Comprehensive Guide: WildFly Deployment on Helios

By Idris Shuaibu
ISU: 463221

Abstract

This guide provides a detailed, step-by-step procedure for new students to deploy a Java Web Archive (WAR) application to a WildFly server running on Helios. We address the entire lifecycle, from building the application to resolving complex networking hurdles.

Port Configuration with +5000 Offset

To avoid conflicts on a shared server, we will use a port offset of **5000**.

- **Application HTTP Port:** 8080 + 5000 = **13080** (Access your app via browser)
- **Management Console Port:** 9990 + 5000 = **14990** (Administrative panel)
- **Example Application Name:** `my-application.war`. This name is used in the URL, e.g., `/my-application`.

1 Phase 0: Application and Server File Preparation

1.1 Building the Web Archive (WAR) File

Context

This step converts your Java project code into a single deployable bundle. If you are using Maven (common for Jakarta EE), the build process generates the WAR file.

Action In your IDE (e.g., IntelliJ, Eclipse), use your build tool (e.g., Maven) to execute the **'package'** goal.

Result The final `my-application.war` file is typically created in the `target/` directory of your project folder.

1.2 Downloading and Unzipping WildFly on Helios

Important

Run the following commands in your Helios **home directory**!

Action 1: Download WildFly

Use `wget` to download the WildFly ZIP file (example shown for version 38.0.0).

```
wget https://github.com/wildfly/wildfly/releases/download/38.0.0.Final/wildfly
```

Action 2: Unzip the Archive

Use `unzip` to extract the server files. This creates the WildFly installation directory (e.g., `wildfly-38.0.0.Final`).

```
unzip wildfly-38.0.0.Final.zip
```

2 Phase 1: Server Setup and Launch

2.1 Understanding Default Port Conflicts

The Problem

On a shared server like Helios, default ports like 8080 (HTTP) and 9990 (Management) are often in use. If WildFly attempts to bind to an in-use port, it throws an "Address already in use" error and immediately shuts down. To solve this, we use a port offset.

2.2 Navigate to the WildFly Binary Directory

All administrative and server-launch commands are executed from the WildFly `bin` directory.

```
cd wildfly-38.0.0.Final/bin
```

2.3 Launch WildFly with a Port Offset (Critical Solution)

Use the `-Djboss.socket.binding.port-offset` argument to shift all default WildFly ports by a large number (e.g., 5000) to find a free range.

```
./standalone.sh -Djboss.socket.binding.port-offset=5000
```

Important

Leave this terminal window open. This process must run in the foreground. If the command returns to the prompt without a "Server Started" message, the server has failed.

3 Phase 2: Networking (SSH Port Forwarding)

3.1 Establish an SSH Tunnel

Problem & Solution

Your local web browser cannot directly access a port on the remote Helios server. You must use SSH local port forwarding to create a secure tunnel, which tricks your browser into thinking the remote port is local.

Action Open a **new terminal window on your local machine** and run the following command. This maps your local ports to the remote server's ports.

```
ssh -L 13080:localhost:13080 -L 14990:localhost:14990
sxxxxxx@helios.cs.ifmo.ru -p 2222
```

Replace `sxxxxxx` with your Helios username.

Important

Keep this second terminal window open as well. If the SSH session closes, the tunnels close, and access is lost.

4 Phase 3: Application Deployment (Choose a Method)

4.1 Method A: Web Management Console (Recommended)

4.1.1 Create a Management User

Action Open a **third terminal window** connected to Helios and navigate to the WildFly `bin` directory. Run the `add-user.sh` script to create an administrator.

```
./add-user.sh
```

Input Choose option **a** (Management User). Create and securely remember the username and password.

4.1.2 Access and Deploy via Browser

1. Open your web browser and go to the forwarded management port: `http://localhost:14990/`
2. Log in with the credentials created in the previous step.
3. Navigate to the **Deployments** tab.
4. Click the **Add** button (+) and upload your local `my-application.war` file.
5. The server will automatically deploy the application.

4.2 Method B: File System Copy (Command Line)

4.2.1 Stop the Server

Go to the first terminal (running `./standalone.sh`) and press **Ctrl+C** to shut down WildFly.

4.2.2 Transfer the WAR File

Action Navigate to the WildFly deployment directory on the remote server.

```
cd wildfly-38.0.0.Final/standalone/deployments/
```

Transfer Use `scp` or WinSCP to transfer your `my-application.war` file from your local machine directly into this folder.

4.2.3 Restart the Server

Go back to the `bin` directory and restart the server as you did in Phase 1. The deployment scanner will automatically detect and deploy the file on startup.

```
./standalone.sh -Djboss.socket.binding.port-offset=5000
```

Important

Wait for the "Server started" message before proceeding.

5 Phase 4: Final Access

5.1 Access Your Application

Action Combine your forwarded HTTP port (**13080**) with the application's context root (the WAR filename without the `.war` extension).

Final URL Open the following URL in your browser: `http://localhost:13080/my-application`

Success! If you see your application, deployment is complete!

Troubleshooting Tip

Diagnosing Port Conflicts

If the server fails to start, it is most likely a port conflict. To diagnose, run a utility like `lsof` on the Linux server to find the conflicting Process ID (PID) and terminate it.

```
Example:  Find what is using port 9990 lsof -i :9990
Terminate the conflicting process (replace <PID> with the
ID found) kill -9 <PID>
```