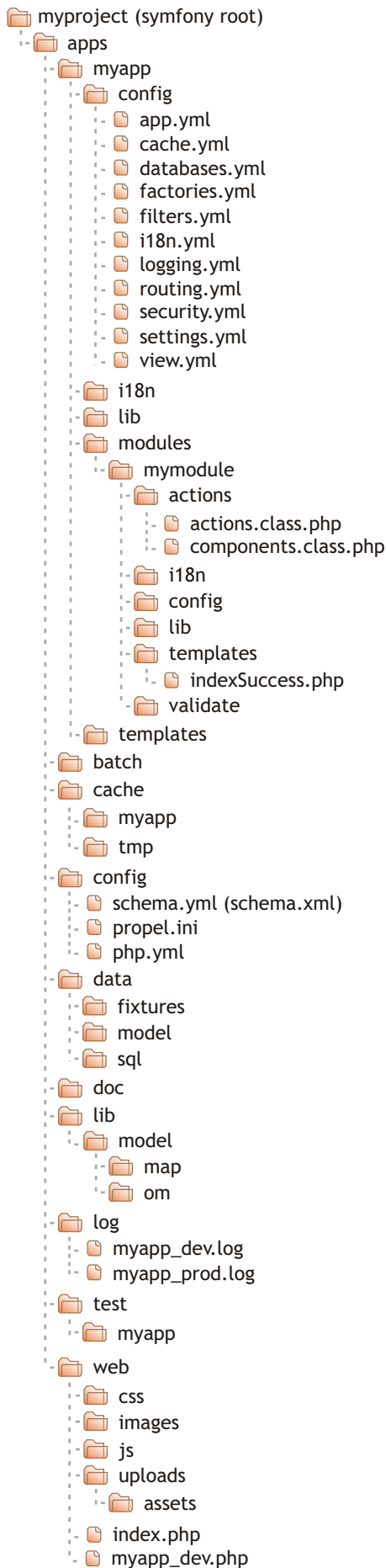


# Directory Structure and CLI

## DEFAULT DIRECTORY STRUCTURE



## COMMAND LINE INTERFACE (CLI)

```

$ symfony -T
Full list of the available admin operations

$ symfony -V
Installed version of the symfony package

$ symfony clear-cache <application_name> [template|config]
Clear the cached information (shortcut: cc)

$ symfony init-project <project_name>
Initialize the project and generate the basic files and directories necessary for runtime

$ symfony init-app <application_name>
To initialize one application. Some php files corresponding to the front controllers of each
default environment are also created in the project root web/ directory:
index.php (prod) and myapp_dev.php (dev)

$ symfony init-module <application_name> <module_name>
To initialize one module. After command, the new module is ready to be used:
http://myapp.example.com/index.php/mymodule

$ symfony propel-build-schema [xml]
Generate the schema.yml for representation of an existing database.
For schema.xml use option xml

$ symfony propel-build-model
Generate the PHP classes for the model, according with the data model described in the
schema.yml. The base data access classes will be automatically created in the
myproject/lib/model/om/ directory:
BaseArticle.php      BaseComment.php
BaseArticlePeer.php  BaseCommentPeer.php

In addition, the actual data access classes will be created in myproject/lib/model:
Article.php          Comment.php
ArticlePeer.php      CommentPeer.php

$ symfony propel-generate-crud <application_name> <module_name> <ClassName>
Scaffolding - Generate a new Propel CRUD module based on a class from the model

$ symfony propel-build-sql
Create the SQL code to create the tables described in the schema.yml, in a
myproject/data/sql/lib.model.schema.sql file

$ symfony propel-build-db
Create an empty database

$ symfony propel-insert-sql
Insert the SQL code from myproject/data/sql/lib.model.schema.sql file into the database

$ symfony sync <environment_name> [go]
Synchronise the current project with another machine

$ symfony propel-init-admin <application_name> <module_name> <ClassName>
Initialize a new Propel admin module based on a class from the model

$ symfony test <application_name>
Launch the test suite for an application

$ symfony plugin-install [local|global] <channel_name>/<plugin_name>
Install a new plugin

$ symfony freeze
Convert the project to an independent, stand-alone application

$ symfony unfreeze
To revert a project to its initial state. It erases the data/symfony/, lib/symfony/ and web/sf/

$ symfony disable <application_name> <environment_name>
Temporarily disable your application, when you need to upgrade a library or a large amount of
data

$ symfony enable <application_name> <environment_name>
Reenables the application and clears the cache

$ symfony clear-controllers
Clears the web/ directory of all controllers other than the ones running in a production
environment. If you do not include the development front controllers in the rsync_exclude.txt
file, this command guarantees that a backdoor will not reveal the internals of your application

$ symfony fix-perms
Fixes directory permissions, to change the log/ and cache/ permissions to 0777
(these directories need to be writable for the framework to work correctly)

$ symfony log-purge
Erases the symfony log files in applications and environments where the logging.yml file
specifies purge: on (which is the default value)
    
```