Online Auction

Tomcat Deployment Manual

**Slippery Rock University of Pennsylvania**

Contributions by:

**Fall 2023**

Timothy Holtz – tth1003@sru.edu

Douglas Maxwell – dsm1015@sru.edu

Luke McElligott – lpm1006@sru.edu

# Table of Contents

[Table of Contents 2](#_Toc151814419)

[Setting up a Local Tomcat Environment 3](#_Toc151814420)

[Prerequisites 3](#_Toc151814421)

[Configuration 3](#_Toc151814422)

[Verify the Installation 4](#_Toc151814423)

[Deployment 4](#_Toc151814424)

[Tomcat Server Deployment 4](#_Toc151814425)

[Robotics Lab Specifications 4](#_Toc151814426)

[Spring project Specifications 4](#_Toc151814427)

[Application Packaging: 5](#_Toc151814428)

[Build WAR file 5](#_Toc151814429)

[Deploy the WAR File to Tomcat 5](#_Toc151814430)

[Access Your Deployed Application 6](#_Toc151814431)

[Monitor and Troubleshoot 6](#_Toc151814432)

[Configure SSL in Tomcat 6](#_Toc151814433)

[Generate a Keystore 6](#_Toc151814434)

[Configure Tomcat for HTTPS 7](#_Toc151814435)

[Test the Configuration 7](#_Toc151814436)

# 

# Setting up a Local Tomcat Environment

## Prerequisites

* Java Development Kit (JDK) 17: Ensure you have JDK 17 installed on your system. You can download it from Oracle's official website or use OpenJDK. <https://www.oracle.com/java/technologies/javase/jdk17-archive-downloads.html>
* Apache Tomcat 10.0.x: You will need the Tomcat 10.0.x distribution, which can be downloaded from the Apache Tomcat website. <https://archive.apache.org/dist/tomcat/tomcat-10/v10.0.27/bin/>
  + \*Note – Tomcat 10.0.x has been superseded by Tomcat 10.1.x, so you must install it from the archive link. I suggest that the lab upgrades to Tomcat 10.1.x to support new security patches. Most of this document will remain relevant.
* Spring Boot 3.1.x: Since the server is running Tomcat 10, we need to support jakarta file structures as opposed to javax (see <https://tomcat.apache.org/migration-10.html>)

## Configuration

Ensure **JAVA\_HOME** is set and points to your JDK 17 installation.

Ex. “C:\Program Files\Java\jdk-17”

Pom.xml snippet for Spring Boot 3.1.x and JDK 17:

<parent>

    <groupId>org.springframework.boot</groupId>

    <artifactId>spring-boot-starter-parent</artifactId>

    <version>3.1.5</version>

    <relativePath/>

</parent>

…

<properties>

    <java.version>17</java.version>

</properties>

## Verify the Installation

1. Open a command prompt or terminal in **Admin** mode.
2. Navigate to the bin directory of your Tomcat installation.
3. Run catalina.bat run on Windows
4. If Tomcat starts successfully, you should see a message indicating that Tomcat is running.
5. Open a web browser and go to **http://localhost:8080**. You should see the Tomcat welcome page.

## Deployment

To deploy your web applications:

Place your WAR file or unpacked web application directory into the webapps folder of your Tomcat installation. Restart Tomcat to deploy the application.

Once Tomcat has started and deployed your application, you can access it in a web browser by entering the following URL:

**http://localhost:8080/your-application-name**

# Tomcat Server Deployment

This section outlines the requirements and steps for deploying a Java web application to a server running Apache Tomcat 10.0.x and using JDK 17. It covers key considerations to ensure compatibility and successful deployment.

## Robotics Lab Specifications

Tomcat Version: 10.0

Java: JDK 17

## Spring project Specifications

Tomcat Target Version: 10.0

Java: JDK 17

Spring Boot: 3.1.x

## Application Packaging:

**WAR File:** Package your Java web application as a Web Application Archive (WAR) file. Tomcat deploys web applications from WAR files.

**Maven Configuration:** Using Maven, ensure that your pom.xml file specifies JDK 17 as the source and target versions, like this:

<properties>

<java.version>17</java.version>

</properties>

…

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<configuration>

<source>17</source>

<target>17</target>

</configuration>

</plugin>

</plugins>

</build>

## Build WAR file

Open a terminal or command prompt. Navigate to the root directory of your project where the pom.xml file is located.

Execute the following command:

mvn clean package

This command will compile your project, run tests (if any), and package your application into a WAR file. The WAR file will be created in the target directory of your project.

## Deploy the WAR File to Tomcat

\*Note – The following steps must be done by a faculty member with access to the Robotics server.

Once you have generated the WAR file, you can deploy it to your Tomcat server:

Locate the generated WAR file in the target directory of your project. The WAR file will have a name like your-application-name.war.

Copy the WAR file to the Tomcat webapps directory. By default, this directory is located in the Tomcat installation folder.

Start or restart Tomcat to deploy the application. You can use the provided startup scripts or commands depending on your operating system:

* On Windows, you can use **startup.bat** in the **TOMCAT\_HOME/bin** directory.

## Access Your Deployed Application

Once Tomcat has started and deployed your application, you can access it in a web browser by entering the following URL:

**http://localhost:8080/your-application-name**

## Monitor and Troubleshoot

During and after deployment, monitor the Tomcat logs located in the **TOMCAT\_HOME/logs** directory for any errors or issues.

# Setup a GitHub Actions Pipeline

This is an optional step but a very helpful one. You can setup a pipeline in GitHub to automatically generate your war file every time you push to the master branch. It’s very simple, just add this file to the root of your project in “/.github/workflows/maven.yml”.

name: Java Maven Build & Publish Artifact

on:

push:

branches: [ "master" ]

jobs:

build\_test:

runs-on: ubuntu-latest

steps:

- uses: actions/checkout@v3

- name: Set up JDK 17

uses: actions/setup-java@v3

with:

java-version: '17'

distribution: 'temurin'

cache: maven

- name: Build with Maven

run: mvn -B package --file Path\_To\_Your\_Pom/pom.xml -DskipTests

publish-job:

runs-on: ubuntu-latest

needs: build\_test

steps:

- uses: actions/checkout@v3

- uses: actions/setup-java@v3

with:

java-version: '17'

distribution: 'temurin'

cache: maven

- run: mvn --batch-mode --update-snapshots verify --file Path\_To\_Your\_Pom /pom.xml -DskipTests

- run: mkdir staging && cp Your\_Project/target/\*.war staging

- uses: actions/upload-artifact@v3

with:

name: Package

path: staging

The build will automatically run next time you push to master. It will be located here in the Actions tab:

A screenshot of a computer

Description automatically generated

# Configure SSL in Tomcat

## Generate a Keystore

Open a Termainal or command prompt and use the following command:

keytool -genkeypair -alias tomcat -keyalg RSA -keysize 2048 -storetype PKCS12 -keystore keystore.p12 -validity 365

This command creates a keystore file named `keystore.p12` that's valid for 365 days.

\*Note - Remember the password you set here. You'll need it later!

## Configure Tomcat for HTTPS

You’ll have to place keystore.p12 somewhere on the target server.

Locate Tomcat’s `conf` directory and find the `server.xml` file. In the file, find the `<connector>` element and configure it for SSL:

<Connector port="8443" protocol="HTTP/1.1"

connectionTimeout="20000"

SSLEnabled="true"

scheme="https"

secure="true"

sslProtocol="TLS"

keystoreFile="path/to/keystore.p12"

keystorePass="your\_keystore\_password"

clientAuth="false"

keyAlias="tomcat"

keystoreType="PKCS12" />

Replace path/to/keystore.p12 with your keystore file's path and your\_keystore\_password with the password you set earlier.

## Test the Configuration

Restart Tomcat and then navigate to `https://yourserver:8443`.

You might see a warning about the server's certificate if you're using a self-signed certificate. This is normal for testing environments.