазина приложений*

1.10.28 Файлы субтитров

Добавлено в версии 3.7.

Weblate может переводить различные файлы субтитров:

- Файл субтитров SubRip (`*.srt`)
- Файл субтитров MicroDVD (`*.sub`)
- Файл субтитров Advanced SubStation Alpha (`*.ass`)
- Файл субтитров SubStation Alpha (`*.ssa`)

Типичная *настройка компонента* в Weblate	
Маска файла	`path/*.srt`
Одноязычный базовый файл	`path/en.srt`
Шаблон для новых переводов	`path/en.srt`
Формат файла	*Файл субтитров SubRip*

См.также:

Субтитры

1.10.29 Excel Open XML

Добавлено в версии 3.2.

Можно импортировать и экспортировать файлы Excel Open XML (.xlsx).

Загружая для перевода файлы XLSX, имейте в виду, что учитывается только активный рабочий лист, и должны быть хотя бы колонки с названием `source` (содержащая исходную строку) и `target` (содержащая перевод). Дополнительно может быть колонка с названием `context` (содержащая путь к контексту строки перевода). Если вы используете скачанный XLSX для экспорта переводов в рабочую книгу Excel, вы уже получите файл в правильном формате.

1.10.30 Файлы HTML

Добавлено в версии 4.1.

Примечание: Поддержка этого формата в настоящее время находится в бета-режиме, приветствуются обратная связь от тестирования.

Переводимое содержимое извлекается из HTML-файлов и предлагается для перевода.

См.также:

HTML

1.10.31 Формат OpenDocument

Добавлено в версии 4.1.

Примечание: Поддержка этого формата в настоящее время находится в бета-режиме, приветствуются обратная связь от тестирования.

Переводимое содержимое извлекается из файлов OpenDocument и предлагается для перевода.

См.также:

Формат OpenDocument

1.10.32 Формат IDML

Добавлено в версии 4.1.

Примечание: Поддержка этого формата в настоящее время находится в бета-режиме, приветствуются обратная связь от тестирования.

Переводимое содержимое извлекается из файлов языка разметки Adobe InDesign и предлагается для перевода.

1.10.33 Другие

Большинство форматов, которые поддерживают сериализацию и которые поддерживает translate-toolkit, могут быть легко поддержаны, но они (пока) не прошли никакого тестирования. В большинстве случаев, чтобы скрыть различия в поведении различных хранилищ translate-toolkit, для Weblate требуется тонкая прослойка.

См.также:

Форматы файлов, относящиеся к переводу

1.10.34 Добавление новых переводов

Изменено в версии 2.18: В версиях до 2.18 поведение добавления новых переводов зависело от формата файла.

Weblate может автоматически начинать новый перевод для всех форматов файлов.

Некоторые форматы ожидают, что они будут начинаться с пустого файла и в него будут включены только переведённые строки (например, *Строковые ресурсы Android*), в то время как другие ожидают, что в файле будут присутствовать все ключи (например, *GNU gettext*). В некоторых ситуациях это на самом деле зависит не от формата, а от фреймворка, который вы используете для работы с файлами перевода (например, с *файлами JSON*).

При указании в *конфигурации компонента* файла *шаблона нового перевода* Weblate будет использовать его для начинания новых переводов. Все существующие переводы при этом будут из файла удалены.

Если поле *Шаблон для новых переводов* на заполнено, а формат файла его требует, создаётся пустой файл, в который добавляются новые строки после их перевода.

Параметр *Стиль кода языка* позволяет вам настраивать код языка, используемый в именах сгенерированных файлов:

Умолчание, основанное на формате файла Зависит от формата файла, для большинства форматов используется стиль POSIX.

Стиль POSIX, использующий подчёркивание как разделитель Обычно используется gettext'ом и связанными с ним инструментами, выдаёт коды языков вида *ru_RU*.

Стиль POSIX, использующий подчёркивание как разделитель и включающий код страны Код языка в стиле POSIX, включающий код страны, даже если в этом нет необходимости (например, *cs_CZ*).

Стиль BCP, использующий в качестве разделителя дефис Обычно используется на веб-платформах, выдаёт коды языков вида *ru-RU*.

Стиль BCP, использующий в качестве разделителя дефис и включающий код страны Код языка в стиле BCP, включающий код страны, даже если в этом нет необходимости (например, *cs-CZ*).

Стиль Android Используется только в приложениях для Android, выдаёт коды языков вида *ru-rRU*.

Стиль Java Используется Java — в основном BCP с устаревшими кодами для китайского языка.

Примечание: Weblate распознаёт любой из этих стилей при разборе файлов перевода, вышеуказанные настройки влияют только на то, в каком стиле создавать новые файлы.

1.10.35 Строки только для чтения

Добавлено в версии 3.10.

Строки из файлов перевода, доступные только для чтения, будут включены, но не могут быть отредактированы в Weblate. Поддержка этой возможности уже встроена в некоторые форматы (*XLIFF* и *Строковые ресурсы Android*), а в других может быть сэмулирована добавлением к строке флага `read-only` — смотрите раздел *Настройка поведения*.

1.11 Интеграция с системой контроля версий

В настоящее время Weblate в качестве систем контроля версий поддерживает *Git* (с расширенной поддержкой *GitHub*, *Gerrit* и *Subversion*) и *Mercurial*.

1.11.1 Доступ к репозиториям

Репозиторий системы контроля версий, который вы хотите использовать, должен быть доступен для Weblate. При использовании общедоступного репозитория вам просто надо указать правильный URL-адрес (например, `https://github.com/WeblateOrg/weblate.git`), но для частных репозиториев или для URL-адресов отправки изменений настройка будет более сложной и требует аутентификации.

Доступ к репозиториям из Hosted Weblate

Для Hosted Weblate существует выделенный пользователь для отправки изменений, зарегистрированный на GitHub, Bitbucket, Codeberg и GitLab (с именем пользователя *weblate*, известный также под именем *Weblate push user*). Вам необходимо добавить этого пользователя в качестве сотрудника и дать ему в вашем репозитории соответствующее разрешение (для клонирования подойдет только чтение, отправки изменений требуется запись). В зависимости от сервиса и настроек вашей организации, это произойдёт или немедленно, или потребует подтверждения со стороны Weblate.

Приглашения на GitHub принимаются автоматически в течение пяти минут, на другие сервисы может потребоваться ручная обработка, поэтому, пожалуйста, наберитесь терпения.

После того, как пользователь *weblate* будет добавлен, вы сможете настроить *Репозиторий исходного кода* и *URL для отправки в репозиторий* по протоколу SSH (например, `git@github.com:WeblateOrg/weblate.git`).

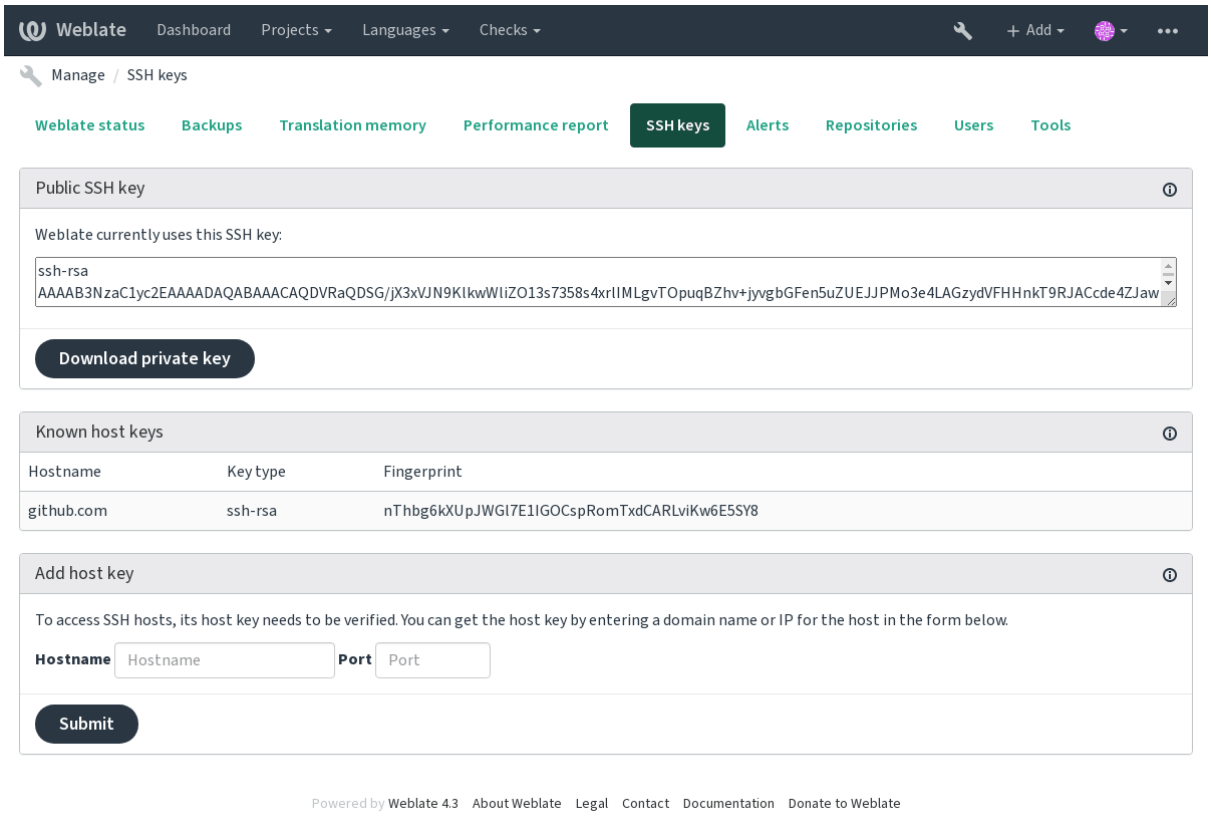
Репозитории по SSH

Наиболее часто используемый метод доступа к частным репозиториям основан на SSH. Чтобы получить таким образом доступ к вышестоящему репозиторию, авторизуйте публичный SSH-ключ Weblate (смотрите раздел *SSH-ключ Weblate*).

Предупреждение: На GitHub каждый ключ может быть добавлен только в один репозиторий, смотрите разделы *Репозитории GitHub* и *Доступ к репозиториям из Hosted Weblate*.

Weblate также сохраняет при первом подключении отпечаток ключа хоста и не сможет подключиться к хосту, если позже ключ будет изменён (смотрите раздел *Проверка SSH-ключей хоста*).

В случае необходимости его изменения это можно сделать из интерфейса администратора Weblate:



SSH-ключ Weblate

Открытый ключ Weblate виден всем пользователям, просматривающим страницу *О Weblate*.

Администраторы могут сгенерировать или посмотреть публичный ключ, который в настоящее время использует Weblate (из блока *SSH-ключи*) на странице входа в интерфейс администратора.

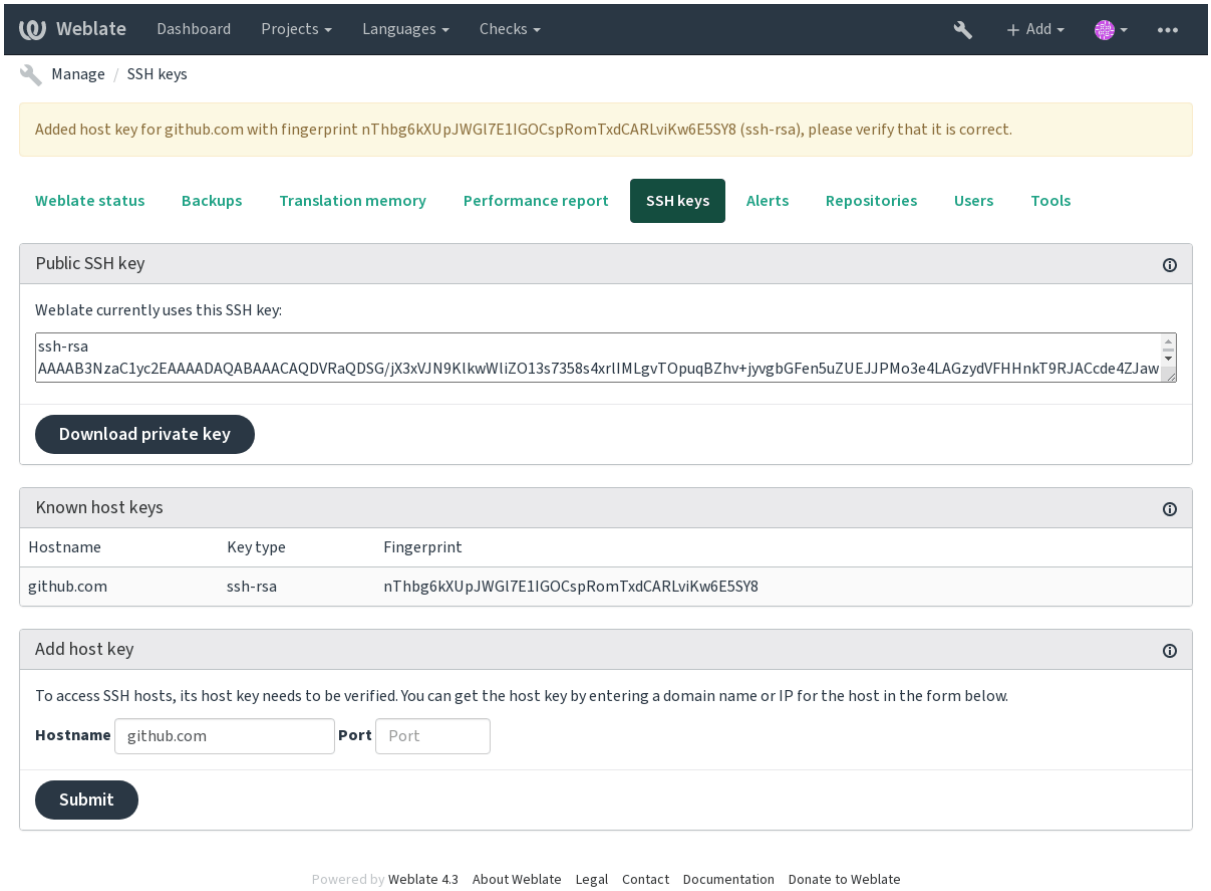
Примечание: Соответствующий закрытый SSH-ключ в настоящее время не может быть запаролен, поэтому убедитесь, что он хорошо защищён.

Подсказка: Сделайте резервную копию сгенерированного закрытого SSH-ключа Weblate.

Проверка SSH-ключей хоста

Weblate автоматически запоминает SSH-ключи хостов при первом к ним обращении и запоминает их для дальнейшего использования.

В случае, если вы хотите проверить их перед подключением к репозиторию, проверьте SSH-ключи хостов серверов, к которым вы собираетесь получить доступ через пункт *Добавить ключ хоста*, в том же разделе административного интерфейса. Введите имя хоста, к которому вы собираетесь получить доступ (например, `gitlab.com`), и нажмите *Подтвердить*. Убедитесь, что его отпечаток совпадает с отпечатком добавленного вами сервера. Они отображаются в сообщении о подтверждении:



Репозитории GitHub

Доступ по SSH возможен (смотрите раздел. *Репозитории по SSH*), но в случае, если вам необходимо получить доступ к более чем одному репозиторию, вы столкнётесь с ограничением GitHub'а по разрешённому использованию ключей SSH (поскольку один ключ может быть использован только для одного репозитория).

В случае, если *ветка для отправки* не задана, проект форкается и изменения отправляются через форк. В случае если она задана, изменения отправляются в вышестоящий репозиторий и выбранную ветку.

Для небольших развёртываний используйте HTTPS-аутентификацию с персональным токеном доступа и вашей GitHub'овской учётной записью, смотрите документ Создание токена доступа для использования из командной строки.

Для более крупных установок обычно лучше создать для Weblate специального пользователя, назначить ему сгенерированный в Weblate открытый SSH ключ (смотрите раздел *SSH-ключ Weblate*) и предоставить ему доступ ко всем репозиториям, которые вы хотите переводить. Этот подход также используется в Hosted Weblate, для этого есть выделенный пользователь *weblate*.

См.также:

Доступ к репозиториям из Hosted Weblate

Внутренние URL-адреса Weblate

Для совместного использования одного репозитория между различными компонентами можно использовать специальный URL-адрес вида `weblate://проект/компонент`. Таким образом, компонент будет совместно использовать конфигурацию репозитория системы контроля версий с соответствующим компонентом (в примере это `проект/компонент`).

Weblate при создании компонента автоматически корректирует URL-адрес репозитория, когда находит компонент с подходящей настройкой репозитория. Вы можете переопределить эту настройку на последнем этапе конфигурации компонента.

Причины для использования такого поведения:

- Экономия дискового пространства на сервере, репозиторий хранится всего один.
- Ускорение обновлений, обновляется только один репозиторий.
- Существует только один экспортируемый репозиторий с переводами Weblate (смотрите раздел *Экспортер Git*).
- Некоторые надстройки могут работать с несколькими компонентами, совместно использующими один репозиторий, например, надстройка *Склеивание Git-коммитов*.

Репозитории по HTTPS

Для доступа к защищённым HTTPS репозиториям включите в URL-адрес имя пользователя и пароль. Не волнуйтесь, Weblate обрежет эту информацию, когда адрес будет показываться пользователям (даже если им разрешено видеть адрес репозитория).

Например, URL-адрес GitHub с добавленной аутентификацией может выглядеть следующим образом: `https://пользователь:ваш_токен_доступа@github.com/WeblateOrg/weblate.git`.

Примечание: Если имя вашего пользователя или пароль содержат специальные символы, то они должны быть закодированы для использования в URL, например `https://user%40example.com:%24password%23@bitbucket.org/`….

Использование прокси

Если вам необходимо получить доступ к репозиториям системы контроля версий по HTTP/HTTPS через прокси-сервер, настройте СКВ на его использование.

Это можно сделать с помощью установки переменных окружения `http_proxy`, `https_proxy` и `all_proxy` (как описано в документации cURL) или принудительно включив его использование в конфигурации системы контроля версий, например:

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

Примечание: Настройка прокси должна выполняться под пользователем, запустившим Weblate (смотрите также раздел *Разрешения файловой системы*) и с `HOME=$DATA_DIR/home` (смотрите *`DATA_DIR`*), иначе Git, запускаемый Weblate, не будет его использовать.

См.также:

Страница руководства по cURL, Документация по команде git config

1.11.2 Git

См.также:

Для получения информации о том, как получить доступ к различным типам репозиториев, смотрите раздел *Доступ к репозиториям*.

Git с принудительной отправкой (force push)

Ведёт себя точно так же, как и сам Git, с той лишь разницей, что он всегда выполняет принудительную отправку. Он предназначен только для случая использования для переводов отдельного репозитория.

> **Предупреждение:** Используйте с осторожностью, поскольку он легко может потерять коммиты в вашем репозитории.

Настройка конфигурации Git'а

Weblate выполняет все команды системы контроля версий с установкой `HOME=$DATA_DIR/home` (смотрите *DATA_DIR*), поэтому изменение конфигурации пользователя должно выполняться в `DATA_DIR/home/.git`.

Удалённые помощники Git

Вы также можете использовать удалённых помощников Git для дополнительной поддержки других систем управления версиями, но будьте готовы к отладке проблем, которые могут за этим последовать.

В настоящее время помощники для Bazaar и Mercurial доступны в отдельных GitHub'овских репозиториях: git-remote-hg и git-remote-bzr. Скачайте их вручную и поместите куда-нибудь в пути поиска (например, в `~/bin`). Убедитесь, что у вас установлены соответствующие системы контроля версий.

После их установки такие удалённые репозитории можно будет использовать при указании репозитория в Weblate.

Для клонирования проекта `gnuhello` из Launchpad с помощью Bazaar используйте:

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

Для клонирования репозитория `hello` с selenic.com через Mercurial используйте:

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

> **Предупреждение:** Неудобство использования удалённых помощников Git'а заключается в, на примере Mercurial, том, что удалённый помощник иногда создаёт новую оконечную фиксацию при обратном проталкивании изменений.

1.11.3 GitHub

Добавлено в версии 2.3.

Он добавляет с помощью API Github'a тонкий слой логики поверх *Git*, который позволяет отправлять изменения перевода в виде запросов на извлечение, вместо того, чтобы отправлять их непосредственно в репозиторий.

Git отправляет изменения непосредственно в репозиторий, в то время как *GitHub* создаёт запросы на извлечение. Для простого доступа к Git-репозиториям последние не нужны.

См.также:

Отправка изменений из Weblate'a

Отправка изменений в GitHub в виде запросов на извлечение

Если вы не хотите отправлять переводы непосредственно в репозиторий GitHub'a, их можно отправить в виде одного или нескольких запросов на извлечение.

You need to configure API credentials to make this work.

См.также:

`GITHUB_USERNAME`, `GITHUB_TOKEN`, `GITHUB_CREDENTIALS`

1.11.4 GitLab

Добавлено в версии 3.9.

Просто добавляет с помощью API GitLab'a тонкий слой логики поверх *Git*, который позволяет отправлять изменения перевода в виде запросов на слияние вместо того, чтобы отправлять их непосредственно в репозиторий.

Нет необходимости использовать его для доступа к Git-репозиториям, обычный *Git* работает так же, единственная разница заключается в том, как выполняется отправка изменений в репозиторий. С помощью *Git* изменения отправляются непосредственно в репозиторий, в то время как *GitLab* создаёт запрос на слияние.

См.также:

Отправка изменений из Weblate'a

Отправка изменений в GitLab в виде запросов на слияние

Если вы не хотите отправлять переводы непосредственно в репозиторий GitLab, их можно отправить в виде одного или нескольких запросов на слияние.

You need to configure API credentials to make this work.

См.также:

`GITLAB_USERNAME`, `GITLAB_TOKEN`, `GITLAB_CREDENTIALS`

1.11.5 Gerrit

Добавлено в версии 2.2.

Добавляет с помощью инструмента git-review тонкий слой логики поверх *Git*, который позволяет отправлять изменения перевода в виде запросов на рецензирование Gerrit, вместо того, чтобы отправлять их непосредственно в репозиторий.

В документации Gerrit подробно описана конфигурация, необходимая для настройки таких репозиториев.

1.11.6 Mercurial

Добавлено в версии 2.1.

Mercurial — это ещё одна система контроля версий, которую вы можете использовать в Weblate напрямую.

Примечание: Weblate должен работать с любой версией Mercurial'a, но иногда в его интерфейсе командной строки происходят несовместимые изменения, которые нарушают его интеграцию с Weblate.

См.также:

Для получения информации о том, как получить доступ к различным типам репозиториев, смотрите раздел *Доступ к репозиториям*.

1.11.7 Subversion

Добавлено в версии 2.8.

Для взаимодействия с хранилищем subversion Weblate использует команду git-svn. Это Perl-скрипт, который позволяет использовать subversion из клиента Git'a, позволяя пользователям поддерживать полный клон внутреннего хранилища и фиксировать в него локально.

Примечание: Weblate автоматически пытается определить компоновку хранилища Subversion — он поддерживает как прямые URL-адреса для веток, так и хранилища со стандартной компоновкой (каталоги branches/, tags/ и trunk/). Подробнее об этом можно прочитать в документации к git-svn. Если ваше хранилище не придерживается стандартной компоновки и вы столкнулись с ошибками, попробуйте включить в URL-адрес хранилища имя ветки, а саму ветку оставить пустой.

Изменено в версии 2.19: До этого существовала поддержка только стандартных компоновок хранилищ.

Учётные данные Subversion

Weblate ожидает, что вы заранее приняли сертификат и, если необходимо, свои учётные данные. Он будет искать возможность поместить их в каталог DATA_DIR. Примите сертификат, используя команду *svn* один раз с переменной окружения *$HOME*, установленной в каталог 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
```

См.также:

DATA_DIR

1.11.8 Локальные файлы

Добавлено в версии 3.8.

Weblate также может работать без удалённой системы контроля версий. Первоначальные переводы импортируются путём их загрузки. Позже вы можете заменить отдельные файлы через загрузку файлов или добавить строки перевода непосредственно из Weblate (в настоящее время этот функционал доступен только для одноязычных переводов).

В фоне Weblate создаст Git-репозиторий, и будет отслеживать все изменения. В случае, если вы позже решите использовать для хранения переводов систему контроля версий, у вас уже будет репозиторий внутри Weblate, на основе которого вы сможете настроить свою интеграцию.

1.12 REST API Weblate'a

Добавлено в версии 2.6: REST API доступен с версии Weblate 2.6.

API доступен по адресу `/api/` и основан на фреймворке Django REST. Вы можете использовать его напрямую или через *Клиент Weblate*.

1.12.1 Аутентификация и общие параметры

API публичного проекта доступен без аутентификации, хотя количество неаутентифицированных запросов сильно ограничено (по умолчанию до 100 запросов в день), так что аутентификацию всё равно рекомендуется использовать. Аутентификация использует токен, который вы можете получить в своём профиле. Используйте его в заголовке `Authorization`:

ANY /

Описанные здесь общее поведение запросов для API, заголовки, коды состояния и параметры применяются ко всем конечным точкам.

Параметры запроса

- **format** – Формат ответа (переопределяет заголовок Accept). Возможные значения зависят от настройки REST-фреймворка, по умолчанию поддерживаются `json` и `api`. Последний предоставляет для API интерфейс веб-браузера.

Заголовки запроса

- Accept – тип содержимого ответа зависит от заголовка Accept
- Authorization – необязательный токен для аутентификации

Заголовки ответа

- Content-Type – зависит от заголовка Accept запроса
- Allow – список разрешённых для объекта методов HTTP

JSON -объект ответа

- **detail** (*string*) – подробное описание сбоя (для кодов статусов HTTP, отличных от 200 OK)
- **count** (*int*) – общее количество элементов для списков объектов
- **next** (*string*) – URL-адрес следующей страницы для списков объектов
- **previous** (*string*) – URL-адрес предыдущей страницы для списков объектов
- **results** (*array*) – результаты для списков объектов
- **url** (*string*) – URL-адрес для доступа к этому ресурсу с помощью API
- **web_url** (*string*) – URL-адрес для доступа к этому ресурсу через веб-браузер

Коды статусов

- 200 OK – при правильной обработке запроса
- 400 Bad Request – при отсутствии параметров формы
- 403 Forbidden – в случае отказа в доступе
- 429 Too Many Requests – при превышении количества запросов из-за ограничения на количество запросов

Примеры аутентификации

Пример запроса:

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

Пример ответа:

```
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/"
}
```

Пример CURL:

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

Примеры передачи параметров

Для метода POST параметры могут быть указаны либо в виде представления формы (*application/x-www-form-urlencoded*), либо в JSON (*application/json*).

Пример запроса формы:

```
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
```

Пример JSON запроса:

```
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"}
```

Пример CURL:

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

Пример CURL с JSON:

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

Ограничение частоты запросов

Частота запросов к API ограничена; конфигурация по умолчанию ограничивает её до 100 запросов в день для анонимных пользователей и до 5000 запросов в час для аутентифицированных пользователей.

Это ограничение можно настроить в `settings.py`; подробнее о его настройке смотрите в разделе Throttling документации фреймворка Django REST.

О состоянии ограничения частоты запросов сообщается в следующих заголовках:

`X-RateLimit-Limit`	Ограничение на максимальное число выполняемых запросов
`X-RateLimit-Remaining`	Оставшийся лимит запросов
`X-RateLimit-Reset`	Количество секунд до сброса окна ограничения количества запросов

Изменено в версии 4.1: Добавлены заголовки состояния ограничения количества запросов.

1.12.2 Точка входа в API

GET /api/

Корневая точка входа в API.

Пример запроса:

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

Пример ответа:

```
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
```

(continues on next page)

(продолжение с предыдущей страницы)

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

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

1.12.3 Пользователи

Добавлено в версии 4.0.

GET /api/users/

Возвращает список пользователей, если у вас есть разрешения на просмотр пользователей для управления. Если их нет, то вы увидите только свои данные.

См.также:

Атрибуты объекта пользователя задокументированы в разделе *GET /api/users/(str:username)/*.

POST /api/users/

Создаёт нового пользователя.

Параметры

- **username** (*string*) – Имя пользователя
- **full_name** (*string*) – Полное имя пользователя
- **email** (*string*) – Электронная почта пользователя
- **is_superuser** (*boolean*) – Является ли пользователь суперпользователем? (необязательный)
- **is_active** (*boolean*) – Активен ли пользователь? (необязательный)

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

Возвращает информацию о пользователях.

Параметры

- **username** (*string*) – Имя пользователя

JSON -объект ответа

- **username** (*string*) – имя пользователя
- **full_name** (*string*) – полное имя пользователя
- **email** (*string*) – электронная почта пользователя
- **is_superuser** (*boolean*) – является ли пользователь суперпользователем
- **is_active** (*boolean*) – активен ли пользователь
- **date_joined** (*string*) – дата создания пользователя
- **groups** (*array*) – ссылка на связанные группы; смотрите описание *GET /api/groups/(int:id)/*

Пример данных в JSON:

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

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

Изменяет параметры пользователя.

Параметры

- **username** (*string*) – Имя пользователя

JSON -объект ответа

- **username** (*string*) – имя пользователя
- **full_name** (*string*) – полное имя пользователя
- **email** (*string*) – электронная почта пользователя
- **is_superuser** (*boolean*) – является ли пользователь суперпользователем
- **is_active** (*boolean*) – активен ли пользователь
- **date_joined** (*string*) – дата создания пользователя

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

Изменяет параметры пользователя.

Параметры

- **username** (*string*) – Имя пользователя

JSON -объект ответа

- **username** (*string*) – имя пользователя
- **full_name** (*string*) – полное имя пользователя
- **email** (*string*) – электронная почта пользователя
- **is_superuser** (*boolean*) – является ли пользователь суперпользователем
- **is_active** (*boolean*) – активен ли пользователь
- **date_joined** (*string*) – дата создания пользователя

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

Удаляет всю информацию о пользователе и помечает пользователя как неактивного.

Параметры

- **username** (*string*) – Имя пользователя

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

Связывает группы с пользователем.

Параметры

- **username** (*string*) – Имя пользователя

Параметры формы

- **string group_id** – Уникальный идентификатор группы

GET /api/users/(**str:** *username*)**/statistics/**
Список статистик пользователя.

Параметры

- **username** (*string*) – Имя пользователя

JSON -объект ответа

- **translated** (*int*) – Количество переводов пользователя
- **suggested** (*int*) – Количество предложений пользователя
- **uploaded** (*int*) – Количество загрузок пользователя
- **commented** (*int*) – Количество комментариев пользователя
- **languages** (*int*) – Количество языков, на которые пользователь может переводить

GET /api/users/(**str:** *username*)**/notifications/**
Список подписок пользователя.

Параметры

- **username** (*string*) – Имя пользователя

POST /api/users/(**str:** *username*)**/notifications/**
Связывает подписки с пользователем.

Параметры

- **username** (*string*) – Имя пользователя

JSON -объект запроса

- **notification** (*string*) – Название зарегистрированного уведомления
- **scope** (*int*) – Область действия уведомления из доступных вариантов
- **frequency** (*int*) – Выбор частоты уведомлений

GET /api/users/(**str:** *username*)**/notifications/**
int: *subscription_id***/** Возвращает связанную с пользователем подписку.

Параметры

- **username** (*string*) – Имя пользователя
- **subscription_id** (*int*) – Идентификатор зарегистрированного уведомления

PUT /api/users/(**str:** *username*)**/notifications/**
int: *subscription_id***/** Редактирует связанную с пользователем подписку.

Параметры

- **username** (*string*) – Имя пользователя
- **subscription_id** (*int*) – Идентификатор зарегистрированного уведомления

JSON -объект запроса

- **notification** (*string*) – Название зарегистрированного уведомления
- **scope** (*int*) – Область действия уведомления из доступных вариантов
- **frequency** (*int*) – Выбор частоты уведомлений

PATCH /api/users/(**str:** *username*)**/notifications/**
int: *subscription_id***/** Редактирует связанную с пользователем подписку.

Параметры

- **username** (*string*) – Имя пользователя
- **subscription_id** (*int*) – Идентификатор зарегистрированного уведомления

JSON -объект запроса

- **notification** (*string*) – Название зарегистрированного уведомления
- **scope** (*int*) – Область действия уведомления из доступных вариантов
- **frequency** (*int*) – Выбор частоты уведомлений

DELETE /api/users/(**str:** *username*)**/notifications/**
int: *subscription_id***/** Удаляет связанную с пользователем подписку.

Параметры

- **username** (*string*) – Имя пользователя
- **subscription_id** – Название зарегистрированного уведомления
- **subscription_id** – целое

1.12.4 Группы

Добавлено в версии 4.0.

GET /api/groups/
Возвращает список групп, если у вас есть разрешения на просмотр групп для управления. Если их нет, то вы можете видеть только те группы, в которые входит пользователь.

См.также:

Атрибуты объекта группы задокументированы в разделе *GET /api/groups/(int:id)/*.

POST /api/groups/
Создаёт новую группу.

Параметры

- **name** (*string*) – Название группы
- **project_selection** (*int*) – Группа проекта, выбранная из предложенных вариантов
- **language_selection** (*int*) – Группа языков, выбранная из предложенных вариантов

GET /api/groups/(**int:** *id*)**/**
Возвращает информацию о группе.

Параметры

- **id** (*int*) – Идентификатор группы

JSON -объект ответа

- **name** (*string*) – название группы
- **project_selection** (*int*) – целое число, соответствующее группе проектов
- **language_selection** (*int*) – целое число, соответствующее группе языков
- **roles** (*array*) – ссылка на связанные роли; смотрите описание *GET /api/roles/(int:id)/*
- **projects** (*array*) – ссылка на связанные проекты; смотрите описание *GET /api/projects/(string:project)/*

- **components** (*array*) – ссылка на связанные компоненты; смотрите описание *GET /api/components/(string:project)/(string:component)/*
- **componentlist** (*array*) – ссылка на связанный список компонентов; смотрите описание *GET /api/component-lists/(str:slug)/*

Пример данных в JSON:

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

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

Изменяет параметры группы.

Параметры

- **id** (*int*) – Идентификатор группы

JSON -объект ответа

- **name** (*string*) – название группы
- **project_selection** (*int*) – целое число, соответствующее группе проектов
- **language_selection** (*int*) – целое число, соответствующее группе языков

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

Изменяет параметры группы.

Параметры

- **id** (*int*) – Идентификатор группы

JSON -объект ответа

- **name** (*string*) – название группы
- **project_selection** (*int*) – целое число, соответствующее группе проектов
- **language_selection** (*int*) – целое число, соответствующее группе языков

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

Удаляет группу.

Параметры

- **id** (*int*) – Идентификатор группы

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

Связывает роли с группой.

Параметры

- **id** (*int*) – Идентификатор группы

Параметры формы

- **string role_id** – Уникальный идентификатор роли

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

Связывает компоненты с группой.

Параметры

- **id** (*int*) – Идентификатор группы

Параметры формы

- **string component_id** – Уникальный идентификатор компонента

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

int: *component_id* Удаляет компонент из группы.

Параметры

- **id** (*int*) – Идентификатор группы
- **component_id** (*int*) – Уникальный идентификатор компонента

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

Связывает проекты с группой.

Параметры

- **id** (*int*) – Идентификатор группы

Параметры формы

- **string project_id** – Уникальный идентификатор проекта

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

int: *project_id* Удаляет проект из группы.

Параметры

- **id** (*int*) – Идентификатор группы
- **project_id** (*int*) – Уникальный идентификатор проекта

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

Связывает языки с группой.

Параметры

- **id** (*int*) – Идентификатор группы

Параметры формы

- **string language_code** – Уникальный код языка

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

string: *language_code* Удаляет язык из группы.

Параметры

- **id** (*int*) – Идентификатор группы
- **language_code** (*string*) – Уникальный код языка

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

Связывает списки компонентов с группой.

Параметры

- **id** (*int*) – Идентификатор группы

Параметры формы

- **string component_list_id** – Уникальный идентификатор списка компонентов

DELETE /api/groups/(int: *id*)/componentlists/
int: *component_list_id* Удаляет список компонентов из группы.

Параметры

- **id** (*int*) – Идентификатор группы
- **component_list_id** (*int*) – Уникальный идентификатор списка компонентов

1.12.5 Роли

GET /api/roles/
Возвращает список всех ролей, связанных с пользователем. Если пользователь является суперпользователем, то возвращается список всех существующих ролей.

См.также:

Атрибуты объекта роли задокументированы в разделе *GET /api/roles/(int:id)/*.

POST /api/roles/
Создаёт новую роль.

Параметры

- **name** (*string*) – Название роли
- **permissions** (*array*) – Список кодовых названий разрешений

GET /api/roles/(int: *id*)/
Возвращает информацию о роли.

Параметры

- **id** (*int*) – Идентификатор роли

JSON -объект ответа

- **name** (*string*) – Название роли
- **permissions** (*array*) – список кодовых названий разрешений

Пример данных в JSON:

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

PUT /api/roles/(int: *id*)/
Изменяет параметры роли.

Параметры

- **id** (*int*) – Идентификатор роли

JSON -объект ответа

- **name** (*string*) – Название роли
- **permissions** (*array*) – список кодовых названий разрешений

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

Изменяет параметры роли.

Параметры

- **id** (*int*) – Идентификатор роли

JSON -объект ответа

- **name** (*string*) – Название роли
- **permissions** (*array*) – список кодовых названий разрешений

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

Удаляет роль.

Параметры

- **id** (*int*) – Идентификатор роли

1.12.6 Языки

GET /api/languages/

Возвращает список всех языков.

См.также:

Атрибуты объекта языка задокументированы в разделе *GET /api/languages/(string:language)/*.

POST /api/languages/

Создаёт новый язык.

Параметры

- **code** (*string*) – Название языка
- **name** (*string*) – Название языка
- **direction** (*string*) – Направление языка
- **plural** (*object*) – Формула множественного числа языка и количество форм множественных чисел

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

Возвращает информацию о языке.

Параметры

- **language** (*string*) – Код языка

JSON -объект ответа

- **code** (*string*) – Код языка
- **direction** (*string*) – Направление текста
- **plural** (*object*) – Объект информации о формах множественного числа языка
- **aliases** (*array*) – Массив псевдонимов языка

Пример данных в JSON:

```
{
    "code": "en",
    "direction": "ltr",
    "name": "English",
    "plural": {
        "id": 75,
        "source": 0,
```

(continues on next page)

(продолжение с предыдущей страницы)

```
        "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*)/

Изменяет параметры языка.

Параметры

- **language** (*string*) – Код языка

JSON -объект запроса

- **name** (*string*) – Название языка
- **direction** (*string*) – Направление языка
- **plural** (*object*) – Данные по формам множественного числа языка

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

Изменяет параметры языка.

Параметры

- **language** (*string*) – Код языка

JSON -объект запроса

- **name** (*string*) – Название языка
- **direction** (*string*) – Направление языка
- **plural** (*object*) – Данные по формам множественного числа языка

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

Удаляет язык.

Параметры

- **language** (*string*) – Код языка

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

Возвращает статистику по языку.

Параметры

- **language** (*string*) – Код языка

JSON -объект ответа

- **total** (*int*) – общее количество строк
- **total_words** (*int*) – общее количество слов
- **last_change** (*timestamp*) – последние изменения в языке
- **recent_changes** (*int*) – общее количество изменений
- **translated** (*int*) – количество переведённых строк

- **translated_percent** (*float*) – процент переведённых строк
- **translated_words** (*int*) – количество переведённых слов
- **translated_words_percent** (*int*) – процент переведённых слов
- **translated_chars** (*int*) – количество переведённых символов
- **translated_chars_percent** (*int*) – процент переведённых символов
- **total_chars** (*int*) – общее количество символов
- **fuzzy** (*int*) – количество неточных переводов
- **fuzzy_percent** (*int*) – процент неточных переводов
- **failing** (*int*) – количество строк с неудачными проверками
- **failing** – процент строк с неудачными проверками

1.12.7 Проекты

GET /api/projects/

Возвращает список всех проектов.

См.также:

Атрибуты объекта проекта задокументированы в разделе *GET /api/projects/(string:project)/*.

POST /api/projects/

Добавлено в версии 3.9.

Создаёт новый проект.

Параметры

- **name** (*string*) – Название проекта
- **slug** (*string*) – Плашка проекта
- **web** (*string*) – Веб-сайт проекта

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

Возвращает информацию о проекте.

Параметры

- **project** (*string*) – URL-плашка проекта

JSON -объект ответа

- **name** (*string*) – название проекта
- **slug** (*string*) – плашка проекта
- **web** (*string*) – веб-сайт проекта
- **components_list_url** (*string*) – URL-адрес списка компонентов; смотрите описание *GET /api/projects/(string:project)/components/*
- **repository_url** (*string*) – URL-адрес статуса репозитория; смотрите описание *GET /api/projects/(string:project)/repository/*
- **changes_list_url** (*string*) – URL-адрес списка изменений; смотрите описание *GET /api/projects/(string:project)/changes/*

Пример данных в JSON:

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

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

Добавлено в версии 4.3.

Редактирует проект запросом на исправление.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента

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

Добавлено в версии 4.3.

Редактирует проект запросом на замену.

Параметры

- **project** (*string*) – URL-плашка проекта

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

Добавлено в версии 3.9.

Удаляет проект.

Параметры

- **project** (*string*) – URL-плашка проекта

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

Возвращает список изменений проекта. По сути, эта точка входа работает так же и принимает те же самые параметры, что и *GET /api/changes/*, только для проекта.

Параметры

- **project** (*string*) – URL-плашка проекта

JSON -объект ответа

- **results** (*array*) – массив объектов компонентов; смотрите описание *GET /api/changes/(int:id)/*

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

Возвращает информацию о состоянии репозитория системы контроля версий. Эта конечная точка содержит только общую сводку по всем репозиториям проекта. Для получения более подробной информации о состоянии используйте конечную точку *GET /api/components/(string:project)/(string:component)/repository/*.

Параметры

- **project** (*string*) – URL-плашка проекта

JSON -объект ответа

- **needs_commit** (*boolean*) – есть ли какие-нибудь отложенные изменения для коммита
- **needs_merge** (*boolean*) – есть ли какие-нибудь изменения в вышестоящем репозитории для объединения
- **needs_push** (*boolean*) – есть ли какие-нибудь локальные изменения для отправки

Пример данных в JSON:

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

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

Выполняет заданную операцию на репозитории системы контроля версий.

Параметры

- **project** (*string*) – URL-плашка проекта

JSON -объект запроса

- **operation** (*string*) – Выполняемые операции: одна из `push`, `pull`, `commit`, `reset` или `cleanup`

JSON -объект ответа

- **result** (*boolean*) – результат операции

Пример CURL:

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

Пример JSON запроса:

```
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"}
```

Пример ответа в 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

{"result":true}
```

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

Возвращает список компонентов перевода в указанном проекте.

Параметры

- **project** (*string*) – URL-плашка проекта

JSON -объект ответа

- **results** (*array*) – массив объектов компонентов; смотрите описание *GET /api/components/(string:project)/(string:component)/*

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

Добавлено в версии 3.9.

Изменено в версии 4.3: The `zipfile` and `docfile` parameters are now accepted for VCS less components, see *Локальные файлы*.

Создаёт компоненты перевода в указанном проекте.

Параметры

- **project** (*string*) – URL-плашка проекта

JSON -объект запроса

- **zipfile** (*file*) – ZIP file to upload into Weblate for translations initialization
- **docfile** (*file*) – Документ для перевода

JSON -объект ответа

- **result** (*object*) – Созданный объект компонента; смотрите описание *GET /api/components/(string:project)/(string:component)/*

Пример CURL:

```
curl \
    --data-binary '{
        "branch": "master",
        "file_format": "po",
        "filemask": "po/*.po",
        "git_export": "",
        "license": "",
        "license_url": "",
        "name": "Weblate",
        "slug": "weblate",
        "repo": "file:///home/nijel/work/weblate-hello",
        "template": "",
        "new_base": "",
        "vcs": "git"
    }' \
    -H "Content-Type: application/json" \
    -H "Authorization: Token TOKEN" \
    http://example.com/api/projects/hello/components/
```

Пример JSON запроса:

```
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": "master",
    "file_format": "po",
    "filemask": "po/*.po",
    "git_export": "",
    "license": "",
    "license_url": "",
    "name": "Weblate",
    "slug": "weblate",
    "repo": "file:///home/nijel/work/weblate-hello",
    "template": "",
    "new_base": "",
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    "vcs": "git"
}
```

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

Возвращает разбитую на страницы статистику по всем языкам проекта.

Добавлено в версии 3.8.

Параметры

- **project** (*string*) – URL-плашка проекта

JSON -объект ответа

- **results** (*array*) – массив объектов статистики перевода
- **language** (*string*) – название языка
- **code** (*string*) – код языка
- **total** (*int*) – общее количество строк
- **translated** (*int*) – количество переведённых строк

- **translated_percent** (*float*) – процент переведённых строк
- **total_words** (*int*) – общее количество слов
- **translated_words** (*int*) – количество переведённых слов
- **words_percent** (*float*) – процент переведённых слов

GET /api/projects/(**string:** *project*)**/statistics/**
Возвращает статистику по проекту.

Добавлено в версии 3.8.

Параметры

- **project** (*string*) – URL-плашка проекта

JSON -объект ответа

- **total** (*int*) – общее количество строк
- **translated** (*int*) – количество переведённых строк
- **translated_percent** (*float*) – процент переведённых строк
- **total_words** (*int*) – общее количество слов
- **translated_words** (*int*) – количество переведённых слов
- **words_percent** (*float*) – процент переведённых слов

1.12.8 Компоненты

GET /api/components/
Возвращает список компонентов перевода.

См.также:

Атрибуты объекта компонента задокументированы в разделе *GET /api/components/(string:project)/(string:component)/*.

GET /api/components/(**string:** *project*)**/**
string: *component***/** Возвращает информацию о компоненте перевода.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента

JSON -объект ответа

- **project** (*object*) – проект перевода; смотрите описание *GET /api/projects/(string:project)/*
- **name** (*string*) – *Название компонента*
- **slug** (*string*) – *Плашка компонента*
- **vcs** (*string*) – *Система контроля версий*
- **repo** (*string*) – *Репозиторий исходного кода*
- **git_export** (*string*) – *URL экспортированного репозитория*
- **branch** (*string*) – *Ветка репозитория*
- **push_branch** (*string*) – *Ветка для отправки*
- **filemask** (*string*) – *Маска файла*
- **template** (*string*) – *Одноязычный базовый файл*

- **edit_template** (*string*) – *Правка базового файла*
- **intermediate** (*string*) – *Файл промежуточного языка*
- **new_base** (*string*) – *Шаблон для новых переводов*
- **file_format** (*string*) – *Формат файла*
- **license** (*string*) – *Лицензия перевода*
- **agreement** (*string*) – *Соглашение участника*
- **new_lang** (*string*) – *Добавление нового перевода*
- **language_code_style** (*string*) – *Стиль кода языка*
- **source_language** (*object*) – объект исходного языка; смотрите описание `GET /api/languages/(string:language)/`
- **push** (*string*) – *URL для отправки в репозиторий*
- **check_flags** (*string*) – *Флаги перевода*
- **priority** (*string*) – *Приоритет*
- **enforced_checks** (*string*) – *Принудительные проверки*
- **restricted** (*string*) – *Ограниченный доступ*
- **repoweb** (*string*) – *Просмотрщик репозитория*
- **report_source_bugs** (*string*) – *Адрес для сообщения об ошибке в исходной строке*
- **merge_style** (*string*) – *Стиль слияния*
- **commit_message** (*string*) – *Сообщения коммита, добавления, удаления, слияния и сообщения надстроек*
- **add_message** (*string*) – *Сообщения коммита, добавления, удаления, слияния и сообщения надстроек*
- **delete_message** (*string*) – *Сообщения коммита, добавления, удаления, слияния и сообщения надстроек*
- **merge_message** (*string*) – *Сообщения коммита, добавления, удаления, слияния и сообщения надстроек*
- **addon_message** (*string*) – *Сообщения коммита, добавления, удаления, слияния и сообщения надстроек*
- **allow_translation_propagation** (*string*) – *Разрешить распространение перевода*
- **enable_suggestions** (*string*) – *Включить предложения*
- **suggestion_voting** (*string*) – *Голосование по предложению*
- **suggestion_autoaccept** (*string*) – *Автоматическое принятие предложений*
- **push_on_commit** (*string*) – *Отправлять при коммите*
- **commit_pending_age** (*string*) – *Возраст изменений для коммита*
- **auto_lock_error** (*string*) – *Блокировать при ошибке*
- **language_regex** (*string*) – *Языковой фильтр*
- **variant_regex** (*string*) – *Регулярное выражение вариантов*
- **repository_url** (*string*) – URL-адрес статуса репозитория; смотрите описание `GET /api/components/(string:project)/(string:component)/repository/`

- **translations_url** (*string*) – URL-адрес списка переводов; смотрите описание *GET /api/components/(string:project)/(string:component)/translations/*
- **lock_url** (*string*) – URL-адрес статуса блокировки; смотрите описание *GET /api/components/(string:project)/(string:component)/lock/*
- **changes_list_url** (*string*) – URL-адрес списка изменений; смотрите описание *GET /api/components/(string:project)/(string:component)/changes/*

Пример данных в JSON:

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

PATCH /api/components/(string: *project*)**/string:** *component***/** Редактирует компонент запросом на исправление.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента
- **source_language** (*string*) – Код исходного языка проекта (необязательный)

JSON -объект запроса

- **name** (*string*) – название компонента

- **slug** (*string*) – плашка компонента
- **repo** (*string*) – URL-адрес репозитория системы контроля версий

Пример CURL:

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

Пример JSON запроса:

```
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"
}
```

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

(continues on next page)

(продолжение с предыдущей страницы)

```
    "vcs": "git",
    "web_url": "http://example.com/projects/hello/weblate/"
}
```

PUT /api/components/(string: *project*)/
string: *component*/ Редактирует компонент запросом на замену.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента

JSON -объект запроса

- **branch** (*string*) – Ветка репозитория системы контроля версий
- **file_format** (*string*) – формат файла переводов
- **filemask** (*string*) – маска файлов перевода в репозитории
- **name** (*string*) – название компонента
- **slug** (*string*) – плашка компонента
- **repo** (*string*) – URL-адрес репозитория системы контроля версий
- **template** (*string*) – базовый файл для одноязычных переводов
- **new_base** (*string*) – базовый файл для добавления новых переводов
- **vcs** (*string*) – система контроля версий

DELETE /api/components/(string: *project*)/
string: *component*/ Добавлено в версии 3.9.

Удаляет компонент.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента

GET /api/components/(string: *project*)/
string: *component*/changes/ Возвращает список изменений компонента. По сути, эта точка входа работает так же и принимает те же самые параметры, что и *GET /api/changes/*, только для компонента.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента

JSON -объект ответа

- **results** (*array*) – массив объектов компонентов; смотрите описание *GET /api/changes/(int:id)/*

GET /api/components/(string: *project*)/
string: *component*/screenshots/ Возвращает список снимков экрана компонента.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента

JSON -объект ответа

- **results** (*array*) – массив снимков экрана компонента; смотрите описание *GET /api/screenshots/(int:id)/*

GET /api/components/(string: *project*)/
string: *component*/lock/ Возвращает состояние блокировки компонента.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента

JSON -объект ответа

- **locked** (*boolean*) – заблокирован ли компонент для обновлений

Пример данных в JSON:

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

POST /api/components/(string: *project*)/
string: *component*/lock/ Устанавливает статус блокировки компонента.

Ответ такой же, как и у *GET /api/components/(string:project)/(string:component)/lock/*.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента

JSON -объект запроса

- **lock** – Логическое значение, стоит ли блокировка или нет.

Пример CURL:

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

Пример JSON запроса:

```
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}
```

Пример ответа в JSON:

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

{"locked":true}
```

GET /api/components/(string: *project*)/
string: *component*/repository/ Возвращает информацию о состоянии репозитория системы контроля версий.

Ответ такой же, как и у *GET /api/projects/(string:project)/repository/*.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента

JSON -объект ответа

- **needs_commit** (*boolean*) – есть ли какие-нибудь отложенные изменения для коммита
- **needs_merge** (*boolean*) – есть ли какие-нибудь изменения в вышестоящем репозитории для объединения
- **needs_push** (*boolean*) – есть ли какие-нибудь локальные изменения для отправки
- **remote_commit** (*string*) – Информация о коммите из удалённого репозитория
- **status** (*string*) – Статус репозитория системы контроля версий по её же данным
- **merge_failure** – Текст, описывающий причину неудачи объединения или `null`, если её нет

POST /api/components/(string: *project*)/
string: *component*/repository/ Выполняет заданную операцию на репозитории системы контроля версий.

Документацию смотрите в описании *POST /api/projects/(string:project)/repository/*.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента

JSON -объект запроса

- **operation** (*string*) – Выполняемые операции: одна из `push`, `pull`, `commit`, `reset` или `cleanup`

JSON -объект ответа

- **result** (*boolean*) – результат операции

Пример CURL:

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

Пример JSON запроса:

```
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"}
```

Пример ответа в 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

{"result":true}
```

GET /api/components/(string: *project*)/
string: *component*/monolingual_base/ Скачивает базовый файл для одноязычных переводов.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента

GET /api/components/(string: *project*)/
string: *component*/new_template/ Скачивает файл шаблона для новых переводов.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента

GET /api/components/(string: *project*)/
string: *component*/translations/ Возвращает список объектов перевода в указанном компоненте.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента

JSON -объект ответа

- **results** (*array*) – массив объектов перевода; смотрите описание *GET /api/translations/(string:project)/(string:component)/(string:language)/*

POST /api/components/(string: *project*)/
string: *component*/translations/ Создаёт новый перевод в указанном компоненте.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента

JSON -объект запроса

- **language_code** (*string*) – код языка перевода; смотрите описание *GET /api/languages/(string:language)/*

JSON -объект ответа

- **result** (*object*) – новый созданный объект перевода

Пример CURL:

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

Пример JSON запроса:

```
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"}
```

Пример ответа в JSON:

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

{
    "failing_checks": 0,
    "failing_checks_percent": 0,
    "failing_checks_words": 0,
    "filename": "po/cs.po",
    "fuzzy": 0,
    "fuzzy_percent": 0.0,
    "fuzzy_words": 0,
    "have_comment": 0,
    "have_suggestion": 0,
    "is_template": false,
    "is_source": false,
    "language": {
        "code": "cs",
        "direction": "ltr",
        "name": "Czech",
        "url": "http://example.com/api/languages/cs/",
        "web_url": "http://example.com/languages/cs/"
    },
    "language_code": "cs",
    "id": 125,
    "last_author": null,
    "last_change": null,
    "share_url": "http://example.com/engage/hello/cs/",
    "total": 4,
    "total_words": 15,
    "translate_url": "http://example.com/translate/hello/weblate/cs/",
    "translated": 0,
    "translated_percent": 0.0,
    "translated_words": 0,
    "url": "http://example.com/api/translations/hello/weblate/cs/",
    "web_url": "http://example.com/projects/hello/weblate/cs/"
}
```

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

string: *component*/statistics/ Возвращает разбитую на страницы статистику по всем перево-

дам компонента.

Добавлено в версии 2.7.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента

JSON -объект ответа

- **results** (*array*) – массив объектов статистики перевода; смотрите описание *GET /api/translations/(string:project)/(string:component)/(string:language)/statistics/*

1.12.9 Переводы

GET /api/translations/

Возвращает список переводов.

См.также:

Атрибуты объекта перевода задокументированы в разделе *GET /api/translations/(string:project)/(string:component)/(string:language)/*.

GET /api/translations/(string: *project*)**/string:** *component***/string:** *language***/** Возвращает информацию о переводе.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента
- **language** (*string*) – Код языка перевода

JSON -объект ответа

- **component** (*object*) – объект компонента; смотрите описание *GET /api/components/(string:project)/(string:component)/*
- **failing_checks** (*int*) – количество строк с неудачными проверками
- **failing_checks_percent** (*float*) – процент строк с неудачными проверками
- **failing_checks_words** (*int*) – количество слов с неудачными проверками
- **filename** (*string*) – имя файла перевода
- **fuzzy** (*int*) – количество строк, помеченных на правку
- **fuzzy_percent** (*float*) – процент строк, помеченных на правку
- **fuzzy_words** (*int*) – количество слов, помеченных на правку
- **have_comment** (*int*) – количество строк с комментариями
- **have_suggestion** (*int*) – количество строк с комментариями
- **is_template** (*boolean*) – использует ли перевод одноязычные файлы
- **language** (*object*) – объект исходного языка; смотрите описание *GET /api/languages/(string:language)/*
- **language_code** (*string*) – код языка, используемый в репозитории; он может отличаться от кода языка в объекте языка
- **last_author** (*string*) – имя последнего автора
- **last_change** (*timestamp*) – метка времени последнего изменения

- **revision** (*string*) – хэш ревизии файла
- **share_url** (*string*) – URL-адрес для принятия участия в переводе, ведущий на вводную страницу
- **total** (*int*) – общее количество строк
- **total_words** (*int*) – общее количество слов
- **translate_url** (*string*) – URL-адрес для перевода
- **translated** (*int*) – количество переведённых строк
- **translated_percent** (*float*) – процент переведённых строк
- **translated_words** (*int*) – количество переведённых слов
- **repository_url** (*string*) – URL-адрес статуса репозитория; смотрите описание *GET /api/translations/(string:project)/(string:component)/(string:language)/repository/*
- **file_url** (*string*) – URL-адрес объекта файла; смотрите описание *GET /api/translations/(string:project)/(string:component)/(string:language)/file/*
- **changes_list_url** (*string*) – URL-адрес списка изменений; смотрите описание *GET /api/translations/(string:project)/(string:component)/(string:language)/changes/*
- **units_list_url** (*string*) – URL-адрес списка строк; смотрите описание *GET /api/translations/(string:project)/(string:component)/(string:language)/units/*

Пример данных в JSON:

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

(continues on next page)

(продолжение с предыдущей страницы)


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


DELETE /api/translations/(string: *project*)/
string: *component*/string: *language*/ Добавлено в версии 3.9.

Удаляет перевод.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента
- **language** (*string*) – Код языка перевода

GET /api/translations/(string: *project*)/
string: *component*/string: *language*/changes/ Возвращает список изменений перевода. По сути, эта точка входа работает так же и принимает те же самые параметры, что и *GET /api/changes/* , только для перевода.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента
- **language** (*string*) – Код языка перевода

JSON -объект ответа

- **results** (*array*) – массив объектов компонентов; смотрите описание *GET /api/changes/(int:id)/*

GET /api/translations/(string: *project*)/
string: *component*/string: *language*/units/ Возвращает список единиц перевода.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента
- **language** (*string*) – Код языка перевода
- **q** (*string*) – Search query string *Поиск* (optional)

JSON -объект ответа

- **results** (*array*) – массив объектов компонента; смотрите описание *GET /api/units/(int:id)/*

POST /api/translations/(**string:** *project*)**/**
string: *component***/string:** *language***/units/** Добавляет новую одноязычную единицу перевода.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента
- **language** (*string*) – Код языка перевода

JSON -объект запроса

- **key** (*string*) – Название единицы перевода
- **value** (*string*) – Значение единицы перевода

POST /api/translations/(**string:** *project*)**/**
string: *component***/string:** *language***/autotranslate/** Запускает автоматический перевод.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента
- **language** (*string*) – Код языка перевода

JSON -объект запроса

- **mode** (*string*) – Режим автоматического перевода
- **filter_type** (*string*) – Тип фильтра автоматического перевода
- **auto_source** (*string*) – Источник автоматического перевода
- **component** (*string*) – Добавлять переводы этого компонента в общую память переводов проекта, дабы получить к ним доступ из других компонентов.
- **engines** (*string*) – Механизмы машинного перевода
- **threshold** (*string*) – Порог оценки

GET /api/translations/(**string:** *project*)**/**
string: *component***/string:** *language***/file/** Скачивает текущий файл перевода в том виде, в котором он сохранён в системе контроля версий (без параметра `format`) или преобразованный в стандартный формат (в настоящее время поддерживаются: Gettext PO, MO, XLIFF и TBX).

Примечание: Эта конечная точка API использует отличающуюся от остального API логику вывода, так как она работает не с данными, а с целым файлом. Отличается логика при установке допустимого параметра `format` — без него вы получите файл перевода, хранящийся в системе контроля версий.

Параметры запроса

- **format** – Используемый формат файла; если не указан, преобразование формата не происходит; поддерживаемые форматы файлов: `po`, `mo`, `xliff`, `xliff11`, `tbx`, `csv`, `xlsx`, `json`, `aresource` и `strings`

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента
- **language** (*string*) – Код языка перевода

POST /api/translations/(string: *project*)/string: *component*/string: *language*/file/ Загружает новый файл с переводами.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента
- **language** (*string*) – Код языка перевода

Параметры формы

- **string conflicts** – Как поступать с конфликтами (`ignore`, `replace-translated` или `replace-approved`)
- **file file** – Загруженный файл
- **string email** – E-mail автора
- **string author** – Имя автора
- **string method** – Метод загрузки (`translate`, `approve`, `suggest`, `fuzzy`, `replace` или `source`), смотрите раздел *Способы импортирования*
- **string fuzzy** – Обработка строк «на правку» (*пусто*, `process` или `approve`)

Пример CURL:

```
curl -X POST \
    -F file=@strings.xml \
    -H "Authorization: Token TOKEN" \
    http://example.com/api/translations/hello/android/cs/file/
```

GET /api/translations/(string: *project*)/string: *component*/string: *language*/repository/ Возвращает информацию о состоянии репозитория системы контроля версий.

Ответ такой же, как и у *GET /api/components/(string:project)/(string:component)/repository/*.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента
- **language** (*string*) – Код языка перевода

POST /api/translations/(string: *project*)/string: *component*/string: *language*/repository/ Выполняет заданную операцию на репозитории системы контроля версий.

Документацию смотрите в описании *POST /api/projects/(string:project)/repository/*.

Параметры

- **project** (*string*) – URL-плашка проекта

- **component** (*string*) – URL-плашка компонента
- **language** (*string*) – Код языка перевода

JSON -объект запроса

- **operation** (*string*) – Выполняемые операции: одна из `push`, `pull`, `commit`, `reset` или `cleanup`

JSON -объект ответа

- **result** (*boolean*) – результат операции

GET /api/translations/(string: *project*)/string: *component*/string: *language*/statistics/ Возвращает подробную статистику по переводам.

Добавлено в версии 2.7.

Параметры

- **project** (*string*) – URL-плашка проекта
- **component** (*string*) – URL-плашка компонента
- **language** (*string*) – Код языка перевода

JSON -объект ответа

- **code** (*string*) – код языка
- **failing** (*int*) – количество неудачных проверок
- **failing_percent** (*float*) – процент неудачных проверок
- **fuzzy** (*int*) – количество строк, нуждающихся в пересмотре
- **fuzzy_percent** (*float*) – процент строк, нуждающихся в пересмотре
- **total_words** (*int*) – общее количество слов
- **translated_words** (*int*) – количество переведённых слов
- **last_author** (*string*) – имя последнего автора
- **last_change** (*timestamp*) – дата последнего изменения
- **name** (*string*) – название языка
- **total** (*int*) – общее количество строк
- **translated** (*int*) – количество переведённых строк
- **translated_percent** (*float*) – процент переведённых строк
- **url** (*string*) – URL-адрес для доступа к переводу (URL-адрес вводной страницы)
- **url_translate** (*string*) – URL-адрес для доступа к переводу (реальный URL-адрес перевода)

1.12.10 Единицы перевода

Добавлено в версии 2.10.

GET /api/units/
Возвращает список единиц перевода.

См.также:

Атрибуты объекта единицы перевода задокументированы в разделе *GET /api/units/(int:id)/*.

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

Изменено в версии 4.3: The `target` and `source` are now arrays to properly handle plural strings.

Возвращает информацию о единице перевода.

Параметры

- **id** (*int*) – Идентификатор единицы перевода

JSON -объект ответа

- **translation** (*string*) – URL-адрес связанного объекта перевода
- **source** (*array*) – исходная строка
- **previous_source** (*string*) – предыдущая исходная строка, использованная для поиска нечёткого соответствия
- **target** (*array*) – целевая строка
- **id_hash** (*string*) – уникальный идентификатор единицы перевода
- **content_hash** (*string*) – уникальный идентификатор исходной строки
- **location** (*string*) – местоположение единицы перевода в исходном коде
- **context** (*string*) – контекст единицы перевода
- **note** (*string*) – примечания к единице перевода
- **flags** (*string*) – флаги единицы перевода
- **state** (*int*) – unit state, 0 - not translated, 10 - needs editing, 20 - translated, 30 - approved, 100 - read only
- **fuzzy** (*boolean*) – является ли перевод единицы перевода неточным или отмеченным на правку
- **translated** (*boolean*) – переведена ли единица перевода
- **approved** (*boolean*) – одобрена ли единица перевода
- **position** (*int*) – позиция единицы перевода в файле перевода
- **has_suggestion** (*boolean*) – содержит ли единица перевода предложения
- **has_comment** (*boolean*) – содержит ли единица перевода комментарии
- **has_failing_check** (*boolean*) – имеются ли у единицы перевода неудачные проверки
- **num_words** (*int*) – количество слов в исходной строке
- **priority** (*int*) – приоритет перевода; по умолчанию равен 100
- **id** (*int*) – идентификатор единицы перевода
- **explanation** (*string*) – Объяснение строки, доступно на исходных единицах перевода, смотрите раздел *Дополнительная информация об исходных строках*
- **extra_flags** (*string*) – Дополнительные флаги строки, доступны на исходных единицах перевода, смотрите раздел *Настройка поведения*
- **web_url** (*string*) – URL-адрес, по которому можно отредактировать единицу перевода
- **souce_unit** (*string*) – Ссылка на исходную единицу перевода; смотрите описание `GET /api/units/(int:id)/`

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

Добавлено в версии 4.3.

Выполняет частичное обновление единицы перевода.

Параметры

- **id** (*int*) – Идентификатор единицы перевода

JSON -объект запроса

- **state** (*int*) – unit state, 0 - not translated, 10 - needs editing, 20 - translated, 30 - approved, 100 - read only
- **target** (*array*) – целевая строка
- **explanation** (*string*) – Объяснение строки, доступно на исходных единицах перевода, смотрите раздел *Дополнительная информация об исходных строках*
- **extra_flags** (*string*) – Дополнительные флаги строки, доступны на исходных единицах перевода, смотрите раздел *Настройка поведения*

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

Добавлено в версии 4.3.

Выполняет полное обновление единицы перевода.

Параметры

- **id** (*int*) – Идентификатор единицы перевода

JSON -объект запроса

- **state** (*int*) – unit state, 0 - not translated, 10 - needs editing, 20 - translated, 30 - approved, 100 - read only
- **target** (*array*) – целевая строка
- **explanation** (*string*) – Объяснение строки, доступно на исходных единицах перевода, смотрите раздел *Дополнительная информация об исходных строках*
- **extra_flags** (*string*) – Дополнительные флаги строки, доступны на исходных единицах перевода, смотрите раздел *Настройка поведения*

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

Добавлено в версии 4.3.

Удаляет единицу перевода.

Параметры

- **id** (*int*) – Идентификатор единицы перевода

1.12.11 Изменения

Добавлено в версии 2.10.

GET /api/changes/

Изменено в версии 4.1: Фильтрация изменений была введена в выпуске 4.1.

Возвращает список изменений в переводе.

См.также:

Атрибуты объекта изменения задокументированы в разделе `GET /api/changes/(int:id)/`.

Параметры запроса

- **user** (*string*) – Имя пользователя для фильтров
- **action** (*int*) – Действие для фильтра, может быть указано несколько раз
- **timestamp_after** (*timestamp*) – Временная метка в формате ISO 8601, изменения после которой нужно вернуть

- **timestamp_before** (*timestamp*) – Временная метка в формате ISO 8601, изменения до которой нужно вернуть

GET /api/changes/(int: *id*)/
Возвращает информацию об изменении в переводе.

Параметры

- **id** (*int*) – Идентификатор изменения

JSON -объект ответа

- **unit** (*string*) – URL-адрес связанного объекта единицы перевода
- **translation** (*string*) – URL-адрес связанного объекта перевода
- **component** (*string*) – URL-адрес связанного объекта компонента
- **glossary_term** (*string*) – URL-адрес связанного объекта термина глоссария
- **user** (*string*) – URL-адрес связанного объекта пользователя
- **author** (*string*) – URL-адрес связанного объекта автора
- **timestamp** (*timestamp*) – временная метка события
- **action** (*int*) – числовой идентификатор действия
- **action_name** (*string*) – текстовое описание действия
- **target** (*string*) – изменённый текст или информация о событии
- **id** (*int*) – идентификатор изменения

1.12.12 Снимки экрана

Добавлено в версии 2.14.

GET /api/screenshots/
Возвращает список снимков экрана для исходных строк.

См.также:

Атрибуты объекта снимка экрана перевода задокументированы в разделе *GET /api/screenshots/(int:id)/*.

GET /api/screenshots/(int: *id*)/
Возвращает информацию о снимке экрана.

Параметры

- **id** (*int*) – Идентификатор снимка экрана

JSON -объект ответа

- **name** (*string*) – название снимка экрана
- **component** (*string*) – URL-адрес связанного объекта компонента
- **file_url** (*string*) – URL-адрес для скачивания файла; смотрите описание *GET /api/screenshots/(int:id)/file/*
- **units** (*array*) – ссылка на связанную информацию об исходной строке; смотрите описание *GET /api/units/(int:id)/*

GET /api/screenshots/(int: *id*)/file/
Скачивает изображение снимка экрана.

Параметры

- **id** (*int*) – Идентификатор снимка экрана

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

Заменяет изображение снимка экрана.

Параметры

- **id** (*int*) – Идентификатор снимка экрана

Параметры формы

- **file image** – Загруженный файл

Пример 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/

Связывает исходную строку со снимком экрана.

Параметры

- **id** (*int*) – Идентификатор снимка экрана

Параметры формы

- **string unit_id** – Идентификатор единицы перевода

JSON -объект ответа

- **name** (*string*) – название снимка экрана
- **translation** (*string*) – URL-адрес связанного объекта перевода
- **file_url** (*string*) – URL-адрес для скачивания файла; смотрите описание *GET /api/screenshots/(int:id)/file/*
- **units** (*array*) – ссылка на связанную информацию об исходной строке; смотрите описание *GET /api/units/(int:id)/*

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

int: *unit_id* Удаляет связь исходной строки со снимком экрана.

Параметры

- **id** (*int*) – Идентификатор снимка экрана
- **unit_id** – ИД единицы перевода исходной строки

POST /api/screenshots/

Создаёт новый снимок экрана.

Параметры формы

- **file image** – Загруженный файл
- **string name** – Название снимка экрана
- **string project_slug** – Плашка проекта
- **string component_slug** – Плашка компонента
- **string language_code** – Код языка

JSON -объект ответа

- **name** (*string*) – название снимка экрана
- **component** (*string*) – URL-адрес связанного объекта компонента
- **file_url** (*string*) – URL-адрес для скачивания файла; смотрите описание *GET /api/screenshots/(int:id)/file/*

- **units** (*array*) – ссылка на связанную информацию об исходной строке; смотрите описание *GET /api/units/(int:id)/*

PATCH /api/screenshots/(int: *id*)/
Редактирует часть информации о снимке экрана.

Параметры

- **id** (*int*) – Идентификатор снимка экрана

JSON -объект ответа

- **name** (*string*) – название снимка экрана
- **component** (*string*) – URL-адрес связанного объекта компонента
- **file_url** (*string*) – URL-адрес для скачивания файла; смотрите описание *GET /api/screenshots/(int:id)/file/*
- **units** (*array*) – ссылка на связанную информацию об исходной строке; смотрите описание *GET /api/units/(int:id)/*

PUT /api/screenshots/(int: *id*)/
Редактирует всю информацию о снимке экрана.

Параметры

- **id** (*int*) – Идентификатор снимка экрана

JSON -объект ответа

- **name** (*string*) – название снимка экрана
- **component** (*string*) – URL-адрес связанного объекта компонента
- **file_url** (*string*) – URL-адрес для скачивания файла; смотрите описание *GET /api/screenshots/(int:id)/file/*
- **units** (*array*) – ссылка на связанную информацию об исходной строке; смотрите описание *GET /api/units/(int:id)/*

DELETE /api/screenshots/(int: *id*)/
Удаляет снимок экрана.

Параметры

- **id** (*int*) – Идентификатор снимка экрана

1.12.13 Списки компонентов

Добавлено в версии 4.0.

GET /api/component-lists/
Возвращает список списков компонентов.

См.также:

Атрибуты объекта списка компонентов задокументированы в разделе *GET /api/component-lists/(str:slug)/*.

GET /api/component-lists/(str: *slug*)/
Возвращает информацию о списке компонентов.

Параметры

- **slug** (*string*) – Плашка списка компонентов

JSON -объект ответа

- **name** (*string*) – название списка компонентов

- **slug** (*string*) – плашка списка компонентов
- **show_dashboard** (*boolean*) – показывать ли его на панели управления
- **components** (*array*) – ссылка на связанные компоненты; смотрите описание *GET /api/components/(string:project)/(string:component)/*
- **auto_assign** (*array*) – правила автоматического назначения

PUT /api/component-lists/(**str:** *slug*)**/**

Изменяет параметры списка компонентов.

Параметры

- **slug** (*string*) – Плашка списка компонентов

JSON -объект запроса

- **name** (*string*) – название списка компонентов
- **slug** (*string*) – плашка списка компонентов
- **show_dashboard** (*boolean*) – показывать ли его на панели управления

PATCH /api/component-lists/(**str:** *slug*)**/**

Изменяет параметры списка компонентов.

Параметры

- **slug** (*string*) – Плашка списка компонентов

JSON -объект запроса

- **name** (*string*) – название списка компонентов
- **slug** (*string*) – плашка списка компонентов
- **show_dashboard** (*boolean*) – показывать ли его на панели управления

DELETE /api/component-lists/(**str:** *slug*)**/**

Удаляет список компонентов.

Параметры

- **slug** (*string*) – Плашка списка компонентов

POST /api/component-lists/(**str:** *slug*)**/components/**

Связывает компонент со списком компонентов.

Параметры

- **slug** (*string*) – Плашка списка компонентов

Параметры формы

- **string component_id** – Идентификатор компонента

DELETE /api/component-lists/(**str:** *slug*)**/components/**
str: *component_slug* Отвязывает компонент от списка компонентов.

Параметры

- **slug** (*string*) – Плашка списка компонентов
- **component_slug** (*string*) – Плашка компонента

1.12.14 Глоссарий

GET /api/glossary/

Возвращает список всех связанных с проектом глоссариев, к которым у пользователя есть доступ.

См.также:

Атрибуты объекта языка задокументированы в разделе *GET /api/languages/(string:language)/*.

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

Возвращает информацию о глоссарии.

Параметры

- **id** (*int*) – Идентификатор глоссария

JSON -объект ответа

- **name** (*string*) – Код языка
- **color** (*string*) – Направление текста
- **source_language** (*object*) – Объект информации о формах множественного числа языка
- **projects** (*array*) – ссылка на связанные проекты; смотрите описание *GET /api/projects/(string:project)/*

Пример данных в JSON:

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

(continues on next page)

(продолжение с предыдущей страницы)

```
                "source": 0,
                "number": 2,
                "formula": "n != 1",
                "type": 1
            },
            "aliases": [
                "english",
                "en_en",
                "base",
                "source",
                "eng"
            ],
            "direction": "ltr",
            "web_url": "http://example.com/languages/en/",
            "url": "http://example.com/api/languages/en/",
            "statistics_url": "http://example.com/api/languages/en/statistics/"
        },
        "web_url": "http://example.com/projects/demo1/",
        "url": "http://example.com/api/projects/demo1/",
        "components_list_url": "http://example.com/api/projects/demo1/
↪components/",
        "repository_url": "http://example.com/api/projects/demo1/repository/",
        "statistics_url": "http://example.com/api/projects/demo1/statistics/",
        "changes_list_url": "http://example.com/api/projects/demo1/changes/",
        "languages_url": "http://example.com/api/projects/demo1/languages/"
    },
    "projects_url": "http://example.com/api/glossary/7/projects/",
    "terms_url": "http://example.com/api/glossary/7/terms/",
    "url": "http://example.com/api/glossary/7/"
}
```

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

Изменяет параметры глоссария.

Параметры

- **id** (*int*) – Идентификатор глоссария

JSON -объект запроса

- **name** (*string*) – Название языка
- **color** (*string*) – Направление языка
- **source_language** (*object*) – Данные по формам множественного числа языка

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

Изменяет параметры глоссария.

Параметры

- **id** (*int*) – Идентификатор глоссария

JSON -объект запроса

- **name** (*string*) – Название языка
- **color** (*string*) – Направление языка
- **source_language** (*object*) – Данные по формам множественного числа языка

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

Удаляет глоссарий.

Параметры

- **id** (*int*) – Идентификатор глоссария

GET /api/glossary/(int: *id*)/projects/
Возвращает проекты, связанные с глоссарием.

Параметры

- **id** (*int*) – Идентификатор глоссария

JSON -объект ответа

- **projects** (*array*) – связанные проекты; смотрите описание *GET /api/projects/(string:project)/*

POST /api/glossary/(int: *id*)/projects/
Связывает проект с глоссарием.

Параметры

- **id** (*int*) – Идентификатор глоссария

Параметры формы

- **string project_slug** – Плашка проекта

DELETE /api/glossary/(int: *id*)/projects/
Remove association of a project with a glossary.

Параметры

- **id** (*int*) – Идентификатор глоссария

Параметры формы

- **string project_slug** – Плашка проекта

GET /api/glossary/(int: *id*)/terms/
Перечисляет термины в глоссарии.

Параметры

- **id** (*int*) – Идентификатор глоссария

POST /api/glossary/(int: *id*)/terms/
Связывает термины с глоссарием.

Параметры

- **id** (*int*) – Идентификатор глоссария

JSON -объект запроса

- **language** (*object*) – Язык термина
- **source** (*string*) – Исходная строка термина
- **target** (*string*) – Целевая строка термина

GET /api/glossary/(int: *id*)/terms/
int: *term_id*/ Возвращает связанный с глоссарием термин.

Параметры

- **id** (*int*) – Идентификатор глоссария
- **term_id** (*int*) – Идентификатор термина

PUT /api/glossary/(int: *id*)/terms/
int: *term_id*/ Редактирует связанный с глоссарием термин.

Параметры

- **id** (*int*) – Идентификатор глоссария
- **term_id** (*int*) – Идентификатор термина

JSON -объект запроса

- **language** (*object*) – Язык термина
- **source** (*string*) – Исходная строка термина
- **target** (*string*) – Целевая строка термина

PATCH /api/glossary/(int: *id*)/terms/
int: *term_id*/ Редактирует связанный с глоссарием термин.

Параметры

- **id** (*int*) – Идентификатор глоссария
- **term_id** (*int*) – Идентификатор термина

JSON -объект запроса

- **language** (*object*) – Язык термина
- **source** (*string*) – Исходная строка термина
- **target** (*string*) – Целевая строка термина

DELETE /api/glossary/(int: *id*)/terms/
int: *term_id*/ Удаляет связанный с глоссарием термин.

Параметры

- **id** (*int*) – Идентификатор глоссария
- **term_id** (*int*) – Идентификатор термина

1.12.15 Обработчики уведомлений

Обработчики уведомлений позволяют внешним приложениям уведомлять Weblate об обновлении репозитория системы контроля версий.

Вы можете использовать конечные точки репозитория для проектов, компонентов и переводов для обновления отдельных репозиториев; документацию смотрите в описании *POST /api/projects/(string:project)/repository/*.

GET /hooks/update/(string: *project*)/
string: *component*/ Не рекомендуется, начиная с версии 2.6: Пожалуйста, используйте вместо этой конечной точки точку *POST /api/components/(string:project)/(string:component)/repository/*, которая корректно работает с аутентификацией для проектов, ограниченных списками контроля доступа.

Запускает обновление компонента (извлечение кода из системы контроля версий и поиск в нём изменений перевода).

GET /hooks/update/(string: *project*)/
Не рекомендуется, начиная с версии 2.6: Пожалуйста, используйте вместо этой конечной точки точку *POST /api/projects/(string:project)/repository/*, которая корректно работает с аутентификацией для проектов, ограниченных списками контроля доступа.

Запускает обновление всех компонентов в проекте (извлечение кода из системы контроля версий и поиск в нём изменений перевода).

POST /hooks/github/
Специальный обработчик для обработки уведомлений GitHub и автоматического обновления соответствующих компонентов.

Примечание: GitHub включает прямую поддержку уведомлений Weblate: включите сервисный обработчик Weblate в настройках репозитория и установите URL-адрес в соответствие с URL-адресом вашей установки Weblate.

См.также:

Автоматическое получение изменений из GitHub'а Инструкция по настройке интеграции с GitHub

https://docs.github.com/en/github/extending-github/about-webhooks Общая информация о веб-обработчиках GitHub

ENABLE_HOOKS Для включения обработчиков для всего Weblate

POST /hooks/gitlab/

Специальный обработчик для обработки уведомлений GitLab и автоматического обновления соответствующих компонентов.

См.также:

Автоматическое получение изменений из GitLab'а Инструкция по настройке интеграции с GitLab

https://docs.gitlab.com/ce/user/project/integrations/webhooks.html Общая информация о веб-обработчиках GitLab

ENABLE_HOOKS Для включения обработчиков для всего Weblate

POST /hooks/bitbucket/

Специальный обработчик для обработки уведомлений Bitbucket и автоматического обновления соответствующих компонентов.

См.также:

Автоматическое получение изменений из Bitbucket'а Инструкция по настройке интеграции с Bitbucket

https://confluence.atlassian.com/bitbucket/manage-webhooks-735643732.html Общая информация о веб-обработчиках Bitbucket

ENABLE_HOOKS Для включения обработчиков для всего Weblate

POST /hooks/pagure/

Добавлено в версии 3.3.

Специальный обработчик для обработки уведомлений Pagure и автоматического обновления соответствующих компонентов.

См.также:

Автоматическое получение изменений из Pagure Инструкция по настройке интеграции с Pagure

https://docs.pagure.org/pagure/usage/using_webhooks.html Общая информация о веб-обработчиках Pagure

ENABLE_HOOKS Для включения обработчиков для всего Weblate

POST /hooks/azure/

Добавлено в версии 3.8.

Специальный обработчик для обработки уведомлений Azure Repos и автоматического обновления соответствующих компонентов.

См.также:

Автоматическое получение изменений из Azure Repos Инструкция по настройке интеграции с Azure

https://docs.microsoft.com/ru-ru/azure/devops/service-hooks/services/webhooks Общая информация о веб-обработчиках Azure Repos

ENABLE_HOOKS Для включения обработчиков для всего Weblate

POST /hooks/gitea/

Добавлено в версии 3.9.

Специальный обработчик для обработки уведомлений Gitea и автоматического обновления соответствующих компонентов.

См.также:

Автоматическое получение изменений из репозиториев Gitea Инструкция по настройке интеграции с Gitea

https://docs.gitea.io/en-us/webhooks/ Общая информация о веб-обработчиках Gitea

ENABLE_HOOKS Для включения обработчиков для всего Weblate

POST /hooks/gitee/

Добавлено в версии 3.9.

Специальный обработчик для обработки уведомлений Gitee и автоматического обновления соответствующих компонентов.

См.также:

Автоматическое получение изменений из репозиториев Gitee Инструкция по настройке интеграции с Gitee

https://gitee.com/help/categories/40 Общая информация о веб-обработчиках Gitee

ENABLE_HOOKS Для включения обработчиков для всего Weblate

1.12.16 Экспорты

Чтобы позволить вам и дальше обрабатывать данные, Weblate предоставляет вам различные виды экспорта.

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

Параметры запроса

- **format** (*string*) – Выходной формат: `json` или `csv`

Не рекомендуется, начиная с версии 2.6: Пожалуйста, используйте вместо этой конечной точки точки *GET /api/components/(string:project)/(string:component)/statistics/* или *GET /api/translations/(string:project)/(string:component)/(string:language)/statistics/*; они также разрешает доступ к проектам, контролируемым списками контроля доступа.

Возвращает статистику по указанному компоненту в заданном формате.

Пример запроса:

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

Пример ответа:

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

[
    {
        "code": "cs",
        "failing": 0,
```

(continues on next page)

(продолжение с предыдущей страницы)


```
        "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/master/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/master/nl/
↪"
    },
    {
        "code": "el",
        "failing": 11,
        "failing_percent": 2.5,
        "fuzzy": 21,
        "fuzzy_percent": 4.8,
        "last_author": null,
        "last_change": null,
        "name": "Greek",
        "total": 436,
        "total_words": 15271,
        "translated": 312,
        "translated_percent": 71.6,
        "translated_words": 3201,
        "url": "http://hosted.weblate.org/engage/weblate/el/",
        "url_translate": "http://hosted.weblate.org/projects/weblate/master/el/
↪"
    }
]
```

1.12.17 RSS-каналы

Изменения в переводах экспортируются в RSS-каналы.

GET /exports/rss/(**string:** *project*)**/**
string: *component***/string:** *language***/** Возвращает RSS-ленту с последними изменениями в переводе.

GET /exports/rss/(**string:** *project*)**/**
string: *component***/** Возвращает RSS-ленту с последними изменениями в компоненте.

GET /exports/rss/(**string:** *project*)**/**
Возвращает RSS-ленту с последними изменениями в проекте.

GET /exports/rss/language/(**string:** *language*)**/**
Возвращает RSS-ленту с последними изменениями в языке.

GET /exports/rss/
Возвращает RSS-ленту с последними изменениями в экземпляре Weblate.

См.также:

RSS в Википедии

1.13 Клиент Weblate

Добавлено в версии 2.7: Полная поддержка вспомогательной команды wlc существует с версии Weblate 2.7. Если вы используете более старую версию, могут возникнуть некоторые несовместимости с API.

1.13.1 Установка

Клиент Weblate поставляется отдельно и включает в себя модуль Python. Для использования нижеприведённых команд необходимо установить модуль `wlc`:

```
pip3 install wlc
```

1.13.2 Docker usage

The Weblate Client is also available as a Docker image.

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

Installing:

```
docker pull weblate/wlc
```

The Docker container uses Weblate's default settings and connects to the API deployed in localhost. The API URL and API_KEY can be configured through the arguments accepted by Weblate.

The command to launch the container uses the following syntax:

```
docker run --rm weblate/wlc [WLC_ARGS]
```

Пример:

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

1.13.3 Начало работы

Конфигурация wlc хранится в `~/.config/weblate` (другие возможные места смотрите в разделе *Файлы конфигурации*), пожалуйста, создайте её в соответствии со своим окружением:

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

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

После этого вы сможете вызывать команды на умолчательном сервере:

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

См.также:

Файлы конфигурации

1.13.4 Обзор

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

Какая операция должна быть выполнена на самом деле указывает параметр `command`.

1.13.5 Описание

Weblate Client — это библиотека Python и инструмент командной строки для удалённого управления Weblate с помощью его API. Инструмент командной строки может быть вызван как команда **wlc**, которая построена на основе модуля *wlc*.

Аргументы

Программа принимает следующие аргументы, определяющие формат вывода или то, какой надо использовать экземпляр Weblate'а. Они должны быть введены до какой-либо команды.

--format `{csv,json,text,html}`
Определяет формат вывода.

--url `URL`
Укажите URL-адрес API. Переопределяет любое значение, найденное в конфигурационном файле, смотрите раздел *Файлы конфигурации*. URL-адрес должен заканчиваться на `/api/`, например, `https://hosted.weblate.org/api/`.

--key `KEY`
Определяет пользовательский ключ API. Переопределяет любое значение, найденное в конфигурационном файле, смотрите раздел *Файлы конфигурации*. Свой ключ вы можете найти в своём профиле на Weblate.

--config `PATH`
Переопределяет путь к конфигурационному файлу, смотрите раздел *Файлы конфигурации*.

--config-section `SECTION`
Переопределяет используемый раздел конфигурационного файла, смотрите раздел *Файлы конфигурации*.

Команды

Доступны следующие команды:

version
Распечатывает текущую версию.

list-languages
Перечисляет используемые в Weblate языки.

list-projects
Перечисляет проекты в Weblate.

list-components
Перечисляет компоненты в Weblate.

list-translations
Перечисляет переводы в Weblate.

show
Показывает объект Weblate (перевод, компонент или проект).

ls
Перечисляет объект Weblate (перевод, компонент или проект).

commit
Фиксирует изменения, внесённые в объект Weblate (перевод, компонент или проект).

pull
Извлекает изменения из удалённого репозитория в объект Weblate (перевод, компонент или проект).

push
Отправляет изменения объекта Weblate (перевода, компонента или проекта) в удалённый репозиторий.

reset
Добавлено в версии 0.7: Поддерживается с версии wlc 0.7.

Сбрасывает изменения в объекте Weblate (переводе, компоненте или проекте) для приведения их в соответствие с удалённым репозиторием.

cleanup
Добавлено в версии 0.9: Поддерживается с версии wlc 0.9.

Удаляет любые неотслеживаемые изменения в объекте Weblate (переводе, компоненте или проекте) для приведения их в соответствие с удалённым репозиторием.

repo
Отображает статус репозитория для данного объекта Weblate (перевода, компонента или проекта).

statistics
Отображает подробную статистику по данному объекту Weblate (переводу, компоненту или проекту).

lock-status
Добавлено в версии 0.5: Поддерживается с версии wlc 0.5.

Отображает статус блокировки.

lock
Добавлено в версии 0.5: Поддерживается с версии wlc 0.5.

Блокирует компонент в Weblate от дальнейшего перевода.

unlock
Добавлено в версии 0.5: Поддерживается с версии wlc 0.5.

Разблокирует перевод компонента Weblate.

changes
Добавлено в версии 0.7: Поддерживается с версии wlc 0.7 и Weblate 2.10.

Отображает изменения для данного объекта.

download
Добавлено в версии 0.7: Поддерживается с версии wlc 0.7.

Скачивает файл перевода.

--convert
Преобразовывает формат файла, если не указан, преобразования на сервере не происходит и файл скачивается в том виде, в котором он сохранён в репозитории.

--output
Указывает файл, в который нужно сохранить вывод, если не задать, вывод будет распечатан в стандартный поток вывода.

upload
Добавлено в версии 0.9: Поддерживается с версии wlc 0.9.

Загружает файл перевода.

--overwrite
Перезаписать существующие переводы во время загрузки.

--input
Файл, из которого читается содержимое, если не задать, чтение будет производиться со стандартного потока ввода.

Подсказка: You can get more detailed information on invoking individual commands by passing `--help`, for example: `wlc ls --help`.

1.13.6 Файлы конфигурации

.weblate, .weblate.ini, weblate.ini Изменено в версии 1.6: The files with *.ini* extension are accepted as well.

Проектный файл конфигурации

C:\Users\NAME\AppData\weblate.ini Добавлено в версии 1.6.

Пользовательский файл конфигурации на Windows.

~/.config/weblate Пользовательский файл конфигурации

/etc/xdg/weblate Системный файл конфигурации

Программа следует спецификации XDG, поэтому вы можете скорректировать пути размещения конфигурационных файлов переменными окружения `XDG_CONFIG_HOME` или `XDG_CONFIG_DIRS`. В Windows предпочтительным местом для файла конфигурации является каталог `APPDATA`.

Можно настроить следующие параметры, находящиеся в разделе `[weblate]` (изменить этот раздел вы можете ключом командной строки *`--config-section`*):

key
Ключ API для доступа к Weblate.

url
Адрес API сервера, по умолчанию установлен в `http://127.0.0.1:8000/api/`.

translation
Путь к переводу по умолчанию — компоненту или проекту.

Файл конфигурации является INI-файлом, например:

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

Дополнительно ключи API могут быть сохранены в разделе `[keys]`:

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

Это позволяет вам сохранять ключи в ваших личных настройках, а конфигурацию `.weblate` в хранить в репозитории системы контроля версий, чтобы wlc знала, с каким сервером ей следует общаться.

1.13.7 Примеры

Печать текущей версии программы:

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

Список всех проектов:

```
$ 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/
```

Также вы можете указать, с каким проектом должен работать wlc:

```
$ cat .weblate
[weblate]
url = https://hosted.weblate.org/api/
translation = weblate/master

$ wlc show
branch: master
file_format: po
source_language: en
filemask: weblate/locale/*/LC_MESSAGES/django.po
git_export: https://hosted.weblate.org/git/weblate/master/
license: GPL-3.0+
license_url: https://spdx.org/licenses/GPL-3.0+
name: master
new_base: weblate/locale/django.pot
project: weblate
repo: git://github.com/WeblateOrg/weblate.git
slug: master
template:
url: https://hosted.weblate.org/api/components/weblate/master/
vcs: git
web_url: https://hosted.weblate.org/projects/weblate/master/
```

При такой настройке закоммитить отложенные изменения в текущем проекте проще простого:

```
$ wlc commit
```

1.14 Python API Weblate'a

1.14.1 Установка

Python API поставляется отдельно, чтобы он был у вас появился, вам нужно установить *клиент Weblate*: `wlc`.

```
pip install wlc
```

1.14.2 wlc

WeblateException

exception `wlc.`**`WeblateException`**

Базовый класс для всех исключений.

Weblate

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

Параметры

- **key** (*str*) – Ключ пользователя
- **url** (*str*) – URL-адрес сервера API, если не указан, используется адрес по умолчанию
- **config** (`wlc.config.WeblateConfig`) – Объект конфигурации, переопределяет любые другие параметры.

Класс доступа к API, определения ключа API и, необязательно, URL-адреса API.

get(*path*)

Параметры **path** (*str*) – Путь запроса

Тип результата object

Выполняет единичный GET-вызов API.

post(*path*, ***kwargs*)

Параметры **path** (*str*) – Путь запроса

Тип результата object

Выполняет единичный GET-вызов API.

1.14.3 wlc.config

WeblateConfig

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

Параметры **section** (*str*) – Используемый раздел конфигурации

Анализатор файлов конфигурации, следующий спецификации XDG.

load(*path=None*)

Параметры **path** (*str*) – Путь к файлу, из которого загружается конфигурация.

Загружает конфигурацию из файла, если файл не указан, загружает её из конфигурационного файла *wlc* (`~/.config/wlc`), расположенного по пути вашей конфигурации XDG (`/etc/xdg/wlc`).

1.14.4 wlc.main

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

Параметры

- **settings** (*list*) – Параметры для переопределения в виде списка кортежей
- **stdout** (*object*) – Файловый объект стандартного потока вывода печати вывода, по умолчанию используется `sys.stdout`
- **args** (*list*) – Аргументы командной строки для обработки, по умолчанию используется `sys.args`

Главная точка входа для интерфейса командной строки.

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

Декоратор для регистрации класса *Command* в основном анализаторе, используемом функцией *main()*.

Command

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

Главный класс для вызова команд.

ГЛАВА 2

Документация для администратора

2.1 Инструкции по настройке

2.1.1 Установка Weblate

Установка с помощью Docker'а

С помощью докеризированного развёртывания Weblate вы можете запустить ваш личный экземпляр Weblate за считанные секунды. Все зависимости Weblate уже включены в образ. В качестве базы данных по умолчанию настроена PostgreSQL.

Требования к оборудованию

Weblate должен без проблем работать на всём современном оборудовании, ниже приведена минимальная конфигурация, необходимая для запуска Weblate на одном узле (сам Weblate, база данных и веб-сервер):

- 2 ГБ оперативной памяти
- 2-х ядерный процессор
- 1 ГБ дискового пространства

Чем больше памяти, тем лучше — она используется для кэширования на всех уровнях (на уровне файловой системы, уровне базы данных и уровне Weblate).

Множество одновременно работающих пользователей увеличивают количество необходимых ядер процессора. Для сотен компонентов перевода рекомендуется не менее 4 Гб оперативной памяти.

Примечание: Фактические требования к вашей установке Weblate сильно зависят от размера управляемых ею переводов.

Установка

В следующих примерах предполагается, что у вас есть рабочее окружение Docker'а, в котором установлен `docker-compose`. Для получения конкретных инструкций ознакомьтесь с документацией к Docker'у.

1. Склонируйте репозиторий weblate-docker:

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

2. Создайте файл `docker-compose.override.yml` со своими настройками. Полный список переменных окружения приведён в разделе *Переменные окружения Docker'а*.

```
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
```

Примечание: Если переменная *WEBLATE_ADMIN_PASSWORD* не установлена, то пользователь-администратор создаётся со случайным паролем, отображаемым при первом запуске.

The provided example makes Weblate listen on port 80, edit the port mapping in the `docker-compose.override.yml` file to change it.

3. Запустите контейнеры Weblate:

```
docker-compose up
```

Наслаждайтесь своим развёрнутым Weblate'ом, он доступен на порту 80 контейнера `weblate`.

Изменено в версии 2.15-2: Недавно настройка изменилась, ранее существовал отдельный контейнер веб-сервера, но с версии 2.15-2 веб-сервер встроен в контейнер Weblate.

Изменено в версии 3.7.1-6: С июля 2019 года (начиная с метки 3.7.1-6) контейнеры больше не запускают своё содержимое под пользователем root. Это изменение привело к изменению открытого порта с 80 на 8080.

См.также:

Вызов команд управления

Контейнер Docker с поддержкой HTTPS

Общие инструкции по развертыванию, пожалуйста, смотрите в разделе *Установка*, в этом разделе упоминаются только отличия от той инструкции.

Использование собственных SSL-сертификатов

Добавлено в версии 3.8-3.

Если вы хотите использовать свой собственный SSL-сертификат, просто поместите его файлы в том данных Weblate (смотрите раздел *Тома контейнеров Docker'а*):

- `ssl/fullchain.pem`, содержащий сертификат и все необходимые сертификаты центров сертификации
- `ssl/privkey.pem`, содержащий закрытый ключ

Обеими этими файлами должен владеть тот же самый пользователь, который запускает контейнер docker 'а, а маска этих файлов должна быть равна `600` (читать и писать в них может только владелец).

Кроме того, контейнер Weblate теперь будет принимать SSL-соединения на порт 4443 и вам нужно будет включить перенаправление портов для HTTPS в переопределении Docker Compose:

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

Если на этом же сервере у вас уже размещены другие сайты, скорее всего, порты `80` и `443` используются обратным прокси, например, NGINX. Для передачи HTTPS соединения из NGINX'а в контейнер docker 'а вы можете использовать следующую конфигурацию:

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

    server_name <SITE_URL>;
    ssl_certificate /etc/letsencrypt/live/<SITE>/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/<SITE>/privkey.pem;

    location / {
            proxy_set_header HOST $host;
            proxy_set_header X-Forwarded-Proto https;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Host $server_name;
            proxy_pass https://127.0.0.1:<EXPOSED_DOCKER_PORT>;
    }
}
```

Замените заполнители `<SITE_URL>`, `<SITE>` и `EXPOSED_DOCKER_PORT` на фактические значения из вашего окружения.

Автоматический выпуск SSL-сертификатов с помощью сервиса Let's Encrypt

Если вы хотите использовать Let's Encrypt для автоматической генерации SSL-сертификатов для общедоступной установки, вам нужно будет добавить дополнительный контейнер Docker с обратным HTTPS прокси — https-portal. Он используется в файле `docker-compose-https.yml`. Затем создайте файл `docker-compose-https.override.yml` со своими настройками:

```
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'
```

При каждом вызове **`docker-compose`** вам нужно будет передавать ему оба этих файла, а затем сделать следующее:

```
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
```

Обновление контейнера Docker'а

Обычно рекомендуется обновлять только контейнер Weblate, а контейнер PostgreSQL сохранять той версии, которая у вас есть, поскольку обновление PostgreSQL довольно болезненно и в большинстве случаев не приносит особых преимуществ.

Вы можете сделать это, оставаясь с существующим docker-compose и просто извлекая последние образы с последующим перезапуском контейнера:

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

При первом запуске база данных Weblate должна быть перенесена автоматически, надобности в дополнительных ручных действиях не возникнет.

Примечание: Обновления через версию 3.0 Weblate не поддерживает. Если вы сидите на версии 2.x и хотите обновиться до версии 3.x, сначала выполните обновление до последней версии 3.0.1-x (на момент написания этого документа это образ `3.0.1-7`), который выполнит миграцию, а затем продолжит обновление до более новых версий.

Также вы можете захотеть обновить репозиторий `docker-compose`, хотя в большинстве случаев необходимости в этом нет. Пожалуйста, остерегайтесь в этом случае изменений версии PostgreSQL, поскольку обновление базы данных является не простым делом, более подробную информацию смотрите в этом замечании на GitHub'е.

Вход под администратором

После настройки контейнера вы можете войти в систему под пользователем *admin* с паролем, указанным в переменной окружения `WEBLATE_ADMIN_PASSWORD`, или случайным паролем, сгенерированным при первом запуске, если он не был установлен явно.

Для сброса пароля пользователя *admin* перезапустите контейнер с переменной `WEBLATE_ADMIN_PASSWORD`, установленной в новое значение пароля.

См.также:

`WEBLATE_ADMIN_PASSWORD`, `WEBLATE_ADMIN_NAME`, `WEBLATE_ADMIN_EMAIL`

Переменные окружения Docker'a

Многие *конфигурационные настройки* Weblate в контейнере Docker могут быть установлены через переменные окружения:

Общие параметры

WEBLATE_DEBUG

Настраивает отладочный режим Django, используя её переменную `DEBUG`.

Пример:

```
environment:
  WEBLATE_DEBUG: 1
```

См.также:

Отключение отладочного режима.

WEBLATE_LOGLEVEL

Настраивает подробность ведения журнала.

WEBLATE_SITE_TITLE

Изменяет название сайта, показываемое в заголовке всех страниц.

WEBLATE_SITE_DOMAIN

Настраивает домен сайта.

Подсказка: В случае, если он не установлен, используется первый элемент из переменной окружения `WEBLATE_ALLOWED_HOSTS`.

См.также:

Установка правильного домена сайта, `SITE_DOMAIN`

WEBLATE_ADMIN_NAME

WEBLATE_ADMIN_EMAIL

Настраивает имя и электронную почту администратора сайта. Используется как для параметра `ADMINS`, так и для создания пользователя *admin* (подробнее смотрите в описании переменной окружения `WEBLATE_ADMIN_PASSWORD`).

Пример:

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

См.также:

Вход под администратором, *Правильная настройка администраторов*, `ADMINS`

WEBLATE_ADMIN_PASSWORD

Устанавливает пароль для пользователя *admin*.

- Если не установлен и пользователя *admin* не существует, он создаётся со случайным паролем, который показывается при первом запуске контейнера.
- Если не установлен и пользователя *admin* не существует, никакие действия не выполняются.
- Если установлен, при каждом запуске контейнера пользователю *admin* устанавливаются соответствующие пароль `WEBLATE_ADMIN_PASSWORD`, имя `WEBLATE_ADMIN_NAME` и адрес электронной почты `WEBLATE_ADMIN_EMAIL`.

Предупреждение: Хранение пароля в файле конфигурации может представлять собой угрозу безопасности. Подумайте об использовании этой переменной только для начальной установки (или пусть Weblate сгенерирует при начальной загрузке случайный пароль) или для восстановления пароля.

См.также:

Вход под администратором, `WEBLATE_ADMIN_PASSWORD`, `WEBLATE_ADMIN_NAME`, `WEBLATE_ADMIN_EMAIL`

WEBLATE_SERVER_EMAIL

WEBLATE_DEFAULT_FROM_EMAIL

Настраивает адрес для исходящих сообщений электронной почты.

См.также:

Настройка отправки электронной почты

WEBLATE_ALLOWED_HOSTS

Настраивает разрешённые имена HTTP-хостов с помощью параметра `ALLOWED_HOSTS`.

По умолчанию установлен в `*`, что позволяет использовать все имена хостов.

Пример:

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

См.также:

`ALLOWED_HOSTS`, *Настройка разрешённых хостов*, *Установка правильного домена сайта*

WEBLATE_REGISTRATION_OPEN

Настраивает статус открытия регистрации, переключая параметр `REGISTRATION_OPEN`.

Пример:

```
environment:
  WEBLATE_REGISTRATION_OPEN: 0
```

WEBLATE_REGISTRATION_ALLOW_BACKENDS

Настраивает через параметр `REGISTRATION_ALLOW_BACKENDS` методы аутентификации, которые можно использовать для создания новой учётной записи.

Пример:

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

(continues on next page)

(продолжение с предыдущей страницы)

WEBLATE_TIME_ZONE

Настраивает используемый в Weblate часовой пояс, смотрите описание параметра `TIME_ZONE`.

Примечание: Для изменения часового пояса самого контейнера Docker используйте переменную окружения `TZ`.

Пример:

```
environment:
  WEBLATE_TIME_ZONE: Europe/Prague
```

WEBLATE_ENABLE_HTTPS

Заставляет Weblate думать, что он работает за обратным HTTPS-прокси, что принуждает Weblate использовать HTTPS в ссылках писем электронной почты и API или устанавливать у кук флаги безопасности.

Примечание: Этот параметр не включает разрешение на приём контейнером Weblate'а соединений по HTTPS, его нужно настроить отдельно, для примеров смотрите раздел *Контейнер Docker с поддержкой HTTPS*.

Пример:

```
environment:
  WEBLATE_ENABLE_HTTPS: 1
```

См.также:

Установка правильного домена сайта

WEBLATE_IP_PROXY_HEADER

Позволяет Weblate'у получать IP-адрес из любого заданного HTTP-заголовка. Используйте его при использовании обратного прокси перед контейнером Weblate.

Включает параметр *IP_BEHIND_REVERSE_PROXY* и устанавливает параметр *IP_PROXY_HEADER*.

Примечание: Формат параметра должен соответствовать ожиданиям Django. Django преобразует необработанные имена HTTP-заголовков следующим образом:

- переводит все символы в верхний регистр
- все дефисы заменяет на подчеркивания
- добавляет префикс `` HTTP_``

Таким образом, заголовок `X-Forwarded-For` отображается на ``HTTP_X_FORWARDED_FOR`.

Пример:

```
environment:
  WEBLATE_IP_PROXY_HEADER: HTTP_X_FORWARDED_FOR
```

WEBLATE_SECURE_PROXY_SSL_HEADER

Кортеж, представляющий собой комбинацию HTTP-заголовков/значений, указывающую, что запрос является безопасным. Он необходим, когда Weblate работает за обратным прокси, выполняющим SSL-терминацию, которая не передаёт стандартные HTTPS-заголовки.

Пример:

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

См.также:

SECURE_PROXY_SSL_HEADER

WEBLATE_REQUIRE_LOGIN

Настраивает логин, необходимый для установки Weblate, используя параметр *LOGIN_REQUIRED_URLS*.

Пример:

```
environment:
  WEBLATE_REQUIRE_LOGIN: 1
```

WEBLATE_LOGIN_REQUIRED_URLS_EXCEPTIONS

WEBLATE_ADD_LOGIN_REQUIRED_URLS_EXCEPTIONS

WEBLATE_REMOVE_LOGIN_REQUIRED_URLS_EXCEPTIONS

Добавляет исключения URL-адресов для логина, необходимого для установки Weblate, используя параметр *LOGIN_REQUIRED_URLS_EXCEPTIONS*.

Вы можете либо заменить параметры астройки целиком, либо изменить значение по умолчанию с помощью переменных ADD и REMOVE.

WEBLATE_GOOGLE_ANALYTICS_ID

Настраивает идентификатор для Google Analytics, изменяя параметр *GOOGLE_ANALYTICS_ID*.

WEBLATE_GITHUB_USERNAME

Настраивает имя пользователя GitHub для GitHub'овских запросов на извлечение, изменяя параметр *GITHUB_USERNAME*.

См.также:

GitHub

WEBLATE_GITHUB_TOKEN

Добавлено в версии 4.3.

Настраивает GitHub'овский токен персонального доступа для GitHub'овских запросов на извлечение через API, изменяя параметр *GITHUB_TOKEN*.

См.также:

GitHub

WEBLATE_GITLAB_USERNAME

Настраивает имя пользователя GitLab для GitLab'овских запросов на слияние, изменяя параметр *GITLAB_USERNAME*

См.также:

GitLab

WEBLATE_GITLAB_TOKEN

Настраивает GitLab'овский токен персонального доступа для GitLab'овских запросов на слияние через API, изменяя параметр *GITLAB_TOKEN*

См.также:

GitLab

WEBLATE_SIMPLIFY_LANGUAGES

Настраивает политику упрощения языка, смотрите описание параметра *SIMPLIFY_LANGUAGES*.

WEBLATE_DEFAULT_ACCESS_CONTROL
Настраивает значение по умолчанию для *управления доступом* в новых проектах, смотрите описание параметра *DEFAULT_ACCESS_CONTROL*.

WEBLATE_DEFAULT_RESTRICTED_COMPONENT
Настраивает значение по умолчанию для *ограниченного доступа* в новых компонентах, смотрите описание параметра *DEFAULT_RESTRICTED_COMPONENT*.

WEBLATE_DEFAULT_TRANSLATION_PROPAGATION
Настраивает значение по умолчанию для *разрешения распространения переводов* в новых компонентах, смотрите описание параметра *DEFAULT_TRANSLATION_PROPAGATION*.

WEBLATE_DEFAULT_COMMITER_EMAIL
Настраивает параметр *DEFAULT_COMMITER_EMAIL*.

WEBLATE_DEFAULT_COMMITER_NAME
Настраивает параметр *DEFAULT_COMMITER_NAME*.

WEBLATE_AKISMET_API_KEY
Настраивает API-ключ Akismet, смотрите описание параметра *AKISMET_API_KEY*.

WEBLATE_GPG_IDENTITY
Настраивает GPG для подписи коммитов, смотрите описание параметра *WEBLATE_GPG_IDENTITY*.

См.также:

Подписание коммитов Git с помощью GnuPG

WEBLATE_URL_PREFIX
Настраивает префикс URL-адреса, на котором запущен Weblate, смотрите описание параметра *URL_PREFIX*.

WEBLATE_SILENCED_SYSTEM_CHECKS
Настраивает проверки, которые вы не хотите видеть, смотрите описание параметра `SILENCED_SYSTEM_CHECKS`.

WEBLATE_CSP_SCRIPT_SRC

WEBLATE_CSP_IMG_SRC

WEBLATE_CSP_CONNECT_SRC

WEBLATE_CSP_STYLE_SRC

WEBLATE_CSP_FONT_SRC
Позволяет настраивать HTTP-заголовок политики безопасности содержимого `Content-Security-Policy`.

См.также:

Политика безопасности содержимого, *CSP_SCRIPT_SRC*, *CSP_IMG_SRC*, *CSP_CONNECT_SRC*, *CSP_STYLE_SRC*, *CSP_FONT_SRC*

WEBLATE_LICENSE_FILTER
Configures *LICENSE_FILTER*.

Параметры машинного перевода

WEBLATE_MT_APERTIUM_APY
Включает машинный перевод от *Apertium* и устанавливает параметр `MT_APERTIUM_APY`

WEBLATE_MT_AWS_REGION

WEBLATE_MT_AWS_ACCESS_KEY_ID

WEBLATE_MT_AWS_SECRET_ACCESS_KEY
Настраивает машинный перевод от *AWS*.

```
environment:
  WEBLATE_MT_AWS_REGION: us-east-1
  WEBLATE_MT_AWS_ACCESS_KEY_ID: AKIAIOSFODNN7EXAMPLE
  WEBLATE_MT_AWS_SECRET_ACCESS_KEY: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

WEBLATE_MT_DEEPL_KEY
Включает машинный перевод от *DeepL* и устанавливает параметр `MT_DEEPL_KEY`

WEBLATE_MT_DEEPL_API_VERSION
Настраивает используемую версию API *DeepL*, смотрите описание параметра `MT_DEEPL_API_VERSION`.

WEBLATE_MT_GOOGLE_KEY
Включает машинный перевод от *Google Translate* и устанавливает параметр `MT_GOOGLE_KEY`

WEBLATE_MT_MICROSOFT_COGNITIVE_KEY
Включает машинный перевод от *Переводчика Microsoft Cognitive Services* и устанавливает параметр `MT_MICROSOFT_COGNITIVE_KEY`

WEBLATE_MT_MICROSOFT_ENDPOINT_URL
Устанавливает параметр `MT_MICROSOFT_ENDPOINT_URL`, обратите внимание, что он должен содержать только доменное имя.

WEBLATE_MT_MICROSOFT_REGION
Устанавливает параметр `MT_MICROSOFT_REGION`

WEBLATE_MT_MICROSOFT_BASE_URL
Устанавливает параметр `MT_MICROSOFT_BASE_URL`

WEBLATE_MT_MODERNMT_KEY
Включает машинный перевод от *ModernMT* и устанавливает параметр `MT_MODERNMT_KEY`.

WEBLATE_MT_MYMEMORY_ENABLED

Включает машинный перевод от *MyMemory* и устанавливает параметр `MT_MYMEMORY_EMAIL` в значение переменной окружения `WEBLATE_ADMIN_EMAIL`.

Пример:

```
environment:
  WEBLATE_MT_MYMEMORY_ENABLED: 1
```

WEBLATE_MT_GLOSBE_ENABLED
Включает машинный перевод от *Glosbe*.

```
environment:
  WEBLATE_MT_GLOSBE_ENABLED: 1
```

WEBLATE_MT_MICROSOFT_TERMINOLOGY_ENABLED
Включает машинный перевод от *терминологической службы Майкрософт*.

```
environment:
  WEBLATE_MT_MICROSOFT_TERMINOLOGY_ENABLED: 1
```

WEBLATE_MT_SAP_BASE_URL

WEBLATE_MT_SAP_SANDBOX_APIKEY

WEBLATE_MT_SAP_USERNAME

WEBLATE_MT_SAP_PASSWORD

WEBLATE_MT_SAP_USE_MT

Настраивает машинный перевод от *SAP Translation Hub*.

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

Параметры аутентификации

LDAP

WEBLATE_AUTH_LDAP_SERVER_URI

WEBLATE_AUTH_LDAP_USER_DN_TEMPLATE

WEBLATE_AUTH_LDAP_USER_ATTR_MAP

WEBLATE_AUTH_LDAP_BIND_DN

WEBLATE_AUTH_LDAP_BIND_PASSWORD

WEBLATE_AUTH_LDAP_CONNECTION_OPTION_REFERRALS

WEBLATE_AUTH_LDAP_USER_SEARCH

WEBLATE_AUTH_LDAP_USER_SEARCH_FILTER

WEBLATE_AUTH_LDAP_USER_SEARCH_UNION

WEBLATE_AUTH_LDAP_USER_SEARCH_UNION_DELIMITER

Конфигурация аутентификации через LDAP.

Пример прямой привязки:

```
environment:
  WEBLATE_AUTH_LDAP_SERVER_URI: ldap://ldap.example.org
  WEBLATE_AUTH_LDAP_USER_DN_TEMPLATE: uid=%(user)s,ou=People,dc=example,dc=net
  # map weblate 'full_name' to ldap 'name' and weblate 'email' attribute to
↪'mail' ldap attribute.
  # another example that can be used with OpenLDAP: 'full_name:cn,email:mail'
  WEBLATE_AUTH_LDAP_USER_ATTR_MAP: full_name:name,email:mail
```

Пример поиска и привязки:

```
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
```

Пример объединения поиска и привязки:

```
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
```

Пример поиска и привязки в Active Directory:

```
environment:
  WEBLATE_AUTH_LDAP_BIND_DN: CN=ldap,CN=Users,DC=example,DC=com
  WEBLATE_AUTH_LDAP_BIND_PASSWORD: password
  WEBLATE_AUTH_LDAP_SERVER_URI: ldap://ldap.example.org
  WEBLATE_AUTH_LDAP_CONNECTION_OPTION_REFERRALS: 0
  WEBLATE_AUTH_LDAP_USER_ATTR_MAP: full_name:name,email:mail
  WEBLATE_AUTH_LDAP_USER_SEARCH: CN=Users,DC=example,DC=com
  WEBLATE_AUTH_LDAP_USER_SEARCH_FILTER: (sAMAccountName=%(user)s)
```

См.также:

Аутентификация через LDAP

GitHub

WEBLATE_SOCIAL_AUTH_GITHUB_KEY

WEBLATE_SOCIAL_AUTH_GITHUB_SECRET

Включает *аутентификацию через GitHub*.

Bitbucket

WEBLATE_SOCIAL_AUTH_BITBUCKET_KEY

WEBLATE_SOCIAL_AUTH_BITBUCKET_SECRET

Включает *аутентификацию через Bitbucket*.

Facebook

WEBLATE_SOCIAL_AUTH_FACEBOOK_KEY

WEBLATE_SOCIAL_AUTH_FACEBOOK_SECRET

Включает *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

Включает *Google OAuth 2*.

GitLab

WEBLATE_SOCIAL_AUTH_GITLAB_KEY

WEBLATE_SOCIAL_AUTH_GITLAB_SECRET

WEBLATE_SOCIAL_AUTH_GITLAB_API_URL

Включает *GitLab OAuth 2*.

Azure Active Directory

WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_KEY

WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_SECRET

Включает аутентификацию через Azure Active Directory, смотрите раздел *Microsoft Azure Active Directory*.

Azure Active Directory с поддержкой Tenant

WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_KEY

WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_SECRET

WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_TENANT_ID

Включает аутентификацию через Azure Active Directory с поддержкой Tenant, смотрите раздел *Microsoft Azure Active Directory*.

Keycloak

WEBLATE_SOCIAL_AUTH_KEYCLOAK_KEY

WEBLATE_SOCIAL_AUTH_KEYCLOAK_SECRET

WEBLATE_SOCIAL_AUTH_KEYCLOAK_PUBLIC_KEY

WEBLATE_SOCIAL_AUTH_KEYCLOAK_ALGORITHM

WEBLATE_SOCIAL_AUTH_KEYCLOAK_AUTHORIZATION_URL

WEBLATE_SOCIAL_AUTH_KEYCLOAK_ACCESS_TOKEN_URL

Включает аутентификацию через Keycloak, смотрите документацию.

Поставщики Linux

Вы можете включить аутентификацию с помощью сервисов аутентификации поставщиков Linux, установив следующие переменные в любое значение.

WEBLATE_SOCIAL_AUTH_FEDORA

WEBLATE_SOCIAL_AUTH_OPENSUSE

WEBLATE_SOCIAL_AUTH_UBUNTU

Slack

WEBLATE_SOCIAL_AUTH_SLACK_KEY

SOCIAL_AUTH_SLACK_SECRET
Включает аутентификацию через Slack, смотрите раздел *Slack*.

SAML

Самоподписанные ключи SAML генерируются автоматически при первом запуске контейнера. Если вы хотите использовать собственные ключи, поместите сертификат и закрытый ключ в файлы `/app/data/ssl/saml.crt` и `/app/data/ssl/saml.key`.

WEBLATE_SAML_IDP_ENTITY_ID

WEBLATE_SAML_IDP_URL

WEBLATE_SAML_IDP_X509CERT
Настройки провайдера идентификационных данных SAML, смотрите раздел *Аутентификация через SAML*.

Другие параметры аутентификации

WEBLATE_NO_EMAIL_AUTH
При установке в любое значение отключает аутентификацию через электронную почту.

Настройки базы данных PostgreSQL

База данных создается в файле `docker-compose.yml`, поэтому эти настройки влияют как на контейнер Weblate, так и на контейнер PostgreSQL.

См.также:

Настройка базы данных для Weblate

POSTGRES_PASSWORD
Пароль PostgreSQL.

POSTGRES_USER
Имя пользователя PostgreSQL.

POSTGRES_DATABASE
Имя базы данных PostgreSQL.

POSTGRES_HOST
Имя хоста или IP-адрес сервера PostgreSQL. По умолчанию равен `database`.

POSTGRES_PORT
Порт сервера PostgreSQL. По умолчанию не установлен (используется значение по умолчанию).

POSTGRES_SSL_MODE
Настраивает обработку SSL сервером PostgreSQL при соединении с сервером, возможные варианты настройки смотрите документе Описания режимов SSL

Параметры резервного копирования базы данных

См.также:

Дампы данных для резервного копирования

WEBLATE_DATABASE_BACKUP
Настраивает ежедневный дамп базы данных с помощью параметра `DATABASE_BACKUP`. По умолчанию установлен в `plain`.

Настройка сервера кэширования

Weblate настоятельно рекомендует использовать Redis и при запуске Weblate'a в Docker'e вы должны предоставить экземпляр Redis'a.

См.также:

Включение кэширования

REDIS_HOST
Имя хоста или IP-адрес сервера Redis. По умолчанию установлен в `cache`.

REDIS_PORT
Порт сервера Redis. По умолчанию установлен в `6379`.

REDIS_DB
Номер базы данных Redis, по умолчанию установлен в `1`.

REDIS_PASSWORD
Пароль сервера Redis, по умолчанию не используется.

REDIS_TLS
Включает использование SSL для соединения с Redis.

REDIS_VERIFY_SSL
Может использоваться для отключения проверки SSL-сертификата для соединениий с Redis.

Настройка почтового сервера

Чтобы работала отправка писем электронной почты, необходимо предоставить почтовый сервер.

Пример конфигурации TLS:

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

Пример конфигурации SSL:

```
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
```

См.также:

Настройка исходящей почты

WEBLATE_EMAIL_HOST
Имя хоста или IP-адрес почтового сервера.

См.также:

WEBLATE_EMAIL_PORT, *WEBLATE_EMAIL_USE_SSL*, *WEBLATE_EMAIL_USE_TLS*, EMAIL_HOST

WEBLATE_EMAIL_PORT
Порт почтового сервера, по умолчанию установлен в 25.

См.также:

EMAIL_PORT

WEBLATE_EMAIL_HOST_USER
Пользователь для аутентификации по электронной почте.

См.также:

EMAIL_HOST_USER

WEBLATE_EMAIL_HOST_PASSWORD
Пароль для аутентификации по электронной почте.

См.также:

EMAIL_HOST_PASSWORD

WEBLATE_EMAIL_USE_SSL
Использовать ли неявное TLS (безопасное) соединение при общении с сервером SMTP. В большей части документации по электронной почте этот тип TLS-соединения называется SSL. Он обычно используется на 465-м порту. Если у вас возникли проблемы, попробуйте включить явный TLS параметром *WEBLATE_EMAIL_USE_TLS*.

См.также:

WEBLATE_EMAIL_PORT, *WEBLATE_EMAIL_USE_TLS*, EMAIL_USE_SSL

WEBLATE_EMAIL_USE_TLS
Использовать ли TLS (безопасное) соединение при общении с сервером SMTP. Он используется для явных TLS-соединений, обычно на портах 587 или 25. Если ваши соединения зависают, попробуйте включить неявный TLS параметром *WEBLATE_EMAIL_USE_SSL*.

См.также:

WEBLATE_EMAIL_PORT, *WEBLATE_EMAIL_USE_SSL*, EMAIL_USE_TLS

WEBLATE_EMAIL_BACKEND
Настраивает бэкенд Django, используемый для отправки сообщений электронной почты.

См.также:

Настройка отправки электронной почты, EMAIL_BACKEND

Отчёты об ошибках

Рекомендуется систематически собирать ошибки от установки, смотрите раздел *Сбор отчётов об ошибках*.

Для включения поддержки Rollbar установите следующие переменные:

ROLLBAR_KEY
Ваш токен доступа к серверу Rollbar.

ROLLBAR_ENVIRONMENT
Ваше окружение Rollbar, по умолчанию установлена в `production`.

Для включения поддержки Sentry установите следующие переменные:

SENTRY_DSN
Ваш DSN Sentry.

SENTRY_ENVIRONMENT
Ваше окружение Sentry (необязательная).

CDN локализации

WEBLATE_LOCALIZE_CDN_URL

WEBLATE_LOCALIZE_CDN_PATH
Добавлено в версии 4.2.1.

Настройки для *CDN локализации JavaScript'а*.

Переменная `WEBLATE_LOCALIZE_CDN_PATH` указывает путь внутри контейнера. Он должен указывать на постоянный том, а не на временное хранилище.

Одной из возможных настроек является направление пути внутрь каталога данных Weblate:

```
environment:
  WEBLATE_LOCALIZE_CDN_URL: https://cdn.example.com/
  WEBLATE_LOCALIZE_CDN_PATH: /app/data/l10n-cdn
```

Примечание: За раздачу файлов, генерируемых Weblate, отвечаете вы, эта переменная только указывает местоположение, куда их надо сохранять.

См.также:

Перевод HTML и JavaScript с помощью сети доставки содержимого Weblate, `LOCALIZE_CDN_URL`, `LOCALIZE_CDN_PATH`

Изменение включённых приложений, проверок, надстроек или автоматических исправлений

Добавлено в версии 3.8-5.

Встроенную конфигурация включённых проверок, надстроек или автоматических исправлений можно настроить с помощью следующих переменных:

WEBLATE_ADD_APPS

WEBLATE_REMOVE_APPS

WEBLATE_ADD_CHECK

WEBLATE_REMOVE_CHECK

WEBLATE_ADD_AUTOFIX

WEBLATE_REMOVE_AUTOFIX

WEBLATE_ADD_ADDONS

WEBLATE_REMOVE_ADDONS

Пример:

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

См.также:

CHECK_LIST, *AUTOFIX_LIST*, *WEBLATE_ADDONS*, INSTALLED_APPS

Параметры контейнера

CELERY_MAIN_OPTIONS

CELERY_NOTIFY_OPTIONS

CELERY_TRANSLATE_OPTIONS

CELERY_BACKUP_OPTIONS

CELERY_BEAT_OPTIONS

Эти переменные позволяют настроить параметры рабочего Celery. Может быть полезно подправить степень параллелизма (`--concurrency 16`) или включить использование другой реализации пула (`--pool=gevent`).

По умолчанию количество параллельных рабочих соответствует количеству процессоров (за исключением рабочего создания резервных копий, который должен быть запущен в единственном экземпляре).

Пример:

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

См.также:

Параметры рабочих Celery, *Фоновые задачи с использованием Celery*

UWSGI_WORKERS

Настраивает количество выполняемых рабочих процессов uWSGI.

По умолчанию установлена в количество процессоров + 1.

Пример:

```
environment:
  UWSGI_WORKERS: 32
```

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

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

Тома контейнеров Docker'а

Контейнер Weblate экспортирует только один том данных. Контейнеры других сервисов (PostgreSQL или Redis) имеют свои собственные тома данных, которые в настоящем документе не рассматриваются.

Том данных используется для хранения таких постоянных данных Weblate, как склонированные репозитории, или для настройки установки Weblate.

Размещение тома Docker'а на хост-системе зависит от конфигурации Docker'а, но обычно он хранится в каталоге `/var/lib/docker/volumes/weblate-docker_weblate-data/_data/`. В контейнере он монтируется под именем `/app/data`.

См.также:

Документация по томам Docker'а

Дальнейшая настройка конфигурации

Вы можете и дальше настраивать установку Weblate в томе данных, смотрите раздел *Тома контейнеров Docker'a*.

Пользовательские файлы конфигурации

Дополнительно вы можете переопределить конфигурацию в файле `/app/data/settings-override.py` (смотрите раздел *Тома контейнеров Docker'a*). Этот файл выполняется после загрузки всех настроек окружения, так что оно полностью готово к работе и может быть использовано для настройки чего угодно.

Замена логотипа и других статических файлов

Добавлено в версии 3.8-5.

Статические файлы, поставляемые с Weblate, можно переопределить, поместив их в каталог `/app/data/python/customize/static` (смотрите раздел *Тома контейнеров Docker'a*). Например, создав файл `/app/data/python/customize/static/favicon.ico`, вы замените им иконку сайта.

Подсказка: При запуске контейнера файлы копируются в соответствующее место, поэтому после изменения содержимого тома Weblate требуется перезапустить.

В качестве альтернативы вы также можете включить свой собственный модуль (смотрите раздел *Настройка Weblate*) и добавить его в контейнер Docker'a отдельным томом, например:

```
weblate:
  volumes:
    - weblate-data:/app/data
    - ./weblate_customization/weblate_customization:/app/data/python/weblate_
↪customization
  environment:
    WEBLATE_ADD_APPS: weblate_customization
```

Добавление собственных модулей Python

Добавлено в версии 3.8-5.

Собственные модули Python вы можете разместить в каталоге `/app/data/python/` (смотрите раздел *Тома контейнеров Docker'a*) — Weblate их затем загрузит, скорее всего, через *пользовательские файлы конфигурации*.

См.также:

Настройка Weblate

Выбор вашей машины — локальные или облачные провайдеры

С помощью Docker Machine вы можете развернуть Weblate как на локальной машине, так и на любом большом количестве облачных развёртываний, например на Amazon AWS, Greenhost и на многих других провайдерах.

Установка на Debian и Ubuntu

Требования к оборудованию

Weblate должен без проблем работать на всём современном оборудовании, ниже приведена минимальная конфигурация, необходимая для запуска Weblate на одном узле (сам Weblate, база данных и веб-сервер):

- 2 ГБ оперативной памяти
- 2-х ядерный процессор
- 1 ГБ дискового пространства

Чем больше памяти, тем лучше — она используется для кэширования на всех уровнях (на уровне файловой системы, уровне базы данных и уровне Weblate).

Множество одновременно работающих пользователей увеличивают количество необходимых ядер процессора. Для сотен компонентов перевода рекомендуется не менее 4 Гб оперативной памяти.

Примечание: Фактические требования к вашей установке Weblate сильно зависят от размера управляемых ею переводов.

Установка

Системные требования

Установите зависимости, необходимые для сборки модулей Python (смотрите раздел *Требования к программному обеспечению*):

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

Установите желаемые необязательные зависимости в зависимости от функций, которые вы собираетесь использовать (смотрите раздел *Необязательные зависимости*):

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

При желании установите программное обеспечение для запуска рабочего сервера, смотрите разделы *Запуск сервера*, *Настройка базы данных для Weblate* и *Фоновые задачи с использованием Celery*. В зависимости от размера вашей установки вы можете захотеть запустить эти компоненты на выделенных серверах.

Инструкции по установке на местах:

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

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

(continues on next page)

(продолжение с предыдущей страницы)

```
# Caching backend: Redis
apt install redis-server

# Database server: PostgreSQL
apt install postgresql postgresql-contrib

# SMTP server
apt install exim4
```

Модули Python

Подсказка: Для установки Weblate мы используем виртуальное окружение, предоставляемое пакетом virtualenv, чтобы отделить Weblate от вашей системы. Если вы не знакомы с этим понятием, обратитесь к документации пакета.

1. Создайте виртуальное окружение для Weblate:

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

2. Активируйте виртуальное окружение для Weblate:

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

3. Установите Weblate, включая все его зависимости:

   ```
   pip install Weblate
   ```

4. Установите драйвер базы данных:

   ```
   pip install psycopg2-binary
   ```

5. Установите желаемые необязательные зависимости в зависимости от функций, которые вы собираетесь использовать (некоторые из них могут потребовать дополнительных системных библиотек, проверьте раздел *Необязательные зависимости*):

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

Настройка Weblate

Примечание: Следующие шаги подразумевают, что виртуальное окружение, используемое Weblate'ом, активно (активировать его можно командой `. ~/weblate-env/bin/activate`). В случае, если это не так, вам придётся указывать полный путь к команде **`weblate`**: `~/weblate-env/bin/weblate`.

1. Скопируйте файл `~/weblate-env/lib/python3.7/site-packages/weblate/settings_example.py` в `~/weblate-env/lib/python3.7/site-packages/weblate/settings.py`.
2. По своему усмотрению подкорректируйте значения в новом файле `settings.py`. Для тестирования вы можете использовать поставляемый пример, но для рабочей среды вы захотите внести в него изменения, смотрите раздел *Настройка конфигурации*.
3. Создайте базу данных и её структуру для Weblate (пример настройки использует PostgreSQL, для установки в рабочую среду ознакомьтесь разделом *Настройка базы данных для Weblate*):

```
weblate migrate
```

4. Создайте учётную запись администратора и скопируйте сгенерированный ей пароль в буфер обмена, также сохраните его для дальнейшего использования:

```
weblate createadmin
```

5. Соберите статические файлы для веб-сервера (смотрите разделы *Запуск сервера* и *Обслуживание статических файлов*):

```
weblate collectstatic
```

6. Сожмите файлы JavaScript и CSS (необязательно, смотрите раздел *Сжатие клиентских активов*):

```
weblate compress
```

7. Запустите рабочие процессы Celery. Для целей разработки это делать не обязательно, но настоятельно рекомендуется. Дополнительную информацию смотрите в разделе *Фоновые задачи с использованием Celery*:

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

8. Запустите сервер разработки (шаги для запуска рабочего сервера смотрите в разделе *Запуск сервера*):

```
weblate runserver
```

После установки

Поздравляем, теперь ваш сервер Weblate запущен и вы можете начать его использовать.

- Теперь вы можете получить доступ к Weblate по адресу `http://localhost:8000/`.
- Войдите в систему с учётными данными администратора, полученными во время установки, или зарегистрируйтесь в качестве нового пользователя.
- Теперь при помощи команды **`weblate`** и когда активно виртуальное окружение Weblate вы можете запускать его команды, смотрите раздел *Команды управления*.
- Остановить тестовый сервер можно комбинацией `Ctrl+C`.

Добавление перевода

1. Откройте интерфейс администратора (`http://localhost:8000/create/project/`) и создайте проект, который вы хотите перевести. Подробнее смотрите в разделе *Конфигурация проекта*.

   Всё, что вам нужно здесь указать — это название проекта и его веб-сайт.

2. Создайте компонент, который является реальным объектом для перевода — он указывает на репозиторий системы контроля версий и задаёт файлы для перевода. Подробнее смотрите в разделе *Конфигурация компонента*.

   Здесь важны следующие поля: название компонента, адрес репозитория системы контроля версий и маска для поиска переводимых файлов. Weblate поддерживает широкий спектр форматов, включая PO-файлы gettext, строки ресурсов Android'a, строки свойств iOS, файлы свойств Java или Qt Linguist, подробнее смотрите в разделе *Поддерживаемые форматы файлов*.

3. После того, как вышеперечисленные действия будут завершены (это может быть довольно длительный процесс в зависимости от размера вашего репозитория системы контроля версий и количества сообщений для перевода), вы сможете начать перевод.

Установка на SUSE и openSUSE

Требования к оборудованию

Weblate должен без проблем работать на всём современном оборудовании, ниже приведена минимальная конфигурация, необходимая для запуска Weblate на одном узле (сам Weblate, база данных и веб-сервер):

- 2 ГБ оперативной памяти
- 2-х ядерный процессор
- 1 ГБ дискового пространства

Чем больше памяти, тем лучше — она используется для кэширования на всех уровнях (на уровне файловой системы, уровне базы данных и уровне Weblate).

Множество одновременно работающих пользователей увеличивают количество необходимых ядер процессора. Для сотен компонентов перевода рекомендуется не менее 4 Гб оперативной памяти.

Примечание: Фактические требования к вашей установке Weblate сильно зависят от размера управляемых ею переводов.

Установка

Системные требования

Установите зависимости, необходимые для сборки модулей Python (смотрите раздел *Требования к программному обеспечению*):

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

Установите желаемые необязательные зависимости в зависимости от функций, которые вы собираетесь использовать (смотрите раздел *Необязательные зависимости*):

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

При желании установите программное обеспечение для запуска рабочего сервера, смотрите разделы *Запуск сервера*, *Настройка базы данных для Weblate* и *Фоновые задачи с использованием Celery*. В зависимости от размера вашей установки вы можете захотеть запустить эти компоненты на выделенных серверах.

Инструкции по установке на местах:

```
# 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
```

Модули Python

Подсказка: Для установки Weblate мы используем виртуальное окружение, предоставляемое пакетом virtualenv, чтобы отделить Weblate от вашей системы. Если вы не знакомы с этим понятием, обратитесь к документации пакета.

1. Создайте виртуальное окружение для Weblate:

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

2. Активируйте виртуальное окружение для Weblate:

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

3. Установите Weblate, включая все его зависимости:

```
pip install Weblate
```

4. Установите драйвер базы данных:

```
pip install psycopg2-binary
```

5. Установите желаемые необязательные зависимости в зависимости от функций, которые вы собираетесь использовать (некоторые из них могут потребовать дополнительных системных библиотек, проверьте раздел *Необязательные зависимости*):

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

Настройка Weblate

Примечание: Следующие шаги подразумевают, что виртуальное окружение, используемое Weblate'ом, активно (активировать его можно командой `. ~/weblate-env/bin/activate`). В случае, если это не так, вам придётся указывать полный путь к команде **`weblate`**: `~/weblate-env/bin/weblate`.

1. Скопируйте файл `~/weblate-env/lib/python3.7/site-packages/weblate/settings_example.py` в `~/weblate-env/lib/python3.7/site-packages/weblate/settings.py`.
2. По своему усмотрению подкорректируйте значения в новом файле `settings.py`. Для тестирования вы можете использовать поставляемый пример, но для рабочей среды вы захотите внести в него изменения, смотрите раздел *Настройка конфигурации*.
3. Создайте базу данных и её структуру для Weblate (пример настройки использует PostgreSQL, для установки в рабочую среду ознакомьтесь разделом *Настройка базы данных для Weblate*):

```
weblate migrate
```

4. Создайте учётную запись администратора и скопируйте сгенерированный ей пароль в буфер обмена, также сохраните его для дальнейшего использования:

```
weblate createadmin
```

5. Соберите статические файлы для веб-сервера (смотрите разделы *Запуск сервера* и *Обслуживание статических файлов*):

```
weblate collectstatic
```

6. Сожмите файлы JavaScript и CSS (необязательно, смотрите раздел *Сжатие клиентских активов*):

```
weblate compress
```

7. Запустите рабочие процессы Celery. Для целей разработки это делать не обязательно, но настоятельно рекомендуется. Дополнительную информацию смотрите в разделе *Фоновые задачи с использованием Celery*:

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

8. Запустите сервер разработки (шаги для запуска рабочего сервера смотрите в разделе *Запуск сервера*):

```
weblate runserver
```

После установки

Поздравляем, теперь ваш сервер Weblate запущен и вы можете начать его использовать.

- Теперь вы можете получить доступ к Weblate по адресу `http://localhost:8000/`.
- Войдите в систему с учётными данными администратора, полученными во время установки, или зарегистрируйтесь в качестве нового пользователя.
- Теперь при помощи команды **`weblate`** и когда активно виртуальное окружение Weblate вы можете запускать его команды, смотрите раздел *Команды управления*.
- Остановить тестовый сервер можно комбинацией `Ctrl+C`.

Добавление перевода

1. Откройте интерфейс администратора (`http://localhost:8000/create/project/`) и создайте проект, который вы хотите перевести. Подробнее смотрите в разделе *Конфигурация проекта*.

   Всё, что вам нужно здесь указать — это название проекта и его веб-сайт.

2. Создайте компонент, который является реальным объектом для перевода — он указывает на репозиторий системы контроля версий и задаёт файлы для перевода. Подробнее смотрите в разделе *Конфигурация компонента*.

   Здесь важны следующие поля: название компонента, адрес репозитория системы контроля версий и маска для поиска переводимых файлов. Weblate поддерживает широкий спектр форматов, включая PO-файлы gettext, строки ресурсов Android'a, строки свойств iOS, файлы свойств Java или Qt Linguist, подробнее смотрите в разделе *Поддерживаемые форматы файлов*.

3. После того, как вышеперечисленные действия будут завершены (это может быть довольно длительный процесс в зависимости от размера вашего репозитория системы контроля версий и количества сообщений для перевода), вы сможете начать перевод.

Установка на RedHat, Fedora и CentOS

Требования к оборудованию

Weblate должен без проблем работать на всём современном оборудовании, ниже приведена минимальная конфигурация, необходимая для запуска Weblate на одном узле (сам Weblate, база данных и веб-сервер):

- 2 ГБ оперативной памяти
- 2-х ядерный процессор
- 1 ГБ дискового пространства

Чем больше памяти, тем лучше — она используется для кэширования на всех уровнях (на уровне файловой системы, уровне базы данных и уровне Weblate).

Множество одновременно работающих пользователей увеличивают количество необходимых ядер процессора. Для сотен компонентов перевода рекомендуется не менее 4 Гб оперативной памяти.

Примечание: Фактические требования к вашей установке Weblate сильно зависят от размера управляемых ею переводов.

Установка

Системные требования

Установите зависимости, необходимые для сборки модулей Python (смотрите раздел *Требования к программному обеспечению*):

```
dnf install \
   libxslt-devel libxml2-devel freetype-devel libjpeg-devel zlib-devel libyaml-
↪devel \
   cairo-devel pango-devel gobject-introspection-devel libacl-devel \
   python3-pip python3-virtualenv python3-devel git
```

Установите желаемые необязательные зависимости в зависимости от функций, которые вы собираетесь использовать (смотрите раздел *Необязательные зависимости*):

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

При желании установите программное обеспечение для запуска рабочего сервера, смотрите разделы *Запуск сервера*, *Настройка базы данных для Weblate* и *Фоновые задачи с использованием Celery*. В зависимости от размера вашей установки вы можете захотеть запустить эти компоненты на выделенных серверах.

Инструкции по установке на местах:

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

# Web server option 2: Apache with ``mod_wsgi``
dnf install apache2 apache2-mod_wsgi

# Caching backend: Redis
dnf install redis

# Database server: PostgreSQL
dnf install postgresql postgresql-contrib

# SMTP server
dnf install postfix
```

Модули Python

Подсказка: Для установки Weblate мы используем виртуальное окружение, предоставляемое пакетом virtualenv, чтобы отделить Weblate от вашей системы. Если вы не знакомы с этим понятием, обратитесь к документации пакета.

1. Создайте виртуальное окружение для Weblate:

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

2. Активируйте виртуальное окружение для Weblate:

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

3. Установите Weblate, включая все его зависимости:

```
pip install Weblate
```

4. Установите драйвер базы данных:

```
pip install psycopg2-binary
```

5. Установите желаемые необязательные зависимости в зависимости от функций, которые вы собираетесь использовать (некоторые из них могут потребовать дополнительных системных библиотек, проверьте раздел *Необязательные зависимости*):

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

Настройка Weblate

Примечание: Следующие шаги подразумевают, что виртуальное окружение, используемое Weblate'ом, активно (активировать его можно командой `. ~/weblate-env/bin/activate`). В случае, если это не так, вам придётся указывать полный путь к команде **`weblate`**: `~/weblate-env/bin/weblate`.

1. Скопируйте файл `~/weblate-env/lib/python3.7/site-packages/weblate/settings_example.py` в `~/weblate-env/lib/python3.7/site-packages/weblate/settings.py`.
2. По своему усмотрению подкорректируйте значения в новом файле `settings.py`. Для тестирования вы можете использовать поставляемый пример, но для рабочей среды вы захотите внести в него изменения, смотрите раздел *Настройка конфигурации*.
3. Создайте базу данных и её структуру для Weblate (пример настройки использует PostgreSQL, для установки в рабочую среду ознакомьтесь разделом *Настройка базы данных для Weblate*):

```
weblate migrate
```

4. Создайте учётную запись администратора и скопируйте сгенерированный ей пароль в буфер обмена, также сохраните его для дальнейшего использования:

```
weblate createadmin
```

5. Соберите статические файлы для веб-сервера (смотрите разделы *Запуск сервера* и *Обслуживание статических файлов*):

```
weblate collectstatic
```

6. Сожмите файлы JavaScript и CSS (необязательно, смотрите раздел *Сжатие клиентских активов*):

```
weblate compress
```

7. Запустите рабочие процессы Celery. Для целей разработки это делать не обязательно, но настоятельно рекомендуется. Дополнительную информацию смотрите в разделе *Фоновые задачи с использованием Celery*:

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

8. Запустите сервер разработки (шаги для запуска рабочего сервера смотрите в разделе *Запуск сервера*):

```
weblate runserver
```

После установки

Поздравляем, теперь ваш сервер Weblate запущен и вы можете начать его использовать.

- Теперь вы можете получить доступ к Weblate по адресу `http://localhost:8000/`.
- Войдите в систему с учётными данными администратора, полученными во время установки, или зарегистрируйтесь в качестве нового пользователя.
- Теперь при помощи команды **`weblate`** и когда активно виртуальное окружение Weblate вы можете запускать его команды, смотрите раздел *Команды управления*.
- Остановить тестовый сервер можно комбинацией `Ctrl+C`.

Добавление перевода

1. Откройте интерфейс администратора (`http://localhost:8000/create/project/`) и создайте проект, который вы хотите перевести. Подробнее смотрите в разделе *Конфигурация проекта*.

   Всё, что вам нужно здесь указать — это название проекта и его веб-сайт.

2. Создайте компонент, который является реальным объектом для перевода — он указывает на репозиторий системы контроля версий и задаёт файлы для перевода. Подробнее смотрите в разделе *Конфигурация компонента*.

   Здесь важны следующие поля: название компонента, адрес репозитория системы контроля версий и маска для поиска переводимых файлов. Weblate поддерживает широкий спектр форматов, включая PO-файлы gettext, строки ресурсов Android'a, строки свойств iOS, файлы свойств Java или Qt Linguist, подробнее смотрите в разделе *Поддерживаемые форматы файлов*.

3. После того, как вышеперечисленные действия будут завершены (это может быть довольно длительный процесс в зависимости от размера вашего репозитория системы контроля версий и количества сообщений для перевода), вы сможете начать перевод.

Установка на macOS

Требования к оборудованию

Weblate должен без проблем работать на всём современном оборудовании, ниже приведена минимальная конфигурация, необходимая для запуска Weblate на одном узле (сам Weblate, база данных и веб-сервер):

- 2 ГБ оперативной памяти
- 2-х ядерный процессор
- 1 ГБ дискового пространства

Чем больше памяти, тем лучше — она используется для кэширования на всех уровнях (на уровне файловой системы, уровне базы данных и уровне Weblate).

Множество одновременно работающих пользователей увеличивают количество необходимых ядер процессора. Для сотен компонентов перевода рекомендуется не менее 4 Гб оперативной памяти.

Примечание: Фактические требования к вашей установке Weblate сильно зависят от размера управляемых ею переводов.

Установка

Системные требования

Установите зависимости, необходимые для сборки модулей Python (смотрите раздел *Требования к программному обеспечению*):

```
brew install pango libjpeg python git libyaml gobject-introspection
pip3 install virtualenv
```

Убедитесь, что pip сможет найти версию `libffi`, предоставляемую homebrew — она понадобится на этапе сборки установки.

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

Установите желаемые необязательные зависимости в зависимости от функций, которые вы собираетесь использовать (смотрите раздел *Необязательные зависимости*):

```
brew install tesseract
```

При желании установите программное обеспечение для запуска рабочего сервера, смотрите разделы *Запуск сервера*, *Настройка базы данных для Weblate* и *Фоновые задачи с использованием Celery*. В зависимости от размера вашей установки вы можете захотеть запустить эти компоненты на выделенных серверах.

Инструкции по установке на местах:

```
# 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
```

Модули Python

Подсказка: Для установки Weblate мы используем виртуальное окружение, предоставляемое пакетом virtualenv, чтобы отделить Weblate от вашей системы. Если вы не знакомы с этим понятием, обратитесь к документации пакета.

1. Создайте виртуальное окружение для Weblate:

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

2. Активируйте виртуальное окружение для Weblate:

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

3. Установите Weblate, включая все его зависимости:

```
pip install Weblate
```

4. Установите драйвер базы данных:

```
pip install psycopg2-binary
```

5. Установите желаемые необязательные зависимости в зависимости от функций, которые вы собираетесь использовать (некоторые из них могут потребовать дополнительных системных библиотек, проверьте раздел *Необязательные зависимости*):

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

Настройка Weblate

Примечание: Следующие шаги подразумевают, что виртуальное окружение, используемое Weblate'ом, активно (активировать его можно командой `. ~/weblate-env/bin/activate`). В случае, если это не так, вам придётся указывать полный путь к команде **`weblate`**: `~/weblate-env/bin/weblate`.

1. Скопируйте файл `~/weblate-env/lib/python3.7/site-packages/weblate/settings_example.py` в `~/weblate-env/lib/python3.7/site-packages/weblate/settings.py`.
2. По своему усмотрению подкорректируйте значения в новом файле `settings.py`. Для тестирования вы можете использовать поставляемый пример, но для рабочей среды вы захотите внести в него изменения, смотрите раздел *Настройка конфигурации*.
3. Создайте базу данных и её структуру для Weblate (пример настройки использует PostgreSQL, для установки в рабочую среду ознакомьтесь разделом *Настройка базы данных для Weblate*):

```
weblate migrate
```

4. Создайте учётную запись администратора и скопируйте сгенерированный ей пароль в буфер обмена, также сохраните его для дальнейшего использования:

```
weblate createadmin
```

5. Соберите статические файлы для веб-сервера (смотрите разделы *Запуск сервера* и *Обслуживание статических файлов*):

```
weblate collectstatic
```

6. Сожмите файлы JavaScript и CSS (необязательно, смотрите раздел *Сжатие клиентских активов*):

```
weblate compress
```

7. Запустите рабочие процессы Celery. Для целей разработки это делать не обязательно, но настоятельно рекомендуется. Дополнительную информацию смотрите в разделе *Фоновые задачи с использованием Celery*:

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

8. Запустите сервер разработки (шаги для запуска рабочего сервера смотрите в разделе *Запуск сервера*):

```
weblate runserver
```

После установки

Поздравляем, теперь ваш сервер Weblate запущен и вы можете начать его использовать.

- Теперь вы можете получить доступ к Weblate по адресу `http://localhost:8000/`.
- Войдите в систему с учётными данными администратора, полученными во время установки, или зарегистрируйтесь в качестве нового пользователя.
- Теперь при помощи команды **`weblate`** и когда активно виртуальное окружение Weblate вы можете запускать его команды, смотрите раздел *Команды управления*.
- Остановить тестовый сервер можно комбинацией `Ctrl+C`.

Добавление перевода

1. Откройте интерфейс администратора (`http://localhost:8000/create/project/`) и создайте проект, который вы хотите перевести. Подробнее смотрите в разделе *Конфигурация проекта*.

   Всё, что вам нужно здесь указать — это название проекта и его веб-сайт.

2. Создайте компонент, который является реальным объектом для перевода — он указывает на репозиторий системы контроля версий и задаёт файлы для перевода. Подробнее смотрите в разделе *Конфигурация компонента*.

   Здесь важны следующие поля: название компонента, адрес репозитория системы контроля версий и маска для поиска переводимых файлов. Weblate поддерживает широкий спектр форматов, включая PO-файлы gettext, строки ресурсов Android'а, строки свойств iOS, файлы свойств Java или Qt Linguist, подробнее смотрите в разделе *Поддерживаемые форматы файлов*.

3. После того, как вышеперечисленные действия будут завершены (это может быть довольно длительный процесс в зависимости от размера вашего репозитория системы контроля версий и количества сообщений для перевода), вы сможете начать перевод.

Установка из исходников

1. Пожалуйста, сперва проследуйте инструкциям по установке для вашей системы:
   - *Установка на Debian и Ubuntu*
   - *Установка на SUSE и openSUSE*
   - *Установка на RedHat, Fedora и CentOS*
2. Получите из Git'а последние исходные коды Weblate'а (или скачайте архив и распакуйте его):

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

   В качестве альтернативы вы можете использовать архивы выпусков. Вы можете скачать их с нашего сайта <https://weblate.org/>. Эти файлы подписаны криптографической подписью, пожалуйста, смотрите раздел *Проверка подписей выпусков*.
3. Установите текущий код Weblate'а в виртуальное окружение:

   ```
   . ~/weblate-env/bin/activate
   pip install -e weblate-src
   ```
4. Скопируйте файл `weblate/settings_example.py` в `weblate/settings.py`.
5. По своему усмотрению подкорректируйте значения в новом файле `settings.py`. Для тестирования вы можете использовать поставляемый пример, но для рабочей среды вы захотите внести в него изменения, смотрите раздел *Настройка конфигурации*.
6. Создайте используемую Weblate'ом базу данных, подробности смотрите в разделе *Настройка базы данных для Weblate*.
7. Создайте Django'вские таблицы, соберите статические файлы и заполните исходные данные (смотрите разделы *Наполнение базы данных* и *Обслуживание статических файлов*):

   ```
   weblate migrate
   weblate collectstatic
   weblate compress
   weblate compilemessages
   ```

   Примечание: Этот шаг следует повторять при обновлении репозитория.

Установка на OpenShift

Примечание: Для этого руководства ищутся участники, имеющие опыт работы с OpenShift, подробности в замечании <https://github.com/WeblateOrg/weblate/issues/2889>.

Weblate поддерживает OpenShift, необходимые интеграционные файлы находятся в основном репозитории в каталоге `openshift3`.

В зависимости от ваших настроек и опыта выберите подходящий для вас способ установки:

- *Установка с помощью Docker'а*, рекомендуется для рабочей среды.
- Установка в виртуальное окружение, рекомендуется для рабочей среды:
  - *Установка на Debian и Ubuntu*
  - *Установка на SUSE и openSUSE*
  - *Установка на RedHat, Fedora и CentOS*

  - *Установка на macOS*
- *Установка из исходников*, рекомендуется для разработки.
- *Установка на OpenShift*.

2.1.2 Требования к программному обеспечению

Операционная система

Weblate работает под Linux, FreeBSD и MacOS. Скорее всего он будет работать и под другими Unix-подобными системами.

Работа Weblate под Windows не поддерживается. Тем не менее, работать под ней он всё ещё может и патчи под неё с радостью принимаются.

Другие сервисы

Для своей работы Weblate использует другие сервисы. Вам понадобятся, по крайней мере, следующие запущенные сервисы:

- Сервер базы данных PostgreSQL, смотрите раздел *Настройка базы данных для Weblate*.
- Сервер Redis для кэша и очереди задач, смотрите раздел *Фоновые задачи с использованием Celery*.
- SMTP-сервер для исходящей почты, смотрите раздел *Настройка исходящей почты*.

Зависимости Python

Weblate написан на Python'е и поддерживает Python 3.6 или более новую версию. Вы можете установить зависимости с помощью команды pip или из пакетов вашего дистрибутива, полный список зависимостей находится в файле `requirements.txt`.

Основные зависимости:

Django https://www.djangoproject.com/

Celery https://docs.celeryproject.org/

Translate Toolkit https://toolkit.translatehouse.org/

translation-finder https://github.com/WeblateOrg/translation-finder

Python Social Auth https://python-social-auth.readthedocs.io/

Фреймворк Django REST https://www.django-rest-framework.org/

Необязательные зависимости

Для работы некоторых функций Weblate необходимы следующие модули. Полный их список можно найти в файле `requirements-optional.txt`.

`Mercurial` (необязательная зависимость для поддержки репозиториев Mercurial) https://www.mercurial-scm.org/

`phply` (необязательная зависимость для поддержки PHP) https://github.com/viraptor/phply

`tesserocr` (необязательная зависимость для поддержки распознавания текста со снимков экрана) https://github.com/sirfz/tesserocr

`akismet` (необязательная зависимость для защиты предложений от спама) https://github.com/ubernostrum/akismet

`ruamel.yaml` (необязательная зависимость для поддержки *файлов YAML*) https://pypi.org/project/ruamel.yaml/

`Zeep` (необязательная зависимость для поддержки *терминологической службы Майкрософт*) https://docs.python-zeep.org/

`aeidon` (необязательная зависимость для поддержки *файлов субтитров*) https://pypi.org/project/aeidon/

Зависимости серверной части базы данных

Weblate поддерживает PostgreSQL, MySQL и MariaDB, подробнее смотрите в разделе *Настройка базы данных для Weblate* и документации по реализации поддержки соответствующей базы данных.

Другие системные требования

В системе должны быть установлены следующие зависимости:

`Git` https://git-scm.com/

Pango, Cairo и связанные заголовочные файлы и данные интроспекции gir https://cairographics.org/, https://pango.gnome.org/, смотрите раздел *Pango и Cairo*

`git-review` (необязательная зависимость для поддержки Gerrit) https://pypi.org/project/git-review/

`git-svn` (необязательная зависимость для поддержки Subversion) https://git-scm.com/docs/git-svn

`tesseract` и его данные (необязательная зависимость для поддержки распознавания текста со снимков экрана https://github.com/tesseract-ocr/tesseract

`licensee` (необязательная зависимость для определения лицензии при создании компонента) https://github.com/licensee/licensee

Зависимости времени компиляции

Для компиляции некоторых *зависимостей Python* может потребоваться установка их зависимостей. Это зависит от того, как вы их устанавливаете, так что для получения помощи по установке обратитесь к документации соответствующих пакетов. Зависимости зависимостей вам не понадобятся, если вы используете предварительно собранные «колёса» при их установке через `pip` или устанавливаете их через пакетный менеджер вашего дистрибутива.

Pango и Cairo

Изменено в версии 3.7.

Weblate использует Pango и Cairo для отрисовки растровых виджетов (смотрите раздел *Продвижение перевода*) и проверки отрисовки текста (смотрите раздел *Управление шрифтами*). Для правильной установки привязок Python сначала нужно установить системные библиотеки — вам нужны и Cairo, и Pango, которым, в свою очередь, нужна GLib. Все они должны быть установлены с файлами для разработки и данными интроспекции GObject.

2.1.3 Проверка подписей выпусков

Выпуски Weblate подписываются криптографической подписью выпускающего разработчика. В настоящее время им является Михаил Чихарж. Отпечаток его PGP-ключа:

```
63CB 1DF1 EF12 CF2A C0EE 5A32 9C27 B313 42B7 511D
```

а дополнительную идентификационную информацию вы можете получить по адресу <https://keybase.io/nijel>.

Вы должны удостовериться, что подпись соответствует скачанному вами архиву. Таким образом, вы сможете быть уверены, что используете тот же код, который был выпущен. Также вы должны проверить дату подписания, чтобы удостовериться, что вы скачали последнюю версию.

Каждый архив сопровождается файлами `.asc`, содержащими его PGP-подпись. После того, как вы разместите оба файла в одном каталоге, вы сможете проверить подпись:

```
$ gpg --verify Weblate-3.5.tar.xz.asc
gpg: assuming signed data in 'Weblate-3.5.tar.xz'
gpg: Signature made Ne 3. března 2019, 16:43:15 CET
gpg:                using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: Can't check signature: public key not found
```

Как видите, GPG жалуется на то, что он не знает открытого ключа. На данный момент вы должны сделать одно из следующего:

- Использовать *wkd* для скачивания ключа:

```
$ gpg --auto-key-locate wkd --locate-keys michal@cihar.com
pub   rsa4096 2009-06-17 [SC]
      63CB1DF1EF12CF2AC0EE5A329C27B31342B7511D
uid           [ultimate] Michal Čihař <michal@cihar.com>
uid           [ultimate] Michal Čihař <nijel@debian.org>
uid           [ultimate] [jpeg image of size 8848]
uid           [ultimate] Michal Čihař (Braiins) <michal.cihar@braiins.cz>
sub   rsa4096 2009-06-17 [E]
sub   rsa4096 2015-09-09 [S]
```

- Скачать связку ключей с сервера Михаила, а затем импортировать её командой:

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

- Скачать и импортировать ключ с одного из серверов ключей:

```
$ gpg --keyserver hkp://pgp.mit.edu --recv-keys␣
↪87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: key 9C27B31342B7511D: "Michal Čihař <michal@cihar.com>" imported
gpg: Total number processed: 1
gpg:              unchanged: 1
```

Это немного улучшит ситуацию — на данный момент вы можете проверить правильность подписи с данного ключа, но всё равно не можете доверять используемому в ключе имени:

```
$ gpg --verify Weblate-3.5.tar.xz.asc
gpg: assuming signed data in 'Weblate-3.5.tar.xz'
gpg: Signature made Ne 3. března 2019, 16:43:15 CET
gpg:                using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: Good signature from "Michal Čihař <michal@cihar.com>" [ultimate]
gpg:                 aka "Michal Čihař <nijel@debian.org>" [ultimate]
gpg:                 aka "[jpeg image of size 8848]" [ultimate]
gpg:                 aka "Michal Čihař (Braiins) <michal.cihar@braiins.cz>"␣
↪[ultimate]
gpg: WARNING: This key is not certified with a trusted signature!
```

(continues on next page)

(продолжение с предыдущей страницы)

```
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 63CB 1DF1 EF12 CF2A C0EE  5A32 9C27 B313 42B7 511D
```

Проблема в том, что выдать ключ с таким именем может кто угодно. Вам нужно убедиться, что ключ действительно принадлежит указанному лицу. Руководство GNU по обеспечению конфиденциальности раскрывает эту тему в главе Проверка достоверности ключей. Самый надёжный метод — лично встретиться с разработчиком и обменяться отпечатками ключей, однако вы также можете положиться на сеть доверия. Таким образом, вы можете транзитивно доверять ключу через подписи других людей, которые встречались с разработчиком лично.

Когда достоверность ключа будет установлена, предупреждение больше появляться не будет:

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

В случае недействительной подписи (архив был изменён), вы получите явную ошибку, независимо от того, доверяете ли вы ключу или нет:

```
$ gpg --verify Weblate-3.5.tar.xz.asc
gpg: Signature made Sun Mar  3 16:43:15 2019 CET
gpg:                using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: BAD signature from "Michal Čihař <michal@cihar.com>" [ultimate]
```

2.1.4 Разрешения файловой системы

Процесс Weblate должен мочь читать и писать в каталог, в котором он хранит свои данные — `DATA_DIR`. Все файлы в этом каталоге должны принадлежать и иметь разрешение на запись для пользователя, под которым запущен Weblate.

В конфигурации по умолчанию они расположены в том же дереве, что и исходный код Weblate, однако вы можете предпочесть переместить их в более подходящее место, например, в `/var/lib/weblate`.

Weblate попытается автоматически создать эти каталоги, но у него ничего не получится, если у него не будет на это разрешений.

Также вам следует внимательно запускать *команды управления*, поскольку они должны выполняться под тем же пользователем, под которым запущен сам Weblate, в противном случае разрешения на некоторые файлы могут быть выставлены неверно.

In the Docker container, all files in the `/app/data` volume have to be owned by weblate user inside the container (UID 1000).

См.также:

Обслуживание статических файлов

2.1.5 Настройка базы данных для Weblate

Рекомендуется запускать Weblate с сервером баз данных PostgreSQL.

См.также:

Использование мощного движка базы данных, Базы данных, *Переход с других баз данных на PostgreSQL*

PostgreSQL

PostgreSQL, как правило, является лучшим выбором для сайтов, написанных на Django. Это эталонная база данных, используемая для реализации слоя баз данных Django.

Примечание: Weblate использует расширение для триграмм, которое в некоторых случаях должно быть установлено отдельно. Ищите пакет `postgresql-contrib` или аналогичный.

См.также:

PostgreSQL notes

Создание базы данных в PostgreSQL

Обычно рекомендуется запускать Weblate в отдельной базе данных и под отдельной учётной записью:

```
# 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 -O weblate weblate
```

Подсказка: Если вы не хотите делать пользователя Weblate суперпользователем в PostgreSQL, вы можете пропустить этот шаг. В этом случае вам придётся выполнить некоторые шаги по миграции вручную, так как суперпользователь PostgreSQL запускает в схеме Weblate следующую команду:

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

Настройка Weblate для использования PostgreSQL

Фрагмент настроек `settings.py` для PostgreSQL:

```
DATABASES = {
    'default': {
        # Database engine
        'ENGINE': 'django.db.backends.postgresql',
        # Database name
        'NAME': 'weblate',
        # Database user
        'USER': 'weblate',
        # Database password
        'PASSWORD': 'password',
        # Set to empty string for localhost
        'HOST': 'database.example.com',
```

(continues on next page)

(продолжение с предыдущей страницы)

```
        # Set to empty string for default
        'PORT': '',
    }
}
```

MySQL и MariaDB

Weblate также можно использовать с MySQL или MariaDB, ознакомитесь с предостережениями по их использованию вместе с Django в разделах Замечания по MySQL и Замечания по MariaDB его документации.

Подсказка: Некоторые функции Weblate лучше работают с *PostgreSQL*. Среди них: поиск и память переводов, обе из которых используют функции полнотекстового поиска базы данных, а их реализация в PostgreSQL превосходит все остальные.

Поэтому для новых установок рекомендуется использовать *PostgreSQL*.

Для Weblate рекомендуется следующая конфигурация:

- Используйте кодовую таблицу `utf8mb4` для представления символов с более высоких плоскостей Юникода (например, эмодзи).
- Настройте сервер на использование `Innodb_large_prefix` для разрешения более длинных индексов на текстовых полях.
- Установите уровень изоляции в `READ COMMITTED`.
- Режим SQL должен быть установлен в `STRICT_TRANS_TABLES`.

2.1.6 Другие настройки

Настройка исходящей почты

Weblate отправляет электронные письма по разным поводам — для активации учётной записи и по различным уведомлениям, настроенным пользователями. Для этого ему необходим доступ к SMTP-серверу.

Настройка почтового сервера производится с помощью следующих настроек: `EMAIL_HOST`, `EMAIL_HOST_PASSWORD`, `EMAIL_HOST_USER` и `EMAIL_PORT`. Их имена говорят сами за себя, а более подробную информацию вы можете найти в документации Django.

Примечание: Проверить, правильно ли работает исходящая почта, можно с помощью команды управления `sendtestemail` (для получения инструкций о том, как вызывать её в различных окружениях, смотрите раздел *Вызов команд управления*).

Работа за обратным прокси

Некоторые функции Weblate полагаются на возможность получения IP-адреса клиента. Среди этих функций *Ограничение частоты запросов*, *Защита от спама* или *Журнал аудита*.

В стандартной конфигурации Weblate достаёт IP-адрес из переменной `REMOTE_ADDR`, которая устанавливается обработчиком WSGI.

Если же вы используете обратный прокси, то это поле, скорее всего, будет содержать его адрес. Вам нужно настроить Weblate на доверие дополнительным HTTP-заголовкам и доставать IP-адрес из них.

Эту настройку нельзя включить по умолчанию, поскольку она разрешит подмену IP-адреса для установок, не использующих обратный прокси. Для самых обычных установок будет достаточно включения параметра *IP_BEHIND_REVERSE_PROXY*, но вам также может понадобиться подправить параметры *IP_PROXY_HEADER* и *IP_PROXY_OFFSET*.

См.также:

Защита от спама, *Ограничение частоты запросов*, *Журнал аудита*, *IP_BEHIND_REVERSE_PROXY*, *IP_PROXY_HEADER*, *IP_PROXY_OFFSET*, SECURE_PROXY_SSL_HEADER

HTTP-прокси

Weblate выполняет команды системы контроля версий, а те принимают конфигурацию прокси из окружения. Рекомендуемый подход заключается в определении настроек прокси в файле `settings.py`:

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

См.также:

Переменные окружения для прокси

2.1.7 Настройка конфигурации

См.также:

Пример конфигурации

Скопируйте `weblate/settings_example.py` в `weblate/settings.py` и подправьте его в соответствии со своими настройками. Скорее всего, вы захотите настроить следующие параметры: `ADMINS`

Список администраторов сайтов для получения уведомлений, когда что-то идёт не так, например, для получения уведомлений о неудачных слияниях, или об ошибках Django.

См.также:

ADMINS

`ALLOWED_HOSTS`

Вы должны установить эту настройку в список хостов, которые должен обслуживать ваш сайт. Например:

```
ALLOWED_HOSTS = ['demo.weblate.org']
```

Или же вы можете указать символ шаблона:

```
ALLOWED_HOSTS = ['*']
```

См.также:

ALLOWED_HOSTS, *WEBLATE_ALLOWED_HOSTS*, *Настройка разрешённых хостов*

`SESSION_ENGINE`

Настраивает то, как будут храниться ваши сессии. В случае, если вы оставили механизм хранения по умолчанию — в базе данных — вы должны настроить расписание для команды **weblate clearsessions**, удаляющей из базы данных данные об устаревших сессиях.

Если вы используете в качестве кэша Redis (смотрите раздел *Включение кэширования*), рекомендуется использовать его также и для сессий:

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

См.также:

Настройка движка сессий, `SESSION_ENGINE`

`DATABASES`

Подключение к серверу базы данных, подробнее об этой настройке читайте в документации Django.

См.также:

Настройка базы данных для Weblate, `DATABASES`, Базы данных

`DEBUG`

Отключите эту настройку для любого рабочего сервера. При включённом режиме отладки Django в случае возникновения ошибки будет показывать трассировки стека пользователям, при его же отключении ошибки будут отправляться на адреса электронной почты из массива `ADMINS` (смотрите выше).

Кроме того, отладочный режим замедляет работу Weblate, поскольку в этом случае Django сохраняет внутри себя гораздо больше информации.

См.также:

`DEBUG`

`DEFAULT_FROM_EMAIL`

Адрес электронной почты отправителя для исходящих сообщений электронной почты, например, для писем регистрации.

См.также:

`DEFAULT_FROM_EMAIL`

`SECRET_KEY`

Ключ, используемый Django для подписывания некоторой информации, сохраняемой в куках, подробнее смотрите в разделе *Секретный ключ Django*.

См.также:

`SECRET_KEY`

`SERVER_EMAIL`

Адрес электронной почты, используемый в качестве адреса отправителя для отправки сообщений администратору, например, уведомлений о неудачных слияниях.

См.также:

`SERVER_EMAIL`

2.1.8 Наполнение базы данных

После того, как ваша конфигурация будет готова, вы можете запустить `weblate migrate` для создания структуры базы данных. Теперь вы можете создавать проекты перевода через интерфейс администратора.

В случае, если вы хотите запустить установку не в интерактивном режиме, вы можете использовать `weblate migrate --noinput`, а затем создать пользователя-администратора при помощи команды `createadmin`.

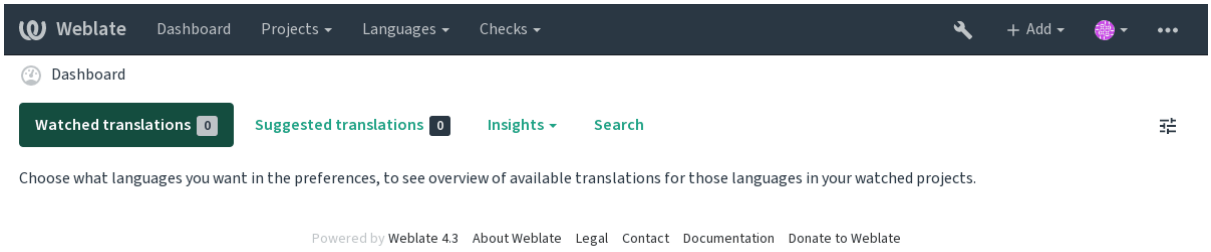
После того, как вы закончите, вы также должны проверить *Отчёт о производительности* в интерфейсе администратора, который может дать вам подсказки по исправлению потенциально неоптимальной конфигурации вашего сайта.

См.также:

Конфигурация, *Управление доступом*

2.1.9 Рабочая среда

Для установки в рабочую среду необходимо провести корректировки, описанные в следующих разделах. Наиболее критичные настройки вызовут предупреждение, которое обозначается восклицательным знаком в верхней панели, если вы вошли в систему под суперпользователем:



Также рекомендуется изучить все проверки, произведённые Django (хотя, возможно, исправлять их все нет необходимости):

```
weblate check --deploy
```

См.также:

Контрольный список развертывания

Отключение отладочного режима

Отключите отладочный режим Django (*DEBUG*), установив:

```
DEBUG = False
```

При включённом отладочном режиме Django сохраняет все выполненные запросы и при возникновении ошибок показывает пользователям обратную трассировку стека, что в рабочей среде не желательно.

См.также:

Настройка конфигурации

Правильная настройка администраторов

Укажите в настройке *ADMINS* правильные адреса администраторов, чтобы определить, кто будет получать электронную почту в случае, если на сервере что-то пойдёт не так, например:

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

См.также:

Настройка конфигурации

Установка правильного домена сайта

Поправьте имя сайта и домен в интерфейсе администратора, иначе ссылки в RSS или регистрационные письма не будут работать. Эта настройка осуществляется с помощью параметра `SITE_DOMAIN`, который должен содержать доменное имя сайта.

Изменено в версии 4.2: До выхода версии 4.2 вместо него использовался фреймворк «сайтов» Django, смотрите раздел Фреймворк «сайтов».

См.также:

Настройка разрешённых хостов, *Правильная настройка HTTPS* `SITE_DOMAIN`, `WEBLATE_SITE_DOMAIN`, `ENABLE_HTTPS`

Правильная настройка HTTPS

Настоятельно рекомендуется запускать Weblate с использованием шифрующего протокола HTTPS. После его включения в настройках необходимо установить параметр `ENABLE_HTTPS`:

```
ENABLE_HTTPS = True
```

Подсказка: Возможно, вы также захотите настроить HSTS, для получения более подробной информации смотрите документ SSL/HTTPS.

См.также:

`ENABLE_HTTPS`, *Настройка разрешённых хостов*, *Установка правильного домена сайта*

Установка правильного значения SECURE_HSTS_SECONDS

Если ваш сайт обслуживается по SSL, вам необходимо рассмотреть возможность установки значения параметра `SECURE_HSTS_SECONDS` в файле `settings.py`, включающего HTTP Strict Transport Security. По умолчанию его значение установлено в 0, как показано ниже.

```
SECURE_HSTS_SECONDS = 0
```

Если для него установлено ненулевое целое значение, то `django.middleware.security.SecurityMiddleware` устанавливает заголовок HTTP Strict Transport Security у всех ответов, которые его ещё не имеют.

Предупреждение: Неправильная настройка этого параметра может привести к необратимому (на некоторое время) выходу из строя вашего сайта. Поэтому сперва ознакомьтесь с документацией по HTTP Strict Transport Security.

Использование мощного движка базы данных

Пожалуйста, используйте для рабочего окружения PostgreSQL, для получения более подробной информации смотрите раздел *Настройка базы данных для Weblate*.

См.также:

Настройка базы данных для Weblate, *Переход с других баз данных на PostgreSQL*, *Настройка конфигурации*, Базы данных

Включение кэширования

Если возможно, используйте Redis из Django, настроив конфигурационную переменную `CACHES`, например:

```
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',
        }
    }
}
```

См.также:

Кеширование аватара, Фреймворк кэширования Django

Кеширование аватара

Помимо кэширования силами Django, Weblate выполняет кэширование аватаров. Для этой цели рекомендуется воспользоваться отдельным файловым кэшем:

```
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,
        },
    }
```

См.также:

`ENABLE_AVATARS`, `AVATAR_URL_PREFIX`, *Аватары*, *Включение кэширования*, Фреймворк кэширования Django

Настройка отправки электронной почты

Weblate'у по разным случаям необходимо отправлять электронные письма и эти письма должны иметь правильный адрес отправителя, поэтому настройте параметры *SERVER_EMAIL* и *DEFAULT_FROM_EMAIL* так, чтобы они соответствовали вашему окружению, например:

```
SERVER_EMAIL = 'admin@example.org'
DEFAULT_FROM_EMAIL = 'weblate@example.org'
```

Примечание: Для отключения отправки писем Weblate'ом установите переменную EMAIL_BACKEND в значение `django.core.mail.backends.dummy.EmailBackend`.

При этом доставка сообщений электронной почты будет отключена *целиком*, включая сообщения для регистрации или сброса пароля.

См.также:

Настройка конфигурации, *Настройка исходящей почты*, EMAIL_BACKEND, DEFAULT_FROM_EMAIL, SERVER_EMAIL

Настройка разрешённых хостов

Django требует, чтобы в параметре *ALLOWED_HOSTS* содержался список доменных имён, которые разрешено обслуживать вашему сайту, если его оставить пустым, любые запросы будут заблокированы.

В случае, если он не настроен на соответствие вашему HTTP-серверу, вы будете получать ошибки вида `Invalid HTTP_HOST header: '1.1.1.1'. You may need to add '1.1.1.1' to ALLOWED_HOSTS.` (`Неверный заголовок HTTP_HOST: '1.1.1.1'. Возможно, вам нужно добавить '1.1.1.1' в ALLOWED_HOSTS.`)

Подсказка: Внутри контейнера Docker он доступен через переменную окружения *WEBLATE_ALLOWED_HOSTS*.

См.также:

ALLOWED_HOSTS, *WEBLATE_ALLOWED_HOSTS*, *Установка правильного домена сайта*

Секретный ключ Django

Параметр *SECRET_KEY* используется Django для подписания файлов кук и вы обязательно должны сгенерировать своё собственное значение, а не использовать приведённое в примере настройки.

Сгенерировать новый ключ вы можете с помощью скрипта `weblate/examples/generate-secret-key`, поставляемого вместе с Weblate.

См.также:

SECRET_KEY

Домашний каталог

Изменено в версии 2.1: Больше он не требуется, теперь Weblate хранит все свои данные в каталоге *DATA_DIR*.

Домашний каталог для пользователя, запустившего Weblate, должен существовать и быть доступен для записи из под этого пользователя. Особенно он необходим, если вы хотите использовать SSH для доступа к частным репозиториям, но доступ к этому каталогу может понадобиться и Git'у (это зависит от используемой вами версии Git'a).

Изменить используемый Weblate каталог вы можете в файле `settings.py`, например, установив его в каталог `configuration` внутри дерева Weblate:

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

Примечание: В Linux и других UNIX-подобных системах путь к домашнему каталогу пользователя определяется в файле `/etc/passwd`. Многие дистрибутивы по умолчанию имеют не записываемый каталог для пользователей, используемый для раздачи веб-содержимого (вроде `apache`, `www-data` или ``wwwrun`), так что вам либо придётся запускать Weblate под другим пользователем, либо изменить эту настройку.

См.также:

Доступ к репозиториям

Загрузка шаблонов

Рекомендуется использовать кэширующий загрузчик шаблонов для Django. Он кэширует обработанные шаблоны, избавляя фреймворк от необходимости разбирать их каждый запрос. Настроить его вы можете с помощью следующего фрагмента кода (самое важное здесь — это настройка `loaders`):

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

См.также:

`django.template.loaders.cached.Loader`

Выполнение задач технического обслуживания

Для оптимальной производительности неплохо бы в фоновом режиме выполнять некоторые задачи по техническому обслуживанию. Теперь это делается автоматически с помощью *фоновых задач Celery*, которые включают в себя:

- Проверку работоспособности состояния конфигурации (ежечасно).
- Фиксацию отложенных изменений (ежечасно), смотрите раздел *Отложенные коммиты* и команду `commit_pending`.
- Обновление предупреждений компонента (ежедневно).
- Обновление удалённых веток (по ночам), смотрите параметр `AUTO_UPDATE`.
- Резервное копирование памяти переводов в JSON (ежедневно), смотрите команду `dump_memory`.
- Задачи по обслуживанию полнотекстового индекса и базы данных (ежедневные и еженедельные задачи), смотрите команду `cleanuptrans`.

Изменено в версии 3.2: Начиная с версии 3.2, по умолчанию эти задачи выполняются через Celery и Weblate уже поставляется с правильной конфигурацией, смотрите раздел *Фоновые задачи с использованием Celery*.

Системные локали и кодировки

Системные локали должны быть настроены на поддерживающие UTF-8. В большинстве дистрибутивов Linux это умолчательная их настройка. В случае, если в вашей системе это не так, измените пожалуйста локали на вариант с поддержкой UTF-8.

Например, путём редактирования файла `/etc/default/locale` и прописывания в нём команды `LANG="C.UTF-8"`.

В некоторых случаях отдельные сервисы имеют отдельные настройки для локалей. Например, при использовании Apache вы можете установить локаль в файле `/etc/apache2/envvars`:

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

Использование пользовательского центра сертификации

Во время HTTP-запросов Weblate проверяет SSL-сертификаты. Если вы используете пользовательский центр сертификации, который цепочке сертификатов по умолчанию не является доверенным, вам придётся добавить его сертификат в качестве доверенного.

Предпочтительный подход — сделать это на системном уровне, подробнее смотрите документацию по дистрибутиву (например, в debian это можно сделать, разместив сертификат центра сертификации в каталоге `/usr/local/share/ca-certificates/` и запустив команду **update-ca-certificates**).

Как только это будет сделано, системные инструменты станут доверять сертификату, среди этих инструментов будет и Git.

Для кода Python вам будет необходимо настроить запросы на использование системной цепочки сертификатов вместо использование цепочки, поставляемой с Python'ом. Этого можно добиться, поместив в `settings.py` следующий фрагмент кода (путь указан для Debian):

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

Сжатие клиентских активов

Weblate поставляется с кучей JavaScript и CSS файлов. Из соображений производительности хорошо бы их сжать перед отправкой клиенту. В конфигурации по умолчанию это делается на лету ценой небольших накладных расходов. В больших установках рекомендуется включить автономный режим сжатия. Это должно быть сделано в конфигурации, и сжатие должно запускаться при каждом обновлении Weblate.

Переключение конфигурации выполняется простым включением параметра `django.conf.settings.COMPRESS_OFFLINE` и настройкой параметра `django.conf.settings.COMPRESS_OFFLINE_CONTEXT` (последний уже включён в пример конфигурации):

```
COMPRESS_OFFLINE = True
```

При каждом развёртывании вам необходимо сжимать файлы, чтобы они соответствовали текущей версии:

```
weblate compress
```

Подсказка: В официальном образе Docker эта функция уже включена.

См.также:

Распространённые сценарии развёртывания, *Обслуживание статических файлов*

2.1.10 Запуск сервера

Чтобы запустить Weblate, вам понадобится несколько сервисов, рекомендуемая конфигурация состоит из:

- Сервера базы данных (смотрите раздел *Настройка базы данных для Weblate*)
- Сервера кэша (смотрите раздел *Включение кэширования*)
- Веб-сервера на клиентской части для раздачи статических файлов и SSL-терминации (смотрите раздел *Обслуживание статических файлов*)
- Сервера Wsgi для генерации динамического содержимого (смотрите раздел *Пример конфигурации для NGINX и uWSGI*)
- Celery для выполнения фоновых задач (смотрите раздел *Фоновые задачи с использованием Celery*)

Примечание: Между некоторыми сервисами существуют зависимости, например, кэш и база данных должны быть запущены до запуска процессов Celery или uwsgi.

В большинстве случаев вы будете запускать все сервисы на одном (виртуальном) сервере, но в случае, если ваша установка испытывает тяжёлые нагрузки, вы можете их разделить. Единственным ограничением является то, что серверы Celery и Wsgi должны иметь доступ к каталогу `DATA_DIR`.

Запуск веб-сервера

Запуск Weblate ничем не отличается от запуска любой другой программы на базе Django. Как правило, Django выполняется как uWSGI или fcgi (смотрите ниже примеры для различных веб-серверов).

В целях тестирования можно использовать встроенный в Django веб-сервер:

```
weblate runserver
```

Предупреждение: НЕ ИСПОЛЬЗУЙТЕ ЭТОТ СЕРВЕР В РАБОЧЕЙ СРЕДЕ. Он не проходил аудиты безопасности или тесты производительности. Также смотрите документацию Django по команде `runserver`.

Подсказка: Встроенный сервер Django обслуживает статические файлы только с включённым параметром *DEBUG*, поскольку он предназначен только для разработки. Для использования в рабочей среде смотрите установки wsgi в разделах *Пример конфигурации для NGINX и uWSGI*, *Пример конфигурации для Apache*, *Пример конфигурации для Apache и Gunicorn* и *Обслуживание статических файлов*.

Обслуживание статических файлов

Изменено в версии 2.4: До версии 2.4 Weblate некорректно использовал фреймворк статических файлов Django, поэтому его настройка была более сложной.

Django необходимо собрать свои статические файлы в одном каталоге. Для этого необходимо выполнить команду `weblate collectstatic --noinput`. Она скопирует статические файлы в каталог, указанный в параметре `STATIC_ROOT` (по умолчанию это каталог `static` внутри каталога *DATA_DIR*).

Статические файлы рекомендуется раздавать непосредственно с вашего веб-сервера, это следует делать для следующих путей:

/static/ Обслуживает статические файлы для Weblate и административного интерфейса (из каталога, определённого параметром `STATIC_ROOT`).

/media/ Используется для загрузки пользовательских мультимедийных данных (например, снимков экрана).

/favicon.ico Должен быть переписан, чтобы переписать правило для раздачи `/static/favicon.ico`.

См.также:

Сжатие клиентских активов, Развёртывание Django, Развёртывание статических файлов

Политика безопасности содержимого

Конфигурация Weblate по умолчанию включает промежуточное ПО `weblate.middleware.SecurityMiddleware` , которое устанавливает связанные с безопасностью заголовки HTTP, такие как `Content-Security-Policy` или `X-XSS-Protection`. По умолчанию они настроены для работы с Weblate и его конфигурацией, но для этого может потребоваться настройка вашего окружения.

См.также:

CSP_SCRIPT_SRC, *CSP_IMG_SRC*, *CSP_CONNECT_SRC*, *CSP_STYLE_SRC*, *CSP_FONT_SRC*

Пример конфигурации для NGINX и uWSGI

Для запуска рабочего веб-сервера используйте обёртку wsgi, устанавливаемую вместе с Weblate (в виртуальном окружении она устанавливается в `~/weblate-env/lib/python3.7/site-packages/weblate/wsgi.py`). Не забудьте также поместить ваше виртуальное окружение в пути поиска Python'а (например, используя установку `virtualenv = /home/user/weblate-env` в uWSGI).

Следующая конфигурация запускает Weblate как uWSGI-приложение под веб-сервером NGINX.

Конфигурация для NGINX (пример конфигурации взят из файла `weblate/examples/weblate.nginx.conf`):

```
# This example assumes Weblate is installed in virtualenv in /home/weblate/weblate-
↪env
# and DATA_DIR is set to /home/weblate/data, please adjust paths to match your␣
↪setup.
server {
    listen 80;
    server_name weblate;
    # Not used
    root /var/www/html;

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

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

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

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

Конфигурация для uWSGI (пример конфигурации взят из файла `weblate/examples/weblate.uwsgi.ini`):

```
# This example assumes Weblate is installed in virtualenv in /home/weblate/weblate-
↪env
# and DATA_DIR is set to /home/weblate/data, please adjust paths to match your␣
↪setup.
[uwsgi]
plugins       = python3
master        = true
protocol      = uwsgi
socket        = 127.0.0.1:8080
wsgi-file     = /home/weblate/weblate-env/lib/python3.7/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
```

(continues on next page)

(продолжение с предыдущей страницы)

```
# 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
```

См.также:

Как совместно использовать Django с uWSGI

Пример конфигурации для Apache

Следующая конфигурация запускает Weblate как WSGI-приложение, вам нужно включить модуль mod_wsgi (пример конфигурации взят из файла `weblate/examples/apache.conf`):

```
#
# VirtualHost for Weblate
#
# This example assumes Weblate is installed in virtualenv in /home/weblate/weblate-
↪env
# and DATA_DIR is set to /home/weblate/data, please adjust paths to match your␣
↪setup.
#
<VirtualHost *:80>
    ServerAdmin admin@weblate.example.org
    ServerName weblate.example.org

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

    # DATA_DIR/static/
    Alias /static/ /home/weblate/data/static/
    <Directory /home/weblate/data/static/>
```

(continues on next page)

(продолжение с предыдущей страницы)

```
        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
    WSGIProcessGroup weblate
    WSGIApplicationGroup %{GLOBAL}

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

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

</VirtualHost>
```

Примечание: Weblate требует Python 3, поэтому, пожалуйста, убедитесь, что вы используете вариант modwsgi, собранный с третьим Python'ом. Обычно он доступен в виде отдельного пакета, например `libapache2-mod-wsgi-py3`.

См.также:

Системные локали и кодировки, Как совместно использовать Django с Apache и mod_wsgi

Пример конфигурации для Apache и Gunicorn

Следующая конфигурация запускает Weblate в Gunicorn и Apache 2.4 (пример конфигурации взят из файла `weblate/examples/apache.gunicorn.conf`):

```
#
# VirtualHost for Weblate using gunicorn on localhost:8000
#
# This example assumes Weblate is installed in virtualenv in /home/weblate/weblate-
↪env
# and DATA_DIR is set to /home/weblate/data, please adjust paths to match your␣
↪setup.
#
<VirtualHost *:443>
    ServerAdmin admin@weblate.example.org
    ServerName weblate.example.org

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

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

(continues on next page)

(продолжение с предыдущей страницы)

```
    </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>
```

См.также:

Как совместно использовать Django с Gunicorn

Запуск Weblate по пути

Изменено в версии 1.3: Поддерживается с версии Weblate 1.3.

Пример конфигурации Apache для обслуживания Weblate по пути ``/weblate`. Снова используя модуль mod_wsgi (пример конфигурации взят из файла `weblate/examples/apache-path.conf`):

```
#
# VirtualHost for Weblate, running under /weblate path
#
# This example assumes Weblate is installed in virtualenv in /home/weblate/weblate-
↪env
# and DATA_DIR is set to /home/weblate/data, please adjust paths to match your␣
↪setup.
#
<VirtualHost *:80>
    ServerAdmin admin@weblate.example.org
    ServerName weblate.example.org

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

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

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

    # Path to your Weblate virtualenv
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    WSGIDaemonProcess weblate python-home=/home/weblate/weblate-env
    WSGIProcessGroup weblate
    WSGIApplicationGroup %{GLOBAL}

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

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

</VirtualHost>
```

Кроме того, вам нужно будет подправить `weblate/settings.py`:

```
URL_PREFIX = '/weblate'
```

2.1.11 Фоновые задачи с использованием Celery

Добавлено в версии 3.2.

Для обработки фоновых задач Weblate использует Celery. Пример настроек поставляется с готовой конфигурацией, которая обрабатывает все задачи на месте, но для рабочей среды вы захотите изменить их на что-то более разумное.

Типичная установка, использующая в качестве серверной части Redis, выглядит так:

```
CELERY_TASK_ALWAYS_EAGER = False
CELERY_BROKER_URL = 'redis://localhost:6379'
CELERY_RESULT_BACKEND = CELERY_BROKER_URL
```

Также вы должны запустить рабочего Celery для обработки задач и запуска запланированных задач, это можно сделать непосредственно в командной строке (что в основном полезно при отладке или разработке):

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

Запуск Celery в качестве системного сервиса

Скорее всего вы захотите запустить Celery в качестве демона, и как выполнить ваше желание описывается в разделе Демонизация документации Celery. Для наиболее распространённой установки Linux с использованием systemd вы можете воспользоваться примерами перечисленных ниже файлов, поставляемых в каталоге `examples`.

Модуль systemd, размещаемый по пути `/etc/systemd/system/celery-weblate.service`:

```
[Unit]
Description=Celery Service (Weblate)
After=network.target

[Service]
Type=forking
User=weblate
Group=weblate
EnvironmentFile=/etc/default/celery-weblate
```

(continues on next page)

(продолжение с предыдущей страницы)

```
WorkingDirectory=/home/weblate
RuntimeDirectory=celery
RuntimeDirectoryPreserve=restart
LogsDirectory=celery
ExecStart=/bin/sh -c '${CELERY_BIN} multi start ${CELERYD_NODES} \
  -A ${CELERY_APP} --pidfile=${CELERYD_PID_FILE} \
  --logfile=${CELERYD_LOG_FILE} --loglevel=${CELERYD_LOG_LEVEL} ${CELERYD_OPTS}'
ExecStop=/bin/sh -c '${CELERY_BIN} multi stopwait ${CELERYD_NODES} \
  --pidfile=${CELERYD_PID_FILE}'
ExecReload=/bin/sh -c '${CELERY_BIN} multi restart ${CELERYD_NODES} \
  -A ${CELERY_APP} --pidfile=${CELERYD_PID_FILE} \
  --logfile=${CELERYD_LOG_FILE} --loglevel=${CELERYD_LOG_LEVEL} ${CELERYD_OPTS}'

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

Конфигурация окружения, размещаемая по пути `/etc/default/celery-weblate`:

```
# Name of nodes to start
CELERYD_NODES="celery notify 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 concurency if you get weblate.E019
CELERYD_OPTS="--beat:celery --queues:celery=celery --prefetch-multiplier:celery=4 \
    --queues:notify=notify --prefetch-multiplier:notify=10 \
    --queues:translate=translate --prefetch-multiplier:translate=4 \
    --concurrency:backup=1 --queues:backup=backup  --prefetch-multiplier:backup=2"

# Logging configuration
# - %n will be replaced with the first part of the nodename.
# - %I will be replaced with the current child process index
#   and is important when using the prefork pool to avoid race conditions.
CELERYD_PID_FILE="/var/run/celery/weblate-%n.pid"
CELERYD_LOG_FILE="/var/log/celery/weblate-%n%I.log"
CELERYD_LOG_LEVEL="INFO"

# Internal Weblate variable to indicate we're running inside Celery
CELERY_WORKER_RUNNING="1"
```

Конфигурация ротации журналов, размещаемая по пути `/etc/logrotate.d/celery`:

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

Примечание: Процесс Celery должен выполняться под тем же пользователем, под которым работает Weblate и процесс WSGI, иначе файлы в каталоге *DATA_DIR* будут сохраняться с разными владельцами, что приведёт к проблемам во время выполнения.

Периодические задачи при помощи Celery beat

Weblate поставляется со встроенной конфигурацией планировщика задач. Вы, однако, можете определить в `settings.py` дополнительные задачи, для примера смотрите раздел *Отложенные коммиты*.

Предполагается, что задачи будут выполняться демоном Celery beat. В случае, если он не работает должным образом, он может быть не запущен или его база данных была повреждена. В этом случае проверьте журналы запуска Celery, чтобы выяснить первопричину проблемы.

Мониторинг состояния Celery

Вы можете использовать команду *celery_queues* для просмотра текущей длины очереди задач Celery. В случае, если очередь станет слишком длинной, вы также увидите ошибку конфигурации в интерфейсе администратора.

> **Предупреждение:** Ошибки Celery по умолчанию записываются только в журнал Celery и пользователю не видны. В случае, если вы хотите видеть сводку по таким ошибкам, рекомендуется настроить *сбор отчётов об ошибках*.

См.также:

Конфигурация и настройки по умолчанию, Руководство по рабочим, Демонизация, Руководство по мониторингу и управлению, *celery_queues*

2.1.12 Мониторинг Weblate

Weblate предоставляет URL-адрес `/healthz/`, который можно использоваться для простой проверки его работоспособности, например, при использовании Kubernetes.

2.1.13 Сбор отчётов об ошибках

Weblate, как и любое другое программное обеспечение, может дать сбой. Для сбора полезной информации о сбоях мы рекомендуем использовать услуги сторонних сервисов. Особенно они полезны в случае сбоя в задачах Celery, которые в противном случае просто записали бы ошибку в журнал, и вы не получили бы о ней никакого уведомления. Weblate поддерживает следующие сервисы:

Sentry

В Weblate встроена поддержка Sentry. Для его использования достаточно установить параметр *SENTRY_DSN* в `settings.py`:

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

Rollbar

Weblate имеет встроенную поддержку Rollbar. Для его использования достаточно следовать инструкциям из документа`Уведомления Rollbar для Python'а <https://docs.rollbar.com/docs/python/>`_.

Вкратце, вам необходимо дописать в `settings.py`:

```
# Add rollbar as last middleware:
MIDDLEWARE = [
    # … other middleware classes …
    'rollbar.contrib.django.middleware.RollbarNotifierMiddleware',
]

# Configure client access
ROLLBAR = {
    'access_token': 'POST_SERVER_ITEM_ACCESS_TOKEN',
    'client_token': 'POST_CLIENT_ITEM_ACCESS_TOKEN',
    'environment': 'development' if DEBUG else 'production',
    'branch': 'master',
    'root': '/absolute/path/to/code/root',
}
```

Всё остальное интегрируется автоматически, теперь у вас будут собираться ошибки как со стороны сервера, так и со стороны клиента.

2.1.14 Перенос Weblate на другой сервер

Перенос Weblate на другой сервер должен быть достаточно простой операцией, однако он хранит данные в нескольких местах, переносить которые следует с некоторой осторожностью. Лучше всего для переноса на время остановить Weblate.

Перенос базы данных

В зависимости от серверной части вашей базы данных, у вас может быть несколько вариантов переноса базы данных. Самый простой из них — это сделать дамп базы на старом сервере и импортировать его на новый. В качестве альтернативы вы можете использовать репликацию, если ваша база данных её поддерживает.

Наилучшим подходом будет использование родных инструментов для работы с базами данных, так как они обычно наиболее эффективны (например, **mysqldump** или **pg_dump**). Если вы выполнить перенос между разными базами данных, то единственным вариантом может оказаться использование команд управления Django для создания дампа и импорта базы данных:

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

Перенос репозиториев системы контроля версий

Также необходимо перенести репозитории системы контроля версий, хранящиеся в каталоге `DATA_DIR`. Вы можете просто скопировать их, либо воспользоваться командой **`rsync`** для более эффективного выполнения переноса.

Прочие заметки

Не забудьте переместить и другие сервисы, которые Weblate мог использовать, например, Redis, задачи Cron или модули пользовательской аутентификации.

2.2 Развёртывание Weblate

Weblate легко устанавливается в вашем облаке. Пожалуйста, изучите подробное руководство по вашей платформе:

- *Установка с помощью Docker'а*
- *Установка на OpenShift*

2.2.1 Чарт Helm

Вы можете установить Weblate на Kubernetes с помощью Helm. Подробные инструкции смотрите в документе <https://github.com/WeblateOrg/helm/tree/master/charts/weblate>.

2.2.2 Стек Bitnami Weblate

Bitnami предоставляет стек Weblate для многих платформ по адресу <https://bitnami.com/stack/weblate>. Во время установки настройки будут скорректированы, дополнительную документацию смотрите в <https://bitnami.com/stack/weblate/README.txt>.

2.2.3 Weblate в YunoHost

Проект самостоятельного хостинга YunoHost предоставляет пакет для Weblate. После установки YunoHost вы сможете установить Weblate как любое другое приложение. Он предоставит вам полностью рабочий стек с резервным копированием и восстановлением, но вам всё равно может понадобиться отредактировать файл настроек для определённых целей использования.

Вы можете воспользоваться интерфейсом администрирования или этой кнопкой (она приведёт вас на ваш сервер):

Также можно использовать интерфейс командной строки:

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

2.3 Обновление Weblate

2.3.1 Обновления образа Docker'a

В официальный образ Docker'a (смотрите раздел *Установка с помощью Docker'a*) уже интегрированы все шаги для обновления. Вручную делать ничего не надо, за исключением извлечения последней версии.

2.3.2 Общие инструкции по обновлению

Перед обновлением, пожалуйста, проверьте текущие *зависимости программного обеспечения*, поскольку они могли измениться. После того, как все зависимости будут установлены или обновлены, пожалуйста, приведите свой `settings.py` в соответствие с изменениями в конфигурации (правильные значения смотрите в файле `settings_example.py`).

Перед обновлением всегда проверяйте *инструкции для конкретных версий*. Если вы пропустили какие-то версии, пожалуйста, при обновлении проследуйте инструкциям для всех версий, которые вы пропустили. Иногда, чтобы обеспечить плавный переход, лучше сперва обновиться до какой-нибудь промежуточной версии. Обновление через несколько версий должно работать, но оно протестировано не так хорошо, как обновление на одну версию.

Примечание: Перед обновлением рекомендуется выполнить полное резервное копирование базы данных, чтобы её можно было откатить в случае неудачи обновления, смотрите раздел *Резервное копирование и перенос Weblate*.

1. Остановите процессы wsgi и Celery. Обновление может выполнять несовместимые изменения в базе данных, поэтому во время обновления всегда безопаснее избегать выполнения старых процессов.
2. Обновите код Weblate.

   При установке через pip это можно сделать следующим образом:

   ```
   pip install -U Weblate
   ```

   При использовании кода из Git'a вам нужно будет извлечь новый исходный код и обновить вашу установку:

   ```
   cd weblate-src
   git pull
   # Update Weblate inside your virtualenv
   . ~/weblate-env/bin/pip install -e .
   # Install dependencies directly when not using virtualenv
   pip install --upgrade -r requirements.txt
   ```

3. Обновите файл конфигурации, пример смотрите в файле `settings_example.py` или обратитесь к разделу *Инструкции для конкретных версий* для получения информации о необходимых шагах.
4. Обновите структуру базы данных:

   ```
   weblate migrate --noinput
   ```

5. Соберите обновлённые статические файлы (смотрите разделы *Запуск сервера* и *Обслуживание статических файлов*):

   ```
   weblate collectstatic --noinput
   ```

6. Сожмите файлы JavaScript и CSS (необязательно, смотрите раздел *Сжатие клиентских активов*):

   ```
   weblate compress
   ```

7. Если вы используете версию из Git'a, вам также при каждом обновлении следует заново сгенерировать файлы локалей. Вы можете сделать это, выполнив следующую команду:

```
weblate compilemessages
```

8. Убедитесь, что ваша установка в порядке (также смотрите раздел *Рабочая среда*):

```
weblate check --deploy
```

9. Перезапустите рабочий процесс celery (смотрите раздел *Фоновые задачи с использованием Celery*).

2.3.3 Инструкции для конкретных версий

Обновление с ветки 2.x

При обновлении с выпусков ветки 2.x всегда сначала выполняйте обновление до версии 3.0.1, а затем продолжайте обновление в ветке 3.x. Обновления, пропустившие этот этап, не поддерживаются и будут работать неправильно.

См.также:

Обновление с версии 2.20 до версии 3.0 в документации Weblate 3.0

Обновление с ветки 3.x

При обновлении с выпусков ветки 3.x всегда сначала выполняйте обновление до версии 4.0.4 или 4.1.1, а затем продолжайте обновление в ветке 4.x. Обновления, пропустившие этот этап, не поддерживаются и будут работать неправильно.

См.также:

Обновление с версии 3.11 до версии 4.0 в документации Weblate 4.0

Обновление с версии 4.0 до версии 4.1

Для выполнения обновления, пожалуйста, следуйте *общим инструкциям по обновлению*.

Заметные изменения конфигурации или зависимостей:

- В файл `settings_example.py` внесено несколько изменений, наиболее заметные из которых касаются промежуточного программного обеспечения, пожалуйста, соответствующим образом подправьте свой файл настроек.
- Появилась поддержка нескольких новых форматов файлов, возможно, вы захотите включить их обработку в случае, если вы изменяли значение параметра *`WEBLATE_FORMATS`*.
- Появились новые проверки качества, возможно, вы захотите включить их, если вы изменяли параметр *`CHECK_LIST`*.
- Параметр `DEFAULT_THROTTLE_CLASSES` был изменён, чтобы разрешать сообщать об ограничении частоты запросов к API.
- Появились новые зависимости и обновились старые.
- Произошло изменение параметра `INSTALLED_APPS`.
- Машинный перевод *DeepL* теперь по умолчанию использует API версии 2, вам может понадобиться подправить параметр *`MT_DEEPL_API_VERSION`* на тот случай, если ваша текущая подписка DeepL не поддерживает эту версию.

См.также:

Общие инструкции по обновлению

Обновление с версии 4.1 до версии 4.2

Для выполнения обновления, пожалуйста, следуйте *общим инструкциям по обновлению*.

Заметные изменения конфигурации или зависимостей:

- Обновление с выпусков ветки 3.x больше не поддерживается, пожалуйста, сначала обновитесь до версии 4.0 или 4.1.
- Появились новые зависимости и обновились старые.
- В файл `settings_example.py` внесено несколько изменений, наиболее заметные из которых касаются нового промежуточного программного обеспечения и изменения порядка приложений.
- Ключи для форматов на основе JSON больше не включают ведущую точку. Строки корректируются во время миграции базы данных, но может понадобиться откорректировать внешние компоненты, если вы полагались на значения ключей после экспорта или в API.
- Конфигурация Celery была изменена, чтобы больше не использовать очередь `memory`. Пожалуйста, подправьте свои скрипты запуска и значение параметра `CELERY_TASK_ROUTES`.
- Домен Weblate'а теперь настраивается через параметры, смотрите описание параметра *`SITE_DOMAIN`* (или переменной окружения *`WEBLATE_SITE_DOMAIN`*). Перед запуском Weblate их необходимо настроить.
- The username and email fields on user database now should be case insensitive unique. It was mistakenly not enforced with PostgreSQL.

См.также:

Общие инструкции по обновлению

Обновление с версии 4.2 до версии 4.3

Для выполнения обновления, пожалуйста, следуйте *общим инструкциям по обновлению*.

Заметные изменения конфигурации или зависимостей:

- В проверках качества произошли некоторые изменения, возможно, вы захотите включить их, если вы изменяли параметр *`CHECK_LIST`*.
- The source language attribute was moved from project to a component what is exposed in the API. You will need to update *Клиент 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).
- Произошло изменение параметра `INSTALLED_APPS`.
- There is a new setting *`SESSION_COOKIE_AGE_AUTHENTICATED`* which complements `SESSION_COOKIE_AGE`.

См.также:

Общие инструкции по обновлению

2.3.4 Обновление с Python'а 2 на Python 3

Weblate больше не поддерживает Python старше версии 3.5. В случае, если вы всё ещё работаете на более старой версии, пожалуйста, сначала выполните миграцию на Python 3 на существующей версии и обновите её только после перехода. Смотрите раздел Обновление с Python'а 2 на Python 3 в документации к Weblate 3.11.1.

2.3.5 Переход с других баз данных на PostgreSQL

Если вы используете Weblate базой данных, отличной от PostgreSQL, вам следует перейти на PostgreSQL, поскольку в версии 4.0 это будет единственная поддерживаемая база данных. Следующие шаги помогут вам осуществить перенос данных между базами данных. Пожалуйста, не забудьте перед переездом остановить веб-сервер и сервер Celery, в противном случае у вас могут появиться несогласованные данные.

Создание базы данных в PostgreSQL

Обычно рекомендуется запускать Weblate в отдельной базе данных и под отдельной учётной записью:

```
# If PostgreSQL was not installed before, set the main password
sudo -u postgres psql postgres -c "\password postgres"

# Create a database user called "weblate"
sudo -u postgres createuser -D -P weblate

# Create the database "weblate" owned by "weblate"
sudo -u postgres createdb -O weblate weblate
```

Переход с использованием JSON-дампов Django

Простейшим способом перехода является использование JSON-дампов Django. Они хорошо работают для небольших установок. На больших сайтах вы вместо него можете использовать утилиту pgloader, смотрите раздел *Переход на PotsgreSQL с помощью pgloader*.

1. Добавьте PostgreSQL в файл `settings.py` в качестве дополнительного подключения к базе данных:

```
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',
```

(continues on next page)

(продолжение с предыдущей страницы)

```
            # Change connection timeout in case you get MySQL gone away error:
            'connect_timeout': 28800,
        }
    },
    'postgresql': {
        # Database engine
        'ENGINE': 'django.db.backends.postgresql',
        # Database name
        'NAME': 'weblate',
        # Database user
        'USER': 'weblate',
        # Database password
        'PASSWORD': 'password',
        # Set to empty string for localhost
        'HOST': 'database.example.com',
        # Set to empty string for default
        'PORT': '',
    }
}
```

2. Запустите миграции и удалите все данные, вставленные в таблицы:

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

3. Сделайте дамп прежней базы данных и импортируйте его в PostgreSQL

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

4. Подправьте параметр `DATABASES`, задав в нём использование по умолчанию только базы данных PostgreSQL и удалив старое соединение.

Теперь Weblate должен быть готов к запуску из базы данных PostgreSQL.

Переход на PotsgreSQL с помощью pgloader

pgloader — это универсальный инструмент миграции для переноса данных на PostgreSQL. Вы можете использовать его для переноса базы данных Weblate.

1. Подправьте свой файл `settings.py`, задав в нём использование базы данных PostgreSQL.
2. Перенесите схему в базу данных PostgreSQL:

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

3. Запустите pgloader для переноса данных. Для переноса базы данных можно использовать следующий скрипт, но вы, возможно, захотите побольше узнать о pgloader'е, чтобы понять, что он делает, и подстроить его для соответствия вашим настройкам:

```
LOAD DATABASE
     FROM      mysql://weblate:password@localhost/weblate
     INTO postgresql://weblate:password@localhost/weblate

WITH include no drop, truncate, create no tables, create no indexes, no␣
↪foreign keys, disable triggers, reset sequences, data only

ALTER SCHEMA 'weblate' RENAME TO 'public'
;
```

2.3.6 Переход с Pootle

Поскольку изначально Weblate был написан как замена для Pootle, он поддерживает перенос учётных записей пользователей из Pootle. Вы можете создать дамп пользователей Pootle, а затем импортировать его, используя команду Django `importusers`.

2.4 Резервное копирование и перенос Weblate

2.4.1 Автоматическое резервное копирование с помощью BorgBackup

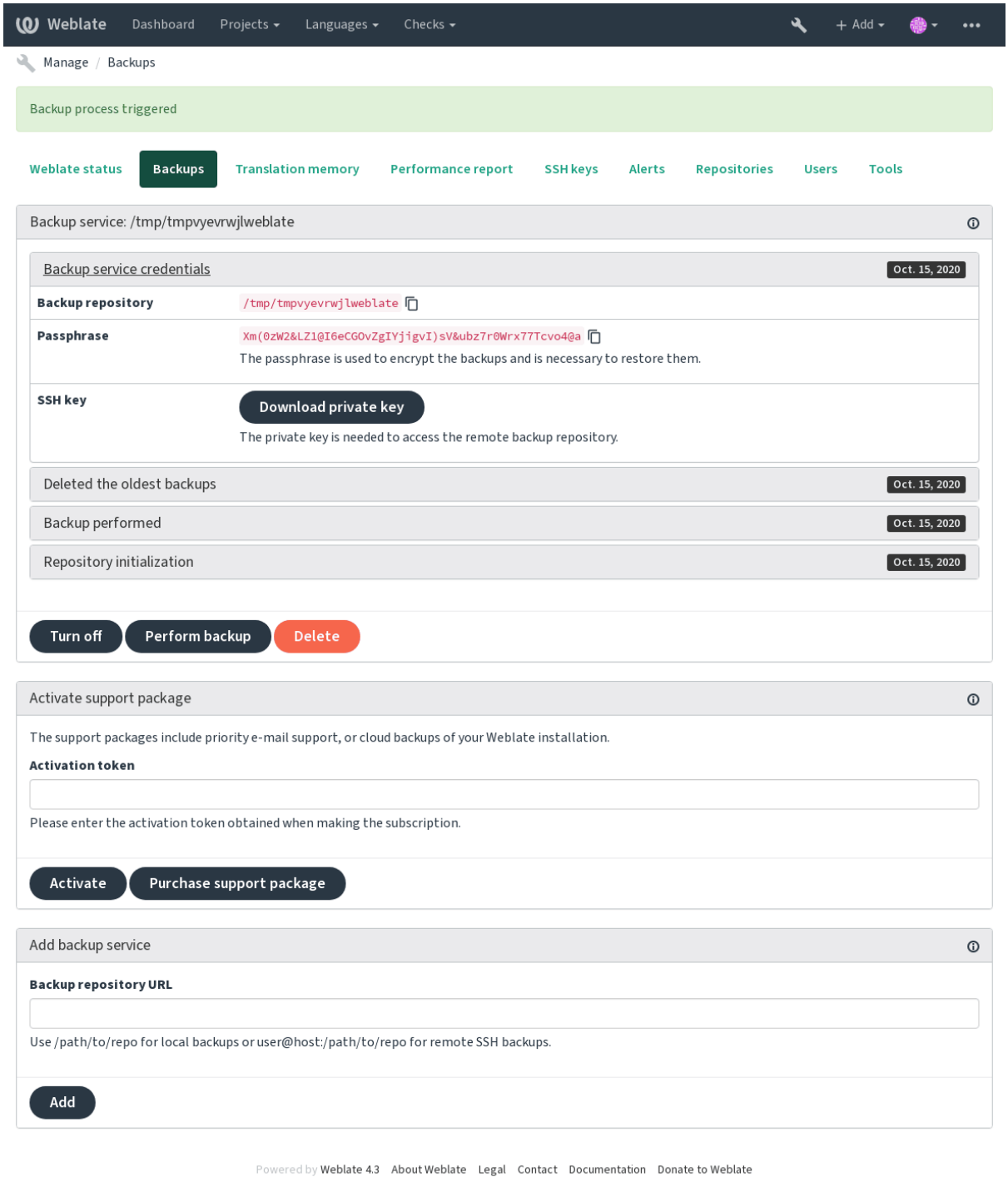
Добавлено в версии 3.9.

В Weblate встроена поддержка создания сервисных резервных копий с помощью BorgBackup. Borg создаёт компактные зашифрованные резервные копии, которые можно безопасно хранить в облаке. Управлять резервными копиями можно через интерфейс управления на вкладке *Резервные копии*.

Предупреждение: В автоматизированные резервные копии включается только база данных PostgreSQL. Резервное копирование других движков баз данных необходимо выполнять вручную. Рекомендуется перейти на PostgreSQL, смотрите разделы *Настройка базы данных для Weblate* и *Переход с других баз данных на PostgreSQL*.

Резервные копии, которые делает Borg, являются инкрементальными, и Weblate настроен на сохранение следующих резервных копий:

- 14 ежедневных резервных копий
- 8 еженедельных резервных копий
- 6 ежемесячных резервных копий



Ключ шифрования Borg

BorgBackup создаёт зашифрованные резервные копии и без кодовой фразы восстановить резервную копию вы не сможете. Парольная фраза генерируется при добавлении новой службы резервного копирования, вы должны скопировать её и сохранить в безопасном месте.

В случае, если вы используете *хранилище резервных копий на Weblate.org*, пожалуйста, сделайте резервную копию вашего закрытого SSH-ключа — он используется для доступа к вашим резервным копиям.

См.также:

borg init

2.4.2 Предоставляемое Weblate'ом хранилище резервных копий

Простейший подход к резервному копированию вашего экземпляра Weblate заключается в покупке сервиса резервного копирования на weblate.org. Процесс активации резервного копирования выполняется в несколько шагов:

1. Покупка услуги резервного копирования на https://weblate.org/support/#backup.
2. Ввод полученного ключа в интерфейсе управления, смотрите раздел *Integrating support*.
3. Weblate подключится к облачному сервису и получит информацию для доступа к резервным копиям.
4. Включение новой конфигурации резервного копирования на вкладке *Резервные копии*.
5. Резервное копирование учётных данных Borg, чтобы иметь возможность восстанавливать резервные копии, смотрите раздел *Ключ шифрования Borg*.

Подсказка: Шаг ручного включения предусмотрен для вашей безопасности. Без вашего согласия никакие данные в хранилище резервных копий, полученное в процессе регистрации, не отправляются.

2.4.3 Использование собственного хранилища резервных копий

Также вы можете использовать своё собственное хранилище для резервного копирования. Загрузка резервных копий на удаленный сервер может осуществляться через SSH, на целевом сервере должен быть установлен BorgBackup.

См.также:

Раздел Общие сведения в документации Borg

Локальная файловая система

Для локальной резервной копии рекомендуется указывать абсолютный путь, например */путь/к/резервной_копии*. Каталог должен быть доступен на запись для пользователя, под которым запущен Weblate (смотрите раздел *Разрешения файловой системы*). В случае, если каталог не существует, Weblate попытается его создать, но для этого ему понадобятся разрешения.

Подсказка: Если Weblate запущен в Docker'е, убедитесь, что местоположение резервных копий отображается на том из контейнера Weblate'а. В противном случае при перезагрузке контейнера Docker выбросит резервные копии.

Одним из вариантов является размещение резервных копий в существующем томе. Подойдёт, к примеру, `/app/data/borgbackup`. Это существующий том в контейнере.

Также вы можете добавить в файл compose Docker'а новый контейнер для резервных копий и использовать, например, путь `/borgbackup`:

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

The directory where backups will be stored have to be owned by UID 1000, otherwise Weblate will not be able to write the backups there.

Удалённые резервные копии

Поддерживается удалённое резервное копирование с помощью SSH. На SSH-сервере должен быть установлен BorgBackup. Weblate подключается к серверу, используя ключ SSH, пожалуйста, убедитесь, что Weblate'овский ключ SSH принимается сервером (смотрите раздел *SSH-ключ Weblate*).

Подсказка: *Предоставляемое Weblate'ом хранилище резервных копий* предоставляет вам автоматическое удалённое резервное копирование.

2.4.4 Восстановление из резервной копии Borg

1. Восстановите доступ к своему репозиторию резервных копий и подготовьте свою парольную фразу для резервного копирования.
2. Получите список существующих на сервере резервных копий с помощью команды `borg list РЕПОЗИТОРИЙ`.
3. Восстановите нужную резервную копию в текущий каталог с помощью команды `borg extract РЕПОЗИТОРИЙ::АРХИВ`.
4. Восстановите базу данных из SQL-дампа, размещённого в каталоге `backup` каталога данных Weblate (смотрите раздел *Дампы данных для резервного копирования*).
5. Скопируйте конфигурацию Weblate (`backups/settings.py`, смотрите раздел *Дампы данных для резервного копирования*) в нужное место, смотрите раздел *Настройка конфигурации*.
6. Скопируйте весь восстановленный каталог данных в место, заданное параметром `DATA_DIR`.

Сеанс Borg может выглядеть так:

```
$ borg list /tmp/xxx
Enter passphrase for key /tmp/xxx:
2019-09-26T14:56:08                  Thu, 2019-09-26 14:56:08␣
↪[de0e0f13643635d5090e9896bdaceb92a023050749ad3f3350e788f1a65576a5]
$ borg extract /tmp/xxx::2019-09-26T14:56:08
Enter passphrase for key /tmp/xxx:
```

См.также:

borg list, borg extract

2.4.5 Ручное резервное копирование

В зависимости от того, что вы хотите сохранить, создайте резервную копию типов данных, которые Weblate сохраняет в соответствующих местах.

Подсказка: Если вы выполняете резервное копирование вручную, вы можете подавить предупреждение Weblate об отсутствии резервного копирования, добавив проверку `weblate.I028` к параметру `SILENCED_SYSTEM_CHECKS` в `settings.py` или к переменной окружения `WEBLATE_SILENCED_SYSTEM_CHECKS` Docker'a.

```
SILENCED_SYSTEM_CHECKS.append("weblate.I028")
```

База данных

Фактическое расположение хранилища зависит от настроек вашей базы данных.

База данных является самым важным хранилищем. Настройте регулярное резервное копирование вашей базы данных, без неё все настройки переводов исчезнут.

Встроенные средства базы данных для резервного копирования

Рекомендуемый подход заключается в создании дампа базы данных с помощью родных для базы данных инструментов, таких как **pg_dump** или **mysqldump**. Обычно он работает лучше, чем резервная копия Django, и восстанавливает полные таблицы со всеми данными.

Эту резервную копию вы можете восстановить в новом выпуске Weblate, при запуске через `migrate` он выполнит все необходимые миграции. За более подробной информацией о том, как выполнить обновление между версиями обратитесь, пожалуйста, к разделу *Обновление Weblate*.

Резервное копирование базы данных средствами Django

В качестве альтернативы вы можете создать резервную копию базы данных при помощи Django'вской команды `dumpdata`. Резервное копирование этим способом не зависит от базы данных и может быть использовано в случае, если вы захотите сменить базу данных.

До восстановления вам следует убедиться, что у вас запущена ровно та же самая версия Weblate, которая использовалась при резервном копировании. Это необходимо, так как структура базы данных меняется между выпусками, и если вы не выполните это условие, то, в конечном итоге, повредите данные тем или иным образом. После установки такой же версии запустите все миграции базы данных с помощью команды `migrate`.

После этого некоторые записи уже будут созданы в базе данных, и они также попадут в резервную копию базы данных. Такие записи рекомендуется удалять вручную в оболочке управления (смотрите раздел *Вызов команд управления*):

```
weblate shell
>>> from weblate.auth.models import User
>>> User.objects.get(username='anonymous').delete()
```

Файлы

Если у вас достаточно места для резервного копирования, просто создайте резервную копию всего каталога *DATA_DIR*. Это беспроигрышный вариант, даже если он включает в себя некоторые файлы, которые вам не нужны. В следующих разделах подробно описано, для каких файлов нужно создавать резервные копии, а какие можно и пропустить.

Дампы данных для резервного копирования

Хранятся в *DATA_DIR* `/backups`.

Weblate сбрасывает сюда различные данные, и для создания более полных резервных копий вы можете включить эти файлы. Файлы обновляются ежедневно (требуется отдельный запущенный сервер Celery beat, смотрите раздел *Фоновые задачи с использованием Celery*). В настоящее время здесь хранятся:

- Настройки Weblate в виде файла `settings.py` (также здесь под именем `settings-expanded.py` лежит обработанная версия этого файла).
- Резервная копия базы данных PostgreSQL в виде файла `database.sql`.

Резервные копии базы данных по умолчанию сохраняются в виде простого текста, но их также можно сжать или полностью выключить, устанавливая параметр *DATABASE_BACKUP*.

Репозитории системы контроля версий

Хранятся в *DATA_DIR* /vcs.

Репозитории системы контроля версий содержат копии ваших вышестоящих репозиториев с изменениями из Weblate'a. Если вы включили для всех ваших компонентов перевода отправку изменений в вышестоящий репозиторий при коммите, в этот репозиторий включаются все изменения из Weblate'a, поэтому на стороне Weblate'a вам уже не нужно создавать резервные копии репозиториев. Они могут быть повторно склонированы из вышестоящих репозиториев без потери данных.

Ключи SSH и GPG

Хранятся в *DATA_DIR* /ssh и *DATA_DIR* /home.

Если вы используете сгенерированные Weblate ключи SSH или GPG, вы должны создать резервную копию этих каталогов, иначе вы потеряете закрытые ключи и вам придётся заново сгенерировать новые.

Загруженные пользователем файлы

Хранятся в *DATA_DIR* /media.

Вы должны сделать резервные копии загруженных пользователями файлов (например, *визуальных контекстов для строк*).

Задачи Celery

Очередь задач Celery может содержать некоторую информацию, но обычно её не требуется сохранять в резервной копии. В худшем случае вы потеряете обновления, которые ещё не были занесены в память переводов. Поскольку при восстановлении в любом случае рекомендуется выполнять обновления полнотекстового индекса или репозитория, нет ничего страшного в том, что они потеряются.

См.также:

Фоновые задачи с использованием Celery

Командная строка для ручного резервного копирования

Используя задание в cron'e, вы можете задать команду bash'a, которая будет выполняться каждый день, например:

```
$ XZ_OPT="-9" tar -Jcf ~/backup/weblate-backup-$(date -u +%Y-%m-%d_%H%M%S).xz␣
↪backups vcs ssh home media fonts secret
```

Строка между кавычками после XZ_OPT позволяет вам выбрать параметры для запуска xz, например, объём памяти, используемый для сжатия; смотрите руководство на странице https://linux.die.net/man/1/xz

Вы можете настроить список папок и файлов в соответствии со своими потребностями. Например, чтобы избежать сохранения памяти переводов (в папке резервных копий), вы можете использовать следующую команду:

```
$ XZ_OPT="-9" tar -Jcf ~/backup/weblate-backup-$(date -u +%Y-%m-%d_%H%M%S).xz␣
↪backups/database.sql backups/settings.py vcs ssh home media fonts secret
```

2.4.6 Восстановление ручных резервных копий

1. Восстановите все данные, сохранённые вами в резервной копии.
2. Обновите все репозитории с помощью команды `updategit`.

```
weblate updategit --all
```

2.4.7 Перенос установки Weblate

Переместите свою установку на другую систему, следуя приведённым выше инструкциям по резервному копированию и восстановлению из резервной копии.

См.также:

Обновление с Python'а 2 на Python 3, *Переход с других баз данных на PostgreSQL*

2.5 Аутентификация

2.5.1 Регистрация пользователя

В установке по умолчанию Weblate использует python-social-auth, форму на сайте для обработки регистраций новых пользователей. После подтверждения своей электронной почты новый пользователь может вносить свой вклад в переводы или пройти аутентификацию, воспользовавшись одним из сторонних сервисов.

Также вы можете отключить регистрацию новых пользователей с помощью установки параметра `REGISTRATION_OPEN`.

Попытки аутентификации учитываются при применении *ограничений на частоту запросов*.

2.5.2 Способы аутентификации

Для аутентификации используется встроенное решение Django, включающее в себя для этого аутентификацию от различных социальных сетей. Использование этого решения означает, что вы сможете импортировать себе базу данных пользователей других проектов, написанных на Django (смотрите раздел *Переход с Pootle*).

Кроме того, аутентификация в Django может быть настроена и с использованием других способов.

См.также:

В разделе *Параметры аутентификации* описано, как настроить аутентификацию в официальном образе Docker'а.

2.5.3 Социальная аутентификация

Благодаря пакету Python Social Auth, Weblate поддерживает аутентификацию с использованием множества сторонних сервисов, таких как GitLab, Ubuntu, Fedora и другие.

Для получения общих инструкций по конфигурации ознакомьтесь, пожалуйста, с их документацией в разделе Фреймворк Django.

Примечание: По умолчанию Weblate полагается на сторонние сервисы аутентификации для предоставления подтверждённого адреса электронной почты. Если некоторые сервисы, которые вы хотите использовать, это не поддерживают, включите принудительную проверку адреса электронной почты на стороне Weblate, настроив их параметр FORCE_EMAIL_VALIDATION. Например:

```
SOCIAL_AUTH_OPENSUSE_FORCE_EMAIL_VALIDATION = True
```

См.также:

Конвейер

Включить отдельный способ довольно просто, это всего лишь вопрос добавления записи в параметр `AUTHENTICATION_BACKENDS` и, возможно, ключей, необходимых для заданного способа аутентификации. Пожалуйста, обратите внимание, что некоторые способы по умолчанию не предоставляют адрес электронной почты пользователя, в этом случае вы должны запросить её явно, иначе Weblate не сможет правильно указать авторство выполненных пользователями переводов.

См.также:

Бэкэнд пакета Python Social Auth

Аутентификация OpenID

Для сервисов, основанных на OpenID, обычно это просто вопрос её включения. Следующий блок кода включает аутентификацию OpenID для OpenSUSE, Fedora и 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',
)
```

См.также:

OpenID

Аутентификация GitHub

Вам нужно будет зарегистрировать приложение в GitHub'е, а затем рассказать Weblate'у все его секреты:

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    'social_core.backends.github.GithubOAuth2',
    'social_core.backends.email.EmailAuth',
    'weblate.accounts.auth.WeblateUserBackend',
)

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

GitHub должен быть сконфигурирован так, чтобы URL-адрес обратного вызова указывал на `https://example.com/accounts/complete/github/`.

Примечание: Weblate предоставляет URL-адрес обратного вызова, вызываемый во время аутентификации, в который включён настроенный домен. В случае, если вы получаете ошибки о несовпадении URL, возможно, вы захотите это исправить, смотрите раздел *Установка правильного домена сайта*.

См.также:

GitHub

Аутентификация Bitbucket

Вам нужно будет зарегистрировать приложение в Bitbucket'е, а затем рассказать Weblate'у все его секреты:

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

# Social auth backends setup
SOCIAL_AUTH_BITBUCKET_KEY = 'Bitbucket Client ID'
SOCIAL_AUTH_BITBUCKET_SECRET = 'Bitbucket Client Secret'
SOCIAL_AUTH_BITBUCKET_VERIFIED_EMAILS_ONLY = True
```

Примечание: Weblate предоставляет URL-адрес обратного вызова, вызываемый во время аутентификации, в который включён настроенный домен. В случае, если вы получаете ошибки о несовпадении URL, возможно, вы захотите это исправить, смотрите раздел *Установка правильного домена сайта*.

См.также:

Bitbucket

Google OAuth 2

Для использования Google OAuth 2 необходимо зарегистрировать приложение на <https://console.developers.google.com/> и включить Google+ API.

URL-адрес перенаправления — `https://СЕРВЕР WEBLATE/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'
```

Примечание: Weblate предоставляет URL-адрес обратного вызова, вызываемый во время аутентификации, в который включён настроенный домен. В случае, если вы получаете ошибки о несовпадении URL, возможно, вы захотите это исправить, смотрите раздел *Установка правильного домена сайта*.

См.также:

Google

Facebook OAuth 2

Как обычно с сервисами OAuth 2, вам необходимо зарегистрировать ваше приложение в Facebook. После этого вы сможете настроить Weblate на его использование:

URL-адрес перенаправления — `https://СЕРВЕР WEBLATE/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']
```

Примечание: Weblate предоставляет URL-адрес обратного вызова, вызываемый во время аутентификации, в который включён настроенный домен. В случае, если вы получаете ошибки о несовпадении URL, возможно, вы захотите это исправить, смотрите раздел *Установка правильного домена сайта*.

См.также:

Facebook

GitLab OAuth 2

Для использования GitLab OAuth 2 необходимо зарегистрировать приложение на <https://gitlab.com/profile/applications>.

URL-адрес перенаправления — `https://СЕРВЕР WEBLATE/accounts/complete/gitlab/` и убедитесь, что вы отметили область видимости *read_user*.

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    'social_core.backends.gitlab.GitLabOAuth2',
    'social_core.backends.email.EmailAuth',
    'weblate.accounts.auth.WeblateUserBackend',
)

# Social auth backends setup
SOCIAL_AUTH_GITLAB_KEY = 'Application ID'
SOCIAL_AUTH_GITLAB_SECRET = 'Secret'
SOCIAL_AUTH_GITLAB_SCOPE = ['read_user']

# If you are using your own GitLab
# SOCIAL_AUTH_GITLAB_API_URL = 'https://gitlab.example.com/'
```

Примечание: Weblate предоставляет URL-адрес обратного вызова, вызываемый во время аутентификации, в который включён настроенный домен. В случае, если вы получаете ошибки о несовпадении URL, возможно, вы захотите это исправить, смотрите раздел *Установка правильного домена сайта*.

См.также:

GitLab

Microsoft Azure Active Directory

Weblate можно настроить на использование общих или конкретных арендаторов для аутентификации.

URL-адрес перенаправления — `https://СЕРВЕР WEBLATE/accounts/complete/azuread-oauth2/` для общей и `https://СЕРВЕР WEBLATE/accounts/complete/azuread-tenant-oauth2/` для конкретной для арендатора аутентификации.

```
# Azure AD common

# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.azuread.AzureADOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# OAuth2 keys
SOCIAL_AUTH_AZUREAD_OAUTH2_KEY = ""
SOCIAL_AUTH_AZUREAD_OAUTH2_SECRET = ""
```

```
# Azure AD Tenant

# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.azuread_tenant.AzureADTenantOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# OAuth2 keys
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_KEY = ""
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_SECRET = ""
# Tenant ID
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_TENANT_ID = ""
```

Примечание: Weblate предоставляет URL-адрес обратного вызова, вызываемый во время аутентификации, в который включён настроенный домен. В случае, если вы получаете ошибки о несовпадении URL, возможно, вы захотите это исправить, смотрите раздел *Установка правильного домена сайта*.

См.также:

Microsoft Azure Active Directory

Slack

Для использования Slack OAuth 2 необходимо зарегистрировать приложение на <https://api.slack.com/apps>.

URL-адрес перенаправления — `https://СЕРВЕР WEBLATE/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 = ''
```

Примечание: Weblate предоставляет URL-адрес обратного вызова, вызываемый во время аутентификации, в который включён настроенный домен. В случае, если вы получаете ошибки о несовпадении URL, возможно, вы захотите это исправить, смотрите раздел *Установка правильного домена сайта*.

См.также:

Slack

Отключение аутентификации по паролю

Аутентификацию по электронной почте и паролю можно отключить, удалив `social_core.backends.email.EmailAuth` из параметра `AUTHENTICATION_BACKENDS`. Никогда не удаляйте оттуда `weblate.accounts.auth.WeblateUserBackend`, он необходим для работы основных функций Weblate.

Совет: Вы всё ещё можете использовать аутентификацию по паролю для интерфейса администратора и для пользователей, которых вы создаёте там вручную. Просто зайдите на `/admin/`.

Например, аутентификация с использованием только провайдера openSUSE Open ID может быть достигнута при помощи следующего кода:

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    'social_core.backends.suse.OpenSUSEOpenId',
    'weblate.accounts.auth.WeblateUserBackend',
)
```

2.5.4 Аутентификация по паролю

По умолчанию `settings.py` поставляется с разумным набором проверок в параметре `AUTH_PASSWORD_VALIDATORS`:

- Пароли не могут быть слишком похожи на другие ваши личные данные.
- Пароли должны содержать минимум 10 символов.
- Пароли не могут быть широко распространёнными паролями.
- Пароли не могут быть полностью цифровыми.
- Пароли не могут состоять из одного символа или только из пробелов.
- Пароли не могут совпадать с предыдущими вашими паролями.

Вы можете настроить этот параметр в соответствии с вашей политикой паролей.

Дополнительно можно установить пакет django-zxcvbn-password, который вычисляет вполне реалистичные оценки сложности пароля и позволяет отклонять пароли с оценкой ниже определённого порога.

2.5.5 Аутентификация через SAML

Добавлено в версии 4.1.1.

Пожалуйста, следуйте инструкциям по настройке Python Social Auth. Значимые отличия:

- Weblate поддерживает один IDP, который в `SOCIAL_AUTH_SAML_ENABLED_IDPS` должен называться `weblate`.
- URL-адрес XML с метаданными SAML — `/accounts/metadata/saml/`.
- Следующие параметры заполняются автоматически: `SOCIAL_AUTH_SAML_SP_ENTITY_ID`, `SOCIAL_AUTH_SAML_TECHNICAL_CONTACT` и `SOCIAL_AUTH_SAML_SUPPORT_CONTACT`

Пример конфигурации:

```
# 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_PUBLIC_CERT = "-----BEGIN CERTIFICATE-----"
SOCIAL_AUTH_SAML_SP_PRIVATE_KEY = "-----BEGIN PRIVATE KEY-----"
SOCIAL_AUTH_SAML_ENABLED_IDPS = {
    "weblate": {
        "entity_id": "https://idp.testshib.org/idp/shibboleth",
        "url": "https://idp.testshib.org/idp/profile/SAML2/Redirect/SSO",
        "x509cert": "MIIEDjCCAvagAwIBAgIBADA ... 8Bbnl+ev0peYzxFyF5sQA==",
        "attr_name": "full_name",
        "attr_username": "username",
        "attr_email": "email",
    }
}
```

См.также:

Настройка SAML в Docker'е, SAML

2.5.6 Аутентификация через LDAP

Аутентификацию через LDAP лучше всего реализовать с помощью пакета *django-auth-ldap*. Установить его вы можете как обычно:

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

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

Предупреждение: При использовании django-auth-ldap версии старше 1.3.0, *автоматические прикрепления группы* для новых созданных пользователей будут работать неправильно.

Примечание: У Python'ьего модуля LDAP версии 3.1.0 есть некоторые несовместимости, которые могут помешать вам использовать эту версию. Если вы получили ошибку AttributeError: „module“ object has no attribute „_trace_level“, то может помочь понижение версии python-ldap до 3.0.0.

После установки пакета вы сможете подключить его к системе аутентификации Django:

```
# Add LDAP backed, keep Django one if you want to be able to login
# 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 login
# 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
```

Примечание: Вы должны удалить значение `'social_core.backends.email.EmailAuth'` из параметра `AUTHENTICATION_BACKENDS`, в противном случае пользователи смогут установить в Weblate свой пароль и пройти аутентификацию, используя его. Оставление `'weblate.accounts.auth.WeblateUserBackend'` необходимо для того, чтобы работали разрешения и анонимные пользователи. Также он позволит вам войти в систему, используя учётную запись локального администратора, если вы его создали (например, с помощью команды `createadmin`).

Использование пароля привязки

Если вы не можете использовать для аутентификации прямую привязку, вам нужно будет использовать поиск, и предоставить для поиска привязываемого пользователя. Например:

```
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)")
```

Интеграция с Active Directory

```
import ldap
from django_auth_ldap.config import LDAPSearch, NestedActiveDirectoryGroupType

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

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

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

# Map groups from AD to Weblate
AUTH_LDAP_GROUP_SEARCH = LDAPSearch("OU=Groups,DC=example,DC=com", ldap.SCOPE_
↪SUBTREE, "(objectClass=group)")
AUTH_LDAP_GROUP_TYPE = NestedActiveDirectoryGroupType()
AUTH_LDAP_FIND_GROUP_PERMS = True

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

См.также:

Аутентификация в Django через LDAP, Аутентификация

2.5.7 Аутентификация через CAS

Аутентификация через CAS может быть выполнена с помощью такого пакета, как *django-cas-ng*.

Первый шаг — раскрытие поля электронной почты пользователя через CAS. Раскрытие нужно настроить на самом сервере CAS, и оно требует, чтобы у вас был запущен CAS как минимум версии v2, поскольку CAS версии v1 вообще не поддерживает атрибуты.

Второй шаг — обновление Weblate'а для использования вашего сервера CAS и атрибутов.

Для установки *django-cas-ng* выполните команду:

```
pip install django-cas-ng
```

После установки пакета вы сможете подключить его к системе аутентификации Django, изменив файл `settings.py`:

```
# 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 = (
    ...,
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    'django_cas_ng'
)
```

Наконец, для связывания поля электронной почты с объектом пользователя может быть использован сигнал. Для этого необходимо импортировать сигнал из пакета *django-cas-ng* и подключить к этому сигналу свой код. Выполнение этого действия в файле настроек может вызвать проблемы, поэтому его код предлагается поместить:

- В метод `django.apps.AppConfig.ready()` конфигурации вашего приложения
- В проектовый файл `urls.py` (если у вас нет моделей)

```
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()
```

См.также:

Django CAS NG

2.5.8 Настройка сторонней аутентификации Django

Как правило, с Weblate должен заработать любой плагин аутентификации Django. Просто следуйте инструкциям для установки плагина и не забудьте оставить в способах аутентификации пользователя Weblate.

См.также:

Аутентификация через LDAP, *Аутентификация через CAS*

Обычно установка состоит из добавления способа аутентификации в параметр `AUTHENTICATION_BACKENDS` и установки приложения аутентификации (если оно есть) в параметр `INSTALLED_APPS`:

```
AUTHENTICATION_BACKENDS = (
    # Add authentication backend here
    'weblate.accounts.auth.WeblateUserBackend',
)

INSTALLED_APPS = (
    ...
    'weblate',
    # Install authentication app here
)
```

2.6 Управление доступом

Изменено в версии 3.0: До версии 3.0 система привилегий Weblate была основана на системе привилегий Django, однако теперь она построена специально для Weblate. Если вы используете более старую версию, пожалуйста, ознакомьтесь с документацией к этой старой версии, информация, данная здесь, не будет к ней применима.

Weblate поставляется с тонко настраиваемой системой привилегий для назначения пользователям прав как в рамках всего экземпляра приложения, так и в рамках отдельных его частей.

Система разрешений построена на основе групп и ролей, в которой роли определяют набор разрешений, а группы назначают их пользователям и переводчикам, для получения более подробной информации смотрите раздел *Пользователи, роли, группы и разрешения*.

После установки создаётся стандартный набор групп; вы можете использовать их, чтобы назначить пользователям роли в рамках всего экземпляра приложения (смотрите раздел *Группы и роли по умолчанию*). Кроме того, если включён *По-проектный контроль доступа*, вы можете назначать пользователей конкретным проектам перевода. Более мелкозернистую конфигурацию поможет обеспечить *Настраиваемый контроль доступа*.

2.6.1 Распространённые настройки

Блокировка Weblate

Для полной блокировки вашей установки Weblate вы можете использовать настройку `REQUIRE_LOGIN`, чтобы заставить пользователей входить в систему и настройку `REGISTRATION_OPEN` для предотвращения новых регистраций.

Разрешения для всего сайта

Для управления разрешениями для всего экземпляра просто добавьте пользователей в группы *Пользователи* (это уже делается по умолчанию механизмом *автоматического назначения групп*), *Рецензенты* и *Управляющие*. Оставляйте все проекты в статусе *Публичный* (смотрите раздел *По-проектный контроль доступа*).

По-проектные разрешения

Переведите ваши проекты в статусы *Защищенный* или *Частный* и управляйте пользователями отдельно для каждого проекта через интерфейс Weblate.

Добавление разрешений к языкам, компонентам или проектам

Дополнительно вы можете предоставить разрешения любому пользователю на основе проекта, компонента или набора языков. Для этого создайте новую группу (например, *Переводчики на чешский*) и сконфигурируйте её для данного ресурса. Членам этой группы будут предоставлены любые назначенные разрешения для выбранных ресурсов.

Это будет отлично работать без дополнительной настройки, если использовать по-проектные разрешения. Для разрешений на весь экземпляр вы, вероятно, захотите удалить эти разрешения из группы *Пользователи* или изменить автоматическое назначение всем пользователям этой группы (смотрите раздел *Автоматические прикрепления группы*).

См.также:

Проверка разрешений

2.6.2 По-проектный контроль доступа

Примечание: При включении списков контроля доступа всем пользователям запрещается доступ к чему бы то ни было в пределах данного проекта, если только вы не добавите для них соответствующие разрешения.

Вы можете ограничить доступ пользователей к отдельным проектам. Эта функция включается в блоке *Управление доступом* конфигурации каждого соответствующего проекта. Она автоматически создаёт для этого проекта несколько групп, смотрите раздел *Предопределённые группы*.

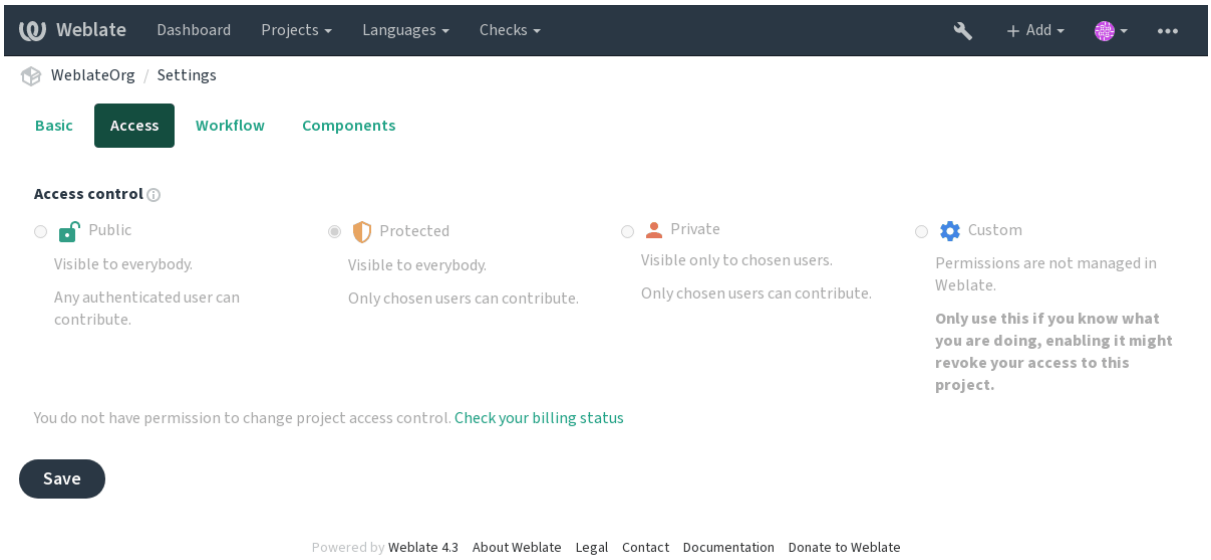
Для настройки :guilabel: *Управление доступом* возможны следующие варианты выбора:

Публичный Проект видят и могут переводить все

Защищённый Проект видят все, но переводить могут только выбранные пользователи

Частный Видеть и переводить проект могут только выбранные пользователи

Настраиваемый Weblate не занимается управлением пользователями, смотрите раздел *Настраиваемый контроль доступа*.



Чтобы разрешить доступ к этому проекту, вы должны добавить привилегию либо непосредственно данному пользователю, либо группе пользователей в административном интерфейсе Django, либо с помощью системы управления пользователями на странице проекта, как описано в разделе *По-проектная настройка доступа*.

Примечание: Даже при включённом списке контроля доступа о вашем проекте будет доступна следующая краткая информация:

- Статистика для всего экземпляра, в том числе по каждому проекту в отдельности.
- Сводка по языкам для всего экземпляра, в том числе по каждому проекту в отдельности.

2.6.3 Автоматические прикрепления группы

Вы можете настроить Weblate для автоматического добавления пользователей в группы на основе их адреса электронной почты. Это автоматическое назначение происходит только в момент создания учётной записи.

Оно может быть настроено в административном интерфейсе Django отдельно для каждой группы (в разделе *Аутентификация*).

Примечание: Weblate при миграциях всегда настраивает автоматическое назначение групп *Пользователи* и *Просматривающие*. В случае, если вы хотите его отключить, просто задайте для них регулярное выражение `^$`, которое никогда ничего не найдёт.

2.6.4 Пользователи, роли, группы и разрешения

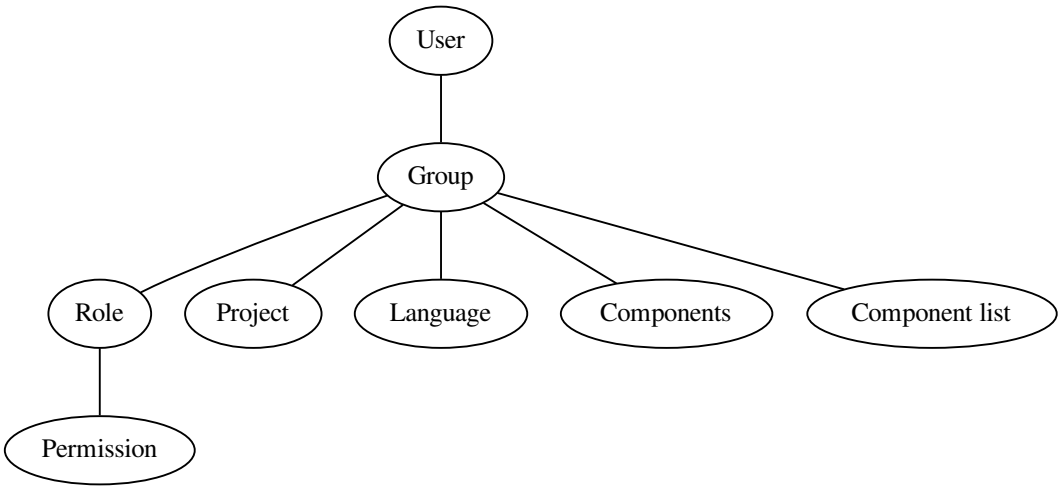
Модели аутентификации состоят из нескольких объектов:

Разрешение Индивидуальные разрешения, определённые Weblate. Вы не можете назначать отдельные разрешения, назначить их можно только через назначение ролей.

Роль Роль определяет набор разрешений. Она позволяет повторно использовать эти наборы в нескольких местах, и упрощает их администрирование.

Пользователь Пользователи могут состоять в нескольких группах.

Группа Группы соединяют роли, пользователей и объекты аутентификации (проекты, языки и списки компонентов).



Проверка разрешений

Всякий раз, когда происходит проверка разрешения для выяснения того, может ли пользователь выполнить данное действие, она производится в соответствующей области, и проверяет следующие вещи в указанном порядке:

1. С компонентом или проектом сравнивается *Список компонентов* в поиске соответствия.
2. С компонентом или проектом сравниваются *Компоненты* в поиске соответствия.
3. С проектом сравниваются *Проекты* в поиске соответствия.

Как вы можете видеть, предоставление пользователю доступа к компоненту автоматически предоставляет ему доступ и к содержащему компонент проекту.

Примечание: Используется только первое сработавшее правило. Таким образом, если задать *Список компонентов*, *Компоненты* и *Проект*, то применится только *Список компонентов*.

Если производится проверка разрешения на перевод, выполняется дополнительный шаг:

4. *Languages* are matched against the scope of translations if set, if not set, this does not match any language.

Подсказка: Для автоматического включения всех языков или проектов вы можете использовать переключатели *Выбор языка* или *Выбор проекта*.

Проверка доступа к проекту

Пользователь должен быть членом группы, связанной с проектом или с каким-либо компонентом внутри проекта. Чтобы получить доступ к проекту, никаких специальных разрешений не требуется, достаточно только членства в группе (это используется в умолчательной группе *Просматривающие*, смотрите раздел *Группы и роли по умолчанию*).

Проверка доступа к компоненту

Пользователь может получить доступ к неограниченному компоненту, как только получит доступ к содержащему компонент проекту. Если включён *Ограниченный доступ*, для доступа к компоненту потребуется явное разрешение на компонент (или на содержащий его список компонентов).

2.6.5 Управление пользователями и группами

Пользователями и группами можно управлять через административный интерфейс Django, доступный по адресу `/admin/`.

По-проектная настройка доступа

Примечание: Эта возможность работает только для проектов под управлением списка контроля доступа, смотрите раздел *По-проектный контроль доступа*.

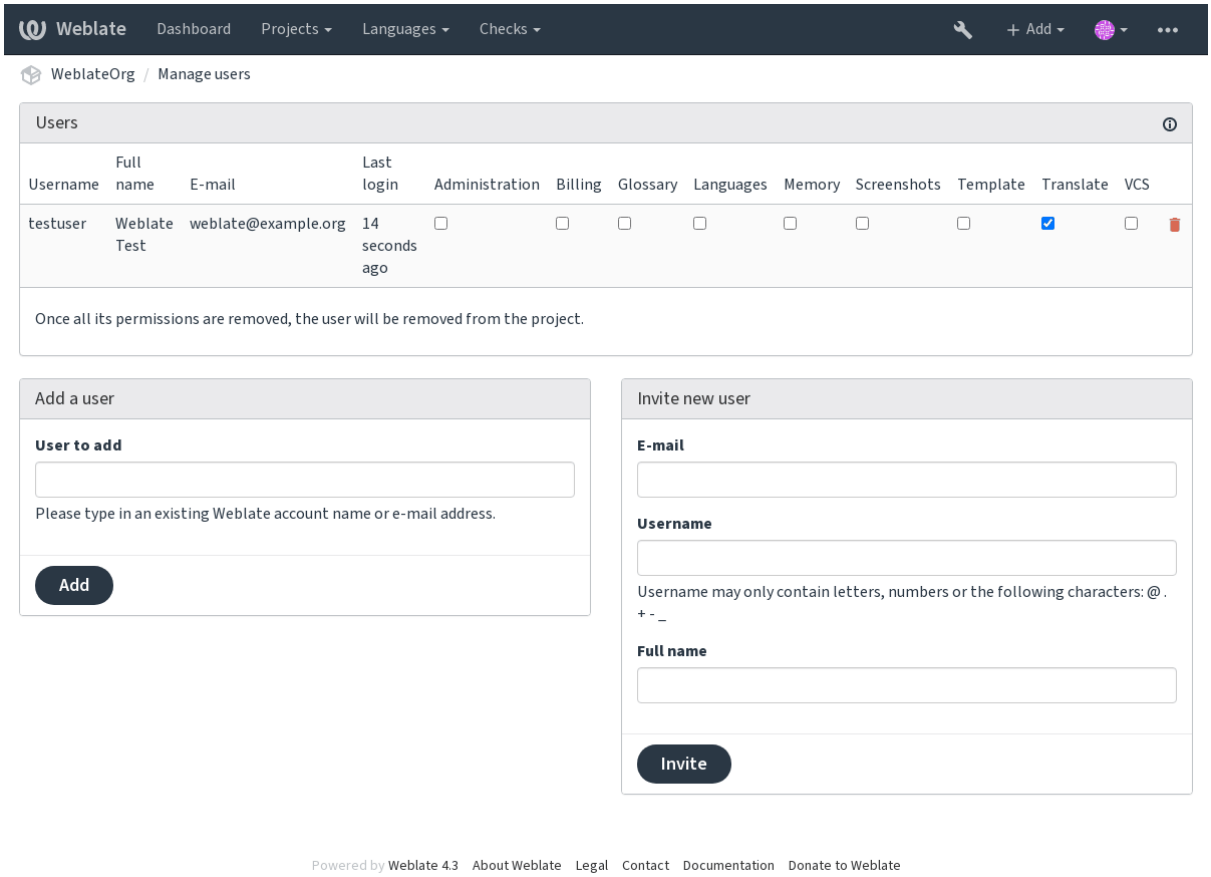
Пользователи с привилегией *Может управлять списками контроля доступа для проекта* (смотрите раздел *Управление доступом*) также могут управлять пользователями в проектах с включённым на странице проекта контролем доступа. Интерфейс позволяет вам:

- Добавлять существующих пользователей в проект
- Приглашать новых пользователей в проект
- Изменять разрешения пользователей
- Отзывать доступы пользователей

Добавлено в версии 3.11.

- Повторно отправлять приглашения по электронной почте, аннулируя все ранее отправленные приглашения

Управление пользователями доступно через меню *Управлять* проекта:



См.также:

По-проектный контроль доступа

Предопределённые группы

Weblate поставляется с предопределённым набором групп для проекта, которые вы можете назначить пользователям.

`Administration`
Даёт все разрешения, доступные в проекте.

`Glossary`
Может управлять глоссарием (добавлять, удалять или загружать записи в глоссарий).

`Languages`
Может управлять языками перевода — добавлять или удалять переводы.

`Screenshots`
Может управлять снимками экрана — добавлять или удалять их, а также связывать их с исходными строками.

`Template`
Может редактировать шаблоны перевода в *одноязычных компонентах* и информацию об исходной строке.

`Translate`
Может переводить проект и загружать переводы, сделанные в автономном режиме.

`VCS`
Может управлять системой контроля версий и имеет доступ к экспортированному репозиторию.

Review
Может одобрять переводы во время рецензирования.

Billing
Имеет доступ к платёжной информации (смотрите раздел *Тарифный план*).

2.6.6 Настраиваемый контроль доступа

При выборе в графе *Управление доступом* пункта *Настраиваемый*, Weblate прекратит управление доступом для данного проекта и всеми пользователями и группами можно будет управлять с помощью административного интерфейса Django. Этот пункт можно использовать для определения более сложного контроля доступа, либо для настройки политики общего доступа для всех проектов на одном экземпляре Weblate. Если вы хотите включить его для всех проектов по умолчанию, пожалуйста, задайте параметр `DEFAULT_ACCESS_CONTROL`.

> **Предупреждение:** При включении этого режима Weblate удалит все *по-проектные списки контроля доступа*, которые он создал для этого проекта. Если вы включаете его, не имея прав администратора, полученных от экземпляра приложения, вы мгновенно потеряете доступ к управлению проектом.

2.6.7 Группы и роли по умолчанию

Список привилегий

Тарифный план (смотрите раздел *Тарифный план*) Просмотр платёжной информации [*Администрирование*, *Биллинг*]

Изменения Скачивание изменений [*Администрирование*]

Комментарии Публикация комментариев [*Администрирование*, *Правка источника*, *Опытный пользователь*, *Рецензирование строк*, *Перевод*]

Удаление комментариев [*Администрирование*]

Компонент Правка параметров компонентов [*Администрирование*]

Блокировка компонента от перевода [*Администрирование*]

Глоссарий Добавление записей в глоссарий [*Администрирование*, *Управление глоссарием*, *Опытный пользователь*]

Правка записей глоссария [*Администрирование*, *Управление глоссарием*, *Опытный пользователь*]

Удаление записей из глоссария [*Администрирование*, *Управление глоссарием*, *Опытный пользователь*]

Загрузка записей в глоссарий [*Администрирование*, *Управление глоссарием*, *Опытный пользователь*]

Автоматические предложения Использование автоматических предложений [*Администрирование*, *Опытный пользователь*]

Проекты Правка параметров проекта [*Администрирование*]

Управление доступом к проекту [*Администрирование*]

Отчёты Скачивание отчётов [*Администрирование*]

Снимки экрана Добавление снимков экрана [*Администрирование*, *Управление снимками экрана*]

Правка снимков экрана [*Администрирование*, *Управление снимками экрана*]

Удаление снимков экрана [*Администрирование*, *Управление снимками экрана*]

Исходные строки Правка исходных строк [*Администрирование*, *Правка источника*]

Строки Добавление новых строк [*Администрирование*]

Игнорирование неудачных проверок [*Администрирование*, *Правка источника*, *Опытный пользователь*, *Рецензирование строк*, *Перевод*]

Правка строк [*Администрирование*, *Правка источника*, *Опытный пользователь*, *Рецензирование строк*, *Перевод*]

Рецензирование строк [*Администрирование*, *Рецензирование строк*]

Правка строк, когда предложения являются обязательными [*Администрирование*, *Рецензирование строк*]

Правка исходных строк [*Администрирование*, *Правка источника*, *Опытный пользователь*]

Предложения Принятие предложения [*Администрирование*, *Правка источника*, *Опытный пользователь*, *Рецензирование строк*, *Перевод*]

Добавление предложений [*Добавление предложений*, *Администрирование*, *Правка источника*, *Опытный пользователь*, *Рецензирование строк*, *Перевод*]

Удаление предложений [*Администрирование*]

Голосование по предложениям [*Администрирование*, *Правка источника*, *Опытный пользователь*, *Рецензирование строк*, *Перевод*]

Переводы Начало нового перевода [*Администрирование*, *Управление языками*, *Опытный пользователь*]

Производство автоматического перевода [*Администрирование*, *Управление языками*]

Удаление существующих переводов [*Администрирование*, *Управление языками*]

Начало перевода на новый язык [*Администрирование*, *Управление языками*]

Загрузки Задание автора загружаемого перевода [*Администрирование*]

Перезапись существующих строк при загрузке новых [*Администрирование*, *Правка источника*, *Опытный пользователь*, *Рецензирование строк*, *Перевод*]

Загрузка переведённых строк [*Администрирование*, *Правка источника*, *Опытный пользователь*, *Рецензирование строк*, *Перевод*]

Система контроля версий (VCS) Доступ к внутреннему репозиторию [*Доступ к репозиторию*, *Администрирование*, *Управление репозиторием*, *Опытный пользователь*]

Коммит изменений во внутренний репозиторий [*Администрирование*, *Управление репозиторием*]

Отправка изменений из внутреннего репозитория [*Администрирование*, *Управление репозиторием*]

Сброс изменений во внутреннем репозитории [*Администрирование*, *Управление репозиторием*]

Просмотр расположения вышестоящего репозитория [*Доступ к репозиторию*, *Администрирование*, *Управление репозиторием*, *Опытный пользователь*]

Обновление внутреннего репозитория [*Администрирование*, *Управление репозиторием*]

Привилегии для всего сайта Использование интерфейса управления

Добавление новых проектов

Добавление определений языка

Управление определениями языка

Управление группами

Управление пользователями

Управление ролями

Управление объявлениями

Управление памятью переводов

Управление списками компонентов

Примечание: Привилегии для всего сайта не предоставляются никаким умолчательным ролям. Они мощны и достаточно близки к статусу суперпользователя — большинство из них влияют на все проекты вашей установки Weblate.

Список групп

Следующие группы создаются после установки (или после выполнения команды `setupgroups`):

Гости Определяет разрешения для неавторизованных пользователей.

Эта группа содержит только анонимных пользователей (смотрите настройку `ANONYMOUS_USER_NAME`).

Вы можете удалить роли из этой группы, чтобы ограничить разрешения для неавторизованных пользователей.

Роли по умолчанию: *Добавление предложений*, *Доступ к репозиторию*

Просматривающие Эта роль обеспечивает видимость публичных проектов для всех пользователей. По умолчанию членами этой группы являются все пользователи.

По умолчанию все пользователи являются членами этой группы, используя *Автоматические прикрепления группы*.

Роли по умолчанию: нет

Пользователи Стандартная группа для всех пользователей.

По умолчанию все пользователи являются членами этой группы, используя *Автоматические прикрепления группы*.

Роли по умолчанию: *Опытный пользователь*

Рецензенты Группа для рецензентов (смотрите раздел *Рабочие процессы перевода*).

Роли по умолчанию: *Рецензирование строк*

Управляющие Группа для администраторов.

Роли по умолчанию: *Администрирование*

Предупреждение: Никогда не удаляйте предопределённые в Weblate группы и пользователей, поскольку это может привести к неожиданным проблемам. Если вы не хотите использовать эти функции, просто удалите из них все привилегии.

2.7 Проекты перевода

2.7.1 Организация переводов

Weblate организует переводимое содержимое проектов/компонентов из системы контроля версий в древовидную структуру.

- Объектом нижнего уровня является *проект*, который должен содержать все связанные переводы (например, перевод приложения различных версий и/или сопроводительную документацию).
- На уровне выше располагается *компонент*, который фактически и является компонентом для перевода, для которого вы определяете используемый репозиторий системы контроля версий и маску файлов для перевода.

- Ещё выше *компонента* находятся отдельные переводы, которые Weblate обрабатывает автоматически, как только файлы перевода (соответствующие маске файлов, определённой в *конфигурации компонента*), появляются в репозитории системы контроля версий.

Weblate поддерживает широкий набор форматов перевода (как двуязычных, так и одноязычных), поддерживаемых набором инструментов Translate Toolkit, смотрите раздел *Поддерживаемые форматы файлов*.

Примечание: Вы можете совместно использовать склонированные репозитории системы контроля версий, используя *внутренние URL-адреса Weblate*. Использование этой возможности настоятельно рекомендуется при наличии множества компонентов, совместно использующих один и тот же репозиторий. Она повышает производительность и уменьшает необходимое дисковое пространство.

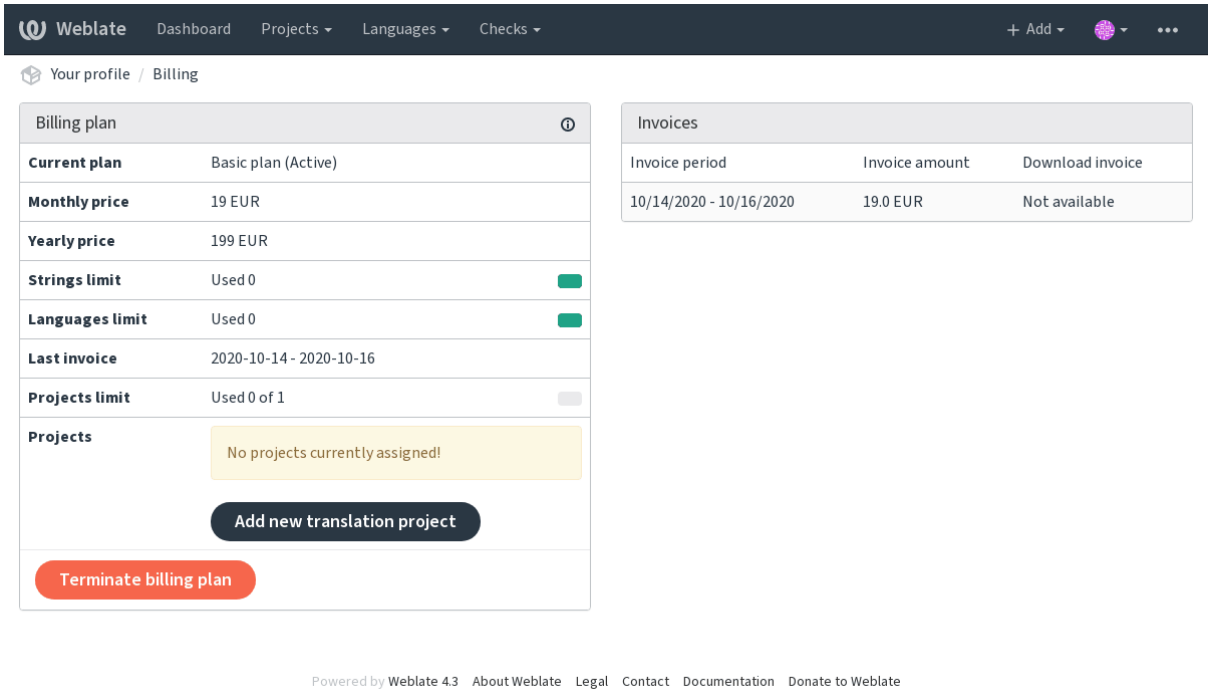
2.7.2 Добавление проектов и компонентов перевода

Изменено в версии 3.2: В Weblate'е появился интерфейс для добавления проектов и компонентов, так что вам больше не нужно использовать для этого *административный интерфейс Django*.

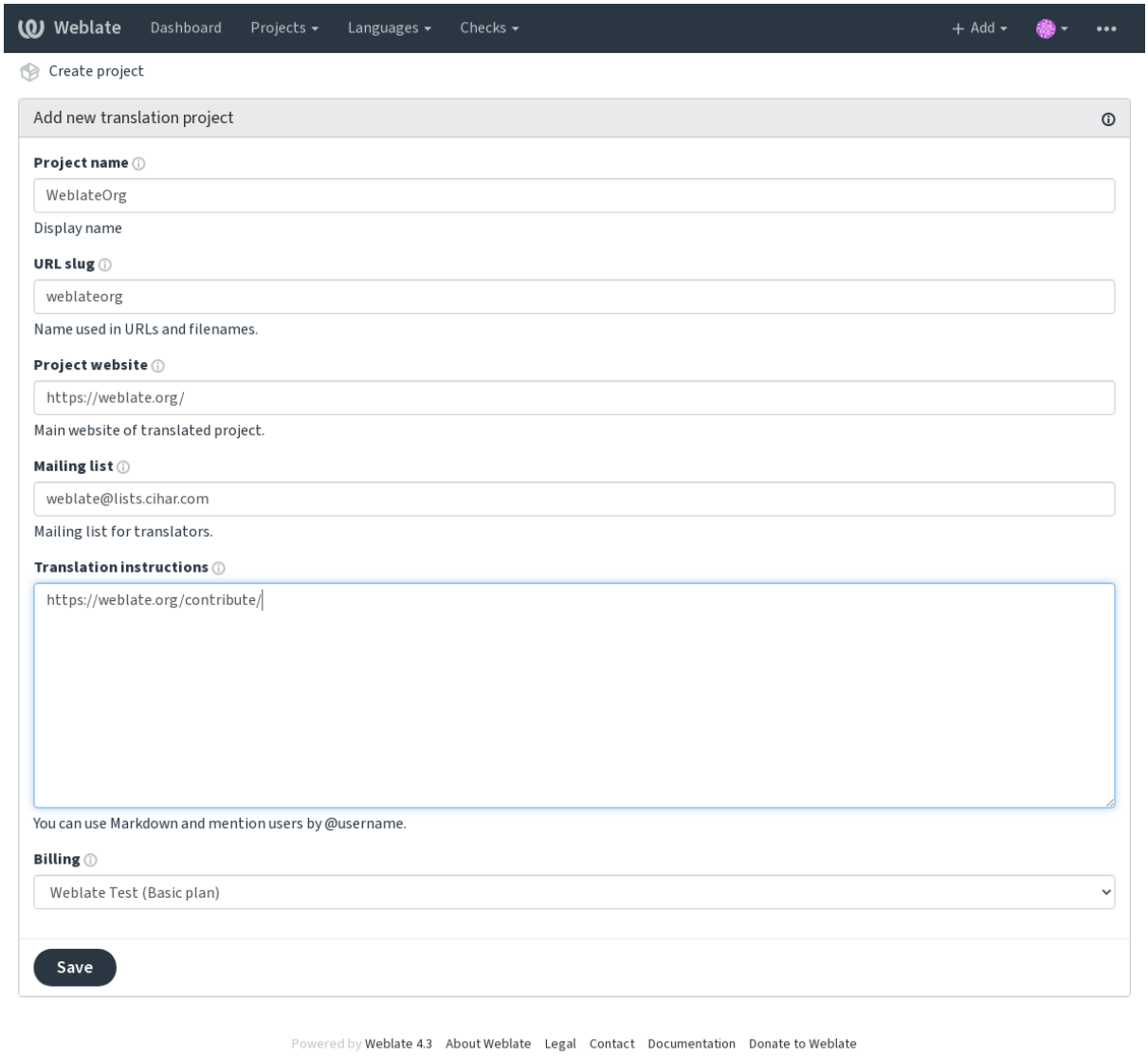
Изменено в версии 3.4: Теперь процесс добавления компонентов состоит из нескольких шагов, с автоматическим обнаружением большинства параметров.

В зависимости от ваших разрешений вы можете создавать новые проекты и компоненты перевода. Для пользователей с разрешением *Добавление новых проектов* их создание всегда разрешено, а если ваш экземпляр использует тарифные планы (как, например, https://hosted.weblate.org/, смотрите раздел *Тарифный план*), вы также можете создавать их на основе разрешений из вашего плана, установленных учётной записью пользователя, управляющего тарификацией.

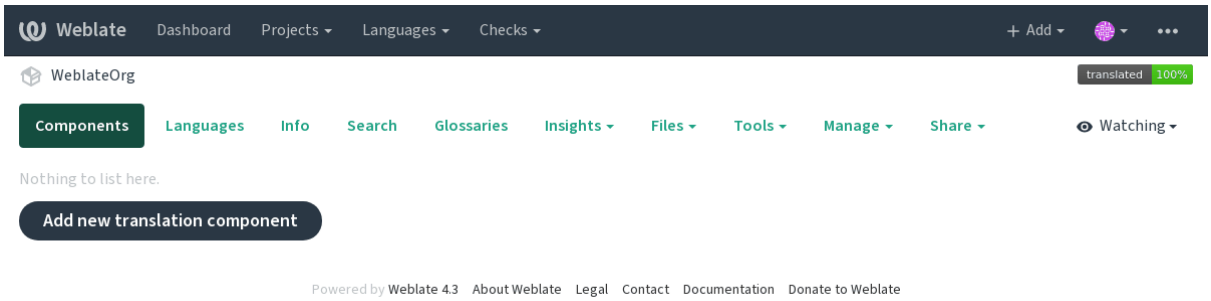
Вы можете просмотреть свой текущий тарифный план на отдельной странице:



Создание проекта можно инициировать оттуда или из меню на навигационной панели; чтобы завершить добавление проекта перевода, заполните основную информацию о нём:



После создания проекта вы попадаете непосредственно на страницу проекта:



Создание нового компонента перевода можно инициировать одним щелчком мыши. Процесс создания компонента многоступенчатый, большинство параметров перевода он определяет автоматически. Существует несколько подходов к созданию компонента:

Из системы контроля версий Создать компонент из удалённого репозитория системы контроля версий.

Из уже существующего компонента Создать дополнительный компонент на основе уже существующего, только с другими файлами.

Дополнительная ветка Создать дополнительный компонент на основе уже существующего, только для другой ветки.

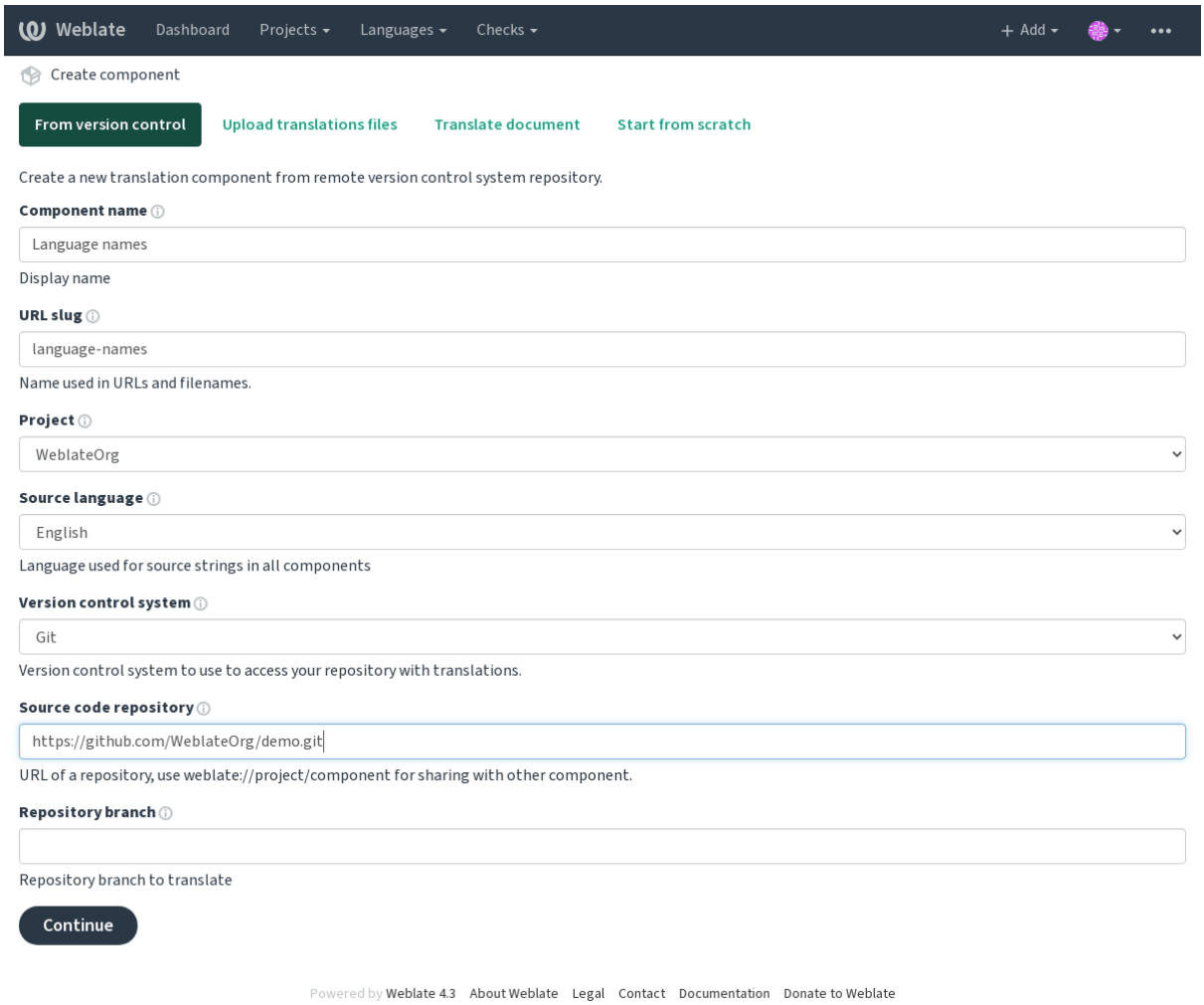
Загрузить файлы перевода Загрузить файлы перевода в Weblate в случае, если у вас нет системы контроля версий или вы не хотите интегрировать её с Weblate. Позже вы сможете обновить содержимое через веб-интерфейс или API.

Перевести документ Загрузить одиночный документ и перевести его.

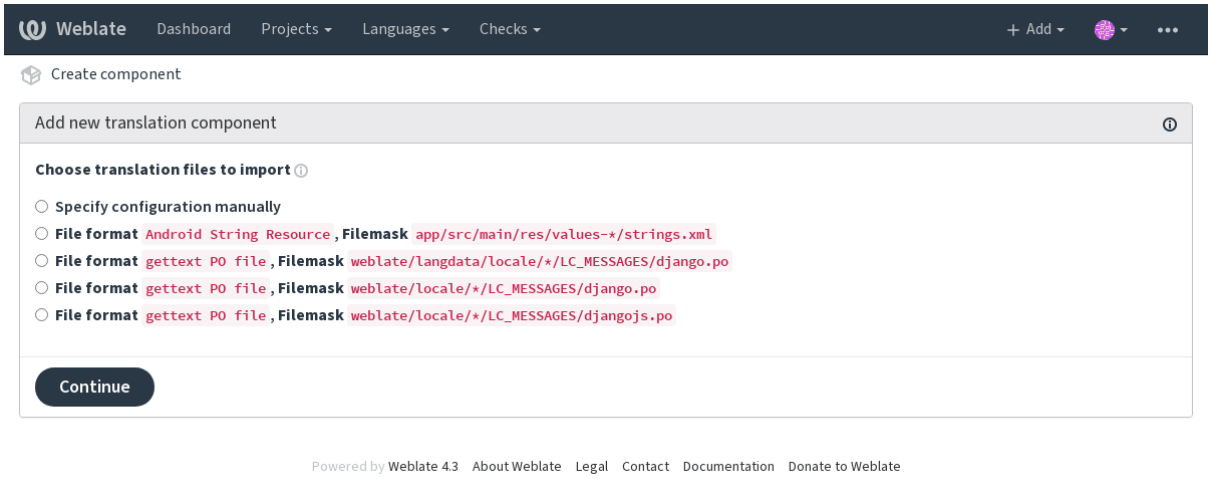
Начать с нуля Создать пустой проект перевода и добавить строки вручную.

Имея существующие компоненты перевода, вы также можете легко добавлять новые для дополнительных файлов или веток из того же репозитория.

Сначала вам нужно указать наименование и расположение репозитория:



На следующей странице вам будет представлен список обнаруженных переводимых ресурсов:



На последнем этапе просмотрите информацию о компоненте перевода и укажите дополнительные сведения:

Weblate Dashboard Projects Languages Checks + Add

Create component

Detected license as MIT, please check whether it is correct.

Add new translation component

Project

WeblateOrg

Component name

Language names

Display name

URL slug

language-names

Name used in URLs and filenames.

Version control system

Git

Version control system to use to access your repository containing translations. You can also choose additional integration with third party providers to submit merge requests.

Source code repository

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

URL of a repository, use weblate://project/component to share it with other component.

Repository branch

Repository branch to translate

Repository push URL

URL of a push repository, pushing is turned off if empty.

Push branch

Branch for pushing changes, leave empty to use repository branch

Repository browser

https://github.com/WeblateOrg/demo/blob/{{branch}}/{{filename}}#L{{line}}

Link to repository browser, use {{branch}} for branch, {{filename}} and {{line}} as filename and line placeholders.

File format

gettext PO file

Filemask

weblate/langdata/locale/*/LC_MESSAGES/django.po

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

Monolingual base language file

Filename of translation base file, containing all strings and their source; it is recommended for monolingual translation formats.

☑ **Edit base file**

Whether users will be able to edit the base file for monolingual translations.

Intermediate language file

Filename of intermediate translation file. In most cases this is a translation file provided by developers and is used when creating actual source strings.

Template for new translations

weblate/langdata/locale/django.pot

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

Translation license

GNU General Public License v3.0 or later

Adding new translation

Create new language file

How to handle requests for creating new translations.

Language code style

Default based on the file format

Customize language code used to generate the filename for translations created by Weblate.

Language filter

^(cs|he|hu)$

Regular expression used to filter translation when scanning for filemask.

Source language

English

Language used for source strings in all components

You will be able to edit more options in the component settings after creating it.

Save

Powered by Weblate 4.3 About Weblate Legal Contact Documentation Donate to Weblate

См.также:

Интерфейс администратора Django, *Конфигурация проекта*, *Конфигурация компонента*

2.7.3 Конфигурация проекта

Создайте проект перевода, а затем добавьте в него новый компонент для перевода. Проект похож на полку, на которую укладываются реальные переводы. Все компоненты одного проекта разделяют между собой общие предложения и глоссарий; переводы также автоматически распространяются во все компоненты одного проекта (если только оно не отключено в конфигурации компонентов), смотрите раздел Memory Management.

Это основные атрибуты для настройки, они предоставляют переводчикам информацию о проекте:

Название проекта

Подробное наименование проекта, используемое для его отображения.

Плашка проекта

Название проекта, подходящее для указания его в URL-адресе.

Веб-сайт проекта

URL-адрес, по которому переводчики могут найти дополнительную информацию о проекте.

Список рассылки

Список рассылки, в котором переводчики могут обсуждать или комментировать переводы.

Инструкции для переводчиков

URL-адрес на сайт, содержащий более подробные инструкции для переводчиков.

Задать заголовок «Language-Team»

Должен ли Weblate управлять заголовком `Language-Team` (сейчас это единственная особая возможность для *GNU gettext*).

Использовать совместную память переводов

Использовать ли совместно используемую память переводов, подробнее смотрите в разделе *Совместная память переводов*.

Вносить вклад в совместную память переводов

Помещать ли переводы в совместно используемую память переводов, подробнее смотрите в разделе *Совместная память переводов*.

Управление доступом

Настройка контроля доступа для каждого проекта, подробнее смотрите в разделе *По-проектный контроль доступа*.

Значение по умолчанию можно изменить установкой параметра `DEFAULT_ACCESS_CONTROL`.

Включить рецензии

Включение процесса рецензирования переводов, смотрите раздел *Выделенные рецензенты*.

Включить рецензии исходных строк

Включение процесса рецензирования исходных строк, смотрите раздел *Рецензирование исходных строк*.

Включить обработчики

Должны ли для этого репозитория использоваться неаутентифицированные *обработчики уведомлений*.

См.также:

Файл промежуточного языка, *Шлюз качества для исходных строк*, *Двуязычные и одноязычные форматы*, *Определения языка*

Псевдонимы языка

Определяет отображение кодов языков при импорте переводов в Weblate. Используйте этот параметр, когда коды языков в ваших репозиториях не согласованы друг с другом, а вы хотите получить в Weblate согласованное представление.

Типичным примером использования может быть отображение американского английского языка на английский: `en_US:en`

Несколько отображений должны разделяться запятой: `en_GB:en,en_US:en`

Подсказка: Коды языков отображаются при успешном сопоставлении с именами файлов перевода, а оно чувствительно к регистру, поэтому убедитесь, что вы используете коды исходного языка в той же форме, в которой они используются в именах файлов.

См.также:

Разбор кодов языков

2.7.4 Конфигурация компонента

Компонент — это группировка каких-то сущностей для перевода. Вы вводите расположение репозитория системы контроля версий и маску файлов, определяющую, какие файлы вы хотите переводить, а Weblate автоматически извлекает код из этого репозитория и находит все соответствующие файлы для перевода.

Примеры типичных конфигураций вы можете найти в разделе *Поддерживаемые форматы файлов*.

Примечание: Рекомендуется держать размер компонентов перевода в разумных пределах — разбивайте перевод по любому критерию, который в вашем случае имеет смысл (по отдельным приложениям или дополнениям, по главам книг или по веб-сайтам).

Weblate легко справляется с переводами с 10000 строк, но с такими объёмами труднее разделять работу между несколькими переводчиками и координировать их действия.

Если определение языка для перевода отсутствует, создаётся пустое определение с именем вида «ru_RU (generated)». Вы должны скорректировать определение и сообщить об этом авторам Weblate, чтобы недостающие языки могли быть включены в следующий выпуск.

Компонент содержит все важные параметры для работы с системой контроля версий, а также для получения из неё переводов:

Название компонента

Подробное наименование компонента, используемое для его отображения.

Плашка компонента

Название компонента, подходящее для указания его в URL-адресе.

Проект компонента

Конфигурация проекта, которому принадлежит компонент.

Система контроля версий

Используемая система контроля версий, подробнее смотрите в разделе *Интеграция с системой контроля версий*.

Репозиторий исходного кода

Репозиторий системы контроля версий, из которого извлекаются изменения.

См.также:

Для получения более подробной информации о том, как указывать URL-адреса, смотрите раздел *Доступ к репозиториям*.

Подсказка: Это может быть либо реальный URL-адрес системы контроля версий, либо адрес вида `weblate://проект/компонент`, указывающий на то, что репозиторий должен совместно использоваться с другим компонентом. Для получения более подробной информации обратитесь к разделу *Внутренние URL-адреса Weblate*.

URL для отправки в репозиторий

URL-адрес репозитория, используемый для отправки изменений. Этот параметр используется только для систем контроля версий *Git* и *Mercurial* и если он пуст, поддержка отправки в них отключается.

См.также:

Для получения более подробной информации о том, как указывать URL-адрес репозитория, смотрите раздел *Доступ к репозиториям*, а для получения более подробной информации об отправке изменений из Weblate'а обратитесь к разделу *Отправка изменений из Weblate'а*.

Просмотрщик репозитория

URL-адрес браузера репозитория, используемый для отображения исходных файлов (местоположения используемых сообщений). Если пустой, такие ссылки генерироваться не будут. Вы можете использовать *шаблонную разметку*.

Например, для ссылок на GitHub, используйте что-то наподобие: `https://github.com/WeblateOrg/hello/blob/{{branch}}/{{filename}}#L{{line}}`

В случае, если ваши пути указываются относительно другого каталога, вы можете удалить ведущий каталог фильтром `parentdir` (смотрите раздел *Шаблонная разметка*): `https://github.com/WeblateOrg/hello/blob/{{branch}}/{{filename|parentdir}}#L{{line}}`

URL экспортированного репозитория

URL-адрес, куда экспортируются изменения, сделанные в Weblate. Он имеет значение, если не используется *непрерывная локализация* или в том случае, когда есть необходимость ручного слияния изменений. Для автоматизации этого процесса для Git-репозиториев вы можете использовать *экспортер Git*.

Ветка репозитория

Какую ветку извлекать из системы контроля версий и где искать переводы.

Ветка для отправки

Ветка, в которую отправлять изменения, оставьте пустой для использования параметра *Ветка репозитория*.

Примечание: This is currently only supported for Git, GitLab and GitHub, it is ignored for other VCS integrations.

Маска файла

Маска файлов для перевода, включая путь. Она должна включать в себя один символ `*`, который заменяется на код языка (для получения информации о том, как он обрабатывается, смотрите раздел *Определения языка*). В случае, если в вашем репозитории содержится более одного файла перевода (например, несколько доменов gettext), вам необходимо создать компонент для каждого из них.

Например, `po/*.po` или `locale/*/LC_MESSAGES/django.po`.

В случае, если имя вашего файла содержит такие специальные символы, как `[` или `]`, они должны быть экранированы как `[[]` и `[]]`.

См.также:

Двуязычные и одноязычные форматы, *Что означает сообщение «Есть больше файлов для одного языка (en)»?*

Одноязычный базовый файл

Базовый файл, содержащий определения строк для *одноязычных компонентов*.

См.также:

Двуязычные и одноязычные форматы, *Что означает сообщение «Есть больше файлов для одного языка (en)»?*

Правка базового файла

Разрешать ли редактирование базового файла для *одноязычных компонентов*.

Файл промежуточного языка

Файл промежуточного языка для *одноязычных компонентов*. В большинстве случаев это файл перевода, предоставляемый разработчиками и используемый при создании реальных исходных строк.

Если задан, исходный перевод основывается на этом файле, тогда как все остальные — на *одноязычном базовом файле*. В случае, если строка в исходном переводе не переведена, перевод на другие языки запрещается. Этот запрет обеспечивает *шлюз качества для исходных строк*.

См.также:

Шлюз качества для исходных строк, *Двуязычные и одноязычные форматы*, *Что означает сообщение «Есть больше файлов для одного языка (en)»?*

Шаблон для новых переводов

Базовый файл, используемый для генерирования новых переводов, например, `.pot` файл для gettext.

Подсказка: Для многих одноязычных форматов Weblate по умолчанию начинает перевод с пустого файла. Используйте этот параметр, если вы хотите, чтобы при создании нового перевода присутствовали все строки с пустым значением.

См.также:

Добавление новых переводов, *Добавление нового перевода*, *Двуязычные и одноязычные форматы*, *Что означает сообщение «Есть больше файлов для одного языка (en)»?*

Формат файла

Формат файла перевода, смотрите также *поддерживаемые форматы файлов*.

Адрес для сообщения об ошибке в исходной строке

Адрес электронной почты, используемый для отправки сообщений об ошибках в исходных строках. На этот адрес также будут отсылаться уведомления о любых сделанных в Weblate комментариях к исходной строке.

Разрешить распространение перевода

Вы можете отключить распространение переводов на этот компонент из других компонентов этого же проекта. Это сильно зависит от того, что вы переводите, иногда желательно использовать разные переводы для одинаковых строк.

Обычно для одноязычных переводов рекомендуется отключать этот параметр, если только вы не используете во всем проекте одни и те же идентификаторы для одинаковых строк.

Значение по умолчанию может быть изменено параметром `DEFAULT_TRANSLATION_PROPAGATION`.

Включить предложения

Принимать ли предложения по переводу этого компонента.

Голосование по предложению

Включает голосование за предложения, смотрите раздел *Голосование по предложению*.

Автоматическое принятие предложений

Автоматически принимать предложения, за которые проголосовали, смотрите раздел *Голосование по предложению*.

Флаги перевода

Настройка проверок качества и другого поведения Weblate'a, смотрите раздел *Настройка поведения*.

Принудительные проверки

Список проверок, которые нельзя проигнорировать, смотрите раздел *Принудительные проверки*.

Лицензия перевода

Лицензия на перевод (не обязательно должна совпадать с лицензией на исходный код).

Соглашение участника

Пользовательское соглашение, которое должно быть одобрено, прежде чем пользователь сможет переводить этот компонент.

Добавление нового перевода

Как обрабатывать запросы на создание новых языков. Доступные варианты:

Связаться с сопровождающими Пользователь может выбрать желаемый язык, а сопровождающие проекта получат об этом уведомление. Добавление (или не добавление) языка в репозиторий зависит от них.

Указать URL на инструкции по переводу Пользователю предоставляется ссылка на страницу, которая описывает процесс начала новых переводов. Используйте этот вариант в случае, если вам требуется более формальный процесс (например, формирование команды людей перед началом фактического перевода).

Создать новый язык Пользователь может выбрать язык и Weblate автоматически создаст для него файл, так что можно сразу начинать переводить.

Запретить добавление новых переводов У пользователя не будет возможности начать новый перевод.

См.также:

Добавление новых переводов.

Стиль кода языка

Настраивает код языка, используемый для генерации имени файла для переводов, созданных Weblate, за более подробной информацией обратитесь к разделу *Добавление новых переводов*.

Стиль слияния

Вы можете настроить обработку обновлений из вышестоящего репозитория. Для некоторых систем контроля версий этот параметр может не поддерживаться. За более подробной информацией обратитесь к разделу *Слияние или перебазирование*.

Значение по умолчанию можно изменить установкой параметра `DEFAULT_MERGE_STYLE`.

Сообщения коммита, добавления, удаления, слияния и сообщения надстроек

Сообщение, используемое при коммите перевода, смотрите раздел *Шаблонная разметка*.

Значения по умолчанию можно изменить установкой параметров `DEFAULT_ADD_MESSAGE`, `DEFAULT_ADDON_MESSAGE`, `DEFAULT_COMMIT_MESSAGE`, `DEFAULT_DELETE_MESSAGE` и `DEFAULT_MERGE_MESSAGE`.

Имя произведшего коммит

Имя коммиттера, используемое для коммитов Weblate'а, автором всегда будет фактический переводчик. Для некоторых систем контроля версий этот параметр может не поддерживаться.

Значение по умолчанию можно изменить установкой параметра `DEFAULT_COMMITER_NAME`.

E-mail делающего коммит

Электронная почта коммиттера, используемая для коммитов Weblate'а, автором всегда будет фактический переводчик. Для некоторых систем контроля версий этот параметр может не поддерживаться. Значение по умолчанию можно изменить установкой параметра `DEFAULT_COMMITER_EMAIL`.

Отправлять при коммите

Должны ли закоммиченные изменения автоматически отправляться в вышестоящий репозиторий. Когда этот параметр включён, отправка инициируется после того, как Weblate закоммитит изменения в свой внутренний репозиторий (смотрите *отложенные коммиты*). Для реального включения функции отправки также должен быть настроен *URL для отправки в репозиторий*.

Возраст изменений для коммита

Устанавливает, насколько старые (в часах) изменения должны коммититься фоновой задачей или командой управления `commit_pending`. В компоненте коммитятся все изменения, если хотя бы одно из них старше этого периода.

Значение по умолчанию можно изменить установкой параметра `COMMIT_PENDING_HOURS`.

Блокировать при ошибке

Позволяет заблокировать компонент при ошибках в репозитории (неудачные извлечения, отправки или слияния изменений). В такой ситуации блокировка позволяет избежать добавления ещё одного конфликта, который пришлось бы разрешать вручную.

Компонент будет автоматически разблокирован, как только в репозитории не останется ошибок.

Исходный язык

Язык, используемый для исходных строк. Изменяйте этот параметр, если вы переводите не с английского.

Подсказка: В случае, если вы переводите двуязычные файлы с английского языка, но хотите иметь возможность делать исправления также и в английском переводе, вы можете выбрать в качестве исходного языка *Английский (разработчик)*, чтобы избежать конфликта между названием исходного языка и существующим переводом.

Для одноязычных переводов в данном случае можно использовать промежуточный перевод, смотрите раздел *Файл промежуточного языка*.

Языковой фильтр

Регулярное выражение, используемое для фильтрации перевода при поиске файлов по маске. Его можно использовать для ограничения списка управляемых Weblate'ом языков.

Примечание: Вам необходимо перечислить коды языков в том же виде, в котором они встречаются в именах файлов.

Несколько примеров фильтров:

Описание фильтра	Регулярное выражение
Только выбранные языки	`^(cs|de|es)$`
Исключить языки	`^(?!(it|fr)$).+$`
Исключить неязыковые файлы	`^(?!(blank)$).+$`
Включить все файлы (по умолчанию)	`^[^.]+$`

Регулярное выражение вариантов

Регулярное выражение, используемое для определения вариантов строки, смотрите раздел *Варианты строк*.

Примечание: Большинство полей могут быть отредактированы владельцами или управляющими проекта через интерфейс Weblate'а.

См.также:

Поддерживает ли Weblate другие системы контроля версий, кроме Git и Mercurial?, *Предупреждения компонента перевода*

Приоритет

Переводчикам вначале предлагаются компоненты с большим приоритетом.

Ограниченный доступ

По умолчанию компонент виден любому, кто имеет доступ к проекту, даже если этот человек не может выполнить в компоненте никаких изменений. Это облегчает поддержание согласованности перевода в рамках проекта.

Включите этот параметр в том случае, если вы хотите предоставить доступ к этому компоненту явно — права, заданные на уровне проекта к нему применяться не будут, и для предоставления доступа вам придётся явно их указать на уровне компонента или списка компонентов.

Значение по умолчанию можно изменить установкой параметра `DEFAULT_RESTRICTED_COMPONENT`.

Подсказка: Это также относится и к управляющим проектов — пожалуйста, убедитесь, что после переключения статуса вы не потеряете доступ к компоненту.

2.7.5 Шаблонная разметка

В нескольких местах, где необходима отрисовка текста, Weblate использует простой язык разметки. Он основан на языке шаблонов Django, так что он может быть достаточно мощным.

В настоящее время он используется:

- Для форматирования сообщений коммитов, смотрите раздел *Конфигурация компонента*
- **В нескольких надстройках**
  - *Обнаружение компонента*
  - *Генератор статистики*
  - *Выполнение скриптов из надстройки*

В шаблонах компонентов доступны следующие переменные:

`{{ language_code }}` Код языка

`{{ language_name }}` Название языка

`{{ component_name }}` Название компонента

`{{ component_slug }}` Плашка компонента

`{{ project_name }}` Название проекта

`{{ project_slug }}` Плашка проекта

{{ url }} URL-адрес перевода

{{ filename }} Имя файла перевода

{{ stats }} Статистика перевода, у неё есть дополнительные атрибуты, примеры ниже.

{{ stats.all }} Общее количество строк

{{ stats.fuzzy }} Количество строк, требующих рецензирования

{{ stats.fuzzy_percent }} Процент строк, требующих рецензирования

{{ stats.translated }} Количество переведённых строк

{{ stats.translated_percent }} Процент переведённых строк

{{ stats.allchecks }} Количество строк с неудачными проверками

{{ stats.allchecks_percent }} Процент строк с неудачными проверками

{{ author }} Автор текущего коммита, доступна только в области действия коммита.

{{ addon_name }} Имя исполняемой в данный момент надстройки, доступна только в сообщении о коммите надстройки.

Следующие переменные доступны в шаблонах браузера репозитория или редактора:

{{branch}} текущая ветка

{{line}} строка в файле

{{filename}} имя файла, также с помощью фильтра `parentdir` вы можете удалить ведущие части, например `{{filename|parentdir}}`

Вы можете объединять их с фильтрами:

```
{{ component|title }}
```

Можете использовать условия:

```
{% if stats.translated_percent > 80 %}Well translated!{% endif %}
```

Для замены символов доступен дополнительный тег:

```
{% replace component "-" " " %}
```

Вы можете объединить его с фильтрами:

```
{% replace component|capfirst "-" " " %}
```

Также есть дополнительный фильтр для работы с именами файлов:

```
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 }}
```

…и другие возможности шаблонов Django.

2.7.6 Скорость импорта

Извлечение из репозитория системы контроля версий и импорт переводов в Weblate может быть длительным процессом, в зависимости от размера ваших переводов. Вот несколько советов:

Оптимизация конфигурации

Конфигурация по умолчанию полезна для тестирования и отладки Weblate, в то же время для установки в рабочую среду в неё необходимо внести некоторые коррективы. Многие из параметров довольно сильно влияют на производительность. Для получения более подробной информации обратитесь к разделу *Рабочая среда*, особенно обратите внимание на следующие темы:

- Настройка Celery для выполнения фоновых задач (смотрите раздел *Фоновые задачи с использованием Celery*)
- *Включение кэширования*
- *Использование мощного движка базы данных*
- *Отключение отладочного режима*

Проверка ограничения по ресурсам

Если вы импортируете огромные переводы или репозитории, вы можете столкнуться с ограничениями по ресурсам вашего сервера.

- Проверьте объём свободной памяти, наличие кэшированных операционной системой файлов перевода значительно повысит производительность.
- Если нужно обработать множество строк, дисковые операции могут стать узким местом — диск насилуется как Weblate'ом, так и базой данных.
- Улучшить производительность фоновых задач могут помочь дополнительные ядра процессора (смотрите раздел *Фоновые задачи с использованием Celery*).

Отключение ненужных проверок

Некоторые проверки качества могут быть довольно дорогими, и если в них нет необходимости, при импорте можно сэкономить некоторое время, если их пропустить. Информацию по их настройке смотрите в описании параметра `CHECK_LIST`.

2.7.7 Автоматическое создание компонентов

Если в вашем проекте есть куча файлов перевода (например, различные домены gettext или части разных Android приложений), вы можете захотеть импортировать их автоматически. Этого можно сделать либо из командной строки с помощью команд `import_project` или `import_json`, либо с помощью установки надстройки *Обнаружение компонента*.

Для использования надстройки вам сначала нужно создать компонент для одного файла перевода (выберите тот, который с наименьшей вероятностью в будущем будет переименован или удалён), и установить надстройку в этот компонент.

При использовании команд управления вам необходимо создать проект, который будет содержать все компоненты, а затем выполнить команду `import_project` или `import_json`.

См.также:

Команды управления, *Обнаружение компонента*

2.8 Определения языка

Для правильного представления различных переводов необходима информация о названии языка, направлении текста, определения форм множественного числа и код языка. В поставку включены определения примерно для 350 языков.

2.8.1 Разбор кодов языков

При разборе переводов Weblate пытается сопоставить код языка (обычно по ISO 639-1) с любым существующим объектом языка.

Дальнейшую настройку этого сопоставления на уровне проекта можно осуществить с помощью *псевдонимов языков*.

Если точное совпадение не может быть найдено, будет сделана попытка подогнать его под наиболее подходящий существующий язык (например, путём игнорирования умолчательного для данного языка кода страны — и выбором `ru` вместо `ru_RU`).

Если и это также не получится, будет создано новое определение языка с использованием настроек по умолчанию (направление текста слева направо, одна форма множественного числа) и язык буден назван как *xx_XX (generated)*. Возможно, позже вы захотите изменить его в интерфейсе администратора (смотрите раздел *Изменение определений языка*) и сообщить об этом в систему отслеживания ошибок (смотрите раздел *Внесение вклада в Weblate*).

Подсказка: В случае, если вы видите в качестве языка что-то нежелательное, вам может понадобиться настроить *языковой фильтр* на игнорирование такого файла при разборе переводов.

2.8.2 Изменение определений языка

Вы можете изменить определения языка в интерфейсе по языкам (по адресу `/languages/`).

Во время редактирования убедитесь, что все поля заполнены правильно (особенно формы множественного числа и направление текста), иначе переводчики не смогут правильно редактировать эти переводы.

2.8.3 Определения языка

Каждый язык состоит из следующих полей:

Код языка

Код, идентифицирующий язык. Weblate предпочитает двухбуквенные коды, как они определены в ISO 639-1, но для языков, для которых нет двухбуквенных кодов, использует коды ISO 639-2 или ISO 639-3. Также он может поддерживать расширенные коды, определенные в BCP 47.

См.также:

Разбор кодов языков

Название языка

Видимое название языка. Названия языков, включённых в Weblate, также локализуются в зависимости от языка пользовательского интерфейса.

Направление текста

Определяет, идёт ли запись в языке справа налево или слева направо. Для большинства языков это свойство корректно определяется автоматически.

Количество форм множественного числа

Количество используемых в языке форм множественного числа.

Формула множественного числа

Совместимая с Gettext формула множественного числа, используемая для определения того, какая форма множественного числа используется для данного количества.

См.также:

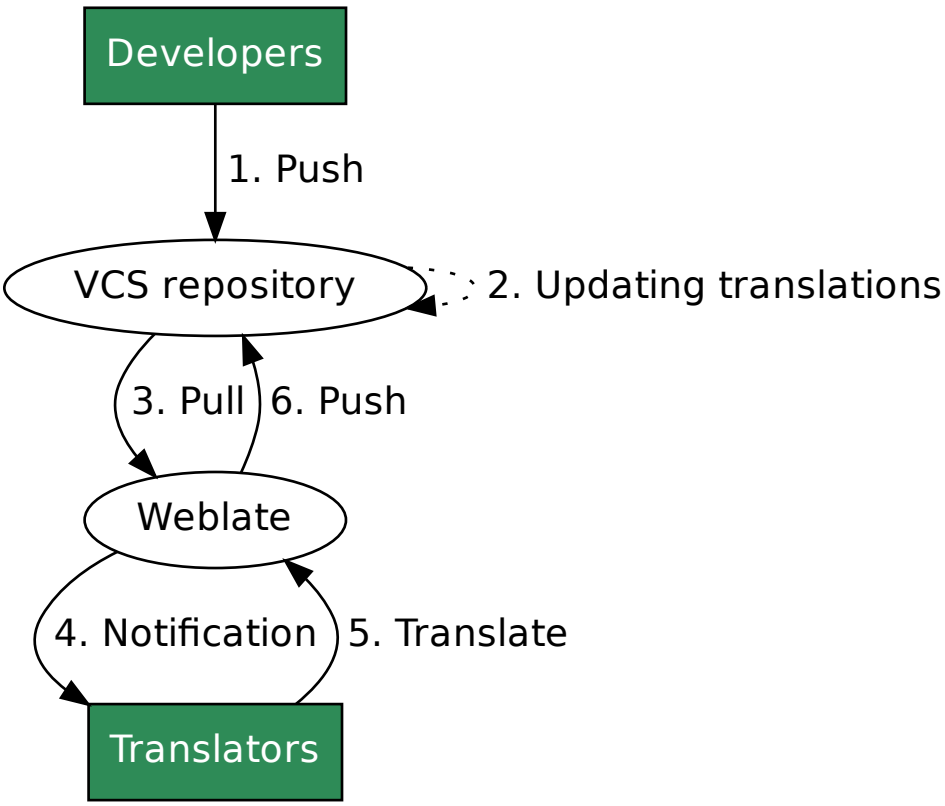
Формы множественного числа, Утилиты GNU gettext: формы множественного числа, Языковые правила для форм множественного числа от Консорциума Юникода

2.9 Непрерывная локализация

Weblate строит такую инфраструктуру, чтобы ваш перевод непрерывно следовал за разработкой. Таким образом, переводчики могут работать над переводами всё время разработки, вместо того, чтобы работать с огромным количеством нового текста непосредственно перед выпуском.

Процесс следующий:

1. Разработчики вносят изменения и отправляют их в репозиторий системы контроля версий.
2. По необходимости обновляются файлы перевода (это зависит от формата файла, смотрите вопрос *Почему Weblate до сих пор показывает старые строки перевода, хотя я уже обновил шаблон?*).
3. Weblate извлекает изменения из репозитория системы контроля версий, смотрите раздел *Обновление репозиториев*.
4. Как только Weblate обнаруживает изменения в переводах, он уведомляет об этом переводчиков в соответствии с настройками их подписок.
5. Переводчики присылают переводы через веб-интерфейс Weblate, или загружают файлы, изменённые ими в автономном режиме.
6. После завершения работы переводчиков Weblate коммитит изменения в локальный репозиторий (смотрите раздел *Отложенные коммиты*) и отправляет их обратно, если у него есть на это разрешение (смотрите раздел *Отправка изменений из Weblate'а*).



2.9.1 Обновление репозиториев

Вы должны каким-то образом настроить обновления репозиториев из их источников.

- Для интеграции с большинством распространённых сервисов хостинга кода используйте *обработчики уведомлений*
- Вручную запускайте обновление либо в разделе управления репозиторием, либо с помощью API или *клиента Weblate*
- Включите параметр `AUTO_UPDATE` для автоматического обновления всех компонентов на вашем экземпляре Weblate'а
- Выполните команду `updategit` (с выбранным проектом проекта или с ключом *–all* для обновления всех проектов)

Всякий раз, когда Weblate обновляет репозиторий, будут срабатывать надстройки «после обновления», смотрите раздел *Надстройки*.

Предотвращение конфликтов слияния

Конфликты слияния из Weblate возникают, когда один и тот же файл изменяется как в Weblate'е, так и за его пределами. Существует два подхода к решению этой проблемы — избегать редактирования файлов вне Weblate'а или интегрировать Weblate в процесс обновления таким образом, чтобы он фиксировал сделанные изменения перед обновлением файлов вне Weblate'а.

Первый подход прост с одноязычными файлами — вы можете добавлять новые строки, не выходя из Weblate'а, и возложить на него любое редактирование файлов. Для двуязычных файлов, как правило, существует некий процесс извлечения сообщений из исходного кода для генерации файлов перевода. В некоторых случаях он может быть разбит на две части — одна, извлекающая, часть генерирует шаблон (например, файл POT gettext'а генерируется с помощью программы **xgettext**), а дальнейший процесс объединяет его с фактическими переводами (файлы PO gettext'а, обновляемые с помощью программы **msgmerge**). Второй шаг вы можете выполнить в Weblate'е, и он удостоверится, что до выполнения этой операции все отложенные изменения были зафиксированы.

Второй подход может быть достигнут путём использования API, заставляющего Weblate отправить на все отложенные изменения и заблокировать перевод, пока вы на своей стороне проводите изменения.

Скрипт для выполнения обновлений может выглядеть следующим образом:

```
# 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
```

Если у вас есть несколько компонентов, совместно использующих один и тот же репозиторий, вам необходимо заблокировать их все по отдельности:

```
wlc lock foo/bar
wlc lock foo/baz
wlc lock foo/baj
```

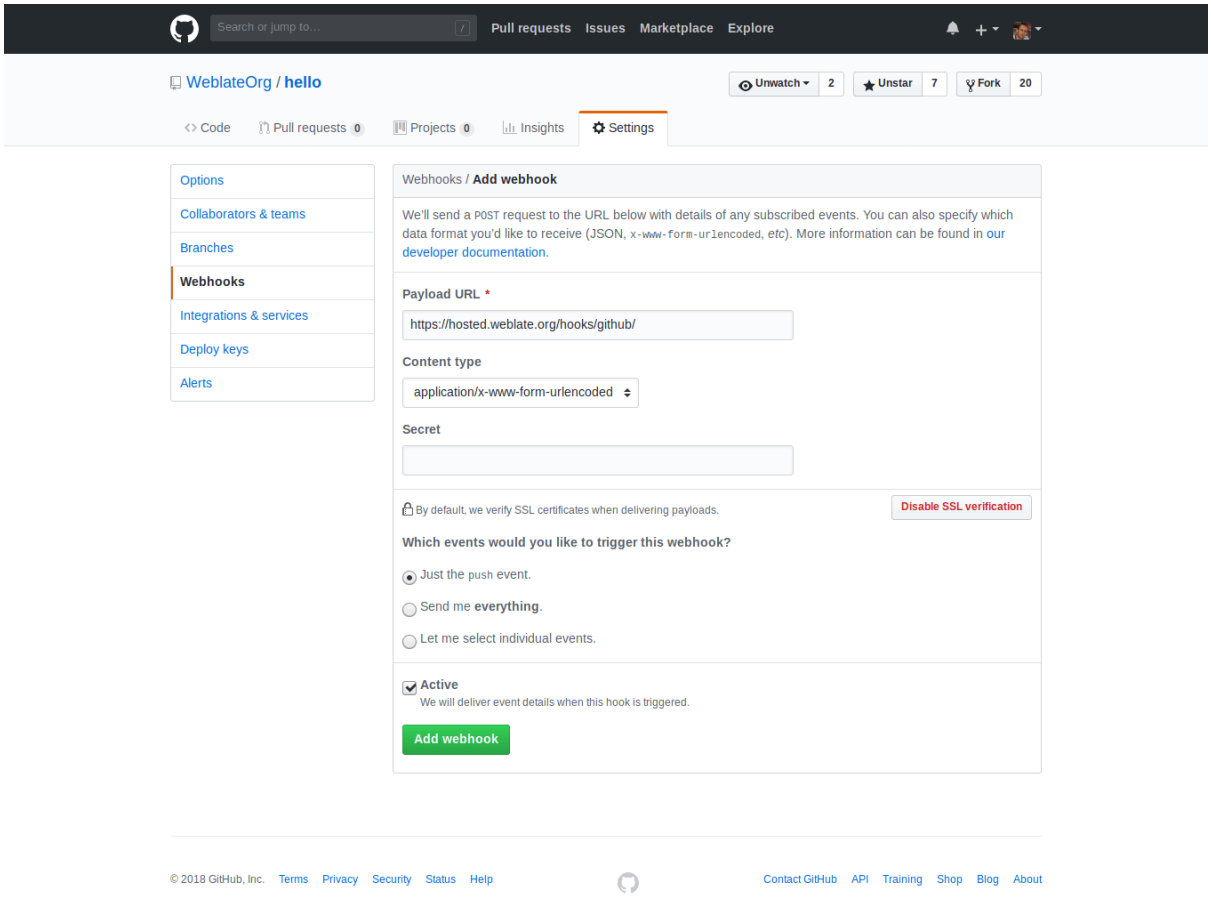
Примечание: В примере используется *клиент Weblate*, который, чтобы иметь возможность управлять Weblate'ом удалённо, нуждается в некоторой настройке (в ключах API). Также вы можете решить задачу при помощи любого HTTP-клиента вместо wlc, к примеру, curl, смотрите раздел API.

Автоматическое получение изменений из GitHub'а

Weblate поставляется со встроенной поддержкой GitHub'а.

Если вы используете Hosted Weblate, рекомендуемый подход заключается в установке приложения Weblate, таким образом вы получите правильную настройку без необходимости в дополнительной настройке. Также оно может быть использовано для отправки изменений обратно в GitHub.

Для получения уведомлений о каждой отправке в репозиторий GitHub, добавьте в настройки репозитория (*Webhooks*) веб-обработчик Weblate, как показано на изображении ниже:



Для заполнения адреса полезной нагрузки (поля *Payload URL*), добавьте к URL-адресу вашего Weblate'а суффикс `/hooks/github/`, например, для сервиса Hosted Weblate, это этот адрес будет таким: `https://hosted.weblate.org/hooks/github/`.

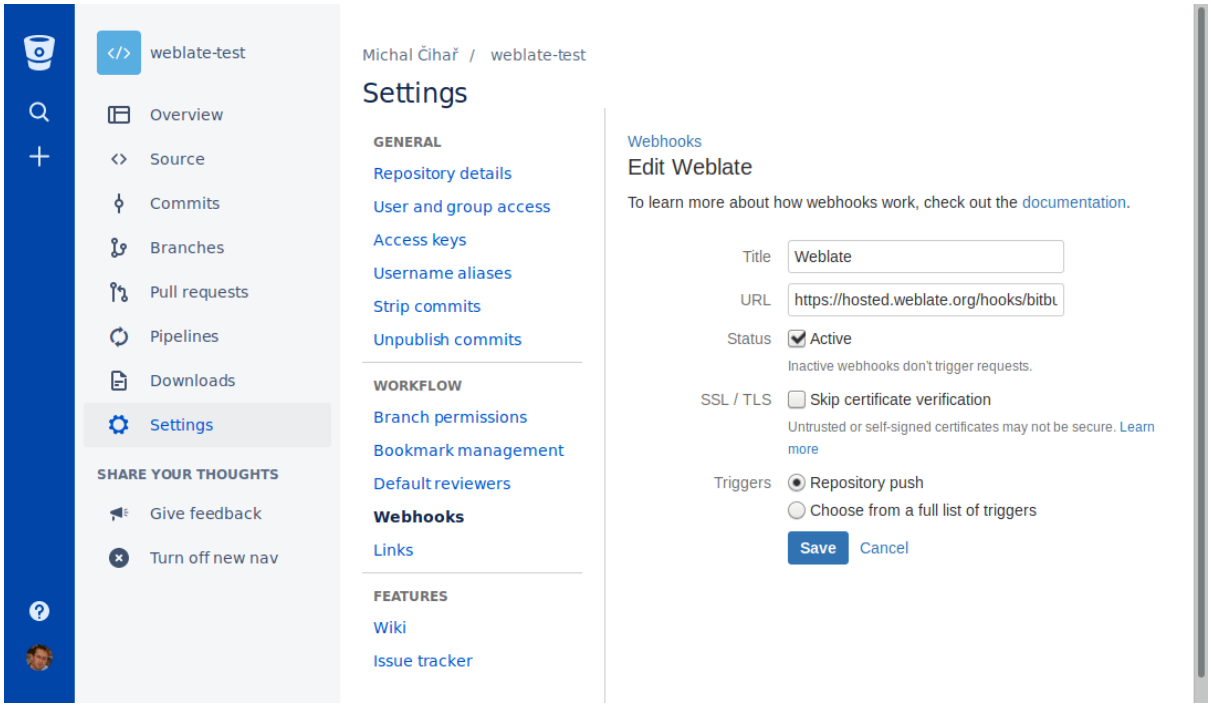
Остальные поля вы можете оставить с настройками по умолчанию (Weblate умеет обрабатывать оба типа содержимого и потребляет только событие *push*).

См.также:

`POST /hooks/github/`, *Доступ к репозиториям из Hosted Weblate*

Автоматическое получение изменений из Bitbucket'a

Weblate поддерживает веб-обработчики Bitbucket, добавьте веб-обработчик, который срабатывает при отправке изменений в репозиторий, указав в качестве *URL* адрес вашей установки Weblate с суффиксом `/hooks/bitbucket/` (например, `https://hosted.weblate.org/hooks/bitbucket/`).



См.также:

`POST /hooks/bitbucket/`, *Доступ к репозиториям из Hosted Weblate*

Автоматическое получение изменений из GitLab'a

Weblate поддерживает обработчики GitLab'a, добавьте веб-обработчик проекта с адресом назначения, равным адресу вашей установки Weblate с суффиксом `/hooks/gitlab/` (например, `https://hosted.weblate.org/hooks/gitlab/`).

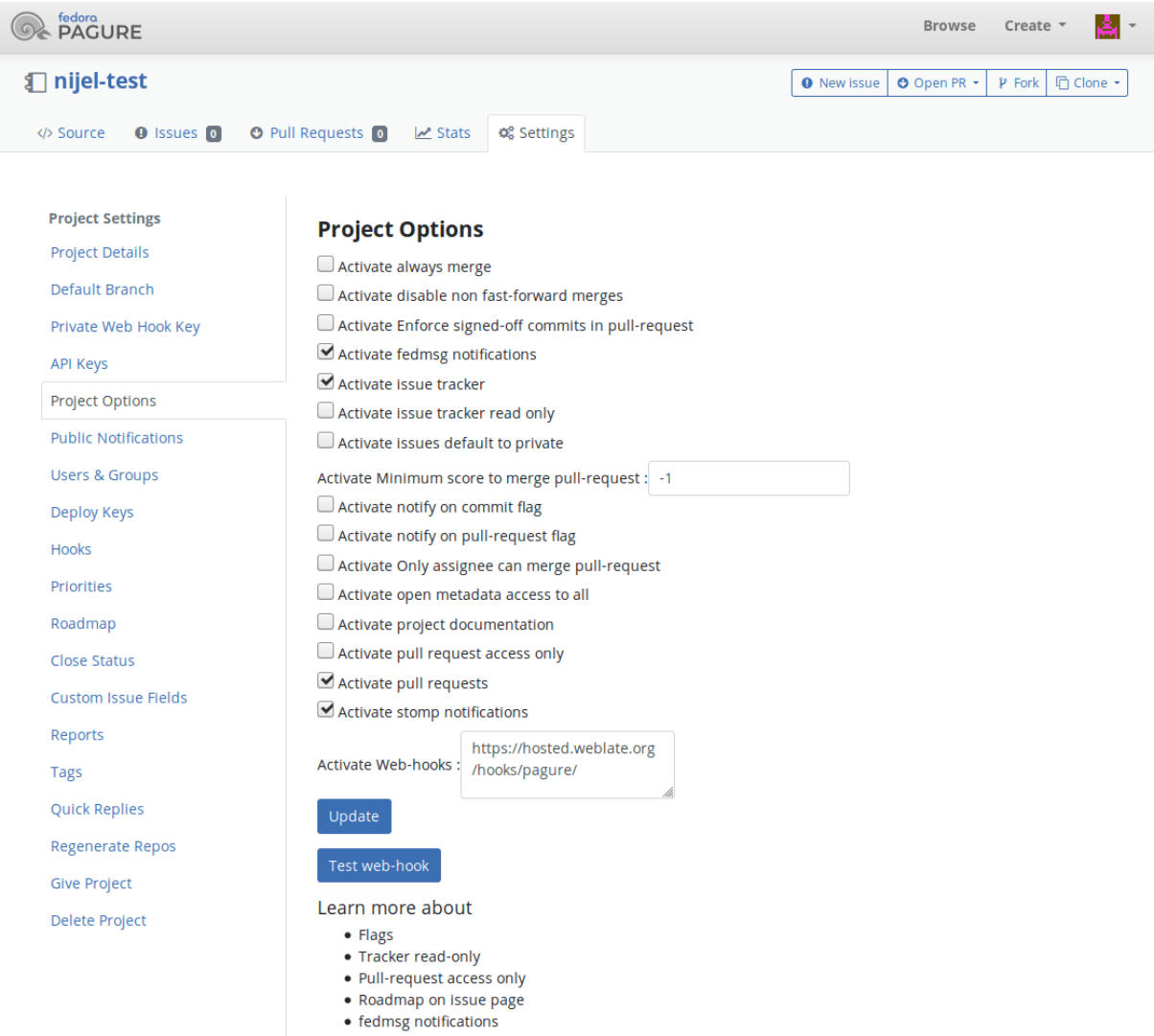
См.также:

`POST /hooks/gitlab/`, *Доступ к репозиториям из Hosted Weblate*

Автоматическое получение изменений из Pagure

Добавлено в версии 3.3.

Weblate поддерживает обработчики Pagure, добавьте веб-обработчик проекта с адресом назначения, равным адресу вашей установки Weblate с суффиксом `/hooks/pagure/` (например, `https://hosted.weblate.org/hooks/pagure/`). Добавлять этот адрес нужно в поле *Activate Web-hooks*, расположенное в блоке *Project options*:



См.также:

POST /hooks/pagure/, *Доступ к репозиториям из Hosted Weblate*

Автоматическое получение изменений из Azure Repos

Добавлено в версии 3.8.

Weblate поддерживает веб-обработчики Azure Repos, добавьте веб-обработчик для события *Code pushed* с адресом назначения, равным адресу вашей установки Weblate с суффиксом `/hooks/azure/` (например, `https://hosted.weblate.org/hooks/azure/`). Добавлять этот адрес нужно в поле *Service hooks*, расположенное в блоке *Project settings*.

См.также:

Веб-обработчики в руководстве по Azure DevOps, *POST /hooks/azure/*, *Доступ к репозиториям из Hosted Weblate*

Автоматическое получение изменений из репозиториев Gitea

Добавлено в версии 3.9.

Weblate поддерживает веб-обработчики Gitea, добавьте веб-обработчик *Gitea Webhook* для события *Push events* с адресом назначения, равным адресу вашей установки Weblate с суффиксом `/hooks/gitea/` (например, `https://hosted.weblate.org/hooks/gitea/`). Добавлять этот адрес нужно в поле *Webhooks*, расположенное в блоке *Settings*.

См.также:

Веб-обработчики в руководстве по Gitea, `POST /hooks/gitea/`, *Доступ к репозиториям из Hosted Weblate*

Автоматическое получение изменений из репозиториев Gitee

Добавлено в версии 3.9.

Weblate поддерживает веб-обработчики Gitee, добавьте веб-обработчик *WebHook* для события *Push* с адресом назначения, равным адресу вашей установки Weblate с суффиксом `/hooks/gitee/` (например, `https://hosted.weblate.org/hooks/gitee/`). Добавлять этот адрес нужно в поле *WebHooks*, расположенное в блоке *Management*.

См.также:

Веб-обработчики в руководстве по Gitee, `POST /hooks/gitee/`, *Доступ к репозиториям из Hosted Weblate*

Автоматическое ночное обновление репозиториев

Weblate по ночам автоматически извлекает изменения из удалённых репозиториев для повышения производительности при последующем слиянии изменений. По желанию вы можете превратить его и в ночное слияние, включив параметр `AUTO_UPDATE`.

2.9.2 Отправка изменений из Weblate'а

Каждый компонент перевода может иметь настроенный URL-адрес для отправки (смотрите описание параметра *URL для отправки в репозиторий*), в этом случае Weblate будет способен отсылать изменения в удалённый репозиторий. Также Weblate может быть настроен на автоматическую отсылку изменений при каждом коммите (это поведение по умолчанию, смотрите описание параметра *Отправлять при коммите*). Если вы не хотите, чтобы изменения отправлялись автоматически, вы можете отправлять их вручную в разделе *Обслуживание репозитория* или через API командой `wlc push`.

Параметры отправки отличаются в зависимости от используемой *системы контроля версий*, подробнее об этом читайте в указанной главе.

Если вам не нужна отправка изменений из Weblate напрямую, он поддерживает создание запросов на извлечение для *GitHub'а*, *GitLab'а* или рецензий *Gerrit*, вы можете активировать их, выбрав в *конфигурации компонента* в качестве *системы контроля версий* *GitHub*, *GitLab* или *Gerrit*.

В целом, для Git'а, GitHub'а и GitLab'а доступны следующие варианты:

Желаемая настройка	*Система контроля версий*	*URL для отправки в репозиторий*	*Ветка для отправки*
Без отправки	*Git*	*пусто*	*пусто*
Отправка напрямую	*Git*	URL-адрес SSH	*пусто*
Отправка в отдельную ветку	*Git*	URL-адрес SSH	Имя ветки
GitHub'овский запрос на извлечение из форка	*GitHub*	*пусто*	*пусто*
GitHub'овский запрос на извлечение из ветки	*GitHub*	URL-адрес SSH[1]	Имя ветки
GitLab 'овский запрос на слияние из форка	*GitLab*	*пусто*	*пусто*
GitLab 'овский запрос на слияние из ветки	*GitLab*	URL-адрес SSH[1]	Имя ветки

Примечание: Также вы можете включить автоматическую отправку изменений после коммитов Weblate'а, это можно сделать установкой параметра *Отправлять при коммите*.

См.также:

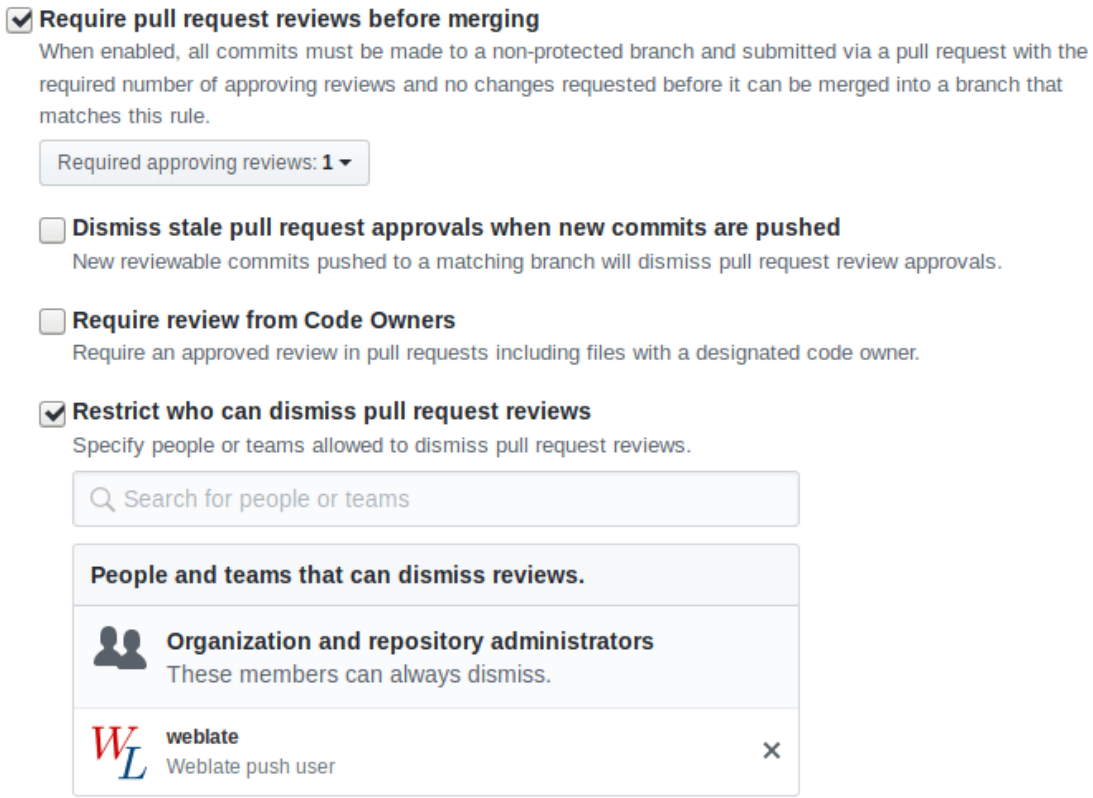
Для настройки ключей SSH смотрите раздел *Доступ к репозиториям*, а для получения информации о том, когда Weblate решает закоммитить изменения — раздел *Отложенные коммиты*.

Защищённые ветки

Если вы используете Weblate на защищённой ветке, вы можете настроить его на использование запросов на извлечение и выполнение рецензирования переводов (что может быть проблематично для языков, которых вы не знаете). Альтернативный подход заключается в отмене этого ограничения для пользователя Weblate.

Например, на GitHub'е это можно сделать в настройках репозитория:

[1] Может быть пустым в случае, если *репозиторий исходного кода* поддерживает отправку изменений.



2.9.3 Слияние или перебазирование

По умолчанию Weblate вливает вышестоящий репозиторий в свой собственный. Это самый безопасный способ в случае, если вы помимо этого работаете с нижележащим репозиторием и другими способами. Если вам это не нужно, вы можете включить перебазирование изменений на голову вышестоящего репозитория, что позволит создавать историю с меньшим количеством коммитов слияния.

Примечание: В случае сложных слияний перебазирование может доставить вам проблем, поэтому тщательно подумайте, хотите ли вы его включать или нет.

2.9.4 Взаимодействие с другими пользователями

Weblate облегчает взаимодействие с другими пользователями с помощью своего API.

См.также:

API

2.9.5 Отложенные коммиты

Поведение Weblate'а заключается в группировке коммитов одного и того же автора в один коммит (если это возможно). Такая группировка значительно сокращает количество коммитов, однако вам может понадобиться явно приказать ему сделать коммиты в случае, если вы хотите синхронизировать репозиторий, например, для выполнения слияния (по умолчанию это действие разрешено для группы *Управляющие*, смотрите раздел *Управление доступом*).

Изменения в этом режиме коммитятся при выполнении любого из следующих условий:

- Кто-то другой изменяет уже изменённую строку.
- Происходит слияние с вышестоящим репозиторием.
- Запрошен явный коммит изменений.
- Изменения старше периода, определённого параметром *Возраст изменений для коммита* в *конфигурации компонента*.

Подсказка: Коммиты создаются для каждого компонента. Так что в случае, если у вас много компонентов, у вас всё равно будет множество коммитов. В этом случае вы можете использовать надстройку *Склеивание Git-коммитов*.

Если вы хотите коммитить изменения чаще и без проверки их возраста, вы можете запланировать для выполнения коммита обычное задание:

```
CELERY_BEAT_SCHEDULE = {
    # Unconditionally commit all changes every 2 minutes
    "commit": {
        "task": "weblate.trans.tasks.commit_pending",
        # Ommiting hours will honor per component settings,
        # otherwise components with no changes older than this
        # won't be committed
        "kwargs": {"hours": 0},
        # How frequently to execute the job in seconds
        "schedule": 120,
    }
}
```

2.9.6 Обработка репозитория скриптами

Настройка взаимодействия Weblate'а с репозиторием заключается в использовании *надстроек*. Для получения информации о том, как через надстройки выполнять внешние скрипты, обратитесь к разделу *Выполнение скриптов из надстройки*.

2.9.7 Поддержание единого перевода в разных компонентах

Если у вас несколько компонентов перевода, вы возможно захотите убедиться, что одни и те же строки имеют один и тот же перевод. Этого можно достичь на нескольких уровнях.

Распространение перевода

При включённой функции распространения перевода (которая включена по умолчанию, смотрите раздел *Конфигурация компонента*), все новые переводы автоматически копируются во все компоненты с совпадающими строками. Такие переводы должным образом засчитываются текущему пользователю-переводчику во всех компонентах.

Примечание: Распространение перевода требует, чтобы в одноязычных форматах файлов перевода ключи строк совпадали, так что об этом следует помнить при создании ключей перевода.

Проверка согласованности

Проверка *Противоречия* срабатывает всякий раз, когда строки отличаются друг от друга. Вы можете использовать это, чтобы отрецензировать такие различия вручную и выбрать правильный перевод.

Автоматический перевод

Автоматический перевод одних компонентов на основе переводов в других может быть способом синхронизации переводов между компонентами. Вы можете либо запустить его вручную (смотрите раздел *Автоматический перевод*), либо заставить его автоматически запускаться при обновлении репозитория с помощью надстройки *Автоматический перевод*.

2.10 Лицензирование переводов

Вы можете указать, по каким лицензиям предоставляются переводы. Особенно оно важно, если переводы находятся в открытом доступе, так как оно помогает определить, для чего они могут быть использованы.

Вы должны указать информацию о лицензии *компонента*. Вам следует избегать требования заключения лицензионного соглашения с участниками, хотя это вполне возможно.

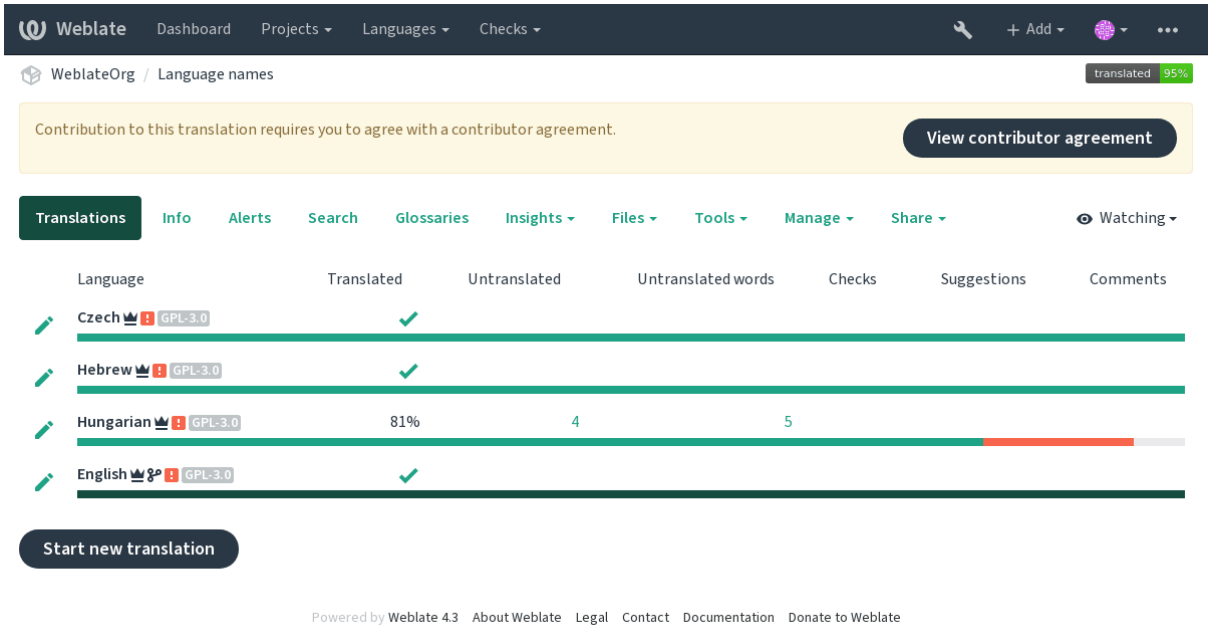
2.10.1 Информация о лицензии

После указании информации о лицензии (имени лицензии и её URL-адреса) эта информация отображается в разделе «Инфо» соответствующего *компонента*.

Обычно, если на перевод не требуется явного согласия, это лучшее место для размещения информации о лицензии. Если ваш проект или перевод не являются свободными, то, скорее всего, предварительное согласие вам необходимо.

2.10.2 Соглашение участника

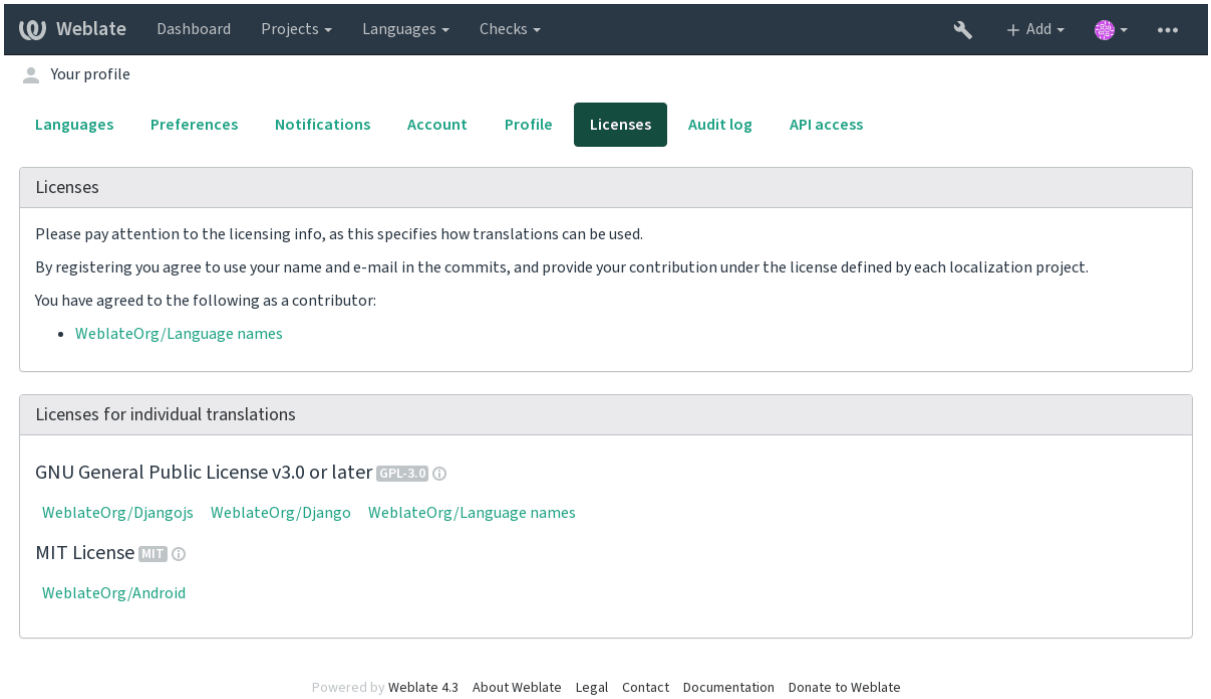
Если вы укажете лицензионное соглашение участника, то вносить свой вклад в перевод смогут только пользователи, с ним согласившиеся. Это хорошо заметный шаг, появляющийся при доступе к переводу:



Введённый текст разбивается на абзацы и может содержать внешние ссылки. Разметку HTML в нём использовать нельзя.

2.10.3 Пользовательские лицензии

Любой пользователь может просмотреть из своего профиля все лицензии на перевод всех публичных проектов на экземпляре:



2.11 Процесс перевода

2.11.1 Голосование по предложению

По умолчанию кто угодно может добавлять предложения, которые могут быть приняты зарегистрированными пользователями. Если с предложением согласны не только зарегистрированные пользователи, для утверждения строки может использоваться голосование по предложениям, которое можно включить в *конфигурации компонента* параметром *Голосование по предложению* и настроить порог голосов для принятия предложения в параметре *Автоматическое принятие предложений* (считается также голос пользователя, сделавшего предложение).

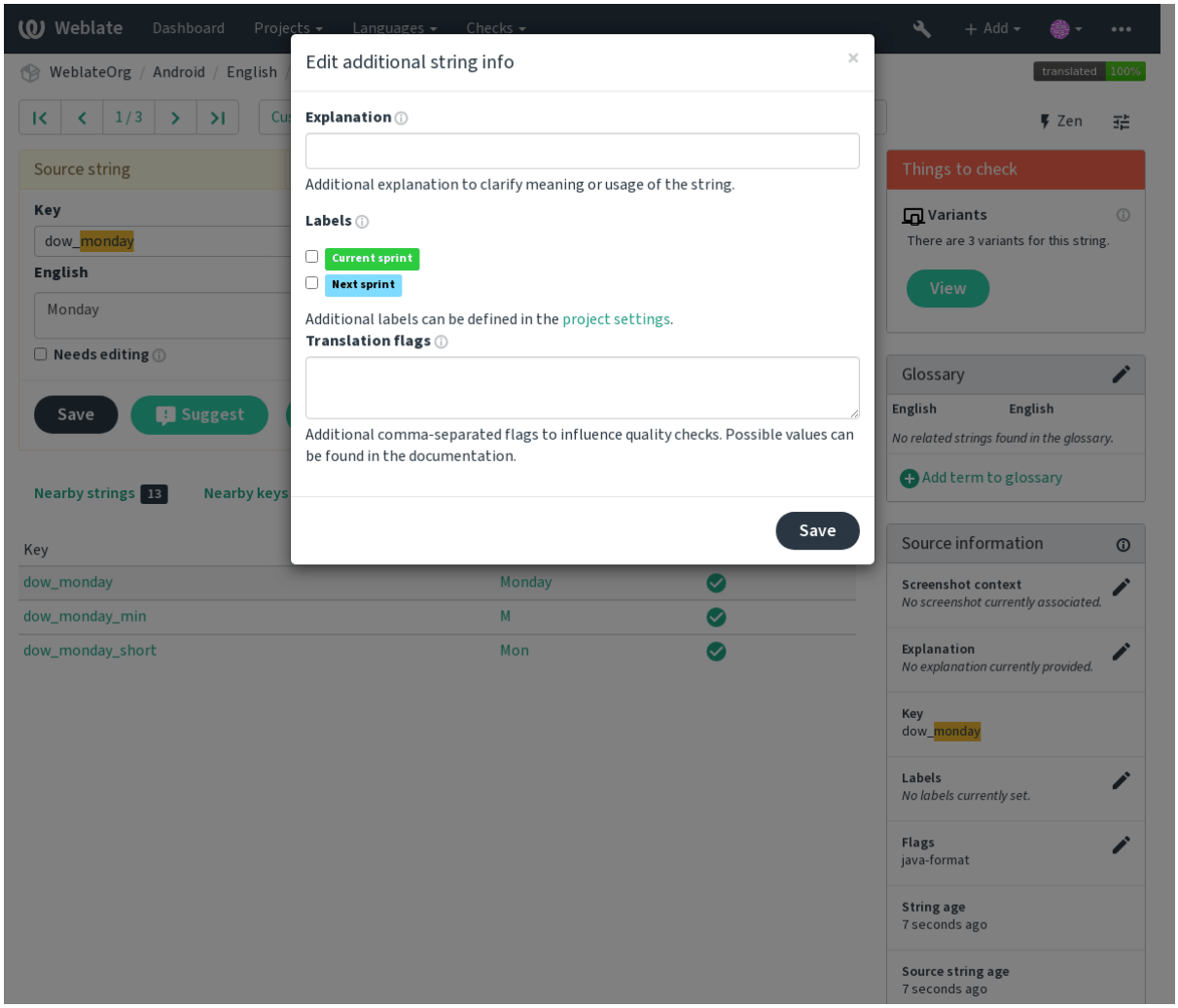
Примечание: После настройки автоматического принятия обычные пользователи теряют привилегию непосредственного сохранения переводов или принятия предложений. Это можно отменить с помощью привилегии *Правка строк, когда предложения являются обязательными* (смотрите раздел *Управление доступом*).

Вы можете скомбинировать этот параметр с *управлением доступом* для получения одной из следующих установок:

- Пользователи предлагают и голосуют за предложения, а ограниченная группа контролирует то, что принимается. — Включите голосование. — Отключите автоматическое принятие. — Не позволяйте пользователям сохранять переводы.
- Пользователи предлагают и голосуют за предложения с автоматическим принятием предложения, как только с ним согласится определённое количество пользователей. — Включите голосование. — Установите желаемое количество голосов для автоматического принятия предложения.
- Необязательное голосование по предложениям (по желанию может использоваться пользователями, когда они не уверены в переводе, и делают вместо него несколько предложений). — Включите только голосование.

2.11.2 Дополнительная информация об исходных строках

Усовершенствуйте процесс перевода с помощью информации, доступной в файлах перевода. Усовершенствование включает в себя объяснение, приоритеты строк, флаги проверок или предоставление визуального контекста. Все эти возможности могут быть установлены в окне *Рецензирование строк*:



Доступ к нему можно получить непосредственно из интерфейса перевода, нажав на иконку правки у разделов *Контекст — снимок экрана* или *Флаги*.

Weblate Dashboard Projects Languages Checks + Add

WeblateOrg / Django / Czech / Translate translated 96%

11 / 26 All strings Position and priority Zen

Translation

Explanation

Help text for automatic translation tool

English

Automatic translation via machine translation uses active machine translation engines to get the best possible translations and applies them in this project.

Czech Clone source NBS

Automatický překlad prostřednictvím strojového překladu používá aktivní enginy strojového překladu pro získání nejlepších možných překladů a použije je na tento projekt.

169/1570

Needs editing

Save Suggest Skip

Nearby strings 26 Comments Automatic suggestions Other languages History

Context	English	Czech	State
	Files	Soubory	
	Automatic translation	Automatický překlad	
	Add new translation string	Add new translation string	
	Translation status	Stav překladu	
	%(count)s word	%(count)s slovo	
	Other components	Další součásti	
	Translation file	Soubor s překladem	
	Download	Stáhnout	
	Browse all translation changes	Procházet všechny změny v překladu.	
	Automatic translation takes existing translations in this project and applies them to the current component. It can be used to push translations to a different branch, to fix inconsistent translations or to translate a new component using translation memory.	Automatický překlad použije stávající překlady v projektu na tuto součást. Může být užitečný pro sloučení překladů z jiné větve, opravu nekonzistentních překladů nebo překlad nové součásti pomocí překladové paměti.	
	Automatic translation via machine translation uses active machine translation engines to get the best possible translations and applies them in this project.	Automatický překlad prostřednictvím strojového překladu používá aktivní enginy strojového překladu pro získání nejlepších možných překladů a použije je na tento projekt.	
	You can add new translation string here, it will automatically appear in all translations.	Zde můžete přidat nový řetězec k překladu, automaticky se objeví ve všech jazycích.	
	The uploaded file will be merged with the current translation. In case you want to overwrite already translated strings, don't forget to enable it.	Nahraný soubor bude sloučen se stávajícími překlady. Pokud chcete přepsat již přeložené řetězce, nezapomeňte to povolit.	
	The uploaded file will be merged with the current translation.	Nahraný soubor bude sloučen se stávajícími překlady.	
	The fulltext search might not work properly as the fulltext index for this translation is not yet up to date.	Fulltextové vyhledávání nemusí fungovat správně, protože fulltextový index pro tento překlad ještě není plně zpracován.	
	Review	Kontrola	
	Review translations touched by other users.	Zkontrolovat překlady od ostatních uživatelů.	
	Start review	Začít kontrolu	
	Percent	Procenta	
	Total	Celkem	
	Failing check	Neúspěšných kontrol	
	Last activity	Poslední aktivita	
	Last change	Poslední změna	
	Last author	Poslední autor	
Question for a mathematics-based CAPTCHA, the %s is an arithmetic problem	What is %s?	Kolik to je?	
	The string uses three dots (...) instead of an ellipsis character (…)		

Glossary

English	Czech	
machine translation	strojový překlad	WeblateOrg
project	projekt	WeblateOrg

Add term to glossary

Source information

Screenshot context
No screenshot currently associated.

Explanation
Help text for automatic translation tool

Labels
No labels currently set.

Flags
No flags currently set.

Source string location
weblate/templates/translation.html:212

String age
a second ago

Source string age
a second ago

Translation file
weblate/locale/cs/LC_MESSAGES/django.po, string 11

Powered by Weblate 4.3 About Weblate Legal Contact Documentation Donate to Weblate

Приоритизация строк

Добавлено в версии 2.0.

Приоритет строк может быть изменён с помощью флага `priority`, чтобы более приоритетные строки предлагались для перевода раньше

Подсказка: Это можно использовать для перевода строк в логическом порядке.

См.также:

Проверки качества

Флаги перевода

Добавлено в версии 2.4.

Изменено в версии 3.3: Ранее поле называлось *Отметки о проверке качества*, но теперь оно настраивает не только проверки.

Умолчательный набор флагов перевода определяется *переводимым компонентом* и файлом перевода. Тем не менее, вы можете использовать данное поле для настройки этих флагов для каждой исходной строки.

См.также:

Проверки качества

Объяснение

Изменено в версии 4.1: В предыдущей версии это поле называлось дополнительным контекстом.

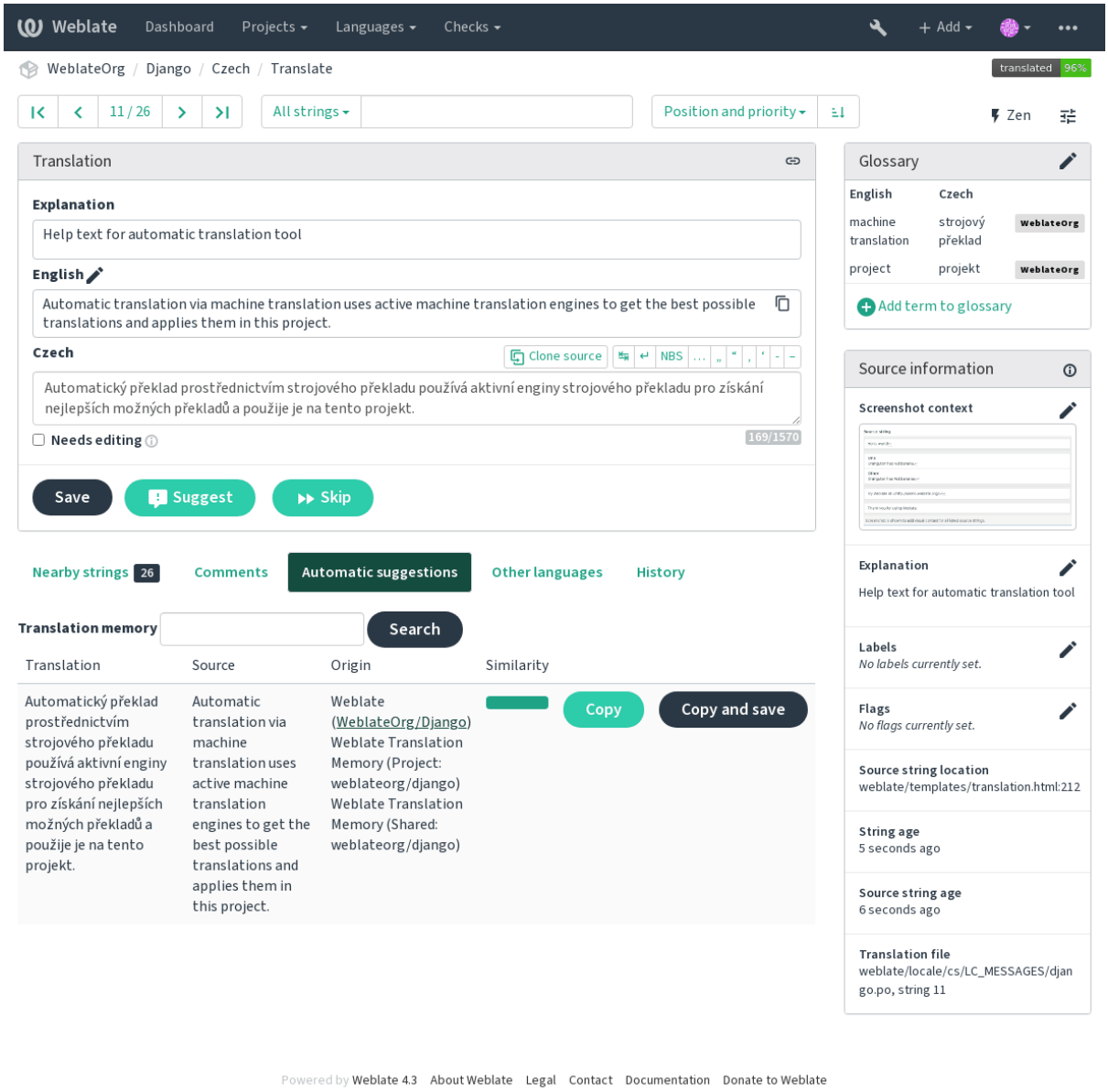
Используйте объяснение для уточнения области действия или использования перевода. В поле работает разметка Markdown, так что вы можете включать сюда ссылки и выделять текст.

Визуальный контекст для строк

Добавлено в версии 2.9.

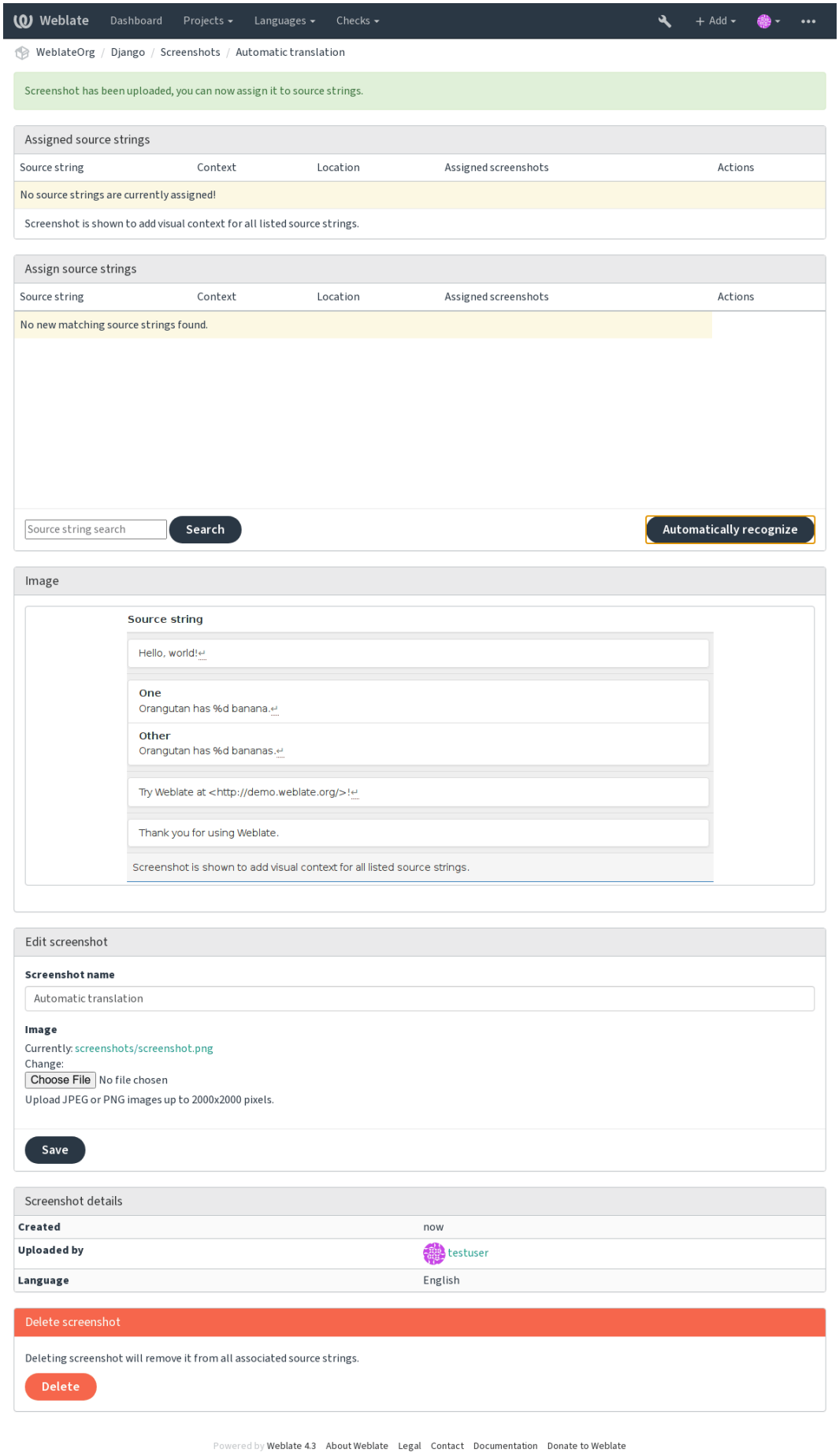
Вы можете загрузить снимок экрана, показывающий данную исходную строку в действии в вашей программе. Это поможет переводчикам понять, где она используется и как её следует переводить.

Загруженный снимок экрана отображается в боковой панели контекста перевода:



В дополнение к окну *Рецензирование строк*, снимки экрана имеют отдельный интерфейс управления, расположенный в меню *Инструменты*. Загружайте снимки экрана и вручную назначайте их исходным строкам, либо используйте для этого оптическое распознавание символов.

После загрузки снимка экрана через этот интерфейс можно управлять снимком и задать его связь с исходной строкой:


Weblate
Dashboard
Projects
Languages
Checks
+ Add
WeblateOrg / Django / Screenshots / Automatic translation
Screenshot has been uploaded, you can now assign it to source strings.
Assigned source strings
Source string
Context
Location
Assigned screenshots
Actions
No source strings are currently assigned!
Screenshot is shown to add visual context for all listed source strings.
Assign source strings
Source string
Context
Location
Assigned screenshots
Actions
No new matching source strings found.
Source string search
Search
Automatically recognize
Image
Source string
Hello, world!
One
Orangutan has %d banana.
Other
Orangutan has %d bananas.
Try Weblate at <http://demo.weblate.org/>!
Thank you for using Weblate.
Screenshot is shown to add visual context for all listed source strings.
Edit screenshot
Screenshot name
Automatic translation
Image
Currently: screenshots/screenshot.png
Change:
Choose File
No file chosen
Upload JPEG or PNG images up to 2000x2000 pixels.
Save
Screenshot details
Created
now
Uploaded by
testuser
Language
English
Delete screenshot
Deleting screenshot will remove it from all associated source strings.
Delete
Powered by Weblate 4.3
About Weblate
Legal
Contact
Documentation
Donate to Weblate

2.12 Проверки и исправления

2.12.1 Пользовательские автоматические исправления

Также в дополнение к стандартным вы можете реализовать своё собственное автоматическое исправление и включить его в список *AUTOFIX_LIST*.

Автоматические исправления — это очень мощный инструмент, но они могут повредить перевод; будьте осторожны при их написании.

Например, следующее автоматическое исправление заменит каждое появление строки `foo` в переводе на строку `bar`:

```
#
# Copyright © 2012 - 2020 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <https://www.gnu.org/licenses/>.
#

from django.utils.translation import gettext_lazy as _

from weblate.trans.autofixes.base import AutoFix


class ReplaceFooWithBar(AutoFix):
    """Replace foo with bar."""

    name = _("Foobar")

    def fix_single_target(self, target, source, unit):
        if "foo" in target:
            return target.replace("foo", "bar"), True
        return target, False
```

Для установки пользовательских проверок необходимо в списке *AUTOFIX_LIST* указать полный путь к классу Python, смотрите раздел *Пользовательские проверки качества, надстройки и автоматические исправления*.

2.12.2 Настройка поведения

Вы можете тонко настроить поведение Weblate'а (в основном проверки) для каждой исходной строки (в рецензировании исходных строк, смотрите раздел *Дополнительная информация об исходных строках*) или в *конфигурации компонента* (*Флаги перевода*). Некоторые форматы файлов также позволяют указывать флаги непосредственно в формате (смотрите раздел *Поддерживаемые форматы файлов*).

Флаги разделяются запятыми, параметры флагов разделяются двоеточием. Для включения в строку пробела или специальных символов вы можете использовать кавычки. Например:

```
placeholders:"special:value":"other value", regex:.*
```

Вот список принимаемых в настоящее время флагов:

rst-text Рассматривать текст как документ reStructuredText, влияет на проверку *Перевод без изменения*.

md-text Рассматривать текст как документ Markdown.

dos-eol Использовать маркеры конца строки DOS вместо маркеров Unix (комбинацию символов `\r\n` вместо символа `\n`).

url Строка должна состоять только из URL-адреса.

safe-html Строка должна быть безопасной для HTML, смотрите проверку *Небезопасный HTML-код*.

read-only Строка доступна только для чтения и не должна редактироваться в Weblate, смотрите проверку *Строки только для чтения*.

priority:N Приоритет строки. Строки с более высоким приоритетом предлагаются на перевод первыми. По умолчанию приоритет равен 100, чем выше приоритет строки, тем раньше она будет предложена для перевода.

max-length:N Ограничивает максимальную длину строки N символами, смотрите проверку *Максимальная длина перевода*.

xml-text Рассматривать текст как документ XML, влияет на проверки *Синтаксис XML* и *XML-разметка*.

font-family:NAME Определяет семейство шрифтов для проверок отрисовки текста, смотрите раздел *Управление шрифтами*.

font-weight:WEIGHT Определяет вес шрифта (свойство `font-weight`) для проверок отрисовки текста, смотрите раздел *Управление шрифтами*.

font-size:SIZE Определяет размер шрифта (свойство `font-size`) для проверок отрисовки текста, смотрите раздел *Управление шрифтами*.

font-spacing:SPACING Определяет разрядку шрифта (свойство `font-spacing`) для проверок отрисовки текста, смотрите раздел *Управление шрифтами*.

placeholders:NAME Ожидаемые в переводе строки-заполнители, смотрите проверку *Символы-заполнители*.

replacements:FROM:TO:FROM2:TO2...` Замены, выполняемые при проверке параметров результирующего текста (например, в проверках *Максимальный размер перевода* или *Максимальная длина перевода*). Типичным их случаем использования является расширение заполнителей для обеспечения того, чтобы текст вписывался в отведённые ему границы даже при использовании длинных имён, например: `replacements:%s:"John Doe"`.

regex:REGEX Регулярное выражение для сопоставления с переводом, смотрите раздел *Регулярное выражение*.

python-format, c-format, php-format, python-brace-format, javascript-format, c-sharp-format, ja
Рассматривает все строки как строки формата, влияет на проверки *Форматированные строки*, *Форматированные строки*, *Форматированные строки*, *Форматированные строки*, *Форматированные строки*, *Форматированные строки*, *Форматированные строки*, *Форматированные строки*, *Форматированные строки*, *Форматированные строки*, *Форматированные строки* и *Перевод без изменения*.

strict-same Отключает использование проверкой «Перевод без изменения» встроенного чёрного списка слов, смотрите проверку *Перевод без изменения*.

ignore-bbcode Пропускает проверку качества «BBcode-разметка».

ignore-duplicate Пропускает проверку качества «Повторяющиеся слова».

ignore-double-space Пропускает проверку качества «Двойной пробел».

ignore-angularjs-format Пропускает проверку качества «Интерполяционная строка AngularJS».

ignore-c-format Пропускает проверку качества «C-формат».

ignore-c-sharp-format Пропускает проверку качества «C#-формат».

ignore-es-format Пропускает проверку качества «Шаблонные литералы ECMAScript».

ignore-i18next-interpolation Пропускает проверку качества «Интерполяция i18next».

ignore-java-format Пропускает проверку качества «Java-формат».

ignore-java-messageformat Пропускает проверку качества «Java MessageFormat».

ignore-javascript-format Пропускает проверку качества «JavaScript-формат».

ignore-percent-placeholders Пропускает проверку качества «Символы-заполнители в процентной нотации».

ignore-perl-format Пропускает проверку качества «Perl-формат».

ignore-php-format Пропускает проверку качества «PHP-формат».

ignore-python-brace-format Пропускает проверку качества «Python-формат: фигурные скобки».

ignore-python-format Пропускает проверку качества «Python-формат».

ignore-qt-format Пропускает проверку качества «Qt-формат».

ignore-qt-plural-format Пропускает проверку качества «Формат множественного числа Qt».

ignore-ruby-format Пропускает проверку качества «Ruby-формат».

ignore-vue-format Пропускает проверку качества «Форматирование Vue I18n».

ignore-translated Пропускает проверку качества «Уже переведено».

ignore-inconsistent Пропускает проверку качества «Противоречия».

ignore-kashida Пропускает проверку качества «Используется кашида».

ignore-md-link Пропускает проверку качества «Ссылки Markdown».

ignore-md-reflink Пропускает проверку качества «Сноски Markdown».

ignore-md-syntax Пропускает проверку качества «Синтаксис Markdown».

ignore-max-length Пропускает проверку качества «Максимальная длина перевода».

ignore-max-size Пропускает проверку качества «Максимальный размер перевода».

ignore-escaped-newline Пропускает проверку качества «Несовпадающие n».

ignore-end-colon Пропускает проверку качества «Несовпадающие двоеточия».

ignore-end-ellipsis Пропускает проверку качества «Несовпадающие многоточия».

ignore-end-exclamation Пропускает проверку качества «Несовпадающие восклицательные знаки».

ignore-end-stop Пропускает проверку качества «Несовпадающие точки».

ignore-end-question Пропускает проверку качества «Несовпадающие вопросительные знаки».

ignore-end-semicolon Пропускает проверку качества «Несовпадающие точки с запятой».

ignore-newline-count Пропускает проверку качества «Несовпадающие переводы строк».

ignore-plurals Пропускает проверку качества «Нет форм множественного числа».

ignore-placeholders Пропускает проверку качества «Символы-заполнители».

ignore-punctuation-spacing Пропускает проверку качества «Пробелы и пунктуация».

ignore-regex Пропускает проверку качества «Регулярное выражение».

ignore-same-plurals Пропускает проверку качества «Одинаковые формы множественного числа».

ignore-begin-newline Пропускает проверку качества «Пустая строка в начале».

ignore-begin-space Пропускает проверку качества «Пробелы в начале».

ignore-end-newline Пропускает проверку качества «Пустая строка в конце».

ignore-end-space Пропускает проверку качества «Пробелы в конце».

ignore-same Пропускает проверку качества «Перевод без изменения».

ignore-safe-html Пропускает проверку качества «Небезопасный HTML-код».

ignore-url Пропускает проверку качества «URL-адреса».

ignore-xml-tags Пропускает проверку качества «XML-разметка».

ignore-xml-invalid Пропускает проверку качества «Синтаксис XML».

ignore-zero-width-space Пропускает проверку качества «Пробел нулевой ширины».

ignore-ellipsis Пропускает проверку качества «Многоточие».

ignore-long-untranslated Пропускает проверку качества «Давно не переводилась».

ignore-multiple-failures Пропускает проверку качества «Множество неудачных проверок».

ignore-unnamed-format Пропускает проверку качества «Несколько переменных без названий».

ignore-optional-plural Пропускает проверку качества «Формы единственного числа».

Примечание: В общем случае правило называется `ignore-*` для любой проверки, где вместо звёздочки подставляется её идентификатор, так что вы можете использовать такое правило даже для своих собственных проверок.

Эти флаги понимаются как в параметрах *компонента*, так и в параметрах исходной строки или в самом файле перевода (например, в файле GNU gettext).

2.12.3 Принудительные проверки

Добавлено в версии 3.11.

Вы можете настроить список проверок, которые нельзя проигнорировать, задав в *конфигурации компонента* параметр *Принудительные проверки*. Все перечисленные проверки не смогут быть проигнорированы из пользовательского интерфейса, а любая строка, для которой такая проверка сработала, помечается флагом *На правку* (смотрите раздел *Состояния перевода*).

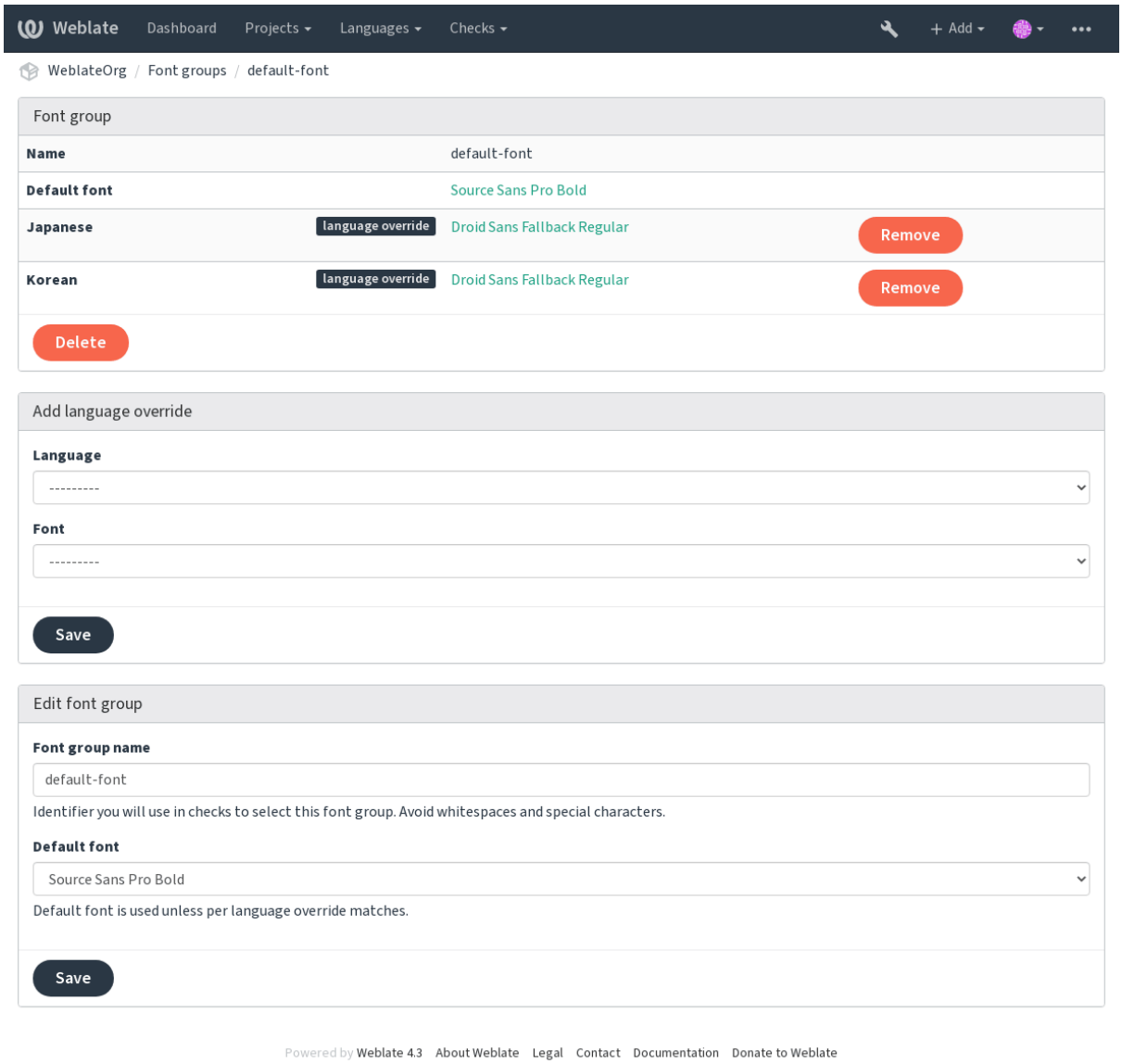
2.12.4 Управление шрифтами

Добавлено в версии 3.7.

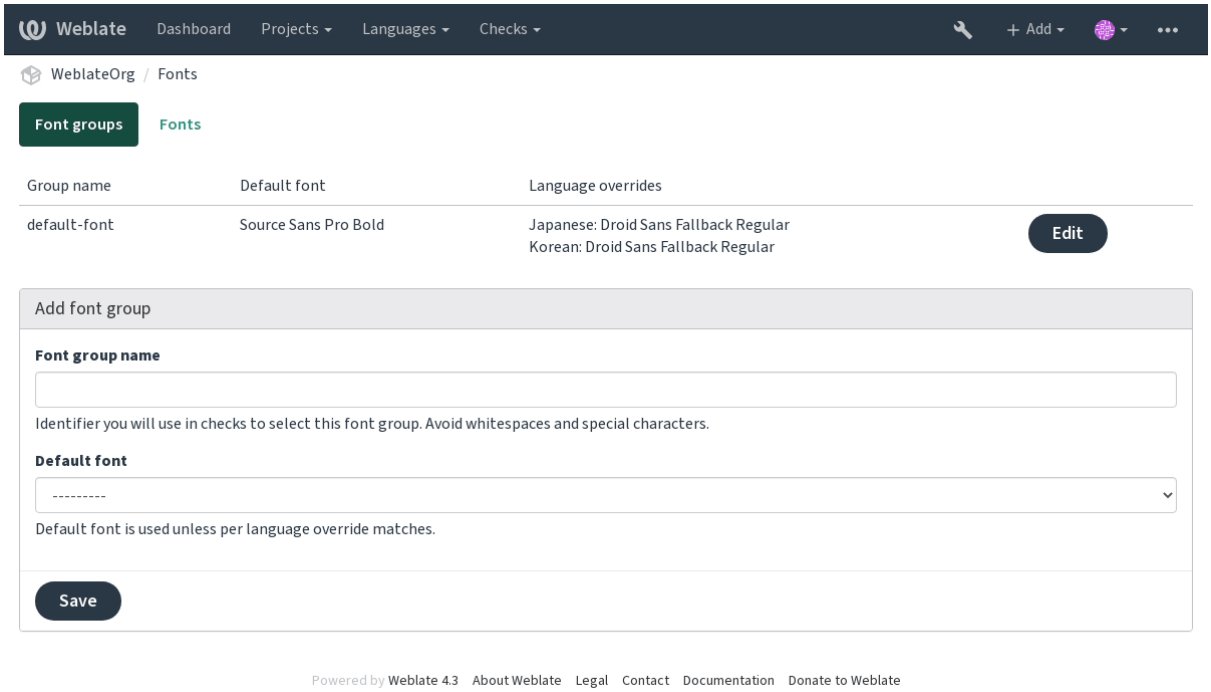
Проверке *Максимальный размер перевода* для вычисления размеров отрисовываемого текста требуется предоставить информацию о шрифте, что можно сделать в Weblate'овском инструменте управления шрифтами, открываемом по ссылке *Шрифты* меню *Управлять* вашего проекта перевода.

В нём можно загружать шрифты TrueType или OpenType, настраивать группы шрифтов и использовать эти группы в проверке.

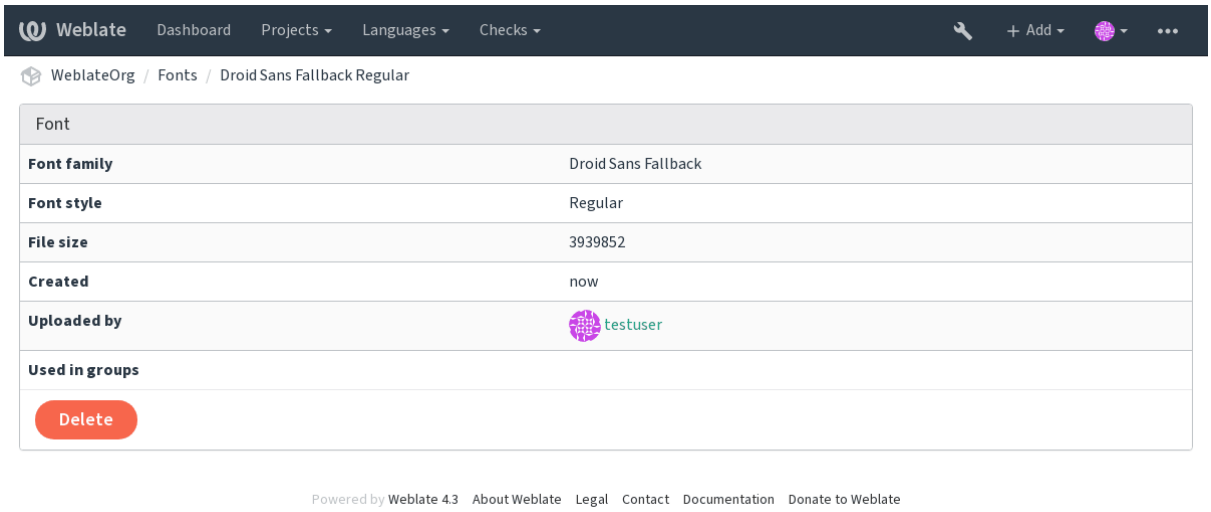
Группы шрифтов позволяют вам определить различные шрифты для разных языков, что обычно необходимо для языков, алфавит которых не использует латиницу:



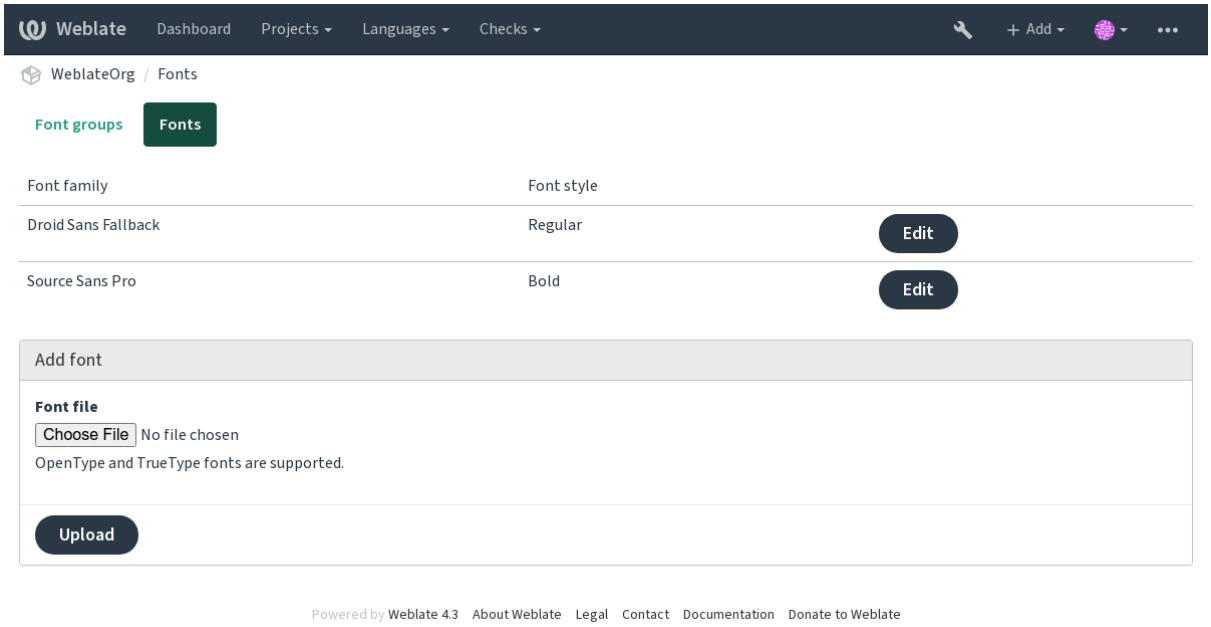
Группы шрифтов идентифицируются по имени, которое не может содержать пробелы или специальные символы, так что их можно легко использовать в определении проверки:



Семейство и начертание шрифта распознаются автоматически после его загрузки:



Вы можете загрузить в Weblate несколько шрифтов:



Чтобы использовать шрифты в проверке длины строки, передайте проверке соответствующие флаги (смотрите раздел *Настройка поведения*). Скорее всего, вам понадобятся следующие из них:

`max-size:500` Определяет максимальную ширину.

`font-family:ubuntu` Определяет используемую группу шрифтов путём указания её идентификатора.

`font-size:22` Определяет размер шрифта.

2.12.5 Написание собственных проверок

Хотя в Weblate встроен широкий спектр проверок качества (смотрите раздел *Проверки качества*), они могут охватывать не всё, что вы хотите проверить. Список выполняемых проверок можно настроить с помощью параметра `CHECK_LIST`, кроме того, вы можете добавить собственные проверки.

1. Создайте подкласс *weblate.checks.Check*
2. Установите несколько атрибутов.
3. Реализуйте либо метод `check` (если вы хотите работать в своём коде с формами множественного числа), либо метод `check_single` (который сам выполняет такую обработку).

Несколько примеров:

Для установки пользовательских проверок необходимо в списке `CHECK_LIST` указать полный путь к классу Python, смотрите раздел *Пользовательские проверки качества, надстройки и автоматические исправления*.

Проверка, что текст перевода не содержит подстроки «foo»

Это довольно простая проверка, которая просто проверяет, не пропущена ли в переводе строка «foo».

```
#
# Copyright © 2012 - 2020 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
```

(continues on next page)

(продолжение с предыдущей страницы)

```
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <https://www.gnu.org/licenses/>.
#
"""Simple quality check example."""

from django.utils.translation import gettext_lazy as _

from weblate.checks.base import TargetCheck


class FooCheck(TargetCheck):

    # Used as identifier for check, should be unique
    # Has to be shorter than 50 characters
    check_id = "foo"

    # Short name used to display failing check
    name = _("Foo check")

    # Description for failing check
    description = _("Your translation is foo")

    # Real check code
    def check_single(self, source, target, unit):
        return "foo" in target
```

Проверка того, что в переводе на чешский язык формы множественного числа отличаются друг от друга

Проверка использует информацию о языке, чтобы проверить, что две формы множественного числа в чешском языке не являются одинаковыми.

```
#
# Copyright © 2012 - 2020 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <https://www.gnu.org/licenses/>.
#
"""Quality check example for Czech plurals."""

from django.utils.translation import gettext_lazy as _
```

(continues on next page)

(продолжение с предыдущей страницы)

```
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 Машинный перевод

В Weblate встроена поддержка нескольких сервисов машинного перевода, которая может быть включена администратором с помощью добавления соответствующих значений в список *MT_SERVICES*. Эти сервисы идут с собственными условиями своего использования, поэтому убедитесь, что вам разрешено использовать их так, как вы хотите.

Исходный язык можно настроить в *конфигурации проекта*.

2.13.1 amaGama

Специальная установка *tmserver*, запущенная авторами Virtaal.

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.tmserver.AmagamaTranslation`.

См.также:

Установка amaGama, Amagama, Память переводов amaGama

2.13.2 Apertium

Свободная платформа ПО машинного перевода, обеспечивающая переводы на ограниченный набор языков.

Рекомендуемый способ использования Apertium — запуск собственного сервера Apertium-APy.

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.apertium.ApertiumAPYTranslation` и заданием параметра *MT_APERTIUM_APY*.

См.также:

MT_APERTIUM_APY, Веб-сайт Apertium, Документация Apertium APy

2.13.3 AWS

Добавлено в версии 3.1.

Amazon Translate — это сервис машинного перевода при помощи нейронных сетей для перевода текста на английский язык и с английского языка на широкое множество поддерживаемых языков.

1. Turn on this service by adding `weblate.machinery.aws.AWSTranslation` to *MT_SERVICES*.
2. Установите модуль *boto3*.
3. Настройте Weblate.

См.также:

MT_AWS_REGION, *MT_AWS_ACCESS_KEY_ID*, *MT_AWS_SECRET_ACCESS_KEY* Документация по Amazon Translate

2.13.4 Машинный перевод через API Baidu

Добавлено в версии 3.2.

Сервис машинного перевода, предоставляемый Baidu.

Этот сервис использует API, и чтобы его использовать, вам нужно получить от Baidu идентификатор и ключ API.

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.baidu.BaiduTranslation` и заданием параметров *MT_BAIDU_ID* и *MT_BAIDU_SECRET*.

См.также:

MT_BAIDU_ID, *MT_BAIDU_SECRET* API переводчика Baidu

2.13.5 DeepL

Добавлено в версии 2.20.

DeepL — это платный сервис, предоставляющий хороший машинный перевод на несколько языков. Вам необходимо приобрести подписку на *DeepL API* или воспользоваться устаревшим планом *DeepL Pro (classic)*.

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.deepl.DeepLTranslation` и заданием параметра *MT_DEEPL_KEY*.

Подсказка: В случае, если у вас есть подписка на CAT-инструменты, вы должны использовать «API v1» вместо «v2», которую по умолчанию использует Weblate (в данном случае это не совсем версия API). Вы можете изменить его с помощью параметра *MT_DEEPL_API_VERSION*.

См.также:

MT_DEEPL_KEY, *MT_DEEPL_API_VERSION*, Веб-сайт DeepL, Цены на DeepL, Документация по API DeepL

2.13.6 Glosbe

Бесплатный словарь и память переводов для почти всех живых языков.

API можно использовать бесплатно, но при условии наличия лицензии на используемый источник данных. Для борьбы со злоупотреблениями существует ограничение на количество вызовов API, которое можно выполнить с одного IP в течение установленного периода времени.

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.glosbe.GlosbeTranslation`.

См.также:

Веб-сайт Glosbe

2.13.7 Google Translate

Сервис машинного перевода, предоставляемый Google.

Этот сервис использует API Переводчика Google, поэтому вам нужно получить ключ API и включить тарифный план в консоли Google API.

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.google.GoogleTranslation` и заданием параметра *MT_GOOGLE_KEY*.

См.также:

MT_GOOGLE_KEY, Документация по переводчику Google

2.13.8 Google Translate API V3 (расширенный)

Сервис машинного перевода, предоставляемый облачными сервисами Google Cloud.

Этот сервис отличается от предыдущего способом аутентификации. Для включения сервиса добавьте в список *MT_SERVICES* значение `weblate.machinery.googlev3.GoogleV3Translation` и задайте параметры

- *MT_GOOGLE_CREDENTIALS*
- *MT_GOOGLE_PROJECT*

Если без *location* у вас не заработает, вам также может потребоваться указать параметр *MT_GOOGLE_LOCATION*.

См.также:

MT_GOOGLE_CREDENTIALS, *MT_GOOGLE_PROJECT*, *MT_GOOGLE_LOCATION* Документация по переводчику Google

2.13.9 Переводчик Microsoft Cognitive Services

Добавлено в версии 2.10.

Сервис машинного перевода, предоставляемый Microsoft на портале Azure в качестве одного из сервисов Cognitive Services.

Weblate реализует API переводчика версии V3.

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.microsoft.MicrosoftCognitiveTranslation` и заданием параметра *MT_MICROSOFT_COGNITIVE_KEY*.

API переводчика текста версии V2

Ключ, который вы используете для API переводчика версии V2, можно использовать с API версии 3.

API переводчика текста версии V3

Вам необходимо зарегистрироваться на портале Azure и использовать полученный там ключ. При использовании новых ключей Azure вам также потребуется установить параметр *MT_MICROSOFT_REGION* в локаль вашего сервиса.

См.также:

MT_MICROSOFT_COGNITIVE_KEY, *MT_MICROSOFT_REGION*, Cognitive Services — API переводчика текста, Портал Microsoft Azure

2.13.10 Microsoft Terminology Service (Терминологическая служба Майкрософт)

Добавлено в версии 2.19.

API терминологической службы Майкрософт позволяет программно получать доступ к терминологии, определениям и строкам пользовательского интерфейса (UI), доступным на языковом портале, через веб-сервис.

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.microsoftterminology.MicrosoftTerminologyService`.

См.также:

API терминологической службы Майкрософт

2.13.11 ModernMT

Добавлено в версии 4.2.

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.modernmt.ModernMTTranslation` и заданием параметра *MT_MODERNMT_KEY*.

См.также:

ModernMT API, *MT_MODERNMT_KEY*, *MT_MODERNMT_URL*

2.13.12 MyMemory

Огромная память переводов с машинным переводом.

Бесплатное анонимное использование в настоящее время ограничено 100 запросами в день, или 1000 запросами в день при указании контактного адреса электронной почты в параметре *MT_MYMEMORY_EMAIL*. Также вы можете запросить у них увеличение лимитов.

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.mymemory.MyMemoryTranslation` и заданием параметра *MT_MYMEMORY_EMAIL*.

См.также:

MT_MYMEMORY_EMAIL, *MT_MYMEMORY_USER*, *MT_MYMEMORY_KEY*, Веб-сайт MyMemory

2.13.13 API машинного перевода NetEase Sight

Добавлено в версии 3.3.

Сервис машинного перевода, предоставляемый Netease.

Этот сервис использует API, поэтому вам необходимо получить от NetEase ключ и секрет.

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.youdao.NeteaseSightTranslation` и заданием параметров *MT_NETEASE_KEY* и *MT_NETEASE_SECRET*.

См.также:

MT_NETEASE_KEY, *MT_NETEASE_SECRET* Платформа перевода Netease Sight

2.13.14 tmserver

Вы можете запустить свой собственный сервер памяти переводов, используя сервер, сходящий в поставку с Translate-toolkit, и настроив общение Weblate'а с ним. Также вы можете использовать его с сервером amaGama, который является расширенной версией tmserver'а.

1. Сперва вам нужно импортировать в память переводов некоторые данные:

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.tmserver.TMServerTranslation`.

```
build_tmdb -d /var/lib/tm/db -s en -t cs locale/cs/LC_MESSAGES/django.po
build_tmdb -d /var/lib/tm/db -s en -t de locale/de/LC_MESSAGES/django.po
build_tmdb -d /var/lib/tm/db -s en -t fr locale/fr/LC_MESSAGES/django.po
```

3. Запустите tmserver, чтобы он слушал ваши запросы:

```
tmserver -d /var/lib/tm/db
```

4. Настройте Weblate для общения с ним:

```
MT_TMSERVER = 'http://localhost:8888/tmserver/'
```

См.также:

MT_TMSERVER, tmserver Установка amaGama, Amagama, Память переводов Amagama

2.13.15 Yandex Translate (Яндекс.Переводчик)

Сервис машинного перевода, предоставляемый Яндексом.

Этот сервис использует API переводчика, поэтому вам необходимо получить от Яндекса ключ API.

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.yandex.YandexTranslation` и заданием параметра *MT_YANDEX_KEY*.

См.также:

MT_YANDEX_KEY, API Яндекс.Переводчика, Яндекс.Переводчик

2.13.16 API машинного перевода Youdao Zhiyun

Добавлено в версии 3.2.

Сервис машинного перевода, предоставляемый Youdao.

Этот сервис использует API, поэтому вам необходимо получить от Youdao идентификатор и ключ API.

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.youdao.YoudaoTranslation` и заданием параметров *MT_YOUDAO_ID* и *MT_YOUDAO_SECRET*.

См.также:

MT_YOUDAO_ID, *MT_YOUDAO_SECRET*, Сервис переводов Youdao Zhiyun с естественных языков

2.13.17 Weblate

Weblate также сам может быть источником машинных переводов. Этот перевод основан на движке полнотекстового поиска Woosh и выдаёт как точные, так и приблизительные соответствия.

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.weblatetm.WeblateTranslation`.

2.13.18 Память переводов Weblate'а

Добавлено в версии 2.20.

Память переводов также может использоваться в качестве источника предложений машинного перевода.

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.machine.WeblateMemory`. Он включён по умолчанию.

2.13.19 SAP Translation Hub

Сервис машинного перевода, предоставляемый SAP.

Для использования этой службы вам необходимо иметь учётную запись SAP (и включить SAP Translation Hub в облачной платформе SAP).

Этот сервис включается добавлением в список *MT_SERVICES* значения `weblate.machinery.saptranslationhub.SAPTranslationHub` и заданием доступа либо к песочнице, либо к рабочему API.

Примечание: Для доступа к API песочницы необходимо задать параметры *MT_SAP_BASE_URL* и *MT_SAP_SANDBOX_APIKEY*.

Для доступа к рабочему API необходимо задать параметры *MT_SAP_BASE_URL*, *MT_SAP_USERNAME* и *MT_SAP_PASSWORD*.

См.также:

MT_SAP_BASE_URL, *MT_SAP_SANDBOX_APIKEY*, *MT_SAP_USERNAME*, *MT_SAP_PASSWORD*, *MT_SAP_USE_MT* SAP Translation Hub API

2.13.20 Пользовательский машинный перевод

Также вы можете реализовать собственные сервисы машинного перевода, написав всего несколько строк кода на Python. В этом примере реализован машинный перевод на фиксированный список языков через модуль Python `dictionary`:

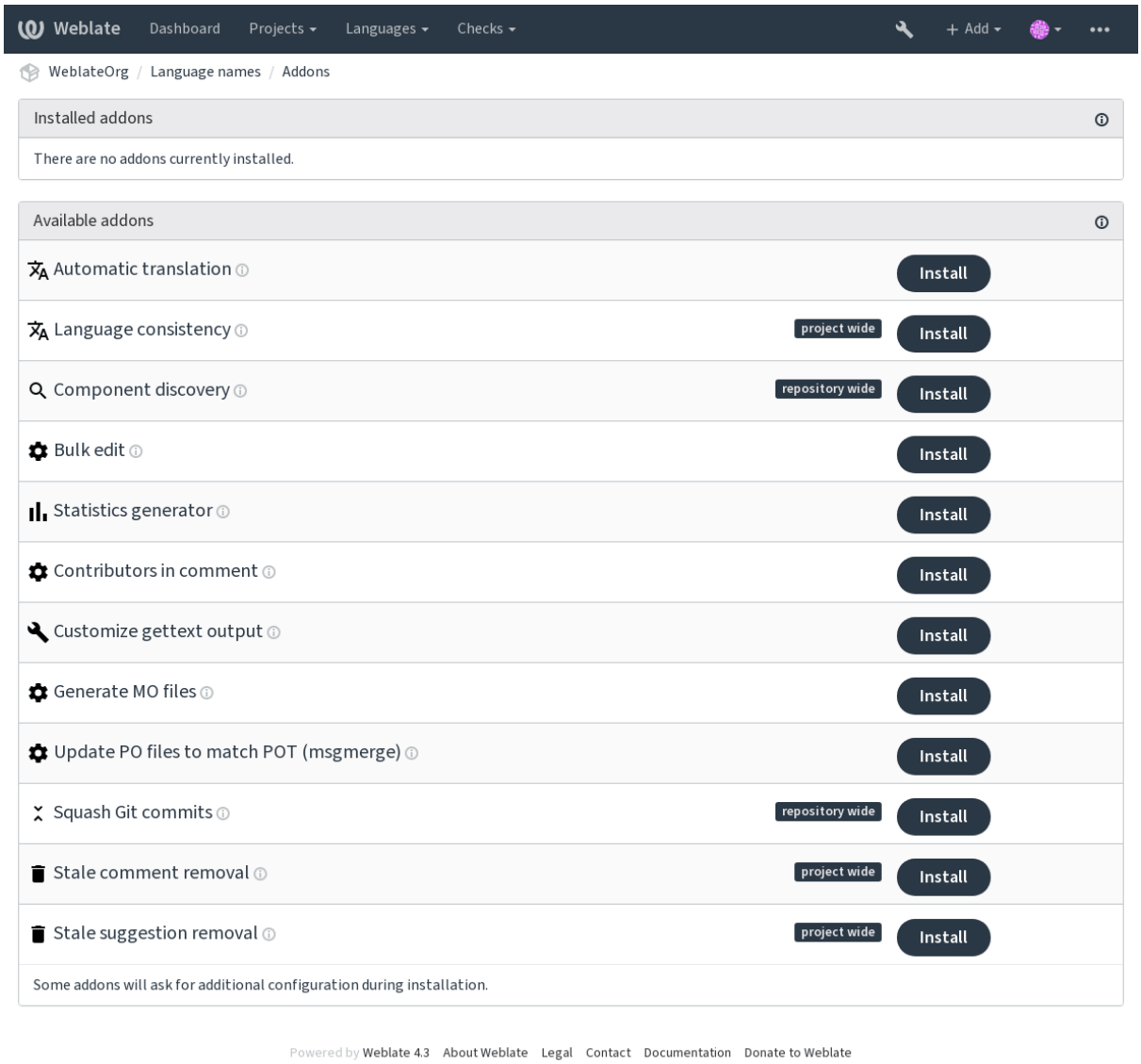
```
#
# Copyright © 2012 - 2020 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <https://www.gnu.org/licenses/>.
#
"""Machine translation example."""

import dictionary

from weblate.machinery.base import MachineTranslation


class SampleTranslation(MachineTranslation):
    """Sample machine translation interface."""

    name = "Sample"

    def download_languages(self):
        """Return list of languages your machine translation supports."""
        return {"cs"}

    def download_translations(self, source, language, text, unit, user, search):
        """Return tuple with translations."""
        for t in dictionary.translate(text):
            yield {"text": t, "quality": 100, "service": self.name, "source": text}
```

Вы можете добавить свой класс в *MT_SERVICES* и Weblate начнёт его использовать.

2.14 Надстройки

Добавлено в версии 2.19.

Надстройки предоставляют способы настройки рабочего процесса перевода. Они могут быть установлены в виде компонента перевода и работать за кулисами. Управление надстройками доступно для администраторов из меню *Управлять* ↓ *Надстройки* соответствующего компонента перевода.



2.14.1 Встроенные надстройки

Автоматический перевод

Добавлено в версии 3.9.

Автоматически переводит строки при помощи машинного перевода или других компонентов.

Эта надстройка срабатывает автоматически при появлении в компоненте новых строк.

См.также:

Автоматический перевод, *Поддержание единого перевода в разных компонентах*

CDN локализации JavaScript'а

Добавлено в версии 4.2.

Добавляет CDN локализации для локализации JavaScript'а или HTML'я.

Его можно использовать для локализации статических HTML-страниц или для загрузки локализации в код JavaScript.

После установки надстройка генерирует для вашего компонента уникальный URL-адрес, который вы можете включить в HTML-документы для их локализации. Подробности смотрите в разделе *Перевод HTML и JavaScript с помощью сети доставки содержимого Weblate*.

См.также:

Настройка надстройки Weblate CDN, *Перевод HTML и JavaScript с помощью сети доставки содержимого Weblate*, *Извлечение строк для Weblate CDN*, *Локализация HTML с помощью Weblate CDN*

Очистка файлов перевода

Обновляет все файлы переводов до соответствия одноязычному базовому файлу. Для большинства форматов файлов это означает удаление устаревших ключей переводов, которых больше нет в базовом файле.

Языковая согласованность

Гарантирует, что все компоненты в рамках одного проекта имеют переводы на все языки проекта.

Она создаёт пустые переводы на языках, в которые компоненты не добавлены.

Отсутствующие языки проверяются каждые 24 часа и при добавлении нового языка в Weblate.

В отличие от большинства других надстроек, эта влияет на весь проект.

Подсказка: Автоматически переводите новые добавленные строки с помощью надстройки *Автоматический перевод*.

Обнаружение компонента

Автоматически добавляет или удаляет компоненты проекта, основываясь на изменениях файлов в системе контроля версий.

Она срабатывает при каждом обновлении репозитория системы контроля версий, а в остальном её работа аналогична команде управления `import_project`. Таким образом, внутри одной системы контроля версий вы можете отслеживать несколько компонентов перевода.

Создайте один основной компонент, который в будущем исчезнет с наименьшей вероятностью; другие компоненты в своей настройке адресов репозитория системы контроля версий будут использовать ведущие на него *внутренние адреса Weblate'а*; настройте надстройку так, чтобы она нашла все компоненты.

Сопоставление осуществляется с помощью регулярных выражений, сложность конфигурации которых компенсируется их мощью. Для распространённых случаев в справочном разделе надстройки можно найти ряд примеров.

Когда вы нажмёте *Сохранить*, надстройка предварительно покажет найденные компоненты, так что вы сможете проверить, действительно ли конфигурация соответствует вашим потребностям:

Weblate Dashboard Projects Languages Checks + Add

WeblateOrg / Language names / Addons / Component discovery

Configure addon

- Please review and confirm the matched components.

Component	Matched files
Following components would be created	
Djangojs	weblate/locale/he/LC_MESSAGES/djangojs.po (he) weblate/locale/hu/LC_MESSAGES/djangojs.po (hu) weblate/locale/cs/LC_MESSAGES/djangojs.po (cs)
Django	weblate/locale/he/LC_MESSAGES/django.po (he) weblate/locale/hu/LC_MESSAGES/django.po (hu) weblate/locale/cs/LC_MESSAGES/django.po (cs)

☐ **I confirm the above matches look correct**

Regular expression to match translation files against

weblate/locale/(?P<language>[^/]*)/LC_MESSAGES/(?P<component>[^/]*)\.po

File format

gettext PO file

Customize the component name

{{ component|title }}

Define the monolingual base filename

Leave empty for bilingual translation files.

Define the base file for new translations

weblate/locale/{{ component }}.pot

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

Language filter

^(cs|he|hu)$

Regular expression to filter translation against when scanning for filemask.

☑ **Clone addons from the main component to the newly created ones**

☐ **Remove components for inexistant files**

The regular expression to match translation files has to contain two named groups to match component and language, some examples:

Regular expression	Example matched files	Description
(?P<language>[^/.]*)/(?P<component>[^/]*)\.po	cs/application.po cs/website.po de/application.po de/website.po	One folder per language containing translation files for components.
locale/(?P<language>[^/.]*)/LC_MESSAGES/(?P<component>[^/]*)\.po	locale/cs/LC_MESSAGES/application.po locale/cs/LC_MESSAGES/website.po locale/de/LC_MESSAGES/application.po locale/de/LC_MESSAGES/website.po	Usual structure for storing gettext PO files.
src/locale/(?P<component>[^/]*)\.(?P<language>[^/.]*)\.po	src/locale/application.cs.po src/locale/website.cs.po src/locale/application.de.po src/locale/website.de.po	Using both component and language name within filename.
locale/(?P<language>[^/.]*)/(?P<component>[^/]*)/(?P=language)\.po	locale/cs/application/cs.po locale/cs/website/cs.po locale/de/application/de.po locale/de/website/de.po	Using language in both path and filename.
res/values-(?P<language>[^/.]*)/strings-(?P<component>[^/]*)\.xml	res/values-cs/strings-about.xml res/values-cs/strings-help.xml res/values-de/strings-about.xml res/values-de/strings-help.xml	Android resource strings, split into several files.

You can use Django template markup in both component name and the monolingual base filename, for example:

`{{ component }}`
Component filename match
`{{ component|title }}`
Component filename with upper case first letter

Save

Powered by Weblate 4.3 About Weblate Legal Contact Documentation Donate to Weblate

См.также:

Шаблонная разметка

Массовая правка

Добавлено в версии 3.11.

Позволяет массово править флаги, метки или состояния для строк.

Автоматическая пометка новых строк может очень пригодиться (начните с поискового запроса `NOT has:label` и добавляйте нужные метки до тех пор, пока все строки не будут правильно помечены). Также вы можете выполнять любые другие автоматические операции с метаданными Weblate.

См.также:

Массовая правка

Пометка неизменённых переводов флагом «На правку»

Добавлено в версии 3.1.

При импорте строки для перевода из системы контроля версий и при её совпадении с исходной строкой она помечается в Weblate флагом «На правку». Особенно это полезно для тех форматов файлов, которые включают в себя все строки, даже если те не переведены.

Пометка новых исходных строк флагом «На правку»

При импорте новой исходной строки из системы контроля версий она помечается в Weblate флагом «На правку». Таким образом, вы можете легко фильтровать и редактировать исходные строки, добавленные разработчиками.

Пометка новых переводов флагом «На правку»

При импорте строки для перевода из системы контроля версий она помечается в Weblate флагом «На правку». Таким образом, вы можете легко фильтровать и редактировать переводы, созданные разработчиками.

Генератор статистики

Генерирует файл, содержащий детальную информацию о переводе.

Для имени файла и его содержимого вы можете использовать шаблоны Django, подробное описание разметки шаблона смотрите в разделе *Шаблонная разметка*.

К примеру, сгенерировать файл со сводкой по каждому переводу можно со следующими настройками:

Имя сгенерированного файла `locale/{{ language_code }}.json`

Содержимое

```
{
   "language": "{{ language_code }}",
   "strings": "{{ stats.all }}",
   "translated": "{{ stats.translated }}",
   "last_changed": "{{ stats.last_changed }}",
   "last_author": "{{ stats.last_author }}",
}
```

См.также:

Шаблонная разметка

Участники в комментарии

Обновляет комментарий в заголовке PO-файла, включая в него имена участников и годы их участия.

Заголовок PO-файла будет содержать список участников и годы их участия:

```
# Michal Čihař <michal@cihar.com>, 2012, 2018, 2019, 2020.
# Pavel Borecki <pavel@example.com>, 2018, 2019.
# Filip Hron <filip@example.com>, 2018, 2019.
# anonymous <noreply@weblate.org>, 2019.
```

Обновление переменной ALL_LINGUAS в файле «configure»

Обновляет переменную ALL_LINGUAS в файлах `configure`, `configure.in` и `configure.ac` при добавлении нового перевода.

Настройка вывода gettext

Позволяет настроить поведение вывода gettext, например, заворот строк.

Она предлагает следующие варианты:

- Переносить строки на 77-м символе и у символов перевода строки
- Переносить строки только у символов перевода строки
- Не переносить строки

Примечание: По умолчанию gettext переносит строки на 77-м символе и у символов перевода строки. С параметром `--no-wrap` он переносит их только у символов перевода строки.

Обновление файла LINGUAS

Обновляет файл LINGUAS при добавлении нового перевода.

Генерирование MO-файлов

Автоматически генерирует MO-файл для каждого изменённого PO-файла.

Обновлять PO-файлы до соответствия POT (msgmerge)

Обновляет все PO-файлы, используя msgmerge, приводя их в соответствие с POT-файлом. Это происходит всякий раз, когда новые изменения извлекаются из вышестоящего репозитория.

Склеивание Git-коммитов

Склеивает Git-коммиты до отправки изменений.

Вы можете выбрать один из следующих режимов:

Добавлено в версии 3.4.

- Все коммиты в один
- По языку
- По файлу

Добавлено в версии 3.5.

- По автору

Исходные сообщения коммита сохраняются, но авторство коммитов теряется, если только не выбран режим *По автору*, или сообщение коммита не настроено на включение в него его авторов.

Добавлено в версии 4.1.

Исходные сообщения коммитов могут быть переопределены пользовательским сообщением коммита.

Дополнительно из исходных сообщений коммитов могут быть удалены завершители (строки коммита вида `Co-authored-by: ...`) с последующим их присоединением к концу сообщения склеенного коммита. Также в этом режиме для каждого переводчика генерируется соответствующий завершитель `Co-authored-by:`.

Настройка вывода в JSON

Позволяет подправить поведение вывода в JSON, например, отступы или сортировку ключей.

Форматирование файла свойств Java

Сортирует файл свойств Java.

Удаление устаревших комментариев

Добавлено в версии 3.7.

Устанавливает сроки удаления комментариев.

Может быть полезна для удаления старых комментариев, которые могли потерять актуальность. Используйте её с осторожностью, поскольку то, что комментарий старый, не означает, что он потерял свою значимость.

Удаление устаревших предложений

Добавлено в версии 3.7.

Устанавливает сроки удаления предложений.

В сочетании с голосованием по предложениям (смотрите раздел *Коллегиальное рецензирование*) может быть очень полезна для удаления предложений, которые не получили достаточного количества положительных голосов в заданный промежуток времени.

Обновление файлов RESX

Добавлено в версии 3.9.

Обновляет все файлы перевода, приводя их в соответствие с одноязычным исходным базовым файлом. Неиспользуемые строки удаляются, а новые добавляются как копии исходных строк.

Подсказка: Если вы хотите только удалить устаревшие ключи перевода, используйте надстройку *Очистка файлов перевода*.

Настройка вывода в YAML

Добавлено в версии 3.10.2.

Позволяет подправить поведение вывода в YAML, например, длину строк и символы перевода строк.

2.14.2 Настройка списка надстроек

Список надстроек настраивается параметром *WEBLATE_ADDONS*. Для добавления ещё одной надстройки просто включите в этот параметр полное имя класса.

2.14.3 Написание надстройки

Также вы можете написать свою собственную надстройку, всё, что вам для этого нужно — это создать подкласс `BaseAddon`, определить метаданные надстройки и реализовать обратный вызов, который будет выполнять обработку.

Вот пример надстройки:

```
#
# Copyright © 2012 - 2020 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <https://www.gnu.org/licenses/>.
#


from django.utils.translation import gettext_lazy as _

from weblate.addons.base import BaseAddon
from weblate.addons.events import EVENT_PRE_COMMIT


class ExampleAddon(BaseAddon):
    # Filter for compatible components, every key is
    # matched against property of component
    compat = {"file_format": {"po", "po-mono"}}
    # List of events addon should receive
    events = (EVENT_PRE_COMMIT,)
    # Addon unique identifier
    name = "weblate.example.example"
    # Verbose name shown in the user interface
    verbose = _("Example addon")
    # Detailed addon description
    description = _("This addon does nothing it is just an example.")

    # Callback to implement custom behavior
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    def pre_commit(self, translation, author):
        return
```

2.14.4 Выполнение скриптов из надстройки

Надстройки также могут использоваться для выполнения внешних скриптов. Раньше выполнение было интегрировано в сам Weblate, но теперь вам нужно написать код для заворачивания вашего скрипта в надстройку.

```
#
# Copyright © 2012 - 2020 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <https://www.gnu.org/licenses/>.
#
"""Example pre commit script."""


from django.utils.translation import gettext_lazy as _

from weblate.addons.events import EVENT_PRE_COMMIT
from weblate.addons.scripts import BaseScriptAddon


class ExamplePreAddon(BaseScriptAddon):
    # Event used to trigger the script
    events = (EVENT_PRE_COMMIT,)
    # Name of the addon, has to be unique
    name = "weblate.example.pre"
    # Verbose name and long descrption
    verbose = _("Execute script before commit")
    description = _("This addon executes a script.")

    # Script to execute
    script = "/bin/true"
    # File to add in commit (for pre commit event)
    # does not have to be set
    add_file = "po/{{ language_code }}.po"
```

Инструкции по установке смотрите в разделе *Пользовательские проверки качества, надстройки и автоматические исправления*.

Скрипт выполняется с текущим каталогом, установленном в корневой каталог репозитория системы контроля версий заданного компонента.

Кроме того, в скрипте доступны следующие переменные окружения:

WL_VCS
Используемая система контроля версий.

WL_REPO
URL-адрес вышестоящего репозитория.

WL_PATH
Абсолютный путь к репозиторию системы контроля версий.

WL_BRANCH
Добавлено в версии 2.11.

Ветка репозитория, настроенная для текущего компонента.

WL_FILEMASK
Маска файлов для текущего компонента.

WL_TEMPLATE
Имя файла шаблона для одноязычных переводов (может быть пустым).

WL_NEW_BASE
Добавлено в версии 2.14.

Имя файла, используемого для создания новых переводов (может быть пустым).

WL_FILE_FORMAT
Формат файла, используемый в текущем компоненте.

WL_LANGUAGE
Язык обрабатываемого в настоящее время перевода (недоступен для обработчиков уровня компонента).

WL_PREVIOUS_HEAD
Предыдущий HEAD при обновлении (доступна только при запуске обработчика «после обновления»).

WL_COMPONENT_SLUG
Добавлено в версии 3.9.

Плашка компонента, используемая для построения URL-адреса.

WL_PROJECT_SLUG
Добавлено в версии 3.9.

Плашка проекта, используемая для построения URL-адреса.

WL_COMPONENT_NAME
Добавлено в версии 3.9.

Название компонента.

WL_PROJECT_NAME
Добавлено в версии 3.9.

Название проекта.

WL_COMPONENT_URL
Добавлено в версии 3.9.

URL-адрес компонента.

WL_ENGAGE_URL
Добавлено в версии 3.9.

URL-адрес для присоединения к переводу проекта.

См.также:

Конфигурация компонента

Обработка репозитория после обновления

Обработка репозитория после обновления может быть использоваться для обновления файлов перевода при изменении исходных файлов в вышестоящем репозитории. При этом помните, что Weblate видит только те файлы, которые были закоммичены систему контроля версий, поэтому в скрипте вам необходимо закоммитить все изменения.

Например, при использовании Gulp это можно сделать следующим кодом:

```
#! /bin/sh
gulp --gulpfile gulp-i18n-extract.js
git commit -m 'Update source strings' src/languages/en.lang.json
```

Обработка переводов до коммита

Используйте скрипт коммита для автоматического внесения изменений в перевод до их коммита в репозиторий.

Перевод передается в виде единственного параметра, содержащего имя файла текущего перевода.

2.15 Память переводов

Добавлено в версии 2.20.

Weblate поставляется со встроенной памятью переводов, состоящей из следующих частей:

- Вручную импортированная память переводов (смотрите раздел *Пользовательский интерфейс*).
- Автоматически сохранённые переводы, выполненные в Weblate'е (зависят от *областей действия памяти переводов*).
- Автоматически импортированные прошлые переводы.

Содержимое памяти переводов может быть применено одним из двух способов:

- Вручную во время перевода, на вкладке *«Автоматизированный перевод»*.
- Автоматически, путём перевода строк с помощью инструмента *Автоматический перевод* или надстройки *Автоматический перевод*.

Подсказки по установке смотрите в *памяти переводов Weblate'а*, которая включена по умолчанию.

2.15.1 Области действия памяти переводов

Добавлено в версии 3.2: В более ранних версиях память переводов могла быть загружена только из файла, соответствующего области действия текущей импортированной памяти переводов.

Области действия памяти переводов позволяют обеспечить как конфиденциальность, так и совместное использование переводов в соответствии с желаемым поведением.

Импортированная память переводов

Импорт произвольных данных памяти переводов с помощью команды `import_memory` делает содержимое памяти переводов доступным для всех пользователей и проектов.

Память переводов пользователя

Автоматически сохраняет все переводы пользователя в персональной памяти переводов соответствующего пользователя.

Память переводов проекта

Все переводы в рамках проекта автоматически сохраняются в памяти переводов проекта, доступной только для этого проекта.

Совместная память переводов

Все переводы в проектах с включённой совместной памятью переводов сохраняются в совместной памяти переводов, доступной для всех проектов.

Пожалуйста, внимательно подумайте, следует ли включать эту функцию для совместно используемых установок Weblate, поскольку это может иметь серьёзные последствия:

- Переводы могут быть использованы кем угодно.
- Это может привести к разглашению секретной информации.

2.15.2 Управление памятью переводов

Пользовательский интерфейс

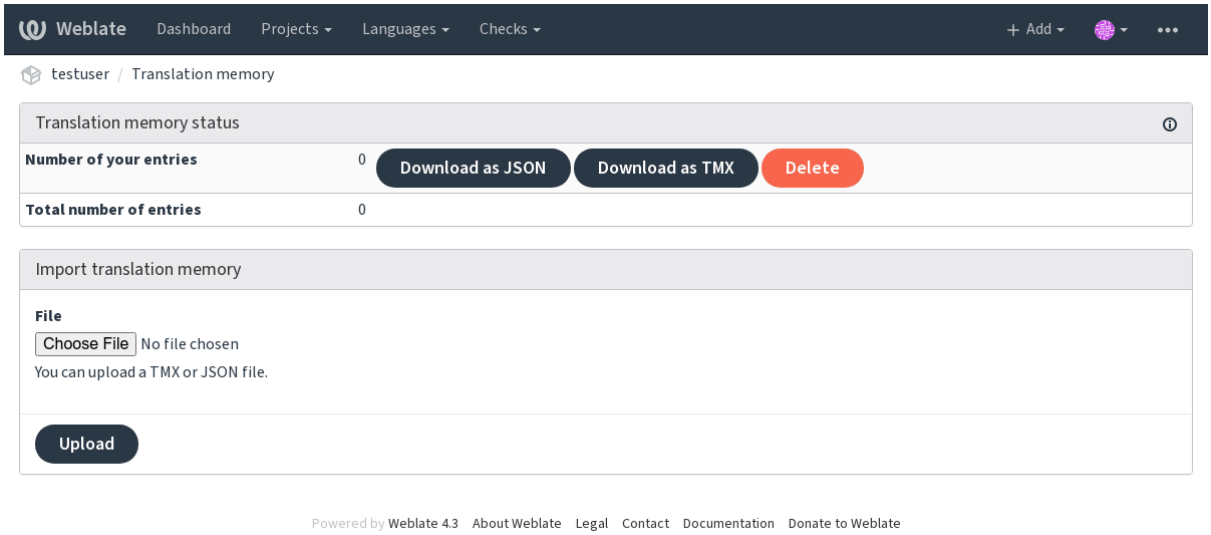
Добавлено в версии 3.2.

В базовом пользовательском интерфейсе вы можете управлять памятью переводов пользователя и проекта. Через него вы можете загружать, стирать или импортировать память переводов.

Подсказка: В Weblate может быть импортирована память переводов в формате JSON, формат TMX предусмотрен для взаимодействия с другими инструментами.

См.также:

Схема памяти переводов Weblate'а



Интерфейс управления

Для управления содержимым памяти переводов существует несколько команд управления. Они работают с памятью переводов целиком, не применяя фильтр по областям (если только этот фильтр не запрошен параметрами):

dump_memory Экспортирует память переводов в виде JSON'а

import_memory Импортирует файлы TMX или JSON в память переводов

2.16 Конфигурация

Все настройки хранятся в файле `settings.py` (как обычно для проектов Django).

Примечание: После изменения любого из этих параметров вам необходимо перезапустить Weblate — как процесс WSGI, так и процесс Celery.

В случае, если Weblate работает через mod_wsgi, для перезагрузки конфигурации вам нужно перезапустить Apache.

См.также:

Также, пожалуйста, ознакомьтесь с документацией Django по настройке параметров самого Django.

2.16.1 AKISMET_API_KEY

Weblate может использовать Akismet для проверки входящих анонимных предложений на спам. Посетите akismet.com, чтобы приобрести ключ API и связать его с сайтом.

2.16.2 ANONYMOUS_USER_NAME

Имена пользователей, которые не входят в систему.

См.также:

Управление доступом

2.16.3 AUDITLOG_EXPIRY

Добавлено в версии 3.6.

Сколько дней Weblate должен хранить журналы аудита, в которых содержится информация о деятельности учётной записи.

По умолчанию установлен в 180 дней.

2.16.4 AUTH_LOCK_ATTEMPTS

Добавлено в версии 2.14.

Максимальное количество неудачных попыток аутентификации до применения ограничения частоты запросов.

В настоящее время оно применяется в следующих местах:

- Страница входа. Удаляет пароль учётной записи, не позволяя пользователю войти в систему без запроса нового пароля.
- Страница сброса пароля. Предотвращает отправку новых писем, избегая рассылки спама пользователям со слишком большим количеством попыток сброса пароля.

По умолчанию установлен в 10.

См.также:

Ограничение частоты запросов,

2.16.5 AUTO_UPDATE

Добавлено в версии 3.2.

Изменено в версии 3.11: Первоначальные возможные значения параметра включено/выключено были изменены для дифференциации принимаемых строк.

Обновление всех репозиториев на ежедневной основе.

Подсказка: Полезно, если вы не используете *обработчики* для автоматического обновления Weblate'овских репозиториев.

Примечание: Варианты включено/выключено в дополнение к строкам существуют для обратной совместимости.

Варианты:

`"none"` Никаких ежедневных обновлений.

`"remote"` или `False` Только вытягиваются изменения из удалённых репозиториев.

`"full"` или `True` Вытягиваются изменения из удалённых репозиториев и сливаются с рабочей копией.

Примечание: Для включения этого параметра необходима работа *фоновых задач Celery*, также он вступит в силу после их перезапуска.

2.16.6 AVATAR_URL_PREFIX

Префикс для генерирования URL-адресов аватаров в выражении `${AVATAR_URL_PREFIX}/avatar/${MAIL_HASH}?${PARAMS}`. Сервисы аватаров работают следующим образом:

Gravatar (по умолчанию), согласно https://gravatar.com/ `AVATAR_URL_PREFIX = 'https://www.gravatar.com/'`

Libravatar, согласно https://www.libravatar.org/ `AVATAR_URL_PREFIX = 'https://www.libravatar.org/'`

См.также:

Кеширование аватара, *ENABLE_AVATARS*, *Аватары*

2.16.7 AUTH_TOKEN_VALID

Добавлено в версии 2.14.

Сколько времени действует токен аутентификации и временный пароль из писем сброса пароля. Установливается в количество секунд, по умолчанию равен 172800 (2 дня).

2.16.8 AUTH_PASSWORD_DAYS

Добавлено в версии 2.15.

Сколько дней должно быть разрешено использовать один и тот же пароль.

Примечание: Смена паролей, сделанная в версиях Weblate до версии 2.15, в этой политике учтена не будет.

По умолчанию установлен в 180 дней.

2.16.9 AUTOFIX_LIST

Список автоматических исправлений, применяемых при сохранении строки.

Примечание: Предоставьте полный путь к классу Python, реализующему интерфейс автоматического исправления.

Доступные исправления:

`weblate.trans.autofixes.whitespace.SameBookendingWhitespace` Выравнивает количество начальных и конечных пробельных символов в переводе с исходной строкой.

`weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis` Заменяет конечные точки (…) на многоточие (…), если оно есть в исходной строке.

`weblate.trans.autofixes.chars.RemoveZeroSpace` Удаляет пробельные символы нулевой ширины, если исходная строка их не содержит.

`weblate.trans.autofixes.chars.RemoveControlChars` Удаляет управляющие символы, если исходная строка их не содержит.

weblate.trans.autofixes.html.BleachHTML Удаляет небезопасную HTML-разметку из строк, помеченных флагом `safe-html` (смотрите проверку *Небезопасный HTML-код*).

Вы можете выбрать, какие исправления использовать:

```
AUTOFIX_LIST = (
    'weblate.trans.autofixes.whitespace.SameBookendingWhitespace',
    'weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis',
)
```

См.также:

Автоматические исправления, *Пользовательские автоматические исправления*

2.16.10 BASE_DIR

Базовый каталог, в котором находятся исходники Weblate'а. Используется для построения нескольких других умолчательных путей:

- *DATA_DIR*

Значение по умолчанию: каталог верхнего уровня исходников Weblate'а.

2.16.11 CSP_SCRIPT_SRC, CSP_IMG_SRC, CSP_CONNECT_SRC, CSP_STYLE_SRC, CSP_FONT_SRC

Настраивает заголовок `Content-Security-Policy` для Weblate'а. Заголовок автоматически генерируется на основе включённых интеграций со сторонними сервисами (Matomo, Google Analytics, Sentry, …).

Все они по умолчанию установлены в пустой список.

** Пример:: **

```
# Enable Cloudflare Javascript optimizations
CSP_SCRIPT_SRC = ["ajax.cloudflare.com"]
```

См.также:

Политика безопасности содержимого, Политика безопасности содержимого (Content Security Policy — CSP)

2.16.12 CHECK_LIST

Список выполняемых для перевода проверок качества.

Примечание: Предоставьте полный путь к классу Python, реализующему интерфейс проверки.

Подправьте список проверок так, чтобы включить в него только нужные вам.

По умолчанию включены все встроенные *проверки*, оттуда вы можете изменить эти настройки. В *примере конфигурации* они по умолчанию закомментированы, так что используются значения по умолчанию. В каждой новой версии Weblate'а туда добавляются новые проверки, если таковые появляются.

Вы можете отключить все проверки:

```
CHECK_LIST = ()
```

Можете включить только некоторые:

```
CHECK_LIST = (
    'weblate.checks.chars.BeginNewlineCheck',
    'weblate.checks.chars.EndNewlineCheck',
    'weblate.checks.chars.MaxLengthCheck',
)
```

Примечание: Изменение этого параметра влияет только на недавно изменённые переводы, результаты существующих проверок будут по-прежнему сохранены в базе данных. Чтобы также применить изменения к сохранённым переводам, запустите команду `updatechecks`.

См.также:

Проверки качества, *Настройка поведения*

2.16.13 COMMENT_CLEANUP_DAYS

Добавлено в версии 3.6.

Удаление комментариев через заданное количество дней. По умолчанию `None`, что означает, что комментарии не удаляются.

2.16.14 COMMIT_PENDING_HOURS

Добавлено в версии 2.10.

Количество часов до коммита ожидающих изменений фоновой задачей.

См.также:

Конфигурация компонента, *Возраст изменений для коммита*, *Выполнение задач технического обслуживания*, `commit_pending`

2.16.15 DATA_DIR

Каталог, в котором Weblate хранит все свои данные. Он содержит ссылки на репозитории систем контроля версий, индекс для полнотекстового поиска и различные конфигурационные файлы для внешних инструментов.

Обычно существуют следующие подкаталоги:

home Домашний каталог, используемый при вызове скриптов.

ssh Ключи и конфигурация SSH.

static Умолчательное расположение статических файлов Django, определяемых переменной `STATIC_ROOT`.

media Умолчательное расположение медиа-файлов Django, определяемых переменной `MEDIA_ROOT`.

vcs Репозитории систем контроля версий.

backups Данные ежедневных резервных копий, для получения подробной информации обратитесь к разделу *Дампы данных для резервного копирования*.

Примечание: Этот каталог должен быть доступен Weblate'у для записи. Запуск его в качестве приложения uWSGI означает, что пользователь `www-data` должен иметь к нему доступ с правами на запись.

Простейший способ этого достичь — сделать пользователя владельцем каталога:

```
sudo chown www-data:www-data -R $DATA_DIR
```

По умолчанию установлен в `$BASE_DIR/data`.

См.также:

BASE_DIR, *Резервное копирование и перенос Weblate*

2.16.16 DATABASE_BACKUP

Добавлено в версии 3.1.

Должны ли резервные копии базы данных храниться в виде простого текста, сжатого файла или не храниться вовсе. Допустимые значения:

- `"plain"`
- `"compressed"`
- `"none"`

См.также:

Резервное копирование и перенос Weblate

2.16.17 DEFAULT_ACCESS_CONTROL

Добавлено в версии 3.3.

Умолчательное значение параметра управления доступом для новых проектов:

`0` *Публичный*

`1` *Защищённый*

`100` *Частный*

`200` *Настраиваемый*

Используйте *Настраиваемый*, если списками контроля доступа вы управляете вручную, что означает, что вы не полагаетесь на внутреннее Weblate'овское управление ими.

См.также:

По-проектный контроль доступа, *Управление доступом*, *Управление доступом*

2.16.18 DEFAULT_RESTRICTED_COMPONENT

Добавлено в версии 4.1.

Значение по умолчанию для ограничения компонента.

См.также:

По-проектный контроль доступа, *Ограниченный доступ*, *Управление доступом*

2.16.19 DEFAULT_ADD_MESSAGE, DEFAULT_ADDON_MESSAGE, DEFAULT_COMMIT_MESSAGE, DEFAULT_DELETE_MESSAGE, DEFAULT_MERGE_MESSAGE

Умолчательные сообщения коммита для различных операций, для получения подробной информации обратитесь к разделу *Конфигурация компонента*.

См.также:

Шаблонная разметка, *Конфигурация компонента*, *Сообщения коммита, добавления, удаления, слияния и сообщения надстроек*

2.16.20 DEFAULT_ADDONS

Умолчательные надстройки, устанавливаемые в каждый созданный компонент.

Примечание: Этот параметр влияет только на новые создаваемые компоненты.

Пример:

```
DEFAULT_ADDONS = {
     # Addon with no parameters
     "weblate.flags.target_edit": {},

     # Addon with parameters
     "weblate.autotranslate.autotranslate": {
         "mode": "suggest",
         "filter_type": "todo",
         "auto_source": "mt",
         "component": "",
         "engines": ["weblate-translation-memory"],
         "threshold": "80",
     }
}
```

См.также:

`install_addon`

2.16.21 DEFAULT_COMMITER_EMAIL

Добавлено в версии 2.4.

Адрес электронной почты коммиттера для созданных компонентов перевода, по умолчанию равен `noreply@weblate.org`.

См.также:

`DEFAULT_COMMITER_NAME`, *Конфигурация компонента*, *E-mail делающего коммит*

2.16.22 DEFAULT_COMMITER_NAME

Добавлено в версии 2.4.

Имя коммиттера для создаваемых компонентов перевода, по умолчанию равен `Weblate`.

См.также:

`DEFAULT_COMMITER_EMAIL`, *Конфигурация компонента*, *Имя произведшего коммит*

2.16.23 DEFAULT_MERGE_STYLE

Добавлено в версии 3.4.

Стиль слияния для любых новых компонентов.

- *rebase* — по умолчанию
- *merge*

См.также:

Конфигурация компонента, *Стиль слияния*

2.16.24 DEFAULT_TRANSLATION_PROPAGATION

Добавлено в версии 2.5.

Значение по умолчанию для настройки распространения перевода, по умолчанию установлен в `True`.

См.также:

Конфигурация компонента, *Разрешить распространение перевода*

2.16.25 DEFAULT_PULL_MESSAGE

Заголовок для новых запросов на извлечение, по умолчанию равен `'Update from Weblate'`.

2.16.26 ENABLE_AVATARS

Включить ли для пользователей Gravatar-аватары. По умолчанию они включены.

Аватары извлекаются и кэшируются на сервере, что снижает риск утечки персональной информации и ускоряет взаимодействие с пользователем.

См.также:

Кеширование аватара, *`AVATAR_URL_PREFIX`*, *Аватары*

2.16.27 ENABLE_HOOKS

Включить ли анонимные удалённые обработчики.

См.также:

Обработчики уведомлений

2.16.28 ENABLE_HTTPS

Отправлять ли ссылки на Weblate по протоколу HTTPS или HTTP. Этот параметр влияет на отправленные сообщения электронной почты и сгенерированные абсолютные URL-адреса.

Подсказка: В конфигурации по умолчанию он также используется для нескольких связанных с HTTPS параметров Django.

См.также:

`SESSION_COOKIE_SECURE`, `CSRF_COOKIE_SECURE`, `SECURE_SSL_REDIRECT`, *Установка правильного домена сайта*

2.16.29 ENABLE_SHARING

Включает/выключает меню *Поделиться*, с помощью которого пользователи могут делиться прогрессом перевода в социальных сетях.

2.16.30 GITLAB_CREDENTIALS

Добавлено в версии 4.3.

Список учётных данных для серверов GitLab'a.

Подсказка: Используйте его в случае, если вы хотите, чтобы для одной конечной точки GitLab'a, определяемой параметрами `GITLAB_USERNAME` и `GITLAB_TOKEN`, Weblate взаимодействовал с несколькими серверами.

```
GITLAB_CREDENTIALS = {
    "gitlab.com": {
        "username": "weblate",
        "token": "your-api-token",
    },
    "gitlab.example.com": {
        "username": "weblate",
        "token": "another-api-token",
    },
}
```

2.16.31 GITLAB_USERNAME

Имя пользователя GitLab, используемое для отправки запросов на слияние для обновления перевода.

См.также:

`GITLAB_CREDENTIALS`, *GitLab*

2.16.32 GITLAB_TOKEN

Добавлено в версии 4.3.

GitLab'овский персональный токен доступа, используемый для вызовов API, необходимых для обновления перевода.

См.также:

GITLAB_CREDENTIALS, *GitLab*, GitLab: Персональный токен доступа

2.16.33 GITHUB_CREDENTIALS

Добавлено в версии 4.3.

Список учётных данных для серверов GitHub'а.

Подсказка: Используйте его в случае, если вы хотите, чтобы для одной конечной точки GitHub'а, определяемой параметрами *GITHUB_USERNAME* и *GITHUB_TOKEN*, Weblate взаимодействовал с несколькими серверами.

```
GITHUB_CREDENTIALS = {
    "api.github.com": {
        "username": "weblate",
        "token": "your-api-token",
    },
    "github.example.com": {
        "username": "weblate",
        "token": "another-api-token",
    },
}
```

2.16.34 GITHUB_USERNAME

Имя пользователя GitHub, используемое для отправки запросов на извлечение для обновления перевода.

См.также:

GITHUB_CREDENTIALS, *GitHub*

2.16.35 GITHUB_TOKEN

Добавлено в версии 4.3.

GitHub'овский персональный токен доступа, используемый для вызовов API, необходимых для отправки запросов на извлечение для обновления перевода.

См.также:

GITHUB_CREDENTIALS, *GitHub*, Создание персонального токена доступа

2.16.36 GOOGLE_ANALYTICS_ID

Идентификатор аналитики Google для включения мониторинга Weblate'а с помощью Google Analytics.

2.16.37 HIDE_REPO_CREDENTIALS

Скрыть учётные данные репозитория от появления в веб-интерфейсе. В случае, если в URL репозитория содержится пользователь и пароль, Weblate спрячет их, когда соответствующая информация будет показываться пользователям.

Например, вместо `https://user:password@git.example.com/repo.git` будет показано просто `https://git.example.com/repo.git`. Аналогичным образом он пытается очистить и сообщения об ошибках системы контроля версий.

Примечание: По умолчанию этот параметр включён.

2.16.38 IP_BEHIND_REVERSE_PROXY

Добавлено в версии 2.14.

Указывает, работает ли Weblate за обратным прокси.

Если установлен в ``True`, то Weblate получает IP адрес из заголовка, определённого параметром *IP_PROXY_HEADER*.

Предупреждение: Убедитесь, что вы и в самом деле используете обратный прокси и что он устанавливает этот заголовок, иначе пользователи смогут подделать IP адрес.

Примечание: По умолчанию этот параметр не включён.

См.также:

Работа за обратным прокси, *Ограничение частоты запросов*, *IP_PROXY_HEADER*, *IP_PROXY_OFFSET*

2.16.39 IP_PROXY_HEADER

Добавлено в версии 2.14.

Указывает, из какого заголовка Weblate должен получить IP-адрес при включении параметра *IP_BEHIND_REVERSE_PROXY*.

По умолчанию установлен в `HTTP_X_FORWARDED_FOR`.

См.также:

Работа за обратным прокси, *Ограничение частоты запросов*, `SECURE_PROXY_SSL_HEADER`, *IP_BEHIND_REVERSE_PROXY*, *IP_PROXY_OFFSET*

2.16.40 IP_PROXY_OFFSET

Добавлено в версии 2.14.

Указывает, какая часть параметра *IP_PROXY_HEADER* используется в качестве IP-адреса клиента.

В зависимости от вашей установки, этот заголовок может состоять из нескольких IP адресов (например, `X-Forwarded-For: a, b, client-ip`), и здесь вы можете настроить, какой адрес из заголовка используется в качестве IP адреса клиента.

> **Предупреждение:** Задание этого параметра влияет на безопасность вашей установки, вы должны настроить его на использование только доверенных прокси для определения IP адреса.

По умолчанию установлен в 0.

См.также:

Работа за обратным прокси, *Ограничение частоты запросов*, `SECURE_PROXY_SSL_HEADER`, *IP_BEHIND_REVERSE_PROXY*, *IP_PROXY_HEADER*

2.16.41 LEGAL_URL

Добавлено в версии 3.5.

URL-адрес, по которому ваш экземпляр Weblate показывает свои юридические документы.

Подсказка: Полезен, если вы размещаете свои юридические документы за пределами Weblate'а для дальнейшего их встраивания в Weblate, для получения подробной информации обратитесь к разделу *Право*.

Пример:

```
LEGAL_URL = "https://weblate.org/terms/"
```

2.16.42 LICENSE_EXTRA

Дополнительные лицензии для включения в список вариантов лицензий.

Примечание: Описание каждой лицензии должно быть кортежем, состоящим из её краткого наименования, длинного наименования и URL-адреса.

Например:

```
LICENSE_EXTRA = [
    (
        "AGPL-3.0",
        "GNU Affero General Public License v3.0",
        "https://www.gnu.org/licenses/agpl-3.0-standalone.html",
    ),
]
```

2.16.43 LICENSE_FILTER

Изменено в версии 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.

Примечание: Этот фильтр использует краткие наименования лицензий.

Например:

```
LICENSE_FILTER = {"AGPL-3.0", "GPL-3.0-or-later"}
```

Following disables the license alert:

```
LICENSE_FILTER = set()
```

См.также:

Предупреждения компонента перевода

2.16.44 LICENSE_REQUIRED

Определяет, требуется ли атрибут лицензии в *конфигурации компонента*.

Примечание: По умолчанию этот параметр выключен.

2.16.45 LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH

Должна ли длина данного перевода быть ограничена. Ограничением является количество символов исходной строки, увеличенное в 10 раз.

Подсказка: Установите его значение в `False` для разрешения более длинных переводов (до 10 000 символов) независимо от длины исходной строки.

Примечание: По умолчанию установлен в `True`.

2.16.46 LOCALIZE_CDN_URL и LOCALIZE_CDN_PATH

Эти параметры конфигурируют надстройку *CDN локализации JavaScript'a*. Параметр `LOCALIZE_CDN_URL` определяет корневой URL-адрес, по которому доступен CDN локализации, а параметр `LOCALIZE_CDN_PATH` определяет путь, по которому Weblate должен сохранять сгенерированные файлы, которые затем будут раздаваться с `LOCALIZE_CDN_URL`.

Подсказка: На Hosted Weblate для этого параметра используется значение `https://weblate-cdn.com/`.

См.также:

CDN локализации JavaScript'a

2.16.47 LOGIN_REQUIRED_URLS

Список URL-адресов, для посещения которых требуется выполнить вход в Weblate (помимо встроенных в Weblate стандартных правил).

Подсказка: Он позволяет вам защитить паролем всю установку:

```
LOGIN_REQUIRED_URLS = (
    r'/(.*)$',
)
REST_FRAMEWORK["DEFAULT_PERMISSION_CLASSES"] = [
    "rest_framework.permissions.IsAuthenticated"
]
```

Подсказка: Также желательно заблокировать доступ к API, как показано в примере выше.

2.16.48 LOGIN_REQUIRED_URLS_EXCEPTIONS

Список исключений для *LOGIN_REQUIRED_URLS*. Если не указан, пользователям будет разрешён доступ к странице входа.

Некоторые исключения, которые вы, возможно, захотите включить:

```
LOGIN_REQUIRED_URLS_EXCEPTIONS = (
    r'/accounts/(.*)$', # Required for login
    r'/static/(.*)$',   # 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
)
```

2.16.49 MATOMO_SITE_ID

Идентификатор сайта в Matomo (бывший Piwik), который вы хотите отслеживать.

Примечание: Эта интеграция не поддерживает Matomo Tag Manager.

См.также:

MATOMO_URL

2.16.50 MATOMO_URL

Полный URL-адрес (включая завершающую косую черту) установки Matomo (бывшего Piwik), которую вы хотите использовать для отслеживания использования Weblate. Для получения более подробной информации обратитесь к <https://matomo.org/>.

Подсказка: Эта интеграция не поддерживает Matomo Tag Manager.

Например:

```
MATOMO_SITE_ID = 1
MATOMO_URL = "https://example.matomo.cloud/"
```

См.также:

MATOMO_SITE_ID

2.16.51 MT_SERVICES

Изменено в версии 3.0: Параметр был переименован с `MACHINE_TRANSLATION_SERVICES` на *MT_SERVICES*` для соответствия названиям других параметров машинного перевода.

Список включённых сервисов машинного перевода.

Примечание: Многим сервисам нужна дополнительная настройка, например, задание ключей API, для получения более подробной информации обратитесь к их документации в разделе *Машинный перевод*.

```
MT_SERVICES = (
    'weblate.machinery.apertium.ApertiumAPYTranslation',
    'weblate.machinery.deepl.DeepLTranslation',
    'weblate.machinery.glosbe.GlosbeTranslation',
    'weblate.machinery.google.GoogleTranslation',
    'weblate.machinery.microsoft.MicrosoftCognitiveTranslation',
    'weblate.machinery.microsoftterminology.MicrosoftTerminologyService',
    'weblate.machinery.mymemory.MyMemoryTranslation',
    'weblate.machinery.tmserver.AmagamaTranslation',
    'weblate.machinery.tmserver.TMServerTranslation',
    'weblate.machinery.yandex.YandexTranslation',
    'weblate.machinery.weblatetm.WeblateTranslation',
    'weblate.machinery.saptranslationhub.SAPTranslationHub',
    'weblate.memory.machine.WeblateMemory',
)
```

См.также:

Машинный перевод, *Автоматические предложения*

2.16.52 MT_APERTIUM_APY

URL-адрес сервера Apertium-APy, https://wiki.apertium.org/wiki/Apertium-apy

См.также:

Apertium, *Машинный перевод*, *Автоматические предложения*

2.16.53 MT_AWS_ACCESS_KEY_ID

Идентификатор ключа доступа к Amazon Translate.

См.также:

AWS, *Машинный перевод*, *Автоматические предложения*

2.16.54 MT_AWS_SECRET_ACCESS_KEY

Секретный ключ API к Amazon Translate.

См.также:

AWS, *Машинный перевод*, *Автоматические предложения*

2.16.55 MT_AWS_REGION

Название используемого региона для Amazon Translate.

См.также:

AWS, *Машинный перевод*, *Автоматические предложения*

2.16.56 MT_BAIDU_ID

Идентификатор клиента для API Baidu Zhiyun, вы можете зарегистрироваться по адресу https://api.fanyi.baidu.com/api/trans/product/index

См.также:

Машинный перевод через API Baidu, *Машинный перевод*, *Автоматические предложения*

2.16.57 MT_BAIDU_SECRET

Секрет клиента для API Baidu Zhiyun, вы можете зарегистрироваться по адресу https://api.fanyi.baidu.com/api/trans/product/index

См.также:

Машинный перевод через API Baidu, *Машинный перевод*, *Автоматические предложения*

2.16.58 MT_DEEPL_API_VERSION

Добавлено в версии 4.1.1.

Версия API, используемая для сервиса DeepL. Версия ограничивает область использования:

v1 Предназначена для CAT-инструментов и может быть использована с пользовательской подпиской.

v2 Предназначена для использования через API, а подписка основана на использовании.

Ранее Weblate классифицировался DeepL как CAT-инструмент, поэтому предполагалось использовать API версии v1, но теперь предполагается использовать API версии v2. Поэтому по умолчанию параметр имеет значение v2, но вы можете изменить его на v1 в случае, если у вас есть существующая CAT-подписка и вы хотите, чтобы Weblate использовал её.

См.также:

DeepL, *Машинный перевод*, *Автоматические предложения*

2.16.59 MT_DEEPL_KEY

Ключ API для API DeepL, вы можете зарегистрироваться по адресу https://www.deepl.com/pro.html

См.также:

DeepL, *Машинный перевод*, *Автоматические предложения*

2.16.60 MT_GOOGLE_KEY

Ключ API для API переводчика Google версии v2, вы можете зарегистрироваться по адресу https://cloud.google.com/translate/docs

См.также:

Google Translate, *Машинный перевод*, *Автоматические предложения*

2.16.61 MT_GOOGLE_CREDENTIALS

JSON-файл учётных данных API версии v3, полученный в консоли облака Google. Пожалуйста, укажите полный путь в формате пути операционной системы. Учётные данные привязаны к сервису-учётной записи, связанной с определённым проектом. Для получения более подробной информации обратитесь по адресу https://cloud.google.com/docs/authentication/getting-started.

2.16.62 MT_GOOGLE_PROJECT

Идентификатор проекта API версии v3 облака Google с активированным тарифным планом и сервисом перевода. Для получения более подробной информации обратитесь по адресу https://cloud.google.com/appengine/docs/standard/nodejs/building-app/creating-project

2.16.63 MT_GOOGLE_LOCATION

API версии v3 App Engine облака Google может зависеть от местоположения. Измените его соответствующим образом, если значение по умолчанию `global` для вас не подходит.

Для получения более подробной информации обратитесь по адресу https://cloud.google.com/appengine/docs/locations

См.также:

Google Translate API V3 (расширенный)

2.16.64 MT_MICROSOFT_BASE_URL

Базовый URL-адрес региона, как определено в разделе «Базовые URL-адреса».

По умолчанию равен `api.cognitive.microsofttranslator.com` для Azure Global.

Для Azure China используйте, пожалуйста, `api.translator.azure.cn`.

2.16.65 MT_MICROSOFT_COGNITIVE_KEY

Клиентский ключ для API переводчика Microsoft Cognitive Services.

См.также:

Переводчик Microsoft Cognitive Services, *Машинный перевод*, *Автоматические предложения*, Cognitive Services — API перевода текста, Портал Microsoft Azure

2.16.66 MT_MICROSOFT_REGION

Префикс региона, как определено в разделе «Проверка подлинности с помощью ресурса с несколькими службами».

2.16.67 MT_MICROSOFT_ENDPOINT_URL

Конечная точка URL-адреса региона, как определено в разделе «Проверка подлинности с помощью маркера доступа».

По умолчанию равен `api.cognitive.microsoft.com` для Azure Global.

Для Azure China используйте, пожалуйста, свою конечную точку на портале Azure Portal.

2.16.68 MT_MODERNMT_KEY

Ключ API для движка машинного перевода ModernMT.

См.также:

ModernMT `MT_MODERNMT_URL`

2.16.69 MT_MODERNMT_URL

URL-адрес ModernMT. По умолчанию для облачного сервиса используется `https://api.modernmt.com/`.

См.также:

ModernMT `MT_MODERNMT_KEY`

2.16.70 MT_MYMEMORY_EMAIL

Идентификационный адрес электронной почты MyMemory. Разрешает выполнять 1000 запросов в день.

См.также:

MyMemory, *Машинный перевод*, *Автоматические предложения*, MyMemory: технические спецификации API

2.16.71 MT_MYMEMORY_KEY

Ключ доступа MyMemory для частной памяти переводов, используйте его совместно с параметром `MT_MYMEMORY_USER`.

См.также:

MyMemory, *Машинный перевод*, *Автоматические предложения*, MyMemory: генератор ключа API

2.16.72 MT_MYMEMORY_USER

Идентификатор пользователя MyMemory для частной памяти переводов, используйте его совместно с параметром `MT_MYMEMORY_KEY`.

См.также:

MyMemory, *Машинный перевод*, *Автоматические предложения*, MyMemory: генератор ключа API

2.16.73 MT_NETEASE_KEY

Ключ приложения для API NetEase Sight, вы можете зарегистрироваться по адресу https://sight.netease.com/

См.также:

API машинного перевода NetEase Sight, *Машинный перевод*, *Автоматические предложения*

2.16.74 MT_NETEASE_SECRET

Секрет приложения для API NetEase Sight, вы можете зарегистрироваться по адресу https://sight.netease.com/

См.также:

API машинного перевода NetEase Sight, *Машинный перевод*, *Автоматические предложения*

2.16.75 MT_TMSERVER

URL-адрес, по которому запущен tmserver.

См.также:

tmserver, *Машинный перевод*, *Автоматические предложения*, tmserver

2.16.76 MT_YANDEX_KEY

Ключ API для API Яндекс.Переводчика, вы можете зарегистрироваться по адресу https://yandex.ru/dev/translate/

См.также:

Yandex Translate (Яндекс.Переводчик), *Машинный перевод*, *Автоматические предложения*

2.16.77 MT_YOUDAO_ID

Идентификатор клиента для API Youdao Zhiyun, вы можете зарегистрироваться по адресу https://ai.youdao.com/product-fanyi-text.s.

См.также:

API машинного перевода Youdao Zhiyun, *Машинный перевод*, *Автоматические предложения*

2.16.78 MT_YOUDAO_SECRET

Секрет клиента для API Youdao Zhiyun, вы можете зарегистрироваться по адресу https://ai.youdao.com/product-fanyi-text.s.

См.также:

API машинного перевода Youdao Zhiyun, *Машинный перевод*, *Автоматические предложения*

2.16.79 MT_SAP_BASE_URL

URL-адрес API сервиса SAP Translation Hub.

См.также:

SAP Translation Hub, *Машинный перевод*, *Автоматические предложения*

2.16.80 MT_SAP_SANDBOX_APIKEY

Ключ API для использования API песочницы

См.также:

SAP Translation Hub, *Машинный перевод*, *Автоматические предложения*

2.16.81 MT_SAP_USERNAME

Ваше имя пользователя SAP

См.также:

SAP Translation Hub, *Машинный перевод*, *Автоматические предложения*

2.16.82 MT_SAP_PASSWORD

Ваш пароль SAP

См.также:

SAP Translation Hub, *Машинный перевод*, *Автоматические предложения*

2.16.83 MT_SAP_USE_MT

Использовать ли также сервисы машинного перевода в дополнение к терминологической базе данных. Возможные значения: `True` или `False`

См.также:

SAP Translation Hub, *Машинный перевод*, *Автоматические предложения*

2.16.84 NEARBY_MESSAGES

Сколько показывать строк около переводимой в настоящее время строки. Это просто значение по умолчанию, пользователи могут настроить его в *своём профиле*.

2.16.85 RATELIMIT_ATTEMPTS

Добавлено в версии 3.2.

Максимальное количество попыток аутентификации до применения ограничения частоты запросов.

По умолчанию установлен в 5.

См.также:

Ограничение частоты запросов, `RATELIMIT_WINDOW`, `RATELIMIT_LOCKOUT`

2.16.86 RATELIMIT_WINDOW

Добавлено в версии 3.2.

Как долго будет приниматься аутентификация после применения ограничения частоты запросов.

Количество секунд, по умолчанию равен 300 (5 минут).

См.также:

Ограничение частоты запросов, `RATELIMIT_ATTEMPTS`, `RATELIMIT_LOCKOUT`

2.16.87 RATELIMIT_LOCKOUT

Добавлено в версии 3.2.

Как долго аутентификация будет заблокирована после применения ограничения частоты запросов.

Количество секунд, по умолчанию равен 600 (10 минут).

См.также:

Ограничение частоты запросов, `RATELIMIT_ATTEMPTS`, `RATELIMIT_WINDOW`

2.16.88 REGISTRATION_ALLOW_BACKENDS

Добавлено в версии 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.

Пример:

```
REGISTRATION_ALLOW_BACKENDS = ["azuread-oauth2", "azuread-tenant-oauth2"]
```

Подсказка: Имена бэкэндов совпадают с именами, используемыми в URL-адресах для аутентификации.

См.также:

`REGISTRATION_OPEN`, *Аутентификация*

2.16.89 REGISTRATION_CAPTCHA

Значение, которое может быть равно только `True` или `False`, указывает, защищена ли регистрация новых учётных записей капчей. Этот параметр является необязательным и если он не указан, будет принято значение по умолчанию `True`.

Если он включён, ко всем страницам, на которых пользователь вводит свой адрес электронной почты, добавляется капча:

- Регистрация новой учётной записи.
- Восстановление пароля.
- Добавление в учётную запись адреса электронной почты.
- Контактная форма для пользователей, не вошедших в систему.

2.16.90 REGISTRATION_EMAIL_MATCH

Добавлено в версии 2.17.

Позволяет вам отфильтровать допустимые для регистрации адреса электронной почты.

По умолчанию равен `.*`, что позволяет зарегистрироваться с любым адресом электронной почты.

Вы можете использовать этот параметр для ограничения регистрации одним доменом электронной почты:

```
REGISTRATION_EMAIL_MATCH = r'^.*@weblate\.org$'
```

2.16.91 REGISTRATION_OPEN

Разрешена ли в настоящее время регистрация новых учётных записей. Этот необязательный параметр может остаться в значении по умолчанию `True` или быть изменённым на `False`.

Этот параметр влияет на встроенную аутентификацию по адресу электронной почты или через Python Social Auth (при помощи параметра *REGISTRATION_ALLOW_BACKENDS* вы можете занести определённые бэкэнды в белый список).

Примечание: При использовании сторонних методов аутентификации, вроде *аутентификации через LDAP*, этот параметр просто скрывает форму регистрации, но новые пользователи всё равно смогут входить в систему и создавать учётные записи.

См.также:

REGISTRATION_ALLOW_BACKENDS, *REGISTRATION_EMAIL_MATCH*, *Аутентификация*

2.16.92 REPOSITORY_ALERT_THRESHOLD

Добавлено в версии 4.0.2.

Порог срабатывания предупреждения для устаревших репозиториев или репозиториев, содержащих слишком много изменений. По умолчанию равен 25.

См.также:

Предупреждения компонента перевода

2.16.93 REQUIRE_LOGIN

Добавлено в версии 4.1.

This enables *LOGIN_REQUIRED_URLS* and configures REST framework to require login for all API endpoints.

Примечание: This is in implemented in the *Пример конфигурации*. For Docker, use *WEBLATE_REQUIRE_LOGIN*.

2.16.94 SENTRY_DSN

Добавлено в версии 3.9.

DSN Sentry для *сбора отчётов об ошибках*.

См.также:

Django integration for Sentry

2.16.95 SESSION_COOKIE_AGE_AUTHENTICATED

Добавлено в версии 4.3.

Set session expiry for authenticated users. This complements `SESSION_COOKIE_AGE` which is used for unauthenticated users.

См.также:

`SESSION_COOKIE_AGE`

2.16.96 SIMPLIFY_LANGUAGES

Использовать простые коды языка для умолчательных комбинаций языка/страны. Например, при переводе `ru_RU` будет использоваться код языка `ru`. Обычно это желательное поведение, поскольку для таких умолчательных комбинаций оно упрощает перечисление языков.

Выключите этот параметр, если вы хотите разделять переводы для каждого варианта.

2.16.97 SITE_DOMAIN

Настраивает домен сайта. Это необходимо для получения корректных абсолютных ссылок во многих местах (например, в письмах активации, в уведомлениях или в RSS-каналах).

Если Weblate работает на нестандартном порту, добавьте его и сюда тоже.

Примеры:

```
# Production site with domain name
SITE_DOMAIN = "weblate.example.com"

# Local development with IP address and port
SITE_DOMAIN = "127.0.0.1:8000"
```

Примечание: Этот параметр должен содержать только доменное имя. Для настройки протокола (включения и принудительного включения HTTPS) используйте параметр *ENABLE_HTTPS*, а для изменения URL-адреса — параметр *URL_PREFIX*.

Подсказка: На контейнере Docker'а домен сайта настраивается через переменную окружения *WEBLATE_ALLOWED_HOSTS*.

См.также:

Установка правильного домена сайта, *Настройка разрешённых хостов*, *Правильная настройка HTTPS* *WEBLATE_SITE_DOMAIN*, *ENABLE_HTTPS*

2.16.98 SITE_TITLE

Заголовок сайта, который будет использоваться как на самом сайте, так и указываться в электронных письмах.

2.16.99 SPECIAL_CHARS

Дополнительные символы для включения в визуальную клавиатуру, смотрите раздел *Визуальная клавиатура*.

Значение по умолчанию:

```
SPECIAL_CHARS = ('\t', '\n', '…')
```

2.16.100 SINGLE_PROJECT

Добавлено в версии 3.8.

Перенаправляет пользователей непосредственно в проект или компонент вместо отображения панели управления. Вы можете установить параметр в `True`, в этом случае он будет работать только в том случае, если в Weblate'е и в самом деле есть только один проект. Или же можно задать плашку проекта, тогда пользователь будет безоговорочно перенаправлен в этот проект.

Изменено в версии 3.11: Теперь параметр также принимает плашку проекта для принудительного отображения этого единственного проекта.

Пример:

```
SINGLE_PROJECT = "test"
```

2.16.101 STATUS_URL

URL-адрес, по которому ваш экземпляр Weblate сообщает о своем статусе.

2.16.102 SUGGESTION_CLEANUP_DAYS

Добавлено в версии 3.2.1.

Автоматическое удаление предложения через заданное количество дней. По умолчанию установлен в ``None`, что означает отсутствие удалений.

2.16.103 URL_PREFIX

Этот параметр позволяет вам запускать Weblate по какому-либо пути (в противном случае он считает, что запущен из корня веб-сервера).

Примечание: Для использования этого параметра вам также нужно настроить ваш сервер на срезание этого префикса. Например, при использовании WSGI это можно сделать, установив псевдоним `WSGIScriptAlias`.

Подсказка: Префикс должен начинаться с символа `/`.

Пример:

```
URL_PREFIX = '/translations'
```

Примечание: Этот параметр не работает со встроенным сервером Django, чтобы он реагировал на этот префикс, вам придётся подправить `urls.py`.

2.16.104 VCS_BACKENDS

Конфигурация доступных систем контроля версий.

Примечание: Weblate пытается использовать все поддерживаемые системы, для которых у вас есть инструменты.

Подсказка: При помощи этого параметра вы можете ограничить их выбор или добавить пользовательские системы контроля версий.

```
VCS_BACKENDS = (
   'weblate.vcs.git.GitRepository',
)
```

См.также:

Интеграция с системой контроля версий

2.16.105 VCS_CLONE_DEPTH

Добавлено в версии 3.10.2.

Настраивает, насколько глубоким должно быть клонирование репозиториев Weblate'ом.

Примечание: В настоящее время этот параметр поддерживается только в *Git*. По умолчанию Weblate делает поверхностные клоны репозиториев, чтобы сделать клонирование быстрее и сэкономить место на диске. В зависимости от использования (например, при использовании пользовательских *надстроек*), вы можете захотеть увеличить глубину клонирования или полностью отключить поверхностное клонирование, что можно сделать, установив этот параметр в значение 0.

Подсказка: В случае, если вы при отправке коммитов из Weblate'а получили ошибку `fatal: protocol error: expected old/new/ref, got 'shallow <commit hash>'`, полностью отключите поверхностные клоны, установив соответствующий параметр:

```
VCS_CLONE_DEPTH = 0
```

2.16.106 WEBLATE_ADDONS

Список доступных для использования надстроек. Чтобы их использовать, они должны быть включены для заданного компонента перевода. По умолчанию включены все встроенные дополнения, при расширении списка вы, вероятно, захотите, чтобы существующие дополнения оставались включёнными, например:

```
WEBLATE_ADDONS = (
    # Built-in addons
    "weblate.addons.gettext.GenerateMoAddon",
    "weblate.addons.gettext.UpdateLinguasAddon",
    "weblate.addons.gettext.UpdateConfigureAddon",
    "weblate.addons.gettext.MsgmergeAddon",
    "weblate.addons.gettext.GettextCustomizeAddon",
    "weblate.addons.gettext.GettextAuthorComments",
    "weblate.addons.cleanup.CleanupAddon",
    "weblate.addons.consistency.LangaugeConsistencyAddon",
    "weblate.addons.discovery.DiscoveryAddon",
    "weblate.addons.flags.SourceEditAddon",
    "weblate.addons.flags.TargetEditAddon",
    "weblate.addons.flags.SameEditAddon",
    "weblate.addons.flags.BulkEditAddon",
    "weblate.addons.generate.GenerateFileAddon",
    "weblate.addons.json.JSONCustomizeAddon",
    "weblate.addons.properties.PropertiesSortAddon",
    "weblate.addons.git.GitSquashAddon",
    "weblate.addons.removal.RemoveComments",
    "weblate.addons.removal.RemoveSuggestions",
    "weblate.addons.resx.ResxUpdateAddon",
    "weblate.addons.autotranslate.AutoTranslateAddon",
    "weblate.addons.yaml.YAMLCustomizeAddon",
    "weblate.addons.cdn.CDNJSAddon",

    # Addon you want to include
    "weblate.addons.example.ExampleAddon",
)
```

См.также:

Надстройки

2.16.107 WEBLATE_EXPORTERS

Добавлено в версии 4.2.

Список доступных экспортеров, предлагающих скачивание переводов или глоссариев в различных форматах файлов.

См.также:

Поддерживаемые форматы файлов

2.16.108 WEBLATE_FORMATS

Добавлено в версии 3.0.

Список доступных для использования форматов файлов.

Примечание: В списке по умолчанию уже есть распространённые форматы.

См.также:

Поддерживаемые форматы файлов

2.16.109 WEBLATE_GPG_IDENTITY

Добавлено в версии 3.1.

Идентификатор, используемый Weblate'ом для подписывания коммитов Git, например:

```
WEBLATE_GPG_IDENTITY = 'Weblate <weblate@example.com>'
```

В GPG'шной связке ключей Weblate ищет соответствующий ключ (файл `home/.gnupg` в каталоге `DATA_DIR`). Если ключ не найден, он генерируется, подробнее смотрите в разделе *Подписание коммитов Git с помощью GnuPG*.

См.также:

Подписание коммитов Git с помощью GnuPG

2.17 Пример конфигурации

Следующий пример поставляется с Weblate'ом в виде файла `weblate/settings_example.py`:

```
#
# Copyright © 2012 - 2020 Michal Čihař <michal@cihar.com>
#
# This file is part of Weblate <https://weblate.org/>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <https://www.gnu.org/licenses/>.
#


import os
import platform
from logging.handlers import SysLogHandler

#
# Django settings for Weblate project.
#
```

(continues on next page)

(продолжение с предыдущей страницы)

```
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",
        # 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,
        },
    }
}

BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))

# Data directory
DATA_DIR = os.path.join(BASE_DIR, "data")

# Local time zone for this installation. Choices can be found here:
# http://en.wikipedia.org/wiki/List_of_tz_zones_by_name
# although not all choices may be available on all operating systems.
# In a Windows environment this must be set to your system time zone.
TIME_ZONE = "UTC"

# Language code for this installation. All choices can be found here:
# http://www.i18nguy.com/unicode/language-identifiers.html
LANGUAGE_CODE = "en-us"

LANGUAGES = (
    ("ar", "ةيبرعلا"),
    ("az", "Azərbaycan"),
    ("be", "Беларуская"),
    ("be@latin", "Biełaruskaja"),
    ("bg", "Български"),
    ("br", "Brezhoneg"),
    ("ca", "Català"),
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    ("cs", "Čeština"),
    ("da", "Dansk"),
    ("de", "Deutsch"),
    ("en", "English"),
    ("el", "Ελληνικά"),
    ("en-gb", "English (United Kingdom)"),
    ("es", "Español"),
    ("fi", "Suomi"),
    ("fr", "Français"),
    ("gl", "Galego"),
    ("he", "עברית"),
    ("hu", "Magyar"),
    ("hr", "Hrvatski"),
    ("id", "Indonesia"),
    ("is", "Íslenska"),
    ("it", "Italiano"),
    ("ja", "□□□"),
    ("kab", "Taqbaylit"),
    ("kk", "Қазақ тілі"),
    ("ko", "□□□"),
    ("nb", "Norsk bokmål"),
    ("nl", "Nederlands"),
    ("pl", "Polski"),
    ("pt", "Português"),
    ("pt-br", "Português brasileiro"),
    ("ru", "Русский"),
    ("sk", "Slovenčina"),
    ("sl", "Slovenščina"),
    ("sq", "Shqip"),
    ("sr", "Српски"),
    ("sv", "Svenska"),
    ("tr", "Türkçe"),
    ("uk", "Українська"),
    ("zh-hans", "□□□"),
    ("zh-hant", "□□□"),
)

SITE_ID = 1

# If you set this to False, Django will make some optimizations so as not
# to load the internationalization machinery.
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

# 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.
```

(continues on next page)

(продолжение с предыдущей страницы)

```
# Don't put anything in this directory yourself; store your static files
# in apps' "static/" subdirectories and in STATICFILES_DIRS.
STATIC_ROOT = os.path.join(DATA_DIR, "static")

# URL prefix for static files.
STATIC_URL = f"{URL_PREFIX}/static/"

# Additional locations of static files
STATICFILES_DIRS = (
    # Put strings here, like "/home/html/static" or "C:/www/django/static".
    # Always use forward slashes, even on Windows.
    # Don't forget to use absolute paths, not relative paths.
)

# List of finder classes that know how to find static files in
# various locations.
STATICFILES_FINDERS = (
    "django.contrib.staticfiles.finders.FileSystemFinder",
    "django.contrib.staticfiles.finders.AppDirectoriesFinder",
    "compressor.finders.CompressorFinder",
)

# Make this unique, and don't share it with anybody.
# You can generate it using weblate/examples/generate-secret-key
SECRET_KEY = ""

_TEMPLATE_LOADERS = [
    "django.template.loaders.filesystem.Loader",
    "django.template.loaders.app_directories.Loader",
]
if not DEBUG:
    _TEMPLATE_LOADERS = [("django.template.loaders.cached.Loader", _TEMPLATE_
↪LOADERS)]
TEMPLATES = [
    {
        "BACKEND": "django.template.backends.django.DjangoTemplates",
        "OPTIONS": {
            "context_processors": [
                "django.contrib.auth.context_processors.auth",
                "django.template.context_processors.debug",
                "django.template.context_processors.i18n",
                "django.template.context_processors.request",
                "django.template.context_processors.csrf",
                "django.contrib.messages.context_processors.messages",
                "weblate.trans.context_processors.weblate_context",
            ],
            "loaders": _TEMPLATE_LOADERS,
        },
    }
]


# GitHub username for sending pull requests.
# Please see the documentation for more details.
GITHUB_USERNAME = None

# GitLab username for sending merge requests.
# Please see the documentation for more details.
GITLAB_USERNAME = None

# Authentication configuration
```

(continues on next page)

(продолжение с предыдущей страницы)

```
AUTHENTICATION_BACKENDS = (
    "social_core.backends.email.EmailAuth",
    # "social_core.backends.google.GoogleOAuth2",
    # "social_core.backends.github.GithubOAuth2",
    # "social_core.backends.bitbucket.BitbucketOAuth",
    # "social_core.backends.suse.OpenSUSEOpenId",
    # "social_core.backends.ubuntu.UbuntuOpenId",
    # "social_core.backends.fedora.FedoraOpenId",
    # "social_core.backends.facebook.FacebookOAuth2",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Custom user model
AUTH_USER_MODEL = "weblate_auth.User"

# Social auth backends setup
SOCIAL_AUTH_GITHUB_KEY = ""
SOCIAL_AUTH_GITHUB_SECRET = ""
SOCIAL_AUTH_GITHUB_SCOPE = ["user:email"]

SOCIAL_AUTH_BITBUCKET_KEY = ""
SOCIAL_AUTH_BITBUCKET_SECRET = ""
SOCIAL_AUTH_BITBUCKET_VERIFIED_EMAILS_ONLY = True

SOCIAL_AUTH_FACEBOOK_KEY = ""
SOCIAL_AUTH_FACEBOOK_SECRET = ""
SOCIAL_AUTH_FACEBOOK_SCOPE = ["email", "public_profile"]
SOCIAL_AUTH_FACEBOOK_PROFILE_EXTRA_PARAMS = {"fields": "id,name,email"}

SOCIAL_AUTH_GOOGLE_OAUTH2_KEY = ""
SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET = ""

# Social auth settings
SOCIAL_AUTH_PIPELINE = (
    "social_core.pipeline.social_auth.social_details",
    "social_core.pipeline.social_auth.social_uid",
    "social_core.pipeline.social_auth.auth_allowed",
    "social_core.pipeline.social_auth.social_user",
    "weblate.accounts.pipeline.store_params",
    "weblate.accounts.pipeline.verify_open",
    "social_core.pipeline.user.get_username",
    "weblate.accounts.pipeline.require_email",
    "social_core.pipeline.mail.mail_validation",
    "weblate.accounts.pipeline.revoke_mail_code",
    "weblate.accounts.pipeline.ensure_valid",
    "weblate.accounts.pipeline.remove_account",
    "social_core.pipeline.social_auth.associate_by_email",
    "weblate.accounts.pipeline.reauthenticate",
    "weblate.accounts.pipeline.verify_username",
    "social_core.pipeline.user.create_user",
    "social_core.pipeline.social_auth.associate_user",
    "social_core.pipeline.social_auth.load_extra_data",
    "weblate.accounts.pipeline.cleanup_next",
    "weblate.accounts.pipeline.user_full_name",
    "weblate.accounts.pipeline.store_email",
    "weblate.accounts.pipeline.notify_connect",
    "weblate.accounts.pipeline.password_reset",
)
SOCIAL_AUTH_DISCONNECT_PIPELINE = (
    "social_core.pipeline.disconnect.allowed_to_disconnect",
    "social_core.pipeline.disconnect.get_entries",
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    "social_core.pipeline.disconnect.revoke_tokens",
    "weblate.accounts.pipeline.cycle_session",
    "weblate.accounts.pipeline.adjust_primary_mail",
    "weblate.accounts.pipeline.notify_disconnect",
    "social_core.pipeline.disconnect.disconnect",
    "weblate.accounts.pipeline.cleanup_next",
)

# Custom authentication strategy
SOCIAL_AUTH_STRATEGY = "weblate.accounts.strategy.WeblateStrategy"

# Raise exceptions so that we can handle them later
SOCIAL_AUTH_RAISE_EXCEPTIONS = True

SOCIAL_AUTH_EMAIL_VALIDATION_FUNCTION = "weblate.accounts.pipeline.send_validation"
SOCIAL_AUTH_EMAIL_VALIDATION_URL = "{0}/accounts/email-sent/".format(URL_PREFIX)
SOCIAL_AUTH_LOGIN_ERROR_URL = "{0}/accounts/login/".format(URL_PREFIX)
SOCIAL_AUTH_EMAIL_FORM_URL = "{0}/accounts/email/".format(URL_PREFIX)
SOCIAL_AUTH_NEW_ASSOCIATION_REDIRECT_URL = "{0}/accounts/profile/#account".format(
    URL_PREFIX
)
SOCIAL_AUTH_PROTECTED_USER_FIELDS = ("email",)
SOCIAL_AUTH_SLUGIFY_USERNAMES = True
SOCIAL_AUTH_SLUGIFY_FUNCTION = "weblate.accounts.pipeline.slugify_username"

# Password validation configuration
AUTH_PASSWORD_VALIDATORS = [
    {
        "NAME": "django.contrib.auth.password_validation.
↪UserAttributeSimilarityValidator"  # noqa: E501, pylint: disable=line-too-long
    },
    {
        "NAME": "django.contrib.auth.password_validation.MinimumLengthValidator",
        "OPTIONS": {"min_length": 10},
    },
    {"NAME": "django.contrib.auth.password_validation.CommonPasswordValidator"},
    {"NAME": "django.contrib.auth.password_validation.NumericPasswordValidator"},
    {"NAME": "weblate.accounts.password_validation.CharsPasswordValidator"},
    {"NAME": "weblate.accounts.password_validation.PastPasswordsValidator"},
    # Optional password strength validation by django-zxcvbn-password
    # {
    #     "NAME": "zxcvbn_password.ZXCVBNValidator",
    #     "OPTIONS": {
    #         "min_score": 3,
    #         "user_attributes": ("username", "email", "full_name")
    #     }
    # },
]

# Allow new user registrations
REGISTRATION_OPEN = True

# Shortcut for login required setting
REQUIRE_LOGIN = False

# Middleware
MIDDLEWARE = [
    "weblate.middleware.RedirectMiddleware",
    "weblate.middleware.ProxyMiddleware",
    "django.middleware.security.SecurityMiddleware",
    "django.contrib.sessions.middleware.SessionMiddleware",
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    "django.middleware.common.CommonMiddleware",
    "django.middleware.csrf.CsrfViewMiddleware",
    "weblate.accounts.middleware.AuthenticationMiddleware",
    "django.contrib.messages.middleware.MessageMiddleware",
    "django.middleware.clickjacking.XFrameOptionsMiddleware",
    "social_django.middleware.SocialAuthExceptionMiddleware",
    "weblate.accounts.middleware.RequireLoginMiddleware",
    "weblate.api.middleware.ThrottlingMiddleware",
    "weblate.middleware.SecurityMiddleware",
]

ROOT_URLCONF = "weblate.urls"

# Django and Weblate apps
INSTALLED_APPS = [
    # Weblate apps on top to override Django locales and templates
    "weblate.addons",
    "weblate.auth",
    "weblate.checks",
    "weblate.formats",
    "weblate.glossary",
    "weblate.machinery",
    "weblate.trans",
    "weblate.lang",
    "weblate_language_data",
    "weblate.memory",
    "weblate.screenshots",
    "weblate.fonts",
    "weblate.accounts",
    "weblate.utils",
    "weblate.vcs",
    "weblate.wladmin",
    "weblate",
    # Optional: Git exporter
    "weblate.gitexport",
    # Standard Django modules
    "django.contrib.auth",
    "django.contrib.contenttypes",
    "django.contrib.sessions",
    "django.contrib.messages",
    "django.contrib.staticfiles",
    "django.contrib.admin.apps.SimpleAdminConfig",
    "django.contrib.admindocs",
    "django.contrib.sitemaps",
    "django.contrib.humanize",
    # Third party Django modules
    "social_django",
    "crispy_forms",
    "compressor",
    "rest_framework",
    "rest_framework.authtoken",
    "django_filters",
]

# Custom exception reporter to include some details
DEFAULT_EXCEPTION_REPORTER_FILTER = "weblate.trans.debug.
↪WeblateExceptionReporterFilter"

# Default logging of Weblate messages
# - to syslog in production (if available)
# - otherwise to console
```

(continues on next page)

(продолжение с предыдущей страницы)

```
# - you can also choose "logfile" to log into separate file
#   after configuring it below

# Detect if we can connect to syslog
HAVE_SYSLOG = False
if platform.system() != "Windows":
    try:
        handler = SysLogHandler(address="/dev/log", facility=SysLogHandler.LOG_
↪LOCAL2)
        handler.close()
        HAVE_SYSLOG = True
    except IOError:
        HAVE_SYSLOG = False

if DEBUG or not HAVE_SYSLOG:
    DEFAULT_LOG = "console"
else:
    DEFAULT_LOG = "syslog"
DEFAULT_LOGLEVEL = "DEBUG" if DEBUG else "INFO"

# A sample logging configuration. The only tangible logging
# performed by this configuration is to send an email to
# the site admins on every HTTP 500 error when DEBUG=False.
# See http://docs.djangoproject.com/en/stable/topics/logging for
# more details on how to customize your logging configuration.
LOGGING = {
    "version": 1,
    "disable_existing_loggers": True,
    "filters": {"require_debug_false": {"()": "django.utils.log.RequireDebugFalse"}
↪},
    "formatters": {
        "syslog": {"format": "weblate[%(process)d]: %(levelname)s %(message)s"},
        "simple": {"format": "%(levelname)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",
```

(continues on next page)

(продолжение с предыдущей страницы)

```
            "facility": SysLogHandler.LOG_LOCAL2,
        },
        # Logging to a file
        # "logfile": {
        #     "level":"DEBUG",
        #     "class":"logging.handlers.RotatingFileHandler",
        #     "filename": "/var/log/weblate/weblate.log",
        #     "maxBytes": 100000,
        #     "backupCount": 3,
        #     "formatter": "logfile",
        # },
    },
    "loggers": {
        "django.request": {
            "handlers": ["mail_admins", DEFAULT_LOG],
            "level": "ERROR",
            "propagate": True,
        },
        "django.server": {
            "handlers": ["django.server"],
            "level": "INFO",
            "propagate": False,
        },
        # Logging database queries
        # "django.db.backends": {
        #     "handlers": [DEFAULT_LOG],
        #     "level": "DEBUG",
        # },
        "weblate": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
        # Logging VCS operations
        "weblate.vcs": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
        # Python Social Auth
        "social": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
        # Django Authentication Using LDAP
        "django_auth_ldap": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
    },
}

# Remove syslog setup if it's not present
if not HAVE_SYSLOG:
    del LOGGING["handlers"]["syslog"]

# List of machine translations
MT_SERVICES = (
    #     "weblate.machinery.apertium.ApertiumAPYTranslation",
    #     "weblate.machinery.baidu.BaiduTranslation",
    #     "weblate.machinery.deepl.DeepLTranslation",
    #     "weblate.machinery.glosbe.GlosbeTranslation",
    #     "weblate.machinery.google.GoogleTranslation",
    #     "weblate.machinery.googlev3.GoogleV3Translation",
    #     "weblate.machinery.microsoft.MicrosoftCognitiveTranslation",
    #     "weblate.machinery.microsoftterminology.MicrosoftTerminologyService",
    #     "weblate.machinery.modernmt.ModernMTTranslation",
    #     "weblate.machinery.mymemory.MyMemoryTranslation",
    #     "weblate.machinery.netease.NeteaseSightTranslation",
    #     "weblate.machinery.tmserver.AmagamaTranslation",
    #     "weblate.machinery.tmserver.TMServerTranslation",
    #     "weblate.machinery.yandex.YandexTranslation",
    #     "weblate.machinery.saptranslationhub.SAPTranslationHub",
    #     "weblate.machinery.youdao.YoudaoTranslation",
    "weblate.machinery.weblatetm.WeblateTranslation",
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    "weblate.memory.machine.WeblateMemory",
)

# Machine translation API keys

# URL of the Apertium APy server
MT_APERTIUM_APY = None

# DeepL API key
MT_DEEPL_KEY = None

# 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

# Title of site to use
SITE_TITLE = "Weblate"
```

(continues on next page)

(продолжение с предыдущей страницы)

```
# Site domain
SITE_DOMAIN = ""

# Whether site uses https
ENABLE_HTTPS = False

# Use HTTPS when creating redirect URLs for social authentication, see
# documentation for more details:
# https://python-social-auth-docs.readthedocs.io/en/latest/configuration/settings.
↪html#processing-redirects-and-urlopen
SOCIAL_AUTH_REDIRECT_IS_HTTPS = ENABLE_HTTPS

# Make CSRF cookie HttpOnly, see documentation for more details:
# https://docs.djangoproject.com/en/1.11/ref/settings/#csrf-cookie-httponly
CSRF_COOKIE_HTTPONLY = True
CSRF_COOKIE_SECURE = ENABLE_HTTPS
# Store CSRF token in session
CSRF_USE_SESSIONS = True
# Customize CSRF failure view
CSRF_FAILURE_VIEW = "weblate.trans.views.error.csrf_failure"
SESSION_COOKIE_SECURE = ENABLE_HTTPS
SESSION_COOKIE_HTTPONLY = True
# SSL redirect
SECURE_SSL_REDIRECT = ENABLE_HTTPS
# Sent referrrer 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 = 2200
SESSION_COOKIE_AGE_AUTHENTICATED = 1209600
# Increase allowed upload size
DATA_UPLOAD_MAX_MEMORY_SIZE = 50000000

# Apply session coookie settings to language cookie as ewll
LANGUAGE_COOKIE_SECURE = SESSION_COOKIE_SECURE
LANGUAGE_COOKIE_HTTPONLY = SESSION_COOKIE_HTTPONLY
LANGUAGE_COOKIE_AGE = SESSION_COOKIE_AGE * 10

# 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 = "{0}/accounts/login/".format(URL_PREFIX)

# URL of logout
LOGOUT_URL = "{0}/accounts/logout/".format(URL_PREFIX)

# Default location for login
LOGIN_REDIRECT_URL = "{0}/".format(URL_PREFIX)
```

(continues on next page)

(продолжение с предыдущей страницы)

```
# 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 = "[{0}] ".format(SITE_TITLE)

# Enable remote hooks
ENABLE_HOOKS = True

# By default the length of a given translation is limited to the length of
# the source string * 10 characters. Set this option to False to allow longer
# translations (up to 10.000 characters)
LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH = True

# Use simple language codes for default language/country combinations
SIMPLIFY_LANGUAGES = True

# Render forms using bootstrap
CRISPY_TEMPLATE_PACK = "bootstrap3"

# List of quality checks
# CHECK_LIST = (
#     "weblate.checks.same.SameCheck",
#     "weblate.checks.chars.BeginNewlineCheck",
#     "weblate.checks.chars.EndNewlineCheck",
#     "weblate.checks.chars.BeginSpaceCheck",
#     "weblate.checks.chars.EndSpaceCheck",
#     "weblate.checks.chars.DoubleSpaceCheck",
#     "weblate.checks.chars.EndStopCheck",
#     "weblate.checks.chars.EndColonCheck",
#     "weblate.checks.chars.EndQuestionCheck",
#     "weblate.checks.chars.EndExclamationCheck",
#     "weblate.checks.chars.EndEllipsisCheck",
#     "weblate.checks.chars.EndSemicolonCheck",
#     "weblate.checks.chars.MaxLengthCheck",
#     "weblate.checks.chars.KashidaCheck",
#     "weblate.checks.chars.PunctuationSpacingCheck",
#     "weblate.checks.format.PythonFormatCheck",
#     "weblate.checks.format.PythonBraceFormatCheck",
#     "weblate.checks.format.PHPFormatCheck",
#     "weblate.checks.format.CFormatCheck",
#     "weblate.checks.format.PerlFormatCheck",
#     "weblate.checks.format.JavaScriptFormatCheck",
#     "weblate.checks.format.CSharpFormatCheck",
#     "weblate.checks.format.JavaFormatCheck",
#     "weblate.checks.format.JavaMessageFormatCheck",
#     "weblate.checks.format.PercentPlaceholdersCheck",
#     "weblate.checks.format.VueFormattingCheck",
#     "weblate.checks.format.I18NextInterpolationCheck",
#     "weblate.checks.format.ESTemplateLiteralsCheck",
#     "weblate.checks.angularjs.AngularJSInterpolationCheck",
#     "weblate.checks.qt.QtFormatCheck",
```

(continues on next page)

(продолжение с предыдущей страницы)

```
#     "weblate.checks.qt.QtPluralCheck",
#     "weblate.checks.ruby.RubyFormatCheck",
#     "weblate.checks.consistency.PluralsCheck",
#     "weblate.checks.consistency.SamePluralsCheck",
#     "weblate.checks.consistency.ConsistencyCheck",
#     "weblate.checks.consistency.TranslatedCheck",
#     "weblate.checks.chars.EscapedNewlineCountingCheck",
#     "weblate.checks.chars.NewLineCountCheck",
#     "weblate.checks.markup.BBCodeCheck",
#     "weblate.checks.chars.ZeroWidthSpaceCheck",
#     "weblate.checks.render.MaxSizeCheck",
#     "weblate.checks.markup.XMLValidityCheck",
#     "weblate.checks.markup.XMLTagsCheck",
#     "weblate.checks.markup.MarkdownRefLinkCheck",
#     "weblate.checks.markup.MarkdownLinkCheck",
#     "weblate.checks.markup.MarkdownSyntaxCheck",
#     "weblate.checks.markup.URLCheck",
#     "weblate.checks.markup.SafeHTMLCheck",
#     "weblate.checks.placeholders.PlaceholderCheck",
#     "weblate.checks.placeholders.RegexCheck",
#     "weblate.checks.duplicate.DuplicateCheck",
#     "weblate.checks.source.OptionalPluralCheck",
#     "weblate.checks.source.EllipsisCheck",
#     "weblate.checks.source.MultipleFailingCheck",
#     "weblate.checks.source.LongUntranslatedCheck",
#     "weblate.checks.format.MultipleUnnamedFormatsCheck",
# )

# List of automatic fixups
# AUTOFIX_LIST = (
#     "weblate.trans.autofixes.whitespace.SameBookendingWhitespace",
#     "weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis",
#     "weblate.trans.autofixes.chars.RemoveZeroSpace",
#     "weblate.trans.autofixes.chars.RemoveControlChars",
# )

# List of enabled addons
# WEBLATE_ADDONS = (
#     "weblate.addons.gettext.GenerateMoAddon",
#     "weblate.addons.gettext.UpdateLinguasAddon",
#     "weblate.addons.gettext.UpdateConfigureAddon",
#     "weblate.addons.gettext.MsgmergeAddon",
#     "weblate.addons.gettext.GettextCustomizeAddon",
#     "weblate.addons.gettext.GettextAuthorComments",
#     "weblate.addons.cleanup.CleanupAddon",
#     "weblate.addons.consistency.LangaugeConsistencyAddon",
#     "weblate.addons.discovery.DiscoveryAddon",
#     "weblate.addons.flags.SourceEditAddon",
#     "weblate.addons.flags.TargetEditAddon",
#     "weblate.addons.flags.SameEditAddon",
#     "weblate.addons.flags.BulkEditAddon",
#     "weblate.addons.generate.GenerateFileAddon",
#     "weblate.addons.json.JSONCustomizeAddon",
#     "weblate.addons.properties.PropertiesSortAddon",
#     "weblate.addons.git.GitSquashAddon",
#     "weblate.addons.removal.RemoveComments",
#     "weblate.addons.removal.RemoveSuggestions",
#     "weblate.addons.resx.ResxUpdateAddon",
#     "weblate.addons.yaml.YAMLCustomizeAddon",
#     "weblate.addons.cdn.CDNJSAddon",
#     "weblate.addons.autotranslate.AutoTranslateAddon",
```

(continues on next page)

(продолжение с предыдущей страницы)

```
# )

# 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",
            "PASSWORD": None,
            "CONNECTION_POOL_KWARGS": {},
        },
        "KEY_PREFIX": "weblate",
    },
    "avatar": {
        "BACKEND": "django.core.cache.backends.filebased.FileBasedCache",
        "LOCATION": os.path.join(DATA_DIR, "avatar-cache"),
        "TIMEOUT": 86400,
        "OPTIONS": {"MAX_ENTRIES": 1000},
    },
}

# Store sessions in cache
SESSION_ENGINE = "django.contrib.sessions.backends.cache"
# Store messages in session
MESSAGE_STORAGE = "django.contrib.messages.storage.session.SessionStorage"

# REST framework settings for API
REST_FRAMEWORK = {
    # Use Django's standard `django.contrib.auth` permissions,
    # or allow read-only access for unauthenticated users.
    "DEFAULT_PERMISSION_CLASSES": [
        # Require authentication for login required sites
        "rest_framework.permissions.IsAuthenticated"
        if REQUIRE_LOGIN
        else "rest_framework.permissions.IsAuthenticatedOrReadOnly"
    ],
    "DEFAULT_AUTHENTICATION_CLASSES": (
        "rest_framework.authentication.TokenAuthentication",
        "weblate.api.authentication.BearerAuthentication",
        "rest_framework.authentication.SessionAuthentication",
    ),
    "DEFAULT_THROTTLE_CLASSES": (
        "weblate.api.throttling.UserRateThrottle",
        "weblate.api.throttling.AnonRateThrottle",
    ),
    "DEFAULT_THROTTLE_RATES": {"anon": "100/day", "user": "5000/hour"},
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    "DEFAULT_PAGINATION_CLASS": ("rest_framework.pagination.PageNumberPagination"),
    "PAGE_SIZE": 20,
    "VIEW_DESCRIPTION_FUNCTION": "weblate.api.views.get_view_description",
    "UNAUTHENTICATED_USER": "weblate.auth.models.get_anonymous",
}

# Fonts CDN URL
FONTS_CDN_URL = None

# Django compressor offline mode
COMPRESS_OFFLINE = False
COMPRESS_OFFLINE_CONTEXT = [
    {"fonts_cdn_url": FONTS_CDN_URL, "STATIC_URL": STATIC_URL, "LANGUAGE_BIDI":␣
↪True},
    {"fonts_cdn_url": FONTS_CDN_URL, "STATIC_URL": STATIC_URL, "LANGUAGE_BIDI":␣
↪False},
]

# Require login for all URLs
if REQUIRE_LOGIN:
    LOGIN_REQUIRED_URLS = (r"/(.*)$",)

# In such case you will want to include some of the exceptions
# LOGIN_REQUIRED_URLS_EXCEPTIONS = (
#     rf"{URL_PREFIX}/accounts/(.*)$",  # Required for login
#     rf"{URL_PREFIX}/admin/login/(.*)$",  # Required for admin login
#     rf"{URL_PREFIX}/static/(.*)$",  # Required for development mode
#     rf"{URL_PREFIX}/widgets/(.*)$",  # Allowing public access to widgets
#     rf"{URL_PREFIX}/data/(.*)$",  # Allowing public access to data exports
#     rf"{URL_PREFIX}/hooks/(.*)$",  # Allowing public access to notification hooks
#     rf"{URL_PREFIX}/healthz/$",  # Allowing public access to health check
#     rf"{URL_PREFIX}/api/(.*)$",  # Allowing access to API
#     rf"{URL_PREFIX}/js/i18n/$",  # JavaScript localization
#     rf"{URL_PREFIX}/contact/$",  # Optional for contact form
#     rf"{URL_PREFIX}/legal/(.*)$",  # Optional for legal app
# )

# Silence some of the Django system checks
SILENCED_SYSTEM_CHECKS = [
    # We have modified django.contrib.auth.middleware.AuthenticationMiddleware
    # as weblate.accounts.middleware.AuthenticationMiddleware
    "admin.E408"
]

# Celery worker configuration for testing
# CELERY_TASK_ALWAYS_EAGER = True
# CELERY_BROKER_URL = "memory://"
# CELERY_TASK_EAGER_PROPAGATES = True
# Celery worker configuration for production
CELERY_TASK_ALWAYS_EAGER = False
CELERY_BROKER_URL = "redis://localhost:6379"
CELERY_RESULT_BACKEND = CELERY_BROKER_URL

# Celery settings, it is not recommended to change these
CELERY_WORKER_MAX_MEMORY_PER_CHILD = 200000
CELERY_BEAT_SCHEDULE_FILENAME = os.path.join(DATA_DIR, "celery", "beat-schedule")
CELERY_TASK_ROUTES = {
    "weblate.trans.tasks.auto_translate": {"queue": "translate"},
    "weblate.accounts.tasks.notify_*": {"queue": "notify"},
    "weblate.accounts.tasks.send_mails": {"queue": "notify"},
    "weblate.utils.tasks.settings_backup": {"queue": "backup"},
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    "weblate.utils.tasks.database_backup": {"queue": "backup"},
    "weblate.wladmin.tasks.backup": {"queue": "backup"},
    "weblate.wladmin.tasks.backup_service": {"queue": "backup"},
}

# 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
AKISMET_API_KEY = None
```

2.18 Команды управления

Примечание: Запуск команд управления под пользователем, отличным от того, под которым работает ваш веб-сервер, может привести к тому, что файлы получат неправильные разрешения, для получения более подробной информации обратитесь к разделу *Разрешения файловой системы*.

Основные команды управления доступны из скрипта `./manage.py` в исходниках Django, или их расширенный набор из скрипта **`weblate`**, устанавливаемом поверх Weblate.

2.18.1 Вызов команд управления

Как упоминалось ранее, вызов команд зависит от того, как вы установили Weblate.

При использовании для Weblate'а виртуального окружения вы можете либо указать полный путь к команде **`weblate`**, либо перед вызовом активировать виртуальное окружение:

```
# Direct invocation
~/weblate-env/bin/weblate

# Activating virtualenv adds it to search path
. ~/weblate-env/bin/activate
weblate
```

Если вы напрямую используете исходный код (либо из tar-архива, либо из Git'а), то скрипт управления `./manage.py` доступен из исходников Weblate'а. Для его запуска выполните следующую команду:

```
python ./manage.py list_versions
```

Если вы установили Weblate с помощью программы установки pip или pip3 или скрипта `./setup.py`, команда **`weblate`** устанавливается в каталог, доступный через системную переменную окружения PATH (или через переменную PATH виртуального окружения), откуда вы можете использовать её для управления Weblate'ом:

```
weblate list_versions
```

В случае использования образа Docker'а скрипт устанавливается так же, как и выше, и вы можете запустить его с помощью команды **docker exec**:

```
docker exec --user weblate <container> weblate list_versions
```

Для команды **docker-compose** процесс аналогичен, достаточно использовать вызов **docker-compose exec**:

```
docker-compose exec --user weblate weblate weblate list_versions
```

В случае, если вам нужно передать ему файл, вы можете временно добавить том:

```
docker-compose exec --user weblate /tmp:/tmp weblate weblate importusers /tmp/
↪users.json
```

См.также:

Установка с помощью Docker'а, *Установка на Debian и Ubuntu*, *Установка на SUSE и openSUSE*, *Установка на RedHat, Fedora и CentOS*

- *Установка из исходников*, рекомендуется для разработки.

2.18.2 add_suggestions

weblate add_suggestions <project> <component> <language> <file>

Добавлено в версии 2.5.

Импортирует перевод из файла, чтобы использовать его как предложение для данного перевода. Команда пропускает дубликаты перевода; добавляются только отличающиеся.

--author USER@EXAMPLE.COM
: Электронная почта автора предложений. До выполнения импорта следует убедится, что этот пользователь существует (при необходимости его можно создать через интерфейс администратора).

Пример:

```
weblate --author michal@cihar.com add_suggestions weblate application cs /tmp/
↪suggestions-cs.po
```

2.18.3 auto_translate

weblate auto_translate <project> <component> <language>

Добавлено в версии 2.5.

Выполняет автоматический перевод на основе переводов других компонентов.

--source PROJECT/COMPONENT
: Задаёт компонент, используемый в качестве доступных для перевода исходных строк. Если не задан, то используются все компоненты проекта.

--user USERNAME
: Задаёт имя пользователя, указываемое в качестве автора переводов. Если не задан, используется «анонимный пользователь».

--overwrite
: Перезаписывать ли существующие переводы.

--inconsistent
: Перезаписывать ли существующие переводы с непоследовательными переводами (смотрите проверку *Противоречия*).

--add
Автоматически добавлять язык, если указанный перевод не существует.

--mt MT
В качестве машинного перевода использовать машинный перевод, а не перевод из других компонентов.

--threshold THRESHOLD
Порог сходства для машинного перевода, по умолчанию установлен в 80.

Пример:

```
weblate auto_translate --user nijel --inconsistent --source weblate/application␣
↪weblate website cs
```

См.также:

Автоматический перевод

2.18.4 celery_queues

weblate celery_queues

Добавлено в версии 3.7.

Отображает длину очереди задач Celery.

2.18.5 checkgit

weblate checkgit <project|project/component>

Распечатывает текущее состояние внутреннего репозитория Git.

Вы можете либо задать конкретный проект или компонент для обновления (например, `weblate/application`), либо использовать ключ `--all` для обновления всех существующих компонентов.

2.18.6 commitgit

weblate commitgit <project|project/component>

Коммитит все возможные отложенные изменения во внутренний репозиторий Git.

Вы можете либо задать конкретный проект или компонент для обновления (например, `weblate/application`), либо использовать ключ `--all` для обновления всех существующих компонентов.

2.18.7 commit_pending

weblate commit_pending <project|project/component>

Коммитит отложенные изменения старше указанного возраста.

Вы можете либо задать конкретный проект или компонент для обновления (например, `weblate/application`), либо использовать ключ `--all` для обновления всех существующих компонентов.

--age HOURS
Возраст в часах для коммита. Если не указан, то используется значение, настроенное в *конфигурации компонента*.

Примечание: Она автоматически выполняется Weblate'ом в фоновом режиме, поэтому вручную вызывать эту команду нет никакой реальной необходимости, кроме разве что принудительного выполнения коммита в более раннее время, чем задано в *конфигурации компонента*.

См.также:

Выполнение задач технического обслуживания, `COMMIT_PENDING_HOURS`

2.18.8 cleanuptrans

`weblate cleanuptrans`

Очищает осиротевшие проверки и предложения перевода. Обычно запускать её вручную нет необходимости, поскольку очистка происходит автоматически в фоновом режиме.

См.также:

Выполнение задач технического обслуживания

2.18.9 createadmin

`weblate createadmin`

Создаёт учётную запись `admin` со случайным паролем, если он не указан явно.

`--password` `PASSWORD`
Взять пароль из командной строки, случайный пароль в этом случае не генерируется.

`--no-password`
Не устанавливать пароль, это может быть полезно при запуске с ключом `--update`.

`--username` `USERNAME`
Использовать указанное имя вместо `admin`.

`--email` `USER@EXAMPLE.COM`
Указывает адрес электронной почты администратора.

`--name`
Указывает (отображаемое) имя администратора.

`--update`
Обновление существующего пользователя (вы можете использовать этот ключ для изменения паролей).

Изменено в версии 2.9: Добавлены параметры `--username`, `--email`, `--name` и `--update`.

2.18.10 dump_memory

`weblate dump_memory`

Добавлено в версии 2.20.

Экспортирует JSON-файл, содержащий содержимое памяти переводов Weblate'а.

См.также:

Память переводов, *Схема памяти переводов Weblate'а*

2.18.11 dumpuserdata

weblate dumpuserdata <file.json>

Сбрасывает данные пользователя в файл для последующего их использования командой *importuserdata*

Подсказка: Она пригодится при переносе или слиянии экземпляров Weblate'а.

2.18.12 import_demo

weblate import_demo

Добавлено в версии 4.1.

Создаёт демонстрационный проект с компонентами на основе проекта <https://github.com/WeblateOrg/demo>.

Может быть полезна при разработке Weblate'а.

2.18.13 import_json

weblate import_json <json-file>

Добавлено в версии 2.7.

Пакетный импорт компонентов, основанных на данных из JSON.

Структура JSON импортируемого файла в значительной степени соответствует объекту компонента (смотрите *GET /api/components/(string:project)/(string:component)/*). Вы должны включить поля `name` и `filemask`.

--project `PROJECT`
Указывает, откуда будут импортироваться компоненты.

--main-component `COMPONENT`
Использовать указанный репозиторий системы контроля версий из этого компонента для всех остальных.

--ignore
Пропускать (уже) импортированные компоненты.

--update
Обновлять (уже) импортированные компоненты.

Изменено в версии 2.9: Параметры `--ignore` и `--update` существуют для работы с уже импортированными компонентами.

Пример JSON-файла:

```
[
  {
    "slug": "po",
    "name": "Gettext PO",
    "file_format": "po",
    "filemask": "po/*.po",
    "new_lang": "none"
  },
  {
    "name": "Android",
    "filemask": "android/values-*/strings.xml",
    "template": "android/values/strings.xml",
    "repo": "weblate://test/test",
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    "file_format": "aresource"
  }
]
```

См.также:

import_memory

2.18.14 import_memory

weblate import_memory <file>

Добавлено в версии 2.20.

Импортирует файл TMX или JSON в память переводов Weblate'а.

--language-map LANGMAP

Позволяет отобразить языки из TMX на память переводов Weblate'а. Коды языков отображаются после выполняемой Weblate'ом нормализации.

Например, `--language-map ru_RU:ru` импортирует все строки с языком `ru_RU` как строки с языком `ru`.

Это может быть полезно в том случае, если локали в ваших файлах TMX не совпадают с теми, которые вы используете в Weblate'е.

См.также:

Память переводов, *Схема памяти переводов Weblate'а*

2.18.15 import_project

weblate import_project <project> <gitrepo> <branch> <filemask>

Изменено в версии 3.0: Команда import_project теперь основана на надстройке *Обнаружение компонента*, что привело к некоторым изменениям в поведении и принимаемых параметрах.

Пакетный импорт компонентов в проект на основе маски файлов.

Параметр *<project>* определяет имя существующего проекта, в который будут импортированы компоненты.

Параметр *<gitrepo>* определяет URL-адрес используемого Git-репозитория, а *<branch>* — его ветку. Для импорта дополнительные компонентов перевода из существующего компонента Weblate'а используйте для параметра *<gitrepo>* адрес вида *weblate://<проект>/<компонент>*.

Параметр *<filemask>* определяет маску искомых файлов в репозитории. Он может быть либо простой строкой с подстановочными знаками, либо использовать всю мощь регулярных выражений.

Простое сопоставление использует `**` для имени компонента и `*` для языка, например: `**/*.po`

Регулярное выражение должно содержать группы с именами *component* и *language*. Например: `(?P<language>[^/]*)/(?P<component>[^-/]*)\.po`

Импорт находит сопоставление с существующими компонентами на основе файлов, и добавляет не найденные компоненты. Уже существующие компоненты он не меняет.

--name-template TEMPLATE

Настраивает имя компонента, используя синтаксис шаблонов Django.

Например: `Документация: {{ component }}`

--base-file-template TEMPLATE

Настраивает базовый файл для одноязычных переводов.

Например: `{{ component }}/res/values/string.xml`

--new-base-template TEMPLATE
Настраивает базовый файл для добавления новых переводов.

Например: `{{ component }}/ts/en.ts`

--file-format FORMAT
Также вы можете указать формат файла (смотрите раздел *Поддерживаемые форматы файлов*), по умолчанию используется автоматическое определение.

--language-regex REGEX
С помощью этого параметра можно указать языковой фильтр (смотрите раздел *Конфигурация компонента*). Должен быть правильным регулярным выражением.

--main-component
Вы можете указать, какой компонент будет выбран в качестве основного — тот, который фактически содержит репозиторий системы контроля версий.

--license NAME
Указывает общую лицензию, лицензию проекта или компонента перевода.

--license-url URL
Указывает URL-адрес, по которому может быть найдена лицензия перевода.

--vcs NAME
В случае, если вам необходимо указать, какую использовать систему контроля версий, вы можете сделать это здесь. Умолчательная система контроля версий — Git.

Чтобы привести вам несколько примеров, давайте попробуем импортировать два проекта.

Первый — перевод Руководства Debian, в котором для каждого языка имеется отдельный каталог с переводами каждой главы:

```
weblate import_project \
    debian-handbook \
    git://anonscm.debian.org/debian-handbook/debian-handbook.git \
    squeeze/master \
    '*/**.po'
```

Далее инструмент Tanaguru, для которого необходимо указать формат файла, шаблон базового файла, а также то, что все компоненты и переводы расположены в одном каталоге:

```
weblate import_project \
    --file-format=properties \
    --base-file-template=web-app/tgol-web-app/src/main/resources/i18n/%s-I18N.
↪properties \
    tanaguru \
    https://github.com/Tanaguru/Tanaguru \
    master \
    web-app/tgol-web-app/src/main/resources/i18n/**-I18N_*.properties
```

Более сложный пример разбора имён файлов для получения правильного компонента и языка из имени файла вида `src/security/Numerous_security_holes_in_0.10.1.de.po`:

```
weblate import_project \
    tails \
    git://git.tails.boum.org/tails master \
    'wiki/src/security/(?P<component>.*)\.(?P<language>[^.]*)\.po$'
```

Отфильтровывание переводов только на выбранные языки:

```
./manage import_project \
    --language-regex '^(cs|sk)$' \
    weblate \
    https://github.com/WeblateOrg/weblate.git \
    'weblate/locale/*/LC_MESSAGES/**.po'
```

Импорт Sphinx-документации, разбитой на несколько файлов:

```
$ weblate import_project --name-template 'Documentation: %s' \
    --file-format po \
    project https://github.com/project/docs.git master \
    'docs/locale/*/LC_MESSAGES/**.po'
```

Импорт Sphinx-документации, разбитой на несколько файлов и каталогов:

```
$ 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'
```

См.также:

Более подробные примеры можно найти в главе *Знакомство с интернационализацией*, в качестве альтернативы вы можете использовать команду `import_json`.

2.18.16 importuserdata

`weblate importuserdata <file.json>`

Импортирует данные пользователя из файла, созданного командой `dumpuserdata`

2.18.17 importusers

`weblate importusers --check <file.json>`

Импортирует пользователей из JSON-дампа Django базы данных auth_users.

`--check`
С этим параметром она просто проверит, можно ли импортировать данный файл и сообщит о возможных конфликтах, связанных с именами пользователей или их адресами электронной почты.

Выгрузить пользователей из существующей установки Django вы можете при помощи команды:

```
weblate dumpdata auth.User > users.json
```

2.18.18 install_addon

Добавлено в версии 3.2.

`weblate install_addon --addon ADDON <project|project/component>`

Installs an addon to a set of components.

`--addon` `ADDON`
Названия устанавливаемого дополнения. Например, `weblate.gettext.customize`.

`--configuration` `CONFIG`
JSON encoded configuration of an addon.

`--update`
Обновляет существующую конфигурацию надстройки.

You can either define which project or component to install the addon in (for example `weblate/application`), or use `--all` to include all existing components.

Для установки надстройки *Настройка вывода gettext* для всех компонентов:

```
weblate install_addon --addon weblate.gettext.customize --config '{"width": -1}' --
↪update --all
```

См.также:

Надстройки

2.18.19 list_languages

`weblate list_languages <locale>`

Перечисляет поддерживаемые языки в разметке MediaWiki — коды языков, английские названия и локализованные названия.

Используется для генерации словаря с названиями языков.

2.18.20 list_translators

`weblate list_translators <project|project/component>`

Показывает переводчиков по языкам проекта:

```
[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.

2.18.21 list_versions

`weblate list_versions`

Lists all Weblate dependencies and their versions.

2.18.22 loadpo

`weblate loadpo <project|project/component>`

Reloads translations from disk (for example in case you have done some updates in the VCS repository).

`--force`
Force update, even if the files should be up-to-date.

`--lang` `LANGUAGE`
Limit processing to a single language.

Вы можете либо задать конкретный проект или компонент для обновления (например, `weblate/application`), либо использовать ключ `--all` для обновления всех существующих компонентов.

Примечание: You seldom need to invoke this, Weblate will automatically load changed files for every VCS update. This is needed in case you manually changed an underlying Weblate VCS repository or in some special cases following an upgrade.

2.18.23 lock_translation

weblate lock_translation <project|project/component>

Запрещает последующий перевод компонента.

Подсказка: Useful in case you want to do some maintenance on the underlying repository.

Вы можете либо задать конкретный проект или компонент для обновления (например, `weblate/application`), либо использовать ключ `--all` для обновления всех существующих компонентов.

См.также:

unlock_translation

2.18.24 move_language

weblate move_language source target

Добавлено в версии 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.

Пример:

```
weblate move_language cze cs
```

After moving the content, you should check whether there is anything left (this is subject to race conditions when somebody updates the repository meanwhile) and remove the *(generated)* language.

2.18.25 pushgit

weblate pushgit <project|project/component>

Pushes committed changes to the upstream VCS repository.

--force-commit
Force commits any pending changes, prior to pushing.

Вы можете либо задать конкретный проект или компонент для обновления (например, `weblate/application`), либо использовать ключ `--all` для обновления всех существующих компонентов.

Примечание: Weblate pushes changes automatically if *Push on commit* in *Конфигурация компонента* is turned on, which is the default.

2.18.26 unlock_translation

weblate unlock_translation <project|project/component>

Разблокировывает данный компонент, делая его доступным для перевода.

Подсказка: Useful in case you want to do some maintenance on the underlying repository.

Вы можете либо задать конкретный проект или компонент для обновления (например, `weblate/application`), либо использовать ключ `--all` для обновления всех существующих компонентов.

См.также:

lock_translation

2.18.27 setupgroups

weblate setupgroups

Configures default groups and optionally assigns all users to that default group.

--no-privs-update
Отключает автообновление существующих групп (только добавляет новые).

--no-projects-update
Prevents automatic updates of groups for existing projects. This allows adding newly added groups to existing projects, see *По-проектный контроль доступа*.

См.также:

Управление доступом

2.18.28 setuplang

weblate setuplang

Обновляет список определенных языков в Weblate.

--no-update
Отключает автообновление существующих языков (только добавляет новые).

2.18.29 updatechecks

weblate updatechecks <project|project/component>

Updates all checks for all strings.

Подсказка: Useful for upgrades which do major changes to checks.

Вы можете либо задать конкретный проект или компонент для обновления (например, `weblate/application`), либо использовать ключ `--all` для обновления всех существующих компонентов.

2.18.30 updategit

`weblate updategit <project|project/component>`

Fetches remote VCS repositories and updates the internal cache.

Вы можете либо задать конкретный проект или компонент для обновления (например, `weblate/application`), либо использовать ключ `--all` для обновления всех существующих компонентов.

Примечание: Usually it is better to configure hooks in the repository to trigger *Обработчики уведомлений*, instead of regular polling by `updategit`.

2.19 Объявления

Изменено в версии 4.0: В предыдущих выпусках эта функция называлась «доской сообщений».

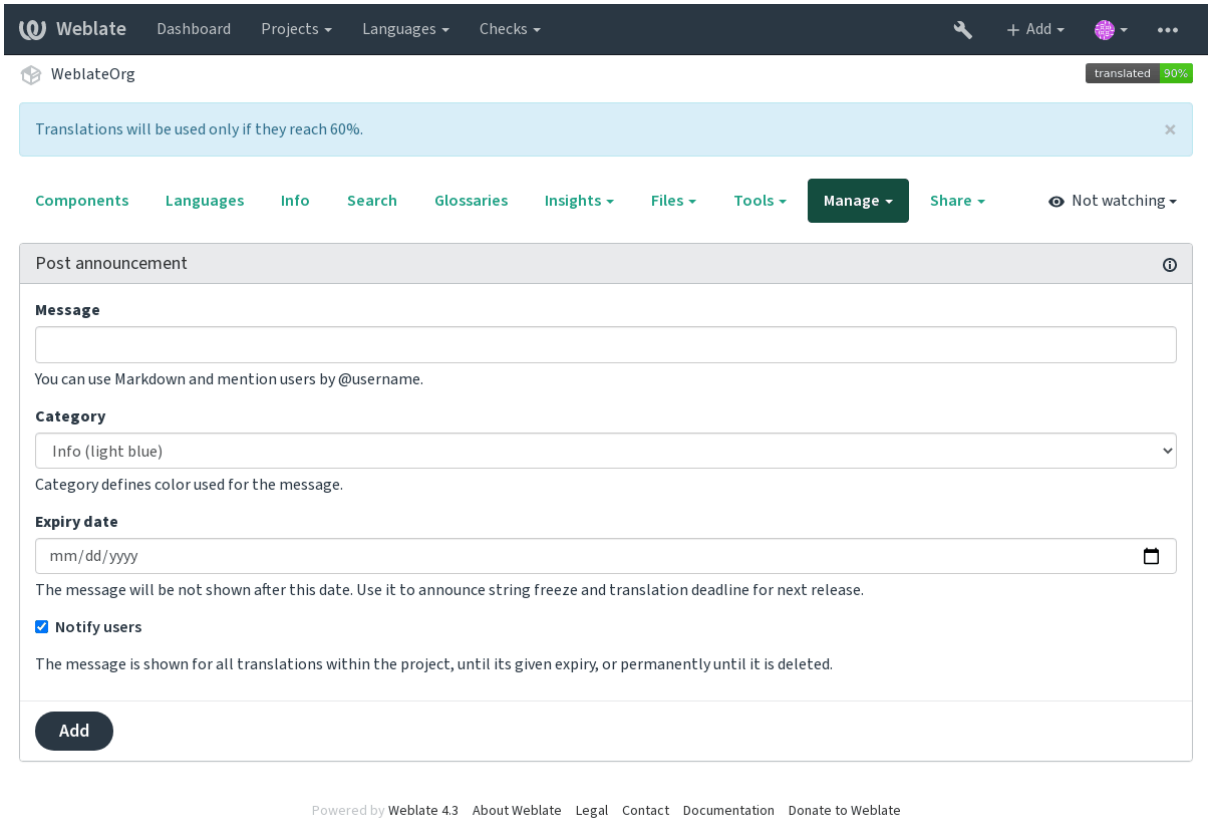
Предоставляет информацию вашим переводчикам путём размещения объявлений для всего сайта, проекта, компонента или языка.

Объявите цель, сроки, статус или укажите цели перевода.

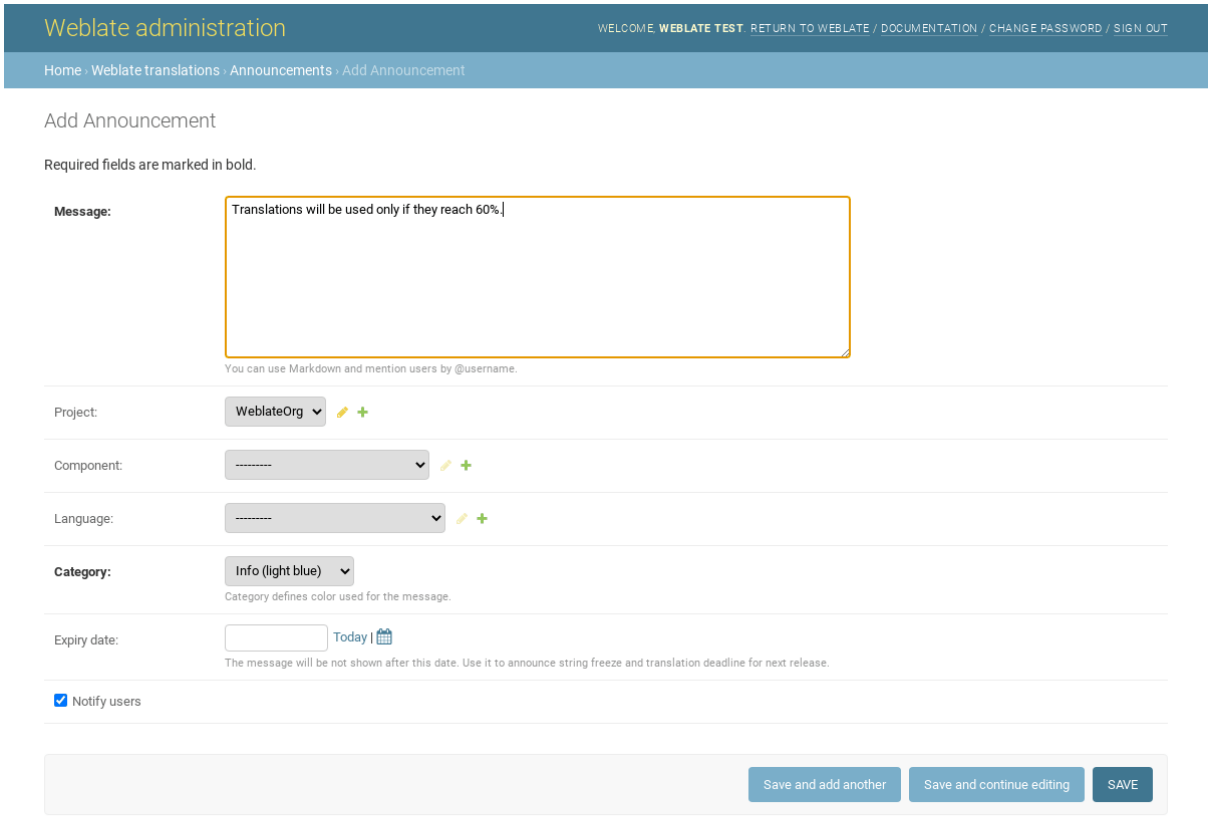
Пользователи будут получать уведомления об объявлениях в отслеживаемых ими проектах (если только они не откажутся от получения таких уведомлений).

Это может быть полезно для различных вещей, начиная от объявления намерений веб-сайта и заканчивая указанием целей для перевода.

Объявления могут быть размещены на каждом уровне через меню *Управлять* при помощи его пункта *Опубликовать объявление*:



Также его можно добавить через интерфейс администратора:



Объявления показываются в соответствии с заданным для них контекстом:

Контекст не указан

Показываются в панели управления (на странице входа).

Указан проект

Показываются внутри проекта, включая все его компоненты и переводы.

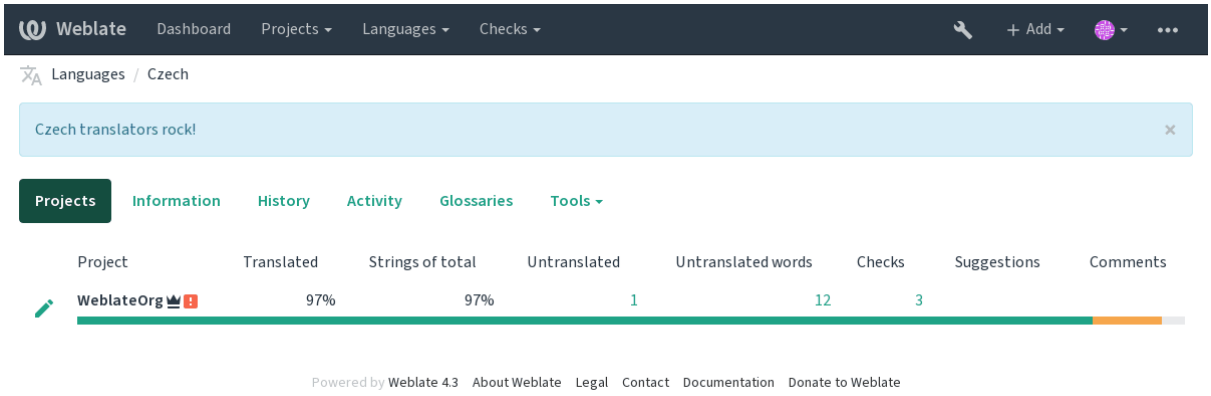
Указан компонент

Показываются для данного компонента и всех его переводов.

Указан язык

Показываются в обзоре по языку и во всех переводах на этот язык.

Вот как оно выглядит на странице обзора языка:



2.20 Списки компонентов

Укажите несколько списков компонентов, чтобы они появились в качестве вариантов выбора на панели управления пользователя, из которых пользователи смогут выбрать один из них, как своё умолчательное представление панели. Для получения более подробной информации смотрите раздел *Панель управления*.

Изменено в версии 2.20: Для каждого списка компонентов, присутствующего в панели управления, будет представлен свой статус.

Названия и содержимое списков компонентов можно указать в интерфейсе администратора в разделе *Списки компонентов*. Каждый список компонентов должен иметь название, отображаемое пользователю, и плашку, представляющую его в URL-адресе.

Изменено в версии 2.13: Для анонимных пользователей настройки панели управления изменяются из интерфейса администратора, в котором определяется, какая панель управления будет показана неаутентифицированным пользователям.

2.20.1 Автоматические списки компонентов

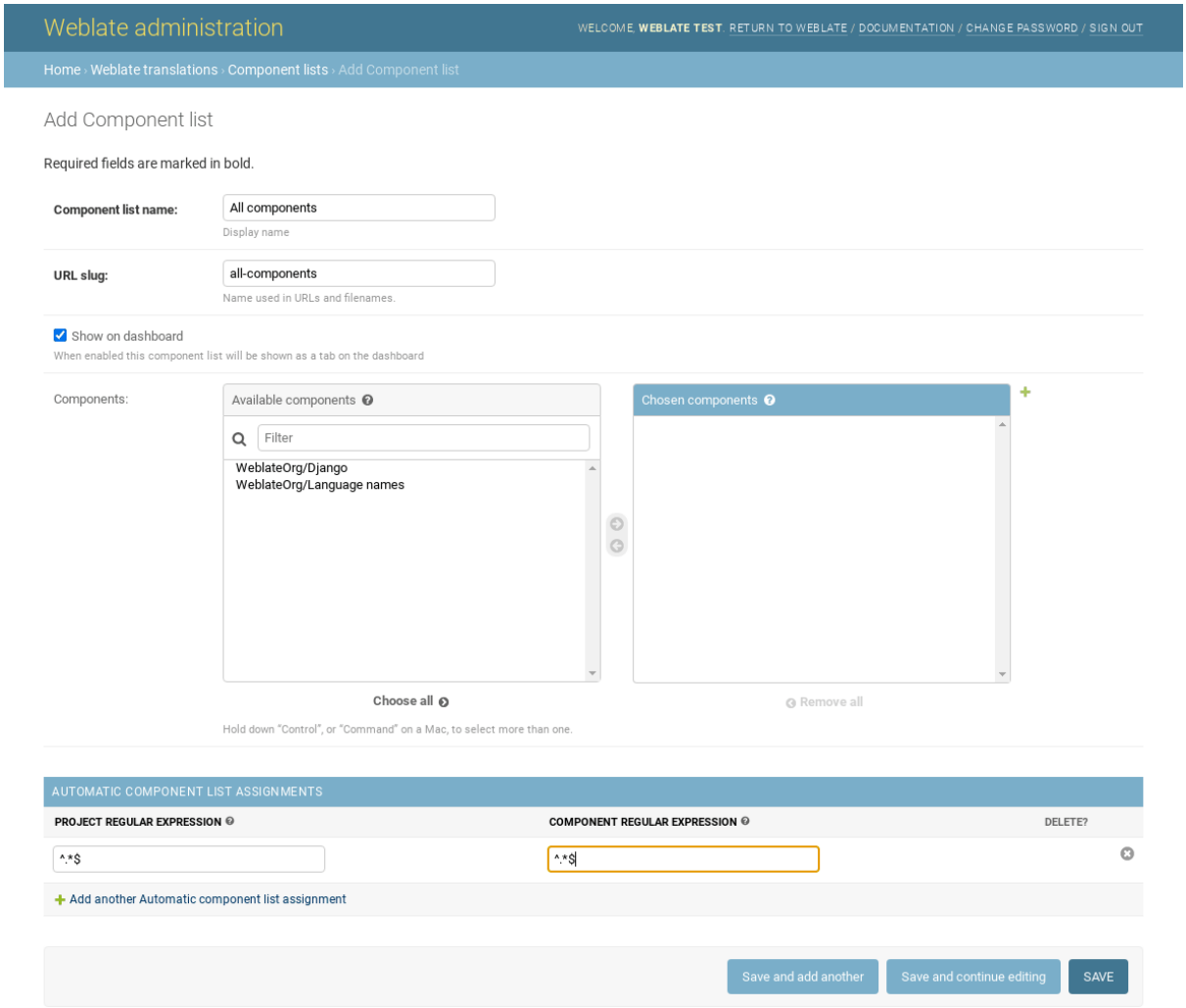
Добавлено в версии 2.13.

Автоматическое добавление компонентов в список на основе их плашки настраивается созданием правил *Автоматического прикрепления списка компонентов*.

- Полезно для ведения списков компонентов для крупных установок, либо в случае, если вы хотите иметь в своей установке Weblate один список компонентов со всеми компонентами.

Подсказка: Создайте список компонентов, содержащий все компоненты вашей установки 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 Дополнительные модули Weblate

Для вашей установки доступно несколько дополнительных модулей.

2.21.1 Экспортер Git

Добавлено в версии 2.10.

Предоставляет доступ на чтение к нижележащему репозиторию Git через HTTP(S).

Установка

1. Добавьте `weblate.gitexport` в установленные приложения в файле `settings.py`:

```
INSTALLED_APPS += (
    'weblate.gitexport',
)
```

2. Экспортируйте существующие репозитории, перенеся свою базу данных после установки:

```
weblate migrate
```

Использование

The module automatically hooks into Weblate and sets the exported repository URL in the *Конфигурация компонента*. The repositories are accessible under the `/git/` part of the Weblate URL, for example `https://example.org/git/weblate/master/`:

```
git clone 'https://example.org/git/weblate/master/'
```

Repositories are available anonymously unless *По-проектный контроль доступа* is turned on. This requires authenticate using your API token (it can be obtained in your *Профиль пользователя*):

```
git clone 'https://user:KEY@example.org/git/weblate/master/'
```

2.21.2 Тарифный план

Добавлено в версии 2.4.

This is used on Hosted Weblate to define billing plans, track invoices and usage limits.

Установка

Добавьте `weblate.billing` в установленные приложения в файле `settings.py`:

```
INSTALLED_APPS += (
    'weblate.billing',
)
```

2. Запустите миграцию базы данных для необязательной установки для модуля дополнительных структур базы данных:

```
weblate migrate
```

Использование

После установки вы сможете в интерфейсе администратора управлять тарификацией. У пользователей с включённой функцией тарификации в их *профиле* появится новая вкладка *Тарифный план*.

The billing module additionally allows project admins to create new projects and components without being superusers (see *Добавление проектов и компонентов перевода*). 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 Право

Добавлено в версии 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 Документ условий обслуживания

legal/documents/privacy.html Документ политики конфиденциальности

legal/documents/summary.html Краткий обзор условий обслуживания и политики конфиденциальности

Примечание: Legal documents for the Hosted Weblate service is available in this Git repository <https://github.com/WeblateOrg/hosted/tree/master/wlhosted/legal/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.

Установка

1. Add `weblate.legal` to installed apps in `settings.py`:

```
INSTALLED_APPS += (
    'weblate.legal',
)

# Optional:

# Social auth pipeline to confirm TOS upon registration/subsequent login
SOCIAL_AUTH_PIPELINE += (
    'weblate.legal.pipeline.tos_confirm',
)

# Middleware to enforce TOS confirmation of signed in users
MIDDLEWARE += [
    'weblate.legal.middleware.RequireTOSMiddleware',
]
```

2. Запустите миграцию базы данных для необязательной установки для модуля дополнительных структур базы данных:

```
weblate migrate
```

3. Отредактируйте юридические документы в каталоге `weblate/legal/templates/legal/` в соответствии с вашими условиями.

Использование

После установки и редактирования юридические документы будут отображаться в пользовательском интерфейсе Weblate.

2.21.4 Аватары

Аватары загружаются и кэшируются на стороне сервера, чтобы уменьшить утечку информации на сайты, обслуживающие их по умолчанию. Встроенную поддержку получения аватаров из настроенных для этого адресов электронной почты можно отключить с помощью параметра `ENABLE_AVATARS`.

В настоящее время Weblate поддерживает:

- Gravatar

См.также:

Кеширование аватара, `AVATAR_URL_PREFIX`, `ENABLE_AVATARS`

2.21.5 Защита от спама

Вы можете защититься от спама в предложениях от неаутентифицированных пользователей, воспользовавшись сервисом akismet.com.

1. Установите модуль Python *akismet*
2. Настройте ключ API Akismet.

Примечание: Защита (среди всего прочего) зависит от IP-адреса клиента, для правильной её настройки, пожалуйста, смотрите раздел *Работа за обратным прокси*.

См.также:

Работа за обратным прокси, `AKISMET_API_KEY`

2.21.6 Подписание коммитов Git с помощью GnuPG

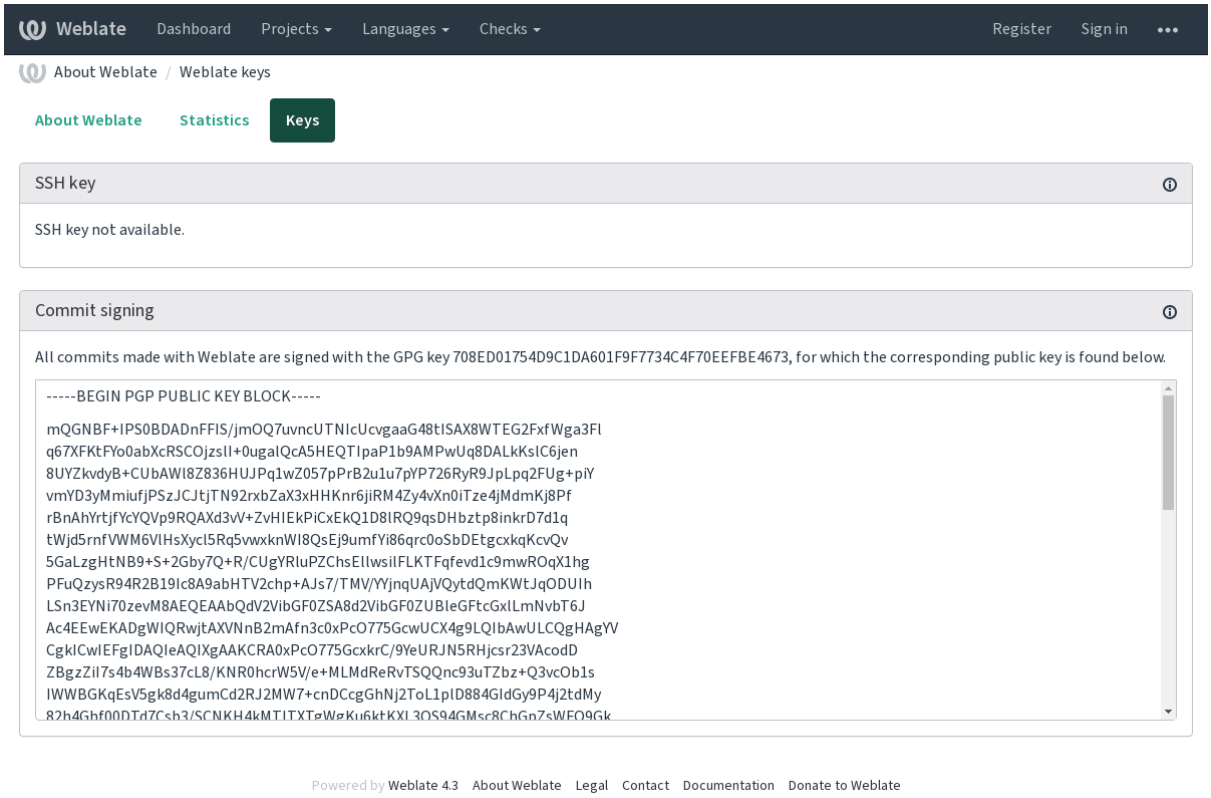
Добавлено в версии 3.1.

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.)

Эта функция требует установки GnuPG 2.1 или новее.

You can find the key in the `DATA_DIR` and the public key is shown on the «About» page:



2. Alternatively you can also import existing keys into Weblate, just set `HOME=$DATA_DIR/home` when invoking gpg.

См.также:

WEBLATE_GPG_IDENTITY

2.21.7 Ограничение частоты запросов

Изменено в версии 3.2: The rate limiting now accepts more fine-grained configuration.

Several operations in Weblate are rate limited. At most *RATELIMIT_ATTEMPTS* attempts are allowed within *RATELIMIT_WINDOW* seconds. The user is then blocked for *RATELIMIT_LOCKOUT*. There are also settings specific to scopes, for example `RATELIMIT_CONTACT_ATTEMPTS` or `RATELIMIT_TRANSLATE_ATTEMPTS`. The table below is a full list of available scopes.

The following operations are subject to rate limiting:

Имя	Область	Allowed attempts	Ratelimit window	Lockout period
Регистрация	`REGISTRATION`	5	300	600
Sending message to admins	`MESSAGE`	5	300	600
Password authentication on login	`LOGIN`	5	300	600
Sitewide search	`SEARCH`	6	60	60
Перевод	`TRANSLATE`	30	60	600
Adding to glossary	`GLOSSARY`	30	60	600

If a user fails to log in *AUTH_LOCK_ATTEMPTS* times, password authentication will be turned off on the account until having gone through the process of having its password reset.

См.также:

Ограничение частоты запросов, *Работа за обратным прокси*

2.22 Настройка Weblate

Расширяйте и настраивайте Weblate, используя Django и Python. Вносите свои изменения в основной репозиторий Weblate'а, чтобы выгоду из них мог извлечь каждый. Это снижает ваши затраты на обслуживание; задачу подержания актуальности кода Weblate'а при изменении внутренних интерфейсов или рефакторинге берут на себя его сопровождающие.

Предупреждение: Ни внутренние интерфейсы, ни шаблоны не считаются стабильным API. Пожалуйста, при каждом обновлении просматривайте внесённые вами изменения, интерфейсы или их семантика могут измениться без предварительного уведомления.

См.также:

Внесение вклада в Weblate

2.22.1 Создание модуля Python

Если вы не знакомы с Python'ом, возможно, вы захотите посмотреть статью Python для начинающих, в которой объясняются основы и указаны дальнейшие учебные пособия.

При написании собственного кода на Python (называемого модулем) его необходимо разместить в каталоге, находящемся либо где-то в системном пути (обычно это путь вида `/usr/lib/python3.7/site-packages/`), либо в каталоге Weblate'а, который также добавляется к путям поиска интерпретатора.

А ещё лучше превратить вашу настройку в правильный пакет Python:

1. Создайте каталог для вашего пакета (мы будем использовать *weblate_customization*).
2. В нём создайте описывающий пакет файл `setup.py`:

```
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. Создайте каталог для модуля Python (также назовём его `weblate_customization`), в котором будет содержаться наш код.
4. В нём создайте файл `__init__.py`, который гарантирует, что Python сможет импортировать модуль.
5. Теперь этот пакет можно установить командой **`pip install -e`**. Более подробную информацию можно найти в документе «Редактируемые» установки.
6. После установки модуль можно использовать в конфигурации Weblate'а (например, `weblate_customization.checks.FooCheck`).

Структура вашего модуля должна выглядеть примерно так:

```
weblate_customization
├── setup.py
└── weblate_customization
    ├── __init__.py
    ├── addons.py
    └── checks.py
```

Пример настройки Weblate'а можно найти по адресу <https://github.com/WeblateOrg/customize-example>, он охватывает все описанные ниже темы.

2.22.2 Изменение логотипа

1. Create a simple Django app containing the static files you want to overwrite (see *Создание модуля Python*).
2. Добавьте его в `INSTALLED_APPS`:

```
INSTALLED_APPS = (
   # Add your customization as first
   'weblate_customization',

   # Weblate apps are here…
)
```

Брендинг присутствует в следующих файлах:

`icons/weblate.svg` Логотип, показываемый в навигационной панели.

`logo-*.png` Веб-иконки для разных разрешений экрана и веб-браузеров.

`favicon.ico` Веб-иконка, используемая устаревшими браузерами.

`weblate-*.png` Аватары ботов или анонимных пользователей. Некоторые веб-браузеры используют их в качестве значков ярлыков.

`email-logo.png` Используется в уведомлениях по электронной почте.

3. Запустите `weblate collectstatic --noinput`, чтобы собрать статические файлы для раздачи их клиентам.

См.также:

Управление статическими файлами (изображениями, JavaScript, CSS и прочими подобными), *Обслуживание статических файлов*

2.22.3 Пользовательские проверки качества, надстройки и автоматические исправления

Для установки в Weblate собственного кода для *автоматических исправлений*, *проверок* или *надстроек*:

1. Разместите файлы в модуле Python, содержащем настройку Weblate'а (смотрите раздел *Создание модуля Python*).
2. Добавьте полное имя его класса Python в список соответствующего параметра (*`WEBLATE_ADDONS`*, *`CHECK_LIST`* или *`AUTOFIX_LIST`*):

```
# Checks
CHECK_LIST += (
    'weblate_customization.checks.FooCheck',
)

# Autofixes
AUTOFIX_LIST += (
```

(continues on next page)

(продолжение с предыдущей страницы)

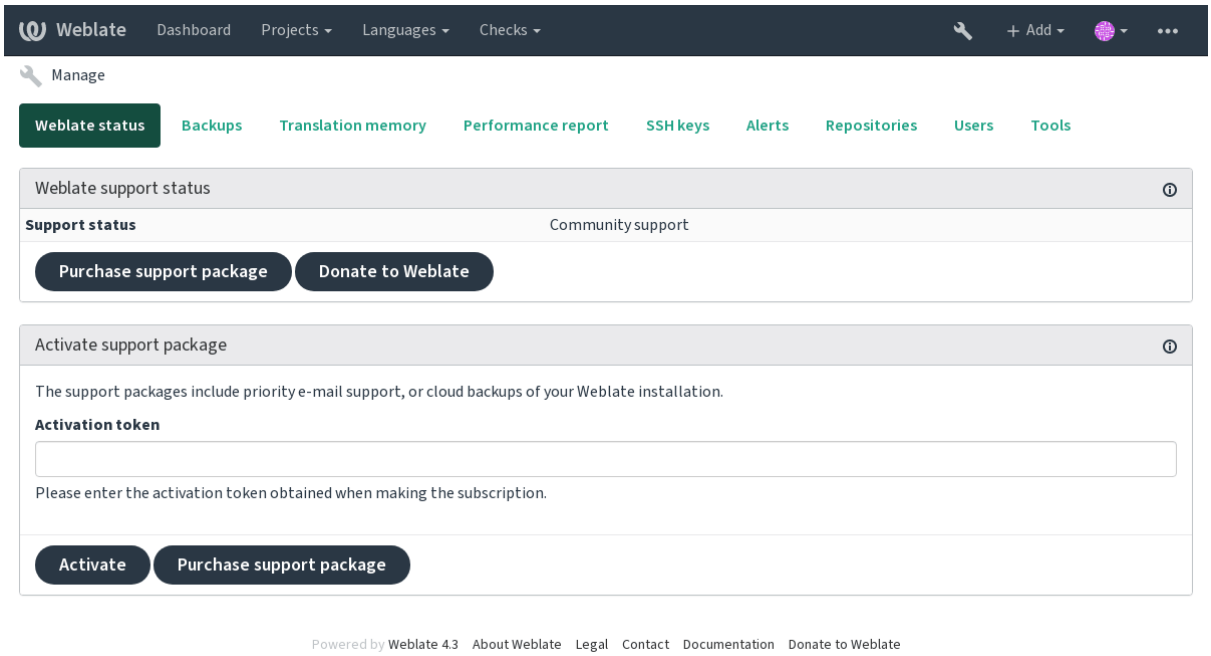
```
  'weblate_customization.autofix.FooFixer',
)

# Addons
WEBLATE_ADDONS += (
  'weblate_customization.addons.ExamplePreAddon',
)
```

См.также:

Пользовательские автоматические исправления, *Написание собственных проверок*, *Написание надстройки*, *Выполнение скриптов из надстройки*

2.23 Интерфейс управления

Интерфейс управления предлагает административные настройки по адресу `/management/`. Он доступен для пользователей, вошедших в систему с правами администратора, доступ к нему осуществляется через иконку гаечного ключа справа вверху:



2.23.1 Интерфейс администратора Django

Предупреждение: В будущем будет удалён, поскольку его использование не рекомендуется — большинством возможностей можно управлять непосредственно из Weblate'а.

Здесь вы можете управлять объектами, хранящимися в базе данных, такими как пользователи или переводы, а также другими настройками:

Weblate administration

WELCOME, **WEBLATE TEST**. RETURN TO WEBLATE / DOCUMENTATION / CHANGE PASSWORD / SIGN OUT

Site administration

REPORTS		
Weblate support status		
Status of repositories		
SSH keys		
Performance report		
Translation memory		

ACCOUNTS		
Audit logs	+ Add	Change
Profiles	+ Add	Change
Verified emails	+ Add	Change

AUTH TOKEN		
Tokens	+ Add	Change

AUTHENTICATION		
Groups	+ Add	Change
Roles	+ Add	Change
Users	+ Add	Change

BILLING		
Billings	+ Add	Change
Invoices	+ Add	Change
Plans	+ Add	Change

FONTS		
Font groups	+ Add	Change
Fonts	+ Add	Change

GLOSSARIES		
Glossaries	+ Add	Change

LEGAL		
Agreements	+ Add	Change

PYTHON SOCIAL AUTH		
Associations	+ Add	Change
Nonces	+ Add	Change
User social auths	+ Add	Change

SCREENSHOTS		
Screenshots	+ Add	Change

TRANSLATION MEMORY		
Memorys	+ Add	Change

WEBLATE LANGUAGES		
Languages	+ Add	Change

WEBLATE TRANSLATIONS		
Announcements	+ Add	Change
Component lists	+ Add	Change
Components	+ Add	Change
Contributor agreements	+ Add	Change
Projects	+ Add	Change

Recent actions

My actions

None available

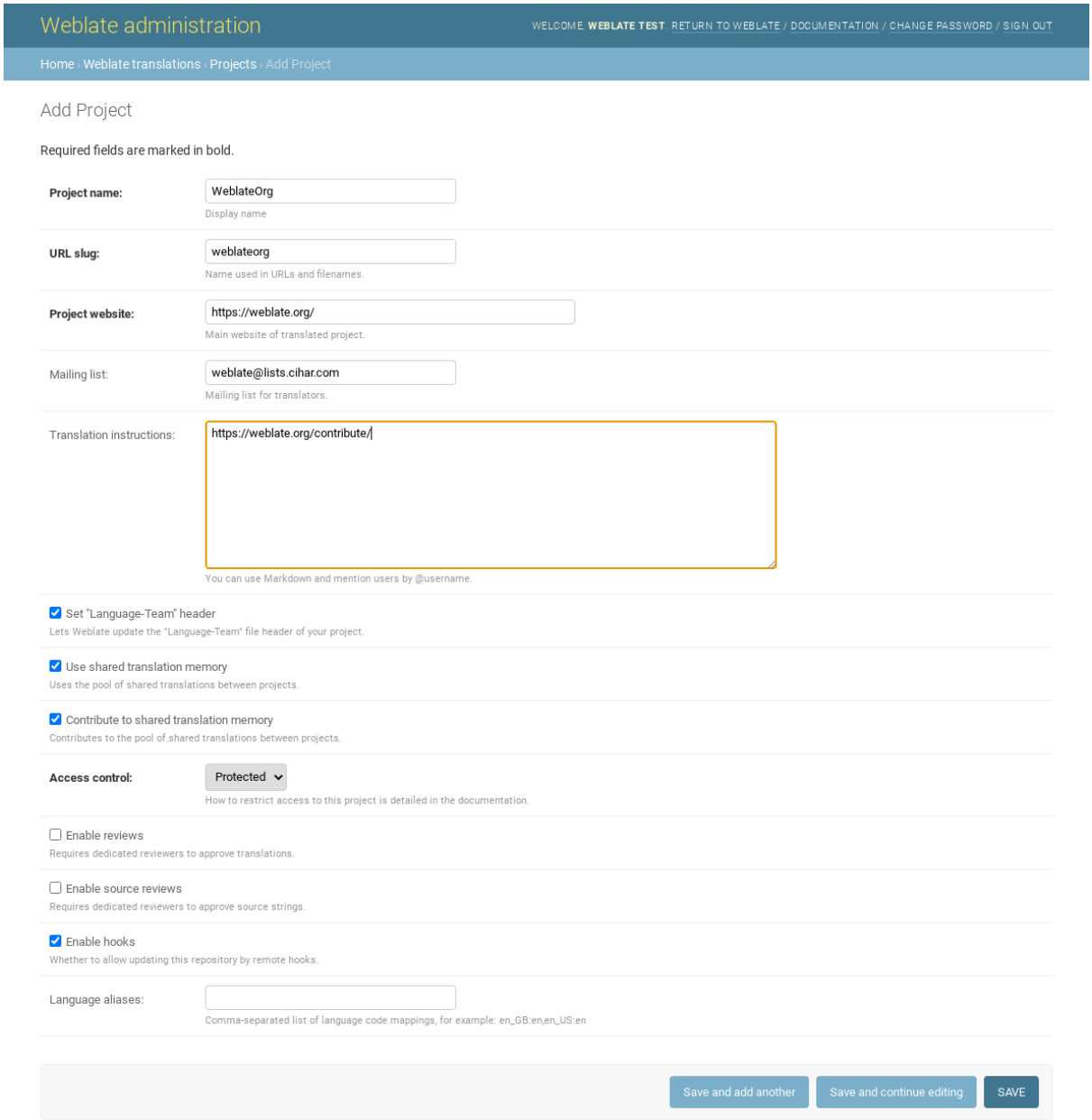
В разделе *Отчёты* вы можете проверить статус вашего сайта, настроить его для *рабочей среды*, или управлять ключами SSH, используемыми для *доступа к репозиториям*.

Управление объектами базы данных под любым разделом. Самым интересным из них является, наверное, *Переводы Weblate'а*, в котором можно управлять переводимыми проектами, смотрите разделы *Конфигурация проекта* и *Конфигурация компонента*.

Раздел *Языки Weblate* содержит определения языков, более подробное объяснение смотрите в разделе *Определения языка*.

Добавление проекта

Добавление проекта служит контейнером для всех компонентов. Обычно вы создаёте один проект для одной части программного обеспечения или книги (для справки по параметрам смотрите раздел *Конфигурация проекта*):



См.также:

Конфигурация проекта

Двуязычные компоненты

После добавления проекта в него можно добавить компоненты перевода (для справки по параметрам смотрите раздел *Конфигурация компонента*):

Weblate administration

WELCOME, **WEBLATE TEST**. RETURN TO WEBLATE / DOCUMENTATION / CHANGE PASSWORD / SIGN OUT

Home › Weblate translations › Components › Add Component

Add Component

IMPORT SPEED DOCUMENTATION

Required fields are marked in bold.

Component name: Language names
Display name

URL slug: language-names
Name used in URLs and filenames.

Project: WeblateOrg

Version control system: Git
Version control system to use to access your repository containing translations. You can also choose additional integration with third party providers to submit merge requests.

Source code repository: https://github.com/WeblateOrg/demo.git
URL of a repository, use weblate://project/component to share it with other component.

Repository push URL:
URL of a push repository, pushing is turned off if empty.

Repository browser: https://github.com/WeblateOrg/demo/blob/{{branch}}/{{filename}}#L{
Link to repository browser, use {{branch}} for branch, {{filename}} and {{line}} as filename and line placeholders.

Exported repository URL:
URL of repository where users can fetch changes from Weblate

Source string bug reporting address:
Email address for reports on errors in source strings. Leave empty for no e-mails.

Repository branch:
Repository branch to translate

Push branch:
Branch for pushing changes, leave empty to use repository branch

Filemask: weblate/langdata/locale/*/LC_MESSAGES/djg
Path of files to translate relative to repository root, use * instead of language code, for example: po/* po or locale/*/LC_MESSAGES/django.po.

Monolingual base language file:
Filename of translation base file, containing all strings and their source; it is recommended for monolingual translation formats.

☑ Edit base file
Whether users will be able to edit the base file for monolingual translations.

Intermediate language file:
Filename of intermediate translation file. In most cases this is a translation file provided by developers and is used when creating actual source strings.

Template for new translations: weblate/langdata/locale/django.pot
Filename of file used for creating new translations. For gettext choose .pot file.

File format: gettext PO file

☐ Locked
Locked component will not get any translation updates.

☑ Allow translation propagation
Whether translation updates in other components will cause automatic translation in this one

☑ Turn on suggestions
Whether to allow translation suggestions at all.

☐ Suggestion voting
Whether users can vote for suggestions.

Autoaccept suggestions: 0
Automatically accept suggestions with this number of votes, use 0 to turn it off.

Translation flags:
Additional comma-separated flags to influence quality checks. Possible values can be found in the documentation.

Enforced checks: []
List of checks which can not be ignored.

Translation license: GNU General Public License v3.0 or later

Contributor agreement:
User agreement which needs to be approved before a user can translate this component.

Adding new translation: Create new language file
How to handle requests for creating new translations.

Language code style: Default based on the file format
Customize language code used to generate the filename for translations created by Weblate.

Merge style: Rebase
Define whether Weblate should merge the upstream repository or rebase changes onto it.

Commit message when translating:
Translated using Weblate ({{ language_name }})

Currently translated at {{ stats.translated_percent }}% ({{ stats.translated }} of {{ stats.all }} strings)

Translation: {{ project_name }}/{{ component_name }}
Translate-URL: {{ url }}

You can use template language for various info, please consult the documentation for more details.

Commit message when adding translation:
Added translation using Weblate ({{ language_name }})
You can use template language for various info, please consult the documentation for more details.

Commit message when removing translation:
Deleted translation using Weblate ({{ language_name }})
You can use template language for various info, please consult the documentation for more details.

Commit message when merging translation:
Merge branch '{{ component_remote_branch }}' into Weblate.
You can use template language for various info, please consult the documentation for more details.

Commit message when addon makes a change:
Update translation files

Updated by "{{ addon_name }}" hook in Weblate.

Translation: {{ project_name }}/{{ component_name }}
Translate-URL: {{ url }}

You can use template language for various info, please consult the documentation for more details.

Committer name: Weblate

Committer e-mail: noreply@weblate.org

☑ Push on commit
Whether the repository should be pushed upstream on every commit.

Age of changes to commit: 24
Time in hours after which any pending changes will be committed to the VCS.

☑ Lock on error
Whether the component should be locked on repository errors.

Source language: English
Language used for source strings in all components

Language filter: ^(cs|hu|ru)$
Regular expression used to filter translation when scanning for filemask.

Variants regular expression:
Regular expression used to determine variants of a string.

Priority: Medium
Components with higher priority are offered first to translators.

☐ Restricted component
Restrict access to the component to only those explicitly given permission.

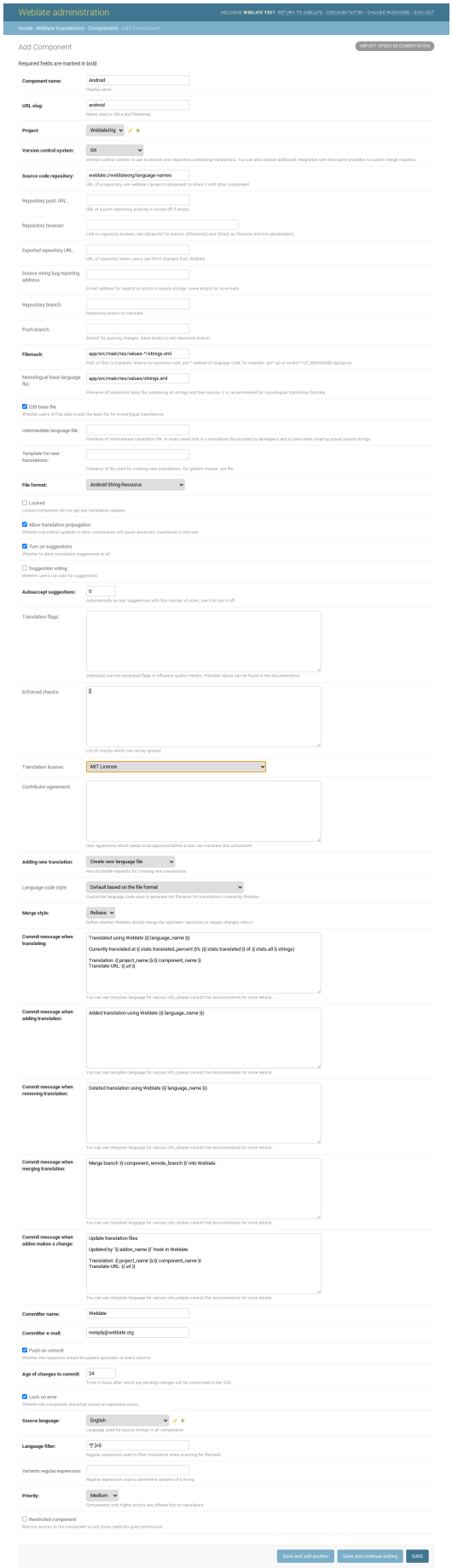
Save and add another | Save and continue editing | SAVE

См.также:

Конфигурация компонента, *Двуязычные и одноязычные форматы*

Одноязычные компоненты

Для облегчения их перевода предоставьте шаблон файла, содержащий отображение идентификаторов сообщений на соответствующие строки на языке оригинала (обычно это английский) (для справки по параметрам смотрите раздел *Конфигурация компонента*):


Weblate administration
WELCOME, WEBLATE TEST. RETURN TO WEBLATE / DOCUMENTATION / CHANGE PASSWORD / SIGN OUT
Home › Weblate translations › Components › Add Component
Add Component
IMPORT SPEED DOCUMENTATION
Required fields are marked in bold.
Component name:
Display name
URL slug:
Name used in URLs and filenames.
Project:
Version control system:
Source code repository:
Repository push URL:
Repository browser:
Exported repository URL:
Source string bug reporting address:
Repository branch:
Push branch:
Filemask:
Monolingual base language file:
Edit base file
Intermediate language file:
Template for new translations:
File format:
Locked
Allow translation propagation
Turn on suggestions
Suggestion voting
Autoaccept suggestions:
Translation flags:
Enforced checks:
Translation license:
MIT License
Contributor agreement:
Adding new translation:
Create new language file
Language code style:
Default based on the file format
Merge style:
Rebase
Commit message when translating:
Commit message when adding translation:
Commit message when removing translation:
Commit message when merging translation:
Commit message when addon makes a change:
Committer name:
Committer e-mail:
Push on commit
Age of changes to commit:
Lock on error
Source language:
English
Language filter:
Variants regular expression:
Priority:
Medium
Restricted component
Save and add another
Save and continue editing
SAVE

См.также:

Конфигурация компонента, *Двуязычные и одноязычные форматы*

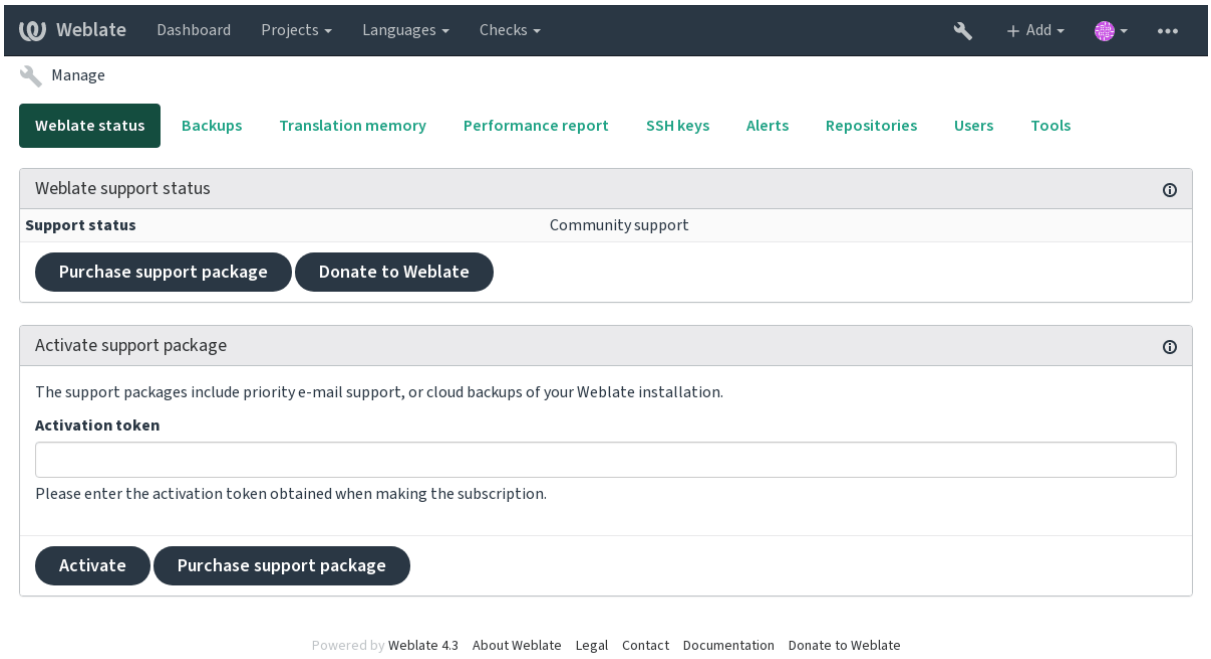
2.24 Getting support for Weblate

Weblate — это свободное программное обеспечение с поддержкой сообщества, выпускаемое на условиях авторского лева. Подписчики получают приоритетную поддержку без дополнительной оплаты. Предоплаченные пакеты помощи доступны для всех. Более подробную информацию о текущих предложениях поддержки вы можете найти по адресу <https://weblate.org/support/>.

2.24.1 Integrating support

Добавлено в версии 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

No other data is submitted.

2.24.3 Integration services

- See if your support package is still valid
- *Предоставляемое Weblate'ом хранилище резервных копий*

Подсказка: Purchased support packages are already activated upon purchase, and can be used without integrating them.

2.25 Правовые документы

Примечание: Herein you will find various legal information you might need to operate Weblate in certain legal jurisdictions. It is provided as a means of guidance, without any warranty of accuracy or correctness. It is ultimately your responsibility to ensure that your use of Weblate complies with all applicable laws and regulations.

2.25.1 ITAR and other export controls

Weblate can be run within your own datacenter or virtual private cloud. As such, it can be used to store ITAR or other export-controlled information, however, end users are responsible for ensuring such compliance.

The Hosted Weblate service has not been audited for compliance with ITAR or other export controls, and does not currently offer the ability to restrict translations access by country.

2.25.2 US encryption controls

Weblate does not contain any cryptographic code, but might be subject export controls as it uses third party components utilizing cryptography for authentication, data-integrity and -confidentiality.

Most likely Weblate would be classified as ECCN 5D002 or 5D992 and, as publicly available libre software, it should not be subject to EAR (see Encryption items NOT Subject to the EAR).

Software components used by Weblate (listing only components related to cryptographic function):

Python Смотрите https://wiki.python.org/moin/PythonSoftwareFoundationLicenseFaq#Is_Python_subject_to_export_laws.3F

GnuPG Optionally used by Weblate

Git Optionally used by Weblate

curl Used by Git

OpenSSL Used by Python and cURL

The strength of encryption keys depends on the configuration of Weblate and the third party components it interacts with, but in any decent setup it will include all export restricted cryptographic functions:

- In excess of 56 bits for a symmetric algorithm
- Factorisation of integers in excess of 512 bits for an asymmetric algorithm
- Computation of discrete logarithms in a multiplicative group of a finite field of size greater than 512 bits for an asymmetric algorithm
- Discrete logarithms in a group different than above in excess of 112 bits for an asymmetric algorithm

Weblate doesn't have any cryptographic activation feature, but it can be configured in a way where no cryptography code would be involved. The cryptographic features include:

- Accessing remote servers using secure protocols (HTTPS)
- Создание PGP подписей для коммитов

См.также:

Export Controls (EAR) on Open Source Software

ГЛАВА 3

Документация для участников

3.1 Внесение вклада в Weblate

Существуют десятки способов внесения вашего вклада в Weblate. Приветствуется любая помощь, будь то написание кода, проектирование внешнего вида, написание документации или спонсорство:

- *Сообщение о проблемах в Weblate'е*
- *Starting contributing code to Weblate*
- *Перевод Weblate*
- *Финансирование разработки Weblate*

3.1.1 Перевод Weblate

Weblate переводится с помощью самого себя, не стесняйтесь принимать участие в работе по обеспечению доступности Weblate на максимально возможном количестве человеческих языков.

3.1.2 Финансирование разработки Weblate

Вы можете профинансировать дальнейшую разработку Weblate на странице приёма пожертвований. Собранные там средства используются для финансирования безвозмездного хостинга для свободных программных проектов, а также для дальнейшей разработки Weblate. Для получения более подробной информации, например, о целях финансирования и о вознаграждении, которое вы можете получить за то, что являетесь спонсором, пожалуйста, посетите *страницу приёма пожертвований*.

Backers who have funded Weblate

List of Weblate supporters:

- Yashiro Ccs
- Cheng-Chia Tseng
- Timon Reinhard
- Cassidy James
- Loic Dachary
- Marozed
- https://freedombox.org/
- GNU Solidario (GNU Health)

Do you want to be in the list? Please see options on the Donate to Weblate.

3.2 Starting contributing code to Weblate

To understand Weblate source code, please first look into *Исходный код Weblate*, *Weblate frontend* and *Weblate internals*.

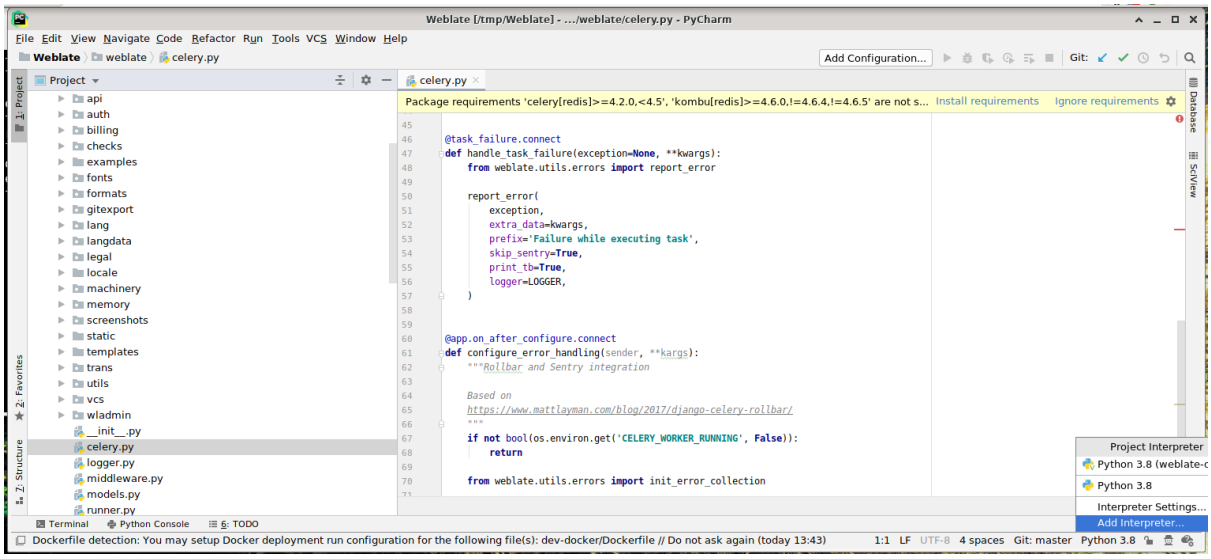
3.2.1 Starting with our codebase

If looking for some bugs to familiarize yourself with the Weblate codebase, look for ones labelled good first issue.

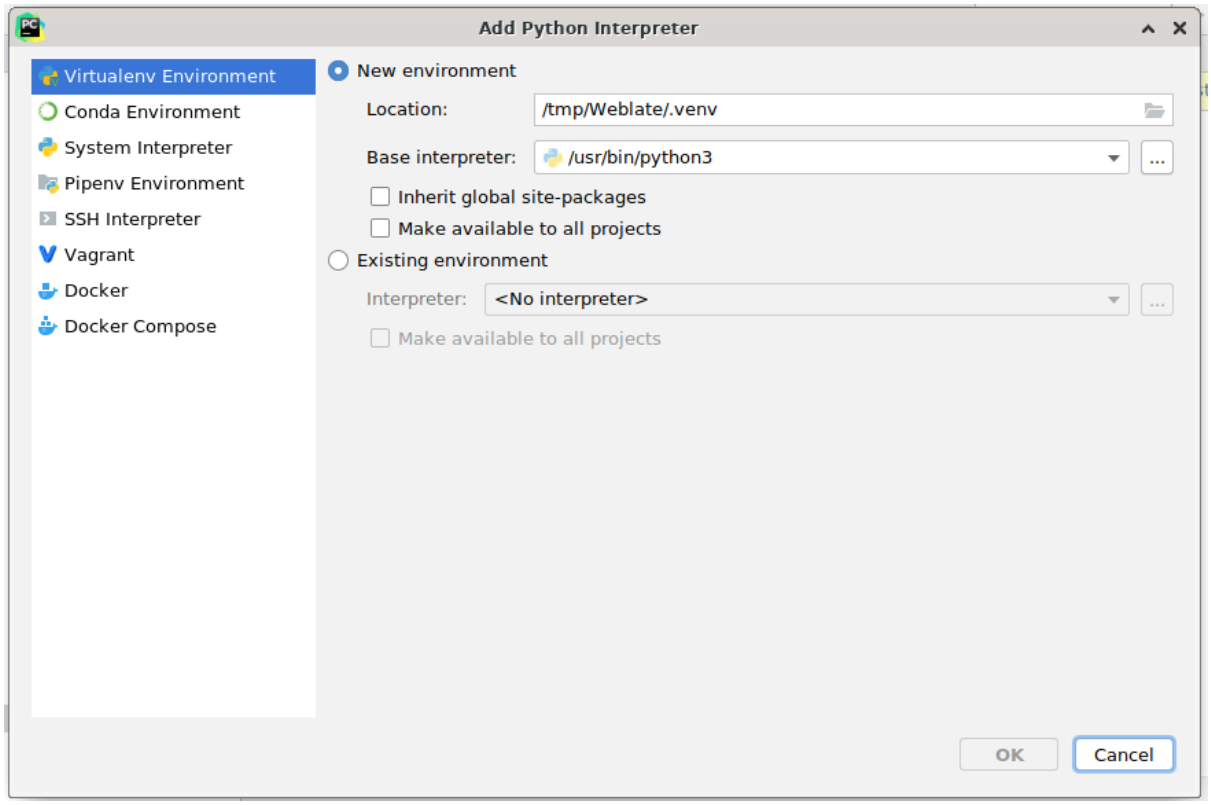
3.2.2 Coding Weblate with PyCharm

PyCharm is a known IDE for Python, here's some guidelines to help you setup Weblate project in it.

Considering you have just cloned the Github repository, just open the folder in which you cloned it in PyCharm. Once the IDE is open, the first step is to specify the interpreter you want:

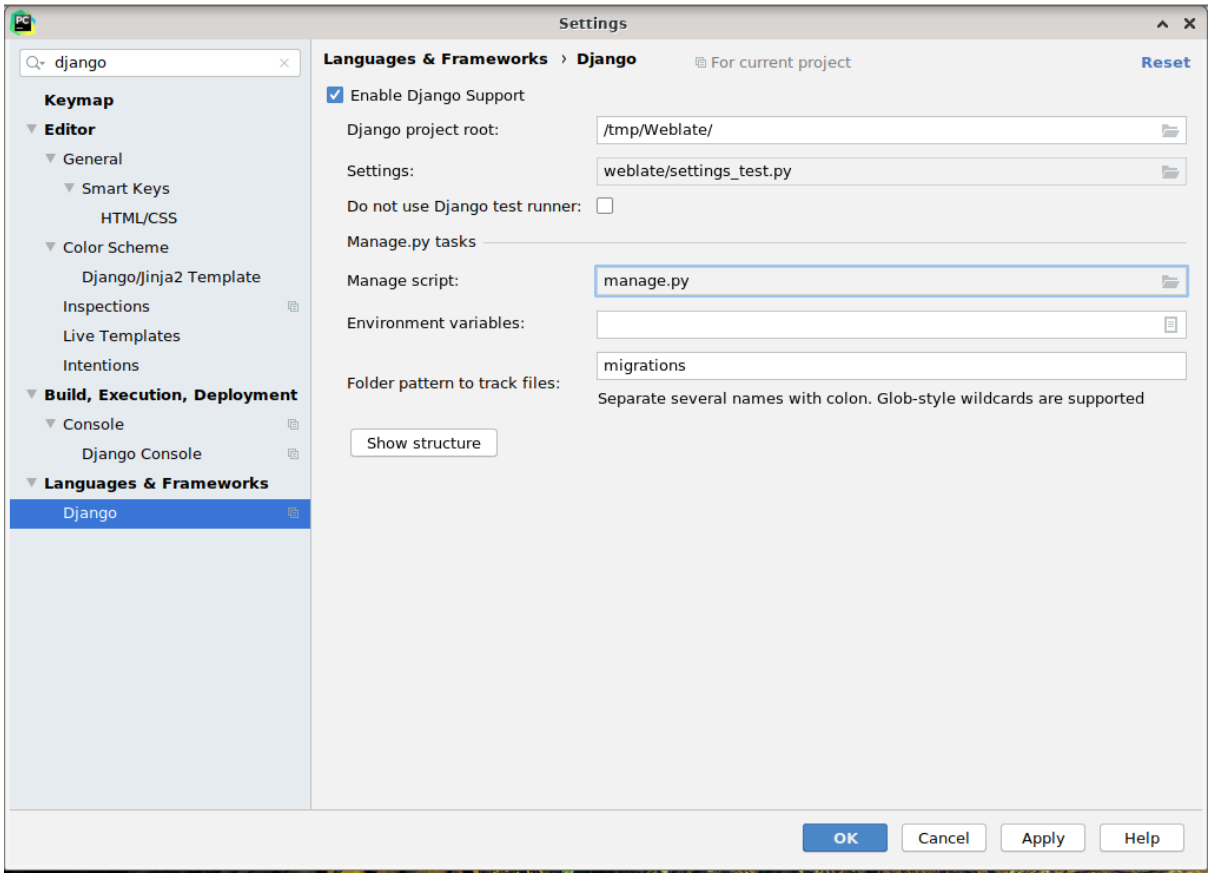


You can either choose to let PyCharm create the virtualenv for you, or select an already existing one:



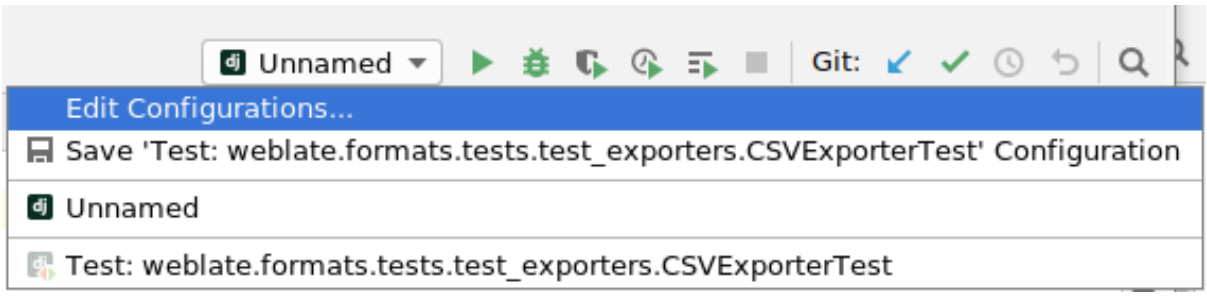
Don't forget to install the dependencies once the interpreter is set: you can do it, either through the console (the console from the IDE will directly use your virtualenv by default), or through the interface when you get a warning about missing dependencies.

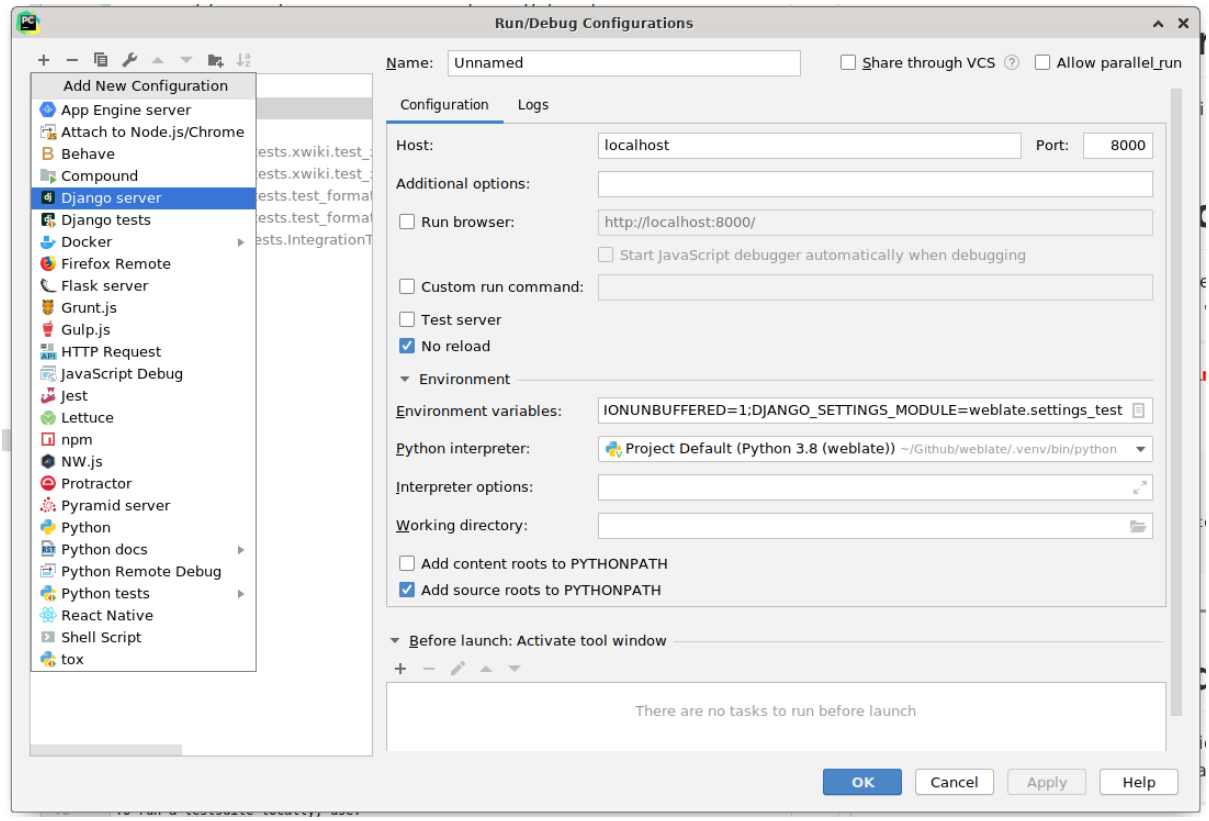
The second step is to set the right information to use natively Django inside PyCharm: the idea is to be able to immediately trigger the unit tests in the IDE. For that you need to specify the root path of Django and the path of one setting:



Be careful, the *Django project root* is the root of the repository, not the weblate sub-directory. About the settings, I personally use the *settings_test* from the repository, but you could create your own setting and set it there.

Last step is to be able to run the server and to put breakpoints on the code to be able to debug it. This is done by creating a new *Django Server* configuration:





Be careful to properly checked «No reload»: you won't get anymore the server live reload if you modify some files, but the debugger will be stopped on the breakpoint you set.

3.2.3 Running Weblate locally

The most comfortable approach to get started with Weblate development is to follow *Установка из исходников*. It will get you a virtual env with editable Weblate sources.

To install all dependencies useful for development, do:

```
pip install -r requirements-dev.txt
```

To start a development server run:

```
weblate runserver
```

Depending on your configuration you might also want to start Celery workers:

```
./weblate/examples/celery start
```

Running Weblate locally in Docker

If you have Docker and docker-compose installed, you can spin up the development environment simply by running:

```
./rundev.sh
```

It will create development Docker image and start it. Weblate is running on <http://127.0.0.1:8080/> and you can sign in with `admin` user and `admin` password. The new installation is empty, so you might want to continue with *Добавление проектов и компонентов перевода*.

The `Dockerfile` and `docker-compose.yml` for this are located in `dev-docker` directory.

The script also accepts some parameters, to execute tests run it with `test` parameter and then specify any `test` parameters, for example:

```
./rundev.sh test --failfast weblate.trans
```

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 stop the background containers run:

```
./rundev.sh stop
```

Running the script without args will recreate Docker container and restart it.

Примечание: Эта установка не подходит для рабочей среды, она включает в себя несколько небезопасных хаков, которые, однако, упрощают разработку.

3.2.4 Bootstraping your devel instance

You might want to use *`import_demo`* to create demo translations and *`createadmin`* to create admin user.

3.3 Исходный код Weblate

Weblate разрабатывается на GitHub'е. Вы вольны форкать код и открывать запросы на извлечение. Заплатки в любой другой форме также приветствуются.

См.также:

Чтобы понять, как Weblate устроен изнутри, посмотрите раздел *Weblate internals*.

3.3.1 Принципы «безопасности по проектированию»

Любой код для Weblate'а должен писаться с оглядкой на документ *Принципы «безопасности по проектированию»*.

3.3.2 Стандарт кодирования

Код должен соответствовать рекомендациям по кодированию PEP-8 и должен быть отформатирован с использованием инструмента форматирования кода **`black`**.

Для проверки качества кода можно использовать **`flake8`**, рекомендуемые плагины перечислены в файле :file:``.pre-commit-config.yaml`, а их конфигурация размещена в файле `setup.cfg`.

Самый простой подход к внедрению всего этого заключается в установке pre-commit. Репозиторий Weblate содержит для него конфигурацию для проверки вменяемости зафиксированных файлов. После его установки (он уже включен в список зависимостей в `requirements-lint.txt`) включите его, запустив `pre-commit install` в каталоге с рабочей копией Weblate. Таким образом все ваши изменения будут проверяться автоматически.

Также вы можете запустить эту проверку вручную, чтобы проверить все файлы, запустите:

```
pre-commit run --all
```

3.4 Отладка Weblate

Bugs can behave as application crashes or as misbehavior. You are welcome to collect info on any such issue and submit it to the issue tracker.

3.4.1 Режим отладки

При включении режима отладки исключения будут отображаться в браузере. Это полезно для отладки проблем в веб-интерфейсе, но для рабочей среды он не подходит, поскольку сильно влияет на производительность и может привести к утечке конфиденциальных данных.

См.также:

Отключение отладочного режима

3.4.2 Журналы Weblate

Weblate can produce detailed logs of what is going in the background. In the default configuration it uses syslog and that makes the log appear either in `/var/log/messages` or `/var/log/syslog` (depending on your syslog daemon configuration).

Docker containers log to their output (as usual in the Docker world), so you can look at the logs using `docker-compose logs`.

См.также:

Пример конфигурации contains `LOGGING` configuration.

3.4.3 Анализ падений программы

In case the application crashes, it is useful to collect as much info about the crash as possible. The easiest way to achieve this is by using third-party services which can collect such info automatically. You can find info on how to set this up in *Сбор отчётов об ошибках*.

3.4.4 Silent failures

Многие задачи выгружаются в Celery для фоновой обработки. Их падения в пользовательском интерфейсе не отображаются, а появляются только в журналах Celery. Настройка *сбора ошибок* поможет вам легче замечать такие сбои.

3.4.5 Performance issues

In case Weblate performs badly in some situation, please collect the relevant logs showing the issue, and anything that might help figuring out where the code might be improved.

In case some requests take too long without any indication, you might want to install *dogslow <https://pypi.org/project/dogslow/>* along with *Сбор отчётов об ошибках* and get pinpointed and detailed tracebacks in the error collection tool.

3.5 Weblate internals

Примечание: 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, 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 *Weblate frontend*.

3.5.2 Модули

Weblate consists of several Django applications (some optional, see *Дополнительные модули Weblate*):

`accounts`

User account, profiles and notifications.

`addons`

Addons to tweak Weblate behavior, see *Надстройки*.

`api`

API based on Django REST framework.

`auth`

Authentication and permissions.

`billing`

Дополнительный модуль *Тарифный план*.

`checks`

Модуль *Проверки качества* строки перевода.

`fonts`

Модуль проверки отрисовки шрифтов.

`formats`

File format abstraction layer based on translate-toolkit.

`gitexport`

Дополнительный модуль *Экспортер Git*.

`lang`

Модуль определения языка и моделей множественного числа.

`legal`

Дополнительный модуль *Право*.

`machinery`

Интеграция сервисов машинного перевода.

`memory`

Встроенная память переводов, смотрите раздел *Память переводов*.

`screenshots`

Модуль управления снимками экрана и распознаванием текста.

`trans`

Основной модуль обработки переводов.

`utils`

Различные вспомогательные утилиты.

`vcs`

Абстракция системы контроля версий.

`wladmin`

Настройка административного интерфейса Django.

3.6 Weblate frontend

The frontend is currently built using Bootstrap, jQuery and few third party libraries.

3.6.1 Dependency management

The yarn package manager is used to update third party libraries. The configuration lives in `scripts/yarn` and there is a wrapper script `scripts/yarn-update` to upgrade the libraries, build them and copy to correct locations in `weblate/static/vendor`, where all third partly frontend code is located.

3.6.2 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.

3.6.3 Локализация

Should you need any user visible text in the frontend code, it should be localizable. In most cases all you need is to wrap your text inside `gettext` function, but there are more complex features available:

```
document.write(gettext('this is to be translated'));

var object_count = 1 // or 0, or 2, or 3, ...
s = ngettext('literal for the singular case',
        'literal for the plural case', object_count);

fmts = ngettext('There is %s object. Remaining: %s',
        'There are %s objects. Remaining: %s', 11);
s = interpolate(fmts, [11, 20]);
// s is 'There are 11 objects. Remaining: 20'
```

См.также:

Translation topic in the Django documentation

3.6.4 Иконки

Weblate currently uses material design icons, in case you are looking for new one, check <https://materialdesignicons.com/>.

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.

3.7 Сообщение о проблемах в Weblate'е

Наша система отслеживания проблем размещена на GitHub'е:

Feel welcome to report any issues with, or suggest improvement of Weblate there. If what you have found is a security issue in Weblate, please consult the «Security issues» section below.

3.7.1 Проблемы безопасности

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.

Alternatively, report to security@weblate.org, which ends up on HackerOne as well.

If you don't want to use HackerOne, for whatever reason, you can send the report by e-mail to michal@cihar.com. You can choose to encrypt it using this PGP key *3CB 1DF1 EF12 CF2A C0EE 5A32 9C27 B313 42B7 511D*. You can also get the PGP key from Keybase.

Примечание: 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

3.8 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.

3.8.1 Continuous integration

Current test results can be found on GitHub Actions and coverage is reported on Codecov.

There are several jobs to verify different aspects:

- Unit tests
- Documentation build and external links
- Migration testing from all supported releases
- Code linting
- Setup verification (ensures that generated dist files do not miss anything and can be tested)

The configuration for the CI is in `.github/workflows` directory. It heavily uses helper scripts stored in `ci` directory. The scripts can be also executed manually, but they require several environment variables, mostly defining Django settings file to use and database connection. The example definition of that is in `scripts/test-database`:

```
# Simple way to configure test database from environment

# Database backend to use postgresql / mysql / mariadb
export CI_DATABASE=${1:-postgresql}

# Database server configuration
export CI_DB_USER=weblate
export CI_DB_PASSWORD=weblate
export CI_DB_HOST=127.0.0.1

# Django settings module to use
export DJANGO_SETTINGS_MODULE=weblate.settings_test
```

The simple execution can look like:

```
. scripts/test-database
./ci/run-migrate
./ci/run-test
./ci/run-docs
./ci/run-setup
```

3.8.2 Local testing

To run a testsuite locally, use:

```
DJANGO_SETTINGS_MODULE=weblate.settings_test ./manage.py test
```

Подсказка: 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 *Настройка базы данных для Weblate* for setup instructions.

The `weblate/settings_test.py` is used in CI environment as well (see *Continuous integration*) and can be tuned using environment variables:

```
# Simple way to configure test database from environment

# Database backend to use postgresql / mysql / mariadb
export CI_DATABASE=${1:-postgresql}

# Database server configuration
export CI_DB_USER=weblate
export CI_DB_PASSWORD=weblate
export CI_DB_HOST=127.0.0.1

# Django settings module to use
export DJANGO_SETTINGS_MODULE=weblate.settings_test
```

Prior to running tests you should collect static files as some tests rely on them being present:

```
DJANGO_SETTINGS_MODULE=weblate.settings_test ./manage.py collectstatic
```

You can also specify individual tests to run:

```
DJANGO_SETTINGS_MODULE=weblate.settings_test ./manage.py test weblate.gitexport
```

Подсказка: The tests can also be executed inside developer docker container, see *Running Weblate locally in Docker*.

См.также:

See Testing in Django for more info on running and writing tests for Django.

3.9 Схемы данных

Для определения структуры внешних JSON-файлов Weblate использует JSON-схему.

3.9.1 Схема памяти переводов Weblate'а

https://weblate.org/schemas/weblate-memory.schema.json			
тип	*массив*		
элементы	*Элемент памяти переводов*		
	тип	*объект*	
	свойства		
	• **category**	*Категория строки*	
		1 — глобальная, 2 — общая, 10000000+ — специфичная для проекта, 20000000+ — специфичная для пользователя	
		тип	*целое число*
		примеры	1
		минимум	0
		значение по умолчанию	1
	• **origin**	*Происхождение строки*	
		Имя файла или название компонента	
		тип	*строка*
		примеры	test
		шаблон	^(.*)$
		значение по умолчанию	
	• **source**	*Исходная строка*	
		тип	*строка*
		примеры	Hello
		шаблон	^(.+)$
		значение по умолчанию	
	• **source_language**	*Исходный язык*	
		ISO 639-1 / ISO 639-2 / IETF BCP 47	
		тип	*строка*
		примеры	en
		шаблон	^([^]+)$
		значение по умолчанию	
	• **target**	*Целевая строка*	
		тип	*строка*
		примеры	Привет
		шаблон	^(.+)$
		значение по умолчанию	
	• **target_language**	*Целевой язык*	
		ISO 639-1 / ISO 639-2 / IETF BCP 47	

continues on next page

Таблица 1 – продолжение с предыдущей страницы

		тип	*строка*
		примеры	ru
		шаблон	^([^]+)$
		значение по умолчанию	
	дополнительные свойства	False	
определения			

См.также:

Память переводов, `dump_memory`, `import_memory`

3.9.2 Файл экспорта данных пользователя Weblate'а

https://weblate.org/schemas/weblate-userdata.schema.json			
тип	*объект*		
свойства			
• **basic**	*Основные данные*		
	тип	*объект*	
	свойства		
	• **username**	*Имя пользователя*	
		тип	*строка*
		примеры	admin
		шаблон	^.*$
		значение по умолчанию	
	• **full_name**	*Полное имя*	
		тип	*строка*
		примеры	Администратор Weblate
		шаблон	^.*$
		значение по умолчанию	
	• **email**	*Адрес электронной почты*	
		тип	*строка*
		примеры	noreply@example.com
		шаблон	^.*$
		значение по умолчанию	
	• **date_joined**	*Дата присоединения*	
		тип	*строка*
		примеры	2019-11-18T18:53:54.862Z
		шаблон	^.*$
		значение по умолчанию	
• **profile**	*Профиль*		
	тип	*объект*	
	свойства		
	• **language**	*Язык*	
		тип	*строка*
		примеры	ru
		шаблон	^.*$
		значение по умолчанию	
	• **suggested**	*Количество предложенных строк*	
		тип	*целое число*

continues on next page

Таблица 2 – продолжение с предыдущей страницы

		примеры	1
		значение по умолчанию	0
	• **translated**	*Количество переведённых строк*	
		тип	*целое число*
		примеры	24
		значение по умолчанию	0
	• **uploaded**	*Количество загруженных снимков экрана*	
		тип	*целое число*
		примеры	1
		значение по умолчанию	0
	• **hide_completed**	*Скрывать завершённые переводы с панели управления*	
		тип	*логический*
		примеры	False
		значение по умолчанию	True
	• **secondary_in_zen**	*Show secondary translations in the Zen mode*	
		тип	*логический*
		примеры	True
		значение по умолчанию	True
	• **hide_source_secondary**	*Hide source if a secondary translation exists*	
		тип	*логический*
		примеры	False
		значение по умолчанию	True
	• **editor_link**	*Editor link*	
		тип	*строка*
		примеры	
		шаблон	^.*$
		значение по умолчанию	
	• **translate_mode**	*Translation editor mode*	
		тип	*целое число*
		примеры	0
		значение по умолчанию	0
	• **zen_mode**	*Zen editor mode*	
		тип	*целое число*
		примеры	0
		значение по умолчанию	0
	• **special_chars**	*Special characters*	
		тип	*строка*
		примеры	
		шаблон	^.*$
		значение по умолчанию	
	• **dashboard_view**	*Default dashboard view*	
		тип	*целое число*
		примеры	1
		значение по умолчанию	0
	• **dashboard_component_list**	*Default component list*	

continues on next page

Таблица 2 – продолжение с предыдущей страницы

		значение по умолчанию	None	
		anyOf	тип	*null*
			тип	*целое число*
	• **languages**	*Translated languages*		
		тип	*массив*	
		значение по умолчанию		
		элементы	*Language code*	
			тип	*строка*
			примеры	ru
			шаблон	^.*$
			значение по умолчанию	
	• **secondary_languages**	*Secondary languages*		
		тип	*массив*	
		значение по умолчанию		
		элементы	*Language code*	
			тип	*строка*
			примеры	sk
			шаблон	^.*$
			значение по умолчанию	
	• **watched**	*Watched projects*		
		тип	*массив*	
		значение по умолчанию		
		элементы	*Project slug*	
			тип	*строка*
			примеры	weblate
			шаблон	^.*$
			значение по умолчанию	
• **auditlog**	*Audit log*			
	тип	*массив*		
	значение по умолчанию			
	элементы	*Items*		
		тип	*объект*	
		свойства		
		• **address**	*IP address*	
			тип	*строка*
			примеры	127.0.0.1
			шаблон	^.*$
			значение по умолчанию	
		• **user_agent**	*User agent*	
			тип	*строка*
			примеры	PC / Linux / Firefox 70.0
			шаблон	^.*$
			значение по умолчанию	
		• **timestamp**	*Timestamp*	
			тип	*строка*

continues on next page

Таблица 2 – продолжение с предыдущей страницы

			примеры	2019-11-18T18:58:30.845Z
			шаблон	^.*$
			значение по умолчанию	
		• **activity**	*Activity*	
			тип	*строка*
			примеры	login
			шаблон	^.*$
			значение по умолчанию	
определения				

См.также:

Профиль пользователя, `dumpuserdata`

3.10 Releasing Weblate

Things to check prior to release:

1. Check newly translated languages by **`./scripts/list-translated-languages`**.
2. Set final version by **`./scripts/prepare-release`**.
3. Make sure screenshots are up to date **`make -C docs update-screenshots`**

Perform the release:

4. Create a release **`./scripts/create-release --tag`** (see below for requirements)

Post release manual steps:

5. Update Docker image.
6. Close GitHub milestone.
7. Once the Docker image is tested, add a tag and push it.
8. Update Helm chart to new version.
9. Include new version in `.github/workflows/migrations.yml` to cover it in migration testing.
10. Increase version in the repository by **`./scripts/set-version`**.

To create tags using the **`./scripts/create-release`** script you will need following:

- GnuPG with private key used to sign the release
- Push access to Weblate git repositories (it pushes tags)
- Configured **`hub`** tool and access to create releases on the Weblate repo
- SSH access to Weblate download server (the Website downloads are copied there)

3.11 О Weblate

3.11.1 Цели проекта

Web-based continuous localization tool with tight *Интеграция с системой контроля версий* supporting a wide range of *Поддерживаемые форматы файлов*, making it easy for translators to contribute.

3.11.2 Название проекта

«Weblate» is a portmanteau of the words «web» and «translate».

3.11.3 Веб-сайт проекта

The landing page is <https://weblate.org/> and a cloud hosted service at <https://hosted.weblate.org/>. This documentation can be found on <https://docs.weblate.org/>.

3.11.4 Логотипы проекта

The project logos and other graphics is available in <https://github.com/WeblateOrg/graphics/> repository.

3.11.5 Руководство

This project is maintained by Michal Čihař <michal@cihar.com>.

3.11.6 Авторы

Weblate was started by Michal Čihař <michal@cihar.com>. Since its inception in 2012, thousands of people have contributed.

3.12 Лицензия

Copyright (C) 2012 - 2020 Michal Čihař <michal@cihar.com>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <https://www.gnu.org/licenses/>.

ГЛАВА 4

История изменений

4.1 Weblate 4.3

Released on October 15th 2020.

- В API включена статистика пользователей.
- Исправлена сортировка компонентов на страницах с постраничной разбивкой результатов.
- Определение исходного языка для глоссария.
- Переписана поддержка GitHub'овских и GitLab'овских запросов на извлечение.
- Исправлен счётчик статистики после удаления предложения.
- Расширен общедоступный профиль пользователя.
- Исправлена конфигурация принудительных проверок.
- Улучшена документация по встроенному резервному копированию.
- Атрибут исходного языка перемещён из проекта в компонент.
- Добавлена проверка форматирования локализации Vue.
- Обычная проверка заполнителей теперь поддерживает регулярные выражения.
- Улучшен внешний вид матричного режима.
- «Автоматизированный перевод» теперь называется «автоматическими предложениями».
- Добавлена поддержка взаимодействия с несколькими экземплярами GitLab'а или GitHub'а.
- API расширен для покрытия обновления проектов, обновления и удаления единиц перевода и глоссариев.
- API единицы перевода теперь правильно обрабатывает строки с формами множественного числа.
- Component creation can now handle ZIP file or document upload.
- Единообразие в кодах состояний ответа API.
- Поддержка разметки markdown в соглашении участника.
- Улучшено отслеживание исходных строк.
- Улучшена совместимость с форматами JSON, YAML и CSV.

- Добавлена поддержка удаления строк.
- Improved performance of file downloads.
- Improved repository management view.
- Automatically enable java-format for Android.
- Added support for localized screenshots.
- Added support for Python 3.9.
- Fixed translating HTML files under certain conditions.

4.2 Weblate 4.2.2

Выпущен 2 сентября 2020 года.

- Исправлено сопоставление исходных строк для форматов на основе JSON.
- Исправлено перенаправление входа для некоторых конфигураций аутентификации.
- Исправлена аутентификация через LDAP с синхронизацией групп.
- Исправлено падение при сообщении о прогрессе автоматического перевода.
- Исправлена склейка коммитов Git'а при включённом добавлении завершителей.
- Исправлено создание компонентов из локальной системы контроля версий с помощью API.

4.3 Weblate 4.2.1

Выпущен 21 августа 2020 года.

- Исправлено сохранение форм множественного числа для некоторых локалей в ресурсах Android'а.
- Исправлено падение надстройки очистки на некоторых файлах XLIFF.
- Разрешена настройка CDN локализации в образе Docker'а.

4.4 Weblate 4.2

Выпущен 18 августа 2020 года.

- Улучшены страницы пользователей и добавлен список пользователей.
- Прекращена поддержка перехода с выпусков серии 3.x, перехода через версии 4.1 или 4.0.
- Добавлены экспорты в несколько одноязычных форматов.
- Улучшены графики деятельности.
- Количество отображаемых соседних строк теперь поддаётся настройке.
- Добавлена поддержка блокировки компонентов, в которых возникли ошибки в репозитории.
- Упрощённая основная навигация (замена кнопок на иконки).
- Улучшена обработка кода языка в интеграции с Google Translate.
- Дополнение Git для склеивания коммитов теперь генерирует завершители `Co-authored-by:`.
- Улучшен анализатор поисковых запросов.
- Улучшена обратная связь с пользователями при проверке строк формата.

- Улучшена производительность при массовых изменениях состояния.
- Добавлены перенаправления для совместимости после переименования проекта или компонента.
- Добавлены уведомления об одобрении строк, блокировке компонентов и изменении лицензии.
- Добавлена поддержка ModernMT.
- Разрешено избегать перезаписи одобренных переводов при загрузке файлов.
- Прекращена поддержка некоторых перенаправлений URL-адресов, существовавших ради совместимости.
- Добавлена проверка для шаблонных литералов ECMAScript.
- Добавлен параметр для отслеживания компонента.
- Из ключей единиц перевода JSON удалена ведущая точка.
- Удалена отдельная очередь Celery для памяти переводов.
- Разрешено одновременно переводить все компоненты одного языка.
- Разрешено настраивать HTTP -заголовки `Content-Security-Policy`.
- Добавлена поддержка задания псевдонимов для языков на уровне проекта.
- Новая надстройка для помощи в локализации HTML или JavaScript, смотрите раздел *CDN локализации JavaScript'a*.
- Теперь домен Weblate настраивается в настройках, смотрите описание параметра `SITE_DOMAIN`.
- Добавлена поддержка поиска по компонентам и проектам.

4.5 Weblate 4.1.1

Выпущен 19 июня 2020 года.

- Исправлено изменение конфигурации автоматического исправления или надстроек в Docker.
- Исправлено возможное падение на странице «О Weblate».
- Улучшена установка скомпилированных файлов локали.
- Исправлено добавление слов в глоссарий.
- Исправлено сочетание клавиш для автоматизированного перевода.
- Удалён отладочный вывод, вызывающий при некоторых настройках отбрасывание событий журнала.
- Исправлена индикация блокировки в списке проектов.
- Исправлено перечисление ключей GPG при некоторых настройках.
- Добавлен параметр для выбора используемой версии API DeepL.
- Добавлена поддержка работы в качестве провайдера SAML-сервиса, смотрите *Аутентификация через SAML*.

4.6 Weblate 4.1

Выпущен 15 июня 2020 года.

- Добавлена поддержка создания новых переводов с включённым кодом страны.
- Добавлена поддержка поиска исходных строк со снимками экрана.
- Расширен состав информации, доступный в статистике.
- Улучшено редактирование поиска на страницах перевода.
- Улучшена обработка одновременных обновлений репозитория.
- В форму создания проекта включён исходный язык.
- В список переводчиков включено количество изменений.
- Исправлена возникающая в некоторых случаях ошибка выбора языка пользовательского интерфейса.
- Разрешение внесения методов регистрации в белый список при закрытых регистрациях.
- Улучшен поиск смежных терминов в глоссарии.
- Улучшены соответствия с памятью переводов.
- Группировка одинаковых результатов автоматизированного перевода.
- Добавление прямой ссылки для редактирования снимка экрана прямо со страницы перевода.
- Улучшен диалог подтверждения удаления.
- В скачиваемый ZIP-файл включены шаблоны.
- В объявлениях добавлена поддержка Markdown и настройка уведомлений.
- Расширенная информация в списках проверок.
- Добавлена поддержка новых форматов файлов: *Строки Laravel PHP*, *Файлы HTML*, *Формат OpenDocument*, *Формат IDML*, *Файлы ресурсов Windows*, *Переводы в INI-файлах*, *Переводы в INI-файлах Inno Setup*, *Файлы свойств GWT*, *JSON-файлы go-i18n* и *Файл ARB*.
- Последовательное использование статуса «пропущена» в качестве статуса пропущенных проверок.
- Добавлена поддержка для настройки включённых надстроек по умолчанию.
- Исправлено сочетание клавиш редактора для пропуска проверок.
- Улучшен машинный перевод строк с заполнителями.
- Показ призрачного перевода для языков пользователя, чтобы упростить ему начало перевода.
- Улучшен разбор кода языка.
- Показ переводов на язык пользователя на первом месте в списке.
- Переименование «форм» на более обобщённые «варианты».
- Добавлены новые проверки качества: *Несколько переменных без названий*, *Давно не переводилась* и *Повторяющиеся слова*.
- Вновь введена поддержка стирания памяти переводов.
- Исправлен параметр для включения в исходной строке игнорирования проверок.
- Добавлена поддержка настройки различных веток, в которые следует отправлять изменения.
- API теперь сообщает об ограничении в частоте запросов в заголовках HTTP.
- Добавлена поддержка Google Translate V3 API (расширенная).
- Добавлена возможность ограничения доступа на уровне компонентов.

- Во флагах перевода добавлена поддержка пробельных и других специальных символов, смотрите раздел *Настройка поведения*.
- Отображаемый текст, если он включён, показывается всегда.
- API теперь поддерживает фильтрацию изменений.
- Добавлена поддержка обмена глоссариями между проектами.

4.7 Weblate 4.0.4

Выпущен 7 мая 2020 года.

- Исправлено выполнение тестового набора в некоторых окружениях с Python'ом 3.8.
- Исправление опечаток в документации.
- Исправлена возникавшая в некоторых случаях ошибка создания компонентов с помощью API.
- Исправлены ошибки JavaScript, ломающие навигацию на мобильных устройствах.
- Исправлено падение при отображении некоторых проверок.
- Исправлен вывод списка снимков экрана.
- Исправлены ежемесячные выдержки с уведомлениями.
- Исправлено поведение промежуточного перевода с единицами перевода, не существовавшими в переводе.

4.8 Weblate 4.0.3

Выпущен 2 мая 2020 года.

- Исправлено возможное падение в отчётах.
- Упоминания пользователей в комментариях теперь не чувствительны к регистру.
- Исправлена миграция PostgreSQL для не суперпользователей.
- Исправлено изменение URL-адреса репозитория при создании компонента.
- Исправлено падение при исчезновении вышестоящего репозитория.

4.9 Weblate 4.0.2

Выпущен 27 апреля 2020 года.

- Улучшена производительность статистики перевода.
- Улучшена производительность смены меток.
- Улучшена производительность массовой правки.
- Улучшена производительность памяти переводов.
- Исправлено возможное падение при удалении компонента.
- Исправлено отображение изменений перевода в некоторых пограничных случаях.
- Улучшено предупреждение о слишком длинной очереди Celery.
- Исправлены возможные ложные срабатывания при проверке согласованности.
- Исправлена взаимоблокировка при изменении репозитория связанного компонента.

- В список изменений, CSV и отчёты включено расстояние редактирования.
- Обход ложных срабатываний проверки интервала между знаками препинания для канадского французского языка.
- Исправлен экспорт в формат XLIFF с заполнителями.
- Исправлено ложное срабатывание с проверкой на символ нулевой ширины.
- Улучшены отчёты об ошибках конфигурации.
- Исправлена загрузка двуязычных исходных текстов.
- Автоматическое определение поддерживаемых языков для машинного перевода DeepL.
- Исправлено отображение индикатора выполнения в некоторых пограничных случаях.
- Исправлено срабатывание некоторых проверок на непереведённых строках.

4.10 Weblate 4.0.1

Выпущен 16 апреля 2020 года.

- Исправлена установка пакета из PyPI.

4.11 Weblate 4.0

Выпущен 16 апреля 2020 года.

- Weblate теперь требует Python 3.6 или новее.
- Добавлен обзор управления предупреждений компонентов.
- Добавлено предупреждение компонента о сломанных URL-адресах браузера репозитория.
- Улучшены страницы входа и регистрации.
- Управление доступом к проекту и конфигурация рабочего процесса интегрированы в настройки проекта.
- Добавлена проверка и подсветка для интерполяции и вложенных строк i18next.
- Добавлена проверка и подсветка для процентных заполнителей.
- Отображение неудачных проверок в предложениях.
- Запись изменений исходной строки в историю.
- Обновление переводчика Microsoft Translator на третью версию его API.
- Повторная реализация серверной части памяти переводов.
- В *Поиск* добавлена поддержка нескольких `is:` поисков.
- Разрешено делать так, чтобы проверка *Перевод без изменения* не попадала во внутренний чёрный список.
- Улучшено извлечение комментариев из одноязычных po файлов.
- «Доски сообщений» переименованы в «объявления».
- Исправлены случайные проблемы с регистрационными письмами.
- Улучшена надстройка обновления LINGUAS-файла: теперь она обрабатывает большее количество синтаксических вариантов.
- Исправлено редактирование одноязычного исходного файла XLIFF.
- В *Поиск* добавлена поддержка точного соответствия.

- Расширенный API, охватывающий снимки экрана, пользователей, группы, списки компонентов и расширенное создание проектов.
- Добавлена поддержка загрузки исходных текстов в двуязычные переводы.
- Добавлена поддержка промежуточного языка от разработчиков.
- Добавлена поддержка рецензирования исходных строк.
- Расширенные параметры скачивания памяти переводов со всей платформы.

4.12 Серия Weblate 3.x

4.12.1 Weblate 3.11.3

Выпущен 11 марта 2020 года.

- Исправлен поиск полей с определённым приоритетом.
- Исправлен предопределённый запрос для недавно добавленных строк.
- Исправлен поиск, возвращающий повторяющиеся совпадения.
- Исправлена отрисовка уведомлений в Gmail.
- Исправлен возврат изменений из истории.
- Добавлены ссылки на события в выдержках с уведомлениями.
- Исправлено электронное письмо для подтверждения удаления учётной записи.
- Добавлена поддержка аутентификации через Slack в контейнере Docker.
- Устранение отправки уведомлений для языков, на которые пользователь не подписан.
- Включение в обзор производительности очередей Celery.
- Исправлены ссылки на документацию для надстроек.
- Уменьшение количества ложно-отрицательных результатов для проверки «Перевод без изменения».
- Повышение версии зависимости bleach для закрытия уязвимости CVE-2020-6802.
- Исправлено перечисление изменений в истории на уровне проекта.
- Исправлено аннулирование статистики в некоторых граничных случаях.
- Исправлен поиск определённых состояний строки.
- Улучшено поведение проверки строк формата при пропущенном проценте.
- Исправлена аутентификация с использованием некоторых сторонних провайдеров.

4.12.2 Weblate 3.11.2

Выпущен 22 февраля 2020 года.

- Исправлена отрисовка предложений.
- Исправлены некоторые строки, для которых ошибочно сообщалось, что в нет слов.

4.12.3 Weblate 3.11.1

Выпущен 20 февраля 2020 года.

- Документированы изменения в настройке Celery.
- Улучшена проверка имени файла при создании компонентов.
- Исправлены минимальные версии некоторых зависимостей.
- Исправлено добавление групп в определённых версиях Django.
- Исправлена ручная отправка изменений в вышестоящий репозиторий.
- Улучшен поиск соответствия в словаре.

4.12.4 Weblate 3.11

Выпущен 17 февраля 2020 года.

- Разрешено использование URL-адреса для отправки в систему контроля версий во время создания компонента через API.
- Проверка ширины отрисованного текста теперь показывает изображение с отрисованным текстом.
- Исправлены ссылки в электронных письмах с уведомлениями.
- Улучшен внешний вид простых текстовых электронных писем.
- Отображение проигнорированных проверок и разрешение сделать их снова активными.
- Отображение соседних ключей в одноязычных переводах.
- Добавлена поддержка группировки строковых форм.
- Рекомендация обновления до новой версии Weblate в проверках системы.
- Предоставление более подробного анализа для предупреждения о повторяющемся языке.
- Включение более подробной информации о лицензиях на страницах проекта.
- Автоматическое преобразование поверхностных локальных копий в полные, если это необходимо.
- Исправлено скачивание строк, требующих действий.
- Новое предупреждение о повторном использовании одной и той же маски файлов.
- Улучшено извлечение мест подстановки в XML.
- Теперь параметр `SINGLE_PROJECT` может включать принудительное перенаправление на выбранный проект.
- Добавлена возможность указания, что описанная в комментарии проблема решена.
- Добавлена массовая правка флагов.
- Добавлена поддержка *меток*.
- Добавлена надстройка «Массовая правка».
- Добавлен параметр для *Принудительные проверки*.
- Увеличен умолчательный срок действия ссылок подтверждения.
- Улучшена интеграция с Matomo.
- Исправлена проверка *Уже переведено* для корректной обработки изменений в исходных строках.
- Расширение настроек автоматического обновления, контролируемых параметром `AUTO_UPDATE`.
- Надстройки LINGUAS теперь выполняют полную синхронизацию переводов с Weblate.

4.12.5 Weblate 3.10.3

Выпущен 18 января 2020 года.

- Поддержка translate-toolkit 2.5.0.

4.12.6 Weblate 3.10.2

Выпущен 18 января 2020 года.

- К проектам добавлена индикация блокировки.
- Исправлена ошибка в CSS, вызывавшая мерцание в некоторых веб-браузерах.
- Исправлен поиск в системах с неанглийскими локалями.
- Улучшено сопоставление репозитория для обработчиков GitHub и Bitbucket.
- Исправлена миграция данных на некоторых установках Python 2.7.
- Разрешена настройка поверхностного клонирования в Git'е.
- Улучшена обработка фоновых уведомлений.
- Исправлена ошибка отправки сломанных форм при использовании браузерной кнопки «назад».
- Новая надстройка для настройки форматирования YAML.
- Исправлена проверка «Одинаковые формы множественного числа»: теперь она не срабатывает в языках с единственной формой множественного числа.
- Исправлен поиск с использованием регулярных выражений по некоторым полям.

4.12.7 Weblate 3.10.1

Выпущен 9 января 2020 года.

- Расширенный API с созданием перевода.
- Исправлено несколько граничных случаев при миграции данных.
- Совместимость с Django 3.0.
- Улучшена производительность очистки данных.
- Добавлена поддержка настройки security.txt.
- Улучшены хлебные крошки в журнале изменений.
- Улучшен список переводов в панели управления.
- Улучшены HTTP-ответы для веб-обработчиков.
- Добавлена поддержка запросов на слияние GitLab в контейнере Docker.

4.12.8 Weblate 3.10

Выпущен 20 декабря 2019 года.

- Улучшен пользовательский интерфейс приложения.
- Добавлена проверка на двойной пробел.
- Исправлено создание новых языков.
- Избежание отправки уведомлений из журнала аудита на удалённые адреса электронной почты.
- Добавлена поддержка строк только для чтения.

- Добавлена поддержка Markdown'а в комментариях.
- Разрешение размещения текста инструкции по переводу в информации о проекте.
- Добавлено копирование в буфер обмена для дополнительных языков.
- Улучшена поддержка Mercurial.
- Улучшена производительность извлечения из Git-репозитория.
- Добавлен поиск по возрасту строки.
- Показ исходного языка для всех переводов.
- Показ контекста для соседних строк.
- Добавлена поддержка уведомлений об операциях с репозиторием.
- Улучшены списки перевода.
- Расширены возможности поиска.
- Добавлена поддержка автоматического перевода строк, помеченных на правку.
- Избежание отправки повторных уведомлений для предупреждений в связанных компонентах.
- Улучшение умолчательного сообщения о запросе на слияние.
- Улучшение индикации состояния строки в Дзен-режиме.
- Добавлена поддержка большего количества языков в Яндекс.Переводчике.
- Улучшен внешний вид электронных писем уведомлений.
- Предоставлена возможность выбора лицензии на перевод.

4.12.9 Weblate 3.9.1

Выпущен 28 октября 2019 года.

- Из резервных копий удалены кое-какие ненужные файлы.
- Исправлено возможное падение в отчётах.
- Исправлена падение при миграции из одной базы данных в другую.
- Добавлена поддержка принудительной отправки изменений в Git-репозитории.
- Снижение риска аннулирования токена регистрации.
- Исправлено удаление учётной записи, упёршейся в ограничение на частоту запросов.
- Добавлен поиск по приоритету.
- Исправлено возможное падение при добавлении строк в JSON-файл.
- Проверка и исправление небезопасного HTML теперь учитывает разметку исходной строки.
- Избегание отправки уведомлений приглашённым, но в то же время удалённым пользователям.
- Исправлено SSL-соединение с redis в Celery, находящимся в контейнере Docker'а.

4.12.10 Weblate 3.9

Выпущен 15 октября 2019 года.

- Включение в скачиваемые файлы метаданных Weblate.
- Улучшен пользовательский интерфейс для неудачных проверок.
- Указание недостающих строк в проверках формата.
- Отдельная проверка на пробелы между знаками препинания французского языка.
- Добавлена поддержка исправления некоторых ошибок проверки качества.
- Добавлено отдельное разрешение на создание новых проектов.
- Расширена статистика по количеству символов.
- Улучшение поддержки языковых кодов в стиле Java.
- Добавлена новая обобщённая проверка на заполнители.
- Добавлена поддержка заполнителей WebExtension JSON.
- Добавлена поддержка формата плоского XML-файла.
- API расширен методами удаления и создания проекта, компонента и перевода.
- Добавлена поддержка веб-обработчиков Gitea и Gitee.
- Добавлена новая пользовательская проверка на основе регулярных выражений.
- Разрешение настройки внесения вклада в общую память переводов.
- Скачивание ZIP-архива добавлено для большего количества файлов перевода.
- Разбор атрибутов maxwidth и font в формате XLIFF сделан совместимым со стандартом.
- Добавлена новая проверка и исправление для безопасной HTML-разметки при переводе веб-приложений.
- В компоненте добавлено предупреждение при обнаружении неподдерживаемой конфигурации.
- Добавлена надстройка автоматического перевода для начального перевода.
- Автоматический перевод расширен для добавления предложений.
- Отображение в обзоре параметров надстройки.
- Sentry теперь поддерживается через современный Sentry SDK вместо Raven.
- Изменены примеры настроек, чтобы они лучше соответствовали окружению рабочей среды.
- Добавлено автоматическое резервное копирование с помощью BorgBackup.
- Разбиение надстройки очистки для RESX на две надстройки очистки и обновления, чтобы избежать нежелательных обновлений файлов.
- Добавлены расширенные возможности поиска.
- Пользователям разрешено скачивать свои собственные отчёты.
- Для помощи в настройке компонентов добавлено руководство по локализации.
- Добавлена поддержка запросов на слияние GitLab.
- Улучшено отображение статуса репозитория.
- Выполнение автоматического перевода в фоновом режиме.

4.12.11 Weblate 3.8

Выпущен 15 августа 2019 года.

- Добавлена поддержка упрощённого создания похожих компонентов.
- Добавлена поддержка разбора флагов перевода из форматов файлов, основанных на XML.
- Журналирование исключений в журнал Celery.
- Улучшена производительность надстроек, работающих с репозиторием.
- Улучшен внешний вид электронных писем уведомлений.
- Исправлено поведение при сбросе пароля.
- Улучшена производительность на большинстве страниц перевода.
- Исправлен список языков, неизвестных Weblate'у.
- Добавлена поддержка клонирования надстроек в обнаруженные компоненты.
- Добавлена поддержка замены содержимого файла загруженным файлом.
- Добавлена поддержка перевода содержимого не из системы контроля версий.
- Добавлен виджет изображения OpenGraph для его использования в социальных сетях.
- Добавлена поддержка анимированных снимков экрана.
- Улучшена обработка одноязычных файлов XLIFF.
- Избегание отправки нескольких уведомлений на одно событие.
- Добавлена поддержка фильтра по изменениям.
- Расширенные предопределённые периоды отчётности.
- Добавлена поддержка веб-обработчиков для репозиториев Azure Repos.
- Новые необязательные уведомления об ожидающих предложениях или непереведённых строках.
- В письма с уведомлениями добавлена ссылка для отписки от них в одно нажатие.
- Исправлены ложные срабатывания в проверке «Уже переведено».
- Новый интерфейс управления для администраторов.
- Приоритет строки теперь можно задать с помощью флагов.
- Добавлены представления управления языком.
- Добавлены проверки на формат строк для библиотеки Qt и Ruby.
- Добавлена конфигурация, лучше подходящая для установок с единственным проектом.
- Уведомление о появлении новой строки при изменении исходной в одноязычном переводе.
- Добавлено отдельное представление для памяти переводов с возможностью поиска.

4.12.12 Weblate 3.7.1

Выпущен 28 июня 2019 года.

- Обновление документации.
- Исправлены некоторые ограничения для версий зависимостей.
- Обновлена база данных языков.
- Обновления локализации.
- Различные мелкие исправления пользовательского интерфейса.

- Улучшена обработка неподдерживаемых, но обнаруженных файлов перевода.
- Более подробный отчёт о не удовлетворённых требованиях к формату файлов.

4.12.13 Weblate 3.7

Выпущен 21 июня 2019 года.

- Добавлена отдельная очередь Celery для уведомлений.
- Использование согласованного с приложением вида для просмотра API.
- В отчёты включена статистика по утверждённым переводам.
- Сообщение о прогрессе при обновлении компонента перевода.
- Разрешено прерывать запущенное фоновое обновление компонентов.
- Расширение языка шаблонов для манипуляций именами файлов.
- Использование шаблонов для ссылки на редактор и адреса браузера репозитория.
- Показ максимальной длины и текущего количества символов при редактировании перевода.
- В проверке «Перевод без изменения» улучшена обработка аббревиатур.
- Обновлена страница входа новых участников.
- Добавлена поддержка настройки надстройки msgmerge.
- При отправке уведомлений добавлена задержка открытия SMTP-соединения.
- Улучшено журналирование ошибок.
- В надстройке генерирования файлов MO разрешено использование пользовательских расположений.
- Добавлены надстройки для очистки старых предложений или комментариев.
- Добавлен параметр для включения горизонтального режима в Дзен-редакторе.
- Повышена производительность импорта множества взаимосвязанных компонентов.
- Исправлены примеры установки, бывшие в некоторых случаях неправильными.
- Улучшено отображение предупреждений в изменениях.
- Добавлен новый виджет горизонтальной статистики.
- Улучшена проверка строк формата для форм множественного числа.
- Добавлен инструмент управления шрифтами.
- Новая проверка размеров отрисовываемого текста.
- Добавлена поддержка форматов субтитров.
- Включена общая статистика по языкам.
- Добавлены отчёты на уровне проекта и на глобальном уровне.
- Улучшен пользовательский интерфейс при отображении статуса перевода.
- Новая цветовая схема и логотип Weblate'a.
- Новый вид растровых значков.

4.12.14 Weblate 3.6.1

Выпущен 26 апреля 2019 года.

- Улучшена обработка одноязычных файлов XLIFF.
- Исправлены выдержки с уведомлениями в некоторых граничных случаях.
- Исправлено предупреждение об ошибке скрипта надстройки.
- Исправлено создание файлов MO для одноязычных файлов PO.
- Исправлено отображение не установленных проверок.
- Обозначение управляемых проектов в списке проектов.
- Разрешено обновление для восстановления из отсутствующего репозитория системы контроля версий.

4.12.15 Weblate 3.6

Выпущен 20 апреля 2019 года.

- Добавлена поддержка скачивания данных пользователя.
- Теперь надстройки автоматически запускаются при установке.
- Улучшены инструкции по разрешению конфликтов слияния.
- Надстройка очистки теперь совместима с переводами метаданных App store.
- Настраиваемый синтаксис кода языка при добавлении новых переводов.
- Предупреждение об использовании Python'а 2 в связи с запланированным прекращением его поддержки в апреле 2020 года.
- Извлечение специальных символов из исходной строки для их показа и быстрой вставке при переводе.
- Расширенная статистика по участникам, отражающая как количество источников, так и количество переводов.
- Администраторы и надстройка проверки согласованности теперь могут добавлять переводы, даже если они отключены для пользователей.
- Исправлено описание переключателя отключения манипуляции заголовком `Language-Team`.
- Уведомление пользователей, упомянутых в комментариях.
- Из настройки компонента удалено автоопределение формата файла.
- Исправлено создание файлов MO для одноязычных файлов PO.
- Добавлены выдержки с уведомлениями.
- Добавлена поддержка заглушения уведомлений компонента.
- Добавлены уведомления о новых предупреждениях, сообщениях на досках сообщений или компонентах.
- Теперь можно настроить уведомления для управляемых проектов.
- Улучшена обработка трёхбуквенных кодов языков.

4.12.16 Weblate 3.5.1

Выпущен 10 марта 2019 года.

- 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 addon 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.
- Терминологическая служба Майкрософт сделана совместимой с текущими выпусками Zeep.
- Обновления локализации.

4.12.17 Weblate 3.5

Выпущен 3 марта 2019 года.

- Improved performance of built-in translation memory.
- Added interface to manage global translation memory.
- Improved alerting on bad component state.
- Added user interface to manage whiteboard messages.
- Addon commit message now can be configured.
- Reduce number of commits when updating upstream repository.
- Fixed possible metadata loss when moving component between projects.
- Improved navigation in the Zen mode.
- Added several new quality checks (Markdown related and URL).
- Added support for app store metadata files.
- Added support for toggling GitHub or Gerrit integration.
- Добавлена проверка на буквы кашиды.
- Добавлен параметр для склейки коммитов на основе их авторства.
- Улучшена поддержка формата файла XLSX.
- Совместимость с Tesseract 4.0.
- Billing addon now removes projects for unpaid billings after 45 days.

4.12.18 Weblate 3.4

Выпущен 22 января 2019 года.

- Added support for XLIFF placeholders.
- Celery can now utilize multiple task queues.
- Added support for renaming and moving projects and components.
- В отчёты включено количество символов.
- Added guided adding of translation components with automatic detection of translation files.
- Customizable merge commit messages for Git.
- Added visual indication of component alerts in navigation.
- Улучшена производительность загрузки файлов перевода.
- Новая надстройка для склеивания коммитов до их отправки.
- Улучшено отображение изменений перевода.
- Changed default merge style to rebase and made that configurable.
- Better handle private use subtags in language code.
- Improved performance of fulltext index updates.
- Extended file upload API to support more parameters.

4.12.19 Weblate 3.3

Выпущен 30 ноября 2018 года.

- Добавлена поддержка удаления компонентов и проектов.
- Improved performance for some monolingual translations.
- Added translation component alerts to highlight problems with a translation.
- Expose XLIFF string resname as context when available.
- Added support for XLIFF states.
- Added check for non writable files in DATA_DIR.
- Improved CSV export for changes.

4.12.20 Weblate 3.2.2

Выпущен 20 октября 2018 года.

- Remove no longer needed Babel dependency.
- Updated language definitions.
- Improve documentation for addons, 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.

4.12.21 Weblate 3.2.1

Выпущен 10 октября 2018 года.

- 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.

4.12.22 Weblate 3.2

Выпущен 6 октября 2018 года.

- Добавлена команда управления install_addon для автоматизации установки надстроек.
- Allow more fine grained ratelimit settings.
- Added support for export and import of Excel files.
- Improve component cleanup in case of multiple component discovery addons.
- 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.

4.12.23 Weblate 3.1.1

Выпущен 27 июля 2018 года.

- Fix testsuite failure on some setups.

4.12.24 Weblate 3.1

Выпущен 27 июля 2018 года.

- Upgrades from older version than 3.0.1 are not supported.
- Allow to override default commit messages from settings.
- Improve webhooks compatibility with self hosted environments.
- Added support for Amazon Translate.
- Compatibility with Django 2.1.
- Django system checks are now used to diagnose problems with installation.
- Removed support for soon shutdown libravatar service.
- New addon to mark unchanged translations as needing edit.
- Add support for jumping to specific location while translating.
- Downloaded translations can now be customized.
- Improved calculation of string similarity in translation memory matches.
- Added support by signing Git commits by GnuPG.

4.12.25 Weblate 3.0.1

Выпущен 10 июня 2018 года.

- Fixed possible migration issue from 2.20.
- Обновления локализации.
- Removed obsolete hook examples.
- Improved caching documentation.
- Fixed displaying of admin documentation.
- Improved handling of long language names.

4.12.26 Weblate 3.0

Выпущен 1 июня 2018 года.

- Rewritten access control.
- Several code cleanups that lead to moved and renamed modules.
- New addon for automatic component discovery.
- Команда управления import_project теперь принимает несколько иные параметры.
- Добавлена базовая поддержка файлов ресурсов Windows.
- New addon to store contributor names in PO file headers.
- The per component hook scripts are removed, use addons instead.
- Add support for collecting contributor agreements.

- Access control changes are now tracked in history.
- New addon 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.

4.13 Серия Weblate 2.x

4.13.1 Weblate 2.20

Выпущен 4 апреля 2018 года.

- 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 addon to configure gettext output wrapping.
- New addon to configure JSON formatting.
- Added support for authentication in API using RFC 6750 compatible Bearer authentication.
- Added support for automatic translation using machine translation services.
- Added support for HTML markup in whiteboard messages.
- Added support for mass changing state of strings.
- Translate-toolkit at least 2.3.0 is now required, older versions are no longer supported.
- Added built in translation memory.
- Added componentlists overview to dashboard and per component list overview pages.
- Added support for DeepL machine translation service.
- Machine translation results are now cached inside Weblate.
- Добавлена поддержка переупорядочивания зафиксированных изменений.

4.13.2 Weblate 2.19.1

Выпущен 20 февраля 2018 года.

- Fixed migration issue on upgrade from 2.18.
- Improved file upload API validation.

4.13.3 Weblate 2.19

Выпущен 15 февраля 2018 года.

- Fixed imports across some file formats.
- Display human friendly browser information in audit log.
- Added TMX exporter for files.
- Various performance improvements for loading translation files.
- Added option to disable access management in Weblate in favor of Django one.

- Improved glossary lookup speed for large strings.
- Compatibility with django_auth_ldap 1.3.0.
- Configuration errors are now stored and reported persistently.
- Honor ignore flags in whitespace autofixer.
- Improved compatibility with some Subversion setups.
- Improved built in machine translation service.
- Added support for SAP Translation Hub service.
- Добавлена поддержка терминологической службы Майкрософт.
- 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 addons to customize translation workflows.

4.13.4 Weblate 2.18

Выпущен 15 декабря 2017 года.

- 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.

4.13.5 Weblate 2.17.1

Выпущен 13 октября 2017 года.

- Fixed running testsuite in some specific situations.
- Обновления локалей.

4.13.6 Weblate 2.17

Выпущен 13 октября 2017 года.

- 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.
- Расширены возможности поиска.
- More fine grained per project ACL configuration.
- Default value of BASE_DIR has been changed.
- Added two step account removal to prevent accidental removal.
- Project access control settings is now editable.
- Added optional spam protection for suggestions using Akismet.

4.13.7 Weblate 2.16

Выпущен 11 августа 2017 года.

- 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.

4.13.8 Weblate 2.15

Выпущен 30 июня 2017 года.

- 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.

4.13.9 Weblate 2.14.1

Выпущен 24 мая 2017 года.

- 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.

4.13.10 Weblate 2.14

Выпущен 17 мая 2017 года.

- 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.

4.13.11 Weblate 2.13.1

Выпущен 12 апреля 2017 года.

- 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.

4.13.12 Weblate 2.13

Выпущен 12 апреля 2017 года.

- Fixed quality checks on translation templates.
- Added quality check to trigger on losing translation.
- Add option to view pending suggestions from user.
- Add option to automatically build component lists.
- Default dashboard for unauthenticated users can be configured.
- Add option to browse 25 random strings for review.
- History now indicates string change.
- Better error reporting when adding new translation.
- Added per language search within project.
- Group ACLs can now be limited to certain permissions.
- The per project ALCs are now implemented using Group ACL.
- Added more fine grained privileges control.
- Various minor UI improvements.

4.13.13 Weblate 2.12

Выпущен 3 марта 2017 года.

- 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.
- Улучшено отображение противоречивых переводов.
- 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.

4.13.14 Weblate 2.11

Выпущен 31 января 2017 года.

- 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 addon.
- 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.

4.13.15 Weblate 2.10.1

Выпущен 20 января 2017 года.

- Do not leak account existence on password reset form (CVE-2017-5537).

4.13.16 Weblate 2.10

Выпущен 15 декабря 2016 года.

- Added quality check to check whether plurals are translated differently.
- Fixed GitHub hooks for repositories with authentication.
- Added optional Git exporter module.
- Поддержка API переводчика Microsoft Cognitive Services.
- 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.

4.13.17 Weblate 2.9

Выпущен 4 ноября 2016 года.

- Расширены параметры для команды управления createadmin.
- 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.

4.13.18 Weblate 2.8

Выпущен 31 августа 2016 года.

- Улучшения в документации.
- Переводы.
- Updated bundled javascript libraries.
- Добавлена команда управления list_translators.
- 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.

4.13.19 Weblate 2.7

Выпущен 10 июля 2016 года.

- Removed Google web translate machine translation.
- Improved commit message when adding translation.
- Fixed Google Translate API for Hebrew language.
- Compatibility with Mercurial 3.8.
- Добавлена команда управления import_json.
- 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.

4.13.20 Weblate 2.6

Выпущен 28 апреля 2016 года.

- Исправлена проверка компонентов с языковым фильтром.
- Улучшена поддержка файлов XLIFF.
- Исправлен машинный перевод для неанглоязычных исходных текстов.
- Добавлен REST API.
- Совместимость с Django 1.10.
- К сообщениям совместной доски добавлены категории.

4.13.21 Weblate 2.5

Выпущен 10 марта 2016 года.

- Исправлен автоматический перевод для владельцев проектов.
- Улучшенная производительность выполнения коммитов и отправки в репозиторий.
- Новая команда управления для добавления предложений из командной строки.
- Добавлена поддержка объединения комментариев при загрузке файлов.
- Добавлена поддержка некоторых расширений GNU к формату функции printf C.
- Улучшения в документации.
- Добавлена поддержка генерации списка переводчиков.
- Добавлена поддержка генерации статистики участников.
- Поиск по всему сайту теперь можно вести только по одному языку.
- Улучшение проверок качества для армянского языка.
- 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.
- Исправлен поиск неудачных проверок по всему сайту.

- Added option to specify source language.
- Улучшена поддержка файлов XLIFF.
- Extended list of options for import_project.
- Improved targeting for whiteboard messages.
- Support for automatic translation across projects.
- Optimized fulltext search index.
- Добавлена команда управления для автоматического перевода.
- 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 review (formerly fuzzy).
- Clarified terminology on strings needing action and not translated strings.
- Поддержка 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.
- Улучшения в документации.
- Added support for configurable automatic group assignments.
- Improved adding of new translations.

4.13.22 Weblate 2.4

Выпущен 20 сентября 2015 года.

- 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.

- Добавлена команда управления для оптимизации полнотекстового индекса.
- 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.
- Добавлена команда управления для изменения имени сайта.
- Add option to configure default committer.
- Add hook after adding new translation.
- Add option to specify multiple files to add to commit.

4.13.23 Weblate 2.3

Выпущен 22 мая 2015 года.

- 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.

4.13.24 Weblate 2.2

Выпущен 19 февраля 2015 года.

- Улучшения производительности.
- 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.

4.13.25 Weblate 2.1

Выпущен 5 декабря 2014 года.

- Added support for Mercurial repositories.
- Replaced Glyphicon font by Awesome.
- Added icons for social authentication services.
- Better consistency of button colors and icons.
- Улучшения в документации.
- Исправления различных ошибок.
- 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.

4.13.26 Weblate 2.0

Выпущен 6 ноября 2014 года.

- New responsive UI using Bootstrap.
- Rewritten VCS backend.
- Улучшения в документации.
- 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.

4.14 Серия Weblate 1.x

4.14.1 Weblate 1.9

Выпущен 6 мая 2014 года.

- Django 1.6 compatibility.
- No longer maintained compatibility with Django 1.4.
- Команды управления для блокировки/разблокировки переводов.
- Improved support for Qt TS files.
- Users can now delete their account.
- Avatars can be disabled.
- Merged first and last name attributes.
- Avatars are now fetched and cached server side.
- Added support for shields.io badge.

4.14.2 Weblate 1.8

Выпущен 7 ноября 2013 года.

- 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.
- Улучшения в документации.
- Improved source strings review.
- Searching across all strings.

- Better tracking of source strings.
- Captcha protection for registration.

4.14.3 Weblate 1.7

Выпущен 7 октября 2013 года.

- 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 translating view.

4.14.4 Weblate 1.6

Выпущен 25 июля 2013 года.

- 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.

4.14.5 Weblate 1.5

Выпущен 16 апреля 2013 года.

- 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.

4.14.6 Weblate 1.4

Выпущен 23 января 2013 года.

- Fixed deleting of checks/comments on string deletion.
- Добавлен параметр для отключения автоматического распространения переводов.
- Added option to subscribe for merge failures.
- Correctly import on projects which needs custom ttkit loader.
- Added sitemaps to allow easier access by crawlers.
- Provide direct links to string in notification e-mails or feeds.
- Various improvements to admin interface.
- В интерфейс администратора добавлены подсказки по настройке Weblate для рабочей среды.
- 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 string tests coverage.

4.14.7 Weblate 1.3

Выпущен 16 ноября 2012 года.

- Compatibility with PostgreSQL database backend.
- Removes languages removed in upstream git repository.
- Improved quality checks processing.
- Added new checks (BB code, XML markup and newlines).
- Support for optional rebasing instead of merge.
- Possibility to relocate Weblate (for example to run it under /weblate path).
- Support for manually choosing file type in case autodetection fails.
- 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.

4.14.8 Weblate 1.2

Выпущен 14 августа 2012 года.

- 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.
- Caching of counted strings with failing checks.
- Checking of requirements during setup.
- Улучшения в документации.

4.14.9 Weblate 1.1

Выпущен 4 июля 2012 года.

- 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.

4.14.10 Weblate 1.0

Выпущен 10 мая 2012 года.

- 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.

4.15 Серия Weblate 0.x

4.15.1 Weblate 0.9

Выпущен 18 апреля 2012 года.

- Исправлен импорт неизвестных языков.
- Улучшен список соседних сообщений.
- Улучшено несколько проверок.
- Обновление документации.

- Добавлены определения для ещё нескольких языков.
- Всякая разная чистка кода.
- Улучшения в документации.
- Изменение макета файла.
- Обновление вспомогательных скриптов до версии Django 1.4.
- Улучшение навигации во время перевода.
- Улучшение обработки переименования po-файлов.
- Better validation while creating component.
- Полная интеграция установки в команду syncdb.
- На всех страницах перевода добавлен список последних изменений.
- Проверка не переведённых строк теперь игнорирует сообщения, содержащие только строку формата.

4.15.2 Weblate 0.8

Выпущен 3 апреля 2012 года.

- Собственный полнотекстовый поиск заменён на Whoosh.
- Различные исправления и улучшения проверок.
- Новая команда updatechecks.
- Множество обновлений переводов.
- Добавлен словарь для хранения наиболее часто используемых терминов.
- Добавлена конечная точка /admin/report/ для обзора состояния репозиториев.
- Сервисы машинного перевода больше не блокируют загрузку страницы.
- Интерфейс управления теперь также содержит полезные действия для обновления данных.
- Запись в журнал сделанных пользователями изменений.
- Возможность откладывания коммита в Git, чтобы генерировать поменьше коммитов от одного пользователя.
- Возможность просмотра неудачных проверок.
- Автоматический перевод с использованием уже переведённых строк.
- Новая страница «О Weblate» теперь показывает использованные версии программного обеспечения.
- Совместимость с Django 1.4.
- Возможность отправки изменений в удалённый репозиторий из веб-интерфейса.
- Добавлено рецензирование переводов, выполненных другими участниками.

4.15.3 Weblate 0.7

Выпущен 16 февраля 2012 года.

- Прямая поддержка GitHub'овских уведомлений.
- Добавлена поддержка очистки осиротевших проверок и переводов.
- Отображение соседних строк во время перевода.
- Отображение похожих строк во время перевода.
- Улучшен поиск строк.

4.15.4 Weblate 0.6

Выпущен 14 февраля 2012 года.

- Добавлены различные проверки для переведённых сообщений.
- Настраиваемый контроль доступа.
- Улучшена обработка переводов с новыми строками.
- Добавлена сортировка таблиц на стороне клиента.
- Если вы обновляетесь, пожалуйста, ознакомьтесь с инструкциями по обновлению.

4.15.5 Weblate 0.5

Выпущен 12 февраля 2012 года.

- **Поддержка машинного перевода с использованием следующих онлайн-сервисов:**
  - Apertium
  - Microsoft Translator
  - MyMemory
- Несколько новых переводов.
- Улучшено слияние изменений из вышестоящего репозитория.
- Улучшенная обработка одновременных git pull и перевода.
- Распространение теперь работает и для неточных изменений.
- Распространение теперь работает и для загрузок файлов.
- Исправлено скачивание файлов при использовании FastCGI (и, возможно, других протоколов).

4.15.6 Weblate 0.4

Выпущен 8 февраля 2012 года.

- В документацию добавлено руководство по использованию.
- Исправлены обработчики API, которые не требовали CSRF-защиты.

4.15.7 Weblate 0.3

Выпущен 8 февраля 2012 года.

- Улучшенное отображение исходной строки для переводов форм множественного числа.
- Новая документация в формате Sphinx.
- Отображение при переводе дополнительных языков.
- Улучшенная страница ошибок со списком существующих проектов.
- Новая статистика по языкам.

4.15.8 Weblate 0.2

Выпущен 7 февраля 2012 года.

- Улучшена проверка некоторых форм.
- Предупреждение пользователей при обновлении профиля.
- Запоминание URL-адреса для входа в систему.
- Именование областей текста при вводе форм множественного числа.
- Автоматическое расширение области перевода.

4.15.9 Weblate 0.1

Выпущен 6 февраля 2012 года.

- Начальный выпуск.

Содержание модулей Python

W

wlc, 151
wlc.config, 151
wlc.main, 152

HTTP Routing Table

/

ANY /, 101

/api

GET /api/, 103

/api/changes

GET /api/changes/, 134
GET /api/changes/(int:id)/, 135

/api/component-lists

GET /api/component-lists/, 137
GET /api/component-lists/(str:slug)/, 137
POST /api/component-lists/(str:slug)/components/, 138
PUT /api/component-lists/(str:slug)/, 138
DELETE /api/component-lists/(str:slug)/, 138
DELETE /api/component-lists/(str:slug)/components/(str:component_slug), 138
PATCH /api/component-lists/(str:slug)/, 138

/api/components

GET /api/components/, 118
GET /api/components/(string:project)/(string:component)/, 118
GET /api/components/(string:project)/(string:component)/changes/, 122
GET /api/components/(string:project)/(string:component)/lock/, 123
GET /api/components/(string:project)/(string:component)/monolingual/base/, 125
GET /api/components/(string:project)/(string:component)/new_template/, 125
GET /api/components/(string:project)/(string:component)/repository/, 123
GET /api/components/(string:project)/(string:component)/screenshots/, 122
GET /api/components/(string:project)/(string:compo
126
GET /api/components/(string:project)/(string:compo
125
POST /api/components/(string:project)/(string:comp
123
POST /api/components/(string:project)/(string:comp
124
POST /api/components/(string:project)/(string:comp
125
PUT /api/components/(string:project)/(string:compo
122
DELETE /api/components/(string:project)/(string:co
122
PATCH /api/components/(string:project)/(string:com
120

/api/glossary

GET /api/glossary/, 139
GET /api/glossary/(int:id)/, 139
GET /api/glossary/(int:id)/projects/, 140
GET /api/glossary/(int:id)/terms/, 141
GET /api/glossary/(int:id)/terms/(int:term_id)/, 141
POST /api/glossary/(int:id)/projects/, 141
POST /api/glossary/(int:id)/terms/, 141
PUT /api/glossary/(int:id)/, 140
PUT /api/glossary/(int:id)/terms/(int:term_id)/, 141
DELETE /api/glossary/(int:id)/, 140
DELETE /api/glossary/(int:id)/projects/, 141
DELETE /api/glossary/(int:id)/terms/(int:term_id)/, 142
PATCH /api/glossary/(int:id)/, 140
PATCH /api/glossary/(int:id)/terms/(int:term_id)/, 142

/api/groups

GET /api/groups/, 107
GET /api/groups/(int:id)/, 107
POST /api/groups/, 107

POST /api/groups/(int:id)/componentlists/, 109
POST /api/groups/(int:id)/components/, 109
POST /api/groups/(int:id)/languages/, 109
POST /api/groups/(int:id)/projects/, 109
POST /api/groups/(int:id)/roles/, 108
PUT /api/groups/(int:id)/, 108
DELETE /api/groups/(int:id)/, 108
DELETE /api/groups/(int:id)/componentlists/(int:component_list_id), 110
DELETE /api/groups/(int:id)/components/(int:component_id), 109
DELETE /api/groups/(int:id)/languages/(string:language_code), 109
DELETE /api/groups/(int:id)/projects/(int:project_id), 109
PATCH /api/groups/(int:id)/, 108

/api/languages

GET /api/languages/, 111
GET /api/languages/(string:language)/, 111
GET /api/languages/(string:language)/statistics/, 112
POST /api/languages/, 111
PUT /api/languages/(string:language)/, 112
DELETE /api/languages/(string:language)/, 112
PATCH /api/languages/(string:language)/, 112

/api/projects

GET /api/projects/, 113
GET /api/projects/(string:project)/, 113
GET /api/projects/(string:project)/changes/, 114
GET /api/projects/(string:project)/components/, 115
GET /api/projects/(string:project)/languages/, 117
GET /api/projects/(string:project)/repository/, 114
GET /api/projects/(string:project)/statistics/, 118
POST /api/projects/, 113
POST /api/projects/(string:project)/components/, 115
POST /api/projects/(string:project)/repository/, 115
PUT /api/projects/(string:project)/, 114
DELETE /api/projects/(string:project)/, 114
PATCH /api/projects/(string:project)/, 114

/api/roles

GET /api/roles/, 110
GET /api/roles/(int:id)/, 110
POST /api/roles/, 110
PUT /api/roles/(int:id)/, 110
DELETE /api/roles/(int:id)/, 111
PATCH /api/roles/(int:id)/, 110

/api/screenshots

GET /api/screenshots/, 135
GET /api/screenshots/(int:id)/, 135
GET /api/screenshots/(int:id)/file/, 135
POST /api/screenshots/, 136
POST /api/screenshots/(int:id)/file/, 135
POST /api/screenshots/(int:id)/units/, 136
PUT /api/screenshots/(int:id)/, 137
DELETE /api/screenshots/(int:id)/, 137
DELETE /api/screenshots/(int:id)/units/(int:unit_i
136
PATCH /api/screenshots/(int:id)/, 137

/api/translations

GET /api/translations/, 127
GET /api/translations/(string:project)/(string:con
127
GET /api/translations/(string:project)/(string:con
129
GET /api/translations/(string:project)/(string:con
130
GET /api/translations/(string:project)/(string:con
131
GET /api/translations/(string:project)/(string:con
132
GET /api/translations/(string:project)/(string:con
129
POST /api/translations/(string:project)/(string:co
130
POST /api/translations/(string:project)/(string:co
131
POST /api/translations/(string:project)/(string:co
131
POST /api/translations/(string:project)/(string:co
130
DELETE /api/translations/(string:project)/(string
129

/api/units

GET /api/units/, 132
GET /api/units/(int:id)/, 132
PUT /api/units/(int:id)/, 134
DELETE /api/units/(int:id)/, 134
PATCH /api/units/(int:id)/, 133

/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/, 106
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)/notifications/(int:subscription_id)/, 107
PATCH /api/users/(str:username)/, 105
PATCH /api/users/(str:username)/notifications/(int:subscription_id)/, 106

/exports

GET /exports/rss/, 146
GET /exports/rss/(string:project)/, 146
GET /exports/rss/(string:project)/(string:component)/, 146
GET /exports/rss/(string:project)/(string:component)/(string:language)/, 146
GET /exports/rss/language/(string:language)/, 146
GET /exports/stats/(string:project)/(string:component)/, 144

/hooks

GET /hooks/update/(string:project)/, 142
GET /hooks/update/(string:project)/(string:component)/, 142
POST /hooks/azure/, 143
POST /hooks/bitbucket/, 143
POST /hooks/gitea/, 144
POST /hooks/gitee/, 144
POST /hooks/github/, 142
POST /hooks/gitlab/, 143
POST /hooks/pagure/, 143

Алфавитный указатель

Символы

.XML resource file
file format, 88
--add
auto_translate command line option, 343
--addon ADDON
install_addon command line option, 349
--age HOURS
commit_pending command line option, 344
--author USER@EXAMPLE.COM
add_suggestions command line option, 343
--base-file-template TEMPLATE
import_project command line option, 347
--check
importusers command line option, 349
--config PATH
wlc command line option, 147
--config-section SECTION
wlc command line option, 147
--configuration CONFIG
install_addon command line option, 349
--convert
wlc command line option, 149
--email USER@EXAMPLE.COM
createadmin command line option, 345
--file-format FORMAT
import_project command line option, 348
--force
loadpo command line option, 350
--force-commit
pushgit command line option, 351
--format {csv,json,text,html}
wlc command line option, 147
--ignore
import_json command line option, 346
--inconsistent
auto_translate command line option, 343
--input
wlc command line option, 149
--key KEY
wlc command line option, 147
--lang LANGUAGE
loadpo command line option, 350
--language-code
list_translators command line option, 350
--language-map LANGMAP
import_memory command line option, 347
--language-regex REGEX
import_project command line option, 348
--license NAME
import_project command line option, 348
--license-url URL
import_project command line option, 348
--main-component
import_project command line option, 348
--main-component COMPONENT
import_json command line option, 346
--mt MT
auto_translate command line option, 344
--name
createadmin command line option, 345
--name-template TEMPLATE
import_project command line option, 347
--new-base-template TEMPLATE
import_project command line option, 347
--no-password
createadmin command line option, 345
--no-privs-update
setupgroups command line option, 352
--no-projects-update

setupgroups command line option, 352
--no-update
setuplang command line option, 352
--output
wlc command line option, 149
--overwrite
auto_translate command line option, 343
wlc command line option, 149
--password PASSWORD
createadmin command line option, 345
--project PROJECT
import_json command line option, 346
--source PROJECT/COMPONENT
auto_translate command line option, 343
--threshold THRESHOLD
auto_translate command line option, 344
--update
createadmin command line option, 345
import_json command line option, 346
install_addon command line option, 349
--url URL
wlc command line option, 147
--user USERNAME
auto_translate command line option, 343
--username USERNAME
createadmin command line option, 345
--vcs NAME
import_project command line option, 348

A

add_suggestions
weblate admin command, 343
add_suggestions command line option
--author USER@EXAMPLE.COM, 343
ADMINS
setting, 191
AKISMET_API_KEY
setting, 300
ALLOWED_HOSTS
setting, 191
Android
file format, 83
ANONYMOUS_USER_NAME
setting, 300
API, 101, 146, 150
Apple strings
file format, 84
ARB
file format, 87
AUDITLOG_EXPIRY
setting, 301
AUTH_LOCK_ATTEMPTS
setting, 301
AUTH_TOKEN_VALID
setting, 302
auto_translate
weblate admin command, 343
auto_translate command line option
--add, 343
--inconsistent, 343
--mt MT, 344
--overwrite, 343
--source PROJECT/COMPONENT, 343
--threshold THRESHOLD, 344
--user USERNAME, 343
AUTO_UPDATE
setting, 301
AUTOFIX_LIST
setting, 302
AVATAR_URL_PREFIX
setting, 302

B

BASE_DIR
setting, 303

C

celery_queues
weblate admin command, 344
changes
wlc command line option, 148
CHECK_LIST
setting, 303
checkgit
weblate admin command, 344
cleanup
wlc command line option, 148
cleanuptrans
weblate admin command, 345
Comma separated values
file format, 88
Command (*класс в wlc.main*), 152
COMMENT_CLEANUP_DAYS
setting, 304
commit
wlc command line option, 148
commit_pending
weblate admin command, 344
commit_pending command line option
--age HOURS, 344
COMMIT_PENDING_HOURS
setting, 304
commitgit
weblate admin command, 344
createadmin
weblate admin command, 345
createadmin command line option
--email USER@EXAMPLE.COM, 345
--name, 345
--no-password, 345

--password PASSWORD, 345
--update, 345
--username USERNAME, 345
CSP_CONNECT_SRC
setting, 303
CSP_FONT_SRC
setting, 303
CSP_IMG_SRC
setting, 303
CSP_SCRIPT_SRC
setting, 303
CSP_STYLE_SRC
setting, 303
CSV
file format, 88

D

DATA_DIR
setting, 304
DATABASE_BACKUP
setting, 305
DATABASES
setting, 192
DEBUG
setting, 192
DEFAULT_ACCESS_CONTROL
setting, 305
DEFAULT_ADD_MESSAGE
setting, 305
DEFAULT_ADDON_MESSAGE
setting, 305
DEFAULT_ADDONS
setting, 306
DEFAULT_COMMIT_MESSAGE
setting, 305
DEFAULT_COMMITER_EMAIL
setting, 306
DEFAULT_COMMITER_NAME
setting, 306
DEFAULT_DELETE_MESSAGE
setting, 305
DEFAULT_FROM_EMAIL
setting, 192
DEFAULT_MERGE_MESSAGE
setting, 305
DEFAULT_MERGE_STYLE
setting, 307
DEFAULT_PULL_MESSAGE
setting, 307
DEFAULT_RESTRICTED_COMPONENT
setting, 305
DEFAULT_TRANSLATION_PROPAGATION
setting, 307
download
wlc command line option, 149
DTD
file format, 90
dump_memory
weblate admin command, 345
dumpuserdata
weblate admin command, 346

E

ENABLE_AVATARS
setting, 307
ENABLE_HOOKS
setting, 307
ENABLE_HTTPS
setting, 307
ENABLE_SHARING
setting, 308

F

file format
.XML resource file, 88
Android, 83
Apple strings, 84
ARB, 87
Comma separated values, 88
CSV, 88
DTD, 90
gettext, 78
go-i18n, 86
GWT properties, 81
i18next, 86
INI translations, 81, 82
Java properties, 81
Joomla translations, 82
JSON, 85
PHP strings, 84
PO, 78
Qt, 83
RC, 90
RESX, 88
Ruby YAML, 89
Ruby YAML Ain't Markup Language, 89
string resources, 83
TS, 83
XLIFF, 79
XML, 90
YAML, 89
YAML Ain't Markup Language, 89

G

get() (*метод wlc.Weblate*), 151
gettext
file format, 78
GITHUB_CREDENTIALS
setting, 309
GITHUB_TOKEN
setting, 309
GITHUB_USERNAME
setting, 309
GITLAB_CREDENTIALS
setting, 308
GITLAB_TOKEN

setting, 308
GITLAB_USERNAME
setting, 308
go-i18n
file format, 86
GOOGLE_ANALYTICS_ID
setting, 309
GWT properties
file format, 81

H

HIDE_REPO_CREDENTIALS
setting, 310

I

i18next
file format, 86
import_demo
weblate admin command, 346
import_json
weblate admin command, 346
import_json command line option
--ignore, 346
--main-component COMPONENT, 346
--project PROJECT, 346
--update, 346
import_memory
weblate admin command, 347
import_memory command line option
--language-map LANGMAP, 347
import_project
weblate admin command, 347
import_project command line option
--base-file-template TEMPLATE, 347
--file-format FORMAT, 348
--language-regex REGEX, 348
--license NAME, 348
--license-url URL, 348
--main-component, 348
--name-template TEMPLATE, 347
--new-base-template TEMPLATE, 347
--vcs NAME, 348
importuserdata
weblate admin command, 349
importusers
weblate admin command, 349
importusers command line option
--check, 349
INI translations
file format, 81, 82
install_addon
weblate admin command, 349
install_addon command line option
--addon ADDON, 349
--configuration CONFIG, 349
--update, 349
IP_BEHIND_REVERSE_PROXY
setting, 310
IP_PROXY_HEADER
setting, 310
IP_PROXY_OFFSET
setting, 310
iPad
перевод, 84
iPhone
перевод, 84

J

Java properties
file format, 81
Joomla translations
file format, 82
JSON
file format, 85

L

LEGAL_URL
setting, 311
LICENSE_EXTRA
setting, 311
LICENSE_FILTER
setting, 311
LICENSE_REQUIRED
setting, 312
LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH
setting, 312
list_languages
weblate admin command, 350
list_translators
weblate admin command, 350
list_translators command line option
--language-code, 350
list_versions
weblate admin command, 350
list-components
wlc command line option, 148
list-languages
wlc command line option, 148
list-projects
wlc command line option, 148
list-translations
wlc command line option, 148
load() (*метод wlc.config.WeblateConfig*), 151
loadpo
weblate admin command, 350
loadpo command line option
--force, 350
--lang LANGUAGE, 350
LOCALIZE_CDN_PATH
setting, 312
LOCALIZE_CDN_URL
setting, 312
lock
wlc command line option, 148
lock_translation
weblate admin command, 351

lock-status
    wlc command line option, 148
LOGIN_REQUIRED_URLS
    setting, 312
LOGIN_REQUIRED_URLS_EXCEPTIONS
    setting, 313
ls
    wlc command line option, 148

M

MACHINE_TRANSLATION_SERVICES
    setting, 314
main() (*в модуле wlc.main*), 152
MATOMO_SITE_ID
    setting, 313
MATOMO_URL
    setting, 313
move_language
    weblate admin command, 351
MT_APERTIUM_APY
    setting, 314
MT_AWS_ACCESS_KEY_ID
    setting, 315
MT_AWS_REGION
    setting, 315
MT_AWS_SECRET_ACCESS_KEY
    setting, 315
MT_BAIDU_ID
    setting, 315
MT_BAIDU_SECRET
    setting, 315
MT_DEEPL_API_VERSION
    setting, 315
MT_DEEPL_KEY
    setting, 316
MT_GOOGLE_CREDENTIALS
    setting, 316
MT_GOOGLE_KEY
    setting, 316
MT_GOOGLE_LOCATION
    setting, 316
MT_GOOGLE_PROJECT
    setting, 316
MT_MICROSOFT_BASE_URL
    setting, 317
MT_MICROSOFT_COGNITIVE_KEY
    setting, 317
MT_MICROSOFT_ENDPOINT_URL
    setting, 317
MT_MICROSOFT_REGION
    setting, 317
MT_MODERNMT_KEY
    setting, 317
MT_MODERNMT_URL
    setting, 317
MT_MYMEMORY_EMAIL
    setting, 318
MT_MYMEMORY_KEY
    setting, 318
MT_MYMEMORY_USER
    setting, 318
MT_NETEASE_KEY
    setting, 318
MT_NETEASE_SECRET
    setting, 318
MT_SAP_BASE_URL
    setting, 319
MT_SAP_PASSWORD
    setting, 320
MT_SAP_SANDBOX_APIKEY
    setting, 319
MT_SAP_USE_MT
    setting, 320
MT_SAP_USERNAME
    setting, 319
MT_SERVICES
    setting, 314
MT_TMSERVER
    setting, 318
MT_YANDEX_KEY
    setting, 319
MT_YOUDAO_ID
    setting, 319
MT_YOUDAO_SECRET
    setting, 319

N

NEARBY_MESSAGES
    setting, 320

P

PHP strings
    file format, 84
PIWIK_SITE_ID
    setting, 313
PIWIK_URL
    setting, 313
PO
    file format, 78
post() (*метод wlc.Weblate*), 151
pull
    wlc command line option, 148
push
    wlc command line option, 148
pushgit
    weblate admin command, 351
pushgit command line option
    --force-commit, 351
Python, 150

Q

Qt
    file format, 83

R

RATELIMIT_ATTEMPTS

setting, 320
RATELIMIT_LOCKOUT
setting, 320
RATELIMIT_WINDOW
setting, 320
RC
file format, 90
register_command() (*в модуле wlc.main*), 152
REGISTRATION_ALLOW_BACKENDS
setting, 321
REGISTRATION_CAPTCHA
setting, 321
REGISTRATION_EMAIL_MATCH
setting, 321
REGISTRATION_OPEN
setting, 321
repo
wlc command line option, 148
REPOSITORY_ALERT_THRESHOLD
setting, 322
REQUIRE_LOGIN
setting, 322
reset
wlc command line option, 148
REST, 101
RESX
file format, 88
RFC
RFC 4646, 75
Ruby YAML
file format, 89
Ruby YAML Ain't Markup Language
file format, 89

S

SECRET_KEY
setting, 192
SENTRY_DSN
setting, 322
SERVER_EMAIL
setting, 192
SESSION_COOKIE_AGE_AUTHENTICATED
setting, 322
SESSION_ENGINE
setting, 191
setting
ADMINS, 191
AKISMET_API_KEY, 300
ALLOWED_HOSTS, 191
ANONYMOUS_USER_NAME, 300
AUDITLOG_EXPIRY, 301
AUTH_LOCK_ATTEMPTS, 301
AUTH_TOKEN_VALID, 302
AUTO_UPDATE, 301
AUTOFIX_LIST, 302
AVATAR_URL_PREFIX, 302
BASE_DIR, 303
CHECK_LIST, 303
COMMENT_CLEANUP_DAYS, 304
COMMIT_PENDING_HOURS, 304
CSP_CONNECT_SRC, 303
CSP_FONT_SRC, 303
CSP_IMG_SRC, 303
CSP_SCRIPT_SRC, 303
CSP_STYLE_SRC, 303
DATA_DIR, 304
DATABASE_BACKUP, 305
DATABASES, 192
DEBUG, 192
DEFAULT_ACCESS_CONTROL, 305
DEFAULT_ADD_MESSAGE, 305
DEFAULT_ADDON_MESSAGE, 305
DEFAULT_ADDONS, 306
DEFAULT_COMMIT_MESSAGE, 305
DEFAULT_COMMITER_EMAIL, 306
DEFAULT_COMMITER_NAME, 306
DEFAULT_DELETE_MESSAGE, 305
DEFAULT_FROM_EMAIL, 192
DEFAULT_MERGE_MESSAGE, 305
DEFAULT_MERGE_STYLE, 307
DEFAULT_PULL_MESSAGE, 307
DEFAULT_RESTRICTED_COMPONENT, 305
DEFAULT_TRANSLATION_PROPAGATION, 307
ENABLE_AVATARS, 307
ENABLE_HOOKS, 307
ENABLE_HTTPS, 307
ENABLE_SHARING, 308
GITHUB_CREDENTIALS, 309
GITHUB_TOKEN, 309
GITHUB_USERNAME, 309
GITLAB_CREDENTIALS, 308
GITLAB_TOKEN, 308
GITLAB_USERNAME, 308
GOOGLE_ANALYTICS_ID, 309
HIDE_REPO_CREDENTIALS, 310
IP_BEHIND_REVERSE_PROXY, 310
IP_PROXY_HEADER, 310
IP_PROXY_OFFSET, 310
LEGAL_URL, 311
LICENSE_EXTRA, 311
LICENSE_FILTER, 311
LICENSE_REQUIRED, 312
LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH, 312
LOCALIZE_CDN_PATH, 312
LOCALIZE_CDN_URL, 312
LOGIN_REQUIRED_URLS, 312
LOGIN_REQUIRED_URLS_EXCEPTIONS, 313
MACHINE_TRANSLATION_SERVICES, 314
MATOMO_SITE_ID, 313
MATOMO_URL, 313
MT_APERTIUM_APY, 314
MT_AWS_ACCESS_KEY_ID, 315
MT_AWS_REGION, 315
MT_AWS_SECRET_ACCESS_KEY, 315

MT_BAIDU_ID, 315
MT_BAIDU_SECRET, 315
MT_DEEPL_API_VERSION, 315
MT_DEEPL_KEY, 316
MT_GOOGLE_CREDENTIALS, 316
MT_GOOGLE_KEY, 316
MT_GOOGLE_LOCATION, 316
MT_GOOGLE_PROJECT, 316
MT_MICROSOFT_BASE_URL, 317
MT_MICROSOFT_COGNITIVE_KEY, 317
MT_MICROSOFT_ENDPOINT_URL, 317
MT_MICROSOFT_REGION, 317
MT_MODERNMT_KEY, 317
MT_MODERNMT_URL, 317
MT_MYMEMORY_EMAIL, 318
MT_MYMEMORY_KEY, 318
MT_MYMEMORY_USER, 318
MT_NETEASE_KEY, 318
MT_NETEASE_SECRET, 318
MT_SAP_BASE_URL, 319
MT_SAP_PASSWORD, 320
MT_SAP_SANDBOX_APIKEY, 319
MT_SAP_USE_MT, 320
MT_SAP_USERNAME, 319
MT_SERVICES, 314
MT_TMSERVER, 318
MT_YANDEX_KEY, 319
MT_YOUDAO_ID, 319
MT_YOUDAO_SECRET, 319
NEARBY_MESSAGES, 320
PIWIK_SITE_ID, 313
PIWIK_URL, 313
RATELIMIT_ATTEMPTS, 320
RATELIMIT_LOCKOUT, 320
RATELIMIT_WINDOW, 320
REGISTRATION_ALLOW_BACKENDS, 321
REGISTRATION_CAPTCHA, 321
REGISTRATION_EMAIL_MATCH, 321
REGISTRATION_OPEN, 321
REPOSITORY_ALERT_THRESHOLD, 322
REQUIRE_LOGIN, 322
SECRET_KEY, 192
SENTRY_DSN, 322
SERVER_EMAIL, 192
SESSION_COOKIE_AGE_AUTHENTICATED, 322
SESSION_ENGINE, 191
SIMPLIFY_LANGUAGES, 323
SINGLE_PROJECT, 324
SITE_DOMAIN, 323
SITE_TITLE, 323
SPECIAL_CHARS, 324
STATUS_URL, 324
SUGGESTION_CLEANUP_DAYS, 324
URL_PREFIX, 324
VCS_BACKENDS, 325
VCS_CLONE_DEPTH, 325
WEBLATE_ADDONS, 325
WEBLATE_EXPORTERS, 326
WEBLATE_FORMATS, 326
WEBLATE_GPG_IDENTITY, 327
setupgroups
weblate admin command, 352
setupgroups command line option
--no-privs-update, 352
--no-projects-update, 352
setuplang
weblate admin command, 352
setuplang command line option
--no-update, 352
show
wlc command line option, 148
SIMPLIFY_LANGUAGES
setting, 323
SINGLE_PROJECT
setting, 324
SITE_DOMAIN
setting, 323
SITE_TITLE
setting, 323
SPECIAL_CHARS
setting, 324
statistics
wlc command line option, 148
STATUS_URL
setting, 324
string resources
file format, 83
SUGGESTION_CLEANUP_DAYS
setting, 324

T

TS
file format, 83

U

unlock
wlc command line option, 148
unlock_translation
weblate admin command, 352
updatechecks
weblate admin command, 352
updategit
weblate admin command, 353
upload
wlc command line option, 149
URL_PREFIX
setting, 324

V

VCS_BACKENDS
setting, 325
VCS_CLONE_DEPTH
setting, 325
version
wlc command line option, 148

W

Weblate (*класс в wlc*), 151
weblate admin command
    add_suggestions, 343
    auto_translate, 343
    celery_queues, 344
    checkgit, 344
    cleanuptrans, 345
    commit_pending, 344
    commitgit, 344
    createadmin, 345
    dump_memory, 345
    dumpuserdata, 346
    import_demo, 346
    import_json, 346
    import_memory, 347
    import_project, 347
    importuserdata, 349
    importusers, 349
    install_addon, 349
    list_languages, 350
    list_translators, 350
    list_versions, 350
    loadpo, 350
    lock_translation, 351
    move_language, 351
    pushgit, 351
    setupgroups, 352
    setuplang, 352
    unlock_translation, 352
    updatechecks, 352
    updategit, 353
WEBLATE_ADDONS
    setting, 325
WEBLATE_ADMIN_EMAIL, 157, 158, 162
WEBLATE_ADMIN_NAME, 157, 158
WEBLATE_ADMIN_PASSWORD, 154, 157, 158
WEBLATE_ALLOWED_HOSTS, 157, 191, 196, 323
WEBLATE_EMAIL_PORT, 168
WEBLATE_EMAIL_USE_SSL, 168
WEBLATE_EMAIL_USE_TLS, 168
WEBLATE_EXPORTERS
    setting, 326
WEBLATE_FORMATS
    setting, 326
WEBLATE_GPG_IDENTITY
    setting, 327
WEBLATE_LOCALIZE_CDN_PATH, 169
WEBLATE_REQUIRE_LOGIN, 322
WEBLATE_SILENCED_SYSTEM_CHECKS, 218
WEBLATE_SITE_DOMAIN, 194, 212, 323
WeblateConfig (*класс в wlc.config*), 151
WeblateException, 151
wlc, 146
    модуль, 151
wlc command line option
    --config PATH, 147
    --config-section SECTION, 147
    --convert, 149
    --format {csv,json,text,html}, 147
    --input, 149
    --key KEY, 147
    --output, 149
    --overwrite, 149
    --url URL, 147
    changes, 148
    cleanup, 148
    commit, 148
    download, 149
    list-components, 148
    list-languages, 148
    list-projects, 148
    list-translations, 148
    lock, 148
    lock-status, 148
    ls, 148
    pull, 148
    push, 148
    repo, 148
    reset, 148
    show, 148
    statistics, 148
    unlock, 148
    upload, 149
    version, 148
wlc.config
    модуль, 151
wlc.main
    модуль, 152

X

XLIFF
    file format, 79
XML
    file format, 90

Y

YAML
    file format, 89
YAML Ain't Markup Language
    file format, 89

⍰

двуязычный формат
    перевод, 75
модуль
    wlc, 151
    wlc.config, 151
    wlc.main, 152
одноязычный формат
    перевод, 75
перевод
    iPad, 84
    iPhone, 84
    двуязычный формат, 75
    одноязычный формат, 75

переменная окружения
CELERY_BACKUP_OPTIONS, 170
CELERY_BEAT_OPTIONS, 170
CELERY_MAIN_OPTIONS, 170
CELERY_NOTIFY_OPTIONS, 170
CELERY_TRANSLATE_OPTIONS, 170
POSTGRES_DATABASE, 166
POSTGRES_HOST, 166
POSTGRES_PASSWORD, 166
POSTGRES_PORT, 166
POSTGRES_SSL_MODE, 166
POSTGRES_USER, 166
REDIS_DB, 167
REDIS_HOST, 167
REDIS_PASSWORD, 167
REDIS_PORT, 167
REDIS_TLS, 167
REDIS_VERIFY_SSL, 167
ROLLBAR_ENVIRONMENT, 168
ROLLBAR_KEY, 168
SENTRY_DSN, 168
SENTRY_ENVIRONMENT, 169
SOCIAL_AUTH_SLACK_SECRET, 166
UWSGI_WORKERS, 170
WEBLATE_ADD_ADDONS, 169
WEBLATE_ADD_APPS, 169
WEBLATE_ADD_AUTOFIX, 169
WEBLATE_ADD_CHECK, 169
WEBLATE_ADD_LOGIN_REQUIRED_URLS_EXCEPTIONS, 160
WEBLATE_ADMIN_EMAIL, 157, 158, 162
WEBLATE_ADMIN_NAME, 157, 158
WEBLATE_ADMIN_PASSWORD, 154, 157, 158
WEBLATE_AKISMET_API_KEY, 161
WEBLATE_ALLOWED_HOSTS, 157, 158, 191, 196, 323
WEBLATE_AUTH_LDAP_BIND_DN, 163
WEBLATE_AUTH_LDAP_BIND_PASSWORD, 163
WEBLATE_AUTH_LDAP_CONNECTION_OPTION_REFERRALS, 163
WEBLATE_AUTH_LDAP_SERVER_URI, 163
WEBLATE_AUTH_LDAP_USER_ATTR_MAP, 163
WEBLATE_AUTH_LDAP_USER_DN_TEMPLATE, 163
WEBLATE_AUTH_LDAP_USER_SEARCH, 163
WEBLATE_AUTH_LDAP_USER_SEARCH_FILTER, 163
WEBLATE_AUTH_LDAP_USER_SEARCH_UNION, 163
WEBLATE_AUTH_LDAP_USER_SEARCH_UNION_DELIMITER, 163
WEBLATE_CSP_CONNECT_SRC, 161
WEBLATE_CSP_FONT_SRC, 161
WEBLATE_CSP_IMG_SRC, 161
WEBLATE_CSP_SCRIPT_SRC, 161
WEBLATE_CSP_STYLE_SRC, 161
WEBLATE_DATABASE_BACKUP, 167
WEBLATE_DEBUG, 157
WEBLATE_DEFAULT_ACCESS_CONTROL, 160
WEBLATE_DEFAULT_COMMITER_EMAIL, 161
WEBLATE_DEFAULT_COMMITER_NAME, 161
WEBLATE_DEFAULT_FROM_EMAIL, 158
WEBLATE_DEFAULT_RESTRICTED_COMPONENT, 161
WEBLATE_DEFAULT_TRANSLATION_PROPAGATION, 161
WEBLATE_EMAIL_BACKEND, 168
WEBLATE_EMAIL_HOST, 167
WEBLATE_EMAIL_HOST_PASSWORD, 168
WEBLATE_EMAIL_HOST_USER, 168
WEBLATE_EMAIL_PORT, 168
WEBLATE_EMAIL_USE_SSL, 168
WEBLATE_EMAIL_USE_TLS, 168
WEBLATE_ENABLE_HTTPS, 159
WEBLATE_GITHUB_TOKEN, 160
WEBLATE_GITHUB_USERNAME, 160
WEBLATE_GITLAB_TOKEN, 160
WEBLATE_GITLAB_USERNAME, 160
WEBLATE_GOOGLE_ANALYTICS_ID, 160
WEBLATE_GPG_IDENTITY, 161
WEBLATE_IP_PROXY_HEADER, 159
WEBLATE_LICENSE_FILTER, 161
WEBLATE_LOCALIZE_CDN_PATH, 169
WEBLATE_LOCALIZE_CDN_URL, 169
WEBLATE_LOGIN_REQUIRED_URLS_EXCEPTIONS, 160
WEBLATE_LOGLEVEL, 157
WEBLATE_MT_APERTIUM_APY, 162
WEBLATE_MT_AWS_ACCESS_KEY_ID, 162
WEBLATE_MT_AWS_REGION, 162
WEBLATE_MT_AWS_SECRET_ACCESS_KEY, 162
WEBLATE_MT_DEEPL_API_VERSION, 162
WEBLATE_MT_DEEPL_KEY, 162
WEBLATE_MT_GLOSBE_ENABLED, 162
WEBLATE_MT_GOOGLE_KEY, 162
WEBLATE_MT_MICROSOFT_BASE_URL, 162
WEBLATE_MT_MICROSOFT_COGNITIVE_KEY, 162
WEBLATE_MT_MICROSOFT_ENDPOINT_URL, 162
WEBLATE_MT_MICROSOFT_REGION, 162
WEBLATE_MT_MICROSOFT_TERMINOLOGY_ENABLED, 162
WEBLATE_MT_MODERNMT_KEY, 162
WEBLATE_MT_MYMEMORY_ENABLED, 162
WEBLATE_MT_SAP_BASE_URL, 162
WEBLATE_MT_SAP_PASSWORD, 163
WEBLATE_MT_SAP_SANDBOX_APIKEY, 163
WEBLATE_MT_SAP_USE_MT, 163
WEBLATE_MT_SAP_USERNAME, 163
WEBLATE_NO_EMAIL_AUTH, 166
WEBLATE_REGISTRATION_ALLOW_BACKENDS, 158

WEBLATE_REGISTRATION_OPEN, 158
WEBLATE_REMOVE_ADDONS, 169
WEBLATE_REMOVE_APPS, 169
WEBLATE_REMOVE_AUTOFIX, 169
WEBLATE_REMOVE_CHECK, 169
WEBLATE_REMOVE_LOGIN_REQUIRED_URLS_EXCEPTIONS, 160
WEBLATE_REQUIRE_LOGIN, 160, 322
WEBLATE_SAML_IDP_ENTITY_ID, 166
WEBLATE_SAML_IDP_URL, 166
WEBLATE_SAML_IDP_X509CERT, 166
WEBLATE_SECURE_PROXY_SSL_HEADER, 159
WEBLATE_SERVER_EMAIL, 158
WEBLATE_SILENCED_SYSTEM_CHECKS, 161, 218
WEBLATE_SIMPLIFY_LANGUAGES, 160
WEBLATE_SITE_DOMAIN, 157, 194, 212, 323
WEBLATE_SITE_TITLE, 157
WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_KEY, 165
WEBLATE_SOCIAL_AUTH_AZUREAD_OAUTH2_SECRET, 165
WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_KEY, 165
WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_SECRET, 165
WEBLATE_SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_TENANT_ID, 165
WEBLATE_SOCIAL_AUTH_BITBUCKET_KEY, 164
WEBLATE_SOCIAL_AUTH_BITBUCKET_SECRET, 164
WEBLATE_SOCIAL_AUTH_FACEBOOK_KEY, 164
WEBLATE_SOCIAL_AUTH_FACEBOOK_SECRET, 164
WEBLATE_SOCIAL_AUTH_FEDORA, 165
WEBLATE_SOCIAL_AUTH_GITHUB_KEY, 164
WEBLATE_SOCIAL_AUTH_GITHUB_SECRET, 164
WEBLATE_SOCIAL_AUTH_GITLAB_API_URL, 165
WEBLATE_SOCIAL_AUTH_GITLAB_KEY, 165
WEBLATE_SOCIAL_AUTH_GITLAB_SECRET, 165
WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_KEY, 164
WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET, 164
WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_WHITELISTED_DOMAINS, 164
WEBLATE_SOCIAL_AUTH_GOOGLE_OAUTH2_WHITELISTED_EMAILS, 164
WEBLATE_SOCIAL_AUTH_KEYCLOAK_ACCESS_TOKEN_URL, 165
WEBLATE_SOCIAL_AUTH_KEYCLOAK_ALGORITHM, 165
WEBLATE_SOCIAL_AUTH_KEYCLOAK_AUTHORIZATION_URL, 165
WEBLATE_SOCIAL_AUTH_KEYCLOAK_KEY, 165
WEBLATE_SOCIAL_AUTH_KEYCLOAK_PUBLIC_KEY, 165
WEBLATE_SOCIAL_AUTH_KEYCLOAK_SECRET, 165
WEBLATE_SOCIAL_AUTH_OPENSUSE, 165
WEBLATE_SOCIAL_AUTH_SLACK_KEY, 166
WEBLATE_SOCIAL_AUTH_UBUNTU, 165
WEBLATE_TIME_ZONE, 159
WEBLATE_URL_PREFIX, 161
WL_BRANCH, 297
WL_COMPONENT_NAME, 297
WL_COMPONENT_SLUG, 297
WL_COMPONENT_URL, 297
WL_ENGAGE_URL, 297
WL_FILE_FORMAT, 297
WL_FILEMASK, 297
WL_LANGUAGE, 297
WL_NEW_BASE, 297
WL_PATH, 297
WL_PREVIOUS_HEAD, 297
WL_PROJECT_NAME, 297
WL_PROJECT_SLUG, 297
WL_REPO, 296
WL_TEMPLATE, 297
WL_VCS, 296