Título del libro

Jorge Bautista

13 de enero de 2014

Prólogo

Este es un libro que menciona de manera breve las características más importantes del framework Grails[1]. No se detiene en detalles ni explica ningún procedimiento de instalación. Simplemente permite que el lector eche un vistazo a la tecnología para comenzar a conocerla o para *ojearla* y buscar *puntas de hilo* a soluciones cuyo resto encontrará en google.

[1] Únicamente las características descritas en el libro *"The Definitive Guide To Grails"*

Capítulo 1

Arquitectura de grails

Grails envuelve a muchas tecnologías relacionados con *java*, abstrayendo su complejidad al proporcionarnos configuraciones predeterminadas que cubren los casos de uso más comunes. Estas configuraciones pueden ahorrarnos días o semanas de trabajo debido a la integración con otras tecnologías.

Aquí se presentan las tecnologías principales sobre las que se Grails está montado:

- Hibernate
- Spring
- Sitemesh
- Tomcat
- H2
- Groovy
- Gant
- JEE

Algunos objetos implícitos e importantes en grails

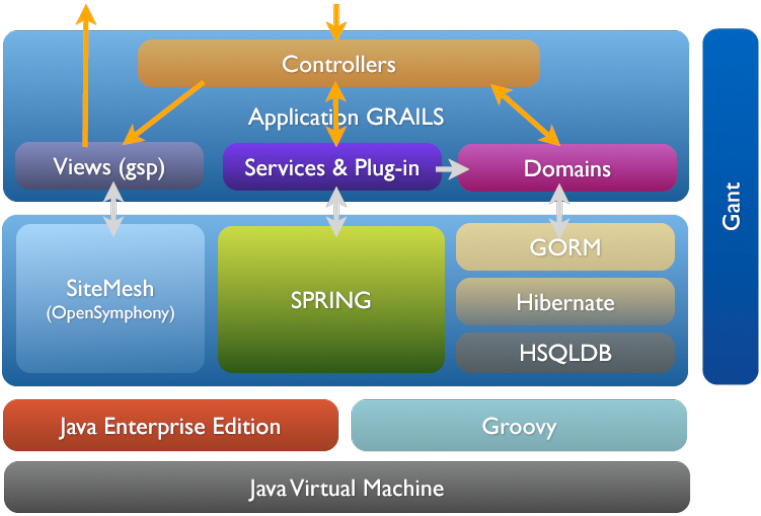

Figura 1.1: Tecnologías que confirman la arquitectura de Grails

Capítulo 2

Inicio rápido

1. Instala Grails y ponlo en el *Path* de tu sistema operativo
2. Verifica que Grails esté en el *Path* ejecutando el siguiente comando desde una línea de comandos y asegurándote de que devuelva la versión instalada de Grails:

   ```
   grails -version
   ```

3. Navega desde una línea de comandos hacia el folder dónde generarás tu aplicación
4. Para autogenerar una aplicación, ejecuta desde una línea de comandos:

   ```
   grails create-app NOMBRE_DE_APP
   ```

5. Para correr tu aplicación en el servidor web integrado, ejecuta desde una línea de comandos y dentro del folder autogenerado por Grails:

   ```
   grails run-app
   ```

6. Accede a la dirección:

   ```
   http://localhost:8080/NOMBRE_DE_APP
   ```

Capítulo 3

Métodos de renderizado

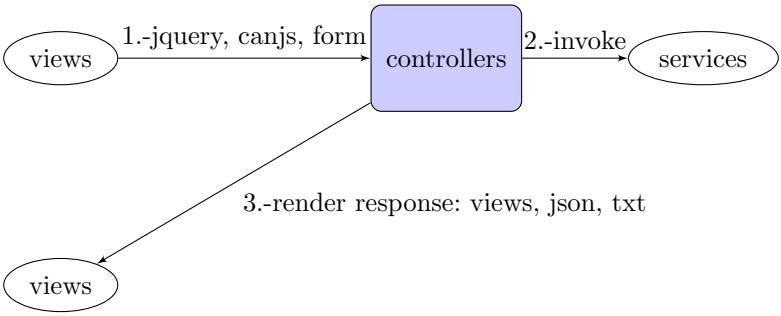

Figura 3.1: Flujo de datos básico de Grails

El resultado enviado de los controllers a las vistas puede ir en varios formatos. Siempre se hace a través de la instrucciíon *render*:

- **Renderizar texto**

```
render 'Welcome to the gTunes store!'
```

- **Solicitar ayuda de un comando**

```
grails help NOMBRE_DE_COMANDO
```

- **Crear aplicación de Grails**

```
grails create-app NOMBRE_DE_APP
```

- **Crear un controller + sus pruebas + sus vistas**

  ```
  grails create-controller [PAQUETE]NOMBRE
  ```

- **Ejecutar pruebas de la aplicación**

  ```
  grails test-app
  ```

- **Ejecutar la aplicación**

  ```
  grails run-app
  ```

Capítulo 4

Pruebas

4.1. Tipos de pruebas en Grails

Grails divide las pruebas en 2 tipos:

Unitarias La ejecución de estas pruebas es rápida, comparada con las de integracoón. La desventaja es que requieren de un uso extensivo de *mocks* y *stubs* para imitar el comportamiento de algunas clases y regresar valores predeterminados para la prueba.

Integración Estas pruebas necesitan que la aplicación esté cargada por completo para poder realizarse, incluyendo la base de datos, por lo que son más lentas.

4.2. Objetos implícitos en clases de prueba

Dentro de las clases de prueba existen varios objetos con los que puedes contar para apoyarte durante su implantación:

controller Es una referencia al controller definido en la anotación *@TestFor*, lo que permite el acceso a los métodos del controller.

params Es una referencia a los parámetros enviados al controller para usarlos en su lógica. Al llenar este objeto los *params* serán enviados al objeto *controller* cuando se invoque alguno de sus métodos.

response Una vez que alguno de los métodos del *controller* sea invocado, el objeto *response* se llenará con los detalles de la respuesta (estatus, texto de respuesta, headers, etc).

Algunas muestras de código para realizar pruebas en Grails:

- **Clase para probar un controller. Autogenerada junto con el controlador *StoreController***

```
@TestFor(StoreController)
class StoreControllerTests {
    void testSomething() {
        fail "Implement me"
    }
}
```

- **Clase para probar el texto de respuesta de un controller al ejecutar su método** index()**:**

```
@TestFor(StoreController)
class StoreControllerTests {
    void testSomething() {
        controller.index()
        assert 'Welcome to the gTunes store!'
            == response.text
    }
}
```

4.3. Comandos útiles en el desarrollo de pruebas

Los resultados de la ejecución de algún comando de pruebas generan archivos[1] con los resultados en los formatos *html, txt* y *xml*.

- **Comando que ejecuta todas las pruebas del sistema**

```
grails test-app
```

- **Clase para probar el texto de respuesta de un controller al ejecutar su método** index()**:**

[1] Los resultados se generan en el folder test/reports

```
@TestFor(StoreController)
class StoreControllerTests {
    void testSomething() {
        controller.index()
        assert 'Welcome to the gTunes store!'
            == response.text
    }
}
```

Apéndice A

Comandos más comunes

- **Consultar versión de Grails**

  ```
  grails -version
  ```

- **Solicitar ayuda de un comando**

  ```
  grails help NOMBRE_DE_COMANDO
  ```

- **Crear aplicación de Grails**

  ```
  grails create-app NOMBRE_DE_APP
  ```

- **Crear un controller + sus pruebas + sus vistas**

  ```
  grails create-controller [PAQUETE]NOMBRE
  ```

- **Ejecutar pruebas de la aplicación**

  ```
  grails test-app
  ```

- **Ejecutar la aplicación**

  ```
  grails run-app
  ```

Índice de figuras

Índice de cuadros

Bibliografía

[1] article-description1

[2] article-description2