**What are test reports in automation?**

Creating a report for a task makes the task 100% complete. The report not only provides a summary but also helps the person to visualize and summarize the results of the task. A report plays a vital role in [test automation](https://www.lambdatest.com/automation-testing) as well. A test automation report helps the testers and the other stakeholders understand the stability of the tests executed before the product goes live. This helps the stakeholders to understand the stability of the application and also provides confidence for going live.

An automation test report should be designed in such a way that it depicts the results of the tests to the end-user. A good test report should hold various test statistics like the total number of automation test cases, the total number of test cases passed, the total number of failed test cases, the percentage of passed and failed cases, classification of the tests based on certain criteria, etc.

In recent years, various reporting tools like [TestNG Reports](https://www.lambdatest.com/blog/how-to-generate-testng-reports-in-jenkins/), JUnit Reports, Allure Reports, etc., are available for generating the test automation report. Out of those, Extent Reports is one of the [best reporting tools for Selenium](https://www.lambdatest.com/blog/best-reporting-tools-for-selenium/) that is being widely used in various organizations. It has gained immense popularity because of its unique features like report customization, integration with different test frameworks, amazing data visualization, etc. In this blog on Extent Reports in Selenium, we shall be looking at the most popularly used test report – the Extent Reports.

**Introduction to Extent Reports**

Extent Report is an open-source library used for generating test reports in [automation testing](https://www.lambdatest.com/automation-testing). It has been more widely used for report generation than the inbuilt reports in various test frameworks because of its enhanced features and customization. It is a simple yet powerful reporting library that can be integrated with the automation framework for generating the automation test report.

Extent Reports can also be integrated with other [popular test automation frameworks](https://www.lambdatest.com/blog/best-test-automation-frameworks-2021/) like JUnit and TestNG. It provides a detailed outlook on the automated test cases in a graphical way. It is very easy to integrate Extent Reports with the automation framework.

Watch this video to know more about test result analysis and reporting in Selenium 4. Also, learn about different selenium reporting tools based on ease of setup, pricing, supported report formats, and more.

Advantages of Extent Reports

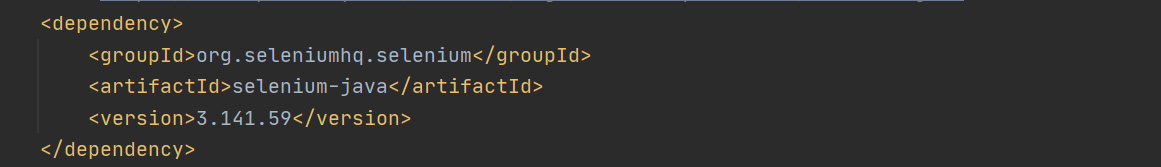
Let us pen down the advantages of using Extent Reports in Selenium for automation testing.

* It is an open-source library.
* It provides a pictorial representation of the test results.
* It can be customized as required.
* It allows the users to attach screenshots and logs in the test report for a detailed summary of the tests.
* It can be easily integrated with other frameworks like JUnit, TestNG, NUnit, etc.
* It can be easily configured with Jenkins, Bamboo, etc.

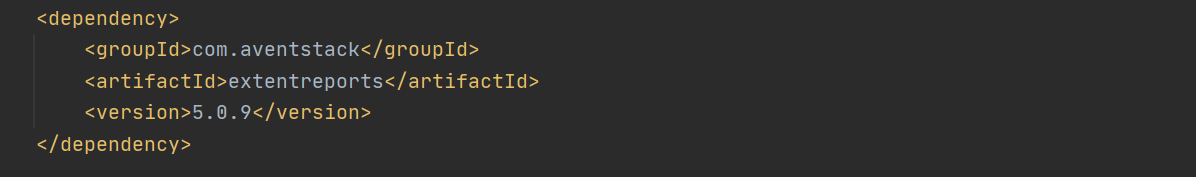
Prerequisites to generate Extent Reports

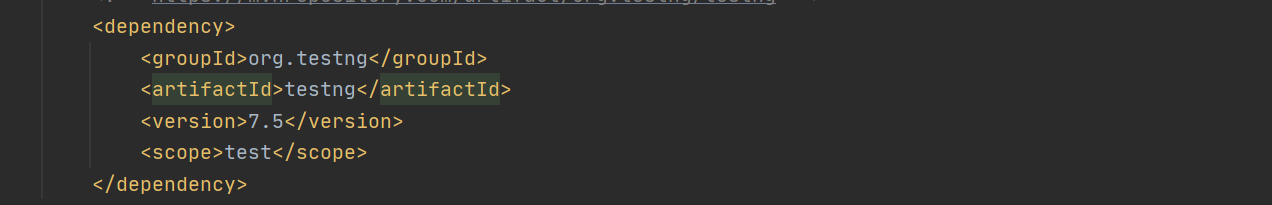
In this blog on how to generate Extent Reports in Selenium, we shall set up and generate our first Extent Report in Selenium for our automation tests implemented with Selenium Java. For more information about Selenium Java, please read our [Selenium Java Tutorial](https://www.lambdatest.com/learning-hub/selenium-java). To set up and generate Extent Reports in Selenium upon test execution, let us see the prerequisites to be done.

1. **Java:** Download and install Java in the system.
2. **Selenium:** For framing our automation test cases, we need to install Selenium by adding a maven dependency in the pom.xml file.



1. **Extent Reports:** To integrate our automation framework, we need the Extent Report dependency either as a maven dependency or a jar file. In the case of a maven project, add maven dependency in the pom.xml file.

  
You can also download the jar file and add the jar to the build path. You can refer to the [mvn repository](https://mvnrepository.com/artifact/com.aventstack/extentreports" \t "_blank) for adding the Extent Report dependency.

1. **TestNG:** To design and execute the tests, we shall use TestNG by adding the TestNG dependency.  
   

Now we have set up for automating a web application and generating the Extent Reports in Selenium.

**How to generate and customize the Extent Reports?**

There are three classes that are used for generating and customizing the Extent Reports in Selenium. They are:

1. ExtentHtmlReporter
2. ExtentReports
3. ExtentTest

The ExtentHtmlReporter is used for creating an HTML file, and it accepts a file path as a parameter. The file path represents the path in which our extent report would be generated.

|  |  |
| --- | --- |
|  | ExtentHtmlReporter htmlReporter =  new    ExtentHtmlReporter(System.getProperty("user.dir")+"/Reports/extentReport.html"); |

In the above line, we have used ExtentHtmlReporter and specified the file path where the Extent Report in Selenium has to be generated.  
In Java, the *getProperty(String key)* method returns the system property specified by the key, which is passed as the argument. Here we have specified “user.dir” as the key, and System.getProperty(“user.dir”) returns our current working directory. So instead of specifying the full path, we can use this to fetch the value of our directory and add the path where we want our Extent Report in Selenium to be saved.  
ExtentHtmlReporter is also used to customize the extent reports. It allows many configurations to be made through the config() method. Some of the configurations that can be made are described below.

1. To set the title of the extent report, we can use setDocumentTitle(“YOUR\_TITLE”).

|  |  |
| --- | --- |
|  | htmlReporter.config().setDocumentTitle("Automation Report"); |

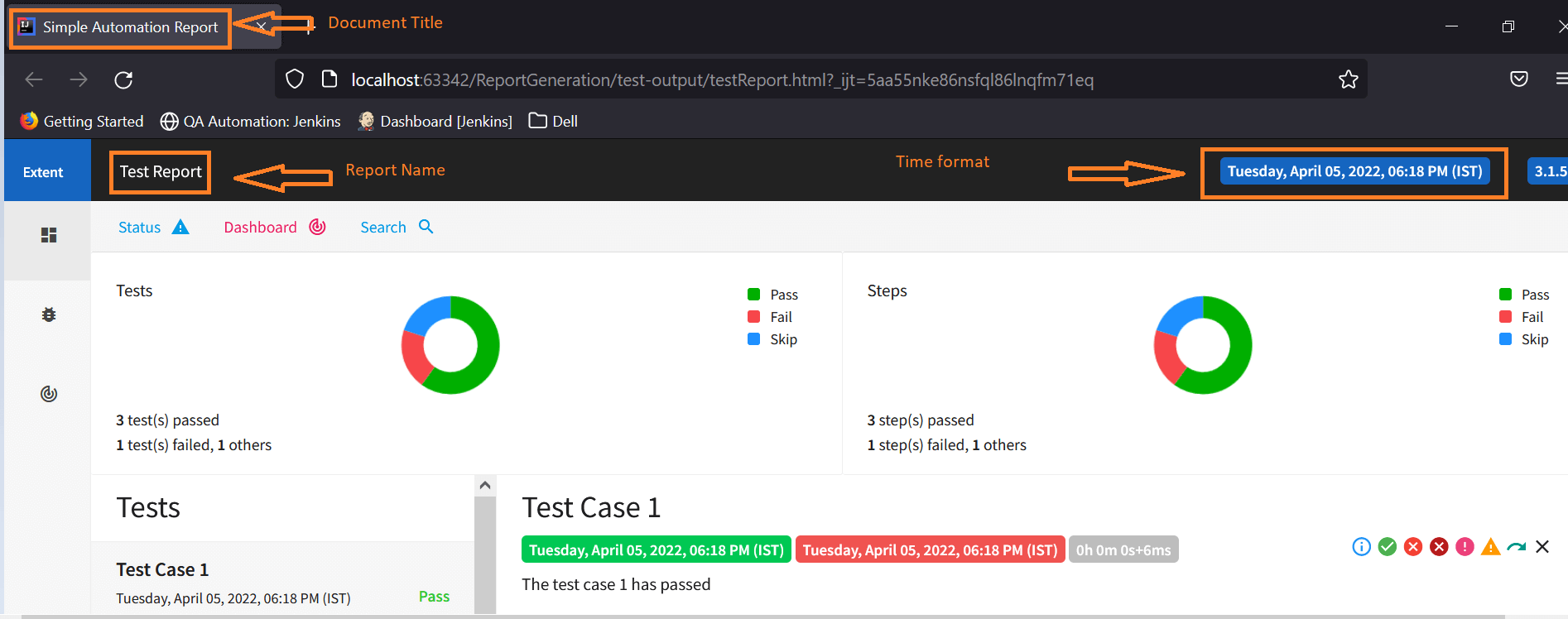
1. To set the report name, we can use setReportName(“YOUR\_REPORT\_NAME”).

|  |  |
| --- | --- |
|  | htmlReporter.config().setReportName("report"); |

1. We can also specify the time format in our extent report by using the setTimeStampFormat() method.

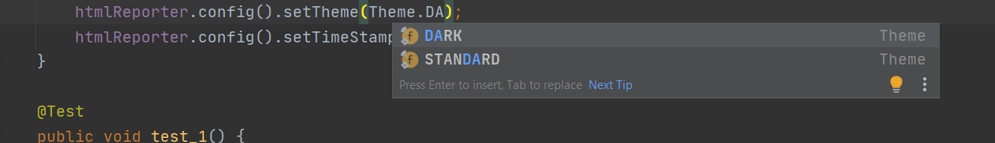
|  |  |
| --- | --- |
|  | htmlReporter.config().setTimeStampFormat("EEEE, MMMM dd, yyyy, hh:mm a '('zzz')'"); |

Now, once the Extent Report in Selenium gets generated, we can visualize all the above changes in the report.



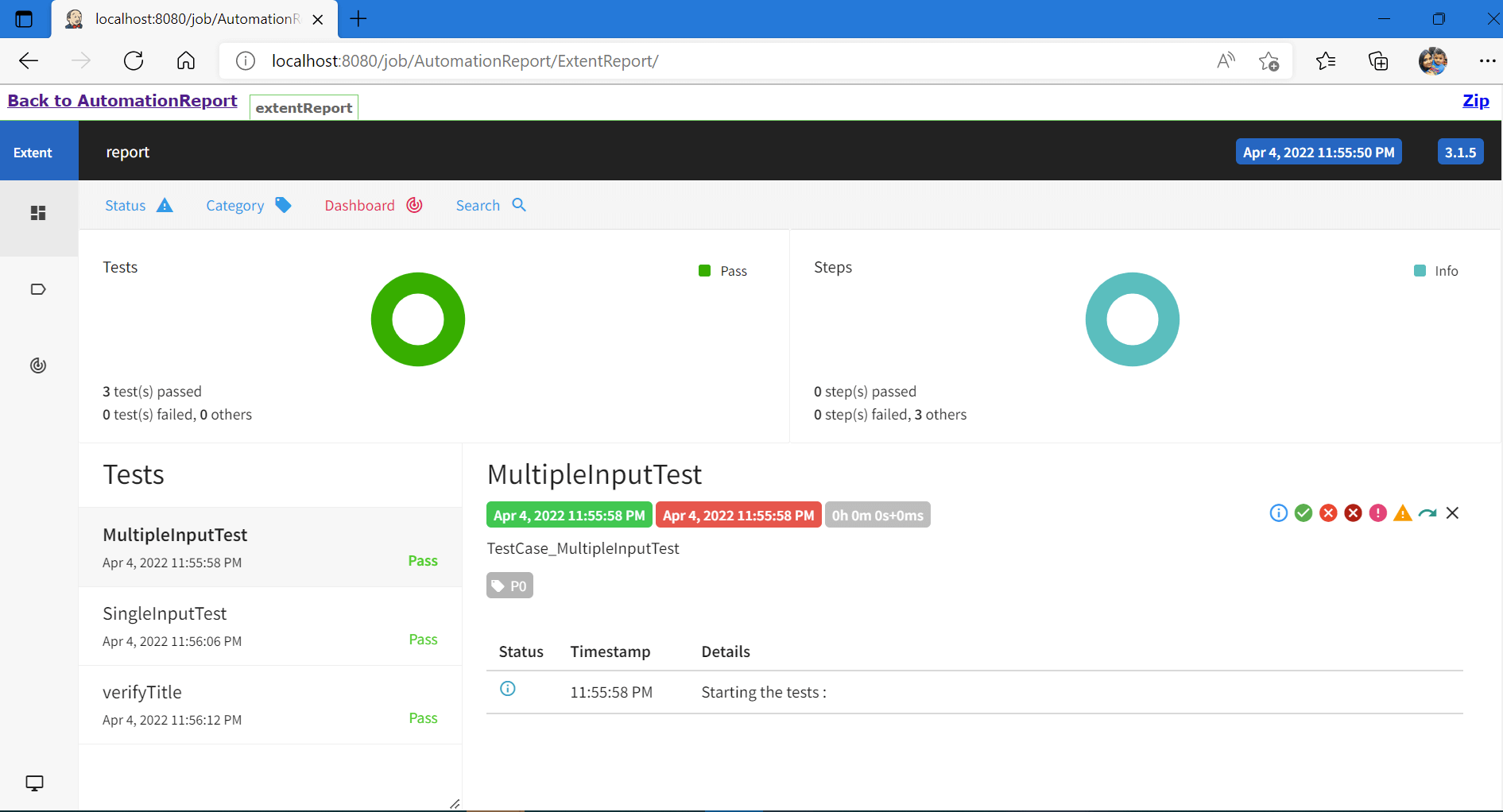
In addition to these changes, if you are a great fan of dark themes, then Extent Reports in Selenium can also be customized with dark themes using the setTheme() method.

We have two themes – STANDARD and DARK for customizing the look and feel of our extent reports.



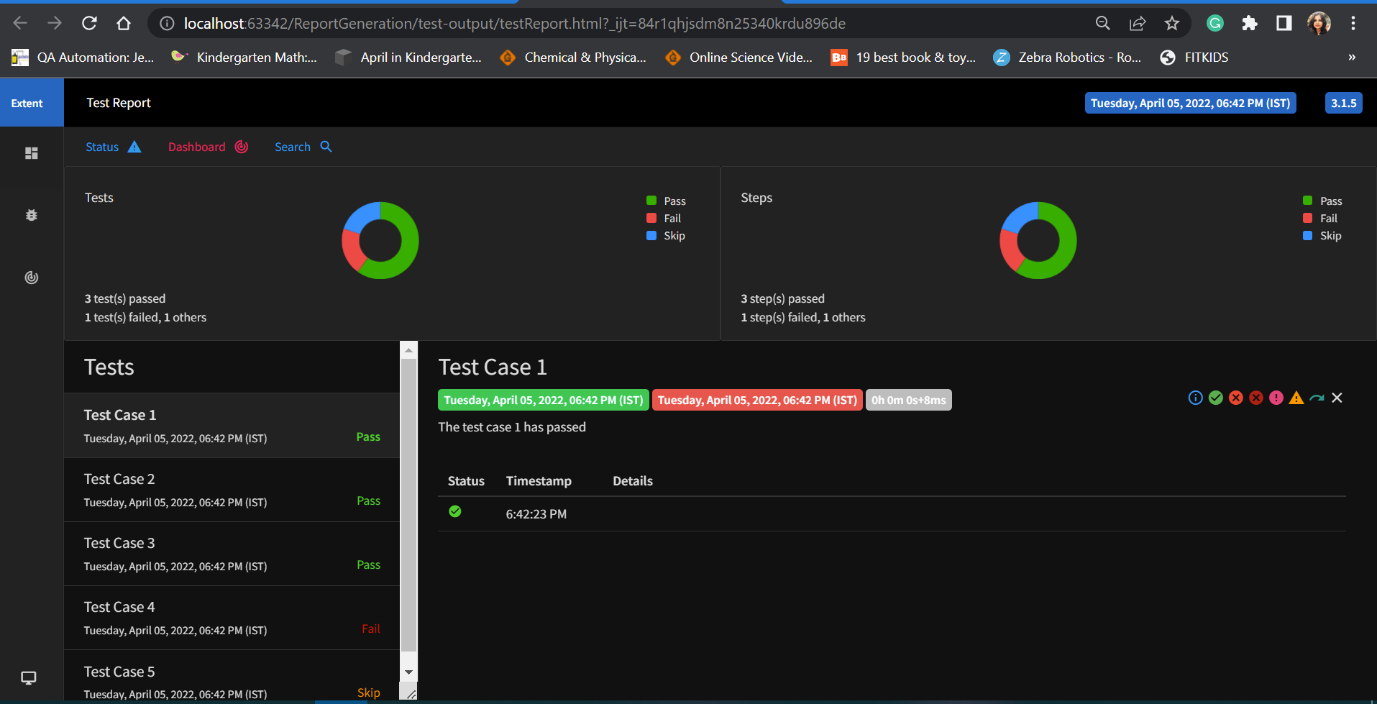
* For Light Theme, we can set the Theme as STANDARD.

|  |  |
| --- | --- |
|  | htmlReporter.config().setTheme(Theme.STANDARD); |

Below is the screenshot of the extent report with STANDARD THEME  


* For the Dark theme, we can set the theme as DARK.

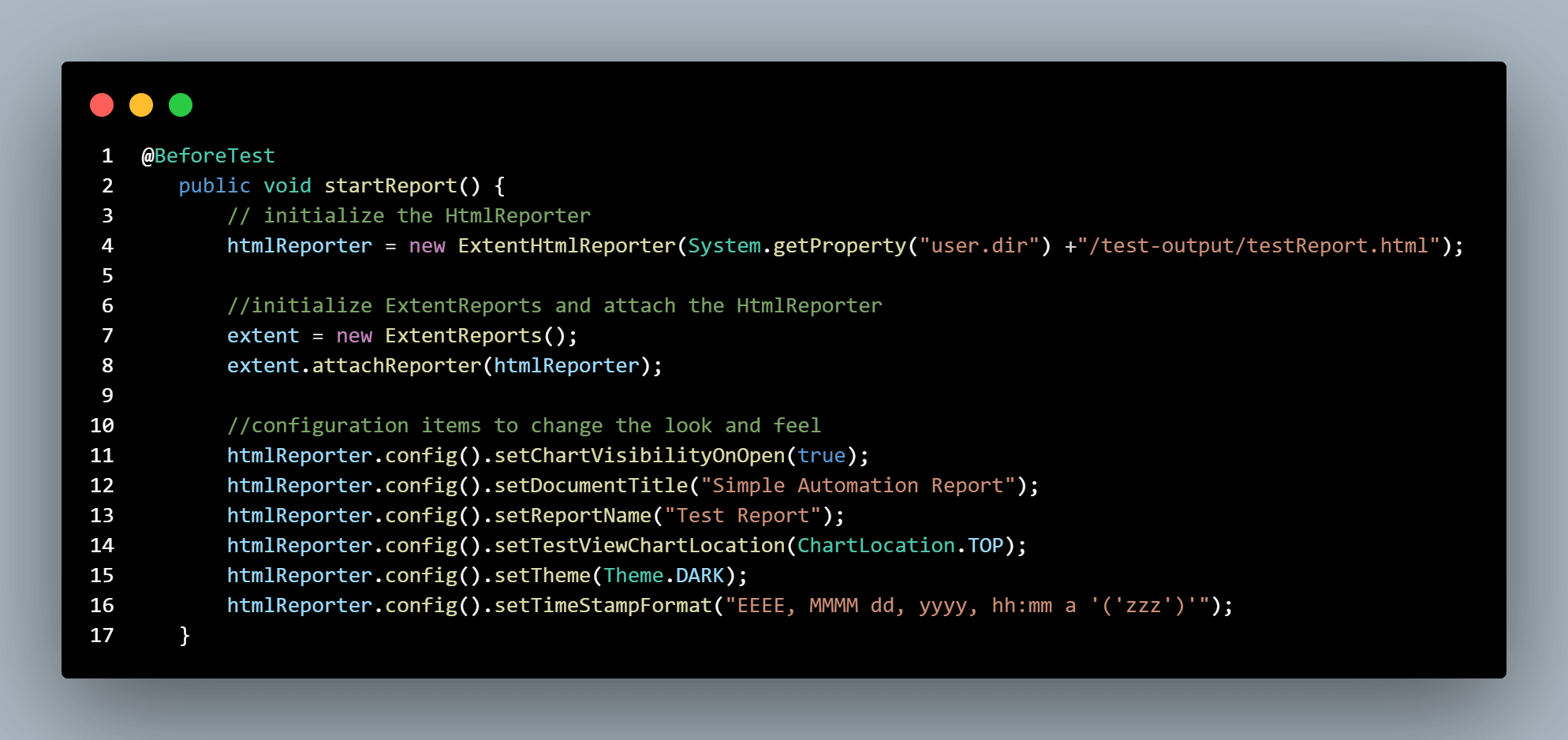
|  |  |
| --- | --- |
|  | htmlReporter.config().setTheme(Theme.DARK); |



In a nutshell,

* The ExtentHtmlReporter class is used for creating the HTML reports.
* The ExtentReports class is used for creating the tests.
* The ExtentTest class is used for generating the logs in the Extent Report.

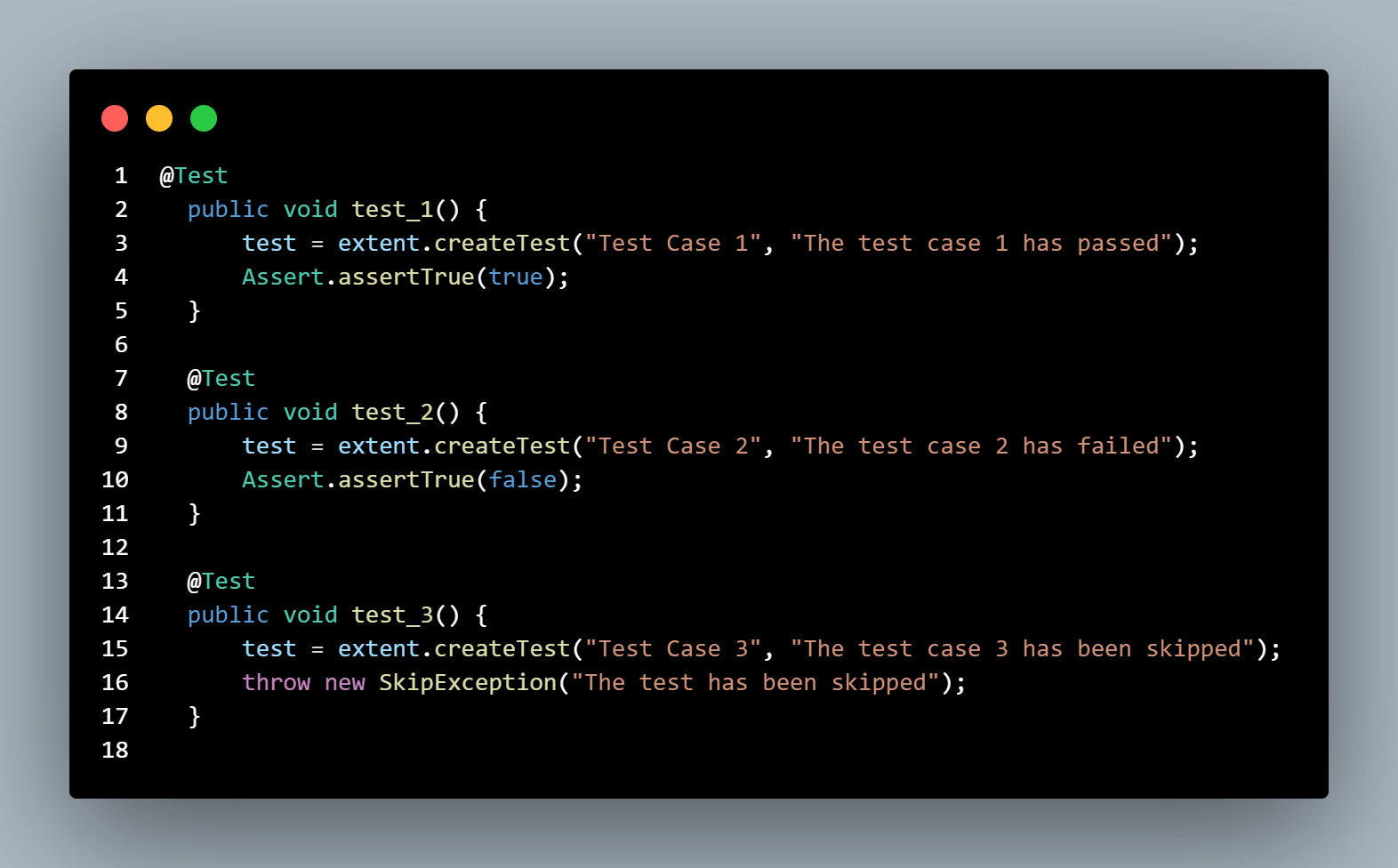
So now, we have gone through various configurations that can be made to customize the look of the Extent Reports in Selenium. We can initialize the ExtentHtmlReporter and add the required configurations in the @BeforeTest [TestNG annotation.](https://www.lambdatest.com/blog/complete-guide-on-testng-annotations-for-selenium-webdriver/)



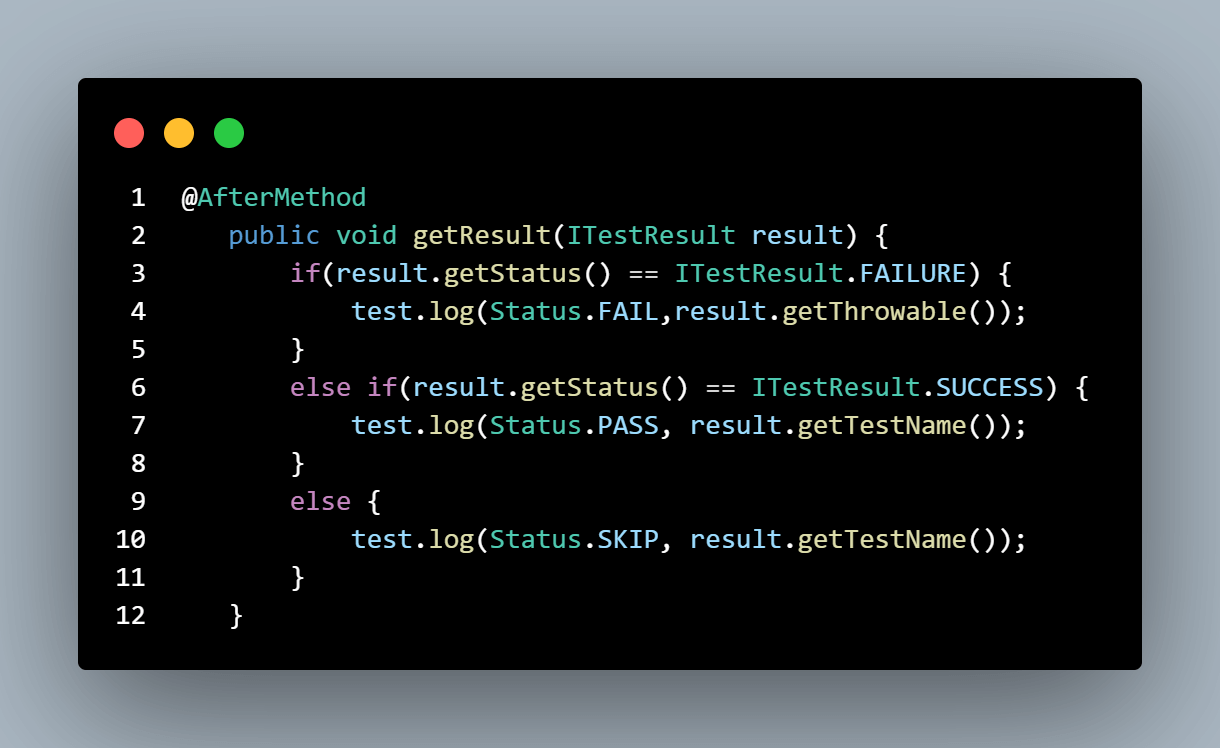
Now let us see how to write simple test cases and generate our first Extent Report in Selenium. We shall create test cases that end in three different statuses like Pass, Fail and Skip.

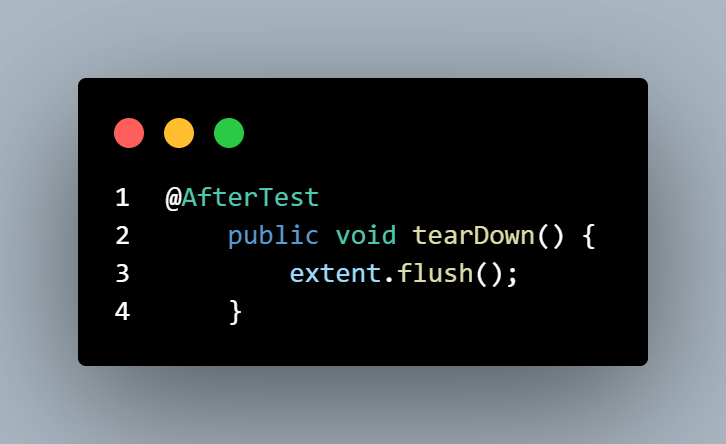
To create a test, we can use createTest(test\_name,test\_description) using ExtentReports class.

1. We have created a simple test test\_1 which ends up in PASS by asserting the condition as true.
2. We have created test\_2, which would fail as we have passed False in the assetTrue condition.
3. We have created test\_3, which would be skipped.



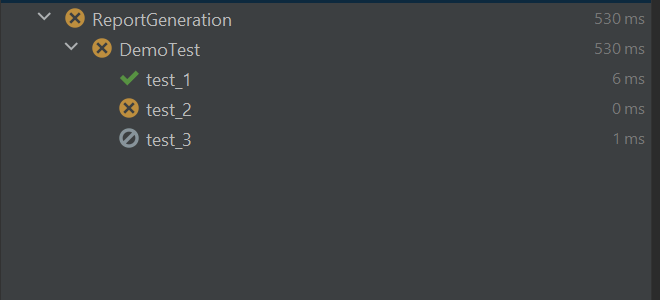
To get the status of our tests and publish it in our report, we can use the log() method from ExtentTest class.

  
In the end, we can use the flush()method, which is used for removing any previous data and creating a new Extent Report in Selenium.



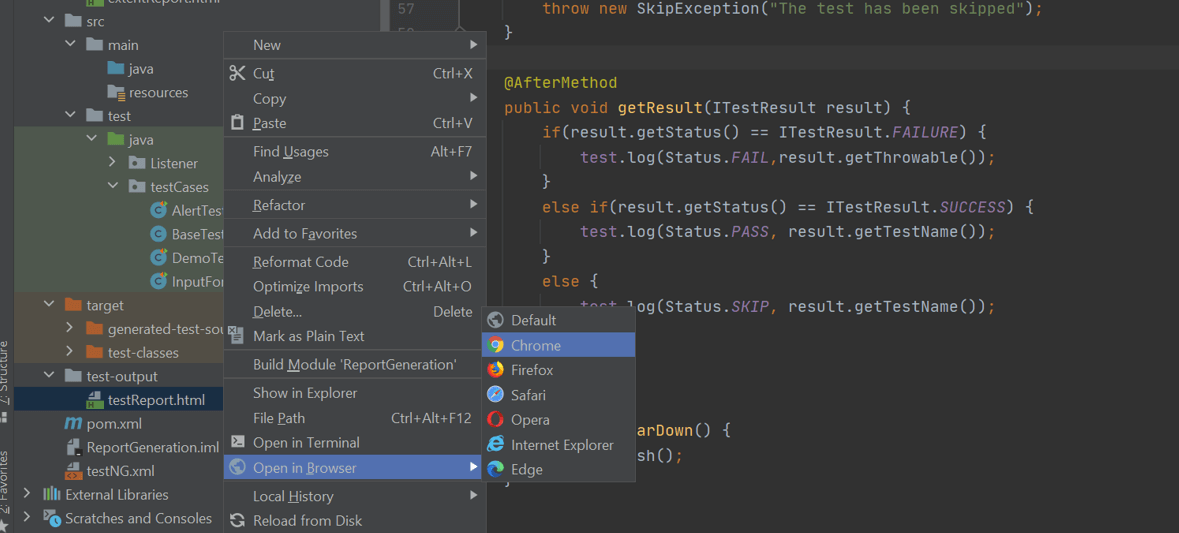
Code snippet for creating simple tests and generating our first extent report:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82 | package testCases;    import com.aventstack.extentreports.ExtentReports;  import com.aventstack.extentreports.ExtentTest;  import com.aventstack.extentreports.Status;  import com.aventstack.extentreports.reporter.ExtentHtmlReporter;  import com.aventstack.extentreports.reporter.configuration.ChartLocation;  import com.aventstack.extentreports.reporter.configuration.Theme;  import org.testng.Assert;  import org.testng.ITestResult;  import org.testng.SkipException;  import org.testng.annotations.AfterMethod;  import org.testng.annotations.AfterTest;  import org.testng.annotations.BeforeTest;  import org.testng.annotations.Test;    public class DemoTest {        ExtentHtmlReporter htmlReporter;        ExtentReports extent;      //helps to generate the logs in the test report.      ExtentTest test;        @BeforeTest      public void startReport() {          // initialize the HtmlReporter          htmlReporter = new ExtentHtmlReporter(System.getProperty("user.dir") +"/test-output/testReport.html");            //initialize ExtentReports and attach the HtmlReporter          extent = new ExtentReports();          extent.attachReporter(htmlReporter);              //configuration items to change the look and feel          //add content, manage tests etc          htmlReporter.config().setChartVisibilityOnOpen(true);          htmlReporter.config().setDocumentTitle("Simple Automation Report");          htmlReporter.config().setReportName("Test Report");          htmlReporter.config().setTestViewChartLocation(ChartLocation.TOP);          htmlReporter.config().setTheme(Theme.STANDARD);          htmlReporter.config().setTimeStampFormat("EEEE, MMMM dd, yyyy, hh:mm a '('zzz')'");      }        @Test      public void test\_1() {          test = extent.createTest("Test Case 1", "The test case 1 has passed");          Assert.assertTrue(true);      }          @Test      public void test\_2() {          test = extent.createTest("Test Case 2", "The test case 2 has failed");          Assert.assertTrue(false);      }        @Test      public void test\_3() {          test = extent.createTest("Test Case 3", "The test case 3 has been skipped");          throw new SkipException("The test has been skipped");      }        @AfterMethod      public void getResult(ITestResult result) {          if(result.getStatus() == ITestResult.FAILURE) {              test.log(Status.FAIL,result.getThrowable());          }          else if(result.getStatus() == ITestResult.SUCCESS) {              test.log(Status.PASS, result.getTestName());          }          else {              test.log(Status.SKIP, result.getTestName());          }      }        @AfterTest      public void tearDown() {          //to write or update test information to reporter          extent.flush();      }  } |

**Test Execution:**  
Upon running the tests, our output console would look similar to the screenshot below.  


Now the Extent Report in Selenium would be generated and saved in the path provided initially.

Right-click on the Extent Report 🡪 Open in Browser 🡪 Chrome (Browser of your choice).



We have successfully generated our first Extent Report in Selenium..!!!

Here you can see the tests that have been marked with different statuses.

