Automated Web App Testing on Desktop and Mobile Browsers

With Sauce Labs you can run automated tests of your web apps on a variety of operating system/browser versions for desktop and mobile browsers. This topic provides an overview of setting up and running an automated Selenium test, with example test scripts that show how to set your connection and authentication to Sauce Labs, set desired capabilities for your tests, and make sure that the Sauce Labs browser cloud can connect to your app.

Set Up

- You need a Sauce Labs account
- You need to have a Selenium environment already set up on your local machine to run automated tests on Sauce Labs. If you need help with that, you can find setup instructions for popular programming languages in the **demo** directories (for example, **demo-java**) of the Sauce Labs Training repository on GitHub.
- You should already have a Selenium script that you run locally, or remotely on a Selenium grid or another service, that you want to update to run on Sauce Labs. If you don't, you can find example scripts that are already set up to run on Sauce using the Sauce Labs demo app, and that you can modify to set up your own tests, in the **demo** directories (for example, **demo-java**) of the Sauce Labs Training repository on GitHub.
- The web app you want to test must be accessible over the Internet for our virtual machines to be able to connect to it. If it's on your local machine or behind a firewall, you'll need to set up and use Sauce Connect Proxy.

Log in to Sauce Labs to Run the Tests

Before you start modifying your existing test scripts or the examples, you should log into Sauce Labs with your username and password, and navigate to **D ashboard > Automated Tests**. As you modify the scripts, you can verify that the changes are valid by running the script on Sauce and making sure that it executes and completes on the dashboard.

Set the RemoteWebDriver Connection

If you have an existing Selenium script that you want to update to run on Sauce Labs, the first thing you need to do is to set the `RemoteWebDriver` to connect to the Sauce Labs browser cloud. For example, if your Selenium test is written in Java, you would set the connection to Sauce Labs like this:

RemoteWebDriver Java Example

```
driver = new RemoteWebDriver(new URL("http://ondemand.saucelabs.com:80/wd/hub"), capabilities);
```

The example scripts in the next section are already set to connect to the Sauce Labs US data center.

IP Ranges and Data Center Endpoints
Sauce Labs has data centers in both the US and EU. You can access Sauce Labs services from either location by providing the appropriate URL endpoints, and whitelisting their associated IP ranges.

- **Status Page for the US Data Center**: https://status.saucelabs.com
- **Status Page for the EU Data Center**: http://status.eu-central-1.saucelabs.com/

Verify Your Authentication Credentials

Once you've set the `RemoteWebDriver` connection to Sauce Labs, you need to add your username and access key for Sauce Labs to your test script, as shown in the examples below. You can find the access key in the **Account Profile** page of the Sauce Labs web interface:

1. Sign in to https://saucelabs.com with the username you set up when you created your account.
   You will use this same username in your test script.
2. To find your access key:
   a. Click your name in the **Account Profile** menu in the upper-right corner.
   b. Click **User Settings**.
   c. Scroll down to **Access Key** and click **Show**.
   d. Enter the password you use to sign in to Sauce Labs to view the access key.
   e. Click the **Copy** icon.

If you want to make sure your credentials work without having to update an existing script, copy and paste your credentials into the code as indicated in the appropriate example script, and then run it from your local machine. This will launch a test to open a browser, connect to the Sauce Labs sample app, and then close the browser and quit the test.

Use Environment Variables for Authentication Credentials

It's a recommended Best Practice to use environment variables for your credentials, rather than hardcoding them into your scripts. This makes it easy to run a test written by anyone from your local machine or continuous integration server, and also prevents security issues with having your authentication credentials exposed in the script. After you've run your first test with hardcoded credentials, you should set it to use environment variables. Set up the environment variables on your local machine as shown in the Best Practices topic, and then you can incorporate them into your scripts as shown in these examples.

Add Desired Capabilities

Desired capabilities are the platform, operating system, and browser you want to test against. You can use the Platform Configurator tool to automatically generates the code snippets for these capabilities.

1. Go to the Platform Configurator.
2. Under **API**, select **Selenium**.

> **Selecting Desired Capabilities for Mobile Native Browser Tests**
>
> To run a test against a mobile native browser, select **Appium**, select the mobile device you want to test with, and then select **Web Testing**.

3. Select the operating system and browser you want to test against.
4. Under **Copy Code**, select the language you are using.
   The Platform Configurator will generate the code snippet for the desired capabilities.
5. Click **Copy**.
6. Paste the code into your Selenium test script, or into one of the example test scripts, and then run the test to see how the web app performs on that platform/OS/browser combination. If you use one of the example test scripts, the test will open the selected browser, load the Sauce Labs sample web app, and close the browser.

Set the Web App to Test

The final step in setting up your Selenium tests on Sauce is to use the `driver.get` command to set the URL of the web app you want to test, as shown in this example. The example tests scripts are all set to run against the Sauce Labs demo app, so you can paste the URL for your web app in place of the `saucedemo` URL to make sure that the Sauce Labs browser cloud can connect to your app. If your app is hosted on your local machine or behind a firewall, you'll need to set up Sauce Connect Proxy, which launches a secure tunnel to establish the connection.

```
driver.navigate().to("https://www.saucedemo.com");
```

Best Practices and Reporting Test Results

Now that you've been able to get a test running on Sauce, there are a few other modifications you can make to your scripts to incorporate testing best practices, and to also make sure that you can report Pass/Fail results to your dashboard. You can find more information on both of these topics under Best Practices for Running Tests and Setting Test Status to Pass or Fail. These example scripts show how to implement the best practices of using timeouts to control text execution times, annotating tests, and setting identifying information for your tests, as well as a way to send Pass/Fail results to the Sauce Labs dashboard.

Video Demo