How to take a screenshot in Selenium?

To take a screenshot in Selenium, we use an interface called **TakesScreenshot**, which enables the Selenium WebDriver to capture a screenshot and store it in different ways. It has a got a method "**getScreenshotAs()** " which captures the screenshot and store it in the specified location.

Below is a fundamental syntax of capturing a screenshot, using Selenium WebDriver, of the currently visible part of the Web page:

```
//Convert webdriver to TakeScreenshot
File screenshotFile = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);
```

In the above code, we convert the WebDriver object (driver) to **TakeScreenshot**. And call **getScreenshotAs()** method to create an image file by providing the parameter ***OutputType**.FILE.

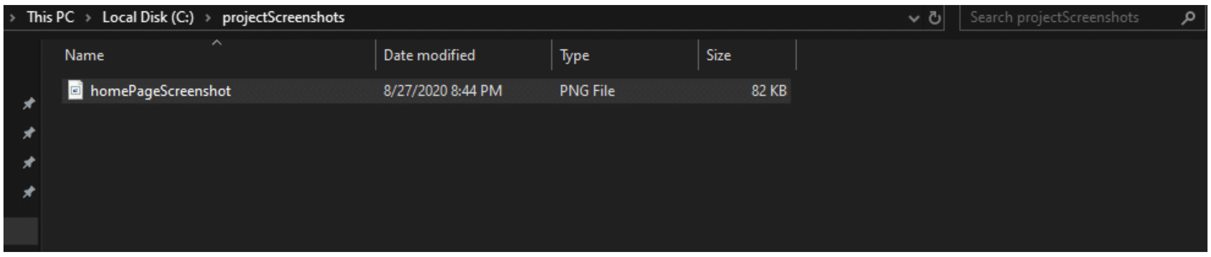
Now we can use this **File** object to copy the image at our desired location, as shown below, using the **FileUtils** Class.

```
FileUtils.copyFile(screenshotFile , new File("C:\\temp\\screenshot.png));
```

Let's try to combine the above code and write code to capture the screenshot of the homepage -

```
import org.apache.commons.io.FileUtils;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

import java.io.File;
import java.io.IOException;

public class ScreenshotDemo {
   public static void main(String[] args) {
      //set the location of chrome browser
      System.setProperty("webdriver.chrome.driver", "C:\\chromedriver.exe");

      // Initialize browser
      WebDriver driver = new ChromeDriver();

      //navigate to url
      driver.get("https://demoqa.com");

     //Take the screenshot
      File screenshot = ((TakesScreenshot) driver).getScreenshotAs(OutputType.FILE);

      //Copy the file to a location and use try catch block to handle exception
      try {
         FileUtils.copyFile(screenshot, new
File("C:\\projectScreenshots\\homePageScreenshot.png"));
      } catch (IOException e) {
         System.out.println(e.getMessage());
      }

      //closing the webdriver
      driver.close();
   }
}
```

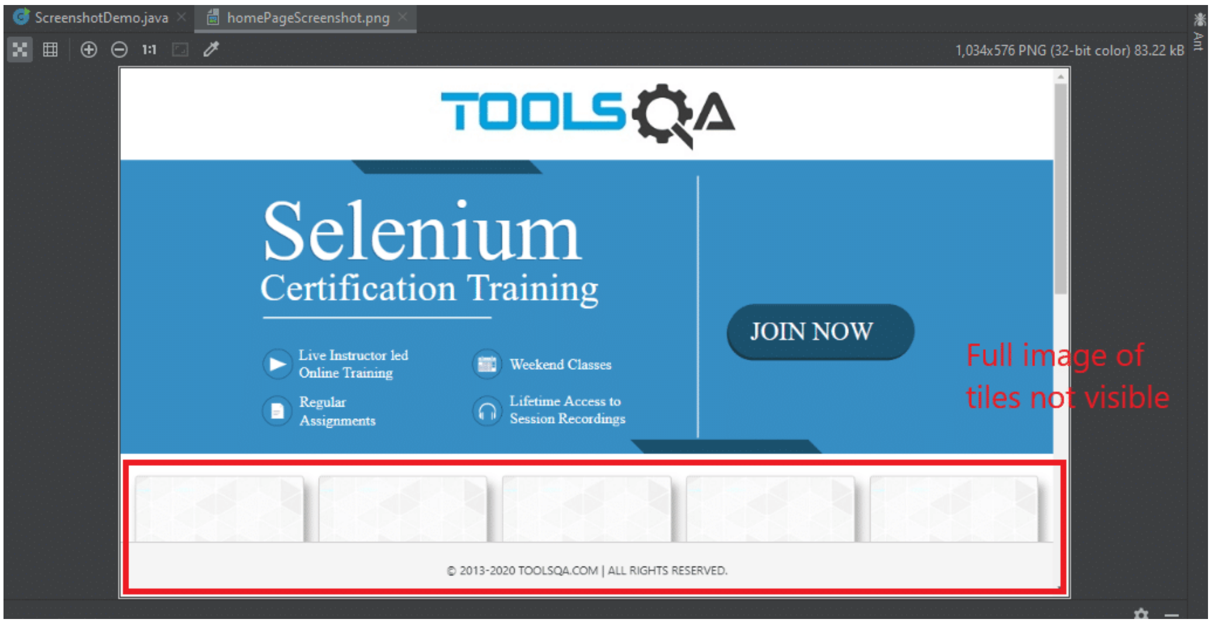
The output of the above program will generate a file in the specified location as below -



This PC > Local Disk (C:) > projectScreenshots

Name	Date modified	Type	Size
homePageScreenshot	8/27/2020 8:44 PM	PNG File	82 KB

If you open this file, you will see the image of the web page captured by Selenium WebDriver is of the viewable part. If you notice the tiles (as highlighted), they don't fully capture. This way, Selenium WebDriver only captured the screenshot of part of the web page, which was visible. We are not able to capture the screenshot of the full page using this method.


ScreenshotDemo.java
homePageScreenshot.png
1,034x576 PNG (32-bit color) 83.22 kB
TOOLSQA
Selenium
Certification Training
Live Instructor led Online Training
Weekend Classes
Regular Assignments
Lifetime Access to Session Recordings
JOIN NOW
Full image of tiles not visible
© 2013-2020 TOOLSQA.COM | ALL RIGHTS RESERVED.

Let's see how to solve this problem to capture a full-page screenshot using Selenium WebDriver.

How to take a screenshot of the full page in Selenium?
Selenium WebDriver doesn't provide the inherent capability to capture screenshot of the whole page. To capture the full page screenshot, we have to use a third-party library named **Ashot**. It provides the ability to take a screenshot of a particular WebElement as well as a full-page screenshot. You can download the jar file of the Ashot library from "**https://jar-download.com/artifacts/ru.yandex.qatools.ashot/ashot** " and then add the same as an external dependency as per the steps specified in the article "**Add External library to Eclipse project**. " Once the **Ashot** library adds to the Eclipse project, we can use the following methods to capture the screenshots of the web page:

- Capture the screen size image, which is the viewable area on the screen. It will be the same as seen in the above section to capture a screenshot of the page that is currently visible. You can do the same task using **Ashot** as well with the below code. There we create an **Ashot** object and call the **takeScreenshot()** method:

```
Screenshot screenshot = new Ashot().takeScreenshot(driver);
```

- Capture the **full page screenshot**, which is more than the currently visible part on the screen. After creating the **AShot** object, we need to call the **shootingStrategy()** method before calling the **takeScreenshot()** method to set up the policy. We can write the below code to do so:

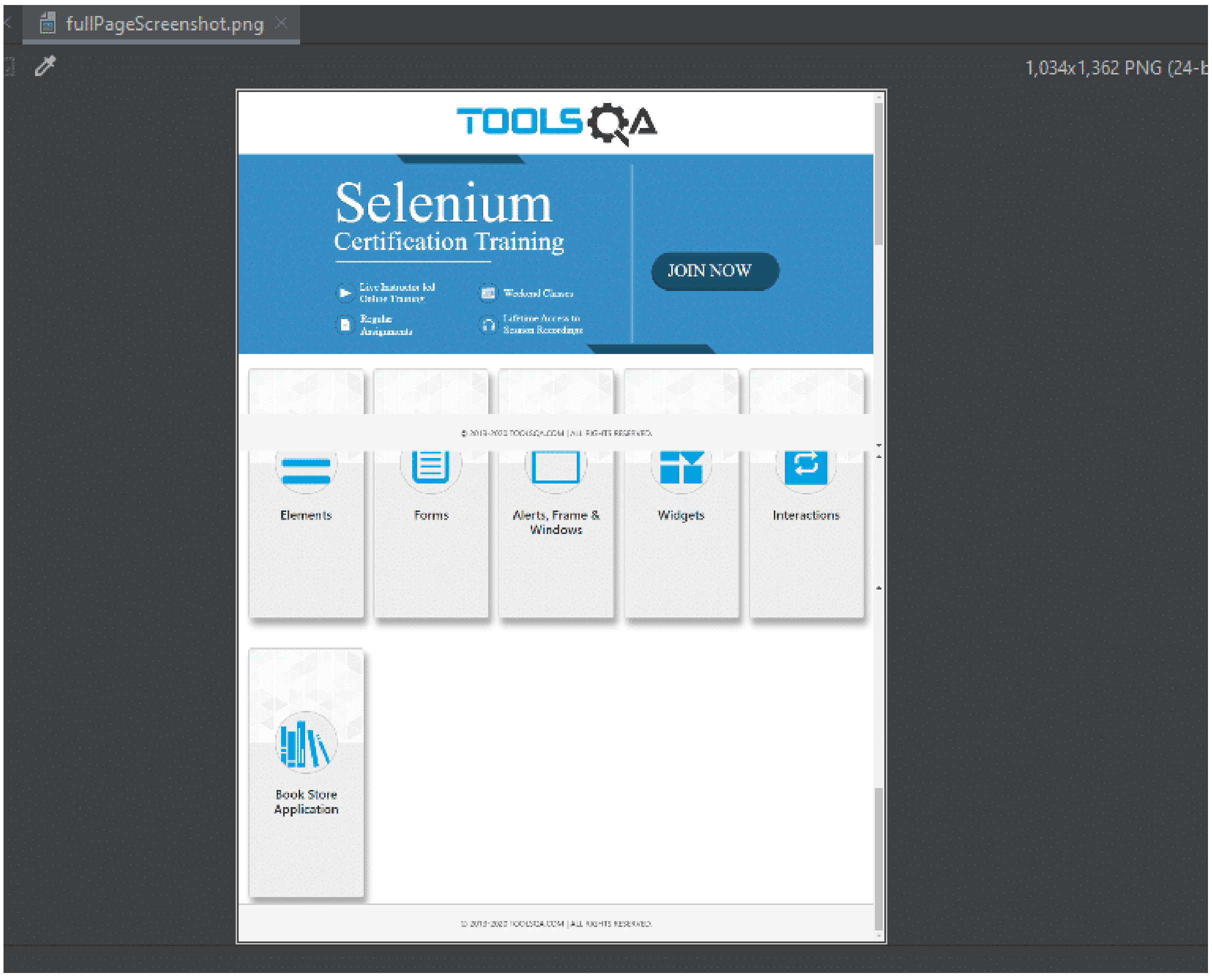
```
Screenshot s=new
AShot().shootingStrategy(ShootingStrategies.viewportPasting(1000)).takeScreenshot(driver)
;
ImageIO.write(s.getImage(),"PNG",new File("<< file path>>"));
```

- In the above code, 1000 is scrolled out time in milliseconds. In other words, it means that the program will scroll for each 1000 msec to take a screenshot.

Let's try to apply it to the **ToolsQA demo site** home page to get the entire page screenshot, including the tiles missing in the viewable area screenshot.

```
import org.apache.commons.io.FileUtils;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import ru.yandex.qatools.ashot.AShot;
import ru.yandex.qatools.ashot.Screenshot;
import ru.yandex.qatools.ashot.shooting.ShootingStrategies;

import javax.imageio.ImageIO;
import java.io.File;
import java.io.IOException;

public class ScreenshotDemo {
   public static void main(String[] args) throws IOException {
      //set the location of chrome browser
      System.setProperty("webdriver.chrome.driver", "C:\\chromedriver.exe");

      // Initialize browser
      WebDriver driver = new ChromeDriver();

      //navigate to url
      driver.get("https://demoqa.com");

      // capture screenshot and store the image
      Screenshot s=new
AShot().shootingStrategy(ShootingStrategies.viewportPasting(1000)).takeScreenshot(driver)
;
      ImageIO.write(s.getImage(),"PNG",new
File("C:\\projectScreenshots\\fullPageScreenshot.png"));

      //closing the webdriver
      driver.close();
   }
}
```

The output of the above code will generate the image in the specified path. It looks like below and contains all the elements of the page -



Another use case can be when we want to capture the screenshot of a particular web element. Selenium WebDriver provides specific ways to achieve the same also. Let's see how we can capture the screenshot of a specific element of the web using Selenium WebDriver?

How to take a screenshot of a particular element in Selenium?
Sometimes, we need a screenshot of a particular element on a page. There are two ways to capture the screenshot of a web element in Selenium-

1. Take the fullscreen image and then crop the image as per the dimensions of the web element.
2. Using the **getScreenshotAs()** method on the web element. (This is available only in selenium version 4.X)

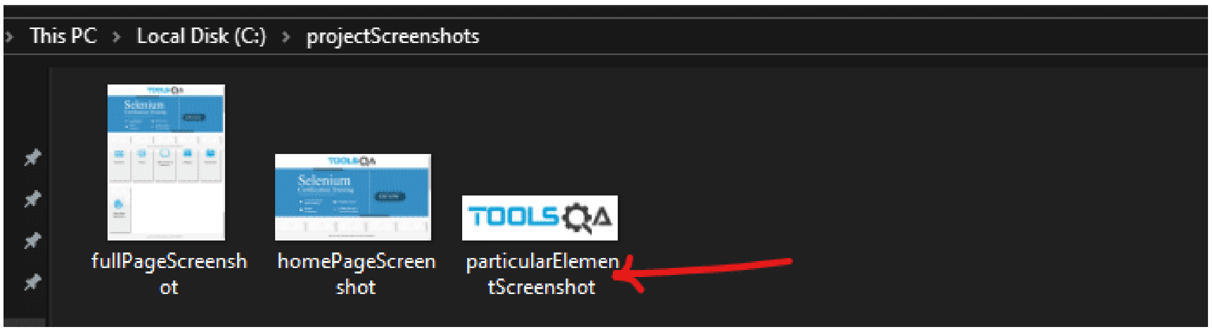
Let's understand both of these methods in the following sections:

How to capture the screenshot of a particular element by cropping the full page screenshot?
You can follow the below steps to take the screenshot of the **logo of ToolsQA** present on the **ToolsQA demo site**.

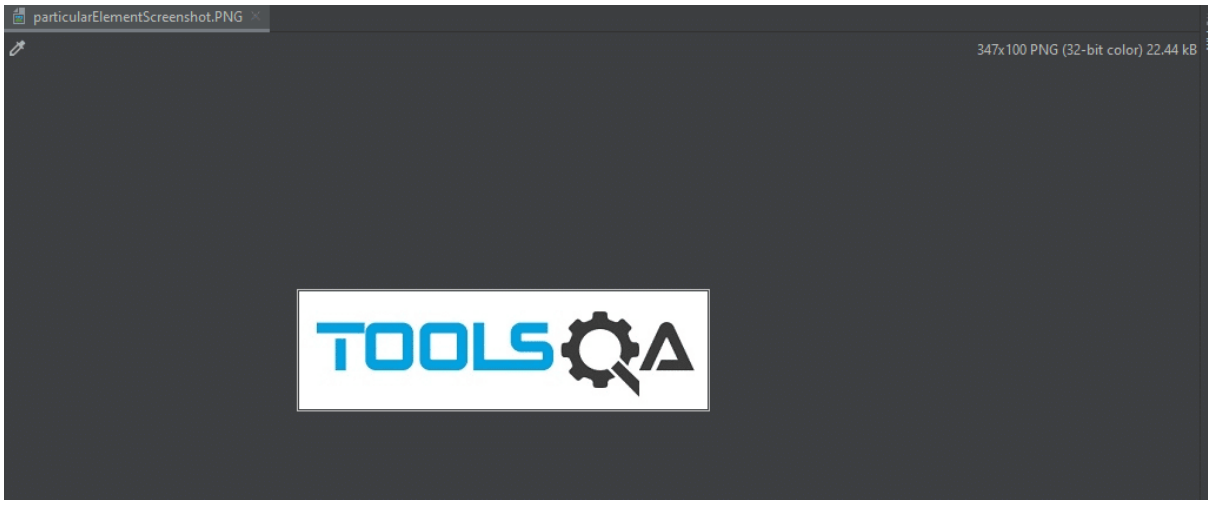
- Capture the viewport screenshot as a buffered Image.
- Get element's height and width using getSize() method.
- Get element's X Y coordinates using Point class.
- Read buffered Image.
- Crop buffered Image using element's x, y coordinate position, and height, width parameters.
- Save cropped Image at destination location physically.

The following code snippet shows how we can achieve the same programmatically:

```
import org.apache.commons.io.FileUtils;
import org.openqa.selenium.*;
import org.openqa.selenium.chrome.ChromeDriver;
import javax.imageio.ImageIO;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;

public class ScreenshotDemo {
    public static void main(String[] args) throws IOException {
        //set the location of chrome browser
        System.setProperty("webdriver.chrome.driver", "C:\\chromedriver.exe");

        // Initialize browser
        WebDriver driver = new ChromeDriver();

        //navigate to url
        driver.get("https://demoqa.com");

        // Locate the element on the web page
        WebElement logo = driver.findElement(By.xpath("//*[@id=\"app\"]/header/a/img"));

        // Get screenshot of the visible part of the web page
        File screenshot = ((TakesScreenshot) driver).getScreenshotAs(OutputType.FILE);

        // Convert the screenshot into BufferedImage
        BufferedImage fullScreen = ImageIO.read(screenshot);

        //Find location of the webelement logo on the page
        Point location = logo.getLocation();

        //Find width and height of the located element logo
        int width = logo.getSize().getWidth();
        int height = logo.getSize().getHeight();

            //cropping the full image to get only the logo screenshot
        BufferedImage logoImage = fullScreen.getSubimage(location.getX(), location.getY(),
            width, height);
        ImageIO.write(logoImage, "png", screenshot);

        //Save cropped Image at destination location physically.
        FileUtils.copyFile(screenshot, new
File("C:\\projectScreenshots\\particularElementScreenshot.PNG"));

        driver.quit();
    }
}
```
The output of the above code will be a file generated in the specified location, which looks like below with only the logo present:



On opening the screenshot file, only the logo is present, as shown below:


particularElementScreenshot.PNG
347x100 PNG (32-bit color) 22.44 kB

So this way, you can crop any of the existing screenshots. It helps you to get a portion or screenshot of a specific web element.

But, Selenium has matured with time and has provided a better and easy method in **Selenium 4.x.x** to capture screenshot of a web element. Let's see how we can get the same?

How to capture the screenshot of a particular element by using the getScreenshotAs() method?
We can capture screenshots of a particular element in **Selenium 4.0** with the **getScreenshotAs** (OutputType.File) method of the **WebElement** class. So, now the method getScreenshotAs() can be directly invoked on the WebElement, for which we want to capture the screenshot:

```
import org.apache.commons.io.FileUtils;
import org.openqa.selenium.*;
import org.openqa.selenium.chrome.ChromeDriver;
import java.io.File;
import java.io.IOException;

public class ScreenshotDemo {
   public static void main(String[] args) throws IOException {
      //set the location of chrome browser
      System.setProperty("webdriver.chrome.driver", "C:\\chromedriver.exe");

      // Initialize browser
      WebDriver driver = new ChromeDriver();

      //navigate to url
      driver.get("https://demoqa.com");

      // Locate the web element
      WebElement logo = driver.findElement(By.xpath("//*[@id=\"app\"]/header/a/img"));

      // capture screenshot with getScreenshotAs() of the WebElement class
      File f = logo.getScreenshotAs(OutputType.FILE);

      FileUtils.copyFile(f, new File("C:\\projectScreenshots\\logoScreeshot.png"));

      driver.close();
   }
}
```

How to take a screenshot in Selenium?
To take a screenshot in Selenium, we use an interface called **TakesScreenshot**, which enables the **Selenium WebDriver** to capture a screenshot and store it in different ways. It has a got a method "**getScreenshotAs()** " which captures the screenshot and store it in the specified location.

Below is a fundamental syntax of capturing a screenshot, using Selenium WebDriver, of the currently visible part of the Web page:

```
//Convert webdriver to TakeScreenshot
File screenshotFile = ((TakesScreenshot) driver).getScreenshotAs(OutputType.FILE);
```

In the above code, we convert the WebDriver object (driver) to **TakeScreenshot**. And call **getScreenshotAs()** method to create an image file by providing the parameter ***OutputType**.FILE.

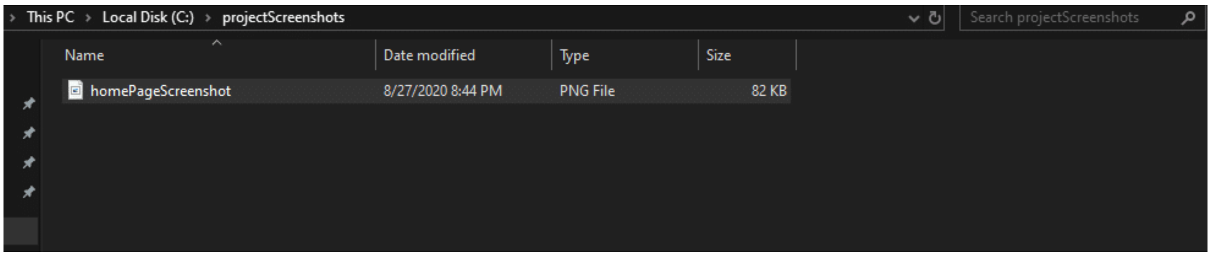
Now we can use this **File** object to copy the image at our desired location, as shown below, using the **FileUtils** Class.

```
FileUtils.copyFile(screenshotFile , new File("C:\\temp\\screenshot.png));
```

Let's try to combine the above code and write code to capture the screenshot of the **ToolsQA demo site** homepage -

```
import org.apache.commons.io.FileUtils;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

import java.io.File;
import java.io.IOException;

public class ScreenshotDemo {
    public static void main(String[] args) {
        //set the location of chrome browser
        System.setProperty("webdriver.chrome.driver", "C:\\chromedriver.exe");

        // Initialize browser
        WebDriver driver = new ChromeDriver();

        //navigate to url
        driver.get("https://demoqa.com");

       //Take the screenshot
        File screenshot = ((TakesScreenshot) driver).getScreenshotAs(OutputType.FILE);

        //Copy the file to a location and use try catch block to handle exception
        try {
            FileUtils.copyFile(screenshot, new
File("C:\\projectScreenshots\\homePageScreenshot.png"));
        } catch (IOException e) {
            System.out.println(e.getMessage());
        }
        //closing the webdriver
        driver.close();
    }
}
```

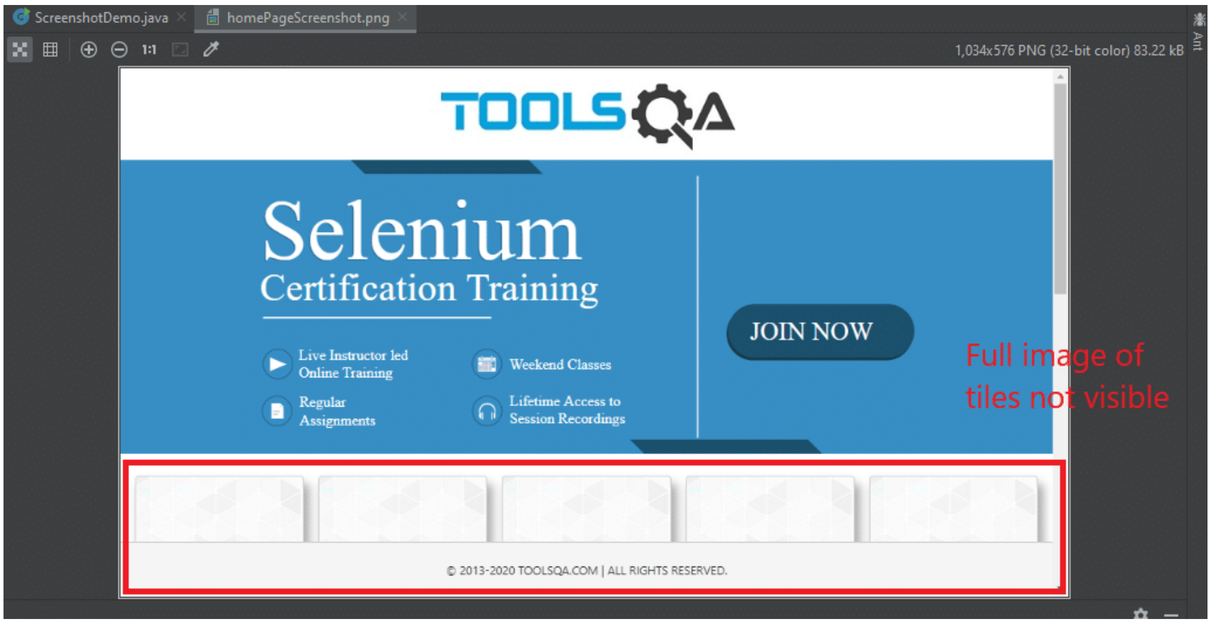
The output of the above program will generate a file in the specified location as below -



This PC › Local Disk (C:) › projectScreenshots
Search projectScreenshots

Name	Date modified	Type	Size
homePageScreenshot	8/27/2020 8:44 PM	PNG File	82 KB

If you open this file, you will see the image of the web page captured by Selenium WebDriver is of the viewable part. If you notice the tiles (as highlighted), they don't fully capture. This way, Selenium WebDriver only captured the screenshot of part of the web page, which was visible. We are not able to capture the screenshot of the full page using this method.



ScreenshotDemo.java
homePageScreenshot.png
1,034x576 PNG (32-bit color) 83.22 kB
TOOLSQA
Selenium
Certification Training
Live Instructor led Online Training
Weekend Classes
Regular Assignments
Lifetime Access to Session Recordings
JOIN NOW
Full image of tiles not visible
© 2013-2020 TOOLSQA.COM | ALL RIGHTS RESERVED.

Let's see how to solve this problem to capture a full-page screenshot using Selenium WebDriver.

How to take a screenshot of the full page in Selenium?
Selenium WebDriver doesn't provide the inherent capability to capture screenshot of the whole page. To capture the full page screenshot, we have to use a third-party library named **Ashot**. It provides the ability to take a screenshot of a particular WebElement as well as a full-page screenshot. You can download the jar file of the Ashot library from

"**https://jar-download.com/artifacts/ru.yandex.qatools.ashot/ashot** " and then add the same as an external dependency as per the steps specified in the article "**Add External library to Eclipse project**. " Once the **Ashot** library adds to the Eclipse project, we can use the following methods to capture the screenshots of the web page:

- Capture the screen size image, which is the viewable area on the screen. It will be the same as seen in the above section to capture a screenshot of the page that is currently visible. You can do the same task using **Ashot** as well with the below code. There we create an **Ashot** object and call the **takeScreenshot()** method:

```
Screenshot screenshot = new Ashot().takeScreenshot(driver);
```

- Capture the **full page screenshot**, which is more than the currently visible part on the screen. After creating the **AShot** object, we need to call the **shootingStrategy()** method before calling the **takeScreenshot()** method to set up the policy. We can write the below code to do so:

```
Screenshot s=new
AShot().shootingStrategy(ShootingStrategies.viewportPasting(1000)).takeScreenshot(driver)
;
ImageIO.write(s.getImage(),"PNG",new File("<< file path>>"));
```

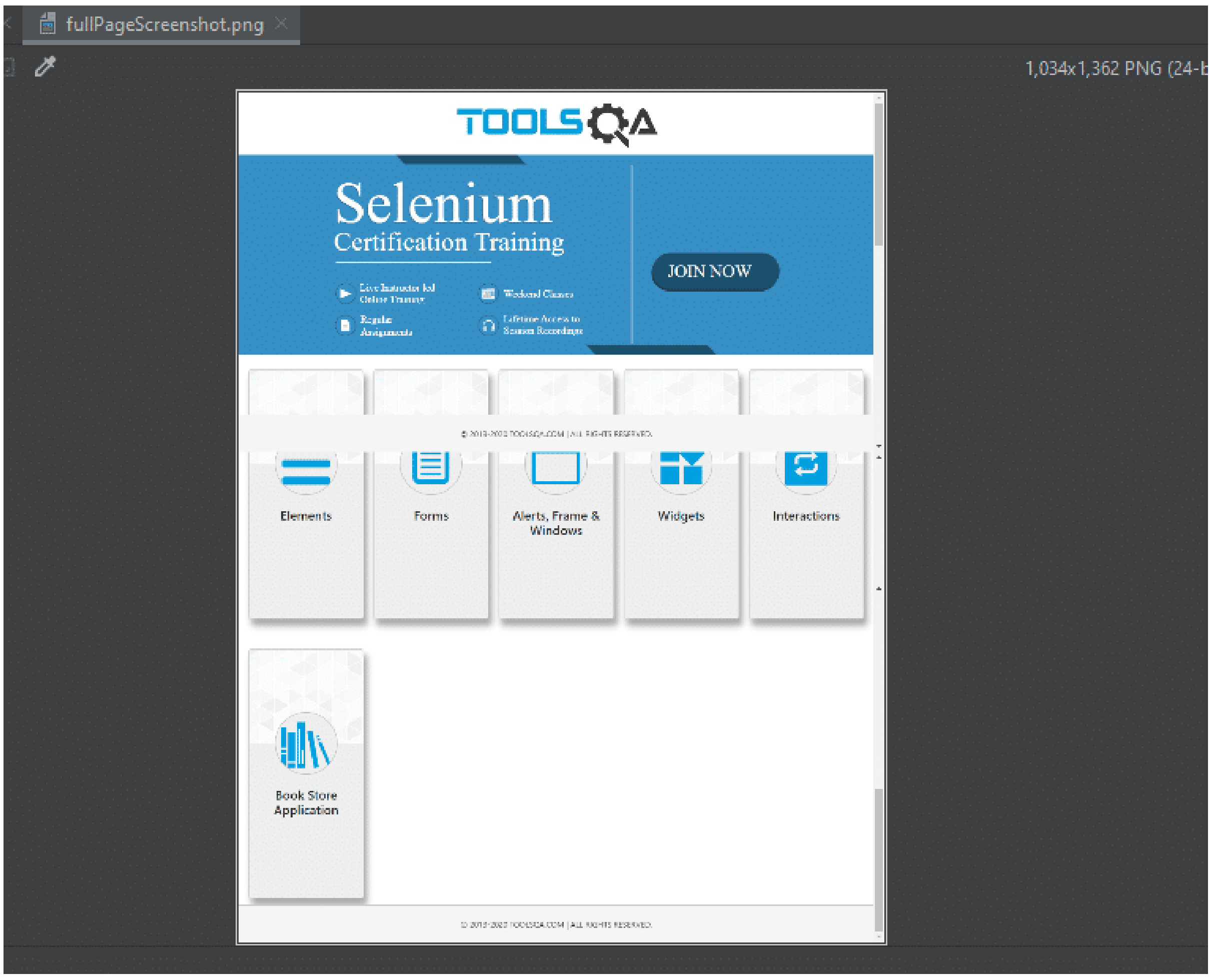
- In the above code, 1000 is scrolled out time in milliseconds. In other words, it means that the program will scroll for each 1000 msec to take a screenshot.

Let's try to apply it to the **ToolsQA demo site** home page to get the entire page screenshot, including the tiles missing in the viewable area screenshot.

```
import org.apache.commons.io.FileUtils;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import ru.yandex.qatools.ashot.AShot;
import ru.yandex.qatools.ashot.Screenshot;
import ru.yandex.qatools.ashot.shooting.ShootingStrategies;

import javax.imageio.ImageIO;
import java.io.File;
import java.io.IOException;

public class ScreenshotDemo {
   public static void main(String[] args) throws IOException {
      //set the location of chrome browser
      System.setProperty("webdriver.chrome.driver", "C:\\chromedriver.exe");

      // Initialize browser
```

```
        WebDriver driver = new ChromeDriver();

        //navigate to url
        driver.get("https://demoqa.com");

        // capture screenshot and store the image
        Screenshot s=new
AShot().shootingStrategy(ShootingStrategies.viewportPasting(1000)).takeScreenshot(driver)
;
        ImageIO.write(s.getImage(),"PNG",new
File("C:\\projectScreenshots\\fullPageScreenshot.png"));

        //closing the webdriver
        driver.close();
    }
}
```

The output of the above code will generate the image in the specified path. It looks like below and contains all the elements of the page -



Another use case can be when we want to capture the screenshot of a particular web element. Selenium WebDriver provides specific ways to achieve the same also. Let's see how we can capture the screenshot of a specific element of the web using Selenium WebDriver?

How to take a screenshot of a particular element in Selenium?
Sometimes, we need a screenshot of a particular element on a page. There are two ways to capture the screenshot of a web element in Selenium-

1. Take the fullscreen image and then crop the image as per the dimensions of the web element.
2. Using the **getScreenshotAs()** method on the web element. (This is available only in selenium version 4.X)

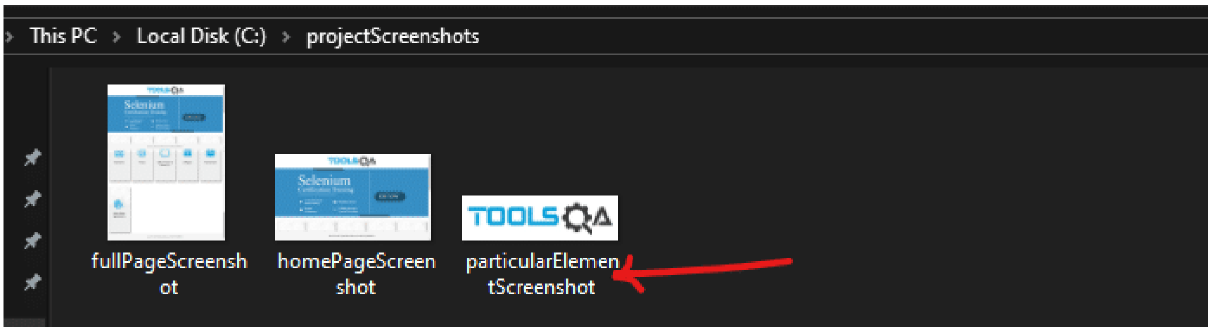
Let's understand both of these methods in the following sections:

How to capture the screenshot of a particular element by cropping the full page screenshot?
You can follow the below steps to take the screenshot of the **logo of ToolsQA** present on the **ToolsQA demo site**.

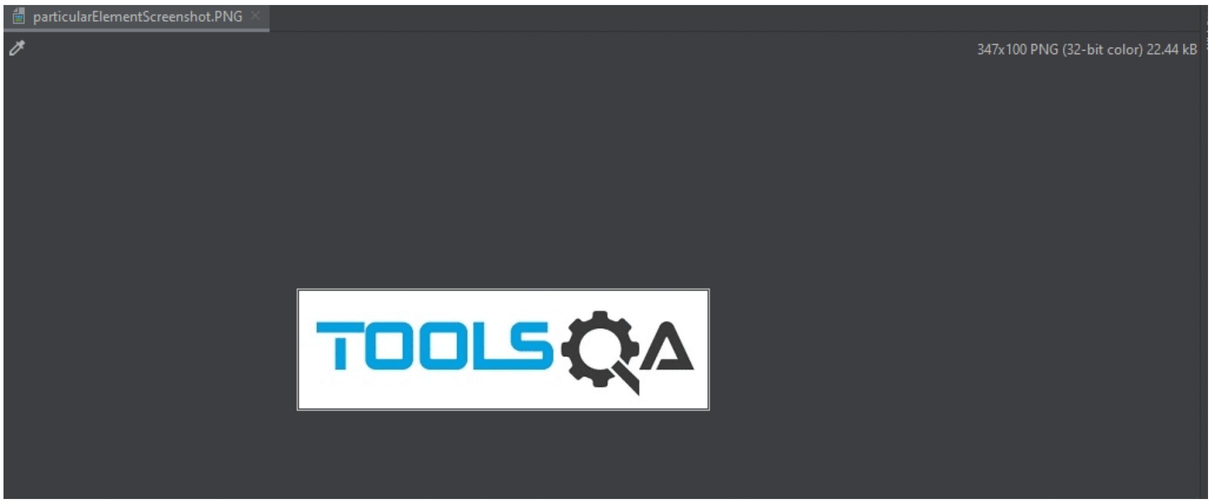
- Capture the viewport screenshot as a buffered Image.
- Get element's height and width using getSize() method.
- Get element's X Y coordinates using Point class.
- Read buffered Image.
- Crop buffered Image using element's x, y coordinate position, and height, width parameters.
- Save cropped Image at destination location physically.

The following code snippet shows how we can achieve the same programmatically:

```
import org.apache.commons.io.FileUtils;
import org.openqa.selenium.*;
import org.openqa.selenium.chrome.ChromeDriver;
import javax.imageio.ImageIO;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;

public class ScreenshotDemo {
    public static void main(String[] args) throws IOException {
        //set the location of chrome browser
        System.setProperty("webdriver.chrome.driver", "C:\\chromedriver.exe");

        // Initialize browser
        WebDriver driver = new ChromeDriver();
        //navigate to url
        driver.get("https://demoqa.com");

        // Locate the element on the web page
        WebElement logo = driver.findElement(By.xpath("//*[@id=\"app\"]/header/a/img"));

        // Get screenshot of the visible part of the web page
        File screenshot = ((TakesScreenshot) driver).getScreenshotAs(OutputType.FILE);

        // Convert the screenshot into BufferedImage
        BufferedImage fullScreen = ImageIO.read(screenshot);

        //Find location of the webelement logo on the page
        Point location = logo.getLocation();

        //Find width and height of the located element logo
        int width = logo.getSize().getWidth();
        int height = logo.getSize().getHeight();

            //cropping the full image to get only the logo screenshot
        BufferedImage logoImage = fullScreen.getSubimage(location.getX(), location.getY(),
            width, height);
        ImageIO.write(logoImage, "png", screenshot);

        //Save cropped Image at destination location physically.
        FileUtils.copyFile(screenshot, new
File("C:\\projectScreenshots\\particularElementScreenshot.PNG"));

        driver.quit();
    }
}
```
The output of the above code will be a file generated in the specified location, which looks like below with only the logo present:


This PC > Local Disk (C:) > projectScreenshots
fullPageScreenshot
homePageScreenshot
particularElementScreenshot

On opening the screenshot file, only the logo is present, as shown below:


particularElementScreenshot.PNG
347x100 PNG (32-bit color) 22.44 kB
TOOLSQA

So this way, you can crop any of the existing screenshots. It helps you to get a portion or screenshot of a specific web element.

But, Selenium has matured with time and has provided a better and easy method in **Selenium 4.x.x** to capture screenshot of a web element. Let's see how we can get the same?

How to capture the screenshot of a particular element by using the getScreenshotAs() method?
We can capture screenshots of a particular element in **Selenium 4.0** with the **getScreenshotAs** (OutputType.File) method of the **WebElement** class. So, now the method getScreenshotAs() can be directly invoked on the WebElement, for which we want to capture the screenshot:

```
import org.apache.commons.io.FileUtils;
import org.openqa.selenium.*;
import org.openqa.selenium.chrome.ChromeDriver;
import java.io.File;
import java.io.IOException;

public class ScreenshotDemo {
   public static void main(String[] args) throws IOException {
      //set the location of chrome browser
      System.setProperty("webdriver.chrome.driver", "C:\\chromedriver.exe");

      // Initialize browser
      WebDriver driver = new ChromeDriver();

      //navigate to url
      driver.get("https://demoqa.com");

      // Locate the web element
      WebElement logo = driver.findElement(By.xpath("//*[@id=\"app\"]/header/a/img"));
```

```
        // capture screenshot with getScreenshotAs() of the WebElement class
        File f = logo.getScreenshotAs(OutputType.FILE);

        FileUtils.copyFile(f, new File("C:\\projectScreenshots\\logoScreeshot.png"));

        driver.close();
    }
}
```