Selenium

What is it?

"*Selenium automates browsers. That's it! What you do with that power is entirely up to you. Primarily, it is for automating web applications for testing purposes, but is certainly not limited to just that. Boring web-based administration tasks can (and should!) also be automated as well.*" - www.seleniumhq.org

Sample Selenium Test

Here is a sample test which request a chrome browser from browserstack and then performs the following:

- navigate to https://web.dev.placodermi.xyz/
- find page element with id "gitref-input" and input value "john"
- find page element with id "filename-input" and input value "test_app.js"
- click the button with id "submit-gitref-filename"
- wait to see if a form named "CCForm" gets loaded.

```
package com.irdeto.placodermi.test.solution.web;

import org.openqa.selenium.By;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.remote.CapabilityType;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.remote.RemoteWebDriver;

import java.net.URL;
import java.util.concurrent.TimeUnit;

public class SeleniumSample {

    public static final String HUB_URL = "http://selenium.dev.pci.irdeto.com:4444/wd/hub/";
    public static final String DEMO_WEB_URL = "https://web.dev.placodermi.xyz/";

    public static final String USERNAME = "XXX_USER_XXX";
    public static final String AUTOMATE_KEY = "XXX_PASSWORD_XXX";
    public static final String BROWSERSTACK_URL = "https://" + USERNAME + ":" + AUTOMATE_KEY + "@hub-cloud.
browserstack.com/wd/hub";
    public static final String CAPABILITY_BROWSERSTACK_LOCAL = "browserstack.local";


    public static void main(String[] args) throws Exception {

        WebDriver driver = null;

        try {
            //use chrome and turn off ssl cert check
            DesiredCapabilities caps = DesiredCapabilities.chrome();
            caps.setCapability(CapabilityType.ACCEPT_SSL_CERTS, true);
            caps.setCapability(CAPABILITY_BROWSERSTACK_LOCAL, true);


//            driver = new RemoteWebDriver(new URL(HUB_URL), caps);
            driver = new RemoteWebDriver(new URL(BROWSERSTACK_URL), caps);
            driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
            driver.manage().timeouts().pageLoadTimeout(30, TimeUnit.SECONDS);
            driver.manage().window().setSize(new Dimension(1920, 1080));

            driver.get(DEMO_WEB_URL);
            WebElement gitRefElement = driver.findElement(By.id("gitref-input"));
            gitRefElement.clear();
            gitRefElement.sendKeys("john");
            String gitref = gitRefElement.getAttribute("value");
            System.out.println("GITREF=" + gitref);

            WebElement filenameElement = driver.findElement(By.id("filename-input"));
            filenameElement.clear();
            filenameElement.sendKeys("test_app.js");

            WebElement enterButtonElement = driver.findElement(By.id("submit-gitref-filename"));
```

```
            enterButtonElement.click();

            boolean found=false;
            int retries=0;
            while(found==false && retries <10) {

                WebElement ccFormElement = driver.findElement(By.name("CCForm"));
                if(ccFormElement!=null){
                    found=true;
                    break;
                }

                //wait for page to load
                Thread.sleep(1000);
                retries++;
            }

            if(found){
                System.out.print("Found CC Form");
            }else{
                System.out.print("Failed to find CC Form");
            }

        }catch(Exception ex){
            ex.printStackTrace();
        }finally{
            if(driver!=null){
                driver.close();
                driver.quit();
            }
        }
    }
}
```

Testing against what?

We have some options here.

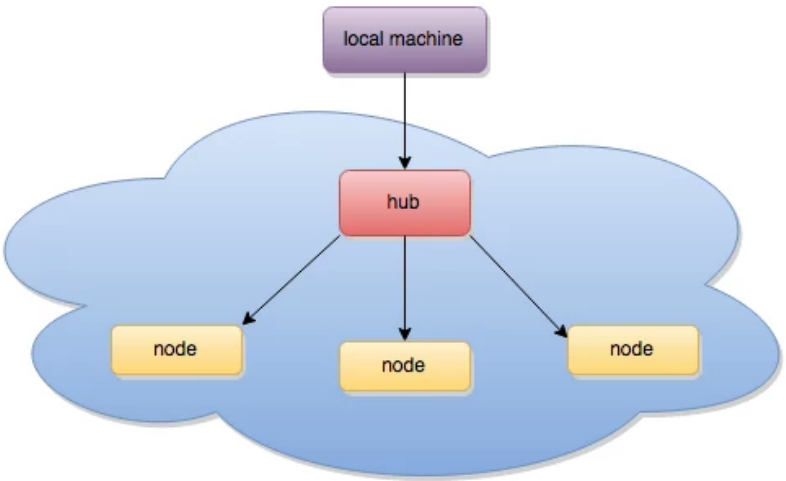
We can test against services offering selenium hosts such as browserstack.com or we can deploy our own.

There are docker images available that will allow us to test against our own hosted browsers:

> https://hub.docker.com/r/selenium/hub/

Deploying our own Selenium Hub

For a good understanding of deploying to a hub: http://testdetective.com/selenium-grid-with-docker/



Here are the step to setup our own docker hub:

```
>docker run -d -p 4444:4444 --name selenium-hub selenium/hub

# VNC on PORT 5900 WITH PASSWORD "secret"

>docker run -d --name selenium-chrome-debug -p 5900:5900 --link selenium-hub:hub selenium/node-chrome-debug


# VNC on PORT 5901 WITH PASSWORD "secret"
>docker run -d --name selenium-firefix-debug -p 5901:5900 --link selenium-hub:hub selenium/node-firefox-debug
```

When testing against our own selenium hub we would connect using:

```
driver = new RemoteWebDriver(new URL("http://selenium.dev.pci.irdeto.com:4444/wd/hub/"), caps);
```

Testing against Browser Stack

Browser stack is a service which allows us to test a large number of browsers against our web pages.

Testing Local Websites

To allow BrowserStack to connect to our local, not public web site, you will need to download an application from browserstack which will setup a tunnel to your local machine.

See https://www.browserstack.com/local-testing for more details.

Setup the tunnel to browserstack:

> ./BrowserStackLocal --key XXX

In order to make this easier, we setup a docker container which connects up to browserstack for us.