5. Web Application Server Administration

Miquel Àngel París i Peñaranda

Web Application Deployment

2nd C-VET Web Application Development

Index

WildFly guided practice

1. Create the Java Application (WAR)

We will create a simple web application using **Java EE (Jakarta EE)** and package it as a **WAR** file.

Step 1: Create a Maven Project

Run the following command in your terminal to create a new Maven project:

```
mvn archetype:generate -DgroupId=iesjc.daw2 -DartifactId=HelloWorld -DarchetypeArtifactId=maven-archetype-webapp -DinteractiveMode=false
```

This will generate a basic web project structure in the HelloWorld folder.

Step 2: Modify `pom.xml`

Open the `pom.xml` file and ensure it includes the following:

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
<modelVersion>4.0.0</modelVersion>

<groupId>iesjc.daw2</groupId>
<artifactId>HelloWorld</artifactId>
<version>1.0-SNAPSHOT</version>
<packaging>war</packaging>
...
<!-- https://mvnrepository.com/artifact/jakarta.servlet/jakarta.servlet-api -->
<dependency>
        <groupId>jakarta.servlet</groupId>
        <artifactId>jakarta.servlet-api</artifactId>
        <version>6.1.0</version>
        <scope>provided</scope>
</dependency>
...

<build>
    <finalName>wildfly</finalName>
    <pluginManagement><!-- lock down plugins versions to avoid using Maven defaults (may be moved to parent pom) -->
        <plugins>
          <plugin>
            <artifactId>maven-clean-plugin</artifactId>
            <version>3.4.0</version>
          </plugin>
          <!-- see http://maven.apache.org/ref/current/maven-core/default-bindings.html#Plugin_bindings_for_war_packaging -->
          <plugin>
            <artifactId>maven-resources-plugin</artifactId>
            <version>3.3.1</version>
            </plugin>
            <plugin>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.13.0</version>
            </plugin>
            <plugin>
                <artifactId>maven-surefire-plugin</artifactId>
                <version>3.3.0</version>
            </plugin>
            <plugin>
                <artifactId>maven-war-plugin</artifactId>
                <version>3.4.0</version>
            </plugin>
            <plugin>
                <artifactId>maven-install-plugin</artifactId>
                <version>3.1.2</version>
            </plugin>
            <plugin>
                <artifactId>maven-deploy-plugin</artifactId>
                <version>3.1.2</version>
            </plugin>
        </plugins>
    </pluginManagement>
</build>
</project>
```

Step 3: Create a Servlet

```
package iesjc.daw2;

import java.io.IOException;
import java.io.PrintWriter;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;

@WebServlet("/hello")
public class HelloWorldServlet extends HttpServlet {
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
response.setContentType("text/html");
PrintWriter out = response.getWriter();
out.println("<h1>Hello, world from WildFly!</h1>");
}
}
```

Step 4: Modify `web.xml` (Optional)

If you prefer configuring the servlet in `web.xml`, edit `src/main/webapp/WEB-INF/web.xml`:

```
<web-app xmlns="http://jakarta.ee/xml/ns/jakartaee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://jakarta.ee/xml/ns/jakartaee http://jakarta.ee/xml/ns/jakartaee/web-app_5_0.xsd" version="5.0">
    <servlet>
```

```
        <servlet-name>HelloWorld</servlet-name>
        <servlet-class>iesjc.daw2.HelloWorldServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>HelloWorld</servlet-name>
        <url-pattern>/hello</url-pattern>
    </servlet-mapping>
</web-app>
```

Step 5: Build the WAR File

Run the following command from the project directory:

```
mvn clean package
```

This will generate a **HelloWorld.war** file inside `target/`.

2. Configure WildFly and Deploy the Application

Step 1: Download and Install WildFly

If you don’t have WildFly installed, download it from the official website:

https://www.wildfly.org/downloads/

Extract the file to your preferred location, for example:

```
tar -xvf wildfly-XX.XX.XX.Final.tar.gz
mv wildfly-XX.XX.XX.Final /opt/wildfly
```

Step 2: Configure the Administrator

If this is your first time using WildFly, create an admin user:

```
cd /opt/wildfly/bin
./add-user.sh
```

Follow the prompts to create an admin user.

Step 3: Start WildFly

To start WildFly in standalone mode, run:

```
cd /opt/wildfly/bin
./standalone.sh
```

If you are on a headless server, you can start it in the background:

Step 4: Access the Admin Console

Open a browser and go to:

http://serverip:port

Log in with the admin user you created.

Step 5: Deploy the Application

You have two options to deploy the `WAR` file:

Option 1: Using the Web Console

1. Navigate to the **Deployments** tab.
2. Click Add → Upload Deployment.
3. Select the **HelloWorld.war** file from `target/`.
4. Click **Enable**.

Option 2: Copying the WAR to the Server

If you don’t have access to the GUI, you can copy the `WAR` file to the `deployments` directory:

```
cp target/HelloWorld.war /opt/wildfly/standalone/deployments/
```

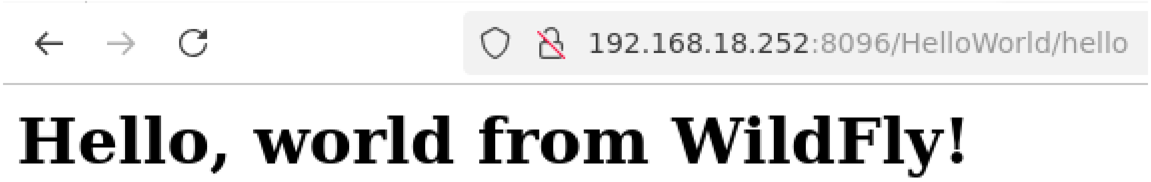
WildFly will automatically detect and deploy the file.

3. Test the Application

Once deployed, open a browser and visit:

http://serverip:port/HelloWorld/hello

You should see:



192.168.18.252:8096/HelloWorld/hello

Hello, world from WildFly!