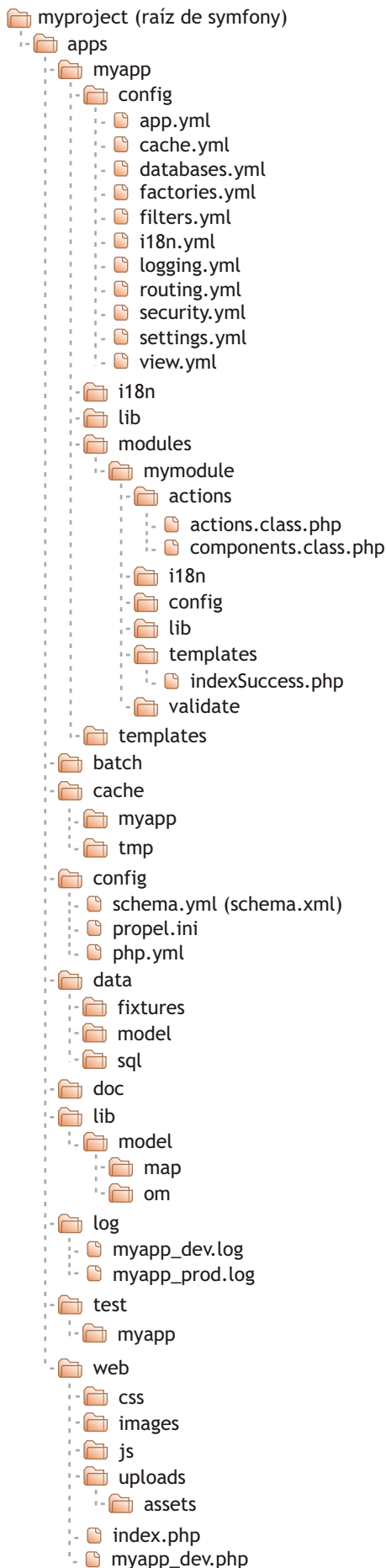


Estructura de Directorios y CLI

ESTRUCTURA DE DIRECTORIOS POR DEFECTO



INTERFAZ DE LÍNEA DE COMANDOS (CLI)

```

$ symfony -T
Muestra la lista completa de las operaciones de administración disponibles

$ symfony -V
Muestra la versión de symfony instalada

$ symfony clear-cache <application_name> [template|config]
Borra la información de la cache (atajo: cc)

$ symfony init-project <project_name>
Inicializa un proyecto y crea los archivos y directorios básicos necesarios para su ejecución

$ symfony init-app <application_name>
Inicializa una aplicación. En el directorio web raíz del proyecto, se crean algunos archivos correspondientes a los controladores frontales de cada uno de los entornos por defecto:
index.php (prod) y myapp_dev.php (dev)

$ symfony init-module <application_name> <module_name>
Inicializa un módulo. Después de ejecutar el comando, ya se puede utilizar el nuevo módulo:
http://myapp.example.com/index.php/mymodule

$ symfony propel-build-schema [xml]
Genera el archivo schema.yml para la representación de una base de datos existente.
Para generar el archivo schema.xml, se utiliza la opción xml

$ symfony propel-build-model
Genera las clases PHP del modelo, según el modelo de datos descrito en el archivo
schema.yml. Las clases base del modelo se crean en el directorio myproject/lib/model/om/:
BaseArticle.php      BaseComment.php
BaseArticlePeer.php  BaseCommentPeer.php
También se crean las clases de acceso a los datos en el directorio myproject/lib/model:
Article.php           Comment.php
ArticlePeer.php       CommentPeer.php

$ symfony propel-generate-crud <application_name> <module_name> <ClassName>
Scaffolding - Genera un nuevo módulo con las funciones CRUD, basado en una clase del modelo

$ symfony propel-build-sql
Genera el código SQL necesario para crear las tablas descritas en el archivo schema.yml.
El archivo generado se guarda en myproject/data/sql/lib.model.schema.sql

$ symfony propel-build-db
Crea una base de datos vacía

$ symfony propel-insert-sql
Inserta el código SQL del archivo myproject/data/sql/lib.model.schema.sql en la base de datos

$ symfony sync <environment_name> [go]
Sincroniza el proyecto actual con otro servidor

$ symfony propel-init-admin <application_name> <module_name> <ClassName>
Inicializa un nuevo módulo de administración, basado en una clase del modelo

$ symfony test <application_name>
Inicia la ejecución de un conjunto de pruebas de una aplicación

$ symfony plugin-install [local|global] <channel_name>/<plugin_name>
Instala un nuevo plugin

$ symfony freeze
Convierte el proyecto en una aplicación independiente (stand-alone)

$ symfony unfreeze
Revierte un proyecto a su estado inicial. Borra los directorios data/symfony/, lib/symfony/ y web/sf/

$ symfony disable <application_name> <environment_name>
Deshabilita temporalmente una aplicación. Se utiliza cuando se actualiza una librería o una gran cantidad de datos

$ symfony enable <application_name> <environment_name>
Activa una aplicación y borra su cache

$ symfony clear-controllers
Borra del directorio web/ todos los controladores que no pretenezcan al entorno de producción.
Si no se han incluido los controladores frontales de desarrollo en el archivo rsync_exclude.txt,
este comando asegura que no exista una puerta trasera que revele información interna de la
aplicación.

$ symfony fix-perms
Establece los permisos de los directorios log/ y cache/ a un valor de 0777
(el framework debe poder escribir en estos directorios para funcionar correctamente)

$ symfony log-purge
Borra los archivos de log de symfony en las aplicaciones y entornos para los que el archivo
logging.yml especifica un valor purge:on (que es el valor por defecto)
    
```