

DESPLIEGUE APLICACIÓN SPRING

Realizado por Ignacio Franco, Andrés Molina, Adán
Bouahouita y Ana Blasco

ENTREGA 3 – GRUPO
3. ACCESO A DATOS.
2º DAM, C.P SAN
IGNACIO.

Contenidos

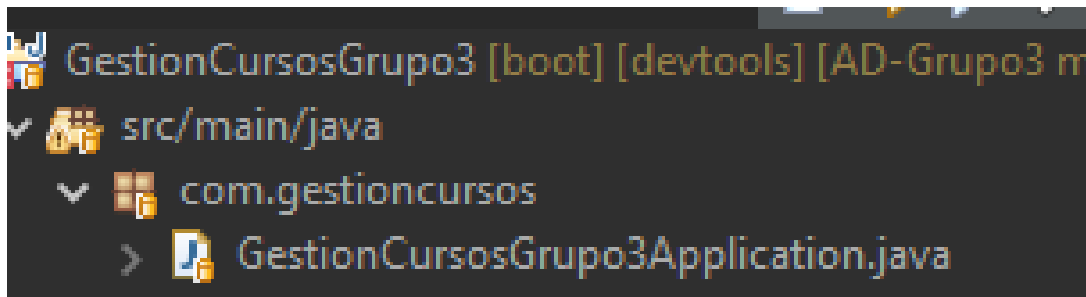
1. Ajustando el proyecto	2
2. Inicializando Tomcat	4
3. Exportando los archivos y desplegando el servidor	6
Webgrafía:	8

Documentación despliegue aplicación Spring

En nuestro trabajo el despliegue de la aplicación se llevará a cabo con **Apache Tomcat y Maven** y para conseguirlo se tendrán que seguir los pasos descritos a continuación.

1. Ajustando el proyecto

Primero tendremos que modificar un poco nuestro proyecto para ser lanzado en un servidor standalone como es Tomcat. Para ello iremos a la clase GestionCursosGrupo4Application:



Que quedará así tras modificarla.

```
package com.gestioncursos;

import org.springframework.boot.SpringApplication;

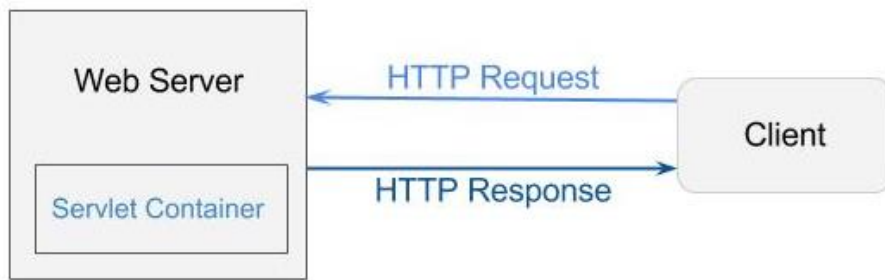
@SpringBootApplication
public class GestionCursosGrupo3Application extends SpringBootServletInitializer{

    public static void main(String[] args) {
        SpringApplication.run(GestionCursosGrupo3Application.class, args);
    }

    @Override
    protected SpringApplicationBuilder configure(SpringApplicationBuilder builder) {
        return builder.sources(GestionCursosGrupo3Application.class);
    }

}
```

Le añadimos el extends SpringBootServletInitializer, que implementa la interfaz WebApplicationInitializer, que automáticamente nos gestiona ServletContext y se comunica con el Servlet Container de java, que nos facilita la comunicación cliente/servidor con arquitectura J2EE.



Tras esto nos vamos al archivo pom.xml y añadimos ciertos detalles; el formato del 'packaging' a 'war', pues es la extensión que usaremos más adelante para Tomcat.

```
<name>gestioncursosgrupo3</name>
<description>AD - Grupo 3</description>
<packaging>war</packaging>
<properties>
    <start-class>com.gestioncursos.Gestio
```

Y la dependencia de Tomcat.

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-tomcat</artifactId>
    <scope>provided</scope>
</dependency>
```

2. Inicializando Tomcat

Primero, habrá que instalar Apache Tomcat en nuestra computadora.

Para ello hay que ir a la web oficial de Tomcat

(<https://tomcat.apache.org/download-90.cgi>) y descargar el zip de 64 bits.

Para el caso concreto de nuestra aplicación he optado por usar la versión 9.0 en vez de la última (la 10.0) ya que la 9 da menos conflictos y problemas de compatibilidad con Java 8 y versiones posteriores.

9.0.71

Please see the [README](#) file for packaging information. It explains what every distribution contains.

Binary Distributions

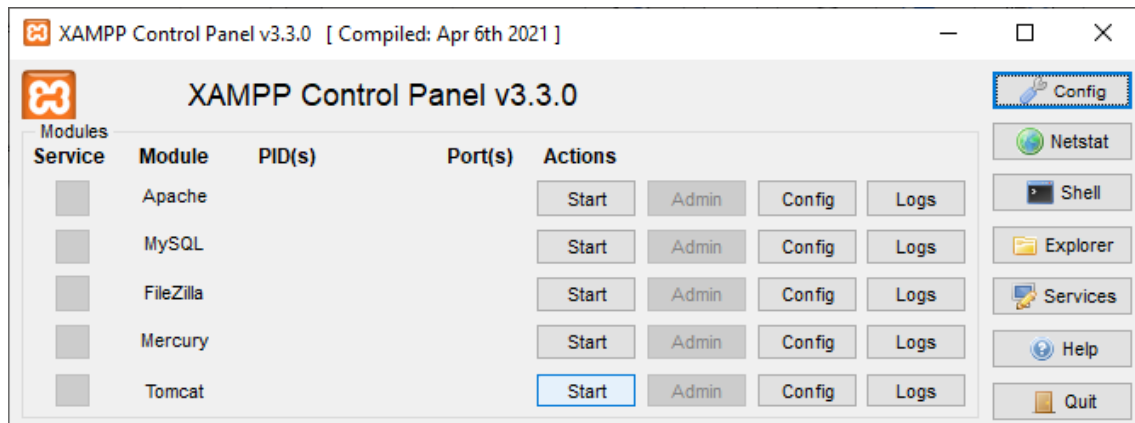
- Core:
 - [zip](#) ([pgp](#), [sha512](#))
 - [tar.gz](#) ([pgp](#), [sha512](#))
 - [32-bit Windows zip](#) ([pgp](#), [sha512](#))
 - [64-bit Windows zip](#) ([pgp](#), [sha512](#))
 - [32-bit/64-bit Windows Service Installer](#) ([pgp](#), [sha512](#))
- Full documentation:
 - [tar.gz](#) ([pgp](#), [sha512](#))
- Deployer:
 - [zip](#) ([pgp](#), [sha512](#))
 - [tar.gz](#) ([pgp](#), [sha512](#))
- Embedded:
 - [tar.gz](#) ([pgp](#), [sha512](#))
 - [zip](#) ([pgp](#), [sha512](#))

Una vez descargado el zip lo extraemos y arrancamos el servidor, lo cual podemos hacer de dos formas:

- Navegando a la subcarpeta \bin del directorio de Tomcat por la consola de comandos y ejecutando el archivo catalina.bat desde ahí.
En nuestro caso:

```
e Microsoft Windows [Versión 10.0.19044.2486]
p:(c) Microsoft Corporation. Todos los derechos reservados.
c:\Users\ana21>cd C:\apache-tomcat-9.0.71-windows-x64\apache-tomcat-9.0.71\bin
t C:\apache-tomcat-9.0.71-windows-x64\apache-tomcat-9.0.71\bin>catalina.bat run
```

- Ejecutándolo desde Xampp.



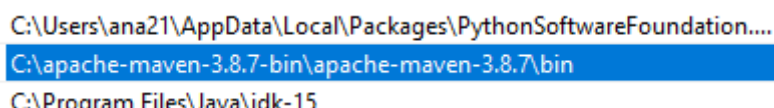
Sea como sea, nuestro servidor Tomcat se iniciará y estará configurado por defecto al puerto 8080. En caso de querer configurarlo podemos ir al archivo server.xml en la carpeta de la aplicación.

3. Exportando los archivos y desplegando el servidor

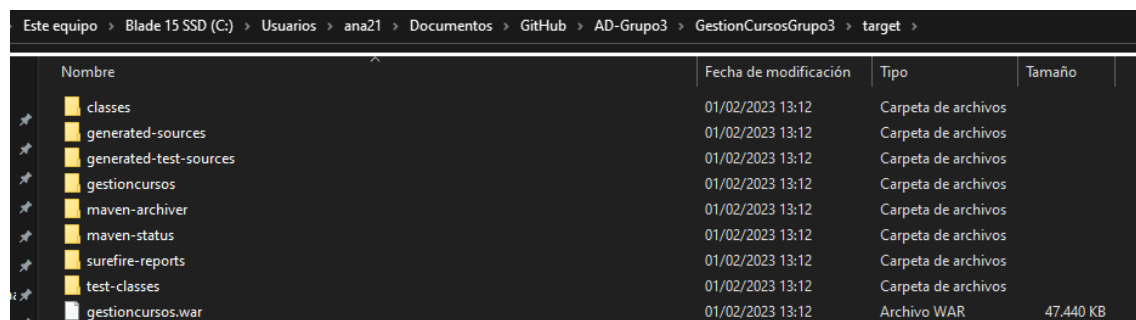
Ahora tenemos que exportar un archivo *.war que pueda ser desplegado en Tomcat. Para ello usaremos Maven. Antes de nada, tendremos que configurar un par de detalles en el pom.xml de nuevo.

```
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-war-plugin</artifactId>
  <version>3.3.1</version>
  <configuration>
    <warName>gestioncursos</warName>
  </configuration>
</plugin>
```

Esta configuración especifica el uso de la versión 3.3.1 de Apache Maven WAR plugin, que es la correcta para nuestra versión de Java (11.0.12). Tras esto ejecutamos el comando ‘mvn clean package’ en el cmd. Si nos da error, será probablemente porque no tenemos configurada la variable de entorno de Maven en nuestro PATH. Si ese es el caso, vamos a las variables de entorno del sistema, editamos ‘path’ y añadimos la ruta \bin de nuestra carpeta de Apache Maven.

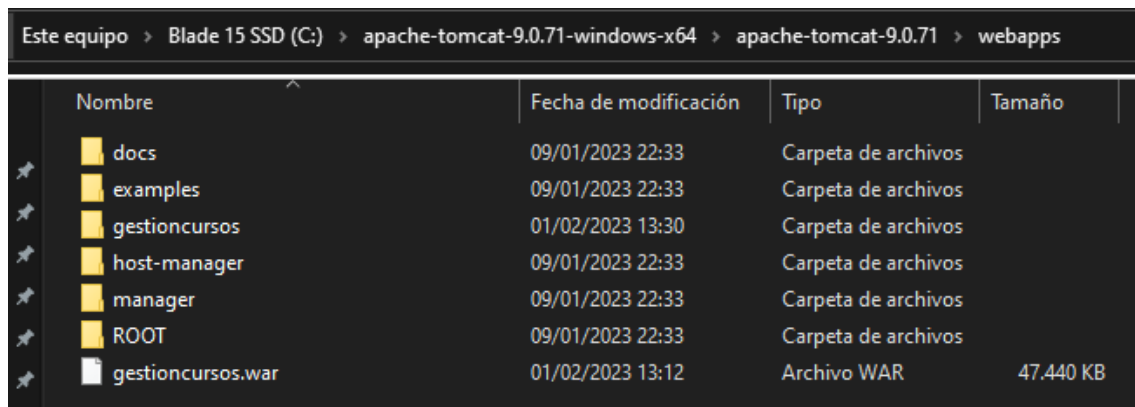
A screenshot of the Windows environment variables 'path' list. The list contains three entries: 'C:\Users\ana21\AppData\Local\Packages\PythonSoftwareFoundation....', 'C:\apache-maven-3.8.7-bin\apache-maven-3.8.7\bin' (which is highlighted in blue), and 'C:\Program Files\Java\jdk-15'.

Una vez hecho esto y ejecutado el comando ‘mvn clean package’, se nos generará un archivo .war en la carpeta ‘target’ de nuestro proyecto de Spring.

A screenshot of a file explorer window showing the 'target' directory of a project. The breadcrumb path is 'Este equipo > Blade 15 SSD (C:) > Usuarios > ana21 > Documentos > GitHub > AD-Grupo3 > GestionCursosGrupo3 > target'. The table below lists the contents of the directory.

Nombre	Fecha de modificación	Tipo	Tamaño
classes	01/02/2023 13:12	Carpeta de archivos	
generated-sources	01/02/2023 13:12	Carpeta de archivos	
generated-test-sources	01/02/2023 13:12	Carpeta de archivos	
gestioncursos	01/02/2023 13:12	Carpeta de archivos	
maven-archiver	01/02/2023 13:12	Carpeta de archivos	
maven-status	01/02/2023 13:12	Carpeta de archivos	
surefire-reports	01/02/2023 13:12	Carpeta de archivos	
test-classes	01/02/2023 13:12	Carpeta de archivos	
gestioncursos.war	01/02/2023 13:12	Archivo WAR	47.440 KB

Lo que tendremos que hacer con este archivo es copiarlo al directorio de Tomcat, en concreto a la subcarpeta \webapps.



Nombre	Fecha de modificación	Tipo	Tamaño
docs	09/01/2023 22:33	Carpeta de archivos	
examples	09/01/2023 22:33	Carpeta de archivos	
gestioncursos	01/02/2023 13:30	Carpeta de archivos	
host-manager	09/01/2023 22:33	Carpeta de archivos	
manager	09/01/2023 22:33	Carpeta de archivos	
ROOT	09/01/2023 22:33	Carpeta de archivos	
gestioncursos.war	01/02/2023 13:12	Archivo WAR	47.440 KB

Tras copiarlo, si esperamos unos segundos, veremos que Tomcat desplegará automáticamente el archivo y lo extraerá a una carpeta con el mismo nombre, ‘gestioncursos’ en nuestro caso.

Llegados a este punto, si vamos a nuestro navegador y escribimos <http://localhost:8080/gestioncursos/> veremos nuestra aplicación desplegada con éxito.

Véase que el nombre usado, ‘gestioncursos’, es altamente conveniente. Eso es porque en el archivo pom.xml hemos especificado el nombre final de la salida. Para ello sólo tenemos que añadir la línea ‘final name’ a <build>, para que coja el nombre del artifactId de nuestro proyecto como nombre del archivo de salida, aunque no es necesario. En mi caso lo he usado porque queda mejor un nombre simple como ‘gestioncursos’ que el predeterminado, que sería algo tal que ‘GestionCursosGrupo3-0.0.1-SNAPSHOT.war’

```
<build>
  <finalName>${project.artifactId}</finalName>
</build>
```

Si hacemos uso de esto, simplemente ejecutamos de nuevo el comando ‘mvn clean package’ y nuestro archivo se generará otra vez en la carpeta target del proyecto.

Webgrafía:

Springcloud

<https://www.springcloud.io/post/2022-09/springboot-tomcat/#gsc.tab=0>

Chat OpenAI

<https://chat.openai.com/chat>

Vaadin

<https://vaadin.com/forum/thread/18571634/spring-boot-application-war-deployment-404-not-found>

Baeldung

<https://www.baeldung.com/spring-boot-war-tomcat-deploy>

The Server Side

<https://www.theserverside.com/blog/Coffee-Talk-Java-News-Stories-and-Opinions/Top-5-ways-to-deploy-a-WAR-file-to-Tomcat>