

Slack Integration document with Selenium Java framework

(without CI tool)

Table of Contents

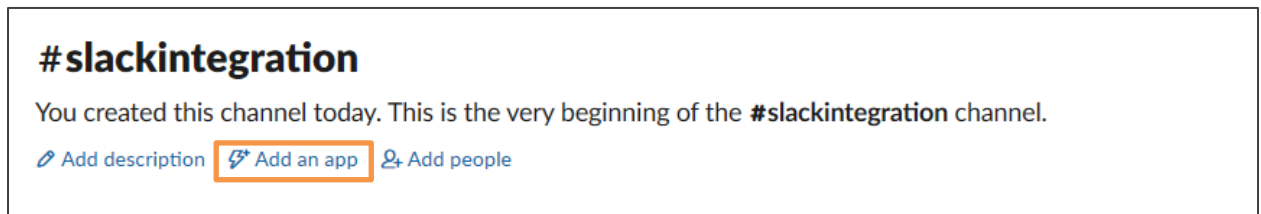
Slack Integration document with Selenium Java framework	1
1. Overview	3
2. Steps To integrate Slack with Selenium Java Framework	3
3. Code Implementation For Slack Integration	7

1. Overview

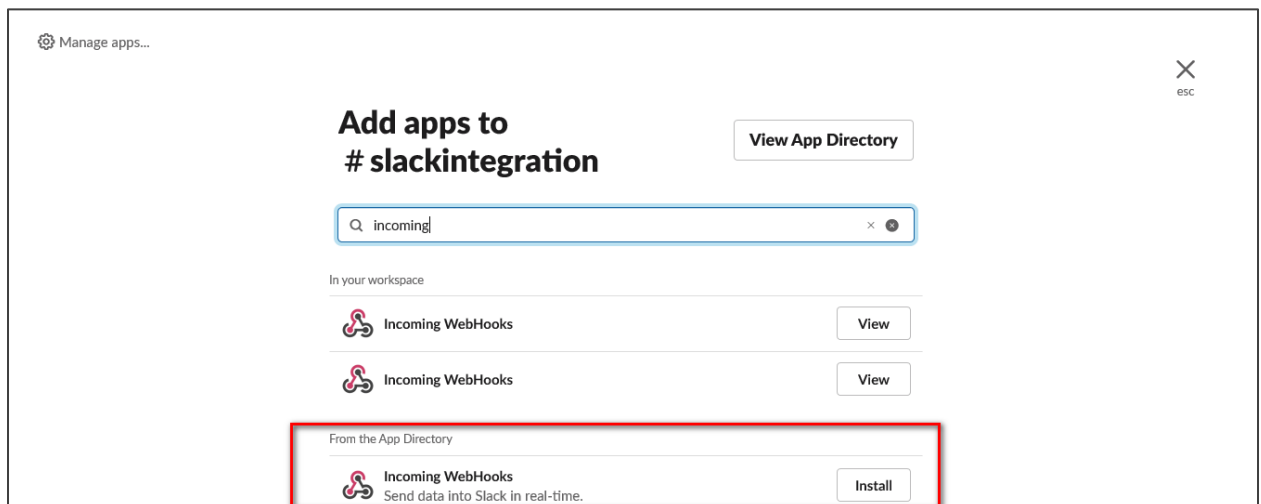
This document provides explanation about how to integrate the selenium Java Framework with Slack notifications to be triggered for automation runs.

2. Steps To integrate Slack with Selenium Java Framework

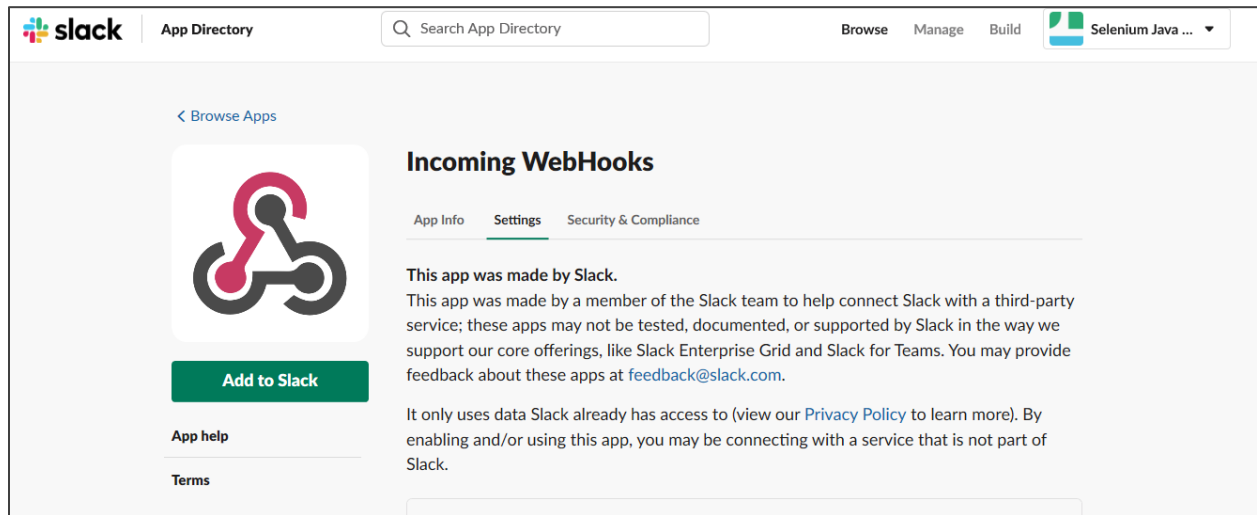
1. Goto any existing Slack channel (here, “#SlackIntegration” channel), and click on “Add an App”:



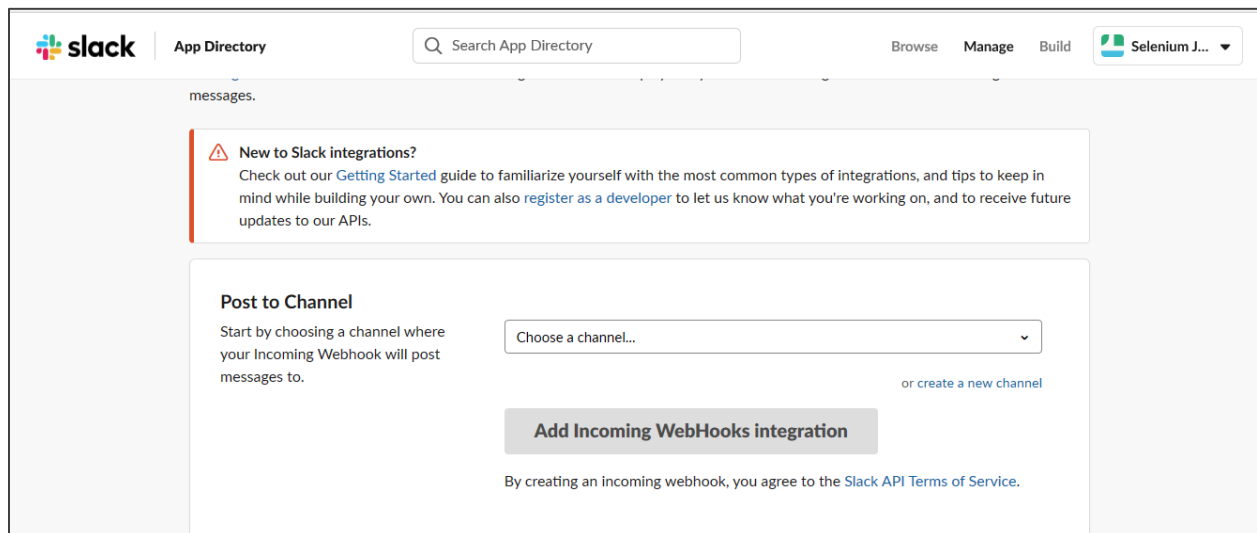
2. Search for “Incoming Webhooks” app and click on “install”



3. Click on “Add to Slack”:



4. Select the Slack channel to post Notifications and click on “Add Incoming WebHooks integration”:

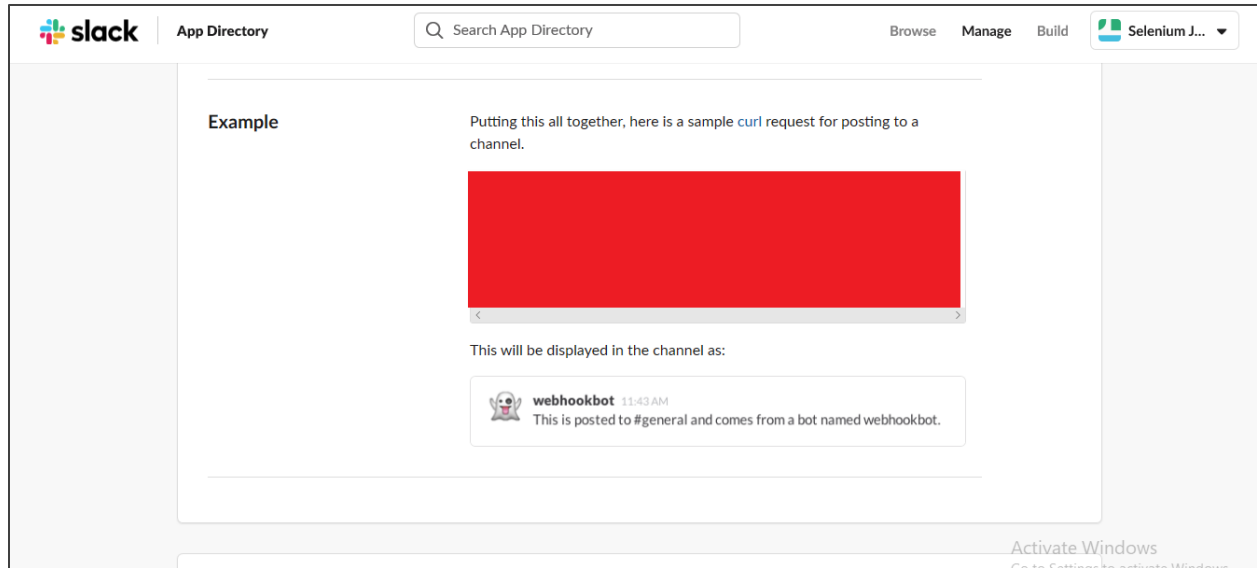


5. The landing page will be as below :

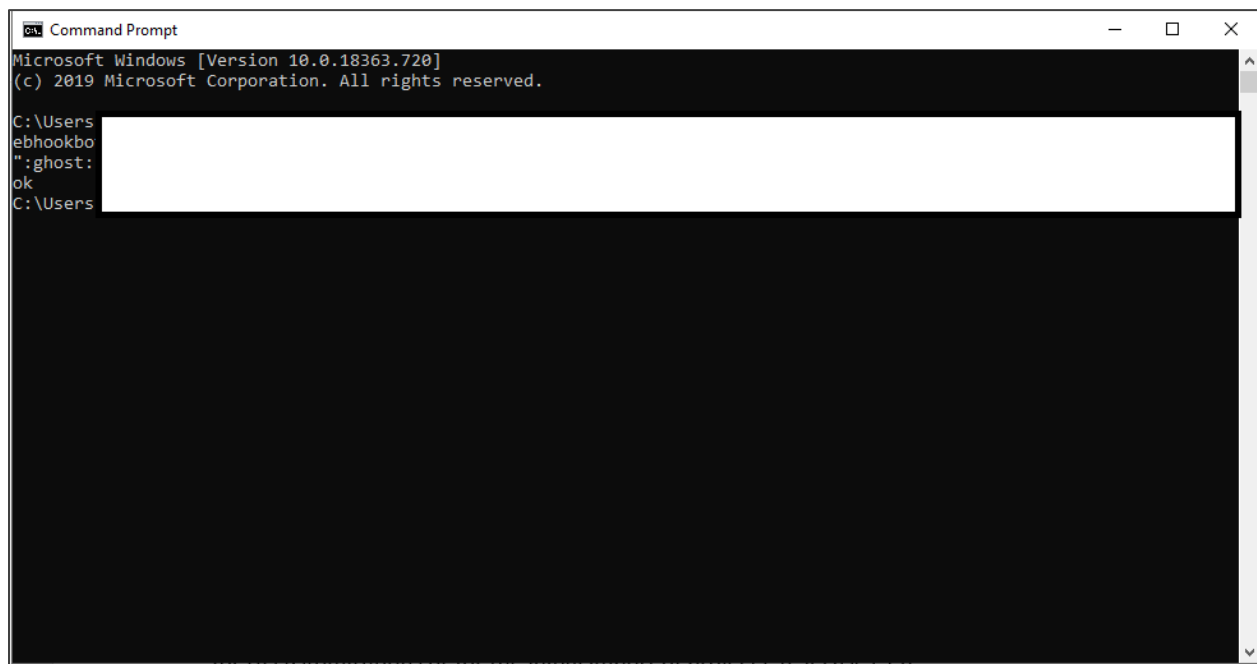
The screenshot shows the Slack App Directory interface. At the top, there's a search bar and navigation links like 'Browse', 'Manage', and 'Build'. The user 'Selenium J...' is logged in. The main heading is 'Incoming WebHooks', with a sub-header 'Added by seshaphani.matcha on May 22nd, 2020'. Below this, there's a description of Incoming WebHooks and a link to 'Message Attachments'. A warning box titled 'New to Slack integrations?' provides additional guidance. A 'Setup Instructions' box is visible at the bottom, with a 'close' button. An 'Activate Windows' watermark is present in the bottom right corner.

6. Copy “WebHook URL” value and “Example” value available in this page and click on “Save Settings”:

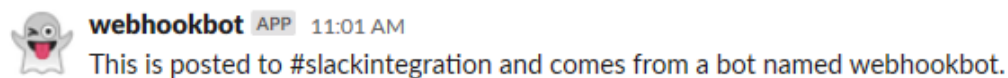
This screenshot shows the 'Setup Instructions' section of the Slack App Directory page. It includes a 'Webhook URL' field, which is highlighted with a red rectangle. Below this, there's a 'Sending Messages' section that explains two options for sending data to the Webhook URL: sending a JSON string as the 'payload' parameter or as the body of a POST request. A note at the bottom states that for a simple message, the JSON payload could contain a 'text' property. The 'Activate Windows' watermark is also visible in the bottom right corner.



7. Open command prompt and paste the copied Example value from step 6

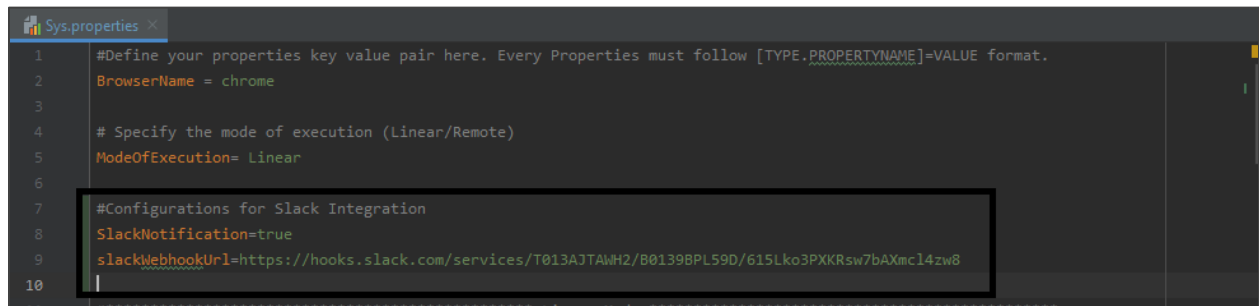


8. Ensure that 'ok' message is displayed on command prompt.
9. Goto slack, we can find the message sent to Slack channel as below which confirms that the webhook is properly configured.



Note: Step 7, 8 and 9 are for testing the slack connection.

10. Goto sys.properties in Selenium Java Framework -> set “true” for “SlackNotification” and paste the copied WebHookURL(Refer to Step 6) for “slackWebhookUrl” in sys.properties.

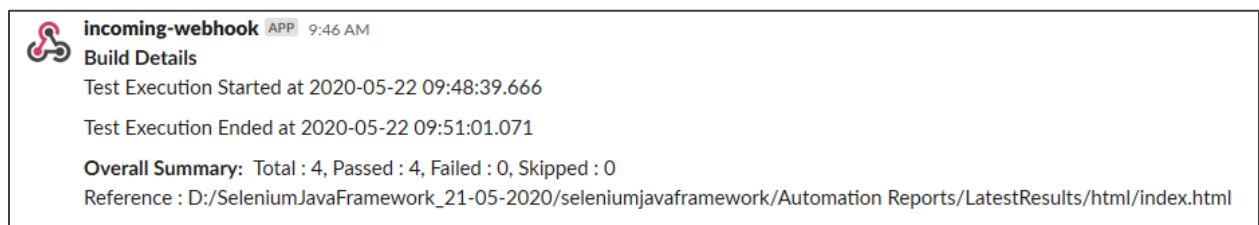


```

1  #Define your properties key value pair here. Every Properties must follow [TYPE.PROPERTYNAME]=VALUE format.
2  BrowserName = chrome
3
4  # Specify the mode of execution (Linear/Remote)
5  ModeOfExecution= Linear
6
7  #Configurations for Slack Integration
8  SlackNotification=true
9  slackWebhookUrl=https://hooks.slack.com/services/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
10

```

11. Execute your tests from Selenium Java Framework. We will get the notifications as below:



incoming-webhook APP 9:46 AM

Build Details

Test Execution Started at 2020-05-22 09:48:39.666

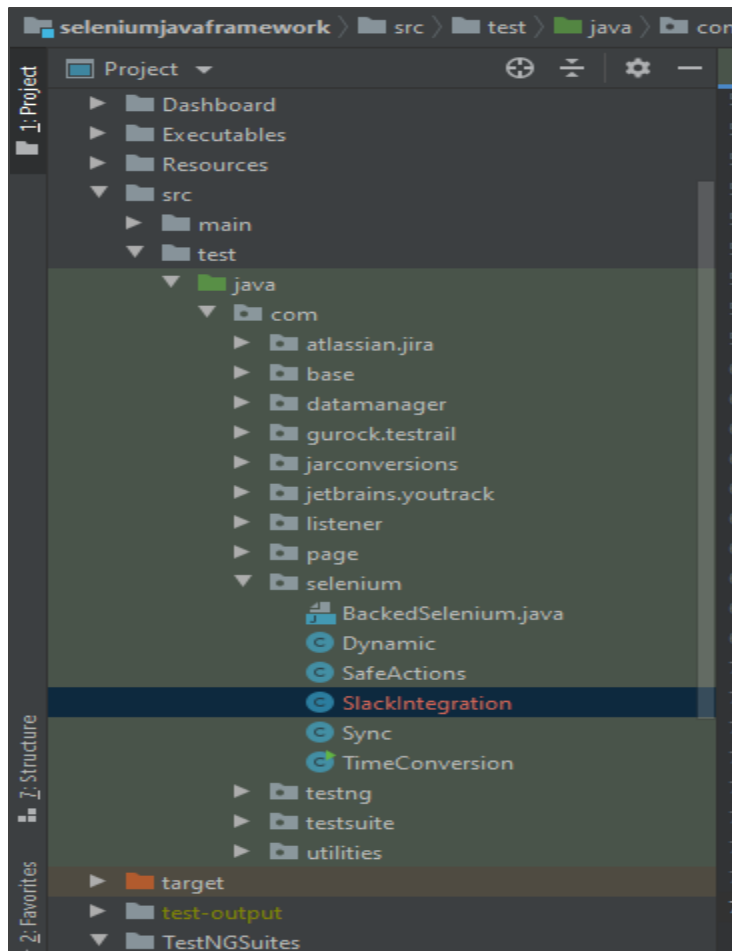
Test Execution Ended at 2020-05-22 09:51:01.071

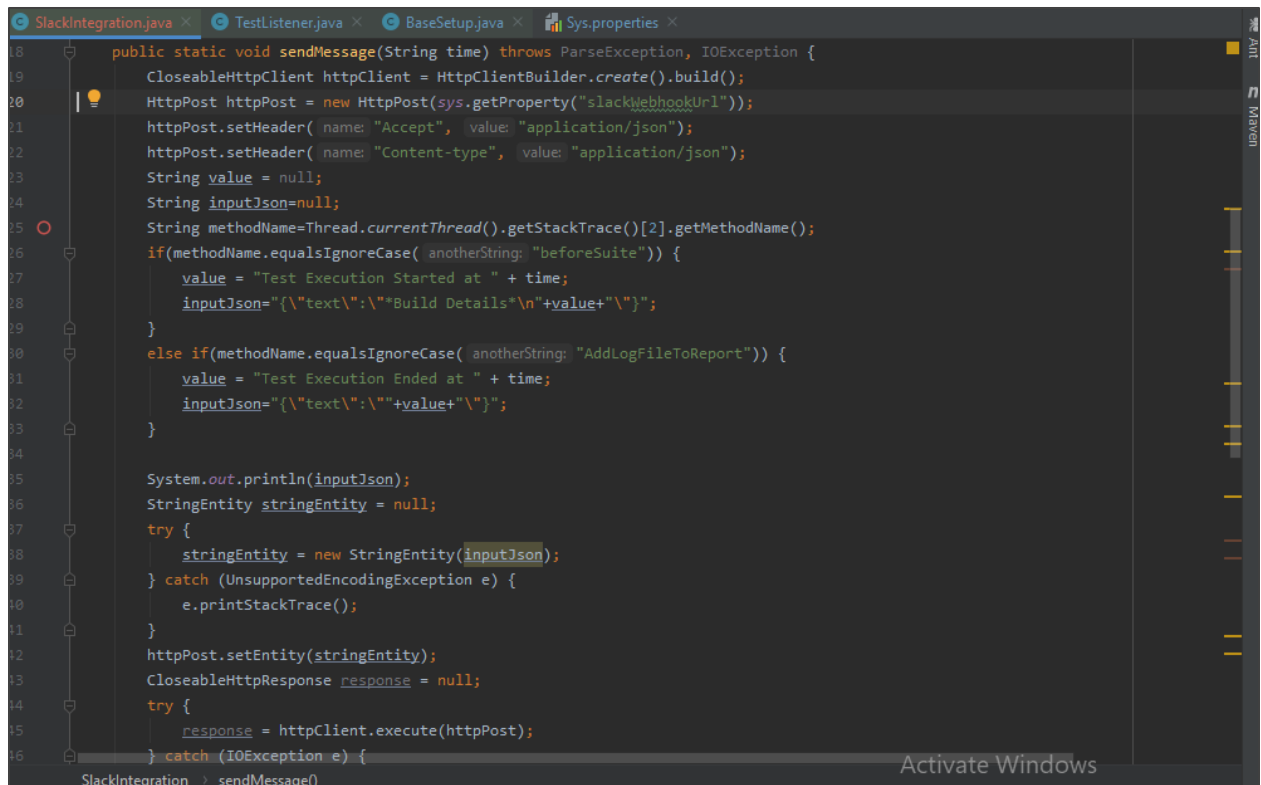
Overall Summary: Total : 4, Passed : 4, Failed : 0, Skipped : 0

Reference : D:/SeleniumJavaFramework_21-05-2020/seleniumjavaframework/Automation Reports/LatestResults/html/index.html

3. Code Implementation For Slack Integration:

1. Created SlackIntegration class to send Test Exection Start time, End Time, Test TotalCount, PassCount, FailCount, SkipCount and Reference to access the index.html file available from Latest results folder in Automation reports.

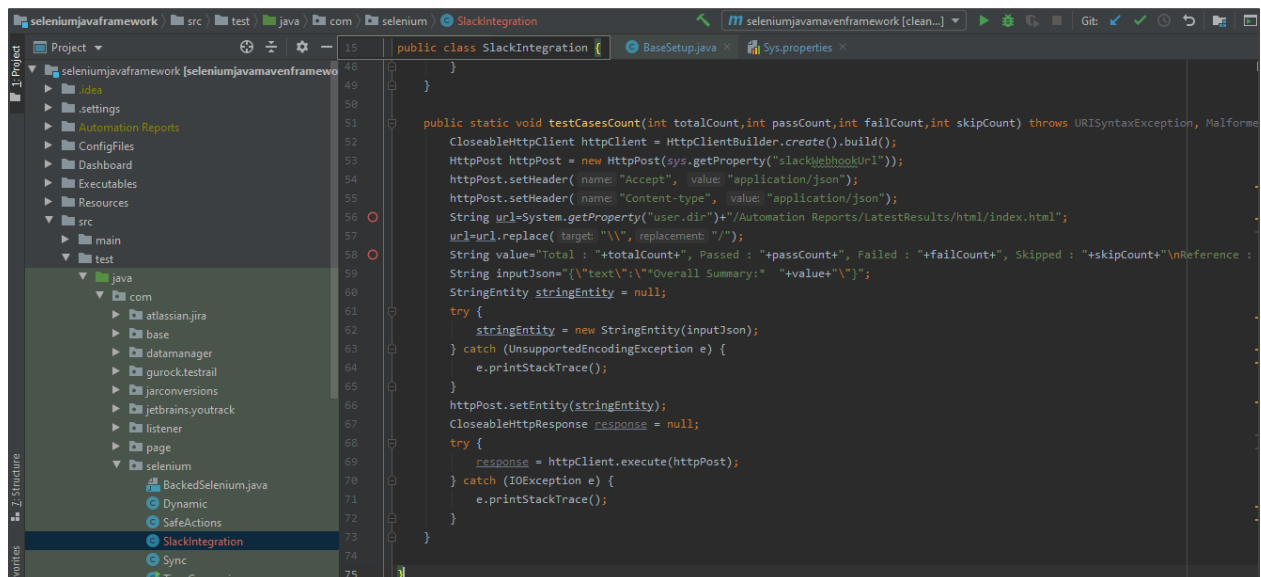




```

18 public static void sendMessage(String time) throws ParseException, IOException {
19     CloseableHttpClient httpClient = HttpClientBuilder.create().build();
20     HttpPost httpPost = new HttpPost(sys.getProperty("slackWebhookUrl"));
21     httpPost.setHeader( name: "Accept", value: "application/json");
22     httpPost.setHeader( name: "Content-type", value: "application/json");
23     String value = null;
24     String inputJson=null;
25     String methodName=Thread.currentThread().getStackTrace()[2].getMethodName();
26     if(methodName.equalsIgnoreCase( anotherString: "beforeSuite")) {
27         value = "Test Execution Started at " + time;
28         inputJson="{\"text\": \"*Build Details*\n"+value+"\"}";
29     }
30     else if(methodName.equalsIgnoreCase( anotherString: "AddLogFileToReport")) {
31         value = "Test Execution Ended at " + time;
32         inputJson="{\"text\": \"*"+value+"\"}";
33     }
34
35     System.out.println(inputJson);
36     StringEntity stringEntity = null;
37     try {
38         stringEntity = new StringEntity(inputJson);
39     } catch (UnsupportedEncodingException e) {
40         e.printStackTrace();
41     }
42     httpPost.setEntity(stringEntity);
43     CloseableHttpResponse response = null;
44     try {
45         response = httpClient.execute(httpPost);
46     } catch (IOException e) {
47
48     }
49 }

```

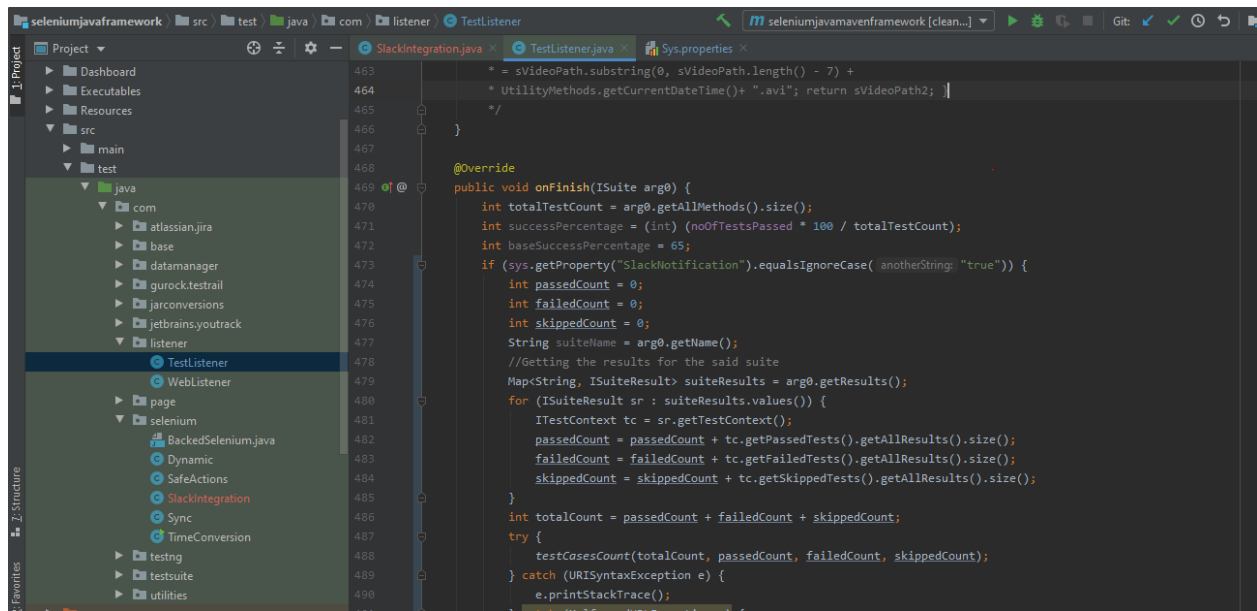


```

15 public class SlackIntegration {
16
17     }
18
19     public static void testCasesCount(int totalCount,int passCount,int failCount,int skipCount) throws URISyntaxException, MalformedURLException {
20         CloseableHttpClient httpClient = HttpClientBuilder.create().build();
21         HttpPost httpPost = new HttpPost(sys.getProperty("slackWebhookUrl"));
22         httpPost.setHeader( name: "Accept", value: "application/json");
23         httpPost.setHeader( name: "Content-type", value: "application/json");
24         String url=System.getProperty("user.dir")+"/Automation Reports/LatestResults/html/index.html";
25         url=url.replace( target: "\\", replacement: "/");
26         String value="Total : "+totalCount+", Passed : "+passCount+", Failed : "+failCount+", Skipped : "+skipCount+"\nReference : "+sys.getProperty("slackWebhookUrl");
27         String inputJson="{\"text\": \"*Overall Summary:*  "+value+"\"}";
28         StringEntity stringEntity = null;
29         try {
30             stringEntity = new StringEntity(inputJson);
31         } catch (UnsupportedEncodingException e) {
32             e.printStackTrace();
33         }
34         httpPost.setEntity(stringEntity);
35         CloseableHttpResponse response = null;
36         try {
37             response = httpClient.execute(httpPost);
38         } catch (IOException e) {
39             e.printStackTrace();
40         }
41     }
42 }

```

2. Updating the Slack channel with execution status total count from TestListener class as follows:



3. Updating the Test Start time and Test End time from BaseSetup class as below:

