Important Notes/Points while creating Selenium Framework

1. **Framework Folder Structure:**

* *Step 1. All packages needs to be created under Src/Test/Java*

1. **BaseTest** – We will be having 1 BaseTest.Java file which will get extended to all Test Cases. This java file will have initial setup including all reference variable initializations like WebDriver, Wait, Property, FileInputStream etc.. and tear down method (to quit the brower)

2. **Utilities** – Utilities package will have one or more java files which will have all common methods like excel reader, taking screenshots, capturing current date etc.. or other java utilities

3. **TestCases** – Test Case may have multiple packages based on screens or testing type. We will be capturing all application test case under it and every Test case java file will inherit (extend) Base Class.java file

4. **Listeners** -

* *Step 2. Under Resources these folders need to be created -*

1. Properties,

2. Excel or test data,

3. Executables, Runners

4. Logs

5. Screenshots

1. Explain Base Class – What needs to be covered in Base Class.java
2. **Initializations of following variables:**  
   - public static WebDriver driver:  
   - public static Properties config = new Properties();  
   - public static Properties or = new Properites();  
   - FileInputStream fis;  
   - WebDriverWait wait;  
   - Logger log = Logger.getLogger("devpinoyLogger"); /\* for Log4J \*/
3. **Methods**:  
   @BeforeSuite  
   public void Setup(){  
   if(driver ==null){  
   fis = new FileInputStream("Location of config property file under Resource Folder of the project)  
   config.load(fis);  
   log.debug("Loaded Config property file !!"); /\*\*\*\*\*\*\*\*\*\*\*\*\*\*To track the Log \*\*\*\*\*\*\*\*\*\*\*\*/  
   fis = new FileInputStream("Location of or property file under Resource Folder of the project)  
   or.load(fis);  
   log.debug("Loaded Or property file !!"); /\*\*\*\*\*\*\*\*\*\*\*\*\*\*To track the Log \*\*\*\*\*\*\*\*\*\*\*\*/  
   if(config.getProperty("browser").equals("chrome"){  
   driver= new ChromeDriver(); } else if(config.getProperty("browser").equals("firefox"){  
   log.debug("Loaded Chrome webdriver!!");  
   dirver = new FirefoxDriver(); }}  
   log.debug("Loaded Chrome webdriver!!"); /\*\*\*\*\*\*\*\*\*\*\*\*\*\*To track the Log \*\*\*\*\*\*\*\*\*\*\*\*/  
   driver.get(config.getProperty("websiteurl");  
   driver.manage.window.maximize();  
     
   @AfterSuite  
   public void teardown(){  
   driver.quit();}
4. **How to load and read Property files**

* public static Properties config = new Properties();
* public static Properties or = new Properites();
* public static FileInputStream = new FileInputStream(“path”);
* config.load(fis);

1. **How to implement common method for Wait (Explicit or Implicit)**
2. **How to implement common method for Taking Screenshot**
3. **How to add common method for Assertion**

*1. Create a method under Base or Utilities file which will check that particular element is available or not. Such as:*   
public boolean isElementAvailable(By by)  
{  
try{  
driver.findElementBy(by);  
return true;  
catch(NoSuchElementPresent e)  
{ return false;}}  
*2. Call this assertion method where we want to verify any element in Test cases files such as:*

Assert.assertTrue(isElementAvailable(or.getProperty("Loginbutton"), "Unable to find Login button");

1. **How to implement Logs**

To implement Logs in Framework, first of all, we need to add Log4j dependency in project POM.xml file. Then we need a Builder file i.e. log4j.property file which needs to be added under Src/Test/Main directly. Also we need to create 2 files (application.log and selenium.log) under Resources Log Folder. Under Builder file, we need to provide these 2 files location.  
After that we need declare Logger Log = Logger.getLogger() under Base File. After that after each steps (totally depends on user) we need to add log.debug("Msg"); Thats all.

- Add Log4J dependency in POM.xml  
- Add 2 log files (Applicaton.log and Selenium.log) under Logs folder under Resources package  
- Need Log4j.properties file and add under your project src/Test/Main folder  
- Change the path for both files Application.log & Selenium.log location path in Log4j.properties file  
- Initialize Logger (Logger log = Logger.getLogger("devpinoyLogger"):

1. **How to implement ReportNG Reports**

In pom xml file, add dependency for ReportNG and Google Juice  
In your TestNG POM XML file, add ReportNG Listeners next to Suite Name  
<listeners>  
<listener class-name = *"org.uncommons.reportng.HTMLReporter"*/>  
<listener class-name = *"org.uncommons.reportng.JUnitXMLReporter"*/>  
</listeners>  
In your Test Case Java file, you can mention as Reporter.log(“Message”); just after where you have used Logger.log(“message”);

1. **How to implement ITestListeners in Framework**

* To add listeners in Framework, what we have to do is :
* Create a customListeners.java class (you can name it whatever you want) under Listeners Package and implements Interface i.e. ITestListener
* Right Click and go to source and click override implemented methods
* This will give a list of all unimplemented methods including **public** **boolean** isEnabled(), on TestPass, onTestFailure etc.
* Under onTestFailure, add Reporter.log(“message”); and other functions like capturing screenshots etc..
* Add Listeners tag inside TestNGXml
* <listeners>
* <listener class-name = "org.uncommons.reportng.HTMLReporter"/>
* <listener class-name = "org.uncommons.reportng.JUnitXMLReporter"/>
* <listener class-name = "com.practice.listeners.CustomListeners"/>
* </listeners>
* We also need to add 1 line of code under Listerner class under Test Failure or wherever we want to capture (System.*setProperty*("org.uncommons.reportng.escape-output", "false");

1. **How to add Screenshot with customized screenshot name and integrate with Listeners.**
2. Create a common method to get current date under Utilities package

* Public static String getscreenshotdate()
* {
* Date d = new Date(); - Make sure you download java.util package for Date Class
* String currentdate = d.toString().replace(“:”, “\_”).replace(“ “, “\_”)+”.png”; - this will replace : and space with \_
* Return currentdate;
* }

1. Create another common method to capture screenshot:

* Public static void ScreenCapture()
* {
* File src = ((TakesScreenshot)driver).getScreenshotAs(FileOutputType.File);
* FileUtils.copyFile(src, new File(“path where we want to put the file”);
* }

1. Under CustomListener.java file, under TestFailureMethod, call ScreenCapture() method to capture screenshot when test fails.
2. Here is the code:

System.*setProperty*("org.uncommons.reportng.escape-output", "false");

**try** {

TestUtilities.*takesScreenshot*(); \*\**calling method*

} **catch** (Exception e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

Reporter.*log*("Capturing Screenshots");

Reporter.*log*("<a target=\"\_blank\"

href="+TestUtilities.*screenshotName*+">Screenshot</a>");

Reporter.*log*("<br>");

Reporter.*log*("<a target=\"\_blank\" href="+TestUtilities.*screenshotName*+"><img src="+TestUtilities.*screenshotName*+"height=200 width = 200></img></a>");

ITestListener.**super**.onTestFailure(result);