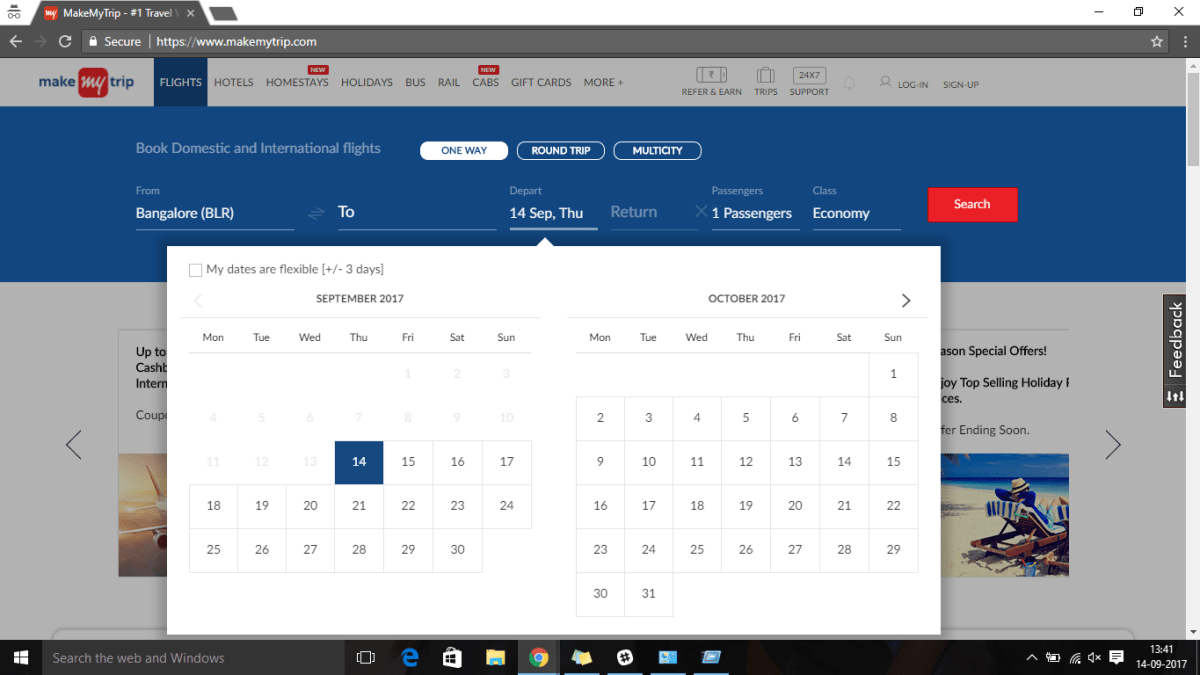
*Hello Folks,*

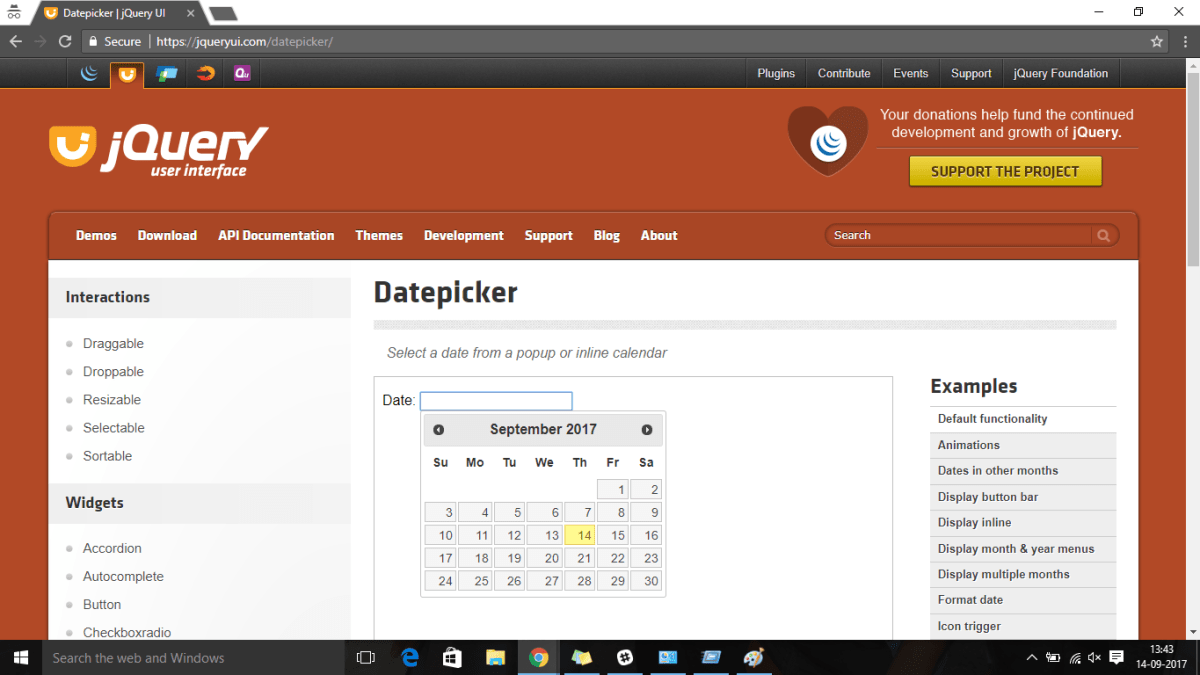
*In this post we will see a very important topic “Handling calendar in Selenium”.*

* We always see some input boxes where we need to select date from calendar widget. When we click on that input box, a calendar widget gets open and we need to select proper date value from that widget.
* You can see different types of calendar widget varying across applications. For example:

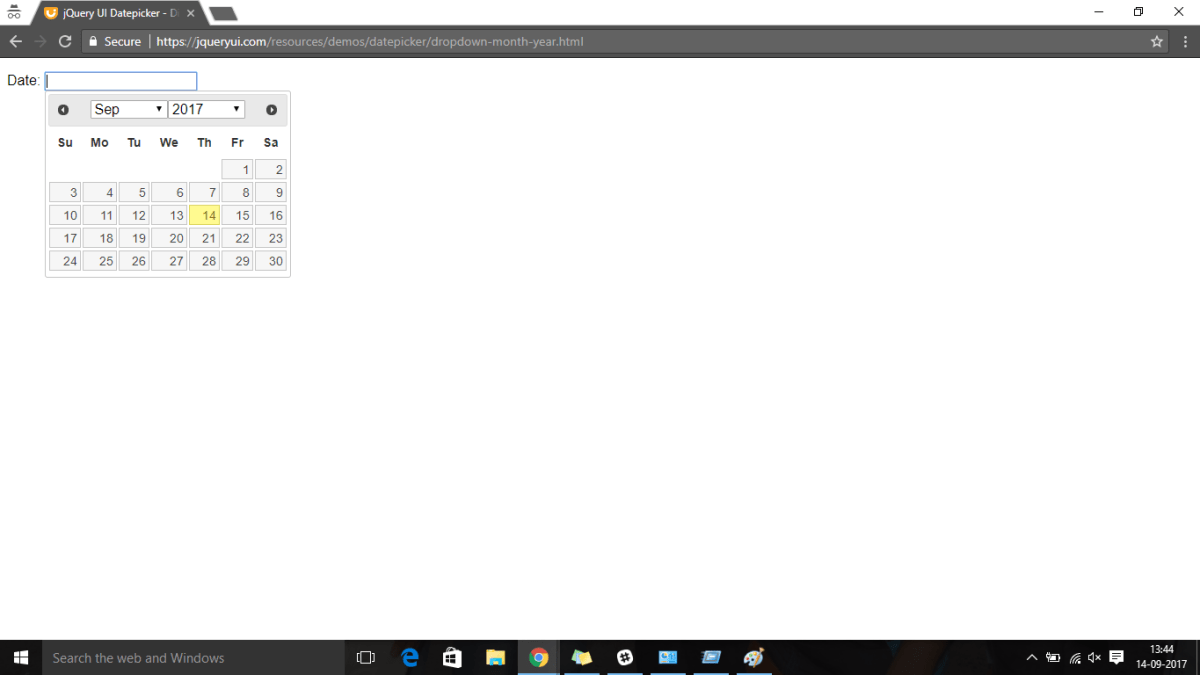
*MakeMyTrip.com:*

**

*Jquery Type 1:*

**

*Jquery Type 2:*

**

* Since every application uses different types of calendars, so it becomes little confusing and difficult to Selenium people to select date from it. You can not write a reusable method easily which can be used to handle all types of calendar.
* Examples shown above are mostly used in applications. So I will explain handling of above calendars in this post. You can use the same logic for other types of calendar as well.

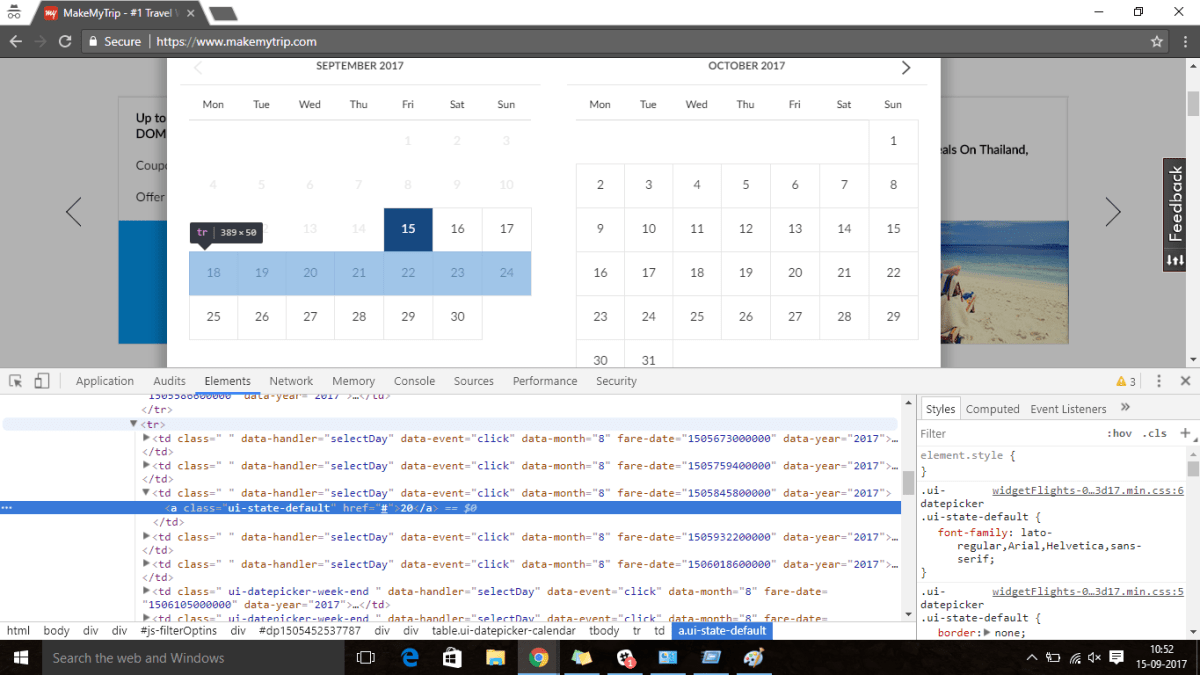
*Table of Contents*

* [Understanding Calendar HTML:](http://makeseleniumeasy.com/2017/09/18/how-to-handle-different-types-of-calendars-in-selenium-part-1/" \l "Understanding_Calendar_HTML" \o "Understanding Calendar HTML:)
* [Handling MakeMyTrip’s calendar:](http://makeseleniumeasy.com/2017/09/18/how-to-handle-different-types-of-calendars-in-selenium-part-1/" \l "Handling_MakeMyTrip%E2%80%99s_calendar" \o "Handling MakeMyTrip’s calendar:)
* [Author: Amod Mahajan](http://makeseleniumeasy.com/2017/09/18/how-to-handle-different-types-of-calendars-in-selenium-part-1/" \l "Author_Amod_Mahajan" \o "Author: Amod Mahajan)

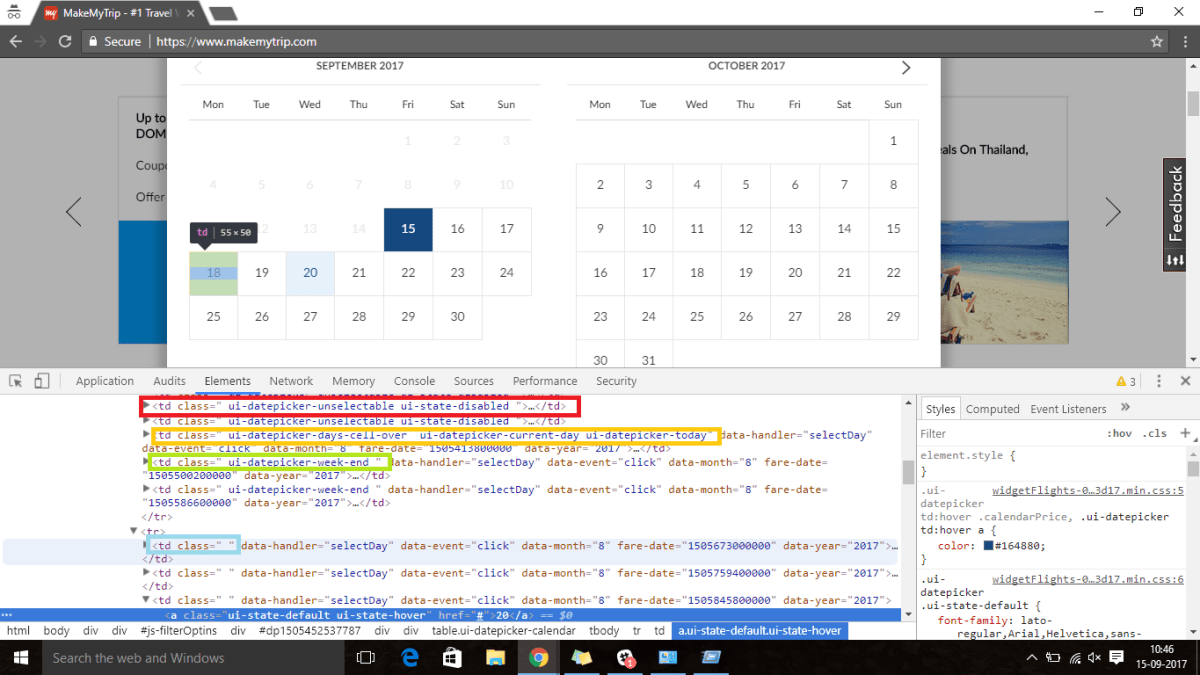
**Understanding Calendar HTML:**

*There are so many types of calendar but there are something common across all types. We must know those common points about calendars.*

* Maximum calendars are a table which consists of rows and tables. Generally week will be created with “tr” and dates will “td”.

**

* “Class” attribute for “td” tag gives more information about if date is past, current , weekend or upcoming. Refer below image, You will see the class name is different.
* Red marked column is past date.
* Yellow marked column is current date.
* Green marked column is week-end.
* Blue marked column is future date.

**

* Every td will have attributes which is “data-month” and “data-year” which gives information about month and year to which td belongs. For example: “data-month” as 2 means day belongs to March month.

**Handling MakeMyTrip’s calendar:**

*Logic:*

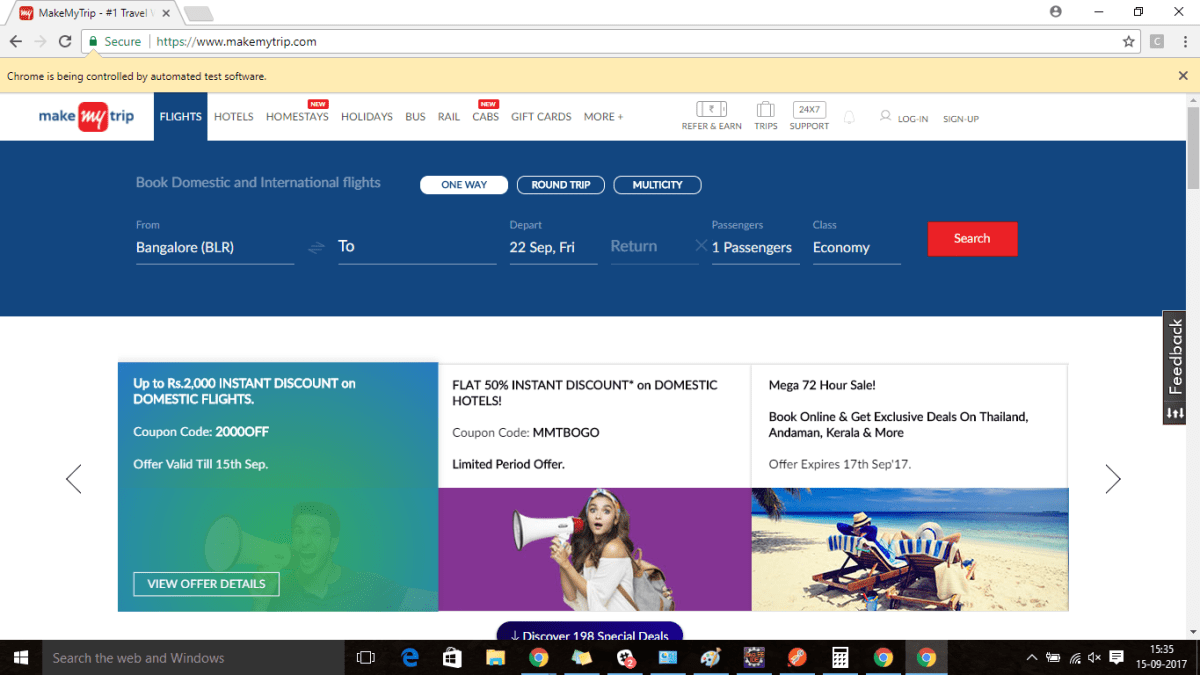
1. We will take Year, Month and day from user.
2. First we will get year in first month(left hand side). If year is not matching with desired year, we will click on next till we get desired year. For example: If I need to select 2018, and current year is 2017, we will keep clicking of next button till we get year as 2018.
3. Once we get desired year, we need to get desired month. We will retrieve month value from left most month and if month is not matching with desired month, we will keep clicking on Next button till we get desired month. For example: If current month is September and desired month is November. We will click on Next button till we get month November in left most month.
4. The major step is finding valid dates of desired month. We should filter past dates and dates of other months. We should also filter empty date columns.
5. Java has a number for each month of year which is equal to actual position of month-1. For example: Java month number for March will be 3-1=2.
6. Using java month number, we can filter dates from other months.

*Java Code:*

|  |  |
| --- | --- |
|  | package MakeSeleniumEasy; |
|  |  |
|  |  |
|  |  |
|  | import java.text.ParseException; |
|  | import java.text.SimpleDateFormat; |
|  | import java.util.Calendar; |
|  | import java.util.Date; |
|  | import java.util.List; |
|  |  |
|  | import org.openqa.selenium.By; |
|  | import org.openqa.selenium.WebDriver; |
|  | import org.openqa.selenium.WebElement; |
|  | import org.openqa.selenium.chrome.ChromeDriver; |
|  |  |
|  | public class HandlingMMTCalendar { |
|  |  |
|  | public static void selectDate(WebElement calendar, String year, String monthName, String day, WebDriver driver) throws ParseException |
|  | { |
|  | //Clicking on calendar to open calendar widget |
|  | calendar.click(); |
|  |  |
|  | // Retrieving current year value |
|  | String currentYear= driver.findElement(By.xpath("//div[@class='ui-datepicker-title']/span[@class='ui-datepicker-year']")).getText(); |
|  |  |
|  | // Click on Next arrow till we get desired year |
|  | if(!currentYear.equals(year)) |
|  | { |
|  | do{ |
|  | driver.findElement(By.xpath("(//span[text()='Next'])[1]")).click(); |
|  | }while(!driver.findElement(By.xpath("//div[@class='ui-datepicker-title']/span[@class='ui-datepicker-year']")).getText().equals(year)); |
|  |  |
|  | } |
|  |  |
|  | // Select desired month after selecting desired year |
|  |  |
|  | String currentMonth= driver.findElement(By.xpath("(//div[@class='ui-datepicker-title']/span[@class='ui-datepicker-month'])[1]")).getText(); |
|  | if(!currentMonth.equalsIgnoreCase(monthName)) |
|  | { |
|  | do{ |
|  | driver.findElement(By.xpath("(//span[text()='Next'])[1]")).click(); |
|  | }while(!driver.findElement(By.xpath("(//div[@class='ui-datepicker-title']/span[@class='ui-datepicker-month'])[1]")).getText().trim().equalsIgnoreCase(monthName)); |
|  |  |
|  | } |
|  | //get java month number for desired month |
|  |  |
|  | int javaMonthInt= HandlingMMTCalendar.getMonthJavaInt(monthName); |
|  |  |
|  | // Find dates of desired month only |
|  |  |
|  | List<WebElement> allDateOfDesiredMonth= driver.findElements(By.xpath("//div[@class='ui-datepicker-group ui-datepicker-group-first']//table/tbody[1]//td[(@class=' ' or @class=' ui-datepicker-week-end ' ) and @data-month='"+javaMonthInt+"']")); |
|  | for(WebElement d:allDateOfDesiredMonth ) |
|  | { |
|  | if(d.getText().trim().equals(day)) |
|  | { |
|  | d.click(); |
|  | break; |
|  | } |
|  | } |
|  |  |
|  |  |
|  |  |
|  | } |
|  |  |
|  | // Code to get java month number |
|  | public static int getMonthJavaInt(String monthName) throws ParseException |
|  | { |
|  |  |
|  | Date date = new SimpleDateFormat("MMMM").parse(monthName); |
|  | Calendar cal = Calendar.getInstance(); |
|  | cal.setTime(date); |
|  | return cal.get(Calendar.MONTH); |
|  | } |
|  |  |
|  | public static void main(String[] args) throws ParseException { |
|  |  |
|  | System.setProperty("webdriver.chrome.driver", "./exefiles/chromedriver.exe"); |
|  | WebDriver driver= new ChromeDriver(); |
|  | driver.manage().window().maximize(); |
|  | driver.get("https://www.makemytrip.com/"); |
|  |  |
|  | // Locating departure date calendar |
|  | WebElement departCal= driver.findElement(By.id("hp-widget\_\_depart")); |
|  | HandlingMMTCalendar.selectDate(departCal, "2017", "September", "22", driver); |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | } |
|  | } |

**[view raw](https://gist.github.com/anonymous/a0c0efd5ea21b8ddaa58ebe426c54442/raw/2439002a6737ccd5a74dc5955eb683f04d488de9/gistfile1.txt)[gistfile1.txt](https://gist.github.com/anonymous/a0c0efd5ea21b8ddaa58ebe426c54442" \l "file-gistfile1-txt)**hosted with  by **[GitHub](https://github.com/)**

*Output:*

**

*You can customize and add more validation for proper valid dates. But whatever you want to do, above logic will help you in selecting date.*

# How to Take a Screenshot in Selenium WebDriver Using Java?

* **Last Updated :** 23 Jan, 2022

Selenium [WebDriver](https://www.geeksforgeeks.org/applications-and-uses-of-selenium-webdriver/) is a collection of open-source APIs which are used to automate the testing of a web application. In order to capture a screenshot in Selenium, one has to utilize the method Takes Screenshot. This notifies WebDriver that it should **take** a **screenshot in Selenium** and store it. **Selenium**[**WebDriver**](https://www.geeksforgeeks.org/applications-and-uses-of-selenium-webdriver/) tool is used to automate web application testing to verify that it works as expected. It supports many browsers. Here we will be taking the example of chrome.

**Syntax:**

File file = ((TakesScreenshot) driver) ;

Here, we will learn how to take a screenshot in the selenium web driver and highlight the element using Java Binding.

Screenshots are required for bug analysis especially in the case of test case failure. Whenever a test case fails, we need some attachment to verify that failure. Selenium can automatically take a screenshot during execution, and we can also mark a border to highlight that element.

**Steps to take a screenshot in selenium web driver:**

**1.**Take a screenshot and store it in a file format

**2.**Copy screenshot to a location using CopyFile method

*FileUtils.copyFile(File, new File(“location where you want to save the image” +FileName+ “.jpeg”));*

**3.** Create a border around the element: Using the JavaScript executor, we can create a border around the desired element.

JavascriptExecutor js = (JavascriptExecutor) driver;

js.executeScript("arguments[0].style.border = '3px solid red'", Element);

Refer to the complete code below for the implementation part

* Java

|  |
| --- |
| // Java program how to take  // a screenshot in Selenium  // WebDriver    **import** java.io.\*;    **class** GFG {  **public** **static** **void** main(String[] args)      {          // Setting webDriver to chrome          System.setProperty("webdriver.chrome.driver",                             "driver path");          driver = **new** ChromeDriver();          driver.get("[https://www.google.co.in](https://www.google.co.in/)");            WebElement Element = driver.findElement(              By.xpath("//input[@name='q']"));            // Assignments to webDriver          MakeBorder(Element);          Thread.sleep(2000);          TakeScreenshot("GooglePage");          driver.quit();      }        // Function to Take screenshot  **public** **static** **void** TakeScreenshot(String FileName)  **throws** IOException      {          // Creating instance of File          File File = ((TakesScreenshot)driver)                          .getScreenshotAs(OutputType.FILE);            FileUtils.copyFile(File,  **new** File("image location"                                      + FileName + ".jpeg"));      }        // Function to Make border  **public** **static** **void** MakeBorder(WebElement Element)      {            JavascriptExecutor js = (JavascriptExecutor)driver;          js.executeScript(              "arguments[0].style.border = '3px solid red'",              Element);      }  } |

**Output**:

Graphical user interface, text

Description automatically generated

Attention reader! Don’t stop learning now. Get hold of all the important [**Java Foundation**](https://practice.geeksforgeeks.org/courses/Java-Foundation?vC=1) and Collections concepts with the [**Fundamentals of Java and Java Collections Course**](https://practice.geeksforgeeks.org/courses/Java-Collections?vC=1) at a student-friendly price and become industry ready. To complete your preparation from learning a language to DS Algo and many more,  please refer [**Complete Interview Preparation Course**](https://practice.geeksforgeeks.org/courses/complete-interview-preparation?utm_source=GeeksforGeeks&utm_medium=Text&utm_campaign=GFG_Article_Bottom_Text_CIP)**.**