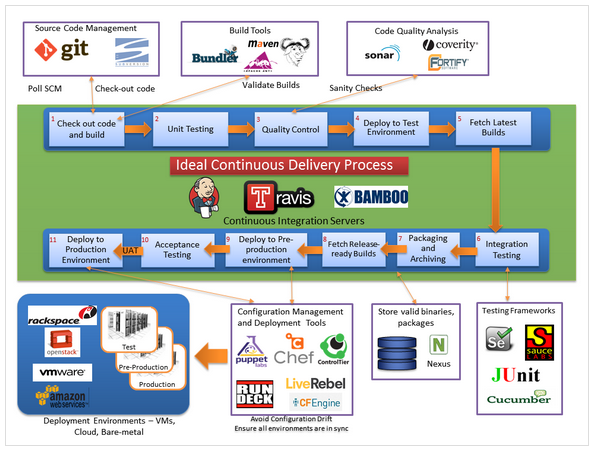
**Hello World DevOps Demo**

## Introduction

The main principles behind DevOps is to be able to do Continuous Integration, Automated Testing, Continuous Deployment, and Continuous Monitoring into an application. This should result in faster and error-prone deployments. A simple Hello World example is created using DevOps principles and tools which shows the necessary steps taken during a deployment pipeline.

The following shows a Continuous Delivery Pipeline and tools that can be used for different steps. The idea is to first set up the pipeline using the tools below and then create a simple application that flows automatically through that pipeline.



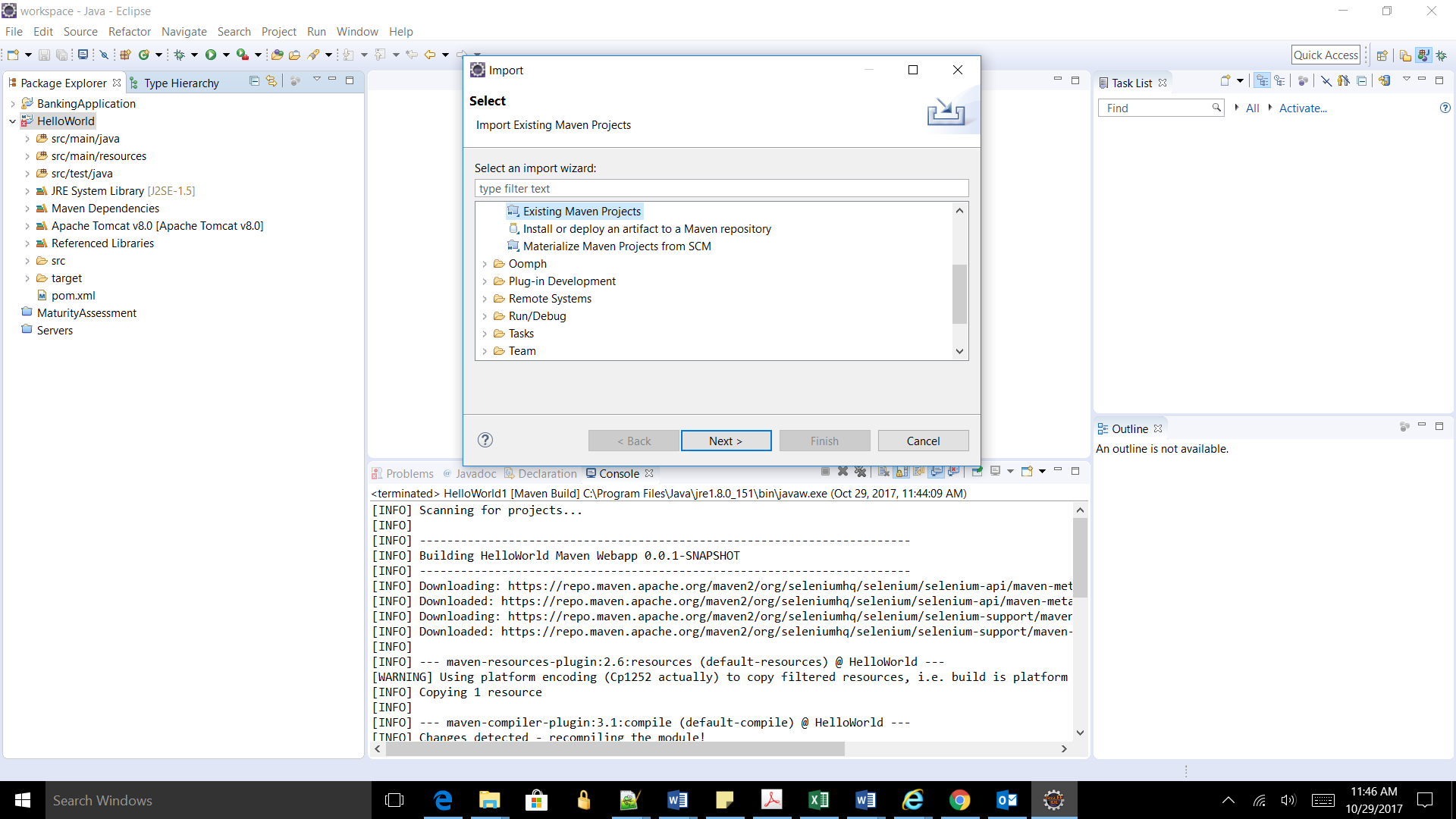
## Tools

The following tools are used to create the Hello World example and achieve Continuous Integration behavior.

* GitHub – Source Code Management
* Jenkins – Continuous Integration Server
* Maven – Build Tool
* SonarQube – Static Code Analysis
* Selenium – Automated Testing

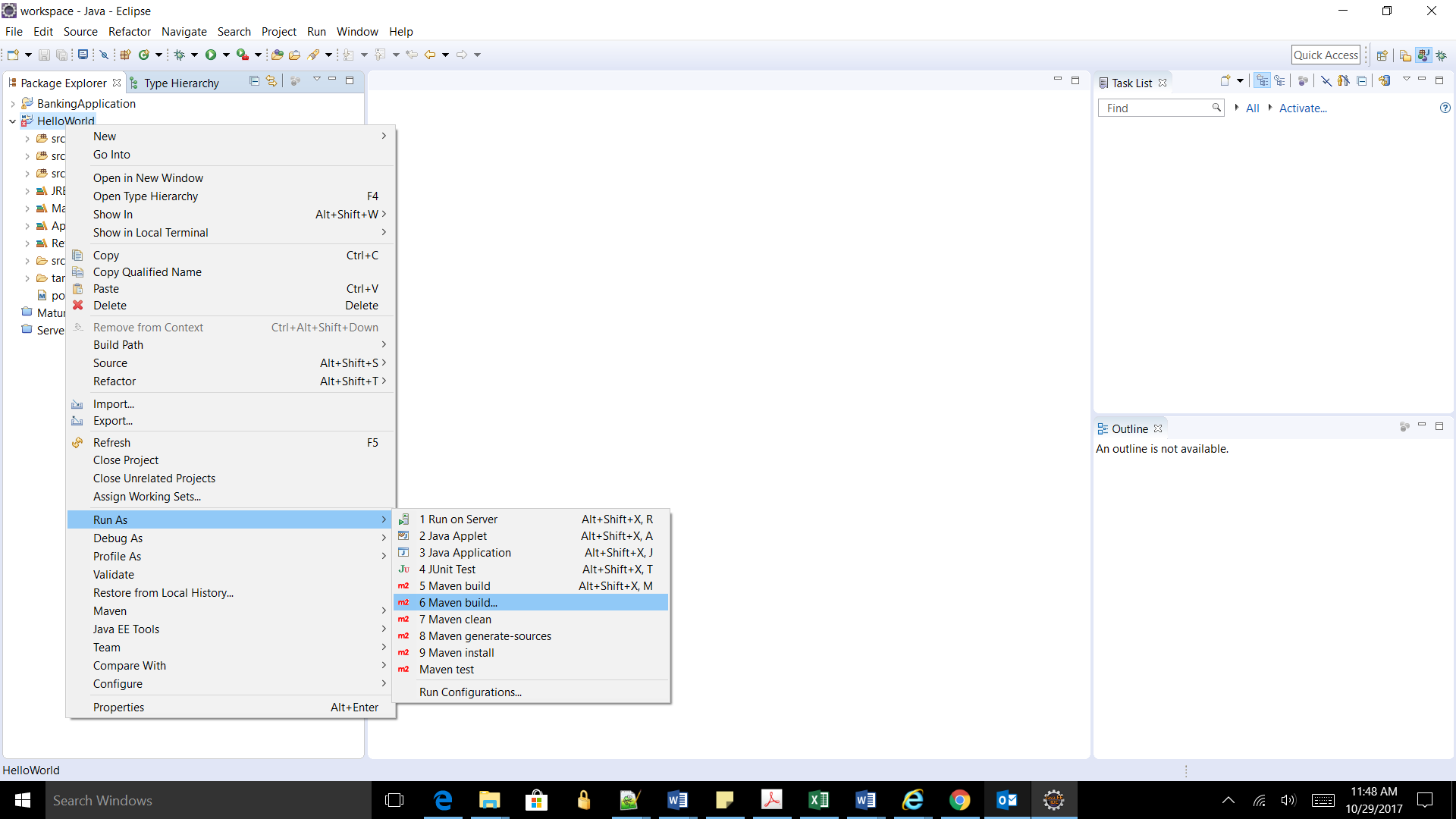
## Hello World Web Application

A simple java Hello World web application was created using maven version 3.3.9. The attached project can be imported into Eclipse by using File -> Import -> Existing Maven Project.

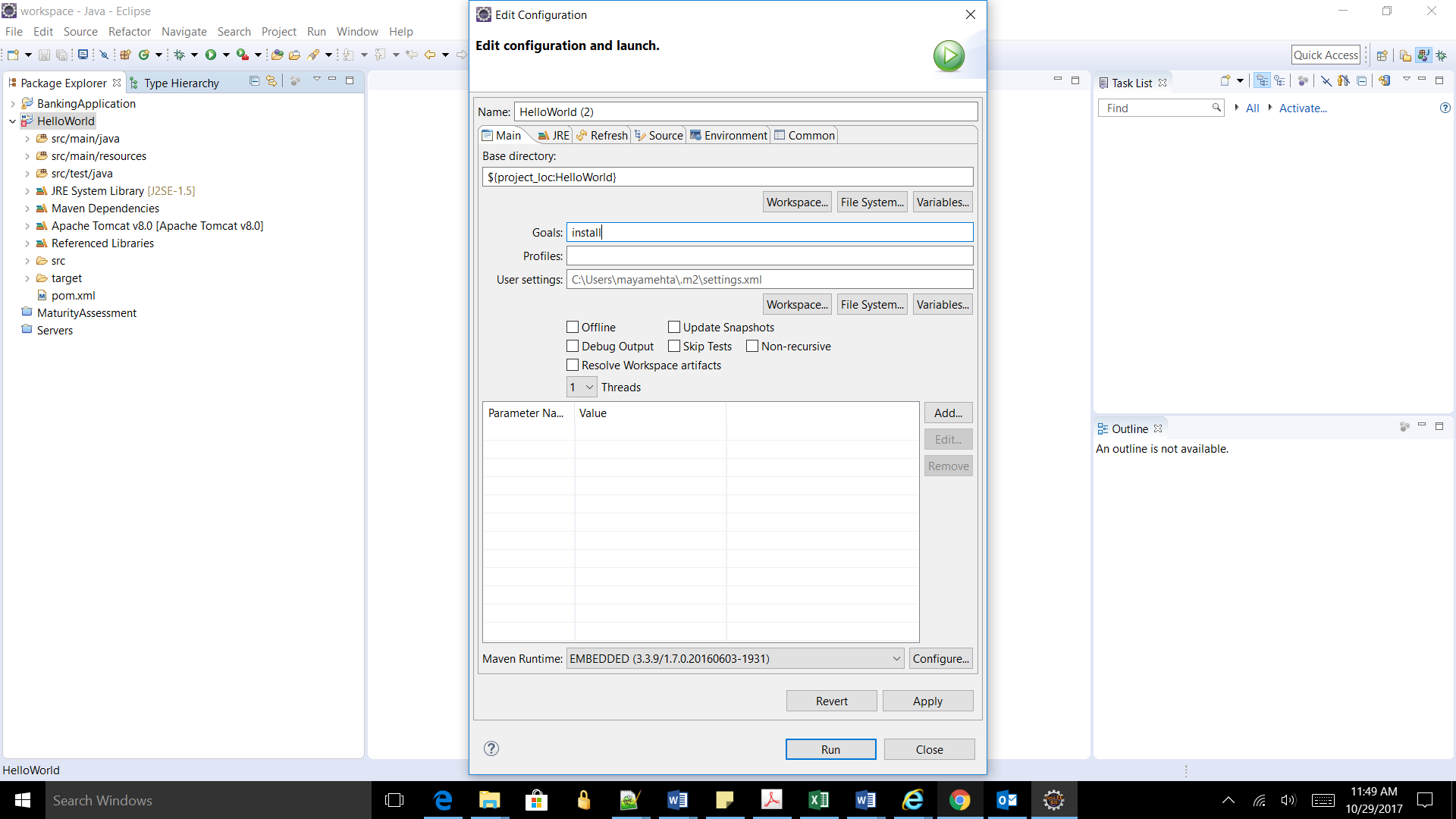


To build the project in Eclipse

Right click on the project and select Run As and Maven Build…



Enter “install” for goals and click Run



The project can also be compiled through command line by installing maven and executing mvn clean install command.

The output is a war file which can be deployed in a Tomcat server.

Eclipse IDE used

Eclipse Java EE IDE for Web Developers

Version: Neon.3 Release (4.6.3)

<https://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/neon3>

Maven

Apache Maven v3.3.9

<https://archive.apache.org/dist/maven/maven-3/3.3.9/>

Apache Tomcat v8.0

<https://tomcat.apache.org/download-80.cgi>

**Jenkins**

Jenkins is a powerful open source application that achieves Continuous Integration and Continuous Delivery of projects with ease. The main benefit of Jenkins is that it can be integrated with a number of testing and deployment technologies.

In this demo Jenkins was used as the build server. During a build process of the application, it pulled code from Github, build it using Maven, and deploy to a tomcat server upon successful build. After the deployment completed, it triggered regression testing which was a separate project on Jenkins. The regression testing was used to verify the functionality of the demo. This testing was done through Selenium.

A tutorial on Jenkins can be found at <https://www.tutorialspoint.com/jenkins/> which provides a step by step instructions on setting up Jenkins on a Tomcat server.