Selenium

by Igor Drobiazko

Agenda

1. About me
2. Selenium in Action
3. Overview of the Selenium architecture
4. Integration into a Continuous Integration process
5. Testing AJAX apps

About Me

- Software developer at HSBC INKA
- Apache Tapestry Committer
- Book author
- Speaker
- drobiazko at apache dot org

Tapestry 5 Book

- The only one german book about Tapestry 5
- ISBN: 978-3-8273-2844-1
- EUR 29,95
- Available as eBook

How to find bugs? This way?

Why Selenium?

- Open Source
- Tests can be written in: HTML, Java, Ruby, Python, Perl, PHP, C#
- Tests are exeuted in a browser
- Selenium extensions
- Powerful and easy to learn

Demo 1: Selenium in Action

What is Selenium?

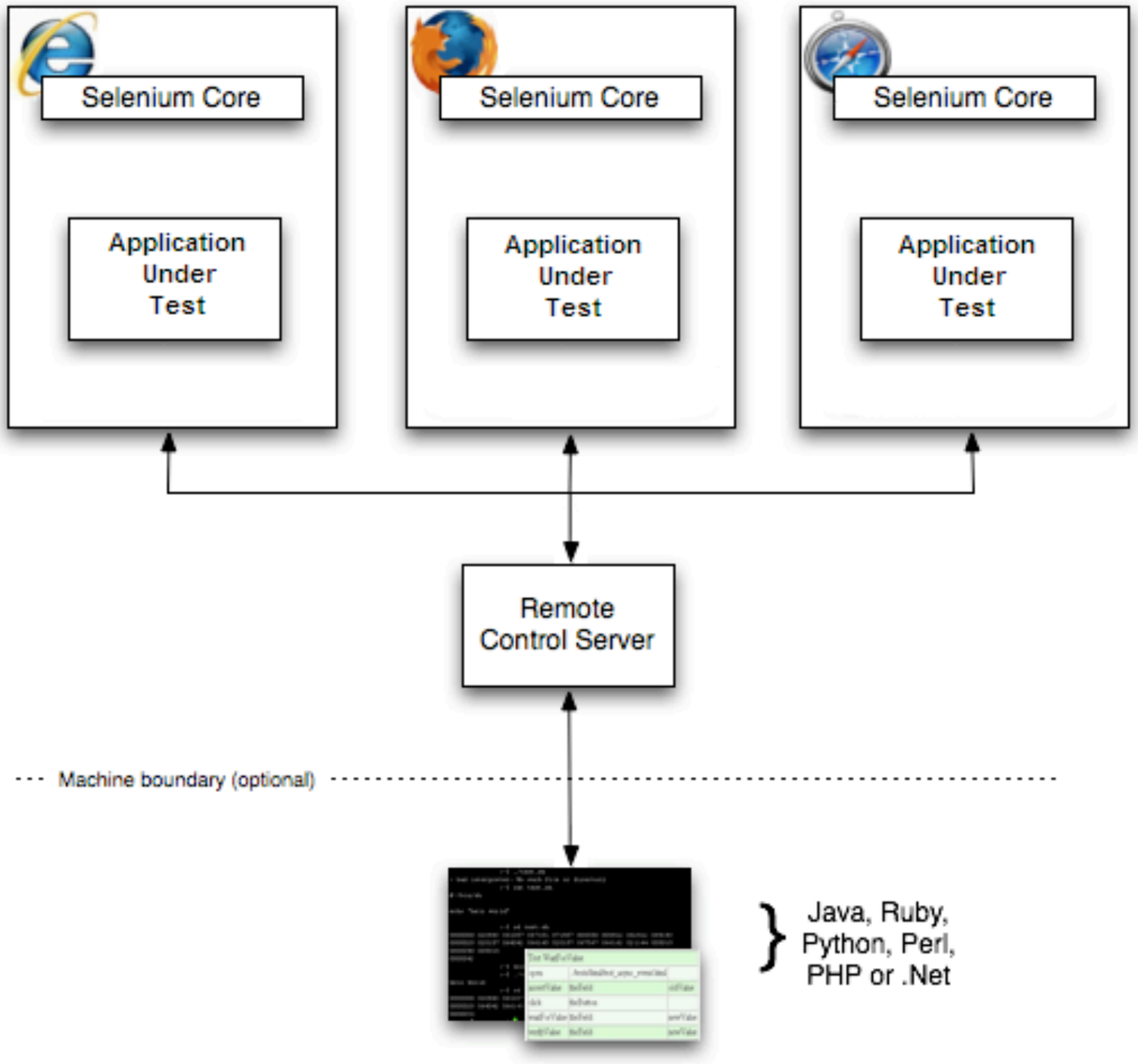
- Selenium-IDE
  - Firefox Add-on
    - The most easy way to start
- Selenium-RC (Remote Control)
  - Selenium Core + Client libraries
- Selenium-Grid

Demo 2: Selenium IDE

Selenium-RC

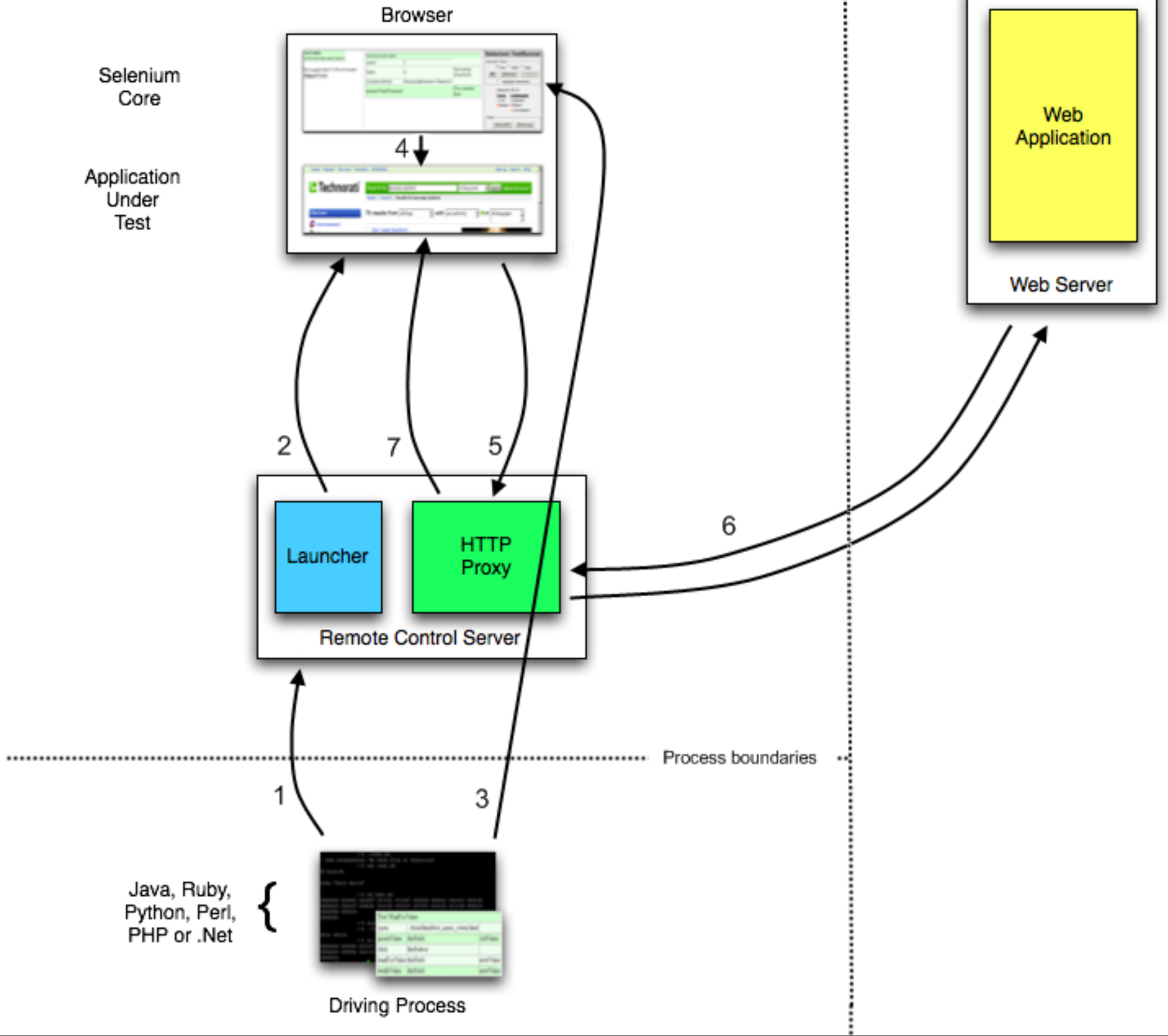
- Client libraries
  - Provides API for running Selenese commands from a test programm
  - Passes the commands to Selenium Server for execution
  - Receives the results
- Selenium Server
  - Starts/stops the browser
  - injects Selenium Core into the browser
  - acts as HTTP proxy

Windows, Linux, or Mac (as appropriate)...



Selenium Core

- Set of JavaScript functions to interpret and execute Selenese commands
- Problem: Same Origin Policy and Cross-site scripting
- Solution: Selenium Server is a proxy between the browser and application under test


Browser
Selenium Core
Application Under Test
4
Web Application
Web Server
2
7
5
6
Launcher
HTTP Proxy
Remote Control Server
Process boundaries
1
3
Java, Ruby, Python, Perl, PHP or .Net
Driving Process

Starting Selenium Server

- Programmatically
- Maven Plugin

Selenium & Maven

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
    http://maven.apache.org/maven-v4_0_0.xsd">
    ...

    <build>
        <plugins>
            <plugin>
                <groupId>org.codehaus.mojo</groupId>
                <artifactId>selenium-maven-plugin</artifactId>
                <executions>
                    <execution>
                        <phase>pre-integration-test</phase>
                        <goals>
                            <goal>start-server</goal>
                        </goals>
                        <configuration>
                            <background>true</background>
                        </configuration>
                    </execution>
                </executions>
            </plugin>
        </plugins>
        ...
    </build>
    ...
</project>
```

Simple Selenium Test

```
public class SimpleSeleniumTest{

    @Test
    public void testSimple() {

        DefaultSelenium selenium = new DefaultSelenium(
                "localhost", 4444, "*firefox", "http://localhost:8080/");

        selenium.start();

        selenium.open("http://localhost:8080/");

        assertEquals("Page Title", selenium.getTitle());

        assertTrue(selenium.isTextPresent("Hello, World!"));

        selenium.stop();
    }
}
```

Selenese

- Set of Selenium Commands
- Actions: Manipulate the state of the application
- Accessors: Examine the state of the application and store the results in variables
- Assertions: Verify the state of the application

Selenese: Actions

- click(locator)
- open(url)
- type(locator, value)
- check(locator)
- waitForPageToLoad(timeout)
- and many more

Selenese: Accessors

- storeText(locator, variableName)
- storeValue(locator, variableName)
- storeChecked(locator, variableName)
- storeAlertPresent(variableName)
- many more

Selenese: Locators

- By identifier
- By element name
- By CSS (Cascading Style Sheets)
- By XPath
- By DOM (Document Object Model)
- Links can be located by their text

Example:Locators

```
<html>
  <body>
    <form id= “loginForm“>
      <input name=“username“ type= “text“/>
    </form>
    <a href= “weiter.html“>Weiter</a>
  </body>
</html>
```

```
identifier=loginForm
identifier=username
id=loginForm
name=username
xpath=//form[@id=‘loginForm‘]
link=Weiter
dom=document.forms[0]
```

Selenese: Assertions vs. Verifications

- assert: fail the test and abort the test case
- verify: fals the test but continue to run the test case

Demo 3: Writing Selenium Tests

Testing AJAX Apps

Testing AJAX Apps

- AJAX enabled applications retrieve data from server asynchronously
- Problem: Selenium is unaware of the AJAX response
- Naive Solution: Pause the test with Thread.sleep();
- Better Solution: selenium.waitForCondition()

Simple AJAX Test

```
public class SimpleSeleniumTest{

    @Test
    public void testAjax() {

        DefaultSelenium selenium = new DefaultSelenium(
                "localhost", 4444, "*firefox", "http://localhost:8080/");

        selenium.start();

        selenium.open("http://localhost:8080/");

        selenium.click("myLink");

        selenium.waitForCondition(
           "selenium.browserbot.getCurrentWindow().$$(\"myCssClass\").size() > 0",
           15000);

        selenium.stop();
    }
}
```

Demo 4: Testing AJAX Apps

Thank you!