Weblate Documentation

Выпуск 1.1

Michal Čihař

июл. 20, 2020

Оглавление

1 About Weblate **3**

1.1 Project goals . . . 3
1.2 Project name . . . 3
1.3 Project website . . . 3
1.4 Authors . . . 3

2 Usage guide **5**

2.1 Registration . . . 5
2.2 Profile information . . . 5
2.3 Projects structure . . . 5
2.4 Translation links . . . 6
2.5 Translating . . . 6
2.6 Dictionary . . . 6
2.7 Suggestions . . . 6
2.8 Machine translation . . . 7
2.9 Checks . . . 7

3 Quick starting guide **11**

3.1 Installing from sources . . . 11
3.2 Using prebuilt appliance . . . 11
3.3 Adding translation . . . 12

4 Installation instructions **13**

4.1 Requirements . . . 13
4.2 Installation . . . 13
4.3 Running server . . . 14
4.4 Prebuilt appliance . . . 16
4.5 Upgrading . . . 16
4.6 Migrating from Pootle . . . 17

5 Configuration **19**

6 Administration **21**

6.1 Adding new resources . . . 21
6.2 Project . . . 21
6.3 Subproject . . . 22
6.4 Automatic creation of subprojects . . . 22

6.5 Updating repositories . . . 22
6.6 Pushing changes . . . 23
6.7 Interacting with others . . . 23
6.8 Access control . . . 23
6.9 Lazy commits . . . 24
6.10 Customizing checks . . . 24

7 Management commands **27**

8 Weblate's Web API **29**
8.1 Notification hooks . . . 29
8.2 Exports . . . 29

9 Frequently Asked Questions **31**
9.1 Requests sometimes fail with too many open files error . . . 31
9.2 Fulltext search is too slow . . . 31
9.3 Rebuilding index has failed with «No space left on device» . . . 31
9.4 Does Weblate support other VCS than Git? . . . 32
9.5 Why does Weblate force to have show all po files in single tree? . . . 32

10 Changes **33**
10.1 weblate 1.1 . . . 33
10.2 weblate 1.0 . . . 33
10.3 weblate 0.9 . . . 33
10.4 weblate 0.8 . . . 34
10.5 weblate 0.7 . . . 35
10.6 weblate 0.6 . . . 35
10.7 weblate 0.5 . . . 35
10.8 weblate 0.4 . . . 35
10.9 weblate 0.3 . . . 36
10.10 weblate 0.2 . . . 36
10.11 weblate 0.1 . . . 36

11 License **37**

12 Indices and tables **39**

Алфавитный указатель **41**

Contents:

Глава 1

About Weblate

1.1 Project goals

Minimalistic web based translation with direct commit to git on each translation made. There is no plan in heavy conflict resolution as these should be primarily handled on git side.

1.2 Project name

The project is named as mixture of words web and translate.

1.3 Project website

You can find project website at <http://weblate.org/>, there is also demonstration server at <http://demo.weblate.org/>. This documentation can be browsed on <http://weblate.readthedocs.org/>.

1.4 Authors

This tool was written by Michal Čihař <michal@cihar.com>.

Глава 2

Usage guide

This document briefly covers how to translate application using Weblate.

2.1 Registration

While everybody can browse projects, view translations or suggest them, only registered users are allowed to actually save changes and are credited for every translation made.

You can register following two simple steps:

1. Fill out the registration form with your credentials
2. Activate registration by following in email you receive
3. Possibly adjust your profile to choose which languages you know

2.2 Profile information

User profile contains your preferences, name and email. Name and email are being used in Git commits, so keep this information accurate.

In preferences, you can choose user interface language, languages which you prefer to translate (list of these will be offered to you on main page) and secondary languages, whose translations will be shown to you while translating.

2.3 Projects structure

Each project can contain various subprojects. The reason for this structure is that all subprojects in a project are expected to have a lot in common. Whenever translation is made in single subproject, it is automatically propagated to others within same project (this is especially useful when translating more version of same project).

2.4 Translation links

Once you navigate to translation, you will be shown set of links which lead to translation. These are results of various checks, like untranslated or fuzzy strings. Should no other checks fire, there will be still link to all translations. Alternatively you can use search field to find translation you need to fix.

2.5 Translating

On translate page, you are shown source string and edit area for translating. Should the translation be plural, multiple source strings and edit areas are shown, each described with label for plural form.

Any special whitespace chars are underlined in red and indicated with grey symbols. Also more than one space is underlined in red to allow translator to keep formatting.

There are various extra information which can be shown on this page. Most of them are coming from the project source code (like context, comments or where the message is being used). When you configure secondary languages in your preferences, translation to these languages will be shown.

Bellow translation can be also shown suggestions from other users, which you can accept or delete.

2.5.1 Translation context

Translation context part allows you to see related information about current string.

Nearby messages Displays messages which are located nearby in translation file. These usually are also used in similar context and you might want to check them to keep translation consistent.

Similar messages Messages which are similar to currently one, which again can help you to stay consistent within translation.

All locations In case message appears in multiple places (eg. multiple subprojects), this tab shows all of them and for inconsistent translations (see *Inconsistent*) you can choose which one to use.

Dictionary Displays words from project dictionary which are used in current message.

Recent edits List of people who have changed this message recently using Weblate.

Project Project information like instructions for translators or information about Git repository.

2.6 Dictionary

Each project can have assigned dictionary for any language. This could be used for storing terminology for given project, so that translations are consistent. You can display terms from currently translated string in bottom tabs.

2.7 Suggestions

As an anonymous user, you have no other choice than making a suggestion. However if you are logged in you can still decide to make only a suggestion instead of saving translation, for example in case you are unsure about the translation and you want somebody else to review it.

2.8 Machine translation

Based on configuration and your language, Weblate provides buttons for following machine translation tools.

2.8.1 MyMemory

Huge translation memory with machine translation.

См.также:

http://mymemory.translated.net/

2.8.2 Apertium

A free/open-source machine translation platform providing translation to limited set of lanugages.

См.также:

http://www.apertium.org/

2.8.3 Microsoft Translator

Machine translation service provided by Microsoft.

См.также:

http://www.microsofttranslator.com/

2.9 Checks

Weblate does wide range of consistency checks on translated messages. The following section describes them in more detail. The checks take account also special rules for different languages, so if you think the result is wrong, please report a bug.

2.9.1 Not translated

The source and translated strings are same at least in one of plural forms. This checks ignores some strings which are quite usually same in all languages.

2.9.2 Starting newline

Source and translated do not both start with a newline.

2.9.3 Trailing newline

Source and translated do not both end with a newline.

2.9.4 Trailing space

Source and translated do not both end with a space.

2.9.5 Trailing stop

Source and translated do not both end with a full stop. Full stop is also checked in various language variants (Chinese, Japanese, Devanagari or Urdu).

2.9.6 Trailing colon

Source and translated do not both end with a colon or colon is not correctly spaced. This includes spacing rules for French or Breton. Colon is also checked in various language variants (Chinese or Japanese).

2.9.7 Trailing question

Source and translated do not both end with question mark or it is not correctly spaced. This includes spacing rules for French or Breton. Question mark is also checked in various language variants (Armenian, Arabic, Chinese, Korean, Japanese, Ethiopic, Vai or Coptic).

2.9.8 Trailing exclamation

Source and translated do not both end with exclamation mark or it is not correctly spaced. This includes spacing rules for French or Breton. Exclamation mark is also check in various langauge variants (Chinese, Japanese, Korean, Armenian, Limbu, Myanmar or Nko).

2.9.9 Python format

Python format string does not match source.

2.9.10 PHP format

PHP format string does not match source.

2.9.11 C format

C format string does not match source.

2.9.12 Missing plurals

Some plural forms are not translated. Check plural form definition to see for which counts each plural form is being used.

2.9.13 Inconsistent

More different translations of one string in a project. This can also lead to inconsistencies in displayed checks. You can find other translations of this string on *All locations* tab.

2.9.14 Invalid text direction

Text direction can be either LTR or RTL.

Глава 3

Quick starting guide

Примечание: This is just a quick guide for installing and starting to use Weblate, please check *Installation instructions* for more detailed instructions.

3.1 Installing from sources

1. Install all required dependencies, see *Requirements*.
2. Grab Weblate sources (either using Git or download a tarball) and unpack them.
3. Edit `settings.py` to match your setup. You will at least need to configure database connection (possibly adding user and creating the database). Check *Configuration* for Weblate specific configuration options.
4. Build Django tables and initial data:

```
./manage.py syncdb
./scripts/generate-locales # If you are using Git checkout
```

5. Configure webserver to serve Weblate, see *Running server*.

3.2 Using prebuilt appliance

1. Download the appliance and start it. You need to choose format depending on your target environment.
2. Everything should be set up immediatelly after boot, though you will want to adjust some settings to improve security, see *Prebuilt appliance*.

3.3 Adding translation

1. Open admin interface (http://example.org/admin/) and create project you want to translate. See *Project* for more details.
2. Create subproject which is the real resource for translating - it points to Git repository and selects which files to translate. See *Subproject* for more details.
3. Once above is completed (it can be lengthy process depending on size of your Git repository and number of messages to translate), you can start translating.

Глава 4

Installation instructions

4.1 Requirements

Django (>= 1.4) https://www.djangoproject.com/

Translate-toolkit http://translate.sourceforge.net/wiki/toolkit/index

GitPython (>= 0.3) https://github.com/gitpython-developers/GitPython

Django-registration (<= 0.7, 0.8 won't work) https://bitbucket.org/ubernostrum/django-registration/

Whoosh http://bitbucket.org/mchaput/whoosh/

4.2 Installation

Install all required components (see above) and adjust `settings.py`. You will probably want to adjust following options:

`ADMINS`

List of site administrators to receive notifications when something goes wrong, for example notifications on failed merge or Django errors.

См.также:

https://docs.djangoproject.com/en/1.4/ref/settings/#admins

`DATABASE`

Connectivity to database server, please check Django's documentation for more details.

Примечание: When using MySQL, don't forget to create database with UTF-8 encoding:

```
CREATE DATABASE <dbname> CHARACTER SET utf8;
```

См.также:

https://docs.djangoproject.com/en/1.4/ref/settings/#databases

`DEBUG`

Disable this for production server.

См.также:

https://docs.djangoproject.com/en/1.4/ref/settings/#debug

`DEFAULT_FROM_EMAIL`

Email sender address for outgoing email, for example registration emails.

См.также:

https://docs.djangoproject.com/en/1.4/ref/settings/#default-from-email

`SERVER_EMAIL`

Email used as sender address for sending emails to administrator, for example notifications on failed merge.

См.также:

https://docs.djangoproject.com/en/1.4/ref/settings/#server-email

After your configuration is ready, you can run `./manage.py syncdb` to create database structure. Now you should be able to create translation projects using admin interface.

You should also login to admin interface (on `/admin/` URL) and adjust default site name to match your domain.

См.также:

Access control

4.3 Running server

Running Weblate is not different from running any other Django based application.

It is recommended to serve static files directly by your webserver, you should use that for following paths:

`/media` Serves `media` directory from Weblate.

`/static/admin` Serves media files for Django admin interface (eg. `/usr/share/pyshared/django/contrib/admin/media/`).

Additionally you should setup rewrite rule to serve `media/favicon.ico` as `favicon.ico`.

См.также:

https://docs.djangoproject.com/en/1.4/howto/deployment/

4.3.1 Sample configuration for Lighttpd

The configuration for Lighttpd web server might look like following (available as `examples/lighttpd.conf`):

```
fastcgi.server = (
    "/weblate.fcgi" => (
        "main" => (
            "socket" => "/var/run/django/weblate.socket",
            "check-local" => "disable",
        )
    ),
)
alias.url = (
    "/media" => "/var/lib/django/weblate/weblate/media/",
    "/static/admin" => "/usr/share/pyshared/django/contrib/admin/static/admin/",
)

url.rewrite-once = (
    "^(/*media.*)$" => "$1",
    "^(/*static.*)$" => "$1",
    "^/*favicon\.ico$" => "/media/favicon.ico",
    "^/*robots\.txt$" => "/media/robots.txt",
    "^(/.*)$" => "/weblate.fcgi$1",
)

expire.url                  = (
    "/media/" => "access 1 months",
    "/static/" => "access 1 months",
    "/favicon.ico" => "access 1 months",
)
```

4.3.2 Sample configuration for Apache

Following configuration runs Weblate as WSGI, you need to have enabled mod_wsgi (available as `examples/apache.conf`):

```
#
# VirtualHost for weblate
#
WSGIPythonPath /usr/share/weblate
<VirtualHost *:80>
    ServerAdmin admin@image.weblate.org
    ServerName image.weblate.org

    DocumentRoot /usr/share/weblate/weblate/media/

    Alias /robots.txt /usr/share/weblate/weblate/media/robots.txt
    Alias /favicon.ico /usr/share/weblate/weblate/media/favicon.ico

    Alias /media/ /usr/share/weblate/weblate/media/
    Alias /doc/ /usr/share/doc/packages/weblate/html/
    Alias /static/admin /usr/lib/python2.7/site-packages/django/contrib/admin/static/admin/

    <Directory /usr/lib/python2.7/site-packages/django/contrib/admin/static/admin/>
        Order deny,allow
        Allow from all
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    </Directory>

    <Directory /usr/share/weblate/weblate/media/>
        Order deny,allow
        Allow from all
    </Directory>

    <Directory /usr/share/doc/packages/weblate/html/>
        Order deny,allow
        Allow from all
    </Directory>

    <Directory /usr/share/weblate/weblate/examples/>
        Order deny,allow
        Allow from all
    </Directory>

    WSGIScriptAlias / /usr/share/weblate/weblate/wsgi.py
    WSGIPassAuthorization On

    <Directory /usr/share/weblate/weblate>
        <Files wsgi.py>
        Order deny,allow
        Allow from all
        </Files>
    </Directory>

</VirtualHost>
```

4.4 Prebuilt appliance

Prebuilt appliance provides preconfigured Weblate running with MySQL database as backend and Apache as webserver. However it comes with standard set of passwords you will want to change:

Username	Password	Scope	Description
root	linux	System	Administrator account, use for local or SSH login
root		MySQL	MySQL administrator
weblate	weblate	MySQL	Account in MySQL database for storing Weblate data
admin	admin	Weblate	Weblate/Django admin user

The appliance is built using SUSE Studio and is based on openSUSE 12.1.

4.5 Upgrading

On upgrade to version 0.6 you should run `./manage.py syncdb` and `./manage.py setupgroups --move` to setup access control as described in installation section.

On upgrade to version 0.7 you should run `./manage.py syncdb` to setup new tables and `./manage.py rebuild_index` to build index for fulltext search.

On upgrade to version 0.8 you should run `./manage.py syncdb` to setup new tables, `./manage.py setupgroups` to update privileges setup and `./manage.py rebuild_index` to rebuild index for fulltext search.

On upgrade to version 0.9 file structure has changed. You need to move `repos` and `whoosh-index` to `weblate` folder. Also running `./manage.py syncdb`, `./manage.py setupgroups` and `./manage.py setuplang` is recommended to get latest updates of privileges and language definitions.

On upgrade to version 1.0 one field has been added to database, you need to invoke following SQL command to adjust it:

```
ALTER TABLE `trans_subproject` ADD `template` VARCHAR(200);
```

4.6 Migrating from Pootle

As Weblate was originally written as replacement from Pootle, it is supported to migrate user accounts from Pootle. All you need to do is to copy `auth_user` table from Pootle, user profiles will be automatically created for users as they log in and they will be asked to update their settings.

Глава 5

Configuration

All settings are stored in `settings.py` (as usual for Django).

`CHECK_LIST`

List of consistency checks to perform on translation.

См.также:

Checks, *Customizing checks*

`COMMIT_MESSAGE`

Message used on each commit Weblate does.

You can use following format strings in the message:

`%(language)s` Language code

`%(subproject)s` Subproject name

`%(project)s` Project name

`ENABLE_HOOKS`

Whether to enable anonymous remote hooks.

См.также:

Notification hooks

`GIT_ROOT`

Path where Weblate will store cloned Git repositories. Defaults to `repos` subdirectory.

`LAZY_COMMITS`

Delay creating Git commits until this is necessary. This heavily reduces number of commits generated by Weblate at expense of temporarily not being able to merge some changes as they are not yet committed.

См.также:

Lazy commits

`MT_APERTIUM_KEY`
API key for Apertium Web Service, you can register at http://api.apertium.org/register.jsp

`MT_MICROSOFT_KEY`
API key for Microsoft Translator service, you can register at http://www.bing.com/developers/createapp.aspx

`NEARBY_MESSAGES`
How many messages around current one to show during translating.

`OFFLOAD_INDEXING`
Offload updating of fulltext index to separate process. This heavily improves responsiveness of online operation on expense of slightly outdated index, which might still point to older content.

While enabling this, don't forget scheduling runs of `./manage.py update_index` in cron or similar tool.

This is recommended setup for production use.

`SIMILAR_MESSAGES`
Number of similar messages to lookup. This is not a hard limit, just a number Weblate tries to find if it is possible.

`SITE_TITLE`
Site title to be used in website and emails as well.

`WHOOSH_INDEX`
Directory where Whoosh fulltext indices will be stored. Defaults to `whoosh-index` subdirectory.

См.также:

https://docs.djangoproject.com/en/1.4/ref/settings/

Глава 6

Administration

Administration of Weblate is done through standard Django admin interface, which is available under `/admin/` URL.

6.1 Adding new resources

All translation resources need to be available as Git repositories and are organized as project/subproject structure.

Weblate supports wide range of translation formats supported by translate toolkit, for example:

- GNU Gettext
- XLIFF
- Java properties
- Windows RC files
- Qt Linguist .ts
- Symbian localization files
- CSV
- INI

См.также:

http://translate.sourceforge.net/wiki/toolkit/formats

6.2 Project

To add new resource to translate, you need to create translation project first. The project is sort of shelf, in which real translations are folded. All subprojects in same project share suggestions and dictionary, also the translations are automatically propagated through the all subproject in single project.

The project has only few attributes giving translators information about project.

6.3 Subproject

Subproject is real resource for translating. You enter Git repository location and file mask which files to translate and Weblate automatically fetches the Git and finds all translated files.

Should the language definition for translation be missing, empty definition is created and named as «cs_CZ (generated)». You should adjust the definition and report this back to Weblate authors so that missing language can be included in next release.

The subproject contains all important parameters for working with Git and getting translations out of it:

Repo Git repository used to pull changes.

This can be either real Git URL or `weblate://project/subproject` indicating that Git repository should be shared with another subproject.

Push Git URL used for pushing, this is completely optional and push support will be disabled when this is empty.

Repoweb URL of repository browser to display source files (location where messages are used). When empty no such links will be generated.

Branch Which brach to checkout from the Git and where to look for translations.

Filemask Mask of files to translate including path. It should include one * replacing language code. In case your Git repository contains more than one translation files (eg. more Gettext domains), you need to create separate subproject for each. For example `po/*.po` or `locale/*/LC_MESSAGES/django.po`.

Примечание: As setup of translation project includes fetching Git repositories, you might want to preseed these, repos are stored in path defined by *`GIT_ROOT`* in `settings.py` in `<project>/<subproject>` directories.

6.4 Automatic creation of subprojects

In case you have project with dozen of po files, you might want to import all at once. This can be achieved using `manage.py import_project`.

First you need to create project which will contain all subprojects and then it's just a matter of running `manage.py import_project`.

См.также:

Management commands

6.5 Updating repositories

You should set up some way how backend repositories are updated from their source. You can either use hooks (see *Notification hooks*) or just regularly run `./manage.py updategit --all`.

With Gettext po files, you might be often bitten by conflict in PO file headers. To avoid it, you can use shipped merge driver (`examples/git-merge-gettext-po`). To use it just put following configuration to your `.gitconfig`:

```
[merge "merge-gettext-po"]
  name = merge driver for gettext po files
  driver = /path/to/weblate/examples/git-merge-gettext-po %O %A %B
```

And enable it's use by defining proper attributes in given repository (eg. in `.git/info/attribute`):

```
*.po merge=merge-gettext-po
```

Примечание: This merge driver assumes the changes in POT files always are done in brach we're trying to merge.

См.также:

http://www.no-ack.org/2010/12/writing-git-merge-driver-for-po-files.html

6.6 Pushing changes

Each project can have configured push URL and in such case Weblate offers button to push changes to remote repo in web interface.

I case you will use SSH for pushing, you need to have key without passphrase (or use ssh-agent for Django) and the remote server needs to be verified by you first, otherwise push will fail.

6.7 Interacting with others

Weblate makes it easy to interact with others using it's API.

См.также:

Weblate's Web API

6.8 Access control

Weblate uses privileges system based on Django. It defines following extra privileges:

- Can upload translation [Users, Managers]
- Can overwrite with translation upload [Users, Managers]
- Can define author of translation upload [Managers]
- Can force commiting of translation [Managers]
- Can update translation from git [Managers]
- Can push translations to remote git [Managers]
- Can do automatic translation using other project strings [Managers]
- Can save translation [Users, Managers]

- Can accept suggestion [Users, Managers]
- Can accept suggestion [Users, Managers]
- Can import dictionary [Users, Managers]
- Can add dictionary [Users, Managers]
- Can change dictionary [Users, Managers]
- Can delete dictionary [Users, Managers]

The default setup (after you run `./manage.py setupgroups`) consists of two groups *Users* and *Managers* which have privileges as descibed above. All new users are automatically added to *Users* group.

To customize this setup, it is recommended to remove privileges from *Users* group and create additional groups with finer privileges (eg. *Translators* group, which will be allowed to save translations and manage suggestions) and add selected users to this group. You can do all this from Django admin interface.

6.9 Lazy commits

Default behaviour (configured by *LAZY_COMMITS*) of Weblate is to group commits from same author into one if possible. This heavily reduces number of commits, however you might need to explicitely tell to do the commits in case you want to get Git repository in sync, eg. for merge (this is by default allowed for Managers group, see *Access control*).

The changes are in this mode committed once any of following conditions is fulfilled:

- somebody else works on the translation
- merge from upstream occurs
- import of translation happens
- translation for a language is completed
- explicit commit is requested

6.10 Customizing checks

Weblate comes with wide range of consistency checks (see *Checks*), though they might not 100% cover all you want to check. The list of performed checks can be adjusted using *CHECK_LIST* and you can also add custom checks. All you need to do is to subclass `trans.checks.Check`, set few attributes and implement either `check` or `check_single` methods (first one if you want to deal with plurals in your code, the latter one does this for you). You will find below some examples.

6.10.1 Checking translation text does not contain «foo»

This is pretty simple check which just checks whether translation does not contain string «foo».

```
from weblate.trans.checks import Check
from django.utils.translation import ugettext_lazy as _

class FooCheck(Check):

    # Used as identifier for check, should be unique
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    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, flags, language, unit):
        return 'foo' in target
```

6.10.2 Checking Czech translation text plurals differ

Check using language information to verify that two plural forms in Czech language are not same.

```
from weblate.trans.checks import Check
from django.utils.translation import ugettext_lazy as _

class PluralCzechCheck(Check):

    # Used as identifier for check, should be unique
    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(self, sources, targets, flags, language, unit):
        if self.is_language(language, ['cs']):
            return targets[1] == targets[2]
        return False
```

Глава 7

Management commands

The ./manage.py is extended with following commands:

`checkgit`
: Prints current state of backend git repository.

  You can either define which subproject to check (eg. `weblate/master`) or use `--all` to check all existing subprojects.

`commitgit`
: Commits any possible pending changes to backend git repository.

  You can either define which subproject to check (eg. `weblate/master`) or use `--all` to check all existing subprojects.

`cleanuptrans`
: Cleanups orphnaed checks and translation suggestions.

`createadmin`
: Creates admin account with pasword admin.

`import_project <project> <gitrepo> <branch> <filemask>`
: Imports subprojects into project based on filemask.

  The *<project>* defines into which project subprojects should be imported (needs to exists).

  The *<gitrepo>* defines URL of Git repository to use, *<branch>* which branch to use.

  List of subprojects to create are automatically obtained from *<filemask>* - it has to contains one double wildcard (**), which is replacement for subproject.

  For example:

```
./manage.py import_project debian-handbook git://anonscm.debian.org/debian-handbook/debian-
↪handbook.git squeeze/master '*/**.po'
```

`loadpo`
: Reloads translations from disk (eg. in case you did some updates in Git repository).

`rebuild_index`
: Rebuilds index for fulltext search. This might be lengthy operation if you have huge set of translation units.

  You can use `--clean` to remove all words from database prior updating.

`update_index`
: Updates index for fulltext search when *`OFFLOAD_INDEXING`* is enabled.

  It is recommended to run this frequently (eg. every 5 minutes) to have index uptodate.

`setupgroups`
: Configures default groups and (if called with `--move`) assigns all users to default group.

  The option `--no-update` disables update of existing groups (only adds new ones).

  См.также:

  Access control

`setuplang`
: Setups list of languages (it has own list and all defined in translate-toolkit).

  The option `--no-update` disables update of existing languages (only add new ones).

`updatechecks`
: Updates all check for all units. This could be useful only on upgrades which do major changes to checks.

  You can either define which project or subproject to update (eg. `weblate/master`) or use `--all` to update all existing subprojects.

`updategit`
: Fetches remote Git repositories and updates internal cache.

  You can either define which project or subproject to update (eg. `weblate/master`) or use `--all` to update all existing subprojects.

Глава 8

Weblate's Web API

8.1 Notification hooks

Notification hooks allow external applications to notify weblate that Git repository has been updated.

`GET /hooks/update/(string:project)/(string:subproject)/`
Triggers update of a subproject (pulling from Git and scanning for translation changes).

`GET /hooks/update/(string:project)/`
Triggers update of all subprojects in a project (pulling from Git and scanning for translation changes).

`POST /hooks/github/`
Special hook for handling Github notifications and automatically updating matching subprojects.

Примечание: The GitHub notification relies on Git repository urls you use to be in form `git://github.com/owner/repo.git`, otherwise automatic detection of used repository will fail.

См.также:

http://help.github.com/post-receive-hooks/

8.2 Exports

Weblate provides various exports to allow you further process the data.

`GET /exports/stats/(string:project)/(string:subproject)/`
Retrieves statistics for given subproject in JSON format.

Example response:

```
[
    {
        "code": "cs",
        "fuzzy": 0,
        "fuzzy_percent": 0.0,
        "last_author": "Michal \u010ciha\u0159",
        "last_change": "2012-03-28T15:07:38+00:00",
        "name": "Czech",
        "total": 436,
        "translated": 436,
        "translated_percent": 100.0,
        "url": "http:/l10n.cihar.com/projects/weblate/master/cs/"
    },
    {
        "code": "nl",
        "fuzzy": 11,
        "fuzzy_percent": 2.5,
        "last_author": null,
        "last_change": null,
        "name": "Dutch",
        "total": 436,
        "translated": 319,
        "translated_percent": 73.2,
        "url": "http:/l10n.cihar.com/projects/weblate/master/nl/"
    },
    {
        "code": "el",
        "fuzzy": 21,
        "fuzzy_percent": 4.8,
        "last_author": null,
        "last_change": null,
        "name": "Greek",
        "total": 436,
        "translated": 312,
        "translated_percent": 71.6,
        "url": "http:/l10n.cihar.com/projects/weblate/master/el/"
    },
]
```

Глава 9

Frequently Asked Questions

9.1 Requests sometimes fail with too many open files error

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

The easiest way to do this is to run:

```
cd repos
for d in */* ; do
    pushd $d
    git gc
    popd
done
```

9.2 Fulltext search is too slow

Depending on various conditions (frequency of updates, server restarts and other), fulltext index might get too fragmented over time. It is recommended to rebuild it from scratch time to time:

```
./manage.py rebuild_index --clean
```

9.3 Rebuilding index has failed with «No space left on device»

Whoosh uses temporary directory to build indices. In case you have small /tmp (eg. using ramdisk), this might fail. Change used temporary directory by passing as `TEMP` variable:

```
TEMP=/path/to/big/temp ./manage.py rebuild_index --clean
```

9.4 Does Weblate support other VCS than Git?

Not currently. Weblate requires distributed VCS and could be probably adjusted to work with anything else than Git, but somebody has to implement this support.

9.5 Why does Weblate force to have show all po files in single tree?

Weblate was designed in a way that every po file is represented as single subproject. This is beneficial for translators, that they know what they are actually translating. If you feel your project should be translated as one, consider merging these po files. It will make life easier even for translators not using Weblate.

Примечание: In case there will be big demand for this feature, it might be implemented in future versions, but it's definitely not a priority for now.

Глава 10

Changes

10.1 weblate 1.1

Relased on July 4th 2012.

- Improved several translations.
- Better validation while creating subproject.
- Added support for shared git repositories across subprojects.
- Do not necessary commit on every attempt to pull remote repo.
- Added support for offloading indexing.

10.2 weblate 1.0

Relased on May 10th 2012.

- Improved validation while adding/saving subproject.
- Experimental support for Android resource files (needs patched ttkit).
- Updates from hooks are run in background.
- Improved installation instructions.
- Improved navigation in dictionary.

10.3 weblate 0.9

Relased on April 18th 2012.

- Fixed import of unknown languages.

- Improved listing of nearby messages.
- Improved several checks.
- Documentation updates.
- Added definition for several more languages.
- Various code cleanups.
- Documentation improvements.
- Changed file layout.
- Update helper scripts to Django 1.4.
- Improved navigation while translating.
- Better handling of po file renames.
- Better validation while creating subproject.
- Integrated full setup into syncdb.
- Added list of recent changes to all translation pages.
- Check for not translated strings ignores format string only messages.

10.4 weblate 0.8

Relased on April 3rd 2012.

- Replaced own full text search with Whoosh.
- Various fixes and improvements to checks.
- New command updatechecks.
- Lot of translation updates.
- Added dictionary for storing most frequently used terms.
- Added /admin/report/ for overview of repositories status.
- Machine translation services no longer block page loading.
- Management interface now contains also useful actions to update data.
- Records log of changes made by users.
- Ability to postpone commit to Git to generate less commits from single user.
- Possibility to browse failing checks.
- Automatic translation using already translated strings.
- New about page showing used versions.
- Django 1.4 compatibility.
- Ability to push changes to remote repo from web interface.
- Added review of translations done by others.

10.5 weblate 0.7

Relased on February 16th 2012.

- Direct support for GitHub notifications.
- Added support for cleaning up orphaned checks and translations.
- Displays nearby strings while translating.
- Displays similar strings while translating.
- Improved searching for string.

10.6 weblate 0.6

Relased on February 14th 2012.

- Added various checks for translated messages.
- Tunable access control.
- Improved handling of translations with new lines.
- Added client side sorting of tables.
- Please check upgrading instructions in case you are upgrading.

10.7 weblate 0.5

Relased on February 12th 2012.

- **Support for machine translation using following online services:**
  - Apertium
  - Microsoft Translator
  - MyMemory
- Several new translations.
- Improved merging of upstream changes.
- Better handle concurrent git pull and translation.
- Propagating works for fuzzy changes as well.
- Propagating works also for file upload.
- Fixed file downloads while using FastCGI (and possibly others).

10.8 weblate 0.4

Relased on February 8th 2012.

- Added usage guide to documentation.
- Fixed API hooks not to require CSRF protection.

10.9 weblate 0.3

Relased on February 8th 2012.

- Better display of source for plural translations.
- New documentation in Sphinx format.
- Displays secondary languages while translating.
- Improved error page to give list of existing projects.
- New per language stats.

10.10 weblate 0.2

Relased on February 7th 2012.

- Improved validation of several forms.
- Warn users on profile upgrade.
- Remember URL for login.
- Naming of text areas while entering plural forms.
- Automatic expanding of translation area.

10.11 weblate 0.1

Relased on February 6th 2012.

- Initial release.

Глава 11

License

Глава 12

Indices and tables

- genindex
- modindex
- search

Алфавитный указатель

Символы

Опция командной строки ./manage.py
- checkgit, 27
- cleanuptrans, 27
- commitgit, 27
- createadmin, 27
- import_project <project> <gitrepo> <branch> <filemask>, 27
- loadpo, 27
- rebuild_index, 27
- setupgroups, 28
- setuplang, 28
- update_index, 28
- updatechecks, 28
- updategit, 28

переменная окружения
- CHECK_LIST, 19, 24
- COMMIT_MESSAGE, 19
- ENABLE_HOOKS, 19
- GIT_ROOT, 19, 22
- LAZY_COMMITS, 19, 24
- MT_APERTIUM_KEY, 19
- MT_MICROSOFT_KEY, 20
- NEARBY_MESSAGES, 20
- OFFLOAD_INDEXING, 20, 28
- SIMILAR_MESSAGES, 20
- SITE_TITLE, 20
- WHOOSH_INDEX, 20

C

CHECK_LIST, 24

checkgit
- Опция командной строки ./manage.py, 27

cleanuptrans
- Опция командной строки ./manage.py, 27

commitgit
- Опция командной строки ./manage.py, 27

createadmin
- Опция командной строки ./manage.py, 27

G

GIT_ROOT, 22

I

import_project <project> <gitrepo> <branch> <filemask>
- Опция командной строки ./manage.py, 27

L

LAZY_COMMITS, 24

loadpo
- Опция командной строки ./manage.py, 27

O

OFFLOAD_INDEXING, 28

R

rebuild_index
- Опция командной строки ./manage.py, 27

S

setupgroups
- Опция командной строки ./manage.py, 28

setuplang
- Опция командной строки ./manage.py, 28

U

update_index
- Опция командной строки ./manage.py, 28

updatechecks
- Опция командной строки ./manage.py, 28

updategit
- Опция командной строки ./manage.py, 28