Upgrading from 5.1 to 5.3

This section describes how to upgrade your existing eZ Publish 5.1 installation to version 5.3. Make sure that you have a working backup of the site before you do the actual upgrade, and make sure that the installation you are performing the upgrade on is offline.

Note on Paths

- */<ezp5-root>/*: The root directory where eZ Publish 5 is installed in, examples: "*/home/myuser/www/*" or "*/var/sites/ezpublish/*"
- */<ezp5-root>/ezpublish_legacy*: Root directory of "Legacy" (aka "Legacy Stack", refers to the eZ Publish 4.x installation which is bundled with eZ Publish 5) normally inside "*ezpublish_legacy/*", example: "*/home/myuser/www/ezpublish_legacy/*"

Check for requirements

PHP 5.3.3 and higher is needed. Further information regarding System Requirements can be found on Requirements Documentation Page.

Upgrade steps

Step 1: Upgrade the distribution files

The easiest way to upgrade the distribution files is to unpack eZ Publish 5.3 to a separate directory and then copy the directories that contain site-specific files from the existing 5.1 installation into "*/<ezp5-root>/*". Make sure you copy the following directories:

- `ezpublish_legacy/design/<your designs>` (do NOT include built-in designs: admin, base, standard or admin2)
- `ezpublish_legacy/var/<your_var_dir>/storage`
- `ezpublish_legacy/var/storage/packages`
- `ezpublish_legacy/settings/siteaccess/<your_siteaccesses>`
- `ezpublish_legacy/settings/override/*`
- `ezpublish_legacy/extension/*` (not including the built-in / standalone ones: ezflow, ezjscore, ezoe, ezodf, ezie, ezmultiupload, ezmbpaex, ez_network, ezprestapiprovider, ezscriptmonitor, ezsi, ezfind)
- src/
- `config.php` & `config.cluster.php`, if applicable
- `ezpublish/config/*`

NB: *Since writable directories and files have been replaced / copied, their permissions might have changed. You may have to reconfigure webserver user permissions on specific folders as explained in the file permissions chapter of the installation process.*

opcode cache
If you use APC, or an op-code cache solution, make sure to clear the cache. The easiest way is to restart your web server.

Step 2: upgrade custom extensions

If you are using custom extensions, the sub-directories inside the "extension" directory will also have to be copied from the existing 5.1 installation into "*/<ezp5-root>/ezpublish_legacy/extension/*". However, make sure that you do not overwrite any extensions that are included in eZ Publish distribution, which currently are (***Note:*** *As of eZ Publish 5.2, these extensions have the same version number as eZ Publish*):

Note that upgrading the distribution files will overwrite the autoload arrays for extensions. You will need to re-generate the autoload arrays for active extensions later.

Important: *If you plan to upgrade your eZ Website Interface, eZ Flow or eZ Demo site package as well, then additional extensions will be updated and the step for re-generate the autoload arrays can be skipped until that is done (links to documentation for upgrading these site packages can be found in the last step of this page).*

Step 3: upgrade the database

For upgrading the database, you have to jump from 5.1 to 5.2 and then from 5.2 to 5.3.

Import to your database the changes provided in :

`/<ezp5-root>/ezpublish_legacy/update/database/<mysql|postgresql>/5.2/dbupdate-5.1.0-to-5.2.0.sql`

`/<ezp5-root>/ezpublish_legacy/update/database/<mysql|postgresql>/5.3/dbupdate-5.2.0-to-5.3.0.sql`

- **Define a Doctrine connection**

MySQL settings : ezpublish.yml or config.yml

```
doctrine:
    dbal:
        connections:
            my_connection:
                driver:   pdo_mysql
                host:     localhost
                port:     3306
                dbname:   my_database
                user:     my_user
                password: my_password
                charset:  UTF8
```

PostGreSQL : ezpublish.yml or config.yml Expand source

```
doctrine:
    dbal:
        connections:
            my_connection:
                driver:   pdo_pgsql
                host:     localhost
                port:     5432
                dbname:   my_database
                user:     my_user
                password: my_password
                charset:  UTF8
```

Pro Tip

Set your base DB params in your `parameters.yml`/`parameters.yml.dist` and refer them here.

parameters.yml

```
parameters:
   database_driver: pdo_mysql
   database_host: localhost
   database_port: 3306
   database_name: ezdemo
   database_user: my_user
   database_password: my_password
   database_charset: UTF8
```

ezpublish.yml / config.yml

```
doctrine:
   dbal:
       connections:
           my_connection:
               driver:   %database_driver%
               host:     %database_host%
               port:     %database_port%
               dbname:   %database_name%
               user:     %database_user%
               password: %database_password%
               charset:  %database_charset%
```

- **Define one or several repositories**

ezpublish.yml

```
ezpublish:
  repositories:
      main: { engine: legacy, connection: my_connection }
```

- **(Optional) Make your SiteAccess config point to the right repository**

ezpublish.yml Expand source

```
ezpublish:
  system:
      my_siteaccess_group:
          repository: main
```

Remove the old connection information
Note : to benefit from the new configuration, don't forget to remove the old configuration

Old database access to remove

```
ezpublish:
    system:
        my_siteaccess_group:
            database:
                type: mysql
                user: my_user
                password: my_password
                server: localhost
                database_name: ezdemo
```

Step 4: Apply 5.2 & 5.3 configuration changes

YAML files

Since default configuration files have been overwritten during step one, the few additions to those files that were made in 5.3 need to be applied manually to the configuration files. All of those changes are **additions**, none of them replaces what you already have. For most of them, at least one, if not all hierarchy elements (monolog, handler, framework, router...) will already be there. All you have to do is add the missing bits in the existing configuration blocks.

In **ezpublish/config/config_dev.yml**, add the configuration for the chromephp log handler:

```
monolog:
    handlers:
        chromephp:
            type: chromephp
            level: info
```

In **ezpublish/config/config.yml**, you need to add a few default values for the framework

```
framework:
    router:
        resource: "%kernel.root_dir%/config/routing.yml"
        strict_requirements: %kernel.debug%
    trusted_proxies: ~
    http_method_override: true

twig:
    debug: %kernel.debug%
    strict_variables: %kernel.debug%
```

Session name

`ezpublish.system.<siteAccessName>.session_name` has been deprecated for defining session name. You now need to use `ezpublish.system.<siteAccessName>.session.name`.

Before:

```
ezpublish:
    system:
        my_siteaccess:
            session_name: SomeSessionName
```

After:

```
ezpublish:
    system:
        my_siteaccess:
            session:
                name: SomeSessionName
```

Routing

In **routing_dev.yml**, add the resource import for the SensioDistributionBundle webconfigurator routes:

```
_configurator:
    resource: "@SensioDistributionBundle/Resources/config/routing/webconfigurator.xml"
    prefix: /_configurator
```

In **routing.yml**, add **new login routes** and the `_ezpublishRestOptionsRoutes` route loader:

```
_ezpublishRestOptionsRoutes:
    resource: "@EzPublishRestBundle/Resources/config/routing.yml"
    prefix: %ezpublish_rest.path_prefix%
    type: rest_options

login:
    path:   /login
    defaults:  { _controller: ezpublish.security.controller:loginAction }
login_check:
    path:   /login_check
logout:
    path:   /logout
```

Security

In ezpublish/config/security.yml, add the following:

```
access_control:
    - { path: ^/login, roles: IS_AUTHENTICATED_ANONYMOUSLY, requires_channel: https }
```

And again in **ezpublish/config/security.yml**, under `ezpublish_front` firewall, update to fit the following (**be sure to remove `ezpublish: true`**):

```
security:
    firewalls:
        ezpublish_front:
            pattern: ^/
            anonymous: ~
            form_login:
                require_previous_session: false
            logout: ~
```

If you have added anything to `parameters.yml`, we suggest that you add your custom settings to `parameters.yml.dist`, so that the composer post-update script handles those, and generates their values correctly.

Templates

In your templates, change your links pointing to `/user/login` and `/user/logout` to appropriate `login` / `login_check` / `logout` routes:

Before

```
<a href="{{ path( 'ez_legacy', {'module_uri': '/user/login'} ) }}">Login</a>
<form action="{{ path( 'ez_legacy', {'module_uri': '/user/login'} ) }}" method="post">
<a href="{{ path( 'ez_legacy', {'module_uri': '/user/logout'} ) }}">Logout</a>
```

After

```
<a href="{{ path( 'login' ) }}">Login</a>
<form action="{{ path( 'login_check' ) }}" method="post">
<a href="{{ path( 'logout' ) }}">Logout</a>
```

ezpublish/EzPublishKernel.php

It is not possible to just copy your old `EzPublishKernel.php` file over from the previous installation, since quite a few changes were made to this file in this release. We suggest that you simply reflect in the new kernel file any changes you made in the previous version.

composer.json

If you had modified `composer.json` to add your own requirements, you must re-apply those changes to the new version, and run **`composer update`**.

Varnish (if applicable)

Anonymous state of a user is not checked through presence of `is_logged_in` cookie any more. Therefore, when using Varnish, you must change the following in your VCL file:

Before

```
# ez_user_hash sub-routine
if (req.http.Cookie !~ "is_logged_in=" ) {
    # User don't have "is_logged_in" cookie => Set a hardcoded anonymous hash
    set req.http.X-User-Hash = "38015b703d82206ebc01d17a39c727e5";
}
```

After

```
# ez_user_hash sub-routine
if (req.http.Cookie !~ "eZSESSID" ) {
    # User don't have session cookie => Set a hardcoded anonymous hash
    set req.http.X-User-Hash = "38015b703d82206ebc01d17a39c727e5";
}
```

Step 5: Update cluster data (if applicable)

If your installation uses the DFS cluster, you are affected by the split DFS tables feature that was added in 5.2. You can also check the DFS setup documentation document on steps 3 and 4 for additional details and usage examples about the newly introduced configurations.

To use the feature, you need to update your DFS database structure. It won't affect existing data, and doesn't require particular measures. Import the following file into your*cluster* database (it should be different from your eZ Publish database):

`/<ezp5-root>/ezpublish_legacy/update/database/mysql/5.2/dbupdate-cluster-5.1.0-to-5.2.0.sql`

The split table feature stores cache and storage into two different tables. For now, your ezdfsfile table contains both cache and storage. Starting from now, eZ Publish will use the newly created table, ezdfsfile_cache, to store cache. Since the table is empty, it will react as if there was no cache, and work without any changes.

However, we recommend that you remove entries related to cache from your ezdfsfile table. There are several options.

Option 1: unclusterize, clear data, and reclusterize

This method requires that you shutdown the website completely for a little while. It mainly applies to small/medium websites (up to a couple thousand content objects).

May require a lot of disk space
Since this method will create a local copy of every storage file (not including cache) from your NFS to the local server, this method may require a large amount of disk space. You can get an estimate of the required space by running the following query on your cluster database:

```
SELECT SUM(size) from ezdfsfile WHERE name LIKE 'var/ezdemo\_site/storage/%';
```

It will give you the total size of storage items, in bytes. Remember to escape _ in your vardir.

Use the following commands from your */<ezp5-root>/ezpublish_legacy/* folder:

```
cd ezpublish_legacy
bin/php/clusterize.php -u
```

Next, truncate the ezdfsfile table from your database:

```
TRUNCATE TABLE ezdfsfile
```

And finally, re-create cluster data based on local data:

```
cd ezpublish_legacy
bin/php/clusterize.php
```

Option 2: perform a background cleanup of the storage table

An upgrade script is provided that will let you cleanup the ezdfsfile table on a live website, even with a large cluster. It will delete a configurable (by default 1000) batch of cache rows from ezdfsfile, and sleep (by default for 100 ms) between batches. Run the following from the legacy root:

```
cd ezpublish_legacy
php update/common/scripts/5.2/cleanupdfscache.php -h
```

The script can be interrupted and restarted anytime without risks for the system. We strongly suggest, if you execute it on a live website, that you monitor your database server's performances, and increase the sleep delay and/or decrease the limit if the SQL server's load is too high.

Step 6: Regenerate the autoload array for extensions

To regenerate the autoload array, execute the following script from the root of your eZ Publish Legacy directory:

```
cd ezpublish_legacy
php bin/php/ezpgenerateautoloads.php --extension
```

Step 7: Link assets

Assets from the various bundles need to be made available for the webserver through the web/ document root.

The following commands will first symlink eZ Publish 5 assets in "Bundles" and the second will symlink assets (design files like images, scripts and css, and files in var folder) from eZ Publish Legacy:

```
php ezpublish/console assets:install --symlink
php ezpublish/console ezpublish:legacy:assets_install --symlink
php ezpublish/console assetic:dump --env=prod
```

Step 8: Update rewrite rules

There are two ways eZ Publish 5 can be installed, either the full install with both the new Symfony stack and the legacy stack, or legacy only. In latter case you only need to point your '4.7 like' rewrite rules to */<ezp5-root>/ezpublish_legacy/* and that's it. Otherwise, update your virtual host according to the eZ Publish 5.2 rewrite rules on confluence and point your host configuration to */<ezp5-root>/web/*.

Step 9: Clear the caches

Whenever an eZ Publish solution is upgraded, all caches must be cleared in a proper way. This should be done from within a system shell: Navigate into the new eZ Publish directory.Run the script using the following shell command:cd /<ezp5-root>/ezpublish_legacy/php bin/php/ezcache.php --clear-all --purgePurging ensures that the caches are physically removed. When the "--purge" parameter is not specified, the caches will be expired but not removed.
Note: Sometimes the script is unable to clear all cache files because of restrictive file/directory permission settings. Make sure that all cache files have been cleared by inspecting the contents of the various cache sub-directories within the "var" directory (typically the "var/cache/" and "var/<name_of_siteaccess>/cache/" directories). If there are any cache files left, you need to remove them manually.

Step 10: Upgrade Extensions (site package)

Next, depending on if you originally installed eZ Flow, eZ Webin or eZ Demo site, follow the steps mentioned in the eZ Webin, eZ Flow or eZ Demo upgrade documentation.