# Lab 4: Using WebDriver-Manager

In this lab, we are going to leverage WebDriverManager by Boni Garcia to setup browser drivers required to execute our tests.

We all know that we need to have browser drivers, .exe files like chromedriver.exe and geckodriver.exe in case of windows environment or binary files like chromedriver and geckodriver in case of linux distributions, in order to run our selenium webdriver automation scripts on chrome and firefox browsers.

And also we need to set the path of these files in our script like below or we need to add location to the class path.

**Example for chrome browser**

```
System.setProperty("webdriver.chrome.driver", "/path/to/binary/chromedriver");
```

**Example for firefox browser:**

```
System.setProperty("webdriver.gecko.driver", "/path/to/binary/geckodriver");
```

If the path is not defined or if the path provided is wrong, we will get an exception like below when running our tests.

```
Exception in thread "main" java.lang.IllegalStateException: The path to the driver
executable must be set by the webdriver.gecko.driver system property; for more
information, see https://github.com/mozilla/geckodriver. The latest version can be
downloaded from https://github.com/mozilla/geckodriver/releases
    at com.google.common.base.Preconditions.checkState(Preconditions.java:847)
    at
org.openqa.selenium.remote.service.DriverService.findExecutable(DriverService.java:125)

    at
org.openqa.selenium.firefox.GeckoDriverService.access$100(GeckoDriverService.java:43)
    at
org.openqa.selenium.firefox.GeckoDriverService$Builder.findDefaultExecutable(GeckoDrive

    at
org.openqa.selenium.remote.service.DriverService$Builder.build(DriverService.java:346)
    at org.openqa.selenium.firefox.FirefoxDriver.toExecutor(FirefoxDriver.java:168)
    at org.openqa.selenium.firefox.FirefoxDriver.(FirefoxDriver.java:125)
    at org.openqa.selenium.firefox.FirefoxDriver.(FirefoxDriver.java:103)
```

To avoid this error, we need to manually download and manage these drivers for each operating systems/environments and that is very painful. We also have to check and update relevant drivers when new versions of the binaries are released or new browsers versions are released along with compatibility for driver to browser.

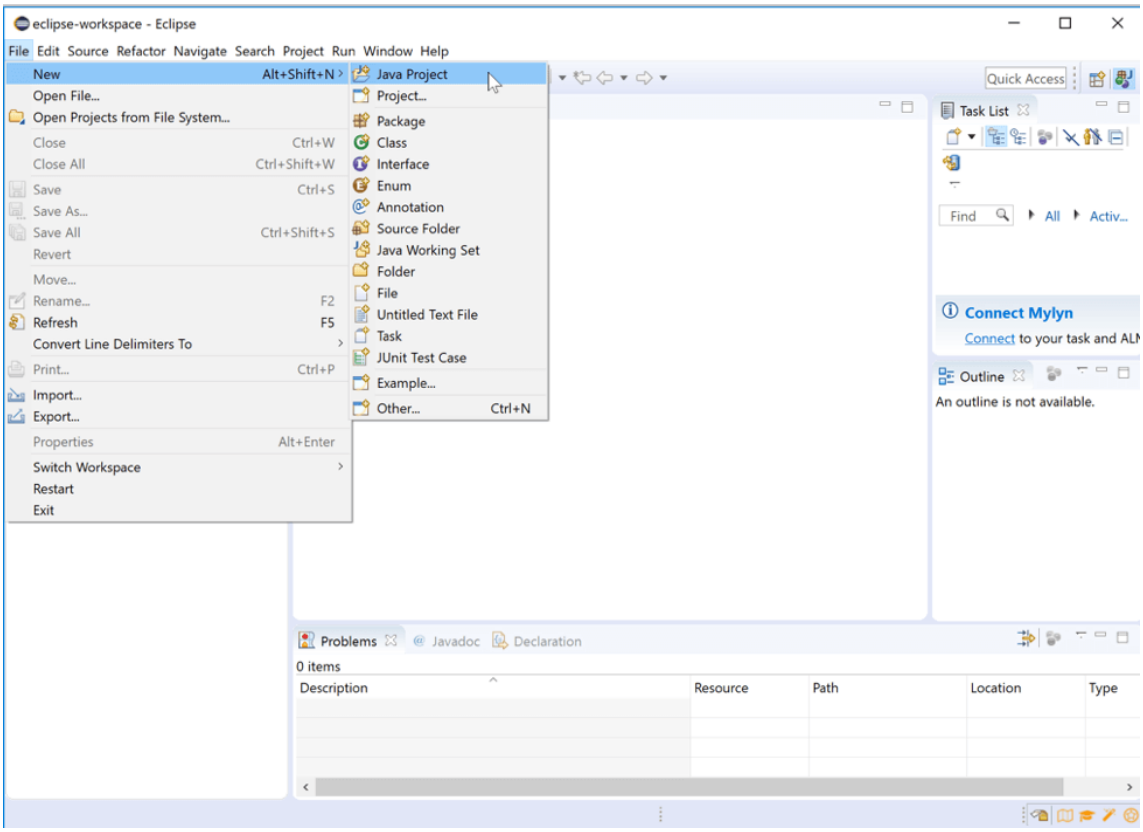
WebDriverManager by Boni Garcia helps us to manage driver related settings automatically. Webdriver manager downloads binaries/executables in an automated way and helps us to avoid all the manual steps that we do previously related to browser drivers to run our tests.

It supports browsers such as Chrome, Firefox, Opera, PhantomJS, Microsoft Edge, or Internet Explorer. You can check that in project page.

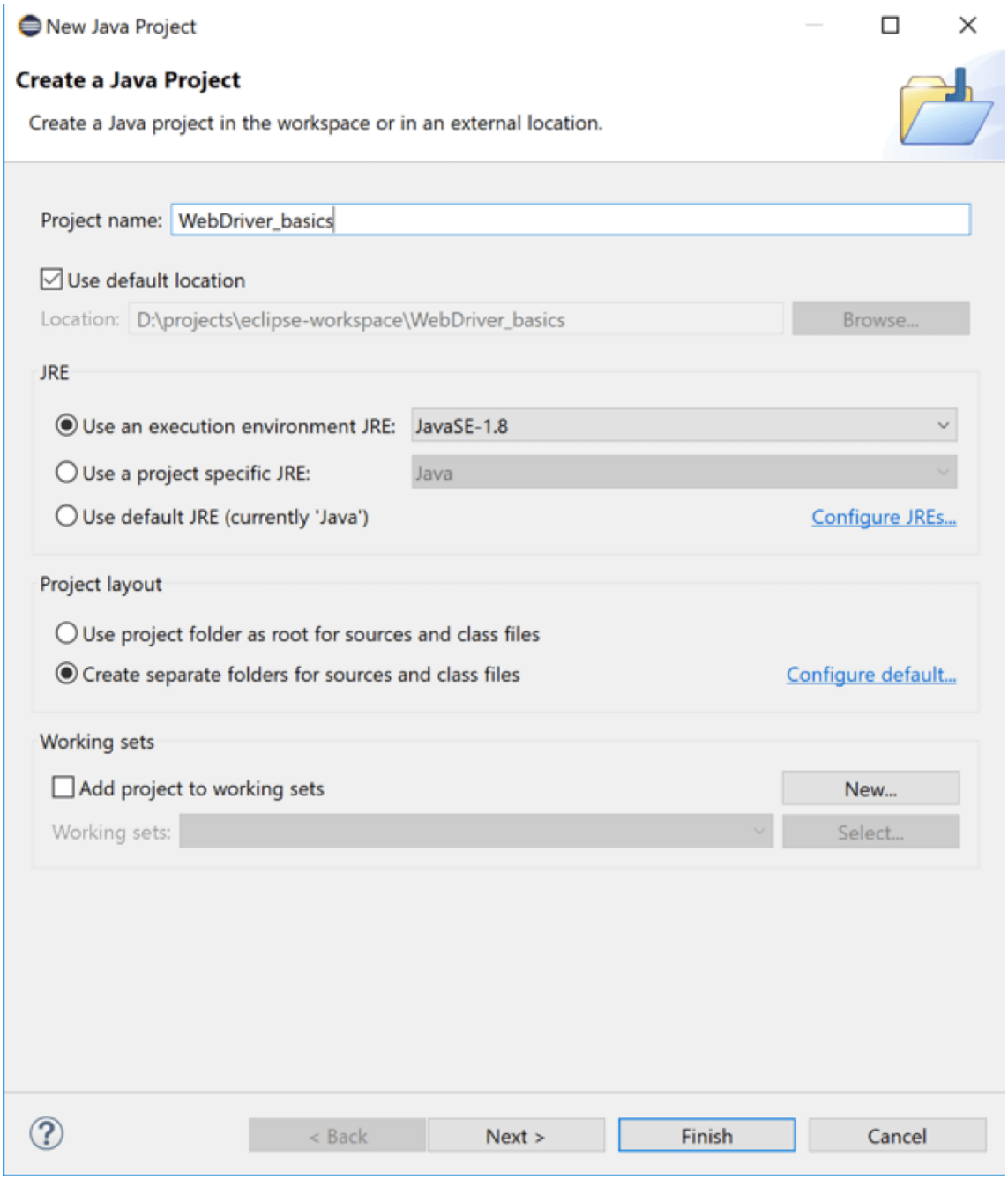
You can also download this jar file and its dependencies to add as external jars in case if your project is not a maven project.

Check maven page for more details of supported builder projects.

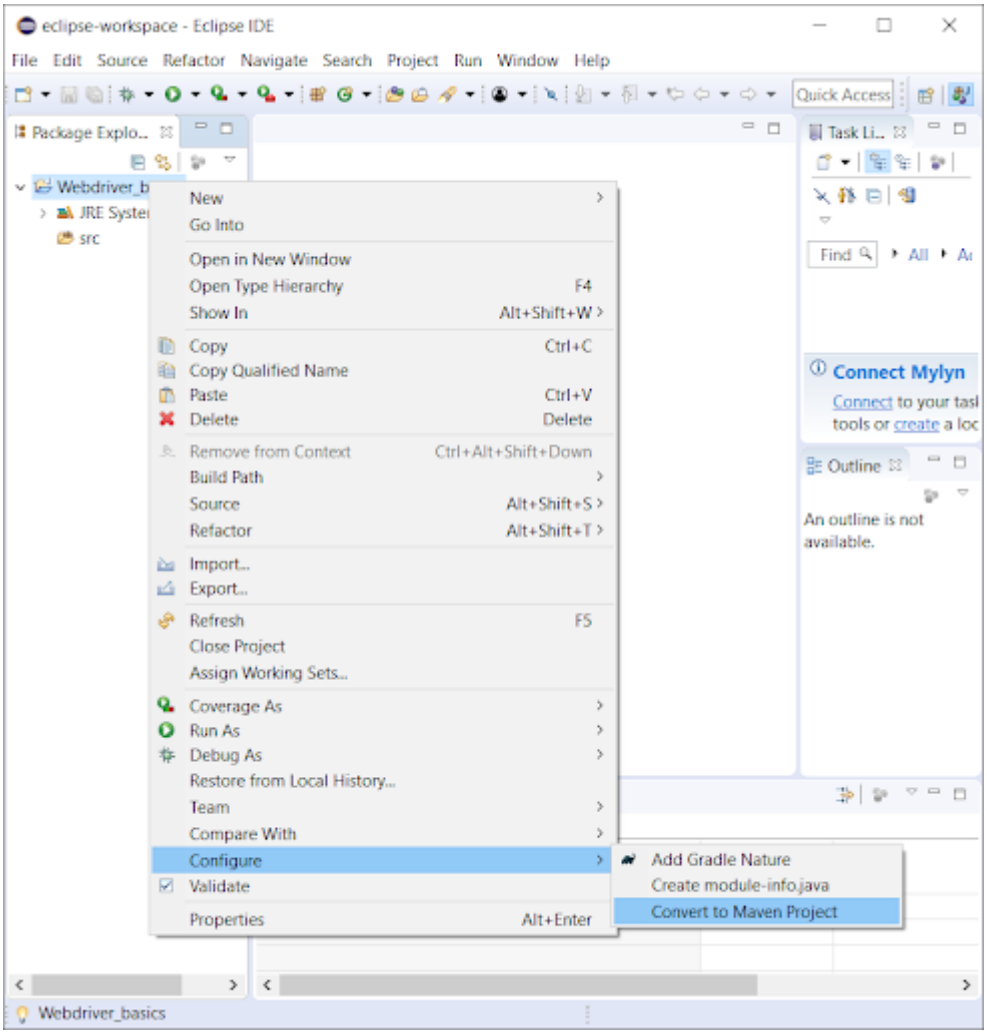
**Creating The Project** Create a simple Java Project created by using the menu File => New => Java Project.




Fill out the basic information on the New Java Project dialog, then click Finish to proceed.




This will bring up a dialogue box to create a pom.xml file. Click “Finish” and you’ll be brought to a screen that contains the file. This file contains information on where to download extra libraries for the project. We’ll add some XML code to the file.




In case of **Maven project**, we need to add the following dependency in pom.xml :-

```
<dependencies>
        <dependency>
            <groupId>org.seleniumhq.selenium</groupId>
            <artifactId>selenium-java</artifactId>
            <version>2.45.0</version>
        </dependency>
        <dependency>
        <groupId>io.github.bonigarcia</groupId>
        <artifactId>webdrivermanager</artifactId>
        <version>3.0.0</version>
        </dependency>
</dependencies>
```

**Example for launching chrome and firefox browser**

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
```

```
import io.github.bonigarcia.wdm.WebDriverManager;

public class WebDriverManagerTest {
    public static void main(String[] args) {
        new WebDriverManagerTest().testDriverManagerChrome();
        new WebDriverManagerTest().testDriverManagerFirefox();
    }

    public void testDriverManagerChrome() {
        WebDriverManager.chromedriver().setup();
        WebDriver driver = new ChromeDriver();
        driver.get("http://www.google.com/");
        System.out.println(driver.getTitle());
        driver.quit();
    }

    public void testDriverManagerFirefox(){
        WebDriverManager.firefoxdriver().setup();
        WebDriver driver = new FirefoxDriver();
        driver.get("http://www.google.com/");
        System.out.println(driver.getTitle());
        driver.quit();
    }
}
```

Webdrivermanager, by default, tries to download the latest version of a given browser driver binary. To use a specific version of driver, pass the driver version like below

```
WebDriverManager.chromedriver().driverVersion("94.0.4606.61").setup();
```

To download specific versions or from specific urls, change respective value of the variables in version.properties or webdrivermanager.properties depending on where it is available.
Also you can add support to any new versions of browser driver by defining it in version.properties.

Have a look at version.properties

Have a look at webdrivermanager.properties

At any time you want to change default values, copy the required files from here into your resources directory with exact same name and update required values in the property file. That's it.