

**Chrome** provides a driver, which can establish the connection between **Selenium WebDriver** & **Google Chrome** and run the **Selenium test in Chrome browser**. Let's comprehend the details and usage of the *Selenium ChromeDriver* to run the automated test on the *Chrome browser* by cover the details under the following sections:

*What is Selenium ChromeDriver?*

*What are the pre-requisites for Selenium ChromeDriver?*

*How to install ChromeDriver on Windows?*

*How to download ChromeDriver on Windows?*

*And how to setup ChromeDriver on Windows?*

*And how to run Selenium tests on Chrome Browser?*

*How to install ChromeDriver on macOS?*

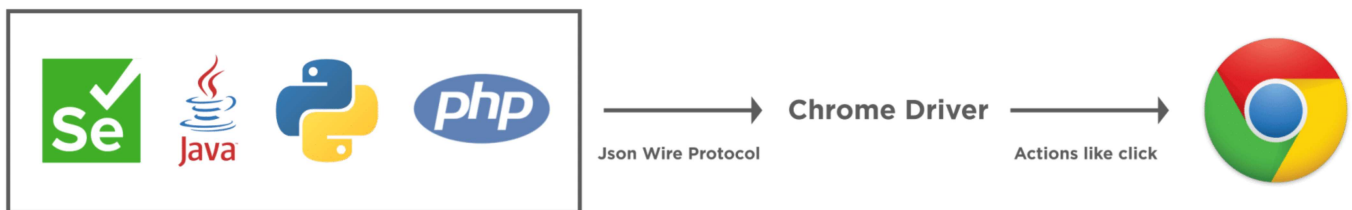
*How to download ChromeDriver on macOS?*

*And how to setup ChromeDriver on macOS?*

*How to install ChromeDriver using Homebrew?*

## What is Selenium ChromeDriver?

**ChromeDriver** is the communication medium that allows users to run their Selenium tests on Chrome browser. It is a *standalone server* that implements the open-source **Selenium WebDriver Chromium Protocol**. The *Selenium* tests interact with the **ChromeDriver** using the **JsonWireProtocol**, which translates the *Selenium* commands into corresponding actions on the *Chrome* browser.



The sole purpose of the *ChromeDriver* is to launch and interact with **Google Chrome**. Without using *ChromeDriver*, it's not possible to run **Selenium** tests on the chrome browser. It is why it is one of the most vital pre-requisite of the test execution on Chrome. A *ChromeDriver* can be used easily by instantiating the object of the *ChromeDriver*, assigning it to a *WebDriver* object, and using that object for browser-based actions.

***What are the pre-requisites for Selenium ChromeDriver?***

Before we can start writing *Selenium* tests or setup *ChromeDriver*, there are few pre-requisites that we should have on our system:

- . **Java JDK:** We require **JDK** or Java Development Kit for writing java programs. It contains **JRE** and other development tools, including compiler and debugger. As we will be writing our selenium tests in java, having JDK is a must. You can get more information about JDK and read its installation guide from here: [How to install Java?](#)
- . **Java IDE:** IDE or Integrated Development Environment helps in writing Java programs. It offers many different features to users to ease their programming requirements. For this tutorial, we will be using Eclipse IDE, although any other Java IDE will be perfectly fine. To more or know how to install Eclipse, visit here: [Install Eclipse](#).
- . **Selenium WebDriver:** To develop Selenium tests, we need Selenium WebDriver. You can download Selenium WebDriver from the [official Selenium site](#), and you can learn how to configure Selenium in the tutorial; [Configure Selenium WebDriver](#). For this tutorial, we will be using **Selenium 4**.

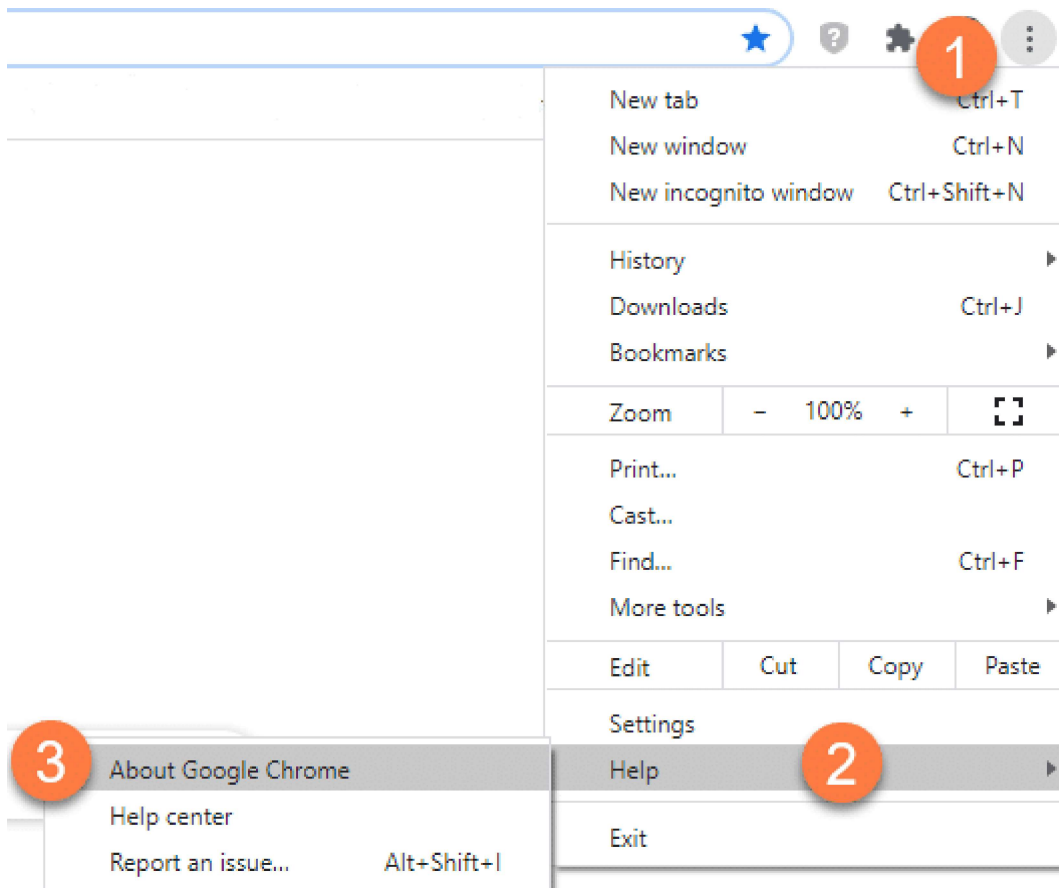
## How to install ChromeDriver on Windows?

Now, as we have learned what *ChromeDriver* is and why do, we need it for executing Selenium tests on the chrome browser. Let's move a little further and learn how to setup *ChromeDriver* with your *Selenium Java project* and execute your tests on Chrome. The first part will be to download the *ChromeDriver*. Let's see how we can do the same on the Windows platform?

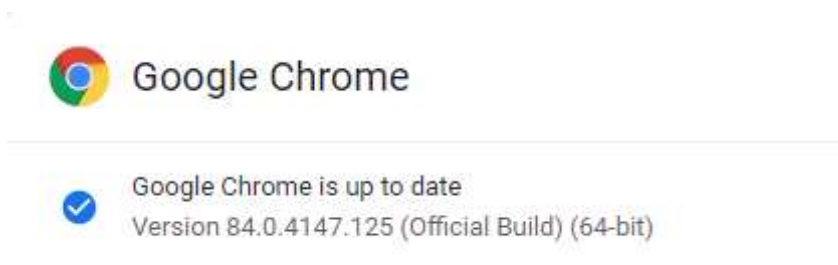
### *How to download ChromeDriver on Windows?*

Before we can download the *ChromeDriver*, we need to check the version of the *Chrome browser* on your system. *ChromeDriver* has a direct compatibility dependency on the Chrome browser version, so you need to download the compatible version of *ChromeDriver*. Follow the steps, as mentioned below, to download a. *ChromeDriver* which is compatible with the *Chrome* browser on your system:

- . Firstly, to check the Chrome browser version on your machine, click on the **three dots** on the right top corner of the browser
- . Secondly, click on **Help** in the menu.
- . Thirdly, click on **About Google Chrome** in the sub-menu.



. After clicking the "**About Google Chrome**" option, the following page will open. Consequently, you will get the Chrome version details as in the image below:



Now, as we have got the *Chrome* browser version so, we can download the compatible *ChromeDriver*. Additionally, to download *ChromeDriver*, navigate to the link of the official [ChromeDriver website](#). Follow the steps as mentioned below to download the *ChromeDriver* executable file:

. On the [ChromeDriver download page](#), there will be links for different *ChromeDriver* version. Based on your Chrome browser version, download the corresponding *ChromeDriver*, as marked in the below image. Subsequently, click on the *ChromeDriver* version that you need to download. As we had the Chrome browser version as "**84**", so we will download the corresponding *ChromeDriver*.

# Downloads

## Current Releases






- If you are using Chrome version 85, please download [ChromeDriver 85.0.4183.38](#)
- If you are using Chrome version 84, please download [ChromeDriver 84.0.4147.30](#)
- If you are using Chrome version 83, please download [ChromeDriver 83.0.4103.39](#)
- For older version of Chrome, please see below for the version of ChromeDriver that supports it.

If you are using Chrome from Dev or Canary channel, please following instructions on the [ChromeDriver Canary](#) page.

For more information on selecting the right version of ChromeDriver, please see the [Version Selection](#) page.

Secondly, clicking on the "**ChromeDriver 84.0.4147.30**" link will take you to the *ChromeDriver* index page. Here, you will get different options of *ChromeDriver* based on your operating system. Additionally, for the Windows operating system, you can choose the *Win32* version as marked in the below image. Yes, even if you have a 64-bit Windows installed on your system, the *Win32* version will work fine.

## Index of /84.0.4147.30/

	<a href="#">Name</a>	Last modified	Size	ETag
	<a href="#">Parent Directory</a>	-	-	-
	<a href="#">chromedriver_linux64.zip</a>	2020-05-28 21:05:07	5.06MB	beffb1bca07d8f4fd23213b292ef963b
	<a href="#">chromedriver_mac64.zip</a>	2020-05-28 21:05:09	6.99MB	b2ff30e148ae11a78e0f13e93b29f271
	<a href="#">chromedriver_win32.zip</a>	2020-05-28 21:05:11	4.63MB	3bf0e106a93382efd7a5bb3b55b182a6
	<a href="#">notes.txt</a>	2020-05-28 21:05:15	0.00MB	a505de7f878e415f1b06a44935f109bf

Thirdly, once the download is complete, extract the zip file and place the "**chromedriver.exe**" at any preferred location on your system.

Now that we have downloaded the *ChromeDriver*, we will open Eclipse and create a new java project. Moreover, we will add all the selenium dependencies to the project. Additionally, to know more about setting up Selenium with Eclipse, you can visit our previous tutorial on the same at [Configure Selenium WebDriver](#).

As a next step, we need to make the *downloaded ChromeDriver executable* available to the Selenium tests. Subsequently, let's see how we can setup *ChromeDriver*, so as we can use the same in the Selenium test cases:

## ***How to setup ChromeDriver on Windows?***

To set up and configure the *ChromeDriver* with the Selenium, the *ChromeDriver*executable file should be accessible in the test script. Additionally, Selenium tests can access the *ChromeDriver* if any of the following ways sets it up:

- . *Setup ChromeDriver using System Properties in Environment Variables.*
- . *Setup ChromeDriver using System Properties in the test script.*

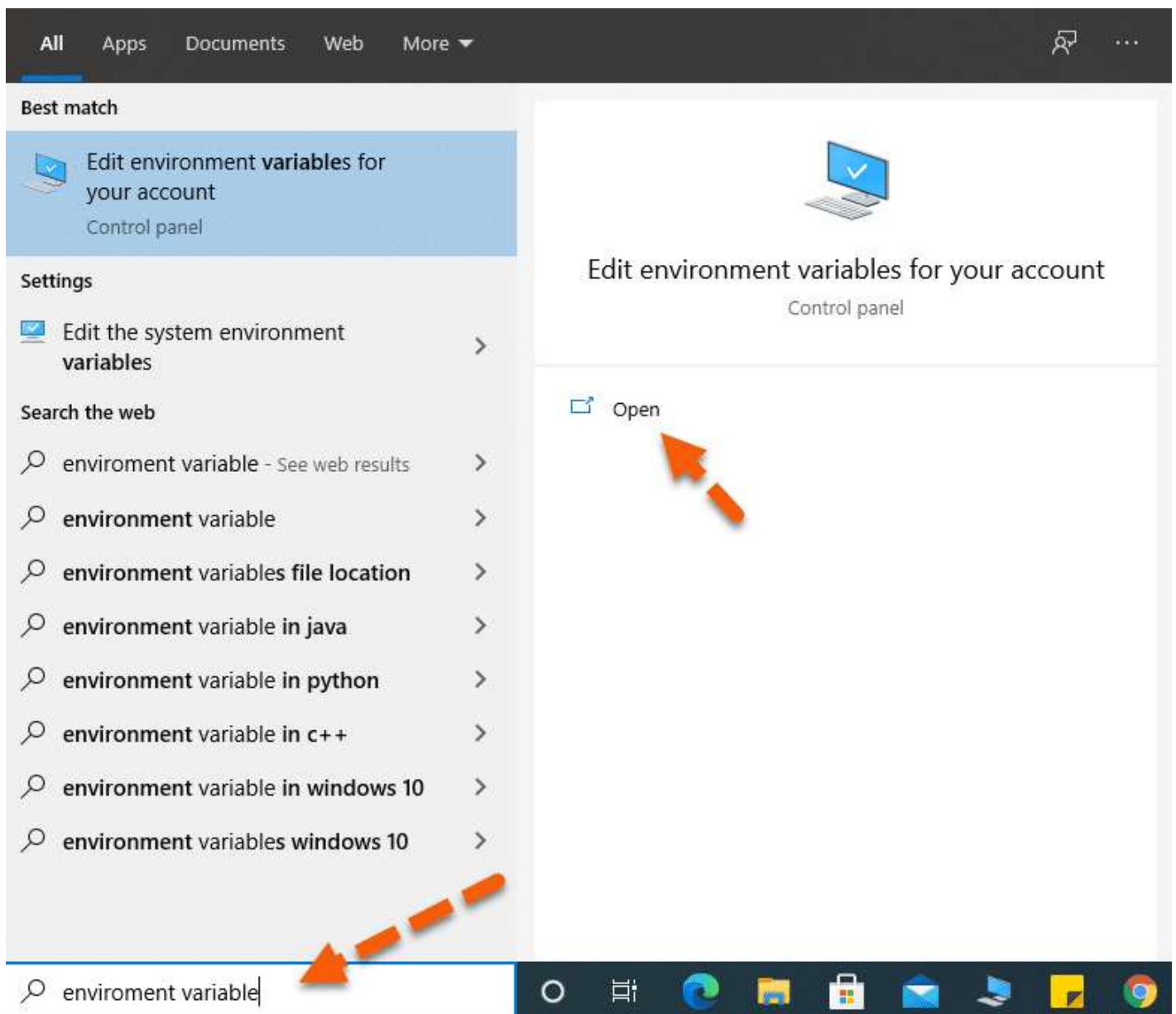
Let's understand all of these and try running our test code with **Selenium 3** or **Selenium 4**.

### ***How to setup ChromeDriver using System Properties in Environment Variables?***

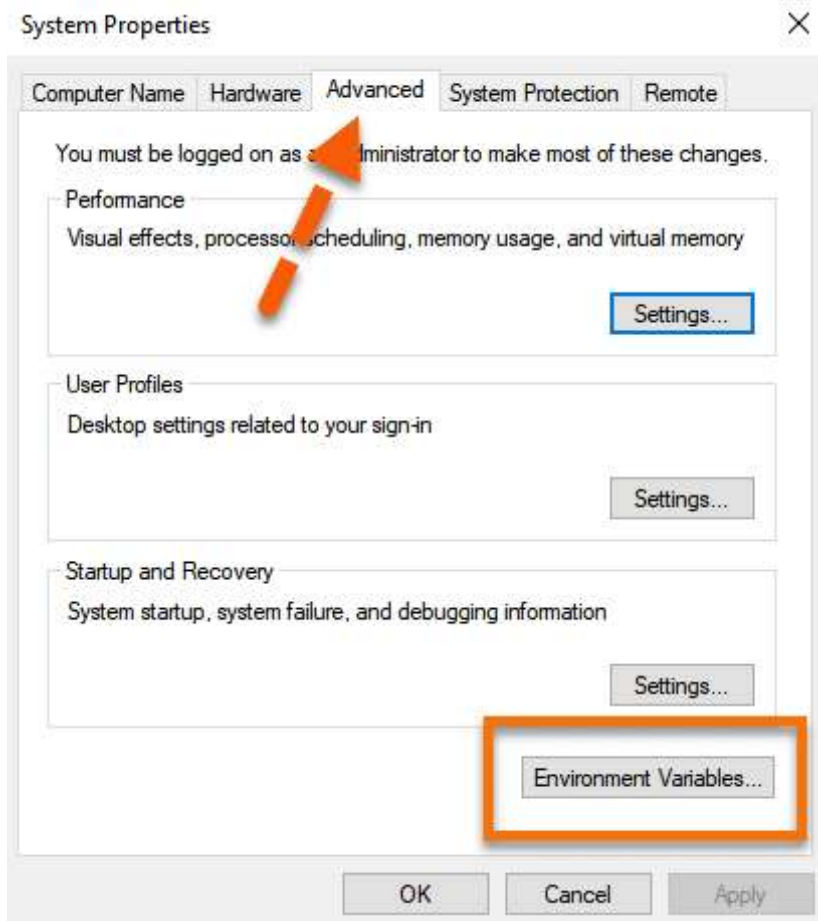
On the Windows operating system, one of the ways to declare system-level variables is by using **Environment Variables**. Users can define either *user-level environment variables* or *system variables*. Moreover, the variable defined here is accessible to all the programs running on the system. We can use the environment variables to set the path of the *ChromeDriver*. So, whenever we create an instance of the *WebDriver*, it will automatically detect the path of the *ChromeDriver* from the system variables and can use the same. Subsequently, let's have a look at steps through which we can do that.

- . First, we need to open the Environment Variable pop-up. To do that, click on the search bar and search for "**Environment Variables**". It will search and display "**Edit environment variables for your account**", as shown in the image below. After that, click on the "**Open**" to open the System Properties pop-up.

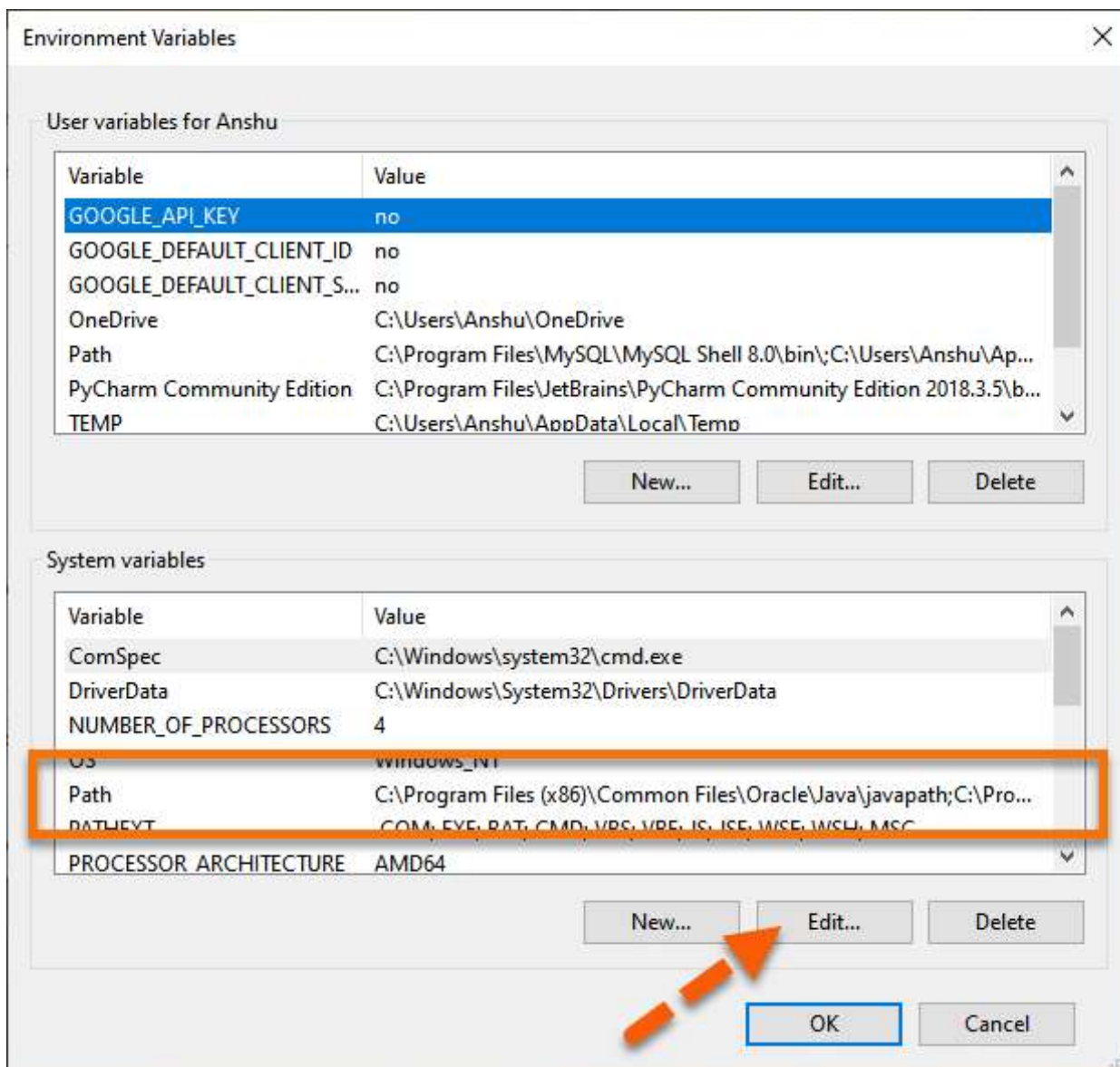




- . Secondly, the "**System Properties**" pop-up will open. In the pop-up, select the "**Advanced**" tab as marked by the arrow. After that, in the Advanced tab, click on the "**Environment Variables**" button.

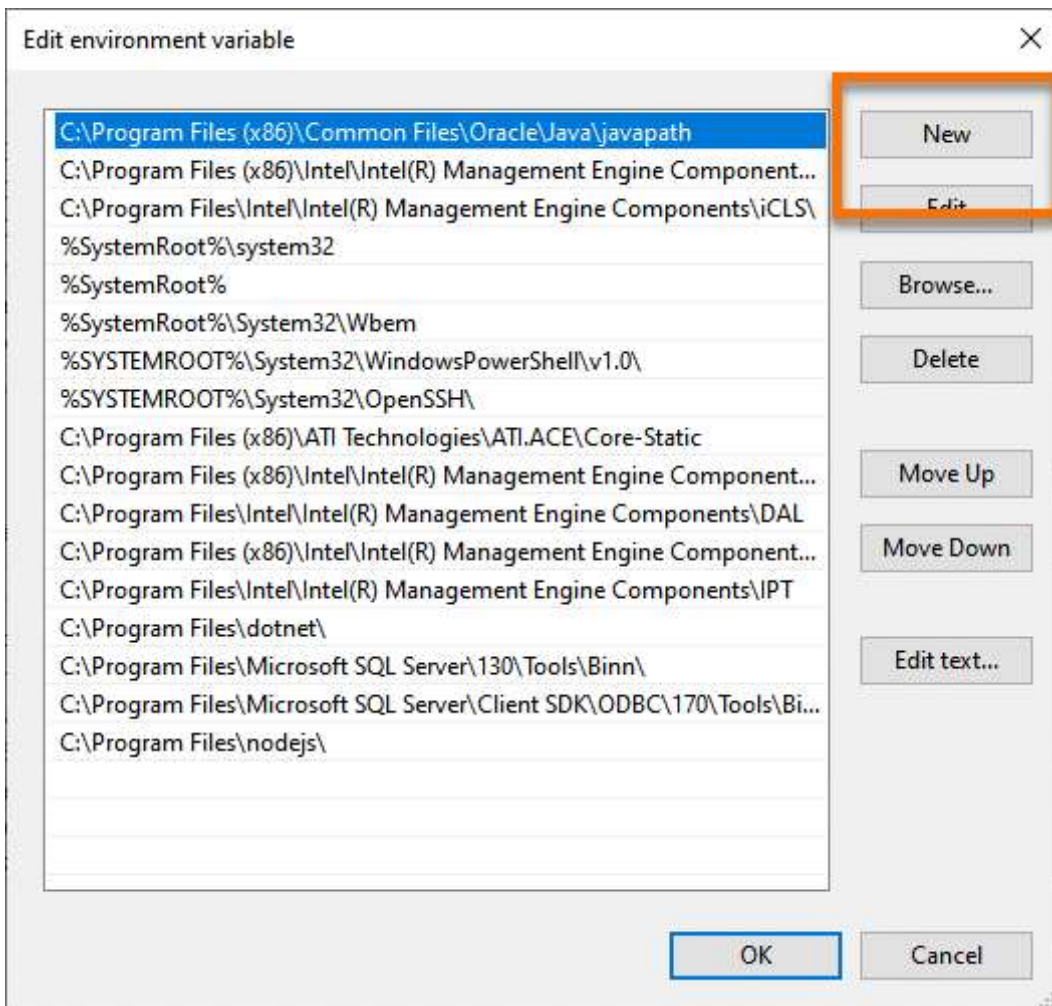


Thirdly, this will open the "**Environment Variables**" pop-up. In the pop-up System variables section, look for the "**path**" variable marked in the below image. After that, click on the path variable to select it. Once selected, click on the "**Edit**" button as marked by the arrow.

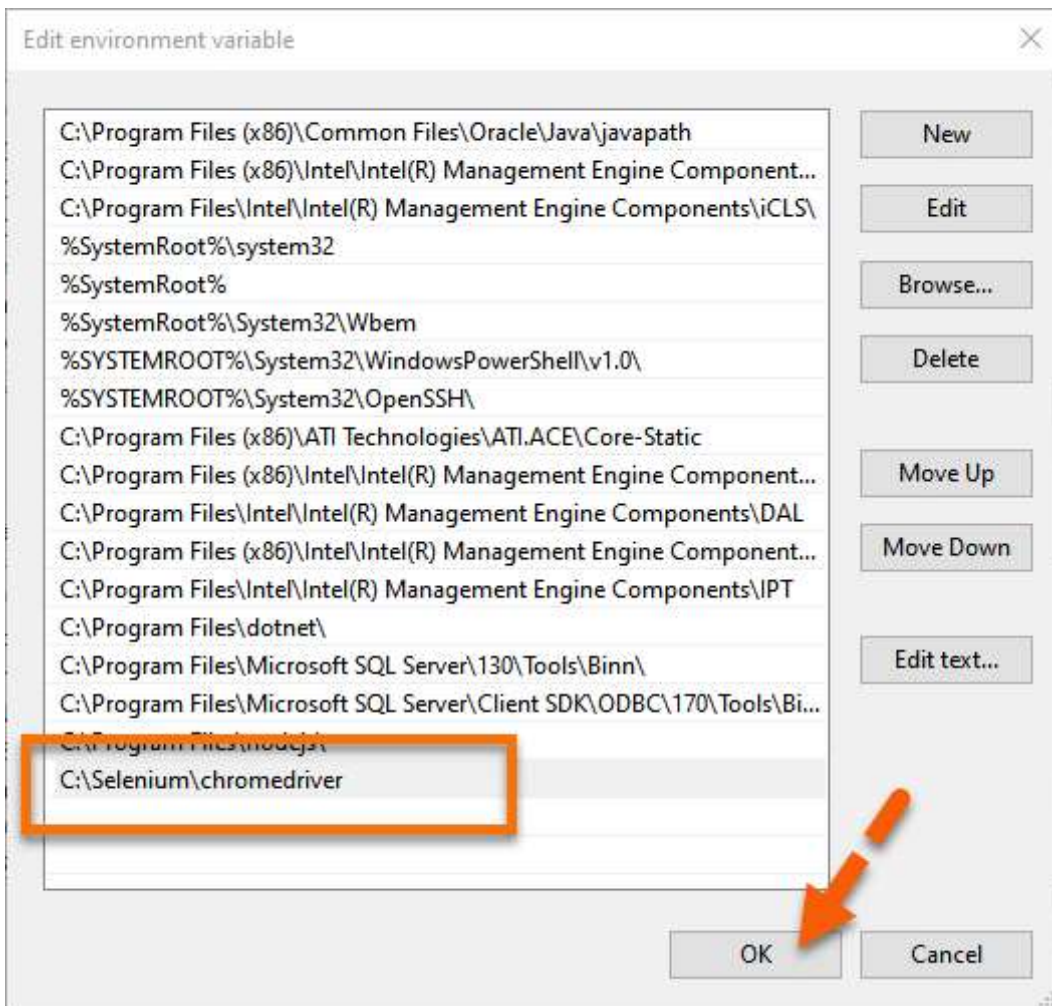


. Fourthly, once the "**Edit environment variable**" pops-up, click on the "**New**" button.





- Fifthly, add the ChromeDriver's folder location to the path. We have placed our driver at the following location "**C:\Selenium\chromedriver**", so we have added the same as the path variable. Once done, click on the "**OK**" button as denoted by the arrow.



## How to run Selenium tests on Chrome Browser using ChromeDriver?

Conclusively, we can now directly initialize the WebDriver instance using the *ChromeDriver*, as shown below:

```
package demoPackage;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class ChromeDriverDemo {
    public static void main(String[] args) throws InterruptedException{

        System.out.println("Execution after setting ChromeDriver path in System
        WebDriver driver=new ChromeDriver();
        driver.get("https://demoqa.com");
        Thread.sleep(3000);
        driver.quit();
        System.out.println("Execution complete");

    }
}
```

On executing the above code, you will see the results below.

```
Execution after setting ChromeDriver path in System Variables
Starting ChromeDriver 84.0.4147.30 (48b3e868b4cc0aa7e8149519690b6f6949e110a8-refs/branch-heads/4147@{#310}) on port 9655
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
[1600254750.877][WARNING]: FromSocketAddr failed on netmask
ChromeDriver was started successfully.
[1600254752.649][WARNING]: This version of ChromeDriver has not been tested with Chrome version 85.
Sep 16, 2020 4:42:32 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: W3C
Execution complete
```

Evidently from the console results, there is no *WebDriver* error, which implies that the *WebDriver* set up is correct. Moreover, you can see the print statements as the entry and exit points of our execution. Correspondingly you will be able to view the execution in your system.

### ***How to initialize ChromeDriver using System Properties in the Selenium test script?***

Instead of using the global instance of *ChromeDriver*, if we want to use a specific version of *ChromeDriver*, we can do the same by explicitly specifying the path of the *ChromeDriver* in the test script itself. In other words, we need to add a single line of code to set up the system properties for the *ChromeDriver*, as shown below:

```
System.setProperty("webdriver.chrome.driver", "<Path of the ChromeDriver Executable>");
```

Conclusively, let us modify the code we used above and see that we can launch the Chrome browser successfully. The modified code would look like this:

```
package demoPackage;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class ChromeDriverDemo {
    public static void main(String[] args) throws InterruptedException{

        System.out.println("Execution after setting ChromeDriver path in System
        System.setProperty("webdriver.chrome.driver", "E:\\drivers\\ChromeDriver
        WebDriver driver=new ChromeDriver();
        driver.get("https://demoqa.com");
        Thread.sleep(3000);
        driver.quit();
        System.out.println("Execution complete");

    }

}
```

You will see that [demoqa.com](https://demoqa.com) opens in the *Chrome* browser without any error and exception.

```
Execution after setting ChromeDriver path in System setProperty method
Starting ChromeDriver 85.0.4183.87 (cd6713ebf92fa1cacc0f1a598df280093af0c5d7-refs/branch-heads/4183@{#1689}) on port 9469
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
[1600255618.884][WARNING]: FromSocketAddr failed on netmask
ChromeDriver was started successfully.
Sep 16, 2020 4:57:00 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: W3C
Execution complete
```

If you noticed, here we used *ChromeDriver* version 85, instead of the default global *ChromerDriver* version of 84. The execution logs indicate that our *WebDriver* session started with the print statement displayed right at the beginning. The lines highlighted in red are some browser logs corresponding to the browser session. Moreover, you can see the browser opening up in your system, and after the website opens, the browser session is closed.

## How to install ChromeDriver on macOS?

The installation and setup of *ChromeDriver* on macOS is almost the same as that of the *Windows* platform. The only difference being the executable for *macOS* will be different, and the way we can include the *ChromeDriver* executable in the System's *PATH* variable is a bit different. Let's see how we can install and setup the *ChromeDriver* on macOS :

### How To Download ChromeDriver on macOS?

On *macOS*, we can download *ChromeDriver* using any of the following ways:

Download executable file from the [Chromium website](https://chromedriver.chromium.org)  
Download using any package manager, such as [Homebrew](https://brew.sh/).






Let's understand the details of both of these ways of downloading and setting up *ChromeDriver* on *macOS* :

### How to download ChromeDriver for macOS from the Chromium website?

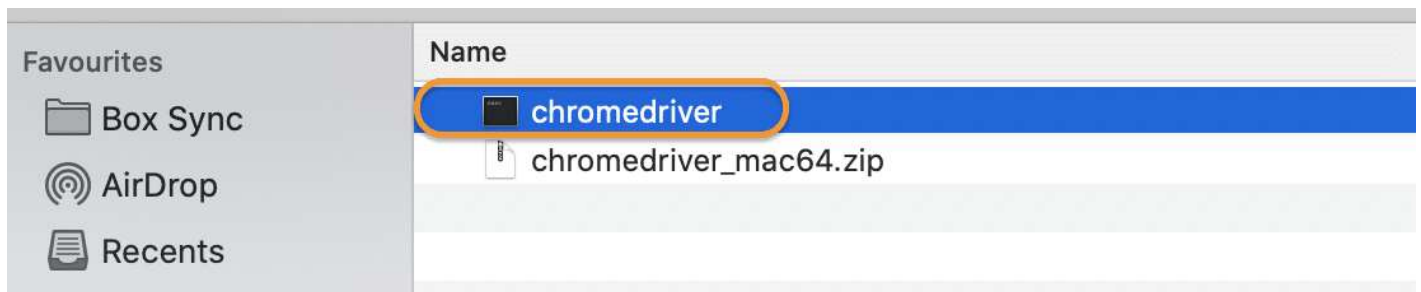
You can download the *ChromerDrive* for *macOS*, same as we did for the *Windows* platform, except for the difference that, now select the binary for the *macOS* platform as shown below:



# Index of /85.0.4183.87/

Name	Last modified	Size	ETag
 <a href="#">Parent Directory</a>		-	
 <a href="#">chromedriver linux64.zip</a>	2020-08-27 20:11:44	5.11MB	49513e90656c825f5acadfb917bb7840
 <a href="#">chromedriver mac64.zip</a>	2020-08-27 20:11:46	7.36MB	6f6a70e5ee4b9df114558323cd3add90
 <a href="#">chromedriver win32.zip</a>	2020-08-27 20:11:48	4.96MB	d8dd294b56d92add7cafea4afb899359
 <a href="#">notes.txt</a>	2020-08-27 20:11:52	0.00MB	734a3176845469073a9ef15f114ff326

It will download a *zip* file, which you can extract in any of the folders of your choice. After extracting, it will show the executable file of *ChromeDriver*, as shown below:



So, now you have the *ChromeDriver* executable file available on your machine, which we can use in our test scripts. Subsequently, let's see how to setup *ChromeDriver* on macOS and use in the Selenium test scripts:

## How To Set Up ChromeDriver on macOS?

Now that you have downloaded the *ChromeDriver*, the next step is to set it up to use it in your test scripts. On macOS also, we can follow the same ways, as on Windows, to set up the *ChromeDriver*:

- . Setup *ChromeDriver* using the System's *PATH* variable.
- . Setup *ChromeDriver* using System Properties in the test script.

The 2nd point is the same setup as the Windows platform, as we are using *JAVA* for test development, and *JAVA* being platform-independent, will have the same behavior across platforms. So, let's see how we can set up the *ChromerDriver* using the System's *PATH* variable:

### How to Setup ChromeDriver using the System's PATH variable?

As we mentioned above, one of the easiest ways to make the executable available globally on the *macOS* is to copy the executable under any the folders which are already in the *PATH* variable. Let's follow the steps mentioned below to achieve the same:

First, identify the folders included in the *PATH* variable using the command '*echo \$PATH*' on the terminal. It will give a sample output, as shown below:

```
echo $PATH
/bin:/sbin:/usr/bin:/usr/local/sbin:/usr/local/bin:/Library/Java/JavaVirtualMachines/jdk1.8.0_181.jdk/Contents/Home:/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin
```

- . Secondly, as we can see, multiple directories are already part of the *PATH* variable. Suppose we choose "*/usr/local/bin*" as a placeholder directory to hold the *ChromeDriver* executable.
- . Thirdly, copy the *ChromeDriver* executable file from the downloaded directory to the "*/usr/local/bin*" directory using the *mv* command as shown below:

```
mv chromedriver /usr/local/bin/
```

Now your *ChromeDriver* is ready to be used in your *Selenium test scripts*. Consequently, now we will write a simple program and execute the same in the *macOS* platform.

```
package demoPackage;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class ChromeDriverDemo {
    public static void main(String[] args) throws InterruptedException{

        System.out.println("ChromeDriver execution on mac!!");
        WebDriver driver=new ChromeDriver();
        driver.get("https://demoqa.com");
        Thread.sleep(3000);
        driver.quit();
        System.out.println("Execution complete on macOS");

    }
}
```

On executing the same, you can find the results in your console window:



ChromeDriver execution on macOS!!

SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".

SLF4J: Defaulting to no-operation (NOP) logger implementation

SLF4J: See <http://www.slf4j.org/codes.html#StaticLoggerBinder> for further details.

Starting ChromeDriver 84.0.4147.30 (48b3e868b4cc0aa7e8149519690b6f6949e110a8-refs/branch-heads/4147@{#310}) on port 16294  
Only local connections are allowed.

Please see <https://chromedriver.chromium.org/security-considerations> for suggestions on keeping ChromeDriver safe.

ChromeDriver was started successfully.

[1600420787.825][WARNING]: This version of ChromeDriver has not been tested with Chrome version 85.

Sep 18, 2020 2:49:47 PM org.openqa.selenium.remote.ProtocolHandshake createSession

INFO: Detected dialect: W3C

Execution complete on macOS

You can see the execution happening successfully without any error. Both the print statements are getting displayed, which indicates that our execution did not face any error. So did you see how easy it was to run\* *ChromeDriver*\* tests in *macOS*? Unlike the Windows system, where you have to remember the path of your driver executable, just placing the driver at a location in *macOS* makes our lives so easy!

## How to install ChromeDriver using Homebrew?

**Homebrew** is one of the package managers available on *macOS*, which can download any binaries which register as a package with *Homebrew*. Luckily *ChromeDriver* is available as a Homebrew package, and we can download and set up the same, with a straightforward command as below:

```
brew cask install chromedriver
```

When we run the above command, it will download and install the *ChromeDriver* in the ***"/usr/local/bin"*** directory, as can be validated from the following output of the above command:

```
=> Downloading https://chromedriver.storage.googleapis.com/85.0.4183.87/chromedriver_mac64.zip
##### 100.0%
=> Verifying SHA-256 checksum for Cask 'chromedriver'.
=> Installing Cask chromedriver
=> Linking Binary 'chromedriver' to '/usr/local/bin/chromedriver'.
🍺 chromedriver was successfully installed!
```

As we can see that the latest stable version of *ChromeDriver* was downloaded and installed in the ***"/usr/local/bin"*** directory, which makes it part of the *PATH* variable and accessible to all the applications on the system.

So, this way, *ChromeDriver* installed and setup on *macOS* with a single command. Other package managers, such as NPM, also provides the capabilities to install *ChromeDriver*, which can be

explored based on which package manager you are using on your machine.