INGENIERÍA WEB Y COMPUTACIÓN EN LA NUBE

Bloque3: Parte servidora (backend)

TEMA 3.6: PRUEBAS CON JUNIT Y SELENIUM

Boni García
boni.garcia@urjc.es

Índice de contenidos

Índice de contenidos

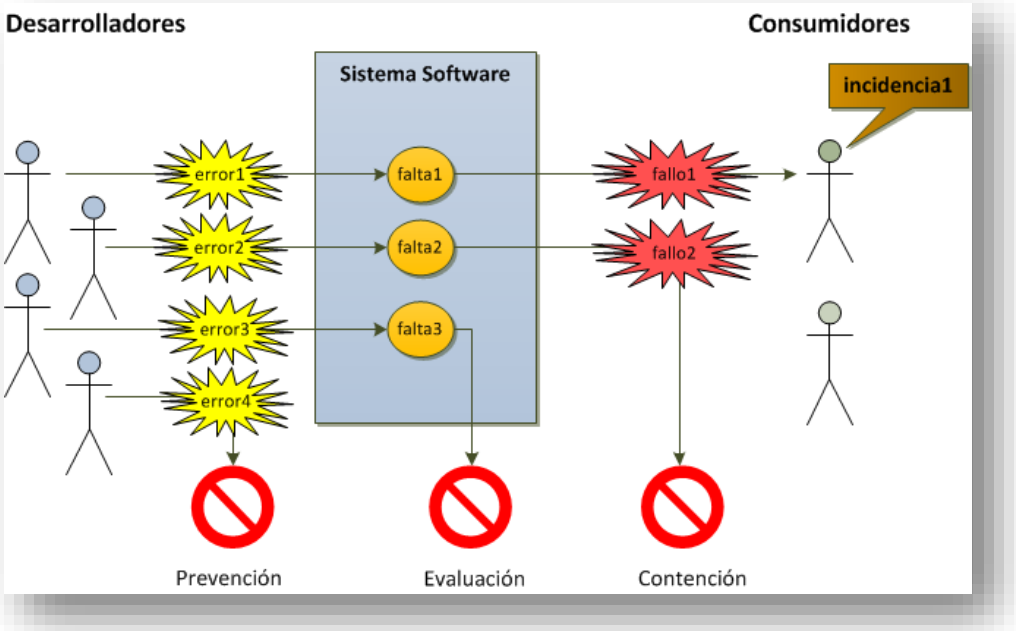
1. Introducción

Evaluación en el software

- La **evaluación en el software** (también conocido como control de calidad o verificación y validación, V&V) es la fase del ciclo de vida destinada a:
  - Evaluar la **calidad** del software asegurando que el producto desarrollado cumple los **requisitos** funcionales y no funcionales establecidos (especificación) y las expectativas del consumidor (cliente/usuario)
  - Identificar los posibles **defectos** (conocidos de forma genérica como ***bugs***)
- No hay un consenso claro del alcance de verificación y validación, por eso se suele agrupar bajo el termino V&V. La definición clásica es:
  - Verificación: *Hacer bien las cosas* (cumplir la especificación)
  - Validación: *Hacer las cosas bien* (cumplir expectativas)

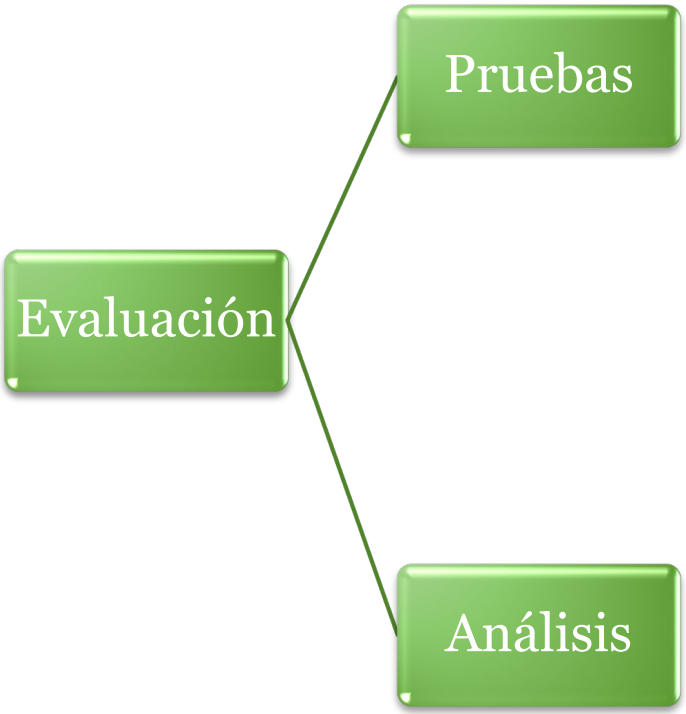
1. Introducción

Taxonomía de defectos



1. Introducción

Tipos de evaluación



- **Dinámico**: implica la ejecución del sistema bajo pruebas
- Niveles: unitario, integración, sistema…
- Técnicas: caja negra, caja blanca, no funcionales…

- **Estático**: implica la evaluación del código fuente
- Técnicas: inspección, revisión, métodos formales, análisis automático…

1. Introducción

Automatización de pruebas

- Las pruebas manuales es un proceso muy **costoso**
- La **automatización** de pruebas ayuda a reducir dichos esfuerzos y se define como:

  "*La aplicación e implementación de tecnología software durante todo el ciclo de pruebas para mejorar la eficacia y eficiencia del mismo*"

- La automatización de pruebas es más efectiva cuando está implementado por un **framework**:
  - Open-source: **JUnit**, **Selenium**, JMeter…
  - Comerciales: HP Unified Functional Testing, IBM Rational Functional Tester…
- Las pruebas automáticas típicamente se ejecutan en un proceso de **integración continúa**

Índice de contenidos

2. JUnit

- JUnit es un framework de pruebas **unitarias** para **Java**
- JUnit ha sido portado a otros lenguajes (familia xUnit): .NET (NUnit), Python (PyUnit), JavaScript (Qunit), …
- Licencia CPL (*Common Public License*)
- Versión actual: 4.12
  - Dependencia Maven:

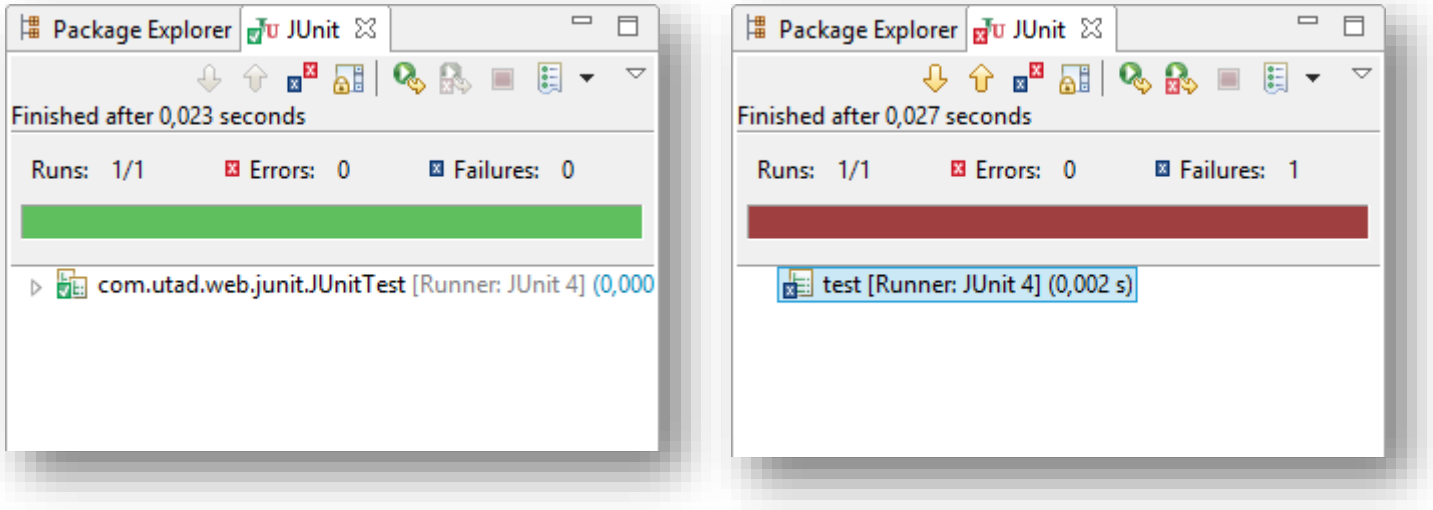
```
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.12</version>
    <scope>test</scope>
</dependency>
```

JUnit

http://junit.org/

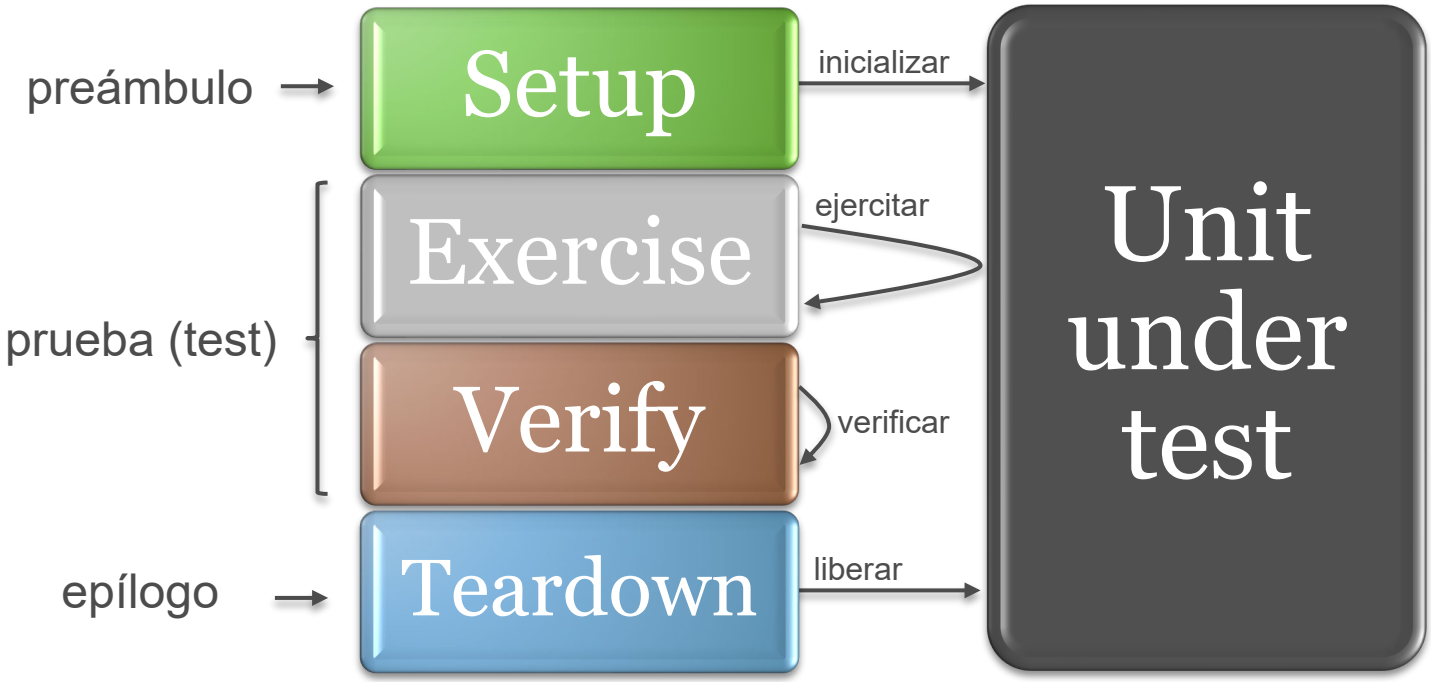
2. JUnit

- JUnit nos permite automatizar la ejecución de una **unidad** software y dar un veredicto sobre la prueba
- JUnit está integrado en IDEs como Eclipse:



2. JUnit

- Un caso de prueba unitario está compuesto por cuatro fases: setup, exercise, verify, teardown:



2. JUnit

- JUnit 4.x:

```
import org.junit.*;

public class JUnitTest {

	@BeforeClass
	public static void setupClass() {
		// Initialization per test case
	}
	@Before
	public void setupTest() {
		// Initialization per test
	}

	@Test
	public void test1() {
		// Exercise and verify
	}
	@Test
	public void test2() {
		// Exercise and verify
	}

	@After
	public void teardownTest() {
		// Finish per test
	}
	@AfterClass
	public static void teardownClass() {
		// Finish per test
	}
}
```

2. JUnit

- Las verificaciones (aserciones o predicados) se hacen en JUnit mediante la clase `Assert`
- Algunos ejemplos de aserciones en JUnit 4.x:

```
import org.junit.Assert;


Assert.assertTrue("The condition is not met", booleanCondition);
Assert.assertFalse("The condition is met", booleanCondition);

Assert.assertArrayEquals("The array is not equal", array1, array2);

Assert.assertNull("The object is null", object1);
Assert.assertNotNull("The object is not null", object2);

Assert.fail("Test failure");
```

2. JUnit

- Ejemplo: pruebas unitarias para la clase `ArrayList` de Java:

```
public class ArrayListTest {

   private List<String> list;
   private final String[] data =
    { "data1", "data2", "data3" };

   @Before
   public void setupTest() {
      list = new ArrayList<String>();
      for (String s : data) {
         list.add(s);
      }
   }


   @After
   public void teardownTest() {
      list.clear();
   }
```

```
   @Test
   public void testContet() {
      for (int i = 0; i < data.length; i++) {
         // Exercise
         String expectedContent = data[i];
         String realContent = list.get(i);

         // Verify
         Assert.assertEquals("Element at position " + i + " should be "
            + expectedContent + " and is " + realContent,
            expectedContent, realContent);
      }
   }

   @Test
   public void testSize() {
      // Exercise
      int expectedSize = data.length;
      int realSize = list.size();

      // Verify
      Assert.assertTrue("List size should be " + expectedSize
         + " and is " + realSize, realSize == expectedSize);
   }
}
```

Índice de contenidos

3. Selenium

- **Selenium** es un **framework** que permite la **automatización** de **pruebas** para aplicaciones **web**
- Licencia Apache 2.0
- Diseñado inicialmente en 2004 por Jason Huggins
- El nombre fue elegido como burla de la herramienta comercial de pruebas Mercury (actualmente HP Unified Functional Testing)

"Selenium is a key mineral which protects the body from Mercury toxicity"

http://www.seleniumhq.org/

3. Selenium

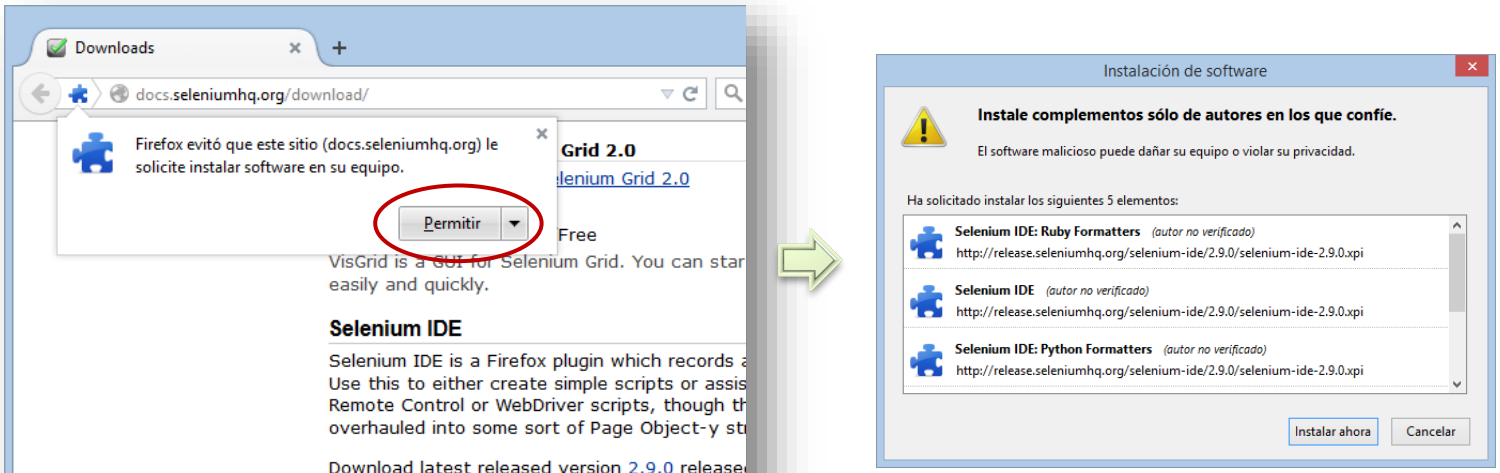
- Selenium está formado por tres componentes:

Proyecto	Descripción
Selenium IDE	Plugin Firefox que permite grabación y reproducción de aplicaciones web
Selenium WebDriver	Control programático de ejecución web con navegadores locales. Sucesor del deprecado Selenium Remote Control (RC)
Selenium Grid	Permite ejecutar Selenium WebDriver en máquinas remotas

3. Selenium

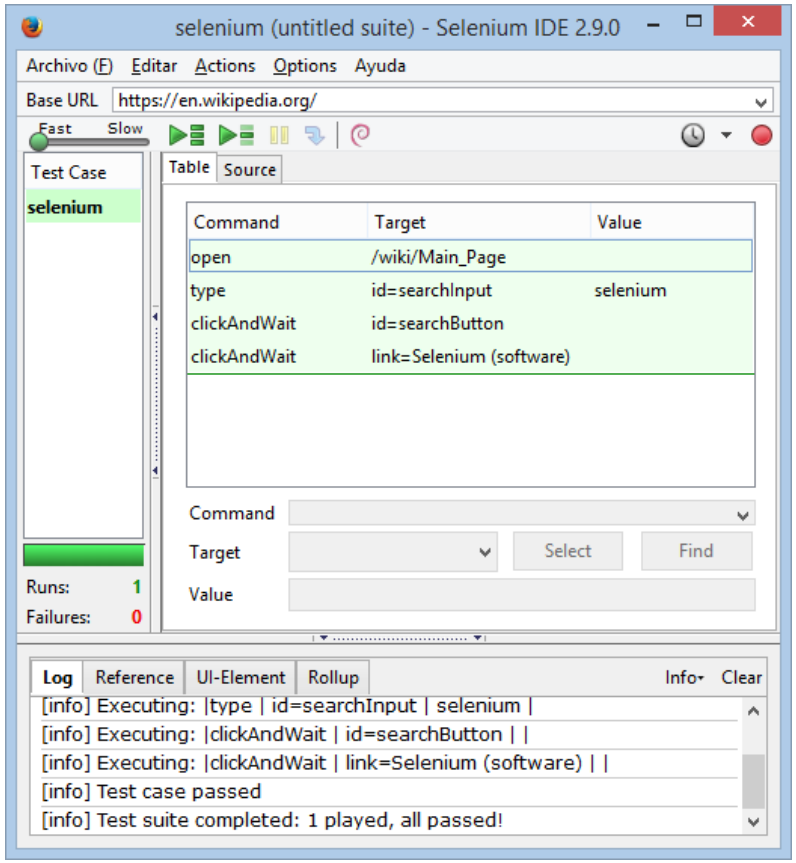
Selenium IDE

- Es un plugin de Firefox que permite **grabar** y **reproducir** interacciones con aplicaciones web
- Versión actual (noviembre de 2015): 2.9.0
  - Se puede instalar en Firefox desde http://docs.seleniumhq.org/download/



3. Selenium

Selenium IDE




```
<link rel="selenium.base" href="https://en.wikipedia.org/" />
<title>selenium</title>
</head>
<body>
<table cellpadding="1" cellspacing="1" border="1">
   <thead>
      <tr><td rowspan="1" colspan="3">selenium</td></tr>
   </thead><tbody>
   <tr>
      <td>open</td>
      <td>/wiki/Main_Page</td>
      <td></td>
   </tr>
   <tr>
      <td>type</td>
      <td>id=searchInput</td>
      <td>selenium</td>
   </tr>
   <tr>
      <td>clickAndWait</td>
      <td>id=searchButton</td>
      <td></td>
    </tr>
    <tr>
       <td>clickAndWait</td>
       <td>link=Selenium (software)</td>
       <td></td>
    </tr>
    </tbody></table>
</body>
</html>
```

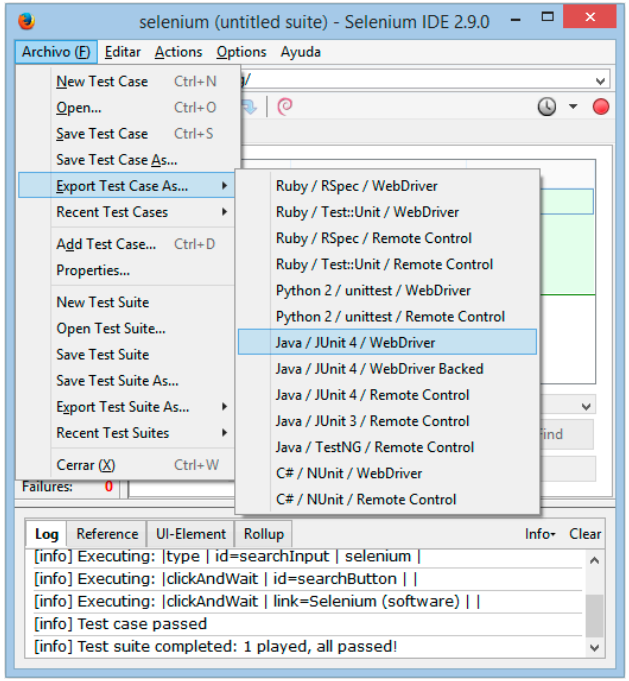
selenium

selenium		
open	/wiki/Main_Page	
type	id=searchInput	selenium
clickAndWait	id=searchButton	
clickAndWait	link=Selenium (software)	

3. Selenium

Selenium IDE

- Permite además exportar la grabación a diferentes lenguajes (Ruby, Python, Java, C#) para ser manejado con WebDriver:



3. Selenium

WebDriver

- Selenium WebDriver permite manejar un navegador web programáticamente
- Compatibilidad:
  - Navegadores: Chrome, Firefox, Internet Explorer, Opera, Safari, Edge
  - Navegadores móviles: Android, iOS, Windows Phone
  - Navegadores "*headless*": HtmlUnit, PhantomJS
  - Sistemas operativos: Windows, Linux, Mac OS X
  - Lenguajes: C#, Haskell, Java, JavaScript, Objective-C, Perl, PHP, Python, R, Ruby

http://docs.seleniumhq.org/projects/webdriver/

3. Selenium

WebDriver

- WebDriver maneja de forma nativa los navegadores, por lo que necesita un programa binario que comunica la API de WebDriver y el navegador:
  - Firefox: No necesita de binario intermediario ya que se instala automáticamente una extensión en Firefox (XPI)
  - Safari: No necesita binario intermediario pero hay que instalar manualmente una extensión
  - Chrome: https://sites.google.com/a/chromium.org/chromedriver/
  - Opera: http://choice.opera.com/developer/tools/operadriver/
  - Internet Explorer: https://code.google.com/p/selenium/wiki/InternetExplorerDriver (y además hay que cambiar la configuración de seguridad)
  - Microsoft Edge: https://www.microsoft.com/en-us/download/details.aspx?id=48212

3. Selenium

WebDriver

- Estructura típica de un caso de prueba Java con Selenium WebDriver:

1. Importar la dependencia de `selenium-java` en nuestro proyecto
2. Instanciar un objeto `WebDriver` (para Chrome, Firefox, etc)
3. Abrir una página web (URL)
4. Localizar elementos (`WebElement`)
5. Interactuar con elementos (hacer click, leer atributos, etc)
6. Verificar que la web bajo pruebas cumple las condiciones esperadas (aserciones)

3. Selenium

WebDriver

- Estructura típica de un caso de prueba con Selenium WebDriver:

1. Importar la dependencia de `selenium-java` en nuestro proyecto:

```
<dependency>
   <groupId>org.seleniumhq.selenium</groupId>
   <artifactId>selenium-java</artifactId>
   <version>2.48.2</version>
   <scope>test</scope>
</dependency>
```

2. Instanciar un objeto `WebDriver` (para Chrome, Firefox, etc):

```
WebDriver driver = new FirefoxDriver();
WebDriver driver = new ChromeDriver();
WebDriver driver = new OperaDriver();
WebDriver driver = new InternetExplorerDriver();
WebDriver driver = new SafariDriver();
```

3. Abrir una página web (URL):

```
driver.get("http://en.wikipedia.org/wiki/Main_Page");
```

3. Selenium

WebDriver

4. Localizar elementos (`WebElement`)

```
// Locate single element
WebElement webElement1 = driver.findElement(By.id("id"));
WebElement webElement2 = driver.findElement(By.name("name"));
WebElement webElement3 = driver.findElement(By.className("class"));
WebElement webElement4 = driver.findElement(By.cssSelector("cssInput"));
WebElement webElement5 = driver.findElement(By.linkText("text"));
WebElement webElement6 = driver.findElement(By.partialLinkText("partial text"));
WebElement webElement7 = driver.findElement(By.tagName("tag name"));
WebElement webElement8 = driver.findElement(By.xpath("/html/body/div[4]"));

// Locate single element list
List<WebElement> webElements = driver.findElements(By...);
```

- XPath (*XML Path Language*) es un lenguaje que permite construir expresiones que recorren y procesan un documento XML (como HTML):

```
/html/body/form[1]
//form[@id='loginForm']
//input[@name='username']
//form[@id='loginForm']/input[1]
```

Más info: http://www.w3schools.com/xpath/

3. Selenium

WebDriver

- Estructura típica de un caso de prueba con Selenium WebDriver:

5. Interactuar con elementos (hacer click, leer atributos, etc). Por ejemplo:

```
webElement1.click();
webElement1.clear();
webElement1.sendKeys("text");

String text = webElement1.getText();
String href = webElement1.getAttribute("href");
String css = webElement1.getCssValue("css");
Dimension dim = webElement1.getSize();

boolean enabled = webElement1.isEnabled();
boolean selected = webElement1.isSelected();
boolean displayed = webElement1.isDisplayed();
```

3. Selenium

WebDriver

- Estructura típica de un caso de prueba con Selenium WebDriver:

6. Verificar que la web bajo pruebas cumple las condiciones esperadas (predicados). Por ejemplo:

```
WebDriverWait wait = new WebDriverWait(driver, 30); // seconds

wait.until(ExpectedConditions.elementToBeClickable(By.id("id1")));
wait.until(ExpectedConditions.elementToBeSelected(By.id("id2")));
wait.until(ExpectedConditions.presenceOfElementLocated(By.id("id3")));
wait.until(ExpectedConditions.textToBePresentInElementLocated(By.tagName("body"), "text"));
wait.until(ExpectedConditions.titleIs("Page title"));
```

  - También podemos configurar una espera por defecto que se aplica a todo el tiempo de vida del objeto WebDriver:

```
driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

ExpectedConditions.textToBePresentInElementLocated(By.tagName("body"), "Hello").apply(driver);
```

3. Selenium

WebDriver

- Ejemplo: Firefox

```
public class FirefoxTest {
   private static final int TIMEOUT = 30; // seconds
   private WebDriver driver;

   @Before
   public void setupTest() {
      driver = new FirefoxDriver();
   }

   @After
   public void teardown() {
      if (driver != null) {
         driver.quit();
      }
   }

   @Test
   public void test() {
      driver.get("http://en.wikipedia.org/wiki/Main_Page");
      driver.findElement(By.id("searchInput")).sendKeys("Software");
      driver.findElement(By.id("searchButton")).click();

      WebDriverWait wait = new WebDriverWait(driver, TIMEOUT);
      wait.until(ExpectedConditions.textToBePresentInElementLocated(
         By.tagName("body"), "Computer software or simply software"));
   }
}
```

3. Selenium

WebDriver

- Ejemplo: Chrome
  - Es necesario descargarse el binario `chromedriver` y exportar su ruta absoluta en la variable de sistema Java *`webdriver.chrome.driver`*:

```
public class ChromeTest {
   protected WebDriver driver;

   @BeforeClass
   public static void setupClass() {
      System.setProperty("webdriver.chrome.driver",
         "/absolute/path/to/chromedriver");
   }
   @Before
   public void setupTest() {
      driver = new ChromeDriver();
   }
   @After
   public void teardown() {
      if (driver != null) {
         driver.quit();
      }
   }
   @Test
   public void test() {
      // Exercise and verify
  }
}
```

3. 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)

```
<dependency>
    <groupId>io.github.bonigarcia</groupId>
    <artifactId>webdrivermanager</artifactId>
    <version>1.3.0</version>
    <scope>test</scope>
</dependency>
```

https://github.com/bonigarcia/webdrivermanager

3. Selenium

WebDriver

- Ejemplo: Chrome (usando webdrivermanager)

```
public class ChromeTest {
   protected WebDriver driver;

   @BeforeClass
   public static void setupClass() {
      ChromeDriverManager.getInstance().setup();
   }
   @Before
   public void setupTest() {
      driver = new ChromeDriver();
   }
   @After
   public void teardown() {
      if (driver != null) {
         driver.quit();
      }
   }
   @Test
   public void test () {
      // Exercise and verify
   }
}
```

3. Selenium

WebDriver

- Ejemplo: Opera

```
public class OperaTest {
   protected WebDriver driver;

   @BeforeClass
   public static void setupClass() {
      System.setProperty("webdriver.opera.driver",
         "/absolute/path/to/operadriver");
   }
   @Before
   public void setupTest() {
      driver = new OperaDriver();
   }
   @After
   public void teardown() {
      if (driver != null) {
         driver.quit();
      }
   }
   @Test
   public void test() {
      // Exercise and verify
   }
}
```

```
public class OperaTest {
   protected WebDriver driver;

   @BeforeClass
   public static void setupClass() {
      OperaDriverManager.getInstance().setup();
   }
   @Before
   public void setupTest() {
      driver = new OperaDriver();
   }
   @After
   public void teardown() {
      if (driver != null) {
         driver.quit();
      }
   }
   @Test
   public void test () {
      // Exercise and verify
   }
}
```

3. Selenium

WebDriver

- Ejemplo: Internet Explorer

```
public class IExplorerTest {
   protected WebDriver driver;

   @BeforeClass
   public static void setupClass() {
      System.setProperty("webdriver.ie.driver",
         "C:/path/to/IEDriverServer.exe");
   }
   @Before
   public void setupTest() {
      driver = new InternetExplorerDriver();
   }
   @After
   public void teardown() {
      if (driver != null) {
         driver.quit();
      }
   }
   @Test
   public void test() {
      // Exercise and verify
  }
}
```

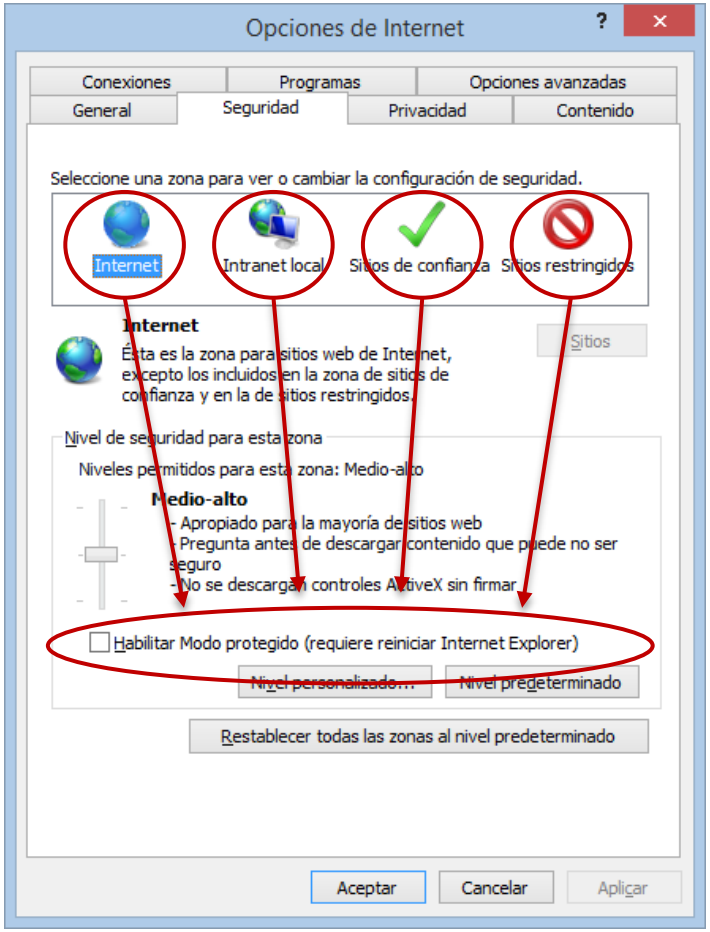
```
public class IExplorerTest {
   protected WebDriver driver;

   @BeforeClass
   public static void setupClass() {
      InternetExplorerDriverManager.getInstance().setup();
   }
   @Before
   public void setupTest() {
      driver = new InternetExplorerDriver();
   }
   @After
   public void teardown() {
      if (driver != null) {
         driver.quit();
      }
   }
   @Test
   public void test () {
      // Exercise and verify
   }
}
```

3. Selenium

WebDriver

- Ejemplo: Internet Explorer
- Además, a partir de Explorer 11 hay que cambiar la configuración de seguridad:



3. Selenium

WebDriver

- Ejemplo: Edge

```
public class EdgeTest {
   protected WebDriver driver;

   @BeforeClass
   public static void setupClass() {
      System.setProperty("edgedriver.ie.driver",
         "C:/path/to/MicrosoftWebDriver.exe");
   }
   @Before
   public void setupTest() {
      driver = new EdgeDriver();
   }
   @After
   public void teardown() {
      if (driver != null) {
         driver.quit();
      }
   }
   @Test
   public void test() {
      // Exercise and verify
  }
}
```

```
public class EdgeTest {
   protected WebDriver driver;

   @BeforeClass
   public static void setupClass() {
      EdgeDriverManager.getInstance().setup();
   }
   @Before
   public void setupTest() {
      driver = new EdgeDriver();
   }
   @After
   public void teardown() {
      if (driver != null) {
         driver.quit();
      }
   }
   @Test
   public void test () {
      // Exercise and verify
   }
}
```

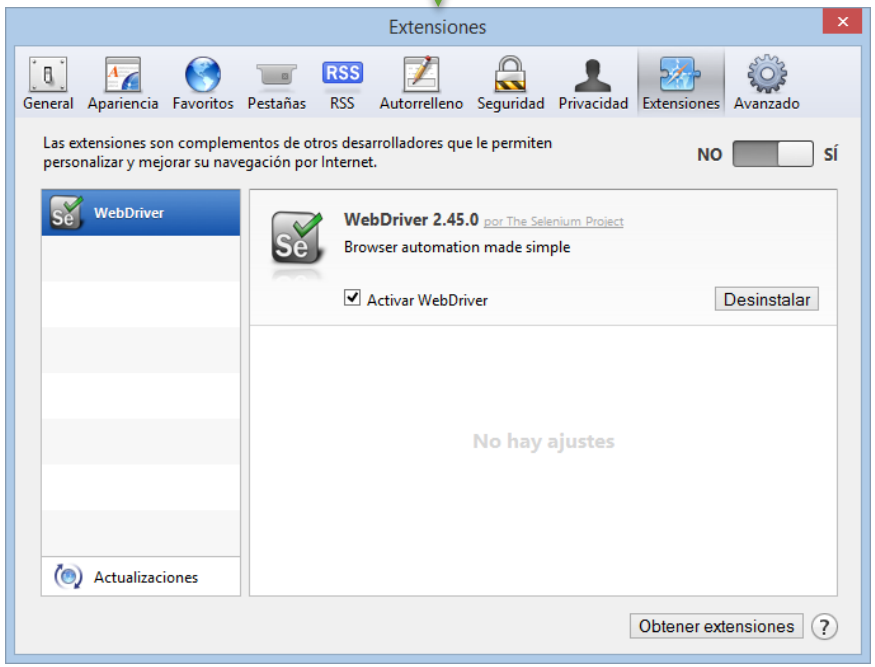
3. Selenium

WebDriver

- Ejemplo: Safari

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

No es necesario binario pero hay que instalar una extensión en Safari disponible en http://www.seleniumhq.org/download/



3. Selenium

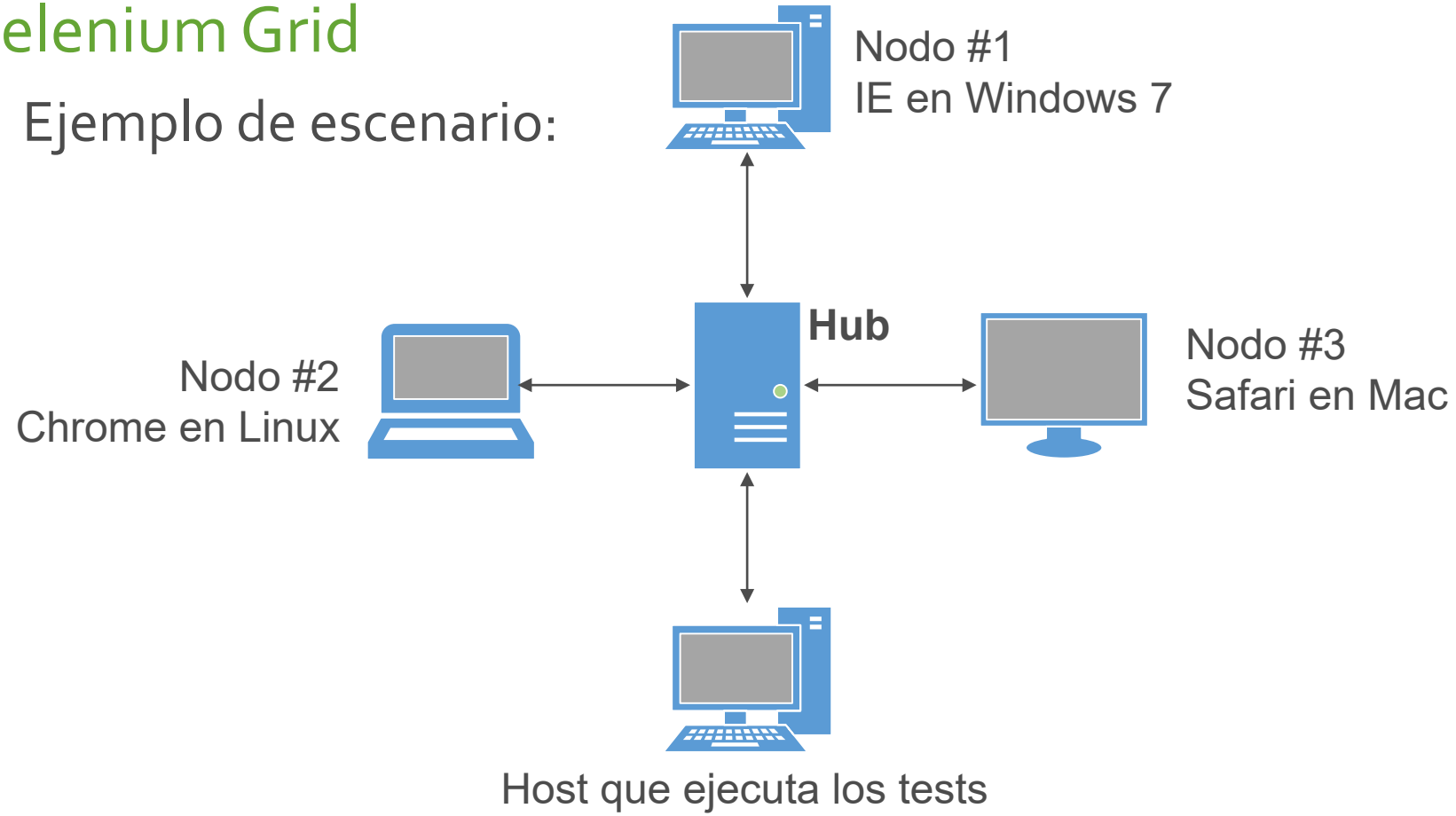
Selenium Grid

- Con Selenium WebDriver usamos los navegadores instalados de forma **local** en la máquina que está ejecutando los tests
- Selenium Grid permite la misma gestión programática de navegadores que ofrece WebDriver pero usando navegadores **remotos**, o sea, instalados en máquinas diferentes
- Arquitectura Selenium Grid:
  - Hub: Pieza central de la infraestructura que orquesta la ejecución de la prueba
  - Nodos: Máquinas que aportan navegadores en los que ejecutar pruebas

3. Selenium

Selenium Grid

- Ejemplo de escenario:



3. Selenium

Selenium Grid

- Cómo arrancar un Hub:
  - Desde línea de comandos:

```
java -jar selenium-server-standalone-2.48.2.jar -role hub -port 4444
```

  - Desde código Java:

```
<dependency>
   <groupId>org.seleniumhq.selenium</groupId>
   <artifactId>selenium-server</artifactId>
   <version>2.48.2</version>
</dependency>
```

```
import org.openqa.grid.internal.utils.*;
import org.openqa.grid.web.Hub;

public class StartHub {
   public static void main(String[] args)
    throws Exception {
      String hubAddress = "x.x.x.x";
      int hubPort = 4444;
      int timeout = 60; // seconds
      GridHubConfiguration config =
         new GridHubConfiguration();
      config.setHost(hubAddress);
      config.setPort(hubPort);
      config.setTimeout(timeout);
      Hub hub = new Hub(config);
      hub.start();
   }
}
```

3. Selenium

Selenium Grid

- Cómo registrar un Nodo (desde línea de comandos):

```
java -jar selenium-server-standalone-2.48.2.jar -role node -port <node-port> -hub
http://<hub-address>:<hub-port>/grid/register -browser browserName=<browser-name>,
version=<browser-version>,maxInstances=<max-instances>,platform=<platform> -maxSession
<max-sessions> -Dwebdriver.chrome.driver=${remoteChromeDriver} -timeout <seconds>
```

  - `hub`: URL en la que escucha el Hub el registro de nodos
  - `port`: puerto del nodo
  - `browserName`: android, chrome, firefox, htmlunit, internet explorer, iphone, opera
  - `version`: versión del navegador
  - `platform`: WINDOWS, LINUX, MAC
  - `maxInstances`: número máximo de navegadores de un tipo determinado
  - `maxSession`: número máximo de navegadores que pueden ser ejecutados en paralelo
  - `timeout`: tiempo en segundos para que el hub libere al nodo

3. Selenium

Selenium Grid

- Cómo registrar un Nodo (desde código Java):

```
RegistrationRequest req = new RegistrationRequest();
DesiredCapabilities cap = new DesiredCapabilities();
cap.setCapability(CapabilityType.PLATFORM, Platform.WIN8_1);
cap.setCapability(CapabilityType.BROWSER_NAME, BrowserType.CHROME);
req.addDesiredCapability(cap);
req.setRole(GridRole.NODE);
Map<String, Object> nodeConfiguration = new HashMap<String, Object>();
nodeConfiguration.put(RegistrationRequest.AUTO_REGISTER, true);
nodeConfiguration.put(RegistrationRequest.HUB_HOST, hubAddress);
nodeConfiguration.put(RegistrationRequest.HUB_PORT, hubPort);
nodeConfiguration.put(RegistrationRequest.PORT, nodePort);
nodeConfiguration.put(RegistrationRequest.REMOTE_HOST, "http://"
+ nodeAddress + ":" + nodePort);
nodeConfiguration.put(RegistrationRequest.PROXY_CLASS,
"org.openqa.grid.selenium.proxy.DefaultRemoteProxy");
nodeConfiguration.put(RegistrationRequest.MAX_SESSION, 1);
nodeConfiguration.put(RegistrationRequest.MAX_INSTANCES, 1);
req.setConfiguration(nodeConfiguration);
SelfRegisteringRemote remote = new SelfRegisteringRemote(req);
remote.startRemoteServer();
remote.startRegistrationProcess();
```

3. Selenium

Selenium Grid

- Cómo instanciar `RemoteWebDriver` (en Chrome y Firefox):

```
DesiredCapabilities capabilities = new DesiredCapabilities();

// Firefox
FirefoxProfile profile = new FirefoxProfile();
capabilities.setCapability(FirefoxDriver.PROFILE, profile);
capabilities.setBrowserName(DesiredCapabilities.firefox().getBrowserName());

// Chrome
ChromeOptions options = new ChromeOptions();
capabilities.setCapability(ChromeOptions.CAPABILITY, options);
capabilities.setBrowserName(DesiredCapabilities.chrome().getBrowserName());

WebDriver driver = new RemoteWebDriver(new URL("http://" + hubAddress + ":"
   + hubPort + "/wd/hub"), capabilities);
```

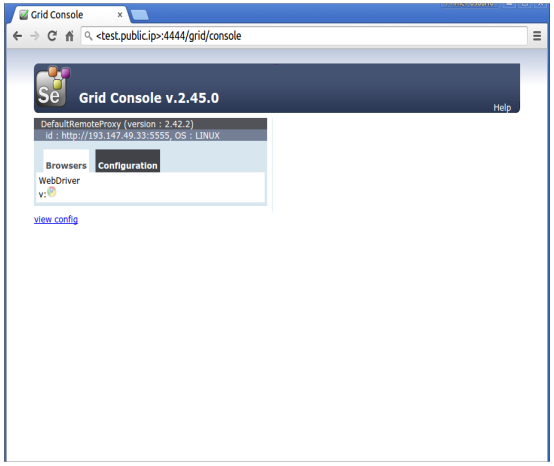
3. Selenium

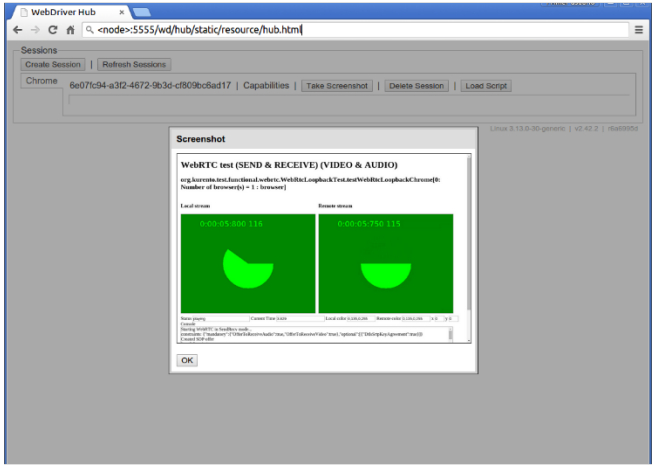
Selenium Grid

- Cómo depurar tests que se están ejecutando:

Consola Grid

http://<hub-address>:<hub-port>/grid/console





Estado de cada Nodo

http://<node-address>:<node-port>/wd/hub/static/resource/hub.html

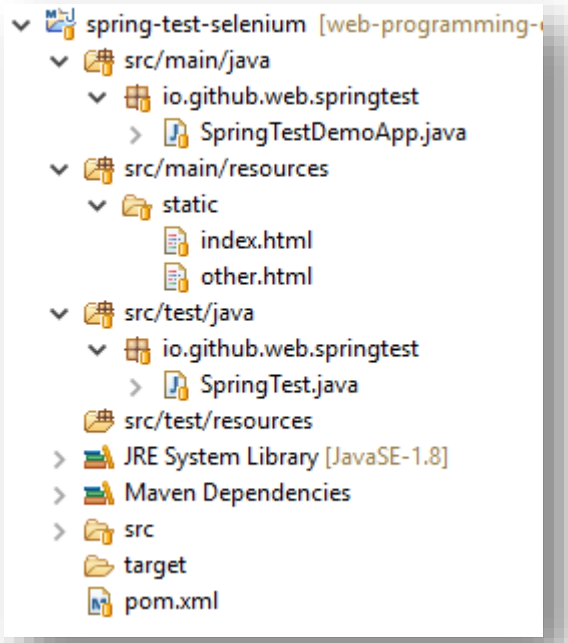
Índice de contenidos

4. Pruebas en aplicaciones web con Spring Boot

- La dependencia (starter) de pruebas en Spring Boot se llama `spring-boot-starter-test`
- Esta dependencia proporciona integración con las siguientes librarías de pruebas:
  - JUnit. Librería de pruebas unitarias. http://junit.org/
  - Hamcrest. Librería de predicados. http://hamcrest.org/
  - Mockito. Librería de mocks. http://mockito.org/

4. Pruebas en aplicaciones web con Spring Boot

- Ejemplo:




```
<parent>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-parent</artifactId>
   <version>1.2.7.RELEASE</version>
</parent>

<dependencies>
   <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-web</artifactId>
   </dependency>
   <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-test</artifactId>
   </dependency>
   <dependency>
      <groupId>org.seleniumhq.selenium</groupId>
      <artifactId>selenium-java</artifactId>
      <version>2.48.2</version>
   </dependency>
   <dependency>
      <groupId>io.github.bonigarcia</groupId>
      <artifactId>webdrivermanager</artifactId>
      <version>1.3.0</version>
   </dependency>
</dependencies>
```

4. Pruebas en aplicaciones web con Spring Boot

- Ejemplo:

```
<!DOCTYPE html>
<html>
<head>
<title>Spring Boot Test</title>
</head>

<body>
<h1>Home page</h1>
Go to <a href="other.html">another</a> page.
</body>
</html>
```

```
<!DOCTYPE html>
<html>
<head>
<title>Spring Boot Test</title>
</head>

<body>
<h1>Other page</h1>
Hello!
</body>
</html>
```

```
@SpringBootApplication
public class SpringTestDemoApp extends
WebMvcConfigurerAdapter {

   public static void main(String[] args) {
      SpringApplication.run(SpringTestDemoApp.class, args);
   }
}
```

4. Pruebas en aplicaciones web con Spring Boot

- Ejemplo:

```
@WebIntegrationTest
@RunWith(SpringJUnit4ClassRunner.class)
@SpringApplicationConfiguration(classes = SpringTestDemoApp.class)
public class SpringTest {

   private static final long TIMEOUT = 30; // seconds
   private WebDriver driver;

   @BeforeClass
   public static void setupClass() {
      ChromeDriverManager.getInstance().setup();
   }

   @Before
   public void setupTest() {
      driver = new ChromeDriver();
   }

   @After
   public void teardown() {
      if (driver != null) {
         driver.quit();
      }
   }
```

⋮

4. Pruebas en aplicaciones web con Spring Boot

- Ejemplo:

```
    @Test
    public void test() {
        // Always wait TIMEOUT seconds
        driver.manage().timeouts().implicitlyWait(TIMEOUT, TimeUnit.SECONDS);

        // Open system under test
        driver.get("http://localhost:8080/");

        // Verify that first page has title "Home page"
        ExpectedConditions.titleIs("Home page").apply(driver);

        // Click on link
        driver.findElement(By.linkText("another")).click();

        // Verify that second page contains the string "Hello"
        ExpectedConditions.textToBePresentInElementLocated(By.tagName("body"),
                "Hello").apply(driver);
    }

}
```

4. Pruebas en aplicaciones web con Spring Boot

- Ejemplo:

```
  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::        (v1.2.7.RELEASE)

2015-10-31 19:31:37.302  INFO 7512 --- [           main]
io.github.web.springtest.SpringTest      : Starting SpringTest on home-win with PID
7512 (C:\Users\boni\Documents\projects\web-programming-examples\spring-test-
selenium\target\test-classes started by boni in
C:\Users\boni\Documents\projects\web-programming-examples\spring-test-selenium)
...
2015-10-31 19:31:41.109  INFO 7512 --- [           main]
s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat started on port(s): 8080 (http)
2015-10-31 19:31:41.112  INFO 7512 --- [           main]
io.github.web.springtest.SpringTest      : Started SpringTest in 4.122 seconds (JVM
running for 7.452)
Starting ChromeDriver 2.20.353145 (343b531d31eeb933ec778dbcf7081628a1396067) on
port 15450
Only local connections are allowed.
2015-10-31 19:32:07.405  INFO 7512 --- [nio-8080-exec-1]
o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring FrameworkServlet
'dispatcherServlet'
...
```