Framework para el desarrollo ágil de aplicaciones

Índice

Introducción

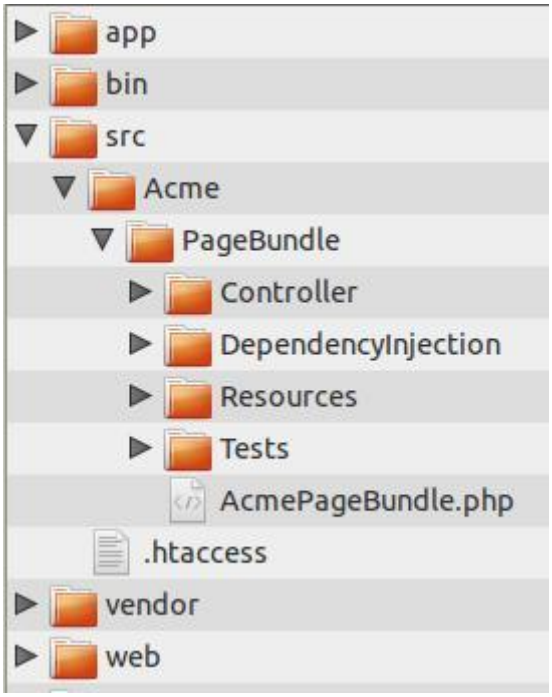
¿Sabes qué es un framework? ¿Consideras que su uso es beneficioso para el desarrollador? ¿Qué frameworks existen? Éstas y otras cuestiones son las que abordaremos a lo largo de este White Paper.

En un primer paso nos centraremos en explicar qué son los frameworks mostrando sus principales ventajas y características, para posteriormente hablar sobre dos de los Framework más utilizados a la hora de crear aplicaciones web, como son Symfony 2 para aplicaciones creadas en PHP y Struts 2 para proyectos donde el lenguaje utilizado es Java.

¿Qué es un framework?

En contra de lo que muchos pudierais pensar, un framework no es ningún software ni herramienta que se ejecuta y que nos ofrece una interfaz gráfica desde la que trabajar, sino que es un conjunto de archivos y directorios que facilitan la creación de aplicaciones, ya que incorporan funcionalidades ya desarrolladas y probadas, implementadas en un determinado lenguaje de programación.

En la siguiente imagen podéis ver cómo sería la estructura de directorios correspondiente a un proyecto desarrollado con Symfony 2.



El objetivo principal de todo framework es facilitar las cosas a la hora de desarrollar una aplicación, haciendo que nos centremos en el verdadero problema y nos olvidemos de implementar funcionalidades que son de uso común como puede ser el registro de un usuario, establecer conexión con la base de datos, manejo de sesiones de usuario o el almacenamiento en base de datos de contenido cacheado.

Ventajas de utilizar un Framework

El uso de un framework a la hora de realizar un proyecto, ofrece importantes ventajas, ventajas ya no sólo al facilitarnos la tarea de la creación de la aplicación, sino otras como en el mantenimiento del código, realizar ampliaciones, etc.

a) Uso de patrones de diseño

Uno de las principales ventajas que ofrecen los framework es el uso de patrones de diseño para el desarrollo de la aplicación. El patrón más utilizado y que casi todos los framework utilizan es el conocido como Modelo – Vista – Controlador (MVC), un modelo que divide el desarrollo en tres capas:

- **Modelo**: Representa los datos de la aplicación y sus reglas de negocio.
- **Vista**: Representa la capa presentación, como representamos los datos a los usuarios.
- **Controlador**: Es el encargado de procesar las peticiones de los usuarios y controla el flujo de ejecución del sistema.

El modelo MVC puede ser implementado sin la necesidad de utilizar un framework, pero la diferencia radica en que el framework nos obliga a utilizarlo, creando de esta forma un código mucho más robusto. Además el uso de este tipo de utilidades nos ayuda a evitar el conocido como "código spaghetti", que consiste en meter funcionalidades en capas que no corresponde, lo que con el paso de tiempo hará que nuestro código sea un verdadero caos, hasta para nosotros mismos.

b) Estructura predefinida de la aplicación

El programador no necesita plantearse la estructura global de la aplicación, ya que esta es proporcionada por el propio framework. Esto tiene la ventaja de que pasado un tiempo, si tenemos que tocar algo en la aplicación, sabremos donde encontrar el archivo en cuestión de forma rápida.

c) Código altamente testeado

Todo el código que forma parte del framework está altamente probado, lo que garantiza el buen funcionamiento del mismo. Nosotros podríamos desarrollar esas mismas funcionalidades, pero nunca podremos garantizar ese nivel de testeo que ofrecen los frameworks.

d) Comunidad de usuarios detrás de cada framework

La gran mayoría de los frameworks tienen detrás a una amplia comunidad de usuarios, de los cuales muchos ayudan en su desarrollo o creando extensiones con funcionalidades extra que podremos utilizar de forma sencilla sin tener que desarrollarlas por nuestra cuenta.

e) Trabajo en equipo

El uso de frameworks facilita el trabajo en equipo, ya que si todos conocen el framework utilizado, conocerán la estructura de directorios y sabrán dónde tienen que ir para realizar una determinada acción.

Desventajas de utilizar un Framework

El uso de los framework también tiene sus limitaciones, por tanto es bueno conocerlas antes de embarcarse en un proyecto. Las desventajas principales son:

a) Tiempo de aprendizaje

Te llevará algún tiempo conocer cómo funcionan los frameworks y, pero la idea es que una vez que te habitúes a ellos los siguientes proyectos tendrán una base más definida y su puesta en marcha será más rápida.

b) Exceso de líneas de código

Muchos autores y expertos en el desarrollo de aplicaciones apuntan que los framework utilizan muchas más líneas de código para realizar ciertas acciones ya que suelen incluir "código basura" que nos podríamos haber ahorrado, aunque esto no es compartido por todo el mundo.

c) Limitaciones

Cuando usas un framework hay partes de él que no puedes modificar, por ello hay que elegir uno que se adapte a lo que buscas. Además, migrar a otro framework es complicado sin tener que reescribir todo el código.

d) Código público

Al estar disponible el framework para todo el mundo, un hacker puede estudiar el código y encontrar debilidades, aunque es algo difícil, porque tendría que saber qué framework estás usando. Además, algunos como Symfony son testados continuamente para reducir al máximo los bugs.

Características de los frameworks

A continuación os dejamos algunas características que suelen incluir todos los frameworks existentes.

- **Abstracción de URLs y sesiones**. No es necesario manejar directamente las URLs ni las sesiones, ya que el framework se encarga de hacerlo.
- **Acceso a datos**. Incluyen herramientas e interfaces necesarias para comunicarse con bases de datos, independientemente del tipo que estemos utilizando.
- **Uso de controladores**. Suelen implementar una serie de controladores para la gestión de los eventos y peticiones realizadas a la aplicación.

- **Autentificación y control de acceso**. Incluyen mecanismos para la identificación de usuarios mediante el uso de login y password.
- **Internalización**. Son mecanismos para poder mostrar la aplicación en todos aquellos idiomas que consideremos oportunos.

¿Cómo elegir el framework a utilizar?

Si realizamos una búsqueda en internet nos aparecerá un gran número de frameworks para utilizar, pero, ¿por cuál decantarnos? Grosso modo deberemos tener en cuenta el tipo de aplicación que vamos a desarrollar, así como el lenguaje de programación que utilizaremos para ello. Otras cosas que nos tenemos que plantear son:

- **Conocimientos del equipo**. Es de suma importancia evaluar el lenguaje con el que se siente cómodo el equipo encargado de desarrollar la aplicación. Si los encargados de su desarrollo controlan PHP, lo más aconsejable es decantarse por uno que haga uso de este tipo de lenguaje.
- **Soporte.** Un punto a tener en cuenta para la elección de un framework es ver la comunidad que hay detrás de él. Es recomendable investigar un poco para saber si están trabajando en nuevas versiones o bien se trata de un proyecto abandonado.
- **Existencia de proyectos desarrollados con él.** No caigamos en la tentación de actuar como conejillos de indias y decantarnos por aquel framework que sepamos que ha sido utilizado para la creación de algún proyecto de éxito.
- **Curva de aprendizaje.** Como hemos explicado, este punto es el que más echa para atrás a las personas a la hora de utilizar un framework, de ahí que sea muy recomendable decantarse por aquel que tenga una curva de aprendizaje más rápida.
- **Soporte para el MVC.** El uso de patrones de diseño es básico en una aplicación bien estructurada, pero el uso del patrón MVC es imprescindible para la mayoría de proyectos. Como nota, algunos frameworks ofrecen MVC como una alternativa, no obligando a su uso. Esto puede ser útil, por ejemplo si quieres implementar tareas internas automatizadas, como emailings, tareas de mantenimiento de la BBDD, etc.
- **Framework que esté desarrollado tomando la seguridad como punto de partida.** En el mercado nos podemos encontrar framework con buenas características pero que dejan de lado la seguridad.

Ejemplos de frameworks que nos podemos encontrar

A continuación os dejamos un listado de frameworks que nos podemos encontrar.

- **Ruby on Rails**. Framework MVC basado en Ruby orientado al desarrollo de aplicaciones web
- **CodeIgniter**. Framework basado en PHP liviano y rápido
- **Djando**. Un framework para Python capaz de crear diseños muy limpios
- **Zend** Framework. Es un framework de código abierto en PHP para desarrollar aplicaciones web y servicios web con PHP 5.
- **Symfony**. Completo framework en PHP diseñado para optimizar el desarrollo de las aplicaciones web basado en el patrón Modelo Vista Controlador
- **Yii**. Framework en PHP basado en componentes
- **Struts.** Herramienta de soporte para el desarrollo de aplicaciones Web bajo el patrón MVC bajo la plataforma Java EE (Java Enterprise Edition)
- **ASP .NET.** Es un framework para aplicaciones web desarrollado y comercializado por Microsoft

Framework Symfony 2 para desarrollar aplicaciones en PHP

Symfony2 es un framework rápido, flexible y fácil de aprender cuya primera versión fue creada en el año 2005 por Fabien Potencier, y en julio del 2011 veía la luz Symfony2, del cual podemos destacar estas características:

- Ha sido creado teniendo en cuenta el rendimiento como una de las principales prioridades, lo que lo ha convertido en uno de los frameworks más rápidos y que menos recursos consume.
- Fácil de utilizar ya que cuenta con una API de desarrollo muy intuitiva.
- Permite la creación de aplicaciones extensibles, es decir, las aplicaciones son creadas por medio de módulos (Bundles) que pueden ser reutilizados en cualquier proyecto.
- Facilita la tarea a los desarrolladores, proporcionando un entorno de desarrollo con una barra de herramientas donde ver en todo momento información interesante para el usuario, como la memoria que se consume, acceso a log de errores detallados, tiempo de ejecución de la aplicación…
- Es Open Source, por lo que somos libres de hacer lo que queramos con él.
- Potente línea de comandos que permite realizar gran parte del trabajo de forma automática, como es la creación del proyecto, implementación de las entidades o el borrado de la caché.

Una vez que hemos instalado Symfony2, obtendremos una estructura de directorios similar a la que podéis ver en la siguiente imagen.

Symfony2 cuenta con unas secciones bien diferenciadas que son las encargadas de aportar todo el potencial a la hora de la creación de aplicaciones. Estas secciones en las que se divide son:

1.- El núcleo
Se trata de la pieza principal del framework y es la encargada de inicializar la configuración de la aplicación y arrancar los bundles que forman parte de ella.

2. Bundles
Un Bundle se puede definir como un módulo encargado de realizar una tarea específica y que puede ser utilizado en cualquier proyecto.

3.- Contenedor de inyección de dependencias
Uno de los puntos fuertes de Symfony2 e inspirado en el framework Spring de Java. Este contenedor de dependencias facilita la tarea al programador, ya que no se debe preocupar de la creación de las dependencias entre objetos, sino que todo lo realiza el framework de forma totalmente transparente.

4.- Manejador de peticiones
Se trata de una de las partes más importantes del framework. Se encarga de encapsular la petición del usuario y devolverle una respuesta. Es un manejador algo especial, ya que notifica eventos a la espera de que un escuchador se haga cargo de ellos y devuelva la información solicitada.

5.- Manejador de eventos
Es el encargado de gestionar todos los eventos que se producen durante la ejecución de la aplicación.
Otra de la característica principal que ofrece Symfony2 son los distintos entornos de trabajo que nos ofrece a la hora de trabajar con él. Los entornos que trae por defecto son los siguientes:

- **Entorno de desarrollo**. Es el entorno utilizado por los desarrolladores mientras se trabaja en la construcción de la aplicación.
- **Entorno de prueba**. Este entorno es utilizado para testear la aplicación de forma automática por medio de las pruebas unitarias.
- **Entorno de muestra**. Se trata del entorno que utilizará el cliente para testear la aplicación en busca de posibles errores.
- **Entorno de producción**. Es el entorno que se activa cuando la aplicación cumple con todos los requisitos y se ha quedado limpia de errores.

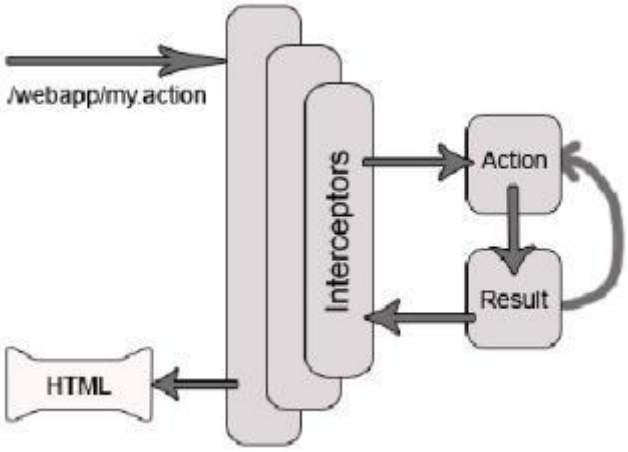
Además de los entornos que ofrece Symfony2 por defecto, el programador puede crearse sus propios entornos de trabajo.

Framework Struts 2 para desarrollar aplicaciones en Java

Struts 2 es un famoso framework para desarrollar aplicaciones en Java creado por Apache y que está basado en el patrón MVC. Está pensado para facilitar las tareas propias del ciclo de vida del software, incluyendo la construcción, desarrollo y mantenimiento de la aplicación. Entre las principales características que ofrece Struts 2 están:

- **Diseño simplificado**. Mientras que en Struts 1 se hacía uso de las clases abstractas, en esta segunda versión se deja de lado esto y se hace uso de Interfaces, que son más sencillas de adaptar al desarrollo y facilita la extensión del mismo. También busca que las clases sean lo más sencillas posible, apareciendo así el concepto POJOs, que no son más que clases que contienen métodos setter y getter para poder recibir valores desde las páginas.
- **Simplificación de los Actions**. En esta versión, cualquier clase de Java que contenga un método execute puede actuar como un Action.
- **Simplificación de los tests**. Como la lógica de negocio está implementada en los Actions, esto facilita la realización de tests unitarios.
- **Uso de anotaciones**. Struts permite realizar la configuración por medio de anotaciones dentro de las clases creadas, en vez de tener que utilizar para ello ficheros XML.
- **Fácil integración con Spring**. Struts puede convivir de forma sencilla con el framework Spring y aprovecharse de la inyección de dependencias que ofrece este último framework.

De forma resumida, el funcionamiento de Struts 2 ante una petición es la siguiente:



1. La aplicación detecta que una petición de información.
2. La petición es interceptada por el FilterDispatcher, que es el encargado de determinar qué Action se ejecuta.
3. Si el Action encargado de procesar la petición utiliza la anotación @Before, ejecutará ese método antes de su código.
4. Si el Action encargado de procesar la petición hace uso de la anotación @After, una vez que ejecuta su código, invocará al método que hace referencia la anotación.
5. Se examina el resultado obtenido del Action y se determina el Result correspondiente.
6. Mediante el Result determinado, se genera la vista que es retornada al cliente.