Instant Selenium Node.js Tests

PASSED

The demo scripts in this repository allow you to run simple automated tests in order to validate your JavaScript Selenium environment and your saucelabs.com account credentials.

> *For Demonstration Purposes Only*

> *The code in these scripts is provided on an "AS-IS" basis without warranty of any kind, either express or implied, including without limitation any implied warranties of condition, uninterrupted use, merchantability, fitness for a particular purpose, or non-infringement. These scripts are provided for educational and demonstration purposes only, and should not be used in production. Issues regarding these scripts should be submitted through GitHub. These scripts are maintained by the Technical Services team at Sauce Labs.*

Prerequisites

- Install Git
- Install NPM/Node
- Install a Framework
    - Mocha
    - Jasmine
- (Optional) Install an IDE

Install Git

Git is a version control system that lets you check out code from a repository, work with that code on your own branch, and then merge that code with any changes that have been made by other developers. Git is an essential tool for distributed development teams, and is a critical component of the continuous integration/continuous development toolchain.

MacOSX:

1. Go to https://git-scm.com/downloads.
2. Under **Downloads**, click **Mac OS X**.
3. When the download completes, double-click the `.dmg` file open the installer package.
4. Double-click the installer package to begin the installation.

> Security Warning
>
> *You may see a warning message that the package can't be opened because it's not from a recognized developer. If this happens, go to System Preferences > Security and Privacy Settings, and click Open Anyway.*

5. Click **Continue** for the installation, and enter your local password to authorize the installation.

Windows:

1. Go to https://git-scm.com/downloads
2. Under **Downloads**, click on **Windows**.
3. When the dialog opens asking if you want to allow the app to make changes to your device, click Yes.
4. Follow the steps in the setup wizard to complete the installation. You should accept all the default settings.

Install NPM and NodeJS

NPM is the primary package manager that will help to resolve dependency issues.

1. Go to https://www.npmjs.com/get-npm](https://www.npmjs.com/get-npm).
2. Click the button that reads "Download Node.js and NPM.
3. The site will detect which operating system you're currently running and present the requesite packages.
4. Choose LTS, and open up the package to run the installer.
5. Follow the prompts to complete the installation.
6. To ensure NPM and Node.js installed correctly, open a terminal/shell and run the following command:

```
node -v
npm -v
```

NodeJS lets you develop and deploy JavaScript applications on desktops and servers. If you which to download and install the package manually refer to the following instructions:

MacOSX:

1. Go to https://nodejs.org/en/download/.
2. Under **LTS**, click **Mac OS Installer X**.
3. When the download completes, double-click the `.pkg` file to open the installer
4. Follow the prompts to complete the installation.

Windows:

1. Go to https://nodejs.org/en/download/.
2. Under **LTS**, click **Windows Installer**.
3. When the download completes, double-click the `.msi` file to open the installer
4. Follow the prompts to complete the installation.

Install a Test Framework

A testing framework is a set of guidelines/rules for desiging test cases. Frameworks themselves consist of test and reporting classes that allow QA engineers to test efficiently. This repo only supports mocha, jasmine, and protractor. For more information regarding other libraries checkout our GitHub repo that catalogs all sample test frameworks.

Install Mocha

1. Open a terminal (OSX) or command prompt (Windows)
2. Navigate to the mocha-examples directory:

   ```
   cd ~/on-boarding-modules/mocha-examples
   ```

3. Resolve the package dependencies:

   ```
   npm install
   ```

4. Run tests:

   ```
   npm test
   ```

Install Jasmine

1. Open a terminal (OSX) or command prompt (Windows)
2. Navigate to the jasmine-examples directory:

   ```
   cd ~/on-boarding-modules/jasmine-examples
   ```

3. Resolve the package dependencies:

   ```
   npm install
   ```

4. Run tests:

   ```
   npm test
   ```

Install an IDE

It's recommended to install and Integrated Developer Environment, or a text editor, to help manage package dependencies, interperters, and overall code execution. There are several options available, some of them are free and some require payment:

- Atom IDE free, IDE developed by GitHub.
- WebStorm free trial, paid after 30 days, developed by JetBrains.
- Komodo Edit, free, text editor, stripped down version of Komodo IDE (paid version).
- Brackets free, text editor, contains a wealth of community plugins and extensions.

Sauce Labs Test Setup

This example script follows the best practice of using environment variables in place of hardcoded authentication credentials. Once you've been able to successfully run the instant test with hardcoded credentials, you can use this test to make sure that you've correctly set up your environment variables for authentication.

> *After the gitpod session launches, navigate to the terminal and run the following commands to save your Sauce Labs Credentials to gitpod as environment variables:*
>
> ```
> eval $(gp env -e SAUCE_USERNAME=******)
> eval $(gp env -e SAUCE_ACCESS_KEY=******)
> ```
>
> *Click the following link if you're unsure how to access your Sauce Labs credentials. Also, if you start a new terminal in gitpod, you have to run the following command to reset envrionment variables:*
>
> ```
> eval $(gp env -e)
> ```
>
> *For more information consult the gitpod documentation*

Run A Sample Test

1. Clone the Repository and set your Sauce Labs Credentials:

```
git clone https://github.com/saucelabs-training/demo-js.git
export SAUCE_USERNAME=********
export SAUCE_ACCESS_KEY=*******
```

2. Navigate to the desired directory (e.g.):

```
cd mocha-chai-tests
```

3. Resolve Dependencies (via `package.json`)

Choosing the Correct 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/

Running Local Tests

Developing websites/apps within your local network is secure and efficient. The drawback to this method is that local assets are not publicly-accessible on the Internet, so the browsers/devices in the Sauce Labs cloud can't load and test your app. The solution is to use Sauce Connect. Sauce Connect is a proxy server that creates a secure tunnel connection between the Sauce Labs virtual machine that runs your test and your local network. You can also use Sauce Connect to test applications and websites that are located within your corporate firewall. Sauce Connect is **not** required to run tests on Sauce Labs, only in situations where the application or website you want to test is not publicly accessible.

Running Tests in Parallel

Now that you're running tests on Sauce, you may wonder how you can run your tests faster. One way to increase speed is by running tests in parallel across multiple virtual machines.

You can run your tests in parallel at two levels, and you can run your tests in parallel across multiple browsers. For example, if you have 10 tests and want to run on five browsers, this would be parallelism of five. You can also run tests across browsers and each test in parallel. Using our previous example this would be more like 50 parallel tests. Doing this requires that your tests are written in a way that they do not collide with one another, as described in our Best Practice topics avoiding external test dependencies and avoiding dependencies between tests.

Check out Using Frameworks to Run Tests in Parallel for more information, and examples of framework setups for Java, Python, and other programming languages.

For more information, check out the example scripts in our GitHub repo.