

# WebDriverManager vs Selenium Manager

SeleniumConf & AppiumConf

28 March 2025

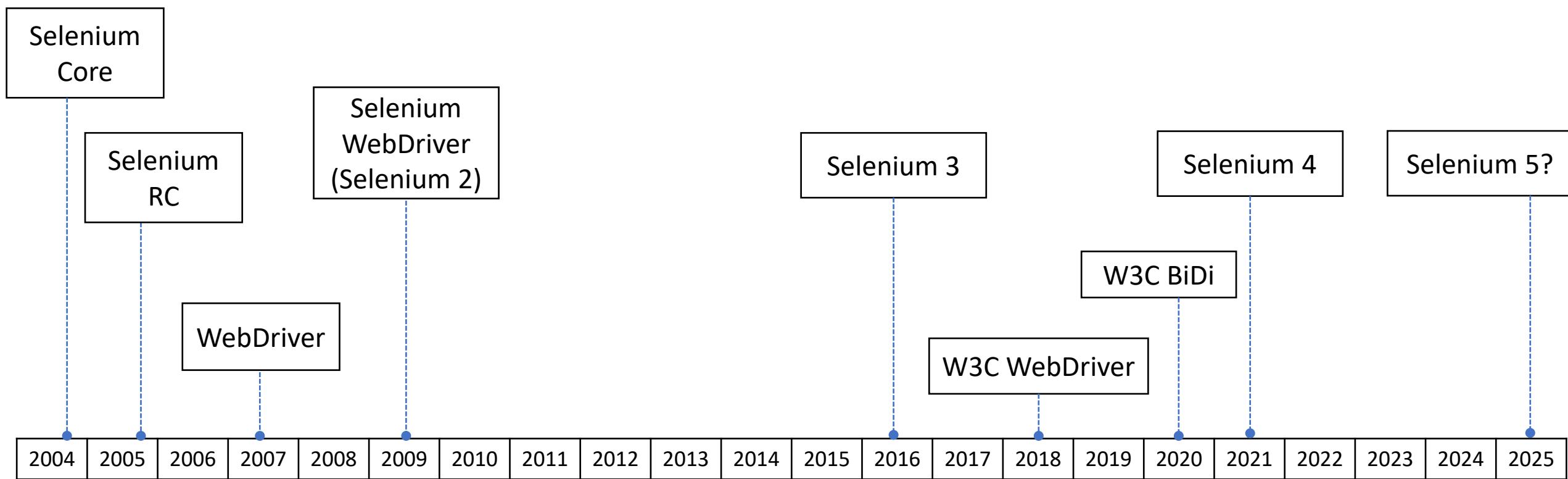
Boni García  
Universidad Carlos III de Madrid, Spain  
[boni.garcia@uc3m.es](mailto:boni.garcia@uc3m.es)



# Selenium is 20 years



# The History of Selenium





POLITÉCNICA

2006-2011



## CONTRIBUTION TO THE AUTOMATION OF SOFTWARE QUALITY CONTROL OF WEB APPLICATIONS

PhD Dissertation

Boni García

Telecommunication Engineer

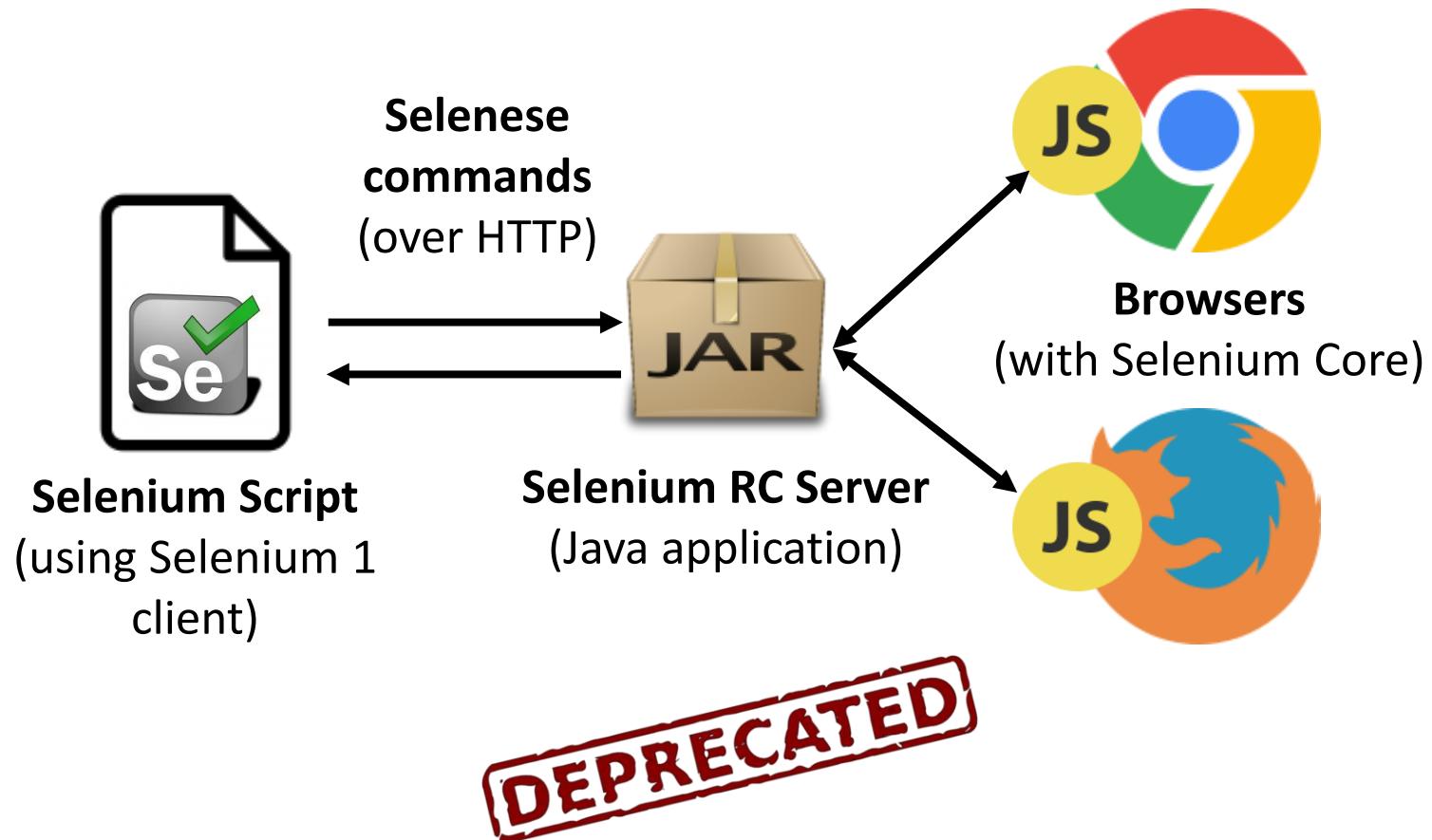
Supervisor

Juan Carlos Dueñas

PhD Telecommunication Engineer

# My History with Selenium

- Selenium Remote Control (RC):





# My History with Selenium

Kurento Tutorial 1: Magic Mirror

localhost:8080

Kurento Tutorial

Tutorial 2: Magic Mirror

This application shows a WebRtcEndpoint connected to itself (loopback) with a FaceOverlay filter in the middle (take a look to the [Media Pipeline](#)). To run this demo follow these steps:

1. Open this page with a browser compliant with WebRTC (Chrome, Firefox).
2. Click on Start button.
3. Grant the access to the camera and microphone. After the SDP negotiation the loopback should start.
4. Click on Stop to finish the communication.

Local stream

Remote stream

Start

Stop

Console

```
Created SDP offer
Local description set
ICE negotiation completed
Invoking SDP offer callback function localhost:8080
SDP answer received, setting remote description
```

© 2014 Kurento

NaevA tec

Kurento CrowdDetector

https://localhost:8443

Kurento Tutorial

### Kurento CrowdDetector

Remote stream

Roi ID: roi0

occupancyLevelMin: 100 10

occupancyLevelMed: 100 35

occupancyLevelMax: 100 65

occupancyNumFramesToEvent: 100 5

fluidityLevelMin: 100 10

fluidityLevelMed: 100 35

fluidityLevelMax: 100 65

fluidityNumFramesToEvent: 100 5

sendOpticalFlowEvent: Yes  No

opticalFlowFramesToEvent: 100 3

opticalFlowFramesToReset: 100 3

opticalFlowAngleOffset: 100 0

processingWidth: 1280 640

Video feed: rtsp://195.55.223.100/axis-media/media.amp

Start

Stop

+ Change Feed

Console

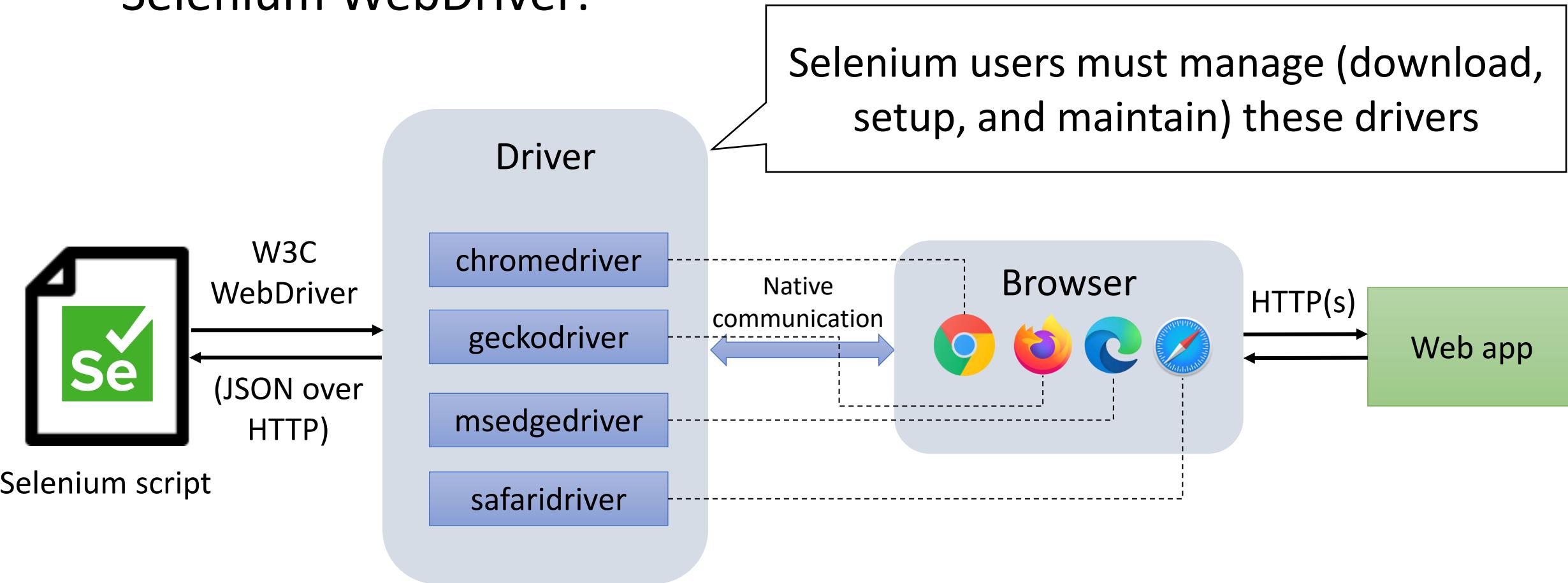
```
Starting video call ...
Creating WebRtcPeer and generating local sdp offer ...
constraints: {"mandatory": {"OfferToReceiveAudio": true, "OfferToReceiveVideo": true}, "optional": [{"DtlsSrtpKeyAgreement": true}]}
Created SDP offer
Local description set
```

© 2014 Kurento

NaevA tec

# My History with Selenium

- Selenium WebDriver:



# My History with Selenium

Fork me on GitHub

```
class ChromeManualTest {  
  
    WebDriver driver;  
  
    @BeforeAll  
    static void setupClass() {  
        System.setProperty("webdriver.chrome.driver", "/path/to/chromedriver");  
    }  
  
    @BeforeEach  
    void setup() {  
        driver = new ChromeDriver();  
    }  
  
    @Test  
    void test() {  
        driver.get("https://bonigarcia.dev/selenium-webdriver-java/");  
        String title = driver.getTitle();  
        assertThat(title).contains("Selenium WebDriver");  
    }  
  
    @AfterEach  
    void teardown() {  
        driver.quit();  
    }  
}
```

<https://github.com/bonigarcia/selenium-examples>

# My History with Selenium



CENTRO UNIVERSITARIO  
DE TECNOLOGÍA Y ARTE DIGITAL

2014-2018



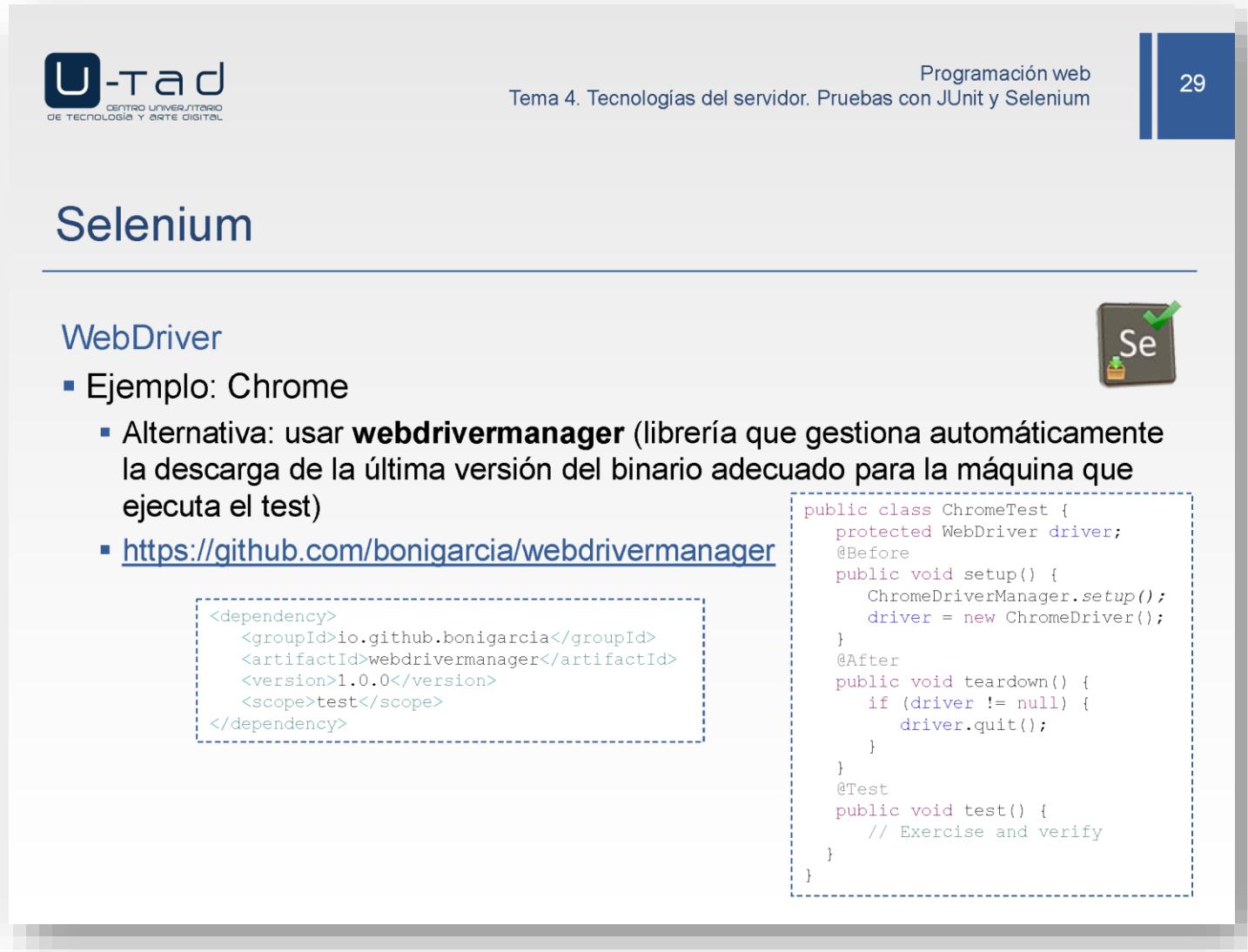
## Tema 4. Tecnologías del servidor. Pruebas con JUnit y Selenium

Programación web

Boni García  
Curso 2014/2015

# My History with Selenium

2014-2018



The screenshot shows a presentation slide from U-Tad. The slide title is "Selenium". It has a subtitle "WebDriver" and a bullet point "Ejemplo: Chrome". Below this is a section titled "Alternativa: usar webdrivermanager" with a link to "https://github.com/bonigarcia/webdrivermanager". To the right is a code snippet for a Java test class named "ChromeTest". The code uses the WebDriverManager library to set up a ChromeDriver instance. A small icon of a computer monitor with a checkmark is visible on the right.

U-tad  
CENTRO UNIVERSITARIO  
DE TECNOLOGÍA Y ARTE DIGITAL

Programación web  
Tema 4. Tecnologías del servidor. Pruebas con JUnit y Selenium

29

## Selenium

### WebDriver

- Ejemplo: Chrome
  - Alternativa: usar **webdrivermanager** (librería que gestiona automáticamente la descarga de la última versión del binario adecuado para la máquina que ejecuta el test)
  - <https://github.com/bonigarcia/webdrivermanager>

```
<dependency>
<groupId>io.github.bonigarcia</groupId>
<artifactId>webdrivermanager</artifactId>
<version>1.0.0</version>
<scope>test</scope>
</dependency>
```

```
public class ChromeTest {
protected WebDriver driver;
@Before
public void setup() {
    WebDriverManager.setup();
    driver = new ChromeDriver();
}
@After
public void teardown() {
    if (driver != null) {
        driver.quit();
    }
}
@Test
public void test() {
    // Exercise and verify
}
```

# My History with Selenium

Add [WebDriverManager](#) to your project:

33

```
<dependency>
    <groupId>io.github.bonigarcia</groupId>
    <artifactId>webdrivermanager</artifactId>
    <version>5.1.0</version>
</dependency>
```

This library downloads the latest version of the WebDriver binary you need and export the proper Java system variable (`webdriver.chrome.driver`, `webdriver.gecko.driver`, `webdriver.opera.driver`, `webdriver.edge.driver`, `webdriver.ie.driver`), simply using one of the following sentences respectively:

```
WebDriverManager.chromedriver().setup();
WebDriverManager.firefoxdriver().setup();
WebDriverManager.operadriver().setup();
WebDriverManager.edgedriver().setup();
WebDriverManager.iedriver().setup();
```

More info on <https://bonigarcia.dev/webdrivermanager/>

Share Edit Delete Flag    edited Feb 17, 2022 at 11:19    answered Apr 11, 2015 at 16:30  
Boni García  
4,471 ● 5 ● 26 ● 44

<https://stackoverflow.com/questions/7450416/selenium-2-chrome-driver/29580245>

# WebDriverManager

WebDriverManager 

<https://bonigarcia.dev/webdrivermanager/>

“ Automated driver management and other helper features for Selenium WebDriver in Java

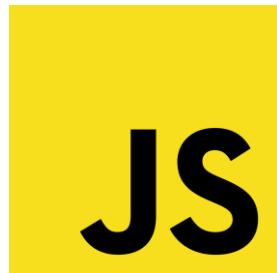
# WebDriverManager – Hello World Test

```
class ChromeWdmTest {  
  
    WebDriver driver;  
  
    @BeforeAll  
    static void setupClass() {  
        WebDriverManager.chromedriver().setup();  
    }  
  
    @BeforeEach  
    void setup() {  
        driver = new ChromeDriver();  
    }  
  
    @Test  
    void test() {  
        driver.get("https://bonigarcia.dev/selenium-webdriver-java/");  
        String title = driver.getTitle();  
        assertThat(title).contains("Selenium WebDriver");  
    }  
  
    @AfterEach  
    void teardown() {  
        driver.quit();  
    }  
}
```

Fork me on GitHub

# WebDriverManager – Other Managers

- The “*managers*” for Selenium WebDriver:



**webdriver-manager**

<https://www.npmjs.com/package/webdriver-manager>



**webdriver-manager**

<https://pypi.org/project/webdriver-manager>



**WebDriverManager.Net**

<https://github.com/rosolko/WebDriverManager.Net>



**webdrivers**

<https://github.com/titusfortner/webdrivers>

# WebDriverManager – SeleniumConf Tokyo 2019

## WebDriverManager - Design

- Currently, WebDriverManager resolution algorithm is much richer

```
graph TD; Se[WebDriverManager] -- "setup()" --> DR{Recently resolved?}; DR -- yes --> IP[Internal preferences]; IP -- TTL --> DR; DR -- no --> CBV[Check browser version]; CBV --> DV[Check driver version]; DV --> VD[Versions database]; VD --> DRA[Driver repository]; DRA --> DC[Driver cache]; DC --> ED[Export driver path]; ED --> IP;
```

The diagram illustrates the WebDriverManager resolution algorithm. It begins with the WebDriverManager object (Se) calling the `setup()` method. The process then follows these steps:

- If the driver was recently resolved (checked via `Internal preferences` with a `TTL`), it checks if the driver is in the internal preferences.
- If the driver is not in the internal preferences, it checks the browser and driver versions online, then checks the versions database.
- If the driver is not found in the versions database, it downloads the driver from the driver repository and stores it in the driver cache.
- Finally, it exports the driver path.

A video player interface at the bottom shows the presentation slide and a video frame of the speaker. The SeleniumConf Tokyo logo is visible in the bottom right corner.

Toolbox for Selenium Tests in Java: WebDriverManager and Selenium-Jupiter - Boni Garcia |SeConfTokyo

<https://www.youtube.com/watch?v=-bekd6QByC8>

# WebDriverManager – Automated Driver Management

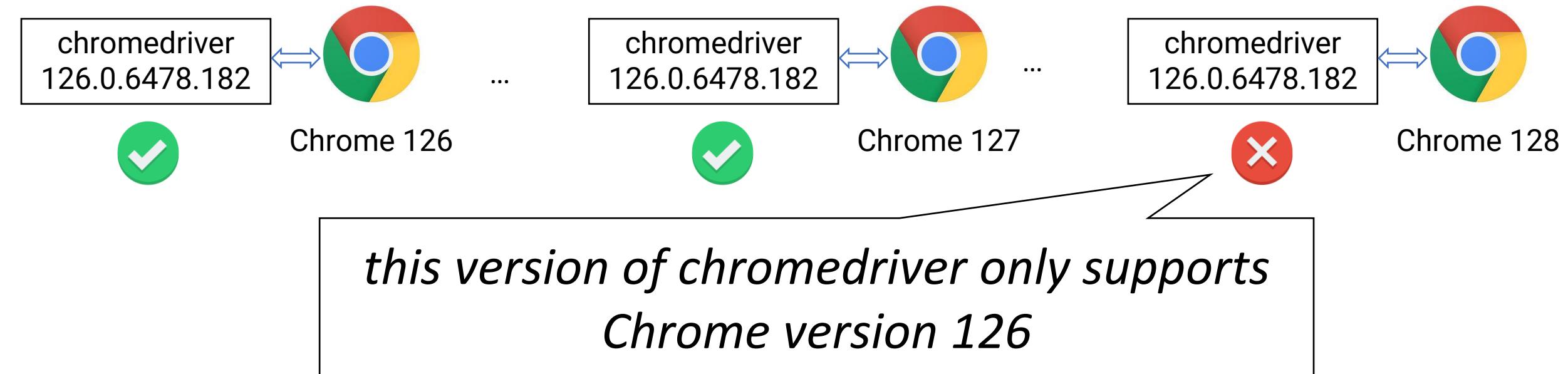
The screenshot shows a page from the journal 'Empirical Software Engineering'. The title of the page is 'Automated driver management for Selenium WebDriver'. It lists authors: Boni García<sup>1</sup>, Mario Muñoz-Orgaño<sup>1</sup>, Carlos Alario-Hoyos<sup>1</sup>, and Carlos Delgado Kloos<sup>1</sup>. The text indicates it was accepted on May 4, 2021, and published online on July 23, 2021. The copyright notice states: '© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021'. Below the title, there is an abstract section describing the Selenium WebDriver framework and the proposed WebDriverManager tool. The abstract notes that WebDriverManager provides a methodology for automating the management process, including execution methods like Java dependency, CLI tool, server, Docker container, and Java agent. It also mentions a survey of WebDriverManager users in 2020, showing widespread adoption and positive usability. The keywords listed are Test automation, Testing tools, and Selenium WebDriver. The communication details at the bottom mention Shin Yoo and provide email addresses for Boni García, Mario Muñoz-Orgaño, Carlos Alario-Hoyos, and Carlos Delgado Kloos.



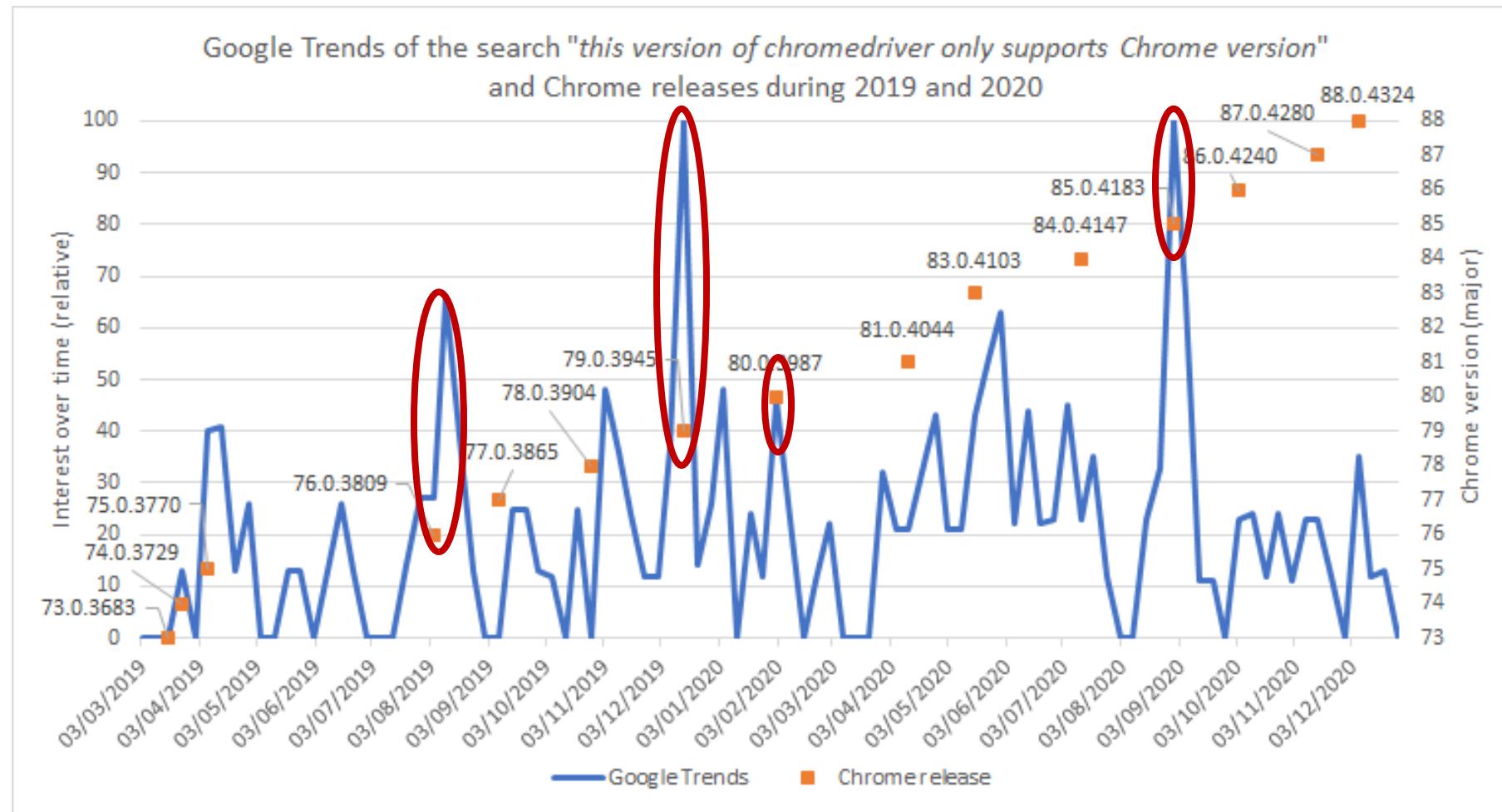
García, B., et al., 2021. Automated Driver Management for Selenium WebDriver.  
*Empirical Software Engineering*, 26(5), pp.1-51.

# WebDriverManager – Automated Driver Management

- Modern web browsers are *evergreen*



# WebDriverManager – Automated Driver Management



<https://trends.google.com/trends/>

# Driver Management in Selenium

- In 2020, the Selenium project surveyed its users:

The screenshot shows a blog post on the Selenium website. The title is "Results of the first ever selenium survey". The post summary states: "Summary of the Selenium survey that was collected". It was written by David Burns (@AutomatedTester) on Tuesday, January 12, 2021. The categories listed are general and survey. The tags listed are webdriver and survey. The main content discusses the survey results, specifically mentioning that 59.5% of respondents wanted Selenium to manage browsers. Below the main content, there are sections for "Batteries included" and "Browser Management". The sidebar on the right contains a search bar, a sidebar menu with links like "Batteries included", "Browser Management", "Frameworks", etc., and two lists: "Categories" and "Tags".

**Batteries included**

**Browser Management**

Unsurprisingly, people find having to manage browsers a task they wish they didn't have to do and wish that Selenium did this. 59.5% of respondents want Selenium to manage the browsers for them. This, though the question didn't ask this, is to include the browser drivers.

**Frameworks**

The results show an interesting view into framework usage. 61% of users use a framework. When we look closer at some of the responses there could be a little language bias in there. Some

Selenium users  
wanted *batteries*  
*included*

<https://www.selenium.dev/blog/2021/selenium-survey-results/>

# Selenium Manager

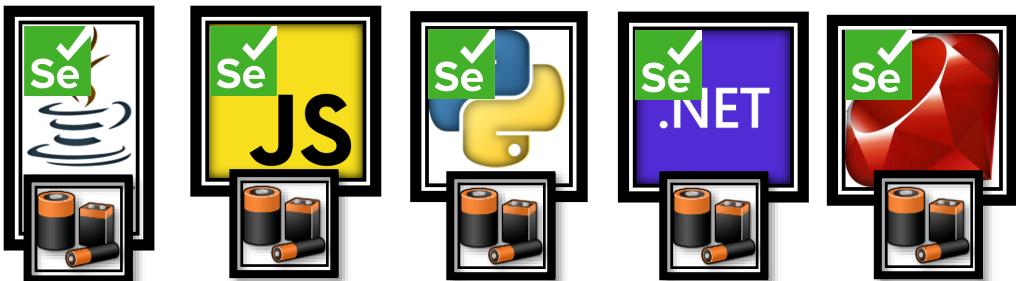


- I became a Selenium committer in August 2022



## Selenium Manager

- It is a CLI (Command-Line Interface) tool
- It has been developed in Rust
- It is shipped in each Selenium release



# Selenium Manager – Hello World Scripts

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

public class HelloWorldJava
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.get("https://www.selenium.dev/");
        driver.quit();
    }
}
```



```
using OpenQA.Selenium;
using OpenQA.Selenium.Chrome;

public class HelloWorldCSharp
    static void Main(string[] args) {
        IWebDriver driver = new ChromeDriver();
        driver.Navigate().GoToUrl("https://www.selenium.dev/");
        driver.Quit();
    }
}
```



```
const { Builder } = require('selenium-webdriver');

async function helloWorldJavaScript() {
    let driver = await new Builder().forBrowser('chrome').build();
    await driver.get('https://www.selenium.dev/');
    await driver.quit();
}
```



```
from selenium import webdriver

def hello_world_python():
    driver = webdriver.Chrome()
    driver.get("https://www.selenium.dev/")
    driver.quit()
```



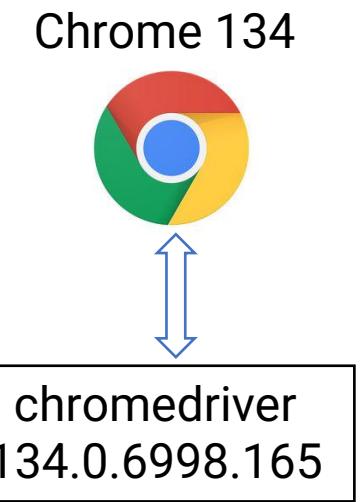
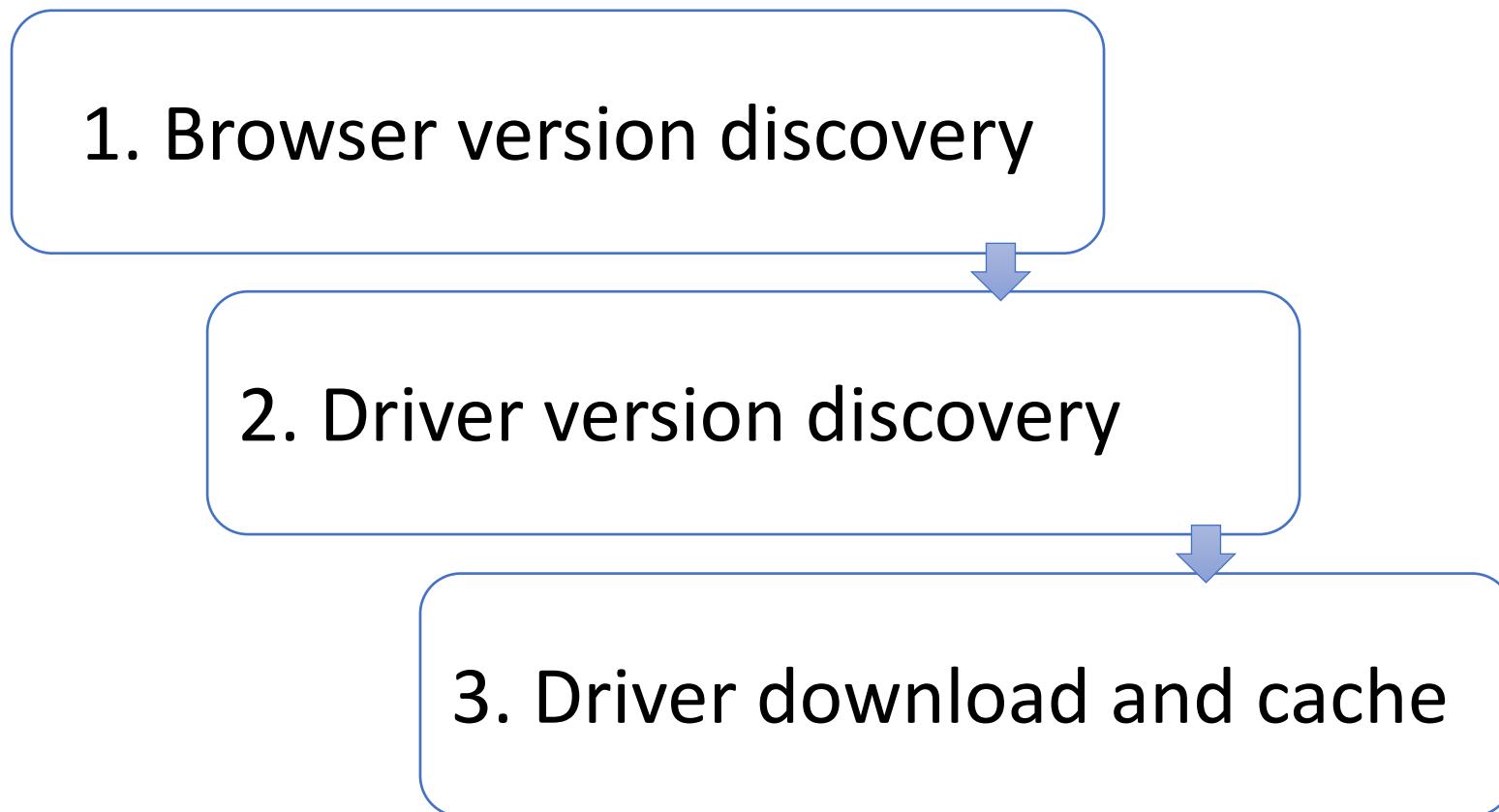
```
require 'selenium-webdriver'

def hello_world_ruby
    driver = Selenium::WebDriver.for :chrome
    driver.navigate.to 'https://www.selenium.dev/'
    driver.quit
end
```



# Selenium Manager – Automated Driver Management

- Selenium Manager automatically discovers, downloads, and caches the drivers required by Selenium



# Selenium Manager – Hello World Test

```
class FirefoxBasicTest {  
  
    WebDriver driver;  
  
    @BeforeEach  
    void setup() {  
        driver = new FirefoxDriver();  
    }  
  
    @Test  
    void test() {  
        driver.get("https://bonigarcia.dev/selenium-webdriver-java/");  
        String title = driver.getTitle();  
        assertThat(title).contains("Selenium WebDriver");  
    }  
  
    @AfterEach  
    void teardown() {  
        driver.quit();  
    }  
}
```

What happens if Firefox is not available the machine running this test?

# Selenium Manager – Automated Browser Management

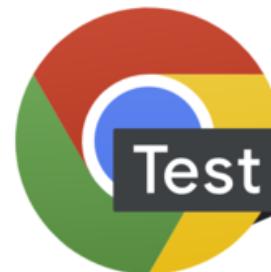
- Selenium Manager automatically discovers, downloads, and caches the browsers driven with Selenium when these browsers are not installed in the local system

	 Test		
			
			
			

\* Requires admin permissions

# Selenium Manager – Automated Browser Management

- **Chrome for Testing (CfT)** is a specialized version of Google Chrome designed specifically for browser automation
  - It is not evergreen (no auto-updates)
  - It is lighter than regular Google Chrome (e.g., user-related components, such as the user profile or password manager are not available in CfT)



<https://googlechromelabs.github.io/chrome-for-testing/>

# Selenium Manager – Automated Browser Management

```
class ChromeVersionTest {  
  
    WebDriver driver;  
  
    @BeforeEach  
    void setup() {  
        ChromeOptions options = new ChromeOptions();  
        options.setBrowserVersion("beta");  
        driver = new ChromeDriver(options);  
    }  
  
    @Test  
    void test() {  
        driver.get("https://bonigarcia.dev/selenium-webdriver-java/");  
        String title = driver.getTitle();  
        assertThat(title).contains("Selenium WebDriver");  
    }  
  
    @AfterEach  
    void teardown() {  
        driver.quit();  
    }  
}
```

Specific browser versions (including "beta", "dev", or "nightly") are supported

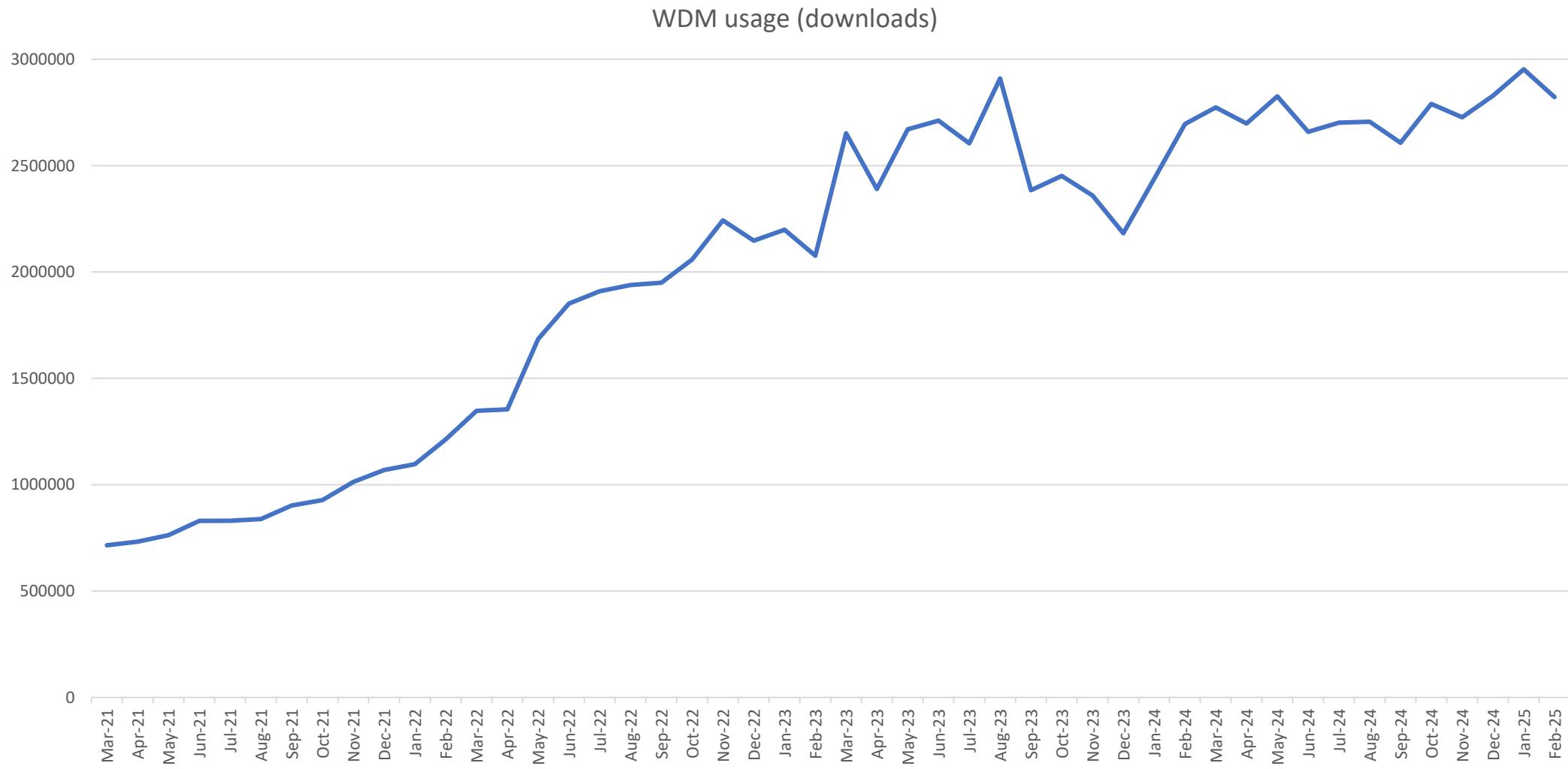
Fork me on GitHub

# Selenium Manager – Other Uses

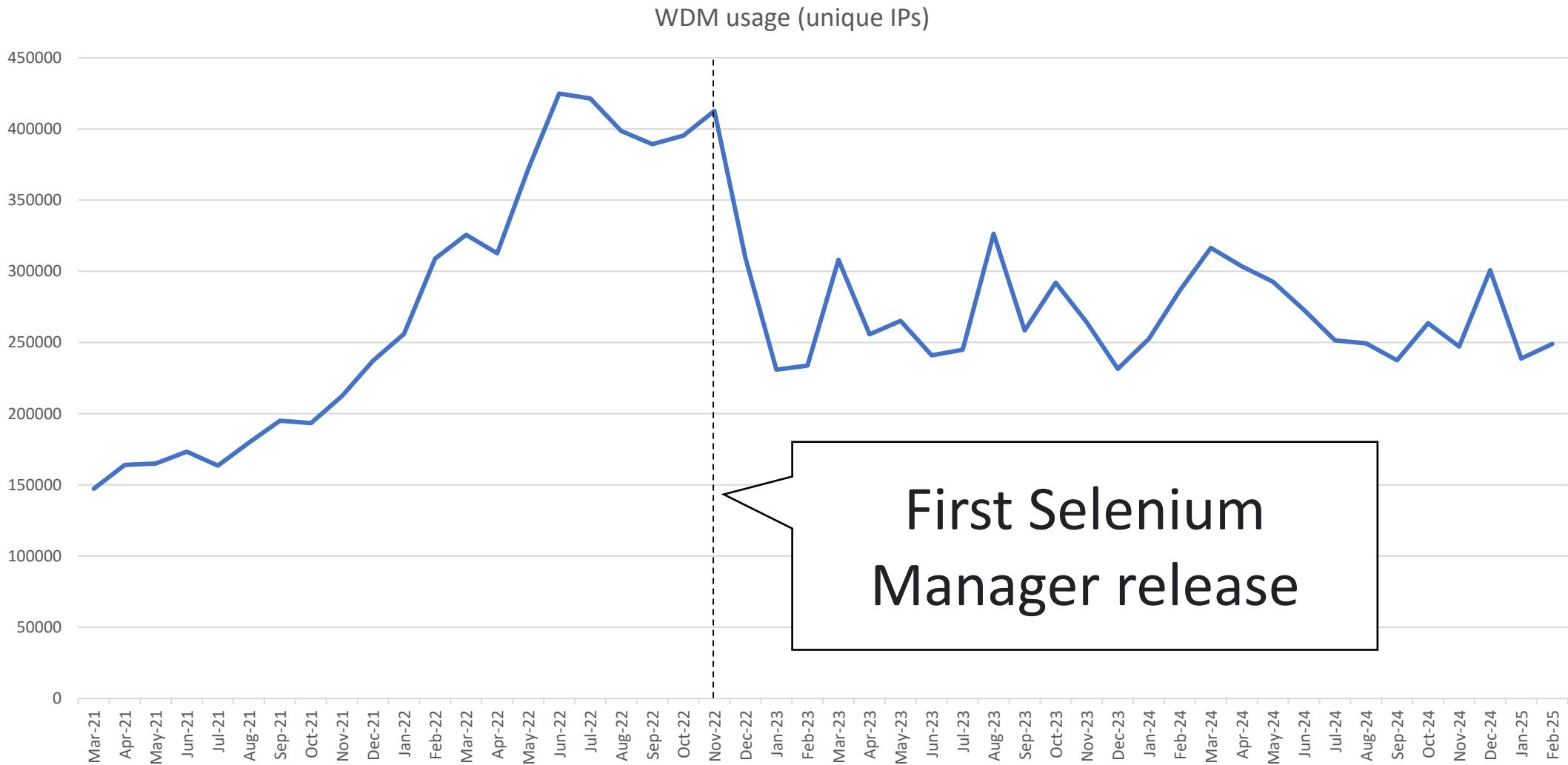
- Another uses of Selenium Manager include:
    - Advanced configuration (with envs or global configuration file)
    - As a standalone CLI tool

[https://www.selenium.dev/documentation/selenium\\_manager/](https://www.selenium.dev/documentation/selenium_manager/)

# What about WebDriverManager?



# What about WebDriverManager?



# WebDriverManager is 10 years



 WebDriverManager 

# WebDriverManager – Automated Browser Management

```
class DockerChromeTest {  
  
    WebDriver driver;  
    WebDriverManager wdm;  
  
    @BeforeEach  
    void setupTest() {  
        wdm = WebDriverManager.chromedriver().browserInDocker();  
        driver = wdm.create();  
    }  
  
    @Test  
    void test() {  
        driver.get("https://bonigarcia.dev/selenium-webdriver-java/");  
        assertThat(driver.getTitle()).contains("Selenium WebDriver");  
    }  
  
    @AfterEach  
    void teardown() {  
        wdm.quit();  
    }  
}
```



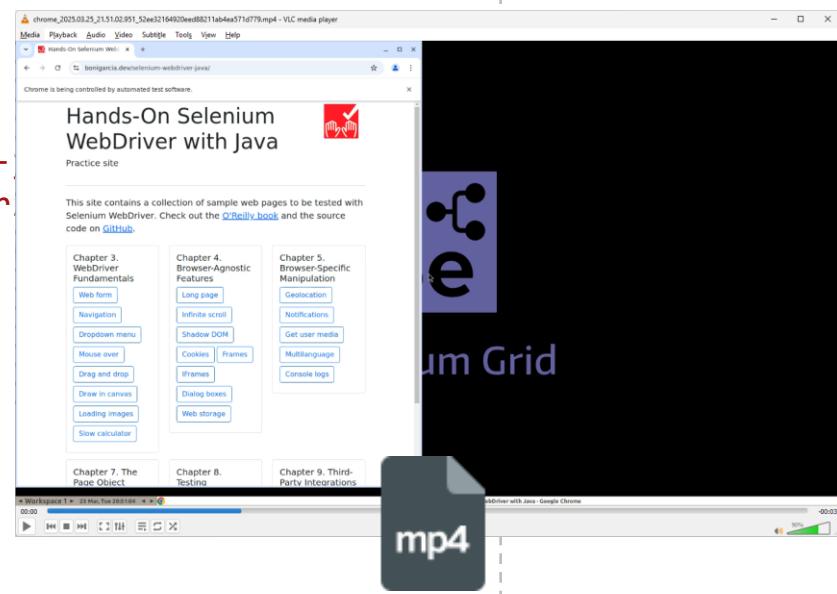
<https://github.com/SeleniumHQ/docker-selenium>

Fork me on GitHub

Fork me on GitHub

# WebDriverManager – Video Recording (I)

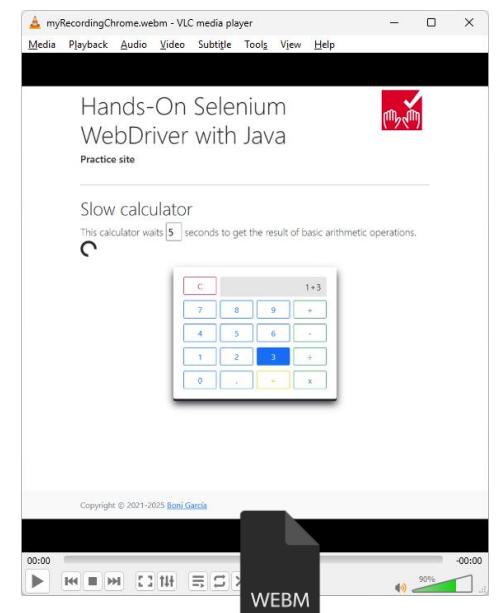
```
class DockerChromeRecordingTest {  
  
    WebDriver driver;  
    WebDriverManager wdm;  
  
    @BeforeEach  
    void setupTest() {  
        wdm = WebDriverManager.chromedriver().browserInDocker().enableRecording();  
        driver = wdm.create();  
    }  
  
    @Test  
    void test() {  
        driver.get("https://bonigarcia.dev/selenium-webdriver-  
        assertThat(driver.getTitle()).contains("Selenium WebDr  
    }  
  
    @AfterEach  
    void teardown() {  
        wdm.quit();  
    }  
}
```



Fork me on GitHub

# WebDriverManager – Video Recording (II)

```
class RecordEdgeTest {  
  
    WebDriver driver;  
    File targetFolder;  
    WebDriverManager wdm;  
  
    @BeforeEach  
    void setup() {  
        wdm = WebDriverManager.edgedriver().watch();  
        driver = wdm.create();  
    }  
  
    @Test  
    void test() {  
        driver.get(  
            "https://bonigarcia.dev/selenium-webdriver-java/slow-calculator.html");  
        wdm.startRecording(),  
        // test logic  
        wdm.stopRecording();  
    }  
  
    @AfterEach  
    void teardown() {  
        driver.quit();  
    }  
}
```



**BrowserWatcher**   
<https://bonigarcia.dev/browserwatcher/>

Fork me on GitHub

# WebDriverManager – Console Logs Gathering

```
class GatherLogsFirefoxTest {  
  
    WebDriverManager wdm;  
    WebDriver driver;  
  
    @BeforeEach  
    void setup() {  
        wdm = WebDriverManager.firefoxdriver().watch();  
        driver = wdm.create();  
    }  
  
    @Test  
    void test() {  
        driver.get(  
            "https://bonigarcia.dev/selenium-webdriver-java/console-logs.html");  
        List<Map<String, Object>> logMessages = wdm.getLogs();  
  
        // handle logs  
    }  
  
    @AfterEach  
    void teardown() {  
        driver.quit();  
    }  
}
```

More info and examples on:  
<https://bonigarcia.dev/webdrivermanager/>

# WebDriverManager vs Selenium Manager

- Is Selenium Manager a replacement for WebDriverManager?
  - For the sole use case of automated driver management, yes
- What are the differences between both tools?

WebDriverManager	Selenium Manager
Java library	Rust CLI application – used by default in all bindings languages (Java, JavaScript, Python, Ruby, .Net) since Selenium 4.6
Automated driver management <ul style="list-style-type: none"><li>• Based on Docker</li></ul>	Automated browser management <ul style="list-style-type: none"><li>• Based on browser binary releases</li></ul>
Other features: <ul style="list-style-type: none"><li>• Video recording</li><li>• Monitoring</li></ul>	Other features: <ul style="list-style-type: none"><li>• Standalone CLI tool</li></ul>

# What is next?

- The development and maintenance of WebDriverManager and Selenium Manager continues
- The backlog of WebDriverManager is currently empty
- There are some open issues in Selenium Manager, such as:
  - Digitally sign selenium-manager.exe (for Windows)
  - Replace WMIC commands (deprecated) by PowerShell (for Windows)
  - Support Electron
- Selenium Manager will be out of beta for Selenium 5

# Test Automation with Selenium: A Survey



Please share your  
experience about  
Selenium here!

# WebDriverManager vs Selenium Manager



Participate in the  
survey here

Thank you very much!

Boni García  
Universidad Carlos III de Madrid, Spain  
[boni.garcia@uc3m.es](mailto:boni.garcia@uc3m.es)

Get these slides on:



<https://bonigarcia.dev/>

