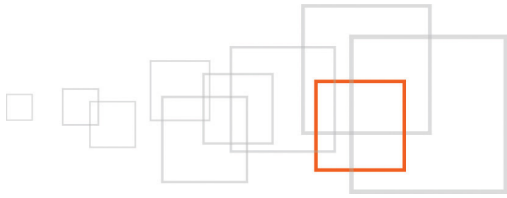
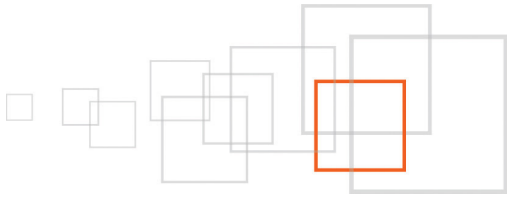
Translating & Localizing eZ Publish using GIT

By Nicolas Pastorino
Andre Rømcke
Review : Bertrand Dunogier
http://share.ez.no



Index



1 Goal description

At the end of this tutorial, you should be able to contribute to the eZ Publish Localization project using GIT. Localizing means adding new translations to eZ Publish, enhancing existing ones, and adding or enhancing locales (currencies, week days names, date formats, etc.).

2 Introduction

GIT, a version control system, like SVN, is now used to develop the eZ Publish Community Project. This fantastic tool allows for a very smooth collaboration on development. In our specific case, the Localization project, and while waiting for a more definitive process removing technical requirements (like having to know SNV or GIT), we will take an intermediate step and use GIT.

As a reminder, the Localization project was so far handled through :

- a project on projects.ez.no, centralizing registration to the team, and the actual localization work : http://projects.ez.no/ezpublish_translation
- a set of forums, for translators to exchange on the topic : http://share.ez.no/forums/translation

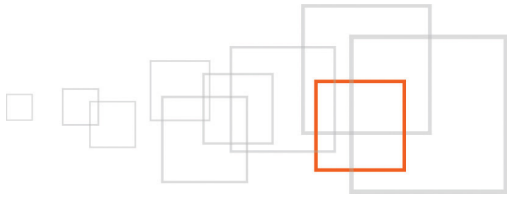
These two tools remain, the only difference being where the translations themselves are stored : it will now be on github.com.

3 Pre-requisites and target population

This tutorial is aimed at anybody willing to help localize eZ Publish. This can mean adding a new language for eZ Publish's administration interface, updating an existing one, adding a new locale (dates, currency formats, etc) or enhancing it.

As mentioned in the introduction, the one pre-requisite before starting on localization is to become a member of the Localization project. To do so, log-in on projects.ez.no, and then click the “Register membership” button on this page : http://projects.ez.no/ezpublish_translation/team/members

You will also need to install the GIT toolbox on your machine, and have a github.com account. Documentation on installing GIT for any platform and setting up Github is best explained on Githubs excellent help pages: http://help.github.com/
There are also some useful links and comments on the earlier share.ez.no blog post: “eZ running on GIT”



4 How to collaborate

When using GIT, the collaboration process looks as follows :

Step	How many times do I do this ?	What is it exactly ?
1. Fork	Once	Creating a copy of the main eZ Publish repository, logically linked with the original one. The GIT way. All it takes is one click.
2. Create localization branch	Once	Keeping your localization efforts in a clean space in your GIT version of eZ Publish. Lets you work on other parts of eZ Publish if you want to, with no conflict. Branching is seamless with GIT.
3. Do changes + Commit	At least once	Make changes, commit, and repeat if you want to do several independent changes.
4. Push	At least once	Synchronizing your local GIT clone with your account on github. After you have pushed you can ask people for feedback on your commits and then re-iterate 2 and 3 until you are happy with the state of your branch, at that point you can create a "Pull Request" (see further below).
5. Emit a pull request	At least once	Once satisfied with your localization job, sending your changes back to the main eZ Publish GIT repository, so that others can see them or elaborate on them. This happens on Github, and is equivalent to asking the 'ezsystems' user to merge in your changes.

As you could notice we start using the GIT/Github jargon here. If you are not super acquainted with it, we recommend to read the appendix section. Let us dive into the details of this workflow.

5 "Forking" eZ Publish

Log-in on github, navigate to https://github.com/ezsystems/ezpublish and click the "Fork" button. This will create a fork of eZ Publish on your own account page (yes, this implies you created your github account beforehand. This is free).

Note: the fork will not update itself with changes done in the original repository, this is done by pulling in

changes and pushing it to your fork (how to do this is explained a bit below).

6 Cloning eZ Publish

We will use command line here. Do not be afraid of this, the git utility is brilliantly done, and will certainly reconcile you with command line. Graphical interfaces exist as well, like TortoiseGIT for windows, but they are not covered in this tutorial and only those using them will be able to assist you.

Using command line, go to the place you want to checkout eZ Publish, for instance the localhost / www folder that your web server points to, to be able to execute eZ Publish directly for testing. Then, take the following (optionally specifying target <folder_name>, and <GitUser> as your github account alias) :

```
$ git clone git@github.com:<GitUser>/ezpublish.git [<folder_name>]
$ cd [ezpublish|<folder_name>]
```

Now we need to add ezsystems as an additional read only remote location, so that changes on the official repository can easily be applied to your own fork. In the following code examples we call it 'upstream':

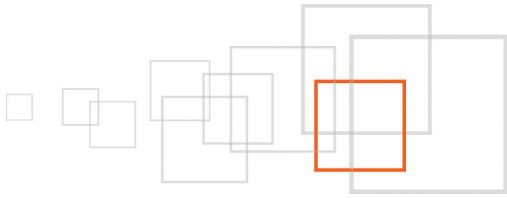
```
$ git remote add upstream git://github.com/ezsystems/ezpublish.git
$ git fetch upstream
```

Optionally we make master branch track upstream, making it easier to pull in changes:

```
$ git config branch.master.remote upstream
```

After that keeping your checkout up to date is a matter of:

```
$ git pull
```



7 Creating and working with your localization branch

Creating a branch is easy, and will let you properly isolate the work on localization to an independent part of your local repository. This amongst other gives you the possibility to work on other things than localization, independently (bug-fixes, features). Here is how to create your localization branch:

Checkout

1. If you haven't created your localization branch yet:
   ```
   $ git checkout -b localization upstream/master
   ```

2. If you already have the branch, but only remotely in your fork:
   ```
   $ git checkout -b localization origin/localization
   ```
   And specify that it should track upstream/master:
   ```
   $ git config branch.localization.remote upstream/master
   ```

3. If you have the branch locally already (check with "`$ git branch`"):
   ```
   $ git checkout localization
   ```

Start localizing

Once you are all set, you can effectively start localizing eZ Publish.

Installing eZ Publish

Firstly, you will have to install eZ Publish. This helps visually see the changes you are making. As for any eZ Publish installation, point your browser to the path where you stored your git repository and the wizard will kick-in. The URL could look like http://localhost/www/ezpublish-4.4. Then follow the steps of the installation wizard. You will be prompted for language choice during the wizard : the main one, and as many secondary ones as you want. We recommend to choose, as the main one, the language towards which you want to translate or localize.

Installing the translation tools

Check the following page : http://projects.ez.no/ezpublish_translation under "Start translating". Download and install linguist, the preferred tool for editing translation.ts files, the files used for translations in eZ Publish.

Prior to translating

Always repatriate possible changes from others before starting working on localization. You can do so by running the following command :

```
$ git pull
```

Then open the administration interface, which shows most of the strings to be translated. This can help visually track the untranslated strings.

Translating, localizing

Let us imagine you are working on the French (France) translation. Using linguist, you will open the following file :

<ezpublish-root>/share/translations/fre-FR/translation.ts

and start making the changes. After having saved them, to test them in your eZ Publish instance, if they are part of the graphical user interface (administration interface for instance), click the “Clear” button on the right-hand vertical toolbar, right under the “Clear cache” entry. This will clear the translation cache, and show your the newly translated strings.

Localization is a community effort, we recommend to announce that you are are working on a specific translation in the forums, so that others can synchronize with you. Also, these forums are helpful to discuss potentially difficult points. In our example, the forum would be : http://share.ez.no/forums/translation/french

Sharing your changes

Regularly, share your changes with the rest of the Localization team. You can do so by following the three “Commit”, “Push”, “Pull request” below.

Commit

If you have added any new files, use “`$ git add <file>`” to stage it for commit. You can, at any moment, see the list of your changes by running :

```
$ git status
```

The output of this command is pretty explanatory.

You then have to commit the changes. You can do so, still from the command line, and after having checked that only the necessary files were modified (cf 'git status' above), and that new files, if any, were added (cf 'git add <file>' above) :

```
$ git commit -am "localization: updated esl-ES translation.ts"
```

It is important to add a message to your commit : this informs other members of the Localization team of what you worked on. Other examples of commit messages could be :

- "localization: update ara-SA locale file"
- "localization: update esl-ES translation.ts and ara-SA locale"

As you can see, all commit messages are prefixed by "localization:", which helps tracking all localization-related activity. We recommend to follow this pattern. Then, when updating a translation, we recommend to mention the full locale code ("esl-ES"), followed by "translation.ts". If updating a locale file, please also mention full locale code ("ara-SA" for instance), followed by "locale". And if you update both or several of those, please have as many corresponding entries, following the naming pattern.

Push

When you feel confident that you want to share your work with the world, you will want to push. It is recommended to do this quite often (along with pull requests, see further below), at least every time you have finished working on a self-contained set of changes. This allows for a smoother collaboration within the Localization team, where translators of a given language most of the time are working on the same file.

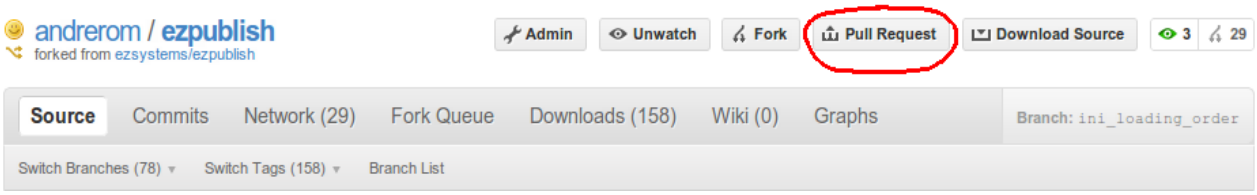
Here is how you can push your localization branch :

```
$ git pull
$ git push origin localization
```

When this is done, git will respond with something like "Your branch is ahead of 'upstream/master' by 1 commit." basically giving you an idea on how many differences there are between your topic branch and the branch you track (master). This is also shown when you use "$ git status".
You may also see statuses like "Your branch .. is ahead .. with 1 commits and behind by 50 commits" meaning you have 1 commit in your topic branch and 50 new commits in master since you last pulled in changes.

Pull Request

When you are confident that your contribution is ready for inclusion in eZ Publish, then all you need to do is click on "Pull request" in the Github GUI on the specific topic branch as selected using the "Switch Branches" drop-down. As said above, it is recommended to do this quite often, at least every time you have finished working on a self-contained set of changes. This allows for a smoother collaboration within the Localization team, where translators of a given language most of the time are working on the same file.



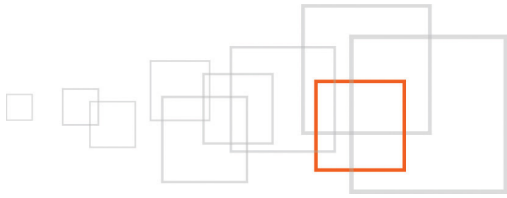
8 Conclusion

This tutorial gives the necessary technical insight on how to use GIT/github to participate to the Localization effort of eZ Publish. If using this tool poses a problem to you, you still have the opportunity to drop the work done to community@ez.no, making sure your valuable Localization contributions are not missed.

Happy localization everyone !

9 Resources

If you have questions when it comes to git, first place to look should probably be Github's FAQ, as it explains things straight forward and also covers Github-related questions. But for a GIT specific article, be sure to check



out one from A list Apart on the subject which also mentions the most important additional GIT resources like Git ready, Pro Git and the official documentation. If you are a cheat-sheet addict, check this out : http://help.github.com/git-cheat-sheets.

Feel free to use the forums (http://share.ez.no/forums/translation), or comments under this tutorial to ask for & share tips & tricks on the usage of GIT/github.

10 About the authors : Nicolas Pastorino & Andre Rømcke

Nicolas Pastorino - http://share.ez.no/community/profile/9804

Andre Rømcke - http://share.ez.no/community/profile/9757

Reviewed by Bertrand Dunogier : http://share.ez.no/community/profile/10106

11 License choice

http://creativecommons.org/licenses/by-sa/3.0

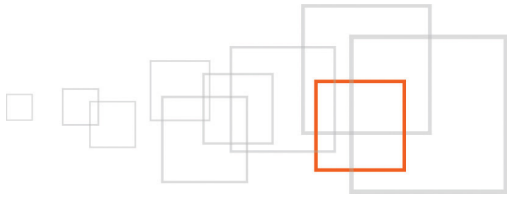
12 Appendix : GIT/Github jargon

Git terms:

- Clone: Clones a whole repository, giving you access to all it's branches and tags locally in one folder (but you'll need to use checkout to select one at a time).
- Commit: Like in svn, but only done locally. Allows you to commit offline and do several small commits to simplify reviews
- Push: Push your local commits to a remote repository
- Pull: Like svn up, pulls in changes from a remote repository
- origin: The origin remote server, this refers to the origin remote GIT server you cloned a repository from, and has nothing to do with forks.
- upstream: And open source term for a third party project your code relies on, in GIT often used for the original project, the one you forked in case of Github. aka ezystems/ezpublish when dealing with eZ Publish.

Github terms:

- Fork: Like a copy of a repository with some knowledge of original repository and compare / status features. To be able to easily share changes you do, as opposed to keeping your changes locally on custom branches or setting up your own GIT server that contains your changes.



- Pull request: A Github feature, makes it possible to notify the original repository committers about your branch and ask him to integrate your work in (make sure you comply with CLA / coding standards before you do)