# Topic-wise assignments:

## Java and Java 8:

1 - Implement stack using queues. -- Done  
2 - Implement Oops concept for Car Class.-Done  
3 - What is Object-Level Locking and Class-Level Locking - Done  
4 - What are Java-8 related changes for Interfaces?-done

5 - What is double-checked locking?

Eqv to Lazy Instantiation in Singleton class

class A{

private static A obj;

private A(){}

public static A getA(){

if (obj == null){

synchronized(Singleton.class){

if (obj == null){

obj = new Singleton();//instance will be created at request time

}

}

}

return obj;

}

public void doSomething(){

//write your code

}

}

6 - What are the disadvantages of normal singleton design pattern implementation and ways to overcome the same.

If singleton class is Serializable, you can serialize the singleton instance. Once it is serialized, you can deserialize it but it will not return the singleton object.

To resolve this issue, you need to override the readResolve() method that enforces the singleton. It is called just after the object is deserialized. It returns the singleton object.

public class A implements Serializable {

//your code of singleton

protected Object readResolve() {

return getA();

}

}

7. - (Lambda expression usage): Adding values/data in application popup-done  
8. Java 8 features:  
Implement with Collections:  
a. forEach() method in Iterable interface (Implemented with Collections) - Done  
b. Implementing multiple inheritance in Classes : Provide implementation logic in the class implementing the interfaces-done  
c. Add reference to static method using Functional interface {Using :: keywork)-done  
d. Performing filter operation through specified conditions, while booking a ticket on Cleartrip application, using the Stream API (java.util.stream)-done  
e. Display the current time (By converting datetime objects to strings) parse dates and time in 2 different time zones (java.time.zone).-done  
f. Implement Comparator interface and print the given strong in reverse order-done

## Selenium:

1 - What are the other ways to insert text into a text-box, if sendKeys is not allowed? For e.g.: Using JavascriptExecutor, Actions Class

driver.execute\_script("document.getElementById('q').value='value here'");

Actions action = new Actions(driver);

action.sendKeys(Keys.ENTER).build().perform();

Actions action = new Actions(driver);

action.keyDown(Keys.CONTROL).sendKeys("a").keyUp(Keys.CONTROL).perform();

2 - How to wait and check for presence of an element on a page, every second(give code)? For E.g.

FluentWait

public void type(WebDriver driver, final String value, final String Inattribute) throws InterruptedException {

driver.manage().timeouts().implicitlyWait(0,TimeUnit.SECONDS);

final Wait<WebDriver> waitExt = new FluentWait<WebDriver>(driver)

.withTimeout(30, TimeUnit.SECONDS)

.pollingEvery(1, TimeUnit.SECONDS)

.ignoring(NoSuchElementException.class)

.ignoring(StaleElementReferenceException.class)

.ignoring(InvalidElementStateException.class)

.ignoring(NullPointerException.class)

.ignoring(WebDriverException.class);

logger.info("type:The Value [" + value + "] going to Enter into the Field [" + Inattribute + "]");

try {

final int retryLimit = 3;

boolean verifiedPass = false;

String finalAttributeValue = null;

for (int retryCnt=0; retryCnt < retryLimit; retryCnt++)

{

WebElement elem = waitExt.until(new Function<WebDriver, WebElement>() {

public WebElement apply(WebDriver driver) {

waitExt.until(ExpectedConditions.elementToBeClickable(getExactAttributeBY(Inattribute))).click();

// WebElement element = getElementWithoutAssertion(driver, Inattribute);

// element.click();

logger.info("type:clear");

//String tmpValue = waitExt.until(ExpectedConditions.elementToBeClickable(getExactAttributeBY(Inattribute))).getAttribute("value");

logger.info("type:tmpValue=" + waitExt.until(ExpectedConditions.elementToBeClickable(getExactAttributeBY(Inattribute))).getAttribute("value"));

waitExt.until(ExpectedConditions.refreshed(ExpectedConditions.elementToBeClickable(getExactAttributeBY(Inattribute)))).clear();

String tmpValue2 = waitExt.until(ExpectedConditions.elementToBeClickable(getExactAttributeBY(Inattribute))).getAttribute("value");

logger.info("type:tmpValue2=" + tmpValue2);

waitExt.until(ExpectedConditions.refreshed(ExpectedConditions.elementToBeClickable(getExactAttributeBY(Inattribute)))).click();

for (int charCnt = 0; charCnt < tmpValue2.length(); charCnt++) {

waitExt.until(ExpectedConditions.refreshed(ExpectedConditions.elementToBeClickable(getExactAttributeBY(Inattribute)))).sendKeys(Keys.BACK\_SPACE);

String tmpValue3 = waitExt.until(ExpectedConditions.refreshed(ExpectedConditions.elementToBeClickable(getExactAttributeBY(Inattribute)))).getAttribute("value");

logger.info("type:tmpValue3=" + tmpValue3);

}

logger.info("type:cleared");

waitExt.until(ExpectedConditions.refreshed(ExpectedConditions.elementToBeClickable(getExactAttributeBY(Inattribute)))).sendKeys(value);

logger.info("type:wait completed 1:" + waitExt.until(ExpectedConditions.refreshed(ExpectedConditions.elementToBeClickable(getExactAttributeBY(Inattribute)))).getAttribute("value").trim());

// element.sendKeys(value);

// logger.info("type:wait completed 2");

waitExt.until(ExpectedConditions.refreshed(ExpectedConditions.elementToBeClickable(getExactAttributeBY(Inattribute)))).sendKeys(Keys.TAB);

//element.sendKeys(Keys.TAB);

WebElement element =waitExt.until(ExpectedConditions.elementToBeClickable(getExactAttributeBY(Inattribute)));

logger.info("type:Sent TAB:" + element.getAttribute("value").trim());

return element;

}

});

logger.info("Verification started..");

//finalAttributeValue = elem.getAttribute("value").trim();

// finalAttributeValue = waitExt.until(ExpectedConditions.visibilityOfElementLocated(getExactAttributeBY(Inattribute))).getAttribute("value");

WebElement element1 = waitExt.until(ExpectedConditions.refreshed(ExpectedConditions.elementToBeClickable(getExactAttributeBY(Inattribute))));

finalAttributeValue = element1.getAttribute("value").trim();

logger.info("Obtained filled value..");

if (value.trim().equalsIgnoreCase(finalAttributeValue) || value.trim().contains(finalAttributeValue)) {

logger.info("Value [" + value + "] Entered and verified Successfully");

verifiedPass = true;

break;

}

else {

int logicalRetryCnt = retryCnt + 1;

logger.info("Retrying [" + String.valueOf(logicalRetryCnt) + "/" + String.valueOf(retryLimit) + "] because entered [" + value + "] but final value in field is [" + finalAttributeValue + "]");

verifiedPass = false;

// Assert.fail();

}

}

driver.manage().timeouts().implicitlyWait(30,TimeUnit.SECONDS);

if (verifiedPass == false){

logger.error("Entered [" + value + "] but final value in field is [" + finalAttributeValue + "]");

// Assert.fail();

}

} catch(StaleElementReferenceException e){

logger.info("Stale element exception caught");

wrap.wait(2000);

//Relocate element

WebElement element = driver.findElement(getExactAttributeBY(Inattribute));

//Validation

String finalAttribute = element.getAttribute("value").trim();

if (finalAttribute.equalsIgnoreCase(value.trim())) {

logger.info("Type successful");

}else {

logger.info("Final value: " + finalAttribute + " does not match with intended value: " + value);

// Assert.fail();

}

} catch (Exception e) {

logger.error("Failed entering Value [" + value + "]");

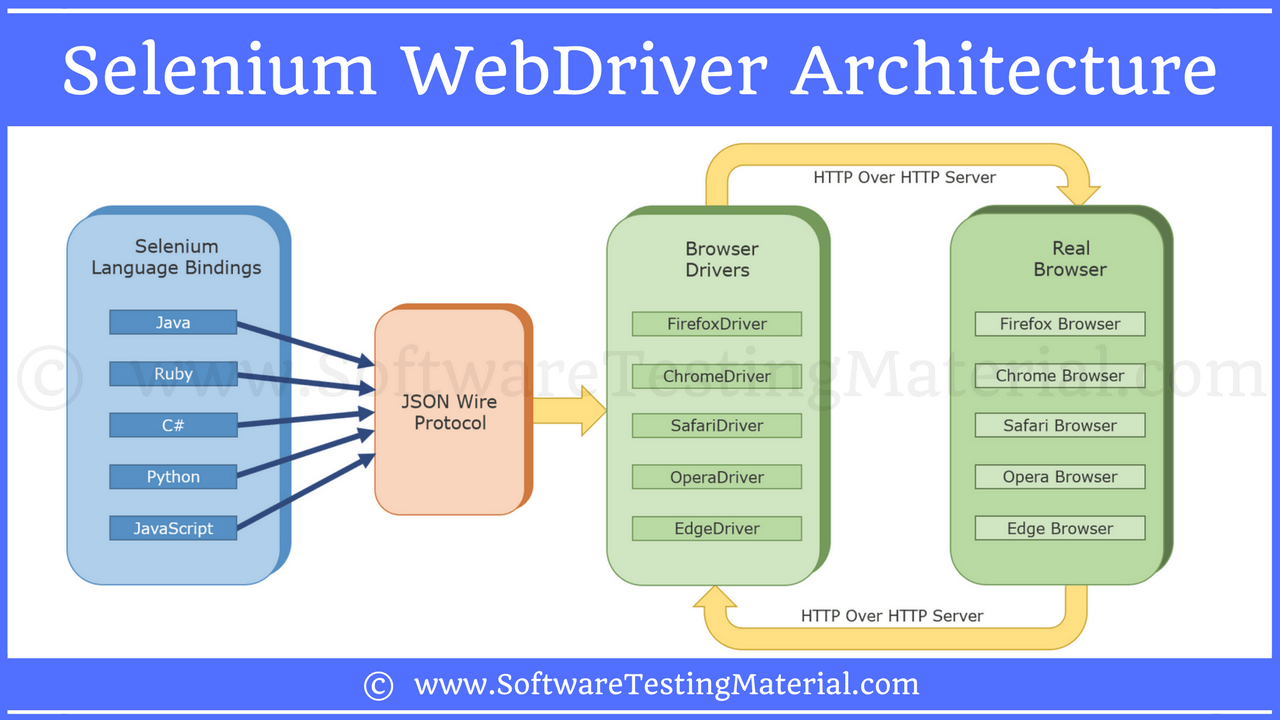
e.printStackTrace();

// Assert.fail();

}

}

3 - Explain firefox driver internal architecture



4 - How to drag and drop an element from one location to another? For E.g. Using Actions Class

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.interactions.Actions;

import org.testng.annotations.Test;

public class DragAndDrop {

WebDriver driver;

@Test

public void DragnDrop()

{

System.setProperty("webdriver.chrome.driver"," E://Selenium//Selenium\_Jars//chromedriver.exe ");

driver= new ChromeDriver();

driver.get("http://demo.guru99.com/test/drag\_drop.html");

//Element which needs to drag.

WebElement From=driver.findElement(By.xpath("//\*[@id='credit2']/a"));

//Element on which need to drop.

WebElement To=driver.findElement(By.xpath("//\*[@id='bank']/li"));

//Using Action class for drag and drop.

Actions act=new Actions(driver);

//Dragged and dropped.

act.dragAndDrop(From, To).build().perform();

}

}

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.interactions.Actions;

import org.testng.annotations.Test;

public class DragAndDrop {

WebDriver driver;

@Test

public void DragnDrop()

{

System.setProperty("webdriver.chrome.driver","E://Selenium//Selenium\_Jars//chromedriver.exe");

driver= new ChromeDriver();

driver.get("http://demo.guru99.com/test/drag\_drop.html");

//Element(BANK) which need to drag.

WebElement From=driver.findElement(By.xpath("//\*[@id='credit2']/a"));

//Using Action class for drag and drop.

Actions act=new Actions(driver);

//Drag and Drop by Pixel.

act.dragAndDropBy(From,135, 40).build().perform();

}

}

5 - "Write code to implement testng listeners? -- Use below:  
IAnnotationTransformer  
IAnnotationTransformer2  
IHookable  
IInvokedMethodListener  
IMethodInterceptor  
IReporter  
ISuiteListener  
ITestListener"

<http://toolsqa.com/selenium-webdriver/testng-listeners/>



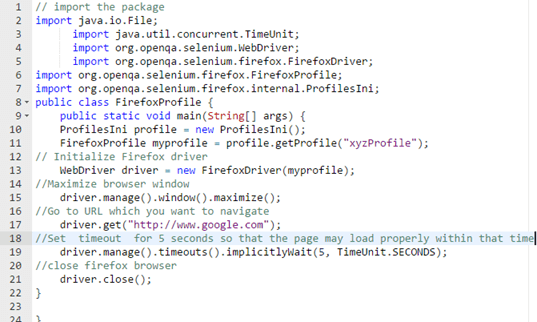
6 - How to create a new fireFox profile and set different capabilities for the same?

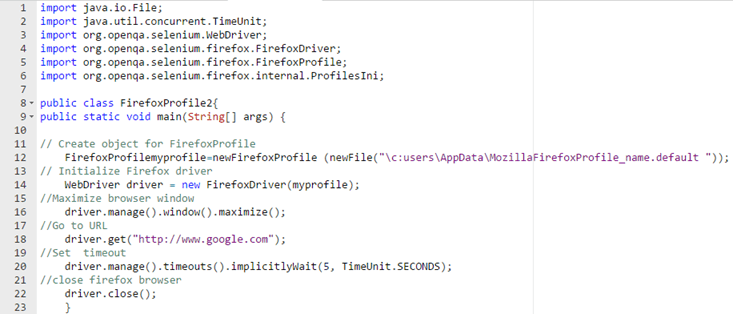
<https://www.guru99.com/firefox-profile-selenium-webdriver.html>

ProfilesIni profile = new ProfilesIni();

FirefoxProfile myprofile = profile.getProfile("xyzProfile");

WebDriver driver = new FirefoxDriver(myprofile)





7 - How to take screenshots using selenium-java?

public String captureScreenShot(WebDriver driver,

String testName) {

SeleniumScreenshotUtil.takeScreenshot(driver, genieScenario);

/\*

String fileSeperator = System.getProperty("file.separator");

String screenShotName;

Date dNow = new Date();

SimpleDateFormat ft = new SimpleDateFormat("dd\_MM\_yyyy\_'at'\_hh\_mm\_ss\_a");

String screenshotTakenTime = ft.format(dNow);

screenShotName = testName + "\_" + screenshotTakenTime + ".jpg";

// screenShotName=testName+".jpg";

try {

File file = new File("Screenshots" + fileSeperator + "Results");

if (!file.exists()) {

logger.info("File created " + file);

file.mkdir();

}

logger.info("Screenshots are stored in path:" + file.getAbsolutePath());

File screenshotFile = ((TakesScreenshot) driver).getScreenshotAs(OutputType.FILE);

File targetFile = new File("Screenshots" + fileSeperator + "Results" + fileSeperator + testName, screenShotName);

FileUtils.copyFile(screenshotFile, targetFile);

return screenShotName;

} catch (Exception e) {

logger.error("An exception occured while taking screenshot " + e.getCause());

return null;

}\*/

return testName;

}

8 - How to operate on windows based pop-up? Is it feasible via Selenium? For Eg: Using AutoIT, Robot Class etc

/ Specify the file location with extension

StringSelection sel = new StringSelection("C:\\Users\\Desktop\\1.doc");

// Copy to clipboard

Toolkit.getDefaultToolkit().getSystemClipboard().setContents(sel,null);

System.out.println("selection" +sel);

**SciCode for AutoIT**

ControlFocus("[Class:#32770]","","Edit1")

Sleep(2000)

ControlSetText("[Class:#32770]","","Edit1","D:\Java\d.txt")

ControlClick ("[Class:#32770]","","Button1")

9 - Is it possible to automate captcha and bar-code? Please explain?

We can automate captcha and bar code using sikuli which is image based automation tool. Whenever the bar code/captcha image comes, let us store the image in file server location. Using OCR technique, read the image text then we can do our validations.

10 - "What are the various ways/alternatives in/using which, one can operate on drop-downs?

package newpackage;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.support.ui.Select;

import org.openqa.selenium.By;

public class accessDropDown {

public static void main(String[] args) {

System.setProperty("webdriver.firefox.marionette","C:\\geckodriver.exe");

String baseURL = "http://demo.guru99.com/test/newtours/register.php";

WebDriver driver = new FirefoxDriver();

driver.get(baseURL);

Select drpCountry = new Select(driver.findElement(By.name("country")));

drpCountry.selectByVisibleText("ANTARCTICA");

//Selecting Items in a Multiple SELECT elements

driver.get("http://jsbin.com/osebed/2");

Select fruits = new Select(driver.findElement(By.id("fruits")));

fruits.selectByVisibleText("Banana");

fruits.selectByIndex(1);

}

}

((JavascriptExecutor) driver).executeScript("return document.getElementById('id').selectedIndex = '" + index + "';)

For example, using Select class, Actions Class, Normal SendKeys, Using JavaScriptExecutor, Creating List of WebElements etc"  
11 - How to handle changing element locator values/dynamic locator values? For Example, using Preceding-sibling, Descendant, Ancestor etc

Xpath=//\*[@type='text']//following::input

Xpath=//\*[@type='text']//following::input[1]

Xpath=//\*[text()='Enterprise Testing']//ancestor::div

Xpath=//\*[text()='Enterprise Testing']//ancestor::div[1]

Xpath=//\*[@id='java\_technologies']/child::li

Xpath=//\*[@id='java\_technologies']/child::li[1]

Xpath=//\*[@type='submit']//preceding::input

Xpath=//\*[@type='submit']//preceding::input[1]

xpath=//\*[@type='submit']//following-sibling::input

xpath=//\*[@type='submit']//preceding-sibling::input

Xpath=//\*[@id='rt-feature']//parent::div

Xpath=//\*[@id='rt-feature']//parent::div[1]

Xpath =//\*[@type='password']//self::input

Xpath=//\*[@id='rt-feature']//descendant::a

Xpath=//\*[@id='rt-feature']//descendant::a[1]

12 - Creating the firefox profile by setting up the capabilities to set the download folder (with some properties to be set for the acceptance of security certificate)

importorg.openqa.selenium.WebDriver;

importorg.openqa.selenium.ie.InternetExplorerDriver;

importorg.openqa.selenium.remote.DesiredCapabilities;

public class IEtestforDesiredCapabilities {

public static void main(String[] args) {

//it is used to define IE capability

DesiredCapabilities capabilities = DesiredCapabilities.internetExplorer();

capabilities.setCapability(CapabilityType.BROWSER\_NAME, "IE");

capabilities.setCapability(InternetExplorerDriver.

INTRODUCE\_FLAKINESS\_BY\_IGNORING\_SECURITY\_DOMAINS,true);

System.setProperty("webdriver.ie.driver", "C:\\IEDriverServer.exe");

//it is used to initialize the IE driver

WebDriver driver = new InternetExplorerDriver(capabilities);

driver.manage().window().maximize();

driver.get("http://gmail.com");

driver.quit();

}

}

13 - Creating test cases by using/exhausting parallel, invocationCount, DataProviderThreadPoolCount attributes/annotations (Explore all the properties/annotations for <test and <suit tags)

Parallel execution using testng

@test(invocationCount=3)🡪 Test will get executed 3 times based on the invocation count.

## Spring:

1 - Write Spring-Java code to implement xml based & annotation based auto-wiring. -Done  
2 - Write Spring-Java code to implement @Component annotation.  
3 - Write Spring-Java code to implement Constructor and Setter/Getter based dependency-injection.-Done  
4 - Write Spring-Java code to showcase collection-injection.-Done

## GIT:

1 - Git Installation & Configuration  
2 - Create local and remote GIT Repository -- You may use GitLab or similar for creating remote repository

Create branch using UI itself.

Git clone clone url

Git add -A

Git commit -m “message”

Git status

Git push origin master

Git pull

Git checkout

Git fetch

Git init

Git branch

3 - Map your selenium project(Implement BDD & Create some sample test-cases) with your local GIT Repository - Done  
4 - Perform GIT Operations as per this link -> <https://kb.epam.com/display/EPMCES/Git> (This includes the basic Git CI implementation as well) – Done

Tried commands:

Git clone clone url

Git add -A

Git commit -m “message”

Git status

Git push origin master

Git pull

Git checkout

Git fetch

Git init

Git branch

## Sonar:

1. Configure SonarCloud as explained in : <https://about.sonarcloud.io/>  
2. Execute the Sonar job, analyse the Blocker, Critical and Major issues, and fix them.

## Maven:

1 - Convert your 'Assignment-Git' Project code into Maven Project and work on different phases (validate, compile, test etc).  
2 - Try integrating Maven & Git  
3 - Set System Properties/Code related properties using Maven

1. <project>
2. [...]
3. <build>
4. <plugins>
5. <plugin>
6. <groupId>org.apache.maven.plugins</groupId>
7. <artifactId>maven-surefire-plugin</artifactId>
8. <version>3.0.0-M1</version>
9. <configuration>
10. <systemPropertyVariables>
11. <propertyName>propertyValue</propertyName>
12. <buildDirectory>${project.build.directory}</buildDirectory>
13. [...]
14. </systemPropertyVariables>
15. </configuration>
16. </plugin>
17. </plugins>
18. </build>
19. [...]
20. </project>

4 - Implement transitive dependency, Include exclude files/directories

<https://maven.apache.org/guides/introduction/introduction-to-dependency-mechanism.html#Transitive_Dependencies>



## Jenkins:

1 - Download Jenkins.war  
2 - Go through the folder where Jenkins.war downloaded.  
3 - Type "java -jar jenkins.war"  
4 - Please check logs in cmd  
5 - Type 'localhost:8080' in browser  
6 - Install required plugins  
7 - You will get default admin and password  
8 - Log in with admin  
9 - Install Maven related plugins  
10 - Click on New Item  
11 - Enter project name  
12 - Select freestyle project-Maven project  
13 - Enter description  
14 - 14)Select Source code management tool.  
15 - In Build Section,mention project pom.xml path  
16 - Set goals and options.  
17 - Install Testng Result plugin  
18 - Select Post-build Actions->Publish testng results  
19 - click on apply and save  
20 - Click on Build now  
21 - You are able to see results in Build details.  
22 - Now, Select Build Triggers->Build periodically check box.  
23 - Set Build Triggers->Build periodically  
24 - Set Values (15 13 \* \* \*)-schedule the build every day of every month of every year at the 15th minute of the 13th hour of the day, and track execution.  
25 - Now, execute some sample tests of your project code by integrating GIT, Maven and Jenkins. Please make Git related Jenkins configuration accordingly.  
26 - Now, perform parallel execution using selenium-grid, in combination with above mentioned steps.  
27 - Generate Serenity reports for your project

## API Automation:

1. Structure of a REST bases request and response

A HTTP Request has five major parts −

* **Verb** − Indicate HTTP methods such as GET, POST, DELETE, PUT etc.
* **URI** − Uniform Resource Identifier (URI) to identify the resource on server.
* **HTTP Version** − Indicate HTTP version, for example HTTP v1.1 .
* **Request Header** − Contains metadata for the HTTP Request message as key-value pairs. For example, client ( or browser) type, format supported by client, format of message body, cache settings etc.
* **Request Body** − Message content or Resource representation.

A HTTP Response has four major parts −

* **Status/Response Code** − Indicate Server status for the requested resource. For example 404 means resource not found and 200 means response is ok.
* **HTTP Version** − Indicate HTTP version, for example HTTP v1.1 .
* **Response Header** − Contains metadata for the HTTP Response message as key-value pairs. For example, content length, content type, response date, server type etc.
* **Response Body** − Response message content or Resource representation.

1. What are headers and how many different types can be there in headers.

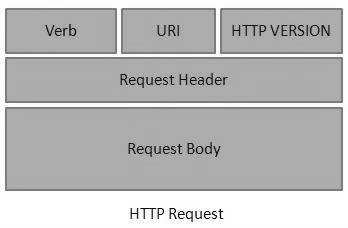
SOAP web services use XML for data exchange between the client application and a web service. A SOAP request consists of the root **Envelope** element that has two child elements - **Header** and **Body**. Header is an optional element that can contain some extra information to be passed to the web service. Body is a required element and contains data specific to the called web service method.

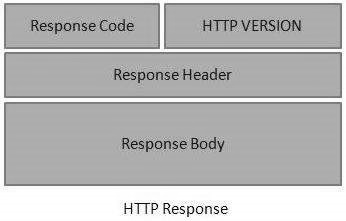
**Headers**. The **REST headers** and parameters contain a wealth of information that can help you track down issues when you encounter them. HTTP **Headers** are an important part of the **API** request and response as they represent the meta-data associated with the **API** request and response.

1. What’s Authorization? In how many ways authorization can be achieved?

 <http://toolsqa.com/rest-assured/authentication-and-authorization-in-rest-webservices/>

1. What message body, how many types it can be of





1. RequestSpecification class and its methods. In how many ways can we build RequestSpecification.

A HTTP Request has five major parts −

* **Verb** − Indicate HTTP methods such as GET, POST, DELETE, PUT etc.
* **URI** − Uniform Resource Identifier (URI) to identify the resource on server.
* **HTTP Version** − Indicate HTTP version, for example HTTP v1.1 .
* **Request Header** − Contains metadata for the HTTP Request message as key-value pairs. For example, client ( or browser) type, format supported by client, format of message body, cache settings etc.
* **Request Body** − Message content or Resource representation.

<http://static.javadoc.io/com.jayway.restassured/rest-assured/1.4.5/com/jayway/restassured/specification/RequestSpecification.html>



1. What’s a response, in how many ways can we get and manipulate the response?

A HTTP Response has four major parts −

* **Status/Response Code** − Indicate Server status for the requested resource. For example 404 means resource not found and 200 means response is ok.
* **HTTP Version** − Indicate HTTP version, for example HTTP v1.1 .
* **Response Header** − Contains metadata for the HTTP Response message as key-value pairs. For example, content length, content type, response date, server type etc.
* **Response Body** − Response message content or Resource representation.

1. What are Request and ResponseSpecBuilder?

<http://static.javadoc.io/com.jayway.restassured/rest-assured/1.2.3/com/jayway/restassured/builder/ResponseSpecBuilder.html>



1. How can we log the request and response? (there is a built in library in RestAssured)

As a part of implemented framework, it is available.

1. How to use conditional logging?

package com.javacodegeeks.snippets.core;

import java.util.logging.Logger;

import java.util.logging.Level;

import java.util.Date;

public class ConditionalLoggingExample {

private Logger logger = Logger.getLogger(ConditionalLoggingExample.class.getName());

public static void main(String[] args) {

ConditionalLoggingExample example = new ConditionalLoggingExample();

example.Method();

}

public void Method() {

// Check if the logging level before enter into the log

if (logger.isLoggable(Level.INFO)) {

logger.info("Entering executeMethod() at : " + new Date());

}

// Method functionality

for (int i = 0; i < 5; i++) {

for (int j = 0; j < 5; j++) {

System.out.print(i + j + " ");

}

System.out.println("");

}

if (logger.isLoggable(Level.INFO)) {

logger.info("Exiting executeMethod() at : " + new Date());

}

}

}

1. How to store a variable so that it can be used in another tests? – Creating static v/b. Creating an object and extends it to another class.
2. What’s JSONPath? How can we invoke it? How to use it for validation? – Available in Framework.
3. What’s XMLPath? How can we invoke it? How to use it for Validation?
4. The examples in this post are all based on the following XML response:
5. [?](https://www.ontestautomation.com/using-jsonpath-and-xmlpath-in-rest-assured/)

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8" ?>  <cars>      <car make="Alfa Romeo" model="Giulia">          <country>Italy</country>          <year>2016</year>      </car>      <car make="Aston Martin" model="DB11">          <country>UK</country>          <year>1949</year>      </car>      <car make="Toyota" model="Auris">          <country>Japan</country>          <year>2012</year>      </car>  </cars> |

@Test

public void checkCountryForFirstCar() {

    given().

    when().

        get("http://path.to/cars").

    then().

        assertThat().

        body("cars.car[0].country", equalTo("Italy"));

}

Similarly, we can check that the last car came on the market in 2012, using the [-1] index (this points us to the last item in a list):

[?](https://www.ontestautomation.com/using-jsonpath-and-xmlpath-in-rest-assured/)

|  |
| --- |
| @Test  public void checkYearForLastCar() {        given().      when().          get("<http://path.to/cars>").      then().          assertThat().          body("cars.car[-1].year", equalTo("2012"));  } |

**Extracting an attribute value**  
Just as easily, you can extract and check the value of an attribute in an XML document. If we want to check that the model of the second car in the list is ‘DB11’, we can do so using the ‘@’ notation:

[?](https://www.ontestautomation.com/using-jsonpath-and-xmlpath-in-rest-assured/)

|  |
| --- |
| @Test  public void checkModelForSecondCar() {        given().      when().          get("<http://path.to/cars>").      then().          assertThat().          body("cars.car[1].@model", equalTo("DB11"));  } |

**Counting the number of occurrences of a specific value**  
Now for something a little more complex: let’s assume we want to check that there’s only one car in the list that is made in Japan. To do this, we’ll need to apply a findAll filter to the country element, and subsequently count the number of items in the list using size():

[?](https://www.ontestautomation.com/using-jsonpath-and-xmlpath-in-rest-assured/)

|  |
| --- |
| @Test  public void checkThereIsOneJapaneseCar() {        given().      when().          get("<http://path.to/cars>").      then().          assertThat().          body("cars.car.findAll{it.country=='Japan'}.size()", equalTo(1));  } |

Likewise, we can also check that there are two cars that are made either in Italy or in the UK, using the in operator:

[?](https://www.ontestautomation.com/using-jsonpath-and-xmlpath-in-rest-assured/)

|  |
| --- |
| @Test  public void checkThereAreTwoCarsThatAreMadeEitherInItalyOrInTheUK() {        given().      when().          get("<http://path.to/cars>").      then().          assertThat().          body("cars.car.findAll{it.country in ['Italy','UK']}.size()", equalTo(2));  } |

**Performing a search for a specific string of characters**  
Finally, instead of looking for exact attribute or element value matches, we can also filter on substrings. This is done using the grep() method (very similar to the Unix command). If we want to check the number of cars in the list whose make starts with an ‘A’, we can do so like this:

[?](https://www.ontestautomation.com/using-jsonpath-and-xmlpath-in-rest-assured/)

|  |
| --- |
| @Test  public void checkThereAreTwoCarsWhoseMakeStartsWithAnA() {        given().      when().          get("<http://localhost:9876/xml/cars>").      then().          assertThat().          body("cars.car.@make.grep(~/A.\*/).size()", equalTo(2));  } |

1. Can we validate the response without storing response in the Response object?

We need to store it then validate the response.

1. What can be the structure of REST API automation framework using Cucumber? Please explain with the flow diagram.
   1. Available in Framework.
2. (Optional) How to use Postman for REST based testing – Tested in Google postman.
3. Demonstrate RestAssured usage for API automation (Create a new Test/Module on the same use case framework) - Available in Framework
4. Use cucumber and excel to feed the input data and expected result –

@SuppressWarnings("deprecation")

public static void convertExcelToMap(String FilePath,String FileName,String sheetName) throws IOException{

mydata.clear();

String MyFile= FilePath+File.separator+FileName;

logger.info("MyFile[before] :"+MyFile);

//String envName = System.getProperty("env");

URL resource = DBUtils.class.getClassLoader().getResource("ExcelData" + File.separator + FileName);

File file = null;

FileInputStream fin = null;

try {

file = new File(resource.toURI());

fin = new FileInputStream(file);

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (URISyntaxException e) {

e.printStackTrace();

}

logger.info("MyFile[after] :"+file.toString());

//FileInputStream fin = new FileInputStream(MyFile);

HSSFWorkbook book = new HSSFWorkbook(fin);

HSSFSheet sheet = book.getSheet(sheetName);

/\*for(int i=0;i<sheet.getPhysicalNumberOfRows();i++)

{

Row currentRow = sheet.getRow(i);

for(int j=0;j<currentRow.getPhysicalNumberOfCells();j++)

{

Cell currentCell = currentRow.getCell(j);

switch (currentCell.getCellType())

{

case Cell.CELL\_TYPE\_STRING:

System.out.print(currentCell.getStringCellValue() + "|");

break;

case Cell.CELL\_TYPE\_NUMERIC:

System.out.print(currentCell.getNumericCellValue() + "|");

break;

case Cell.CELL\_TYPE\_BLANK:

System.out.print("<blank>|");

break;

}

}

System.out.println("\n");

}\*/

rowCount = sheet.getPhysicalNumberOfRows();

Row HeaderRow = sheet.getRow(0);

for (int i = 1; i < sheet.getPhysicalNumberOfRows(); i++) {

Row currentRow = sheet.getRow(i);

HashMap<String, String> currentHash = new HashMap<String, String>();

for (int j = 0; j < currentRow.getPhysicalNumberOfCells(); j++) {

Cell currentCell = currentRow.getCell(j);

switch (currentCell.getCellType()) {

case Cell.CELL\_TYPE\_STRING:

currentHash.put(HeaderRow.getCell(j).getStringCellValue(), currentCell.getStringCellValue());

break;

case Cell.CELL\_TYPE\_NUMERIC:

currentHash.put(HeaderRow.getCell(j).getStringCellValue(), String.valueOf(currentCell.getNumericCellValue()));

break;

}

}

mydata.add(currentHash);

//System.out.println("\n\n"+currentHash);

}

//logger.info(mydata);

//logger.info(mydata.get(0));

}

public static String readColumn(String column,int row){

logger.info("Row id is" + row);

HashMap<String, String> map = mydata.get(row);

String result=null;

for (Entry<String, String> entry : map.entrySet()) {

if(entry.getKey().equalsIgnoreCase(column)){

logger.info(entry.getValue());

result = entry.getValue();

}

}

return result;

}

1. [POST]Post a tweet using twitter public API - Available in framework
2. [POST]Post a tweet using twitter public API with wrong credentials(negative) – Wrong credentials message.
3. [DELETE]Delete the same tweet – 204 error message.
4. [GET]get last 10 published tweet and display those on console. If there are less than 10 tweets, display what’s available – Available in framework
5. API testing refresher

<https://www.guru99.com/api-testing.html>



1. What is SOAP
2. What are constraints in SOAP?
3. What is REST, Difference between SOAP and REST

|  |  |  |
| --- | --- | --- |
| **No.** | **SOAP** | **REST** |
| 1) | SOAP is a **protocol**. | REST is an **architectural style**. |
| 2) | SOAP stands for **Simple Object Access Protocol**. | REST stands for **REpresentational State Transfer**. |
| 3) | SOAP **can't use REST** because it is a protocol. | REST **can use SOAP** web services because it is a concept and can use any protocol like HTTP, SOAP. |
| 4) | SOAP **uses services interfaces to expose the business logic**. | REST **uses URI to expose business logic**. |
| 5) | **JAX-WS** is the java API for SOAP web services. | **JAX-RS** is the java API for RESTful web services. |
| 6) | SOAP **defines standards**to be strictly followed. | REST does not define too much standards like SOAP. |
| 7) | SOAP **requires more bandwidth** and resource than REST. | REST **requires less bandwidth** and resource than SOAP. |
| 8) | SOAP **defines its own security**. | RESTful web services **inherits security measures** from the underlying transport. |
| 9) | SOAP **permits XML** data format only. | REST **permits different** data format such as Plain text, HTML, XML, JSON etc. |
| 10) | SOAP is **less preferred** than REST. | REST **more preferred** than SOAP. |

1. What’s request metadata – Refer Q No 1
2. REST request and response structure - Refer Q No 1

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

# Use-Case:

Automation round trip multi-city flight reservation ticket, using the ClearTrip Application.

**Expected Output:**

\*Maven

1) Create a Maven Project

2) Add required dependencies -- Serenity, JBehave-Core/Cucumber, Junit, TestNG, Maven Plugins, Spring, Rest-Assured, Hemcrest etc

\*BDD

3) Create/Write Story/Feature for any 2 scenarios from the below list for ClearTrip Application (<https://www.cleartrip.com/>):

\*Java & Selenium (Java 8 features to be implemented)

3.a) Use collections for storing/retrieving data from excel/application. For example -- Various Dropdowns like Adults, Children, Infants etc

3.b) Use OOPs Concepts like Interfaces, Inheritance, Polymorphism wherever applicable.

3.c) Follow common-coding-lags sheet(shared already) wherever possible.

Scenarios:

a. Search Flights One-Way

b. Search Flights Round-Trip

c. Search Flights Multi-City

\*Test Designs using Excel

4) Create TestData Repository for the scenarios being automated. Use Excel for reading/writing data.

\*Java & Selenium

5) Implement page object model for ClearTrip Application. Try using preceding-sibling, descendants, ancestor etc within element locator values. Try using all if possible.

\*API

5.1) For rest Web-Services test-cases pertaining to above mentioned scenarios, please validate input & output data responses. For example, Status-Code checks, header-level validations, content validations etc.

\*Java & Selenium

6) Create Step-Definition classes.

\*Java & Selenium

7) Create workflow classes. Example -- Read Test-Data & Insert various element values within Clear-Trip app, on Search-Flights navigations etc.

\*Java & Selenium

8) Create Selenium utility classes. -- For example: Waits, Actions Class, Select Class, JavaScript Executor etc. Write code to handle Ajax Calls efficiently.

\*Spring - Autowiring

9) Use Spring container for wiring required configurations with respective classes (use @Autowired & @Component as well, wherever applicable). All POM Pages & Step Definition Classes should be pre-initialized before usage.

Sample: UserName Password, Application URL etc.

\*TestNG & script execution

10) Include Runner file into your pom.xml

\*Git

11) Create a fresh Local & Remote git repository, attach your project with the local git repository and push your entire project-code over git. You can use gitHub.

\*Jenkins

12) Install Jenkins and create new Maven Project. Provide required details. Select Source Code Management as GIT, provide your GIT repository URL. Add Git Credentials if required.

\*Git

13) Check for required installation & configuration related to Git, Maven & Jenkins for Jenkins. Do required Global Tools configurations settings(). Afterwards, Under Build Section, provide Maven Version and Goals and save the configuration.

\*Jenkins

14) Trigger Jenkins Job manually as well as using Cron-Jobs for build scheduling.

15) Validate Serenity Reports generation and execution status, along with analysis of failed tests.