Automated Web Tests with Selenium 2

Java Forum Stuttgart 2013

Mario Goller

Trivadis AG

BASEL BERN LAUSANNE ZÜRICH DÜSSELDORF FRANKFURT A.M. FREIBURG I.BR. HAMBURG MÜNCHEN STUTTGART WIEN

trivadis
makes IT easier.

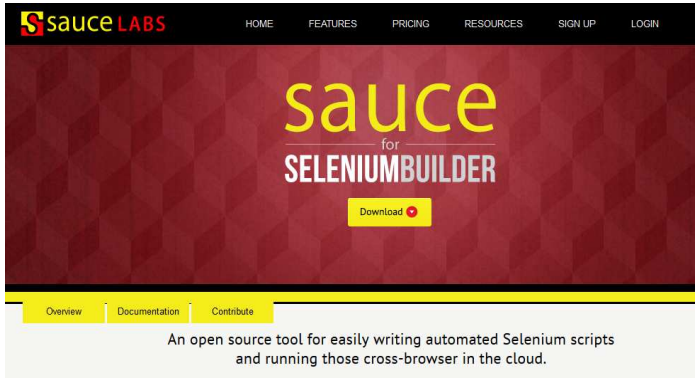
AGENDA

1. **Selenium & WebDriver**
2. Locating Elements
3. User Interactions
4. Page Objects
5. Selenium GRID
6. Testing on Mobile Devices

Selenium History

- Started in 2004 in by **Jason Huggins** who worked in ThoughtWorks
  - (originated from TW Time Sheet application)
- Jason joins Google in 2007
- Currently CTO of SauceLabs
- **Simon Stewart** Started work on WebDriver in Google

http://google-opensource.blogspot.in/2009/05/introducing-webdriver.html



Who uses Selenium

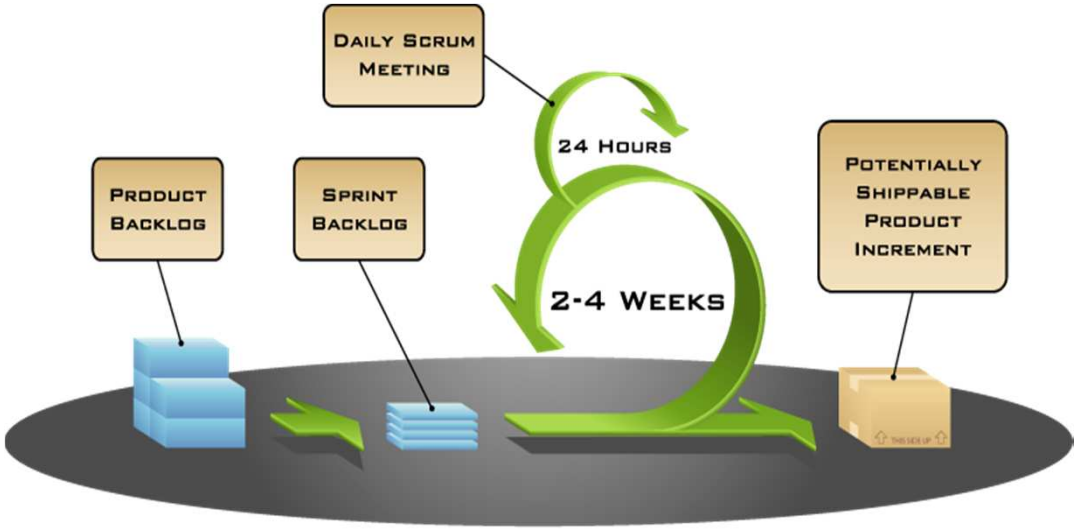
Challenges

Language/
Framework

Browser

Platform

Challenges



SCRUM

XP

Advantages of Test Automation

- Frequent and quick regression testing
- Rapid feedback to developers
- Virtually unlimited iterations of test case execution
- Support for Agile and extreme development methodologies
- Increased test coverage
- Enables QA team to utilize bandwidth in exploratory testing.

What is Selenium?

What is Selenium?

Selenium IDE

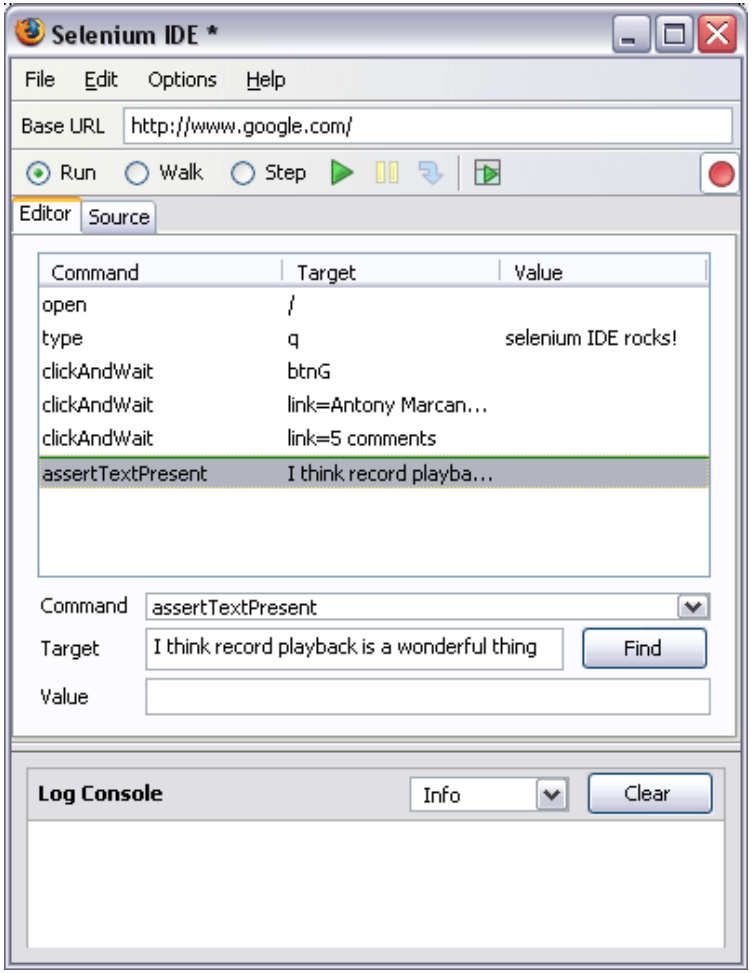
Selenium RC

Selenium GRID

Selenium Core
(WebDriver)

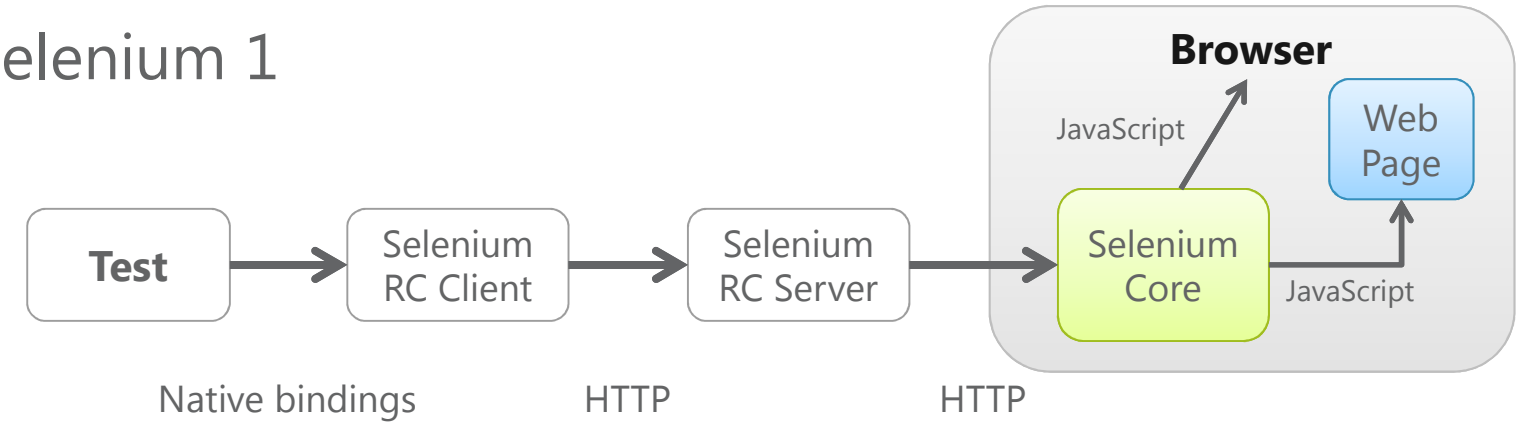
Selenium IDE

- Firefox plug-in
- Integrated Development Environment for Selenium Tests.
- Provides record and playback functionality.
- Provides the ability to debug test scripts.
- Auto complete common selenium commands
- Export tests in different formats like Ruby, Python, JUnit so on.
- Supports user extensions

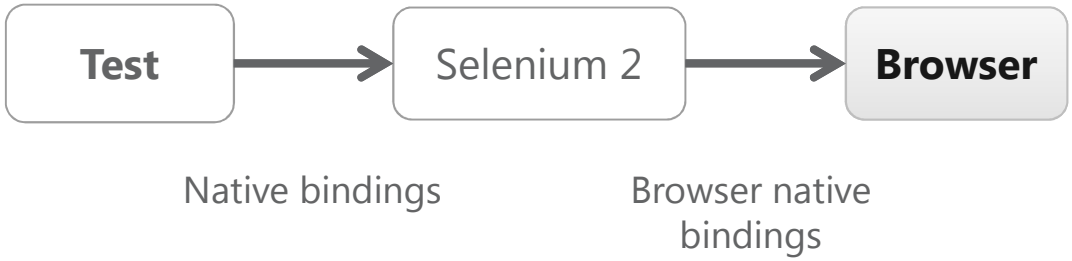


Selenium Versions

Selenium 1



Selenium 2

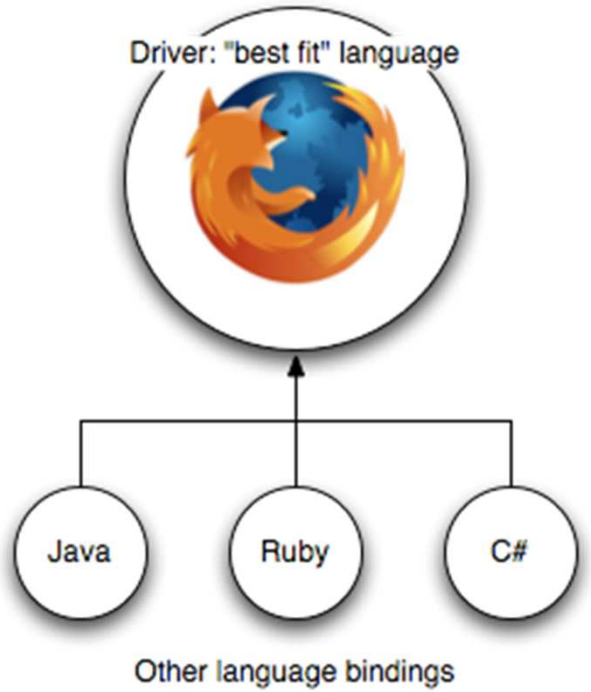


Selenium 2 (WebDriver)

- Object Oriented, smaller, compact API
  - make it easier to work with compared to RemoteControl
- Mimics the way that your users work with your site and apps
- Offers advanced user interactions APIs
  - Drag and Drop
  - Clicking multiple elements with Control key

Selenium WebDriver

- ChromeDriver
- InternetExplorerDriver
- FirefoxDriver
- OperaDriver
- AndroidDriver
- iPhoneDriver
- RemoteWebDriver



- Support for WebDriver is baked into the browser itself: **your tests run fast and are stable**

Selenium WebDriver

- Set up the WebDriver in JUnit

```
@Before
public void setUp() throws Exception {
  driver = new FirefoxDriver();
  baseUrl = "http://www.myapp.com/";
  driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);
}
```

```
@Before
public void setUp() throws Exception {
  driver = new InternetExplorerDriver();
  ...
}
```

```
@Before
public void setUp() throws Exception {
  driver = new ChromeDriver();
  ...
}
```

AGENDA

1. Selenium & WebDriver
2. **Locating Elements**
3. User Interactions
4. Page Objects
5. Selenium GRID
6. Testing on Mobile Devices

Locating Elements

- By ID
  - `driver.findElement(By.id(<element ID>))`
- By name
  - `driver.findElement(By.name(<element name>))`
- By class name
  - `driver.findElement(By.className(<element class>))`
- By tag name
  - `driver.findElement(By.tagName(<htmltagname>))`
- By CSS / XPath
  - `driver.findElement(By.cssSelector(<css selector>))`
  - `driver.findElement(By.xpath (<xpath query expression>))`

AGENDA

1. Selenium & WebDriver
2. Locating Elements
3. **User Interactions**
4. Page Objects
5. Selenium GRID
6. Testing on Mobile Devices

Page Interactions

- `webElement.click()`
- `webElement.sendKeys(...)`
- `webElement.submit()`
- Actions class -> Mouse Events / Drag and Drop


```
@Test
public void testDragDrop() {
  WebElement source = driver.findElement(By.id("draggable"));
  WebElement target = driver.findElement(By.id("droppable"));

  Actions builder = new Actions(driver);
  builder.dragAndDrop(source, target).perform();
  try
  {
    assertEquals("Dropped!", target.getText());
  } catch (Error e) {
    verificationErrors.append(e.toString());
  }
}
```

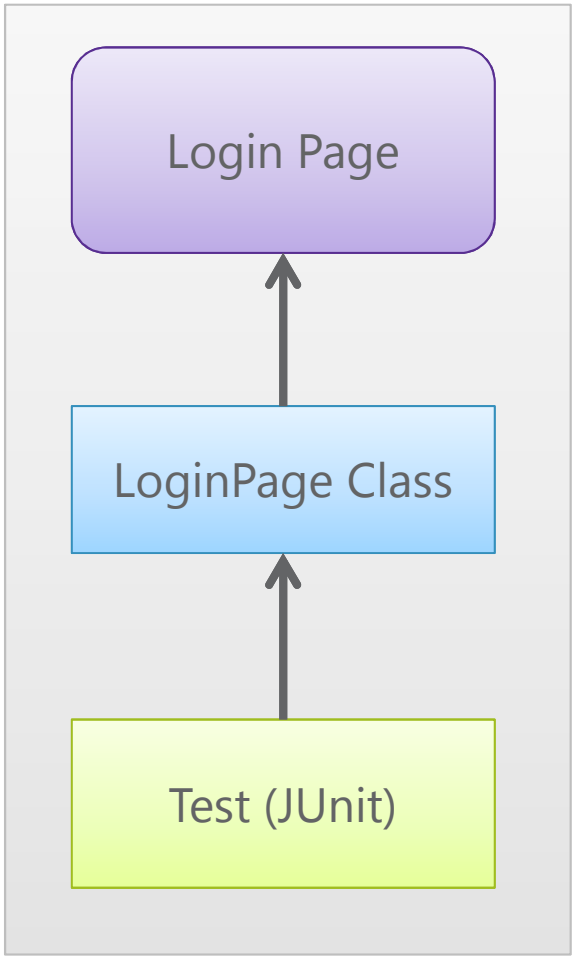
AGENDA

1. Selenium & WebDriver
2. Locating Elements
3. User Interactions
4. **Page Objects**
5. Selenium GRID
6. Testing on Mobile Devices

Using Page Objects

- Consolidates the code for interaction with any give UI element
- Allows ou to model the UI in your tests
- Exposes methods that reflect the things a user can see and do on that webpage

```
@Test
public void testLogin() throws Exception {
     LoginPage loginPage = new LoginPage(driver);
     HomePage homePage =
          loginPage.loginAs("peter", "secure");
     assertEquals("Welcome",homePage.getTitle());
}
```



Page Object - Example

```
public class LoginPage {

     private final WebDriver driver;

     @FindBy(id="button_submit")
     public WebElement submitButton;

     public LoginPage(WebDriver driver) {
          this.driver = driver;
          ...
     }

     public HomePage loginAs(String username, String password) {
          executeLogin(username, password);
          return new HomePage(driver);
     }

     private void executeLogin(String username, String password) {
          driver.findElement(By.id("field_username")).sendKeys(username);
          driver.findElement(By.id("field_password")).sendKeys(password);
          submitButton.submit();
     }
}
```

Advanced Topics

- Execute JavaScript

```
JavascriptExecutor js = (JavascriptExecutor) driver;

String title = (String) js.executeScript("return document.title");
assertEquals("Google", title);
```

- Taking Screenshots

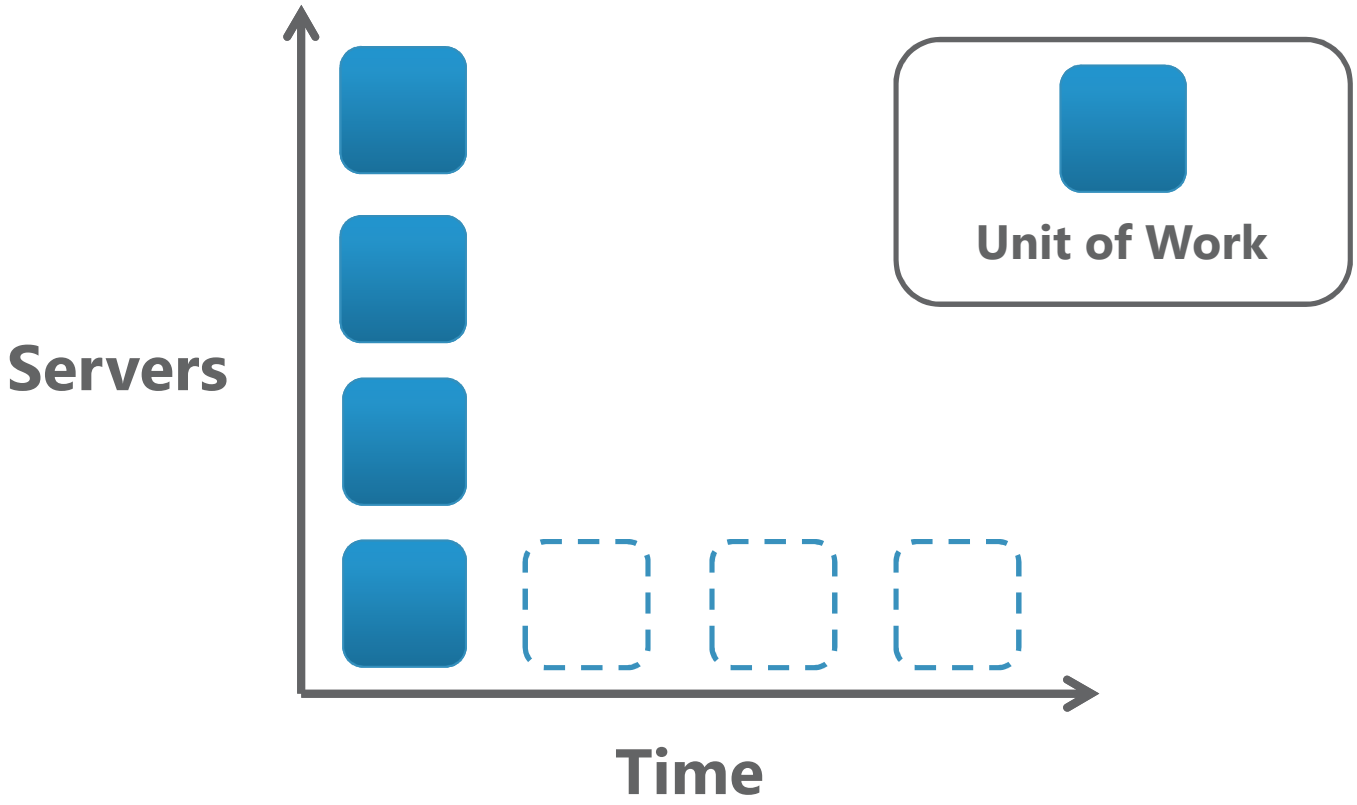
```
@Test
public void testTakesScreenshot() {
     try {
          File scrFile = ((TakesScreenshot) driver)
          .getScreenshotAs(OutputType.FILE);
          FileUtils.copyFile(scrFile, new File("c:\\tmp\\main_page.png"));
     } catch (Exception e) {
          e.printStackTrace();
     }
}
```

AGENDA

1. Selenium & WebDriver
2. Locating Elements
3. User Interactions
4. Page Objects
5. **Selenium GRID**
6. Testing on Mobile Devices

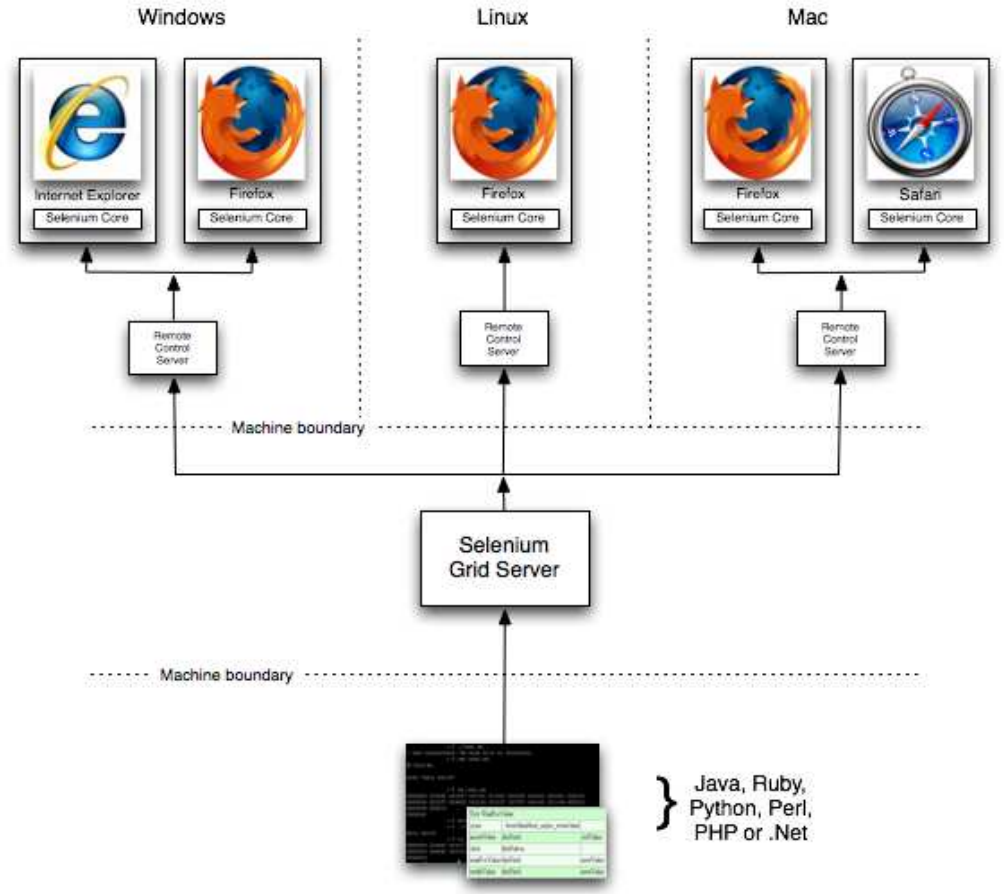
Selenium GRID

- Parallelizing tasks



What is Selenium Grid?

- Maintain a cluster of Selenium nodes
- Configure tests for different environments (Browser/Platform)
- Parallelize your tests



Start up the Selenium Hub

Prerequisites:

- A Java 5+ JRE
- selenium-server-standalone.jar

```
$ java –jar selenium-server-standalone.jar –role hub
```

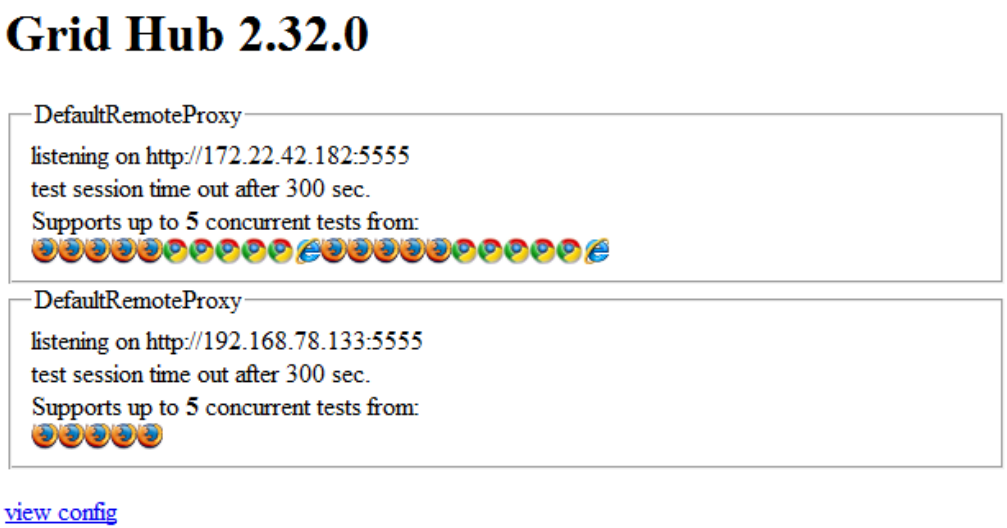
Three options (in order of precedence):

- Grid 1 compatible YAML file
- JSON configuration file
- Command-line flags
- View current hub config in console (Click “View Config” link)

The Selenium Grid Console

Web console to see grid state and config:

- Open a web browser to:
  - http://localhost:4444/grid/console
- Beta console available at:
  - http://localhost:4444/grid/beta/console



Starting up a Node

Prerequisites:

- A Java 5+ JRE
- selenium-server-standalone.jar
- Run:

```
$ java –jar selenium-server-standalone.jar –role node
                    –hubHost localhost
```

Two options (in order of precedence):

- JSON configuration file
- Command-line flags
- View current config in beta console (Click “Configuration” tab for node

RemoteWebDriver

```
public void myTest() throws Exception {

    //FirefoxProfile fp = new FirefoxProfile();
    //DesiredCapabilities dc = DesiredCapabilities.firefox();
    //dc.setCapability(FirefoxDriver.PROFILE, fp);

    ChromeOptions options = new ChromeOptions();
    DesiredCapabilities dc = DesiredCapabilities.chrome();
    dc.setCapability(ChromeOptions.CAPABILITY, options);
    WebDriver driver = new RemoteWebDriver(dc);

    WebDriver driver = new RemoteWebDriver(new URL(
    "http://localhost:4444/wd/hub"), DesiredCapabilities.firefox());

    driver.get("http://www.google.com");

}
```

Parallelizing Tests

- Grid helps you organize resources and spread load
- …but you still need to parallelize your tests
- TestNG makes this pretty straightforward
- Works also for JUnit:
  - See article Adam Goucher wrote for SauceLabs's blog

Multi-Browser Parallel Testing

```java
public class ParallelRunning {
   private WebDriver driver = null;

   @BeforeTest
   @Parameters({ "BROWSER" })
   public void setup(String BROWSER) {
       if (BROWSER.equals("FF")) {
           driver = new FirefoxDriver();
       } else if (BROWSER.equals("IE")) {
           driver = new InternetExplorerDriver();
       } else if (BROWSER.equals("CH")) {
           driver = new ChromeDriver();
       }
   }

   @Test
   public void testParallel() throws Exception {
       driver.get("http://www.google.com");
       ...
       driver.quit();
   }
}
```

TestNG

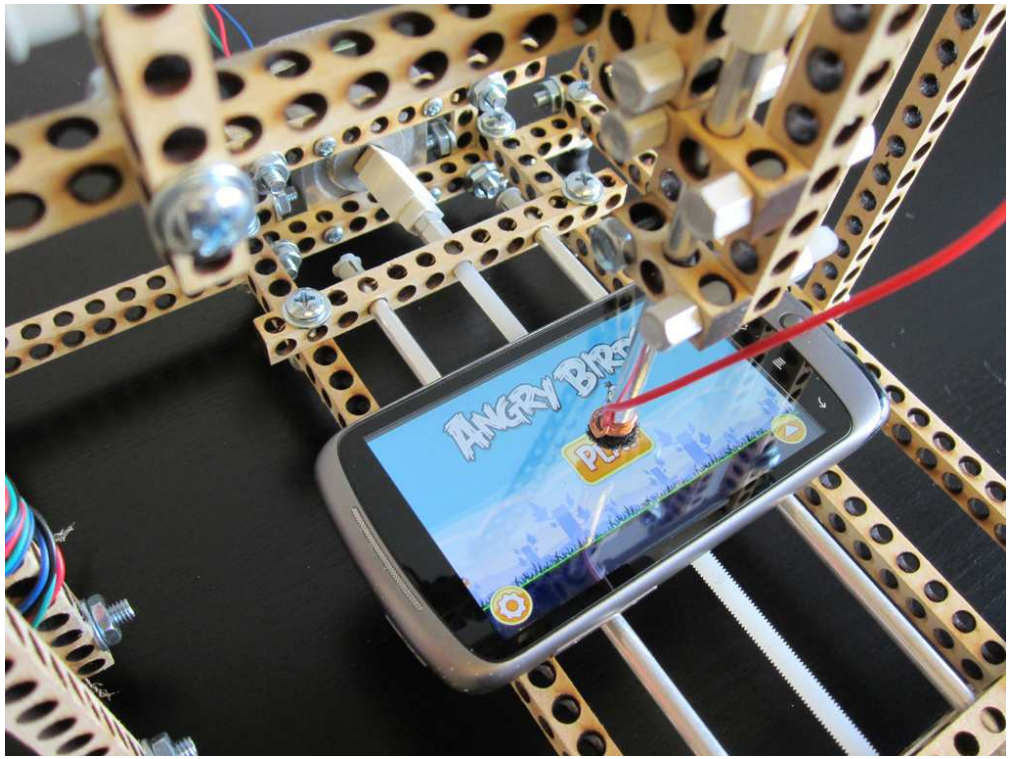
```xml
<suite name="webDriver" parallel="tests">
  <test name="WebDriverDemo Witn FF" preserve-
order="true">
  <parameter name="BROWSER" value="FF" />
  <classes>
   <class name="com.web.ParallelRunning" />
  </classes>
 </test>
  <test name="WebDriverDemo with IE" preserve-
order="true">
  <parameter name="BROWSER"
value="IE"></parameter>
  <classes>
   <class name="com.web.ParallelRunning"></class>
  </classes>
 </test>
</suite>
```

AGENDA

1. Selenium & WebDriver
2. Locating Elements
3. User Interactions
4. Page Objects
5. Selenium GRID
6. **Testing on Mobile Devices**

Testing Mobile Devices

… with Robots?



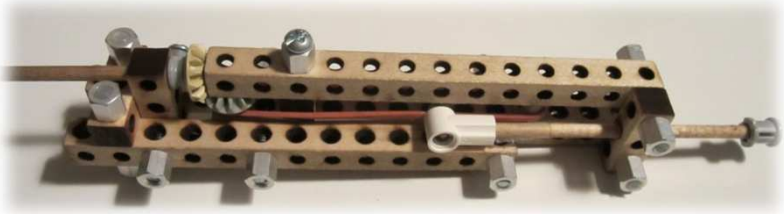
Bitbeambot (http://bitbeam.org)

BitbeamBot

- An open source hardware CNC robot for testing applications on mobile device

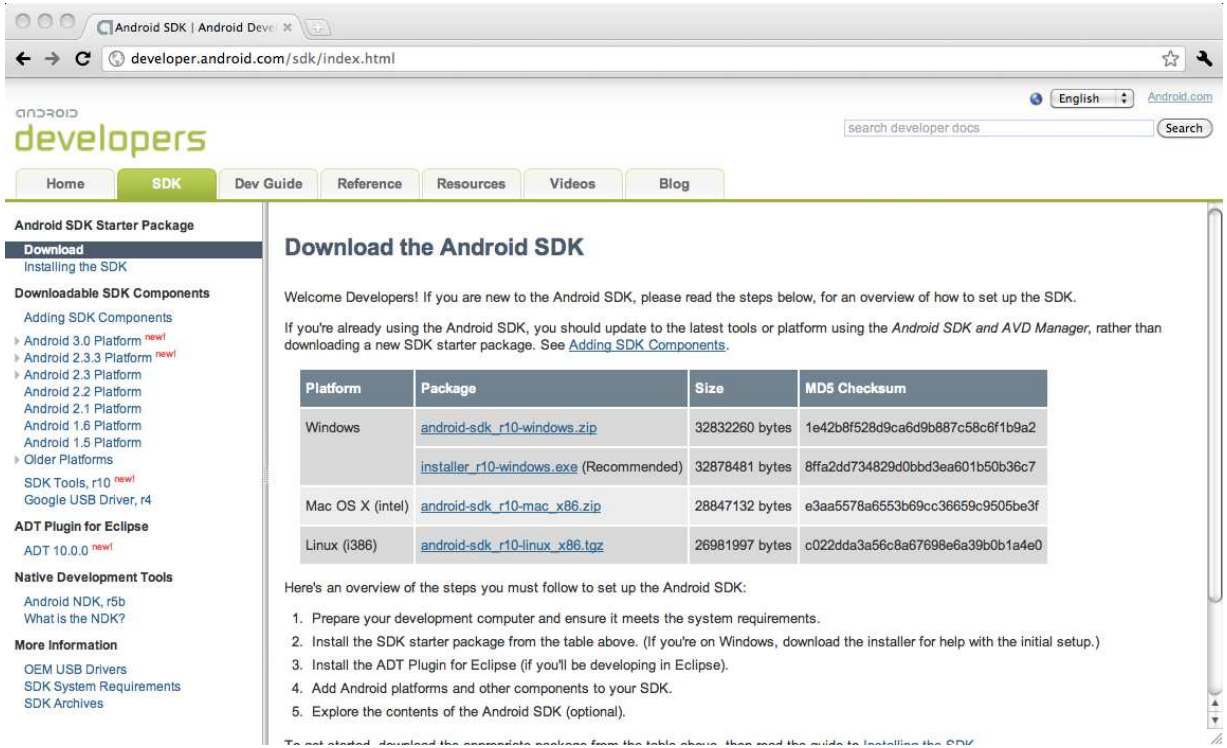
Why?

- **BitbeamBot is an experiment to take Selenium out of the screen and into the real world.**
- Selenium is a software-based robot
- Selenium mimics and automates how users interact with an application
- For mobile, this means handling real devices



Mobile Testing with Selenium

- How to…



Mobile Testing with Selenium

- Create AVD

```
$ ./android create avd -n my_android -t 8 -c 100M
```

- Start Emulator

```
$ ./emulator -avd my_android -no-audio -no-boot-anim -scale .8 &
```

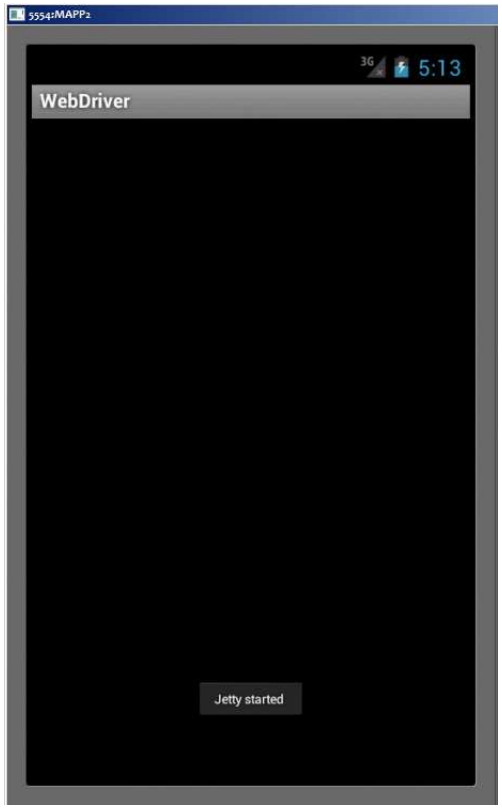
- Install Selenium APK

```
$ cd ~/android_sdk/platform-tools/
$ ./adb -e install -r ~/selenium/android/prebuilt/android-server.apk
```

```
$ ./adb forward tcp:8080 tcp:8080
```

Mobile Testing with Selenium

- Launch the WebDriver App:



Selenium - Why to use?

- Communicate with browsers in 'native' way via drivers
- Not limited to JavaScripts
  - For example: upload files, work with popups, iFrames etc.
- Support for a lot of browsers including mobile phone browsers
- Support many languages, including Java , C#, Ruby, Python and so on
- **Free & Open Source !!!**

Selenium & Continuous Integration

- Run Selenium tests as part of the build
  - Works with both Core and Driven modes (each time a developer checks in)
- Can generate HTML reports, published to team members
- Helps catch bugs ASAP
  - Addresses risk of catching bugs late in the cycle

Questions ?

THANK YOU.

Trivadis AG

Mario Goller

Europastrasse 5
CH-8152 Glattbrugg (ZH)

Tel. +41-44-808 70 20
Fax +41-44-808 70 21

info@trivadis.com
www.trivadis.com

BASEL BERN LAUSANNE ZÜRICH DÜSSELDORF FRANKFURT A.M. FREIBURG I.BR. HAMBURG MÜNCHEN STUTTGART WIEN

trivadis
makes IT easier.