Docs as Code

Release

Apr 08, 2018

Contents

1 Introduction **3**
1.1 Intro . . . 3

2 Start **5**
2.1 Getting Started . . . 5

3 Zanata **7**
3.1 Zanata . . . 7

4 Sphinx **15**
4.1 Sphinx . . . 15

5 Readthedocs **19**
5.1 Readthedocs . . . 19

6 Indices and tables **21**

If you buy a car or a construction set for a wardrobe, the user manual or the building instruction will be shipped with the product. Also software, bought on a CD, has a handout or handbook. Your Open Source software should also has a manual. And should do it something better :-)

CHAPTER 1

Introduction

1.1 Intro

Here is a solution described as Sphinx how to reach a multi-language documentation service.

1.1.1 Why Sphinx?

Sphinx is used to document Python. Documentation strings from Python modules are fetched and Sphinx will generate documentation from that.

1.1.2 Output

For the output there are different Builders. You can choose between single- or multi-page HTML, TXT, PDF, or ePub. There are also other output formats like JSON, XML, or man-pages. With hieroglyph you can also generate slides. The builder will also make syntax- and link checks for the documents.

1.1.3 Input

The documents are written in restructuredtext

The syntax is pretty easy, i.e. print tables. Or print code:

```
# python -m sphinx.ext.intersphinx 'http://www.sphinx-doc.org/en/master/objects.inv
```

Or `download the document`.

Markdown is also supported as source.

1.1.4 Templating

Other things in Sphinx are templating. Default template engine is Jinja. You can use variables in loops to work with data from json files or others.

1.1.5 Themes

Last but not least to highlight are Themes. There are some themes included in Sphinx and the standard theme has multiple options to configure. But you can also build yuur own theme, or look what Python Packages are available for Sphinx Theme

1.1.6 Search Engine

A search engine is included.

1.1.7 Not enough?

And mark a timestamp without and script language

Apr 08, 2018

1.1.8 And who is using Sphinx?

- Ansible Doc
- OpenStack Doc

others:

CHAPTER 2

Start

2.1 Getting Started

Common workflow of documentation is:

`Write -> Build -> Publish`

Our workflow will be:

`Write -> Build -> Translate -> Publish`

2.1.1 Text Editor

There ara variuous tools for different platforms with RST support

- Online
- vim extension
- Notepad++
- Visual Studio Code

2.1.2 Build Tools

It's all described in *Sphinx*. Sphinx package includes all tools to check and build the documents.

2.1.3 Translation Platform

In the chapter *Zanata* it's described, how Zanata can help us to translate text strings in differenz languages.

2.1.4 Publishing

Compiled HTML pages can served on our own webserver (see *LXD Container*) Or we use the pretty online service *Readthedocs*)

2.1.5 LXD Container

To build our own sphinx build and translation platform, we simple use LXD Container. If your LXD environment is initialized, just do it:

```
lxc launch ubuntu:16.04 zanata
lxc exec zanata -- bash
```

Of course, any other Ubuntu system can be used.

CHAPTER 3

Zanata

3.1 Zanata

3.1.1 Translate the world

or: How to build your own infrastructure for multi-language documentation.

Imagine there are about 194 countries on the world, most of them with one or more then one language. And countless dialects, like Saxon. How you want to reach all your customer if your products or services are only described in ONE language?

Zanata is a web-based translation platform for translators, content creators and developers to manage localization projects. Zanata is Open Source. The Installation is divided in server and client.

3.1.2 Server

Zanata requires Java, Apache, Wildfly, MySQL, and an e-mail service. For easy installation we use Puppet, specially puppet-zanata All installation steps are done by **root**

Install deb packages:

```
apt-get update
apt install -y git puppet-common python-pip
pip install pip -U
```

Clone Git repositories:

```
git clone -b dev https://github.com/eumel8/puppet-zanata /etc/puppet/modules/zanata
git clone -b v1.2.4 https://github.com/biemond/biemond-wildfly /etc/puppet/modules/
↪wildfly
git clone https://github.com/puppetlabs/puppetlabs-apache /etc/puppet/modules/apache
git clone -b 3.10.0 https://github.com/puppetlabs/puppetlabs-mysql /etc/puppet/
↪modules/mysql
git clone https://github.com/puppetlabs/puppetlabs-stdlib /etc/puppet/modules/stdlib
git clone https://git.openstack.org/openstack-infra/puppet-jeepyb /etc/puppet/modules/
↪jeepyb
git clone -b v0.5.1 https://github.com/voxpupuli/puppet-archive /etc/puppet/modules/
↪archive
git clone https://git.openstack.org/openstack-infra/puppet-httpd /etc/puppet/modules/
↪httpd
git clone https://git.openstack.org/openstack-infra/puppet-logrotate /etc/puppet/
↪modules/logrotate
git clone https://git.openstack.org/openstack-infra/puppet-vcsrepo /etc/puppet/
↪modules/vcsrepo
git clone -b 1.1.3 https://github.com/puppetlabs/puppetlabs-firewall /etc/puppet/
↪modules/firewall
git clone -b 1.5.0 https://github.com/camptocamp/puppet-postfix.git /etc/puppet/
↪modules/postfix
```

Create MySQL client file:

```
cat > /root/.my.cnf << EOF
[client]
user=root
host=localhost
password=12345678
EOF
```

Create Puppet manifest:

```
cat > /etc/puppet/manifest/site.pp << EOF
#!/usr/bin/puppet apply

  class {'::mysql::server':
    root_password           => "12345678",
    remove_default_accounts => true,
    create_root_my_cnf      => true,
  }
```

```
  mysql::db { 'zanata':
    user     => 'zanata',
    password => '12345678',
    host     => 'localhost',
    grant    => ['all'],
    require  => [
      Class['mysql::server'],
      File['/root/.my.cnf'],
    ],
  }

  include postfix

  class { '::zanata':
    mysql_host                  => 'localhost',
    zanata_db_username          => 'zanata',
    zanata_db_password          => '12345678',
    zanata_openid_provider_url  => 'https://openstackid.openstack.org',
    zanata_listeners            => ['ajp'],
    zanata_admin_users          => 'admin',
    zanata_default_from_address => 'noreply@example.com',
    zanata_main_version         => 4,
    zanata_url                  => 'https://github.com/zanata/zanata-platform/
↪releases/download/platform-4.3.3/zanata-4.3.3-wildfly.zip',
    zanata_checksum             => 'eaf8bd07401dade758b677007d2358f173193d17',
    zanata_wildfly_version      => '10.1.0',
    zanata_wildfly_install_url  => 'https://repo1.maven.org/maven2/org/wildfly/
↪wildfly-dist/10.1.0.Final/wildfly-dist-10.1.0.Final.tar.gz',
  }


  class { '::zanata::apache':
    vhost_name              => $vhost_name,
    require                 => Class['::zanata']
  }

  include logrotate
  logrotate::file { 'console.log':
    log     => '/var/log/wildfly/console.log',
    options => [
                'daily',
                'rotate 30',
                'missingok',
                'dateext',
                'copytruncate',
                'compress',
                'delaycompress',
                'notifempty',
                'maxage 30',
                ],
    require => Service['wildfly'],
  }

  include jeepyb

  logrotate::file { 'register-zanata-projects.log':
    log     => '/var/log/register-zanata-projects.log',
```

```
    options => [
      'compress',
      'missingok',
      'rotate 30',
      'daily',
      'notifempty',
      'copytruncate',
    ],
  }
EOF
```

Execute the manifest:

```
/etc/puppet/manifest/site.pp
```

Check on the web browser if Zanata server is running. Depends on the filesystem , it's required to configure JBoss:

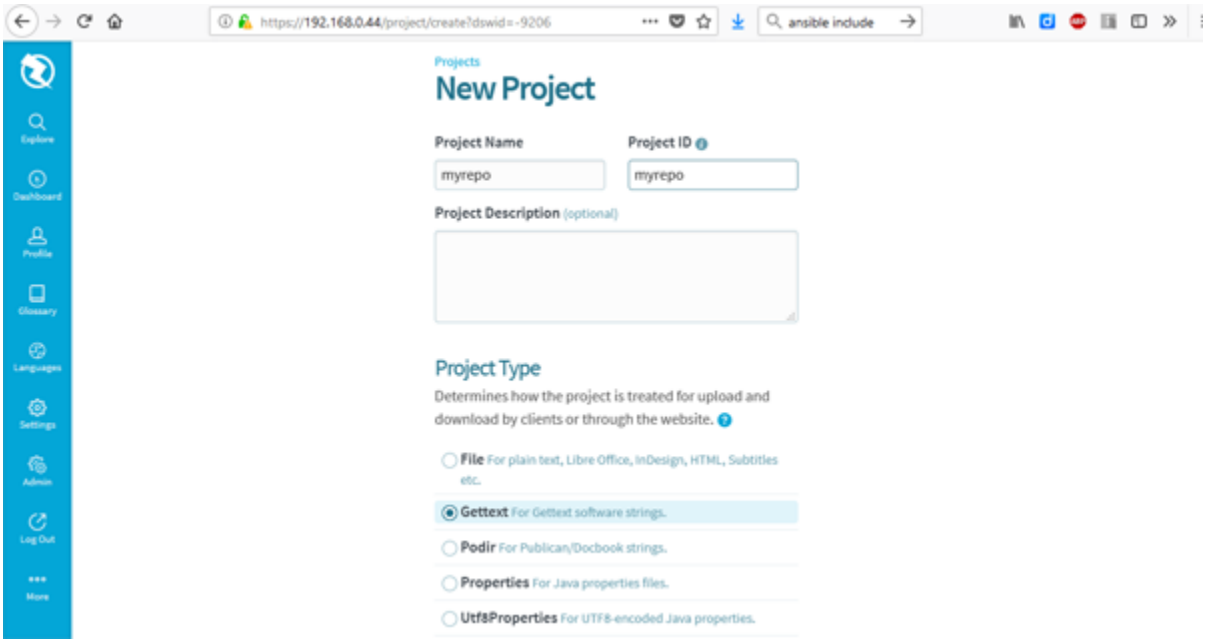
```
cd /opt/wildfly/bin
./jboss-cli.sh
connect
/subsystem=messaging-activemq/server=default:write-attribute(name=journal-type,␣
↪value=NIO)
exit
systemctl restart wildfly.service
```

With Puppet the configured admin user is **admin**. That means, if you start the register process, go through the OpenID login process, you should use the username **admin** to get admin privileges for your account. After login generate your API key in User Settings->Client in your profile. The API key is for authorization on the server:

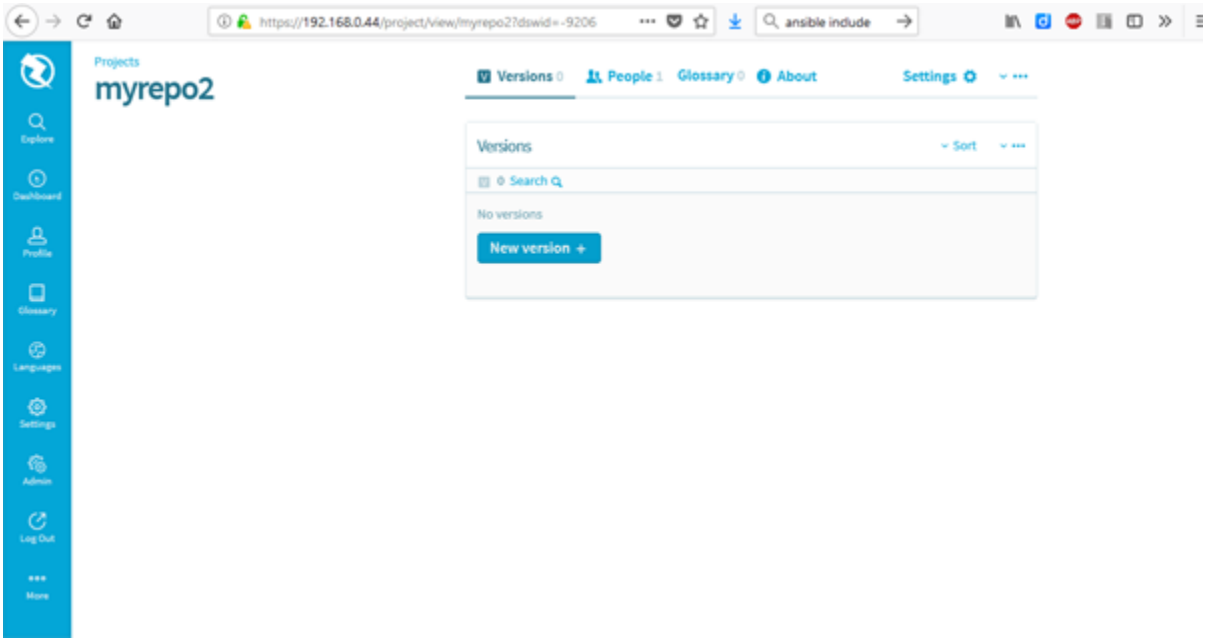
```
cat > ~/.config/zanata.ini << EOF
[servers]
zanata.lxd.url=https://zanata.lxd/
zanata.lxd.username=admin
zanata.lxd.key=cc8492e637f9f2c161d89bb3fb776783
EOF
```

Now you may setup a new project in Zanata and add languages to the project. Of course, you should join a language team. Here are some screenshots from what do you can expect:

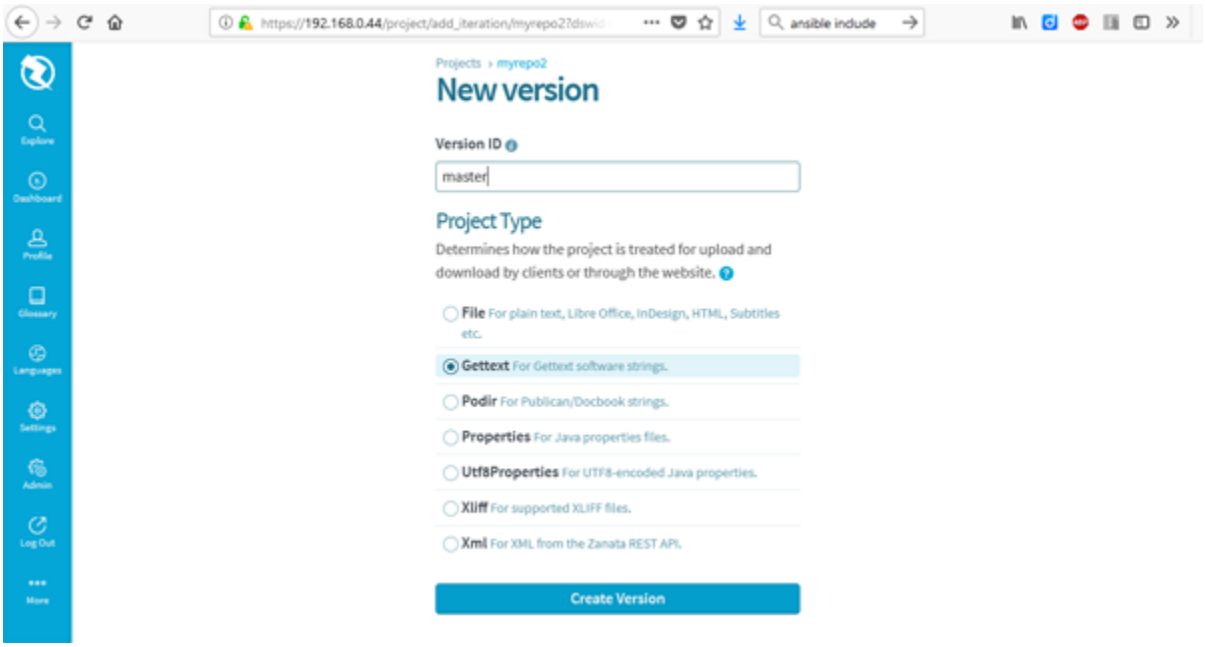
New project:



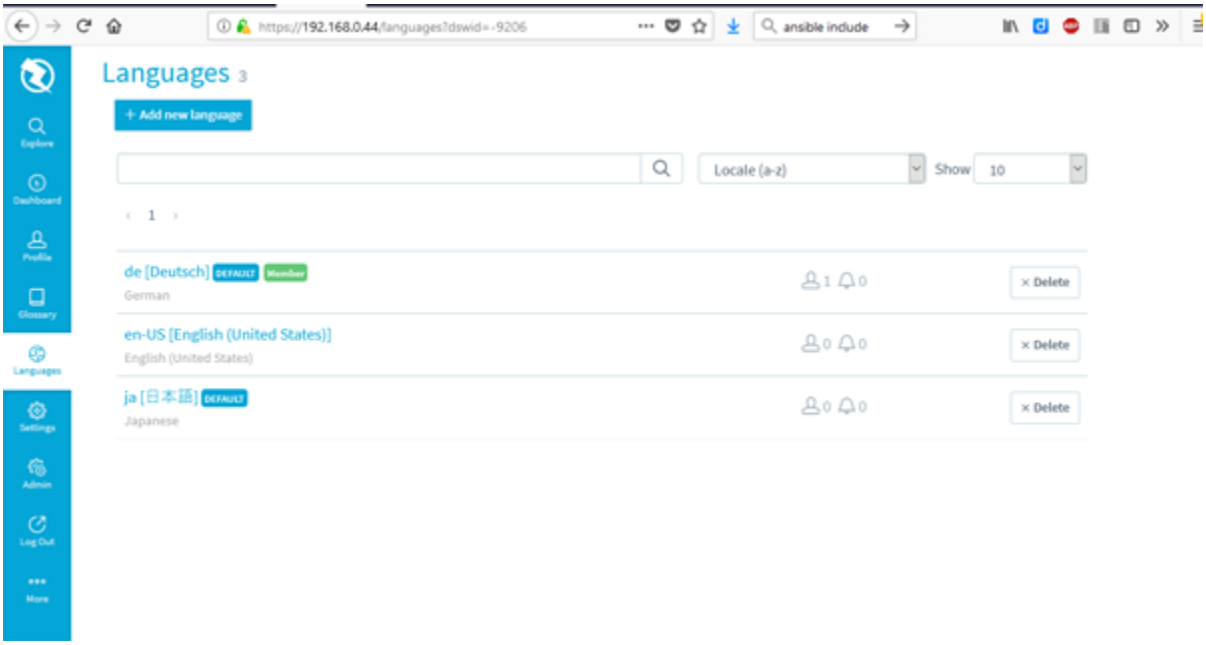
New version:



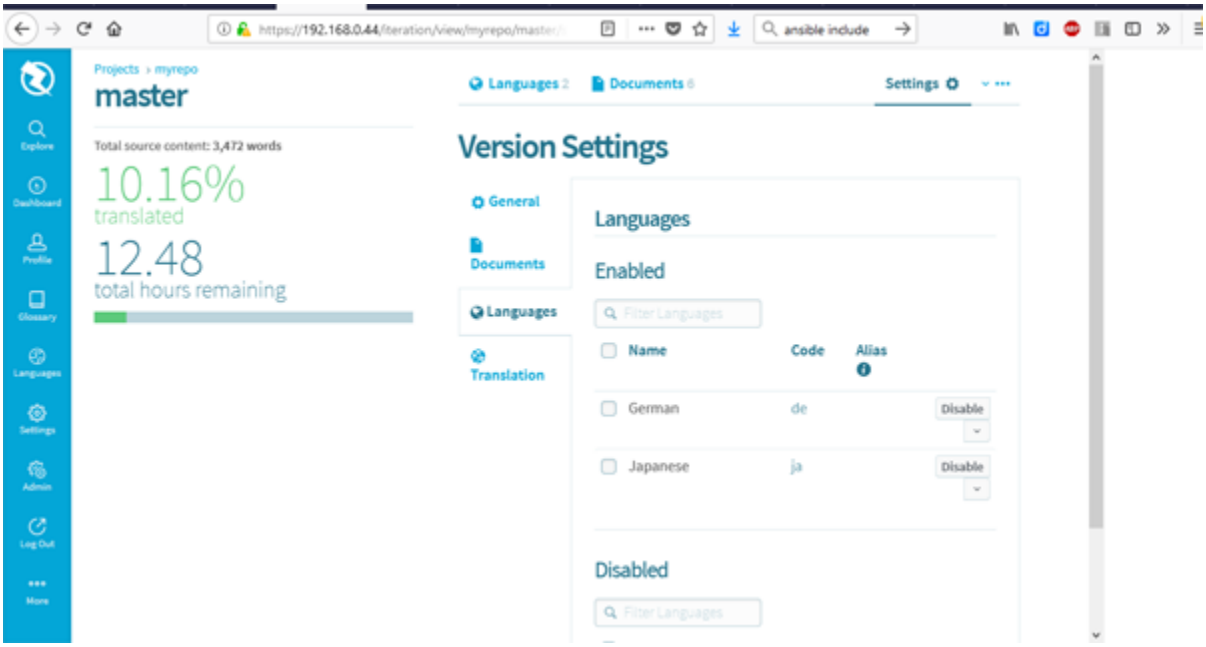
Configure gettext:



Configure languages:



Add languages to project:



More information for Zanata usage are available on OpenStack Contributor Guide

Zanata Docs .

3.1.3 Client

Thank goodness Zanata client installation is much easier:

```
cd /opt
curl -s http://repo2.maven.org/maven2/org/zanata/zanata-cli/4.3.3/zanata-cli-4.3.3-
↪dist.tar.gz | tar xz
ln -s /opt/zanata-cli-4.3.3/bin/zanata-cli /usr/local/bin
```

That's it :-)

CHAPTER 4

Sphinx

4.1 Sphinx

Sphinx, the Python Documentation Generator, provides Internationalization . The workflow is described there.

btw: documentation reference can be build with intersphinx Check available options:

```
python -m sphinx.ext.intersphinx 'http://www.sphinx-doc.org/en/master/objects.inv'
```

4.1.1 Installation

Install requirements:

```
pip install -r requirements.txt
```

4.1.2 Init

To initial a new documentation project use:

```
sphinx-quickstart
```

4.1.3 Zanata Project File

To interact to Zanata server with the client a project configuration file named zanata.xml is required:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<config xmlns="http://zanata.org/namespace/config/">
  <url>https://zanata.lxd/</url>
  <project>myrepo</project>
  <project-version>master</project-version>
```

```
  <project-type>gettext</project-type>
  <src-dir>.</src-dir>
  <trans-dir>.</trans-dir>
  <rules>
    <rule pattern="**/*.pot">{path}/{locale_with_underscore}/LC_MESSAGES/{filename}.po
↪</rule>
  </rules>
</config>
```

url must match with url name in zanata.ini

4.1.4 Gettext

Build gettext files for Zanata:

```
sphinx-build -b gettext doc/myguide/source/ doc/myguide/source/locale/
```

4.1.5 Push to server

Push pot files to Zanata:

```
zanata-cli -e push --disable-ssl-cert
```

Normaly self-signed certificate will used in private environment like this. The option –disable-ssl-cert will force the client to connect.

Now the translation files are on the server and translatable in the configured languages. If this done you can download it again.

4.1.6 Pull from server

Pull po files from Zanata:

```
zanata-cli -e pull -l de  --disable-ssl-cert
```

Use language option -l de for German files, without that all language files will be download.

Optional: If you modified the language files locally, you can push them to the server:

```
zanata-cli -e push --push-type trans --disable-ssl-cert
```

4.1.7 Build and publish

At the end, build and publish the docs:

```
sphinx-build -b html doc/myguide/source/ doc/myguide/build/html/en
sphinx-build -b html doc/myguide/source/ doc/myguide/build/html/de -D language='de'
```

Or build and publish slides within hieroglyph module:

```
sphinx-build -b slides doc/myguide/source/ doc/myguide/build/slides/en
sphinx-build -b slides doc/myguide/source/ doc/myguide/build/slides/de -D language='de
↪'
```

CHAPTER 5

Readthedocs

5.1 Readthedocs

5.1.1 Introduction

Readthedocs is an online service to host technical documentation, especially in Sphinx format. If your documentation code lives on GitHub a webhook can configured to trigger Readthedocs to fetch and build Sphinx documentation.

5.1.2 Workflow

```
Git Commit -> Webhook -> Sphinx Build -> Publish
```

5.1.3 Use cases

Readthedocs provides different output formats like HTML, ePub, and PDF in different languages and different versions (branches).

You can configure your own sub-domain like http://docs-i18n.readthedocs.io/ or use your own name space and configure aliases or forwards.

5.1.4 Example

Ansible OTC

CHAPTER 6

Indices and tables

- search