Updating eZ Platform

This page explains how to update eZ Platform to a new version.

In the instructions below, replace `<version>` with the version of eZ Platform you are updating to (for example: `v1.7.0`). If you are testing a release candidate, use the latest rc tag (for example: `v1.7.1-rc1`).

Version-specific steps

> **Some versions introduce new features that require taking special steps; they are marked on this page with yellow tags.**
>
> If you intend to skip a version (for example, update directly from v1.3 to v1.5 without getting v1.4), remember to look at all the intermediate steps as well – this means you need to perform both the steps for v1.4 and v1.5.

Update procedure

1. Check out a version

1.1. From the project's root, create a new branch from the project's master, or from the branch you're updating on:

From your master branch

```
git checkout -b <branch_name>
```

This creates a new project branch for the update based on your current project branch, typically `master`. An example `<branch_name>` would be `update-1.4`.

1.2. If it's not there, add `ezsystems/ezplatform` (or `ezsystems/ezstudio`, when updating an Enterprise installation) as an upstream remote:

From your new update branch

```
git remote add ezplatform
http://github.com/ezsystems/ezplatform.git
or
git remote add ezstudio
http://github.com/ezsystems/ezstudio.git
```

1.3. Then pull the tag into your branch.

> If you are unsure which version to pull, run `git ls-remote --tags` to list all possible tags.

Related:

- Coming to eZ Platform from eZ Publish Platform
- Migration from eZ Publish

In this topic:

From your new update branch

```
git pull ezplatform <version>
or
git pull ezstudio <version>
```

At this stage you may get conflicts, which are a normal part of the procedure and no reason to worry. The most common ones will be on `composer.json` and `composer.lock`.

The latter can be ignored, as it will be regenerated when we execute `composer update` later. The easiest is to checkout the version from the tag and add it to the changes:

> If you get a **lot** of conflicts (on the `doc` folder for instance), and eZ Platform was installed from the share.ez.no tarball, it might be because of incomplete history. You will have to run `git fetch ezplatform --unshallow` (or `git fetch ezstudio --unshallow`) to load the full history, and run the merge again.

From your new update branch

```
git checkout --theirs composer.lock && git add
composer.lock
```

If you do not keep a copy in the branch, you may also run:

From your new update branch

```
git rm composer.lock
```

2. Merge composer.json

Manual merging

Conflicts in `composer.json` need to be fixed manually. If you're not familiar with the diff output, you may checkout the tag's version and inspect the changes. It should be readable for most:

From your new update branch

```
git checkout --theirs composer.json && git diff HEAD
composer.json
```

You should see what was changed, as compared to your own version, in the diff output. The update changes the requirements for all of the `ezsystems/` packages. Those changes should be left untouched. All of the other changes will be removals of what you added for your own project. Use `git checkout -p` to selectively cancel those changes:

```
git checkout -p composer.json
```

Answer no (do not discard) to the requirement changes of `ezsystems` dependencies. Answer `yes` (discard) to removals of your changes.

Once you are done, inspect the file, either using an editor or by running `git diff composer.json`. You may also test the file's sanity with `composer validate`, and test the dependencies by running `composer update --dry-run`. (will output what it would do to dependencies, without applying the changes.

Once finished, run `git add composer.json` and commit.

Fixing other conflicts (if any)

Depending on the local changes you have done, you may get other conflicts on configuration files, kernel, etc.

There shouldn't be many, and you should be able to figure out which value is the right one for all of them:

- Edit the file, and identify the conflicting changes. If a setting you have modified has also been changed by us, you should be able to figure out which value is the right one.
- Run `git add conflicting-file` to add the changes

3. Update the app

At this point, you should have a `composer.json` file with the correct requirements. Run `composer update` to update the dependencies.

```
composer update
```

If you want to first test how the update proceeds without actually updating any packages, you can try the command with the `--dry-run` switch:

```
composer update --dry-run
```

On PHP conflict | 16.02 and later requires PHP 5.5 or higher
Because from release 16.02 onwards eZ Platform is compatible only with PHP 5.5 and higher, the update command above will fail if you use an older PHP version. Please update PHP to proceed.

4. Update database

This step is only relevant for some releases:

Form Builder

1.7.0

To enable the Form Builder feature in eZ Platform Enterprise Edition, import the following file:

```
mysql -p -u <database_user> <database_name> <
vendor/ezsystems/ezstudio-form-builder/bundle/R
esources/install/form_builder.sql
```

Date Based Publisher

1.7.0

To enable the Date-Based Publisher feature in Enterprise, import the following file:

```
mysql -p -u <database_user> <database_name> <
vendor/ezsystems/date-based-publisher/bundle/Re
sources/install/datebasedpublisher_scheduled_ve
rsion.sql
```

In order to activate Date-Based Publisher open console (terminal) and use:

```
crontab -e
```

and add below configuration at the end of edited file.

For production environment:

```
* * * * *   (cd /path/to/your/ezstudio-project
&& app/console ezpublish:cron:run -e=prod)
```

For development environment:

```
* * * * *   (cd /path/to/your/ezstudio-project
&& app/console ezpublish:cron:run -e=dev)
```

5. Dump assets

The web assets must be dumped again if you are using the `prod` environment. In `dev` this happens automatically:

```
php app/console assetic:dump -e=prod
```

If you encounter problems, additionally clear the cache and install assets:

```
php app/console cache:clear -e=prod
php app/console assets:install --symlink -e=prod
php app/console assetic:dump -e=prod
```

6. Commit, test and merge

Once all the conflicts have been resolved, and `composer.lock` updated, the merge can be committed. Note that you may or may not keep `composer.lock`, depending on your version management workflow. If you do not wish to keep it, run `git reset HEAD <file>` to remove it from the changes. Run `git commit`, and adapt the message if necessary. You can now verify the project and once the update has been approved, go back to `master`, and merge your update branch:

```
git checkout master
git merge <branch_name>
```

Your eZ Platform should now be up-to-date with the chosen version!