Changes Suggested

Section-

 [**Getting Started**](https://deploy-preview-1479--selenium-dev.netlify.app/documentation/webdriver/getting_started/)

* [Install Library](https://deploy-preview-1479--selenium-dev.netlify.app/documentation/webdriver/getting_started/install_library/) -- this also needs change. There is an assumption that people will be aware of what maven is gradle is. Why do we need to mention project build tools to get started with Selenium ? Maybe we can simplify.
* [First Script](https://deploy-preview-1479--selenium-dev.netlify.app/documentation/webdriver/getting_started/first_script/) - rename required… ( in steps, we haven’t shown import statements for a programming language, that should be step 1, and then move further)

Should be kept Simple to launch a web browser- and close it. That’s all. As introduction to web element not done. No assertion ideally to be put here.

* [Upgrade to Selenium 4](https://deploy-preview-1479--selenium-dev.netlify.app/documentation/webdriver/getting_started/upgrade_to_selenium_4/) – This should be in some separate section, as in a blogg/ or some other main section to be made about Selenium versions and there such information could be kept for past/ present/future.
* [Using Selenium](https://deploy-preview-1479--selenium-dev.netlify.app/documentation/webdriver/getting_started/using_selenium/) – Should itself be a main section.

(<https://deploy-preview-1479--selenium-dev.netlify.app/documentation/webdriver/getting_started/using_selenium/>)

**Organizing and Executing Selenium Code**

**(Using Selenium**

~~Scaling Selenium execution with an IDE and a Test Runner library~~

~~If you want to run more than a handful of one-off scripts, you need to be able to organize and work with your code. This page should give you ideas for how to actually do productive things with your Selenium code.~~

The power which Selenium brings to automate the browser, can be leveraged in various use cases for Selenium. Most prominent, and one of the main reason Selenium came into being was to find a way to automate browser for testing purpose. But it isn’t restricted just to that. Administration work, creating RPA’s, test frameworks, Web Scraping(???) there are many such ways people have been using Selenium.

Common Uses

Most people use Selenium to execute automated tests for web applications~~, but Selenium support any use case of browser automation.~~

Testing

To use Selenium to facilitate and fasten the testing of web applications, we would require –

1. Selenium Language Bindings
2. Assertion library depending on the language

Please note there are many solutions available for options We will be restricting this document to following selection per programming language

--show tab

--java – junit/

--csharp – nunit/

-- pyunit

Running Selenium for testing requires making assertions on actions taken by Selenium. So a good assertion library is required. Additional features to provide structure for tests require use of [Test Runner](https://deploy-preview-1479--selenium-dev.netlify.app/documentation/webdriver/getting_started/using_selenium/#test-runners)

IDEs

Regardless of how you use Selenium code, you won’t be very effective writing or executing it without a good Integrated Developer Environment. Here are some common options…

* Eclipse
* IntelliJ
* PyCharm
* RubyMine
* Rider
* WebStorm
* VS Code

Test Runner

Even if you aren’t using Selenium for testing, if you have advanced use cases, it might make sense to use a test runner to better organize your code. Being able to use before/after hooks and run things in groups or in parallel can be very useful.

Choosing

There are many different test runners available.

All the code examples in this documentation can be found in (or is being moved to) our example directories that use test runners and get executed every release to ensure all the code is correct and updated. Here is a list of test runners with links. The first item is the one that is used by this repository and the one that will be used for all examples on this page.

* Java
* Python
* CSharp
* Ruby
* JavaScript
* Kotlin
* JUnit 5

Installing

This is very similar to what was required in [Install a Selenium Library](https://deploy-preview-1479--selenium-dev.netlify.app/documentation/webdriver/getting_started/install_library/)

* Java
* Python
* CSharp
* Ruby
* JavaScript
* Kotlin

Popular test runners for Java are [JUnit] and [TestNG]

**Maven**

**Gradle**

Asserting

* Java
* Python
* CSharp
* Ruby
* JavaScript
* Kotlin

[Add Example](https://deploy-preview-1479--selenium-dev.netlify.app/documentation/about/contributing/#creating-examples)

Setting Up and Tearing Down

* Java
* Python
* CSharp
* Ruby
* JavaScript
* Kotlin

[Add Example](https://deploy-preview-1479--selenium-dev.netlify.app/documentation/about/contributing/#creating-examples)

Executing

* Java
* Python
* CSharp
* Ruby
* JavaScript
* Kotlin

Maven

mvn clean test

Copy

Gradle

gradle clean test

Copy

Examples

In [First script](https://deploy-preview-1479--selenium-dev.netlify.app/documentation/webdriver/getting_started/first_script/), we saw each of the components of a Selenium script. Here’s an example of that code using a test runner:

* Java
* Python
* CSharp
* Ruby
* JavaScript
* Kotlin

package dev.selenium.getting\_started;

import org.junit.jupiter.api.Test;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import java.time.Duration;

import static org.junit.jupiter.api.Assertions.assertEquals;

public class UsingSelenium {

 @Test

 public void eightComponents() {

 WebDriver driver = new ChromeDriver();

 driver.get("https://www.selenium.dev/selenium/web/web-form.html");

 String title = driver.getTitle();

 assertEquals("Web form", title);

 driver.manage().timeouts().implicitlyWait(Duration.ofMillis(500));

 WebElement textBox = driver.findElement(By.name("my-text"));

 WebElement submitButton = driver.findElement(By.cssSelector("button"));

 textBox.sendKeys("Selenium");

 submitButton.click();

 WebElement message = driver.findElement(By.id("message"));

 String value = message.getText();

 assertEquals("Received!", value);

 driver.quit();

 }

}

Copy

[**View full example on GitHub**](https://github.com/SeleniumHQ/seleniumhq.github.io/blob/fw/examples/java/src/test/java/dev/selenium/getting_started/UsingSeleniumTest.java)

Next Steps

Take what you’ve learned and build out your Selenium code!

As you find more functionality that you need, read up on the rest of our [WebDriver documentation](https://deploy-preview-1479--selenium-dev.netlify.app/documentation/webdriver/).

Last modified September 15, 2023: [rename page and add section on Usage (16157e32dd)](https://github.com/SeleniumHQ/seleniumhq.github.io/commit/16157e32dd9c210d58f6293748acbd027d701d84)