OS2Web Github Guide

Deployment

After a release is tested and tagged, to deploy the updates:

Goto site.

```
$ cd /var/www/[site_name]/public_html Check status.
```

If the repo is not clean, clean it up, ask for help, do something. Never just update, unless it's your mess, and you know what youre doing.

```
$ git status
```

Get all tags from origin.

```
$ git fetch --tags
```

Take a database backup (in this case it's a drupal site)

```
$ drush sql-dump | gzip > /tmp/[site_name_tld].sql.gz
```

Checkout the new tag.

```
$ git checkout [tag]
```

The codebase is now updated, if features need reverting or something like that, now is the time.

If everything went wrong

Checkout the previous tag.

```
$ git checkout [tag]
```

Restore database

```
$ gunzip -c /tmp/[site_name_tld].sql.gz | drush sql-cli
```

Sometimes its nessecary to drop the database before import.

```
$ drush sql-drop -y # Sometimes its nessecary to drop the database before import.
```

Development / Release

A note about the production site

Production should always run on a tagged release. No feature/hotfix branches in production.

Code should never be changed on the production site.

If all other possibilities are exhausted, and the fix can only be made on the production environment. All changes must me made in a branch seperate from master. This must be approved by […].

Development

All new features, are to be made in seperate (feature) branches on the repository. And then merged into master.

The production sites using the repo, cannot (must not) work off a feature branch. All releases must be tagged, see "Version naming". And only releases (tags) must be checked out on production.

Test (development) sites can use feature branches. The feature branch beeing tested must be up to date with master. So that it represents what will eventually be put in production.

Branch naming

Reserved branches

master = The master branch, this should always be production ready!

Development branches

f/ = feature branch

b/ = bugfix branch

h/ = hotfix branch

example:

f/234_funky-new-menu

h/EOE-393-34234_smoking-hot-import-fix

The ticket/story/issue number for the branch must be the first part of the branch name. Followed by an underscore _ and a short description of what the branch contains. The description should be kept to maximum 6 words

Example

In the example below. A project needs 2 new features.

```
          1.      2.      3. 4. 5.


      7.x-1.3             7.x.1.4
  --*-----*--------------------* master
          |\                /
          | *------*-----*---*   f/1_Awesome-feature
           \          /
            *----*--*--*        f/2_Even-awesomer-feature
```

1. Version 7.x-1.3 is currently in production

Two feature branches are based of master at this point, f/1_Awesome-feature and f/2_Even-awesomer-feature.

2. Development happens in the two branches.

3. The two features are ready for test. To be able to test the 2 at the same time, f/2_Even-awesomer-feature is merged into f/1_Awesome-feature, and this branch is checked out on the test server.

4. Something needed fixing, so this was done in the f/1_Awesome-feature branch.

5. Everything is great, a PR is created to the master branch.

After a review the PR is merged and master is tagged with the next release number.

This tag can now be checked out on production.

Testing pre-release

This is targeted releases of larger products that undergo both alpha, beta and qualification. But the principles can be used for projects that only have one testperiod as well.

Error/bugfixing

Error and bugfixing will begin ASAP after the first issue is reported. All changes will be made in a branch named after the issue(s) beeing adressed.

End of workday:

All fixes ready for deployment must be pushed to master, and a pull-request created.

Start of (next) workday:

All submitted pull-requests will be reviewed and merged by the assignee. After all merges have been approved and merged, the master branch will be tagged with the next release version.

Release

When a feature branch has been tested, and the PR is merged. The repo must be tagged with the next version number. In general the version name, should conform to the naming standards for the major project (Drupal, eZ, Prestashop).