Hybrid Framework

Framework- A framework is a set of rules to solve a problem in structured manner.

Types Of Framework-

- Linear Framework
- Data Driven test framework
- Library architecture framework
- Keyword Driven framework
- Hybrid framework
- Behavior driven framework

Features-

- Reusability
- Scalability
- Robust
- Manage large no of script execution
- Maintainability

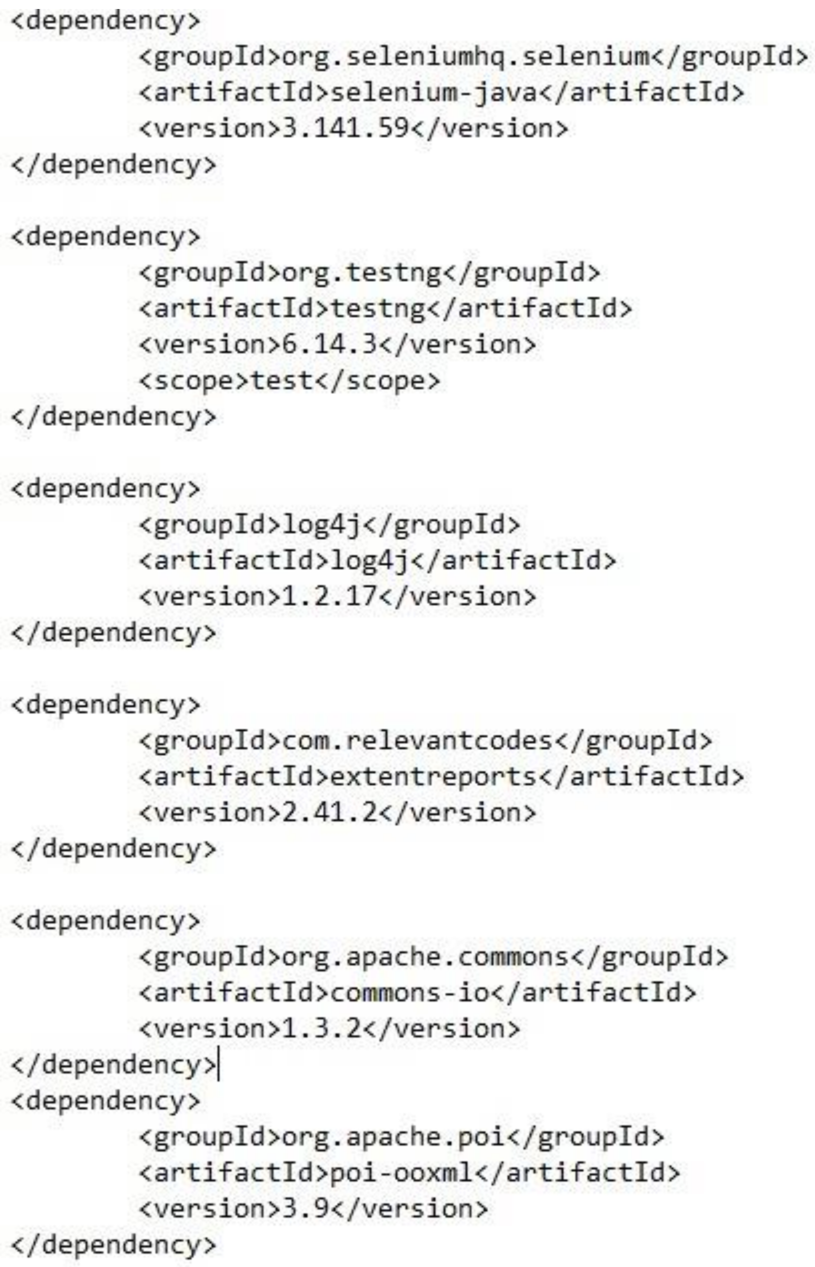
Advantages-

- Reusability of the code
- Maximum coverage
- Easy recovery
- Low-cost maintenance
- Minimal manual intervention
- Easy reporting

Components of Designing Framework-

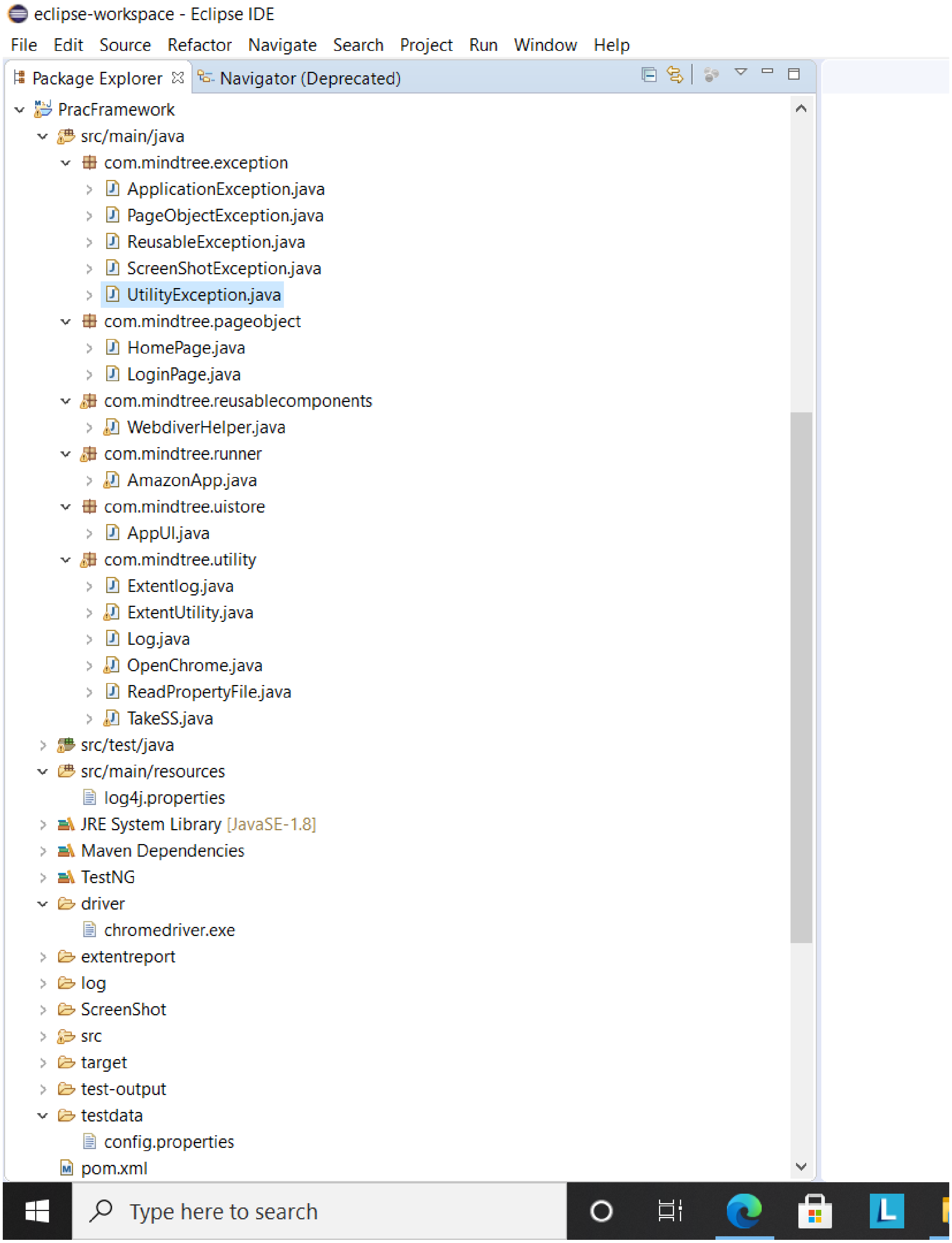
- POM (Page Object Model)
- Data: Xlsx/Xls/XML/JSON/Properties File
- Log
- Report
- Exceptions
- Reusable Components
- Runner
- Ui store
- Utilities

Add Dependency in POM.XML-



```
<dependency>
        <groupId>org.seleniumhq.selenium</groupId>
        <artifactId>selenium-java</artifactId>
        <version>3.141.59</version>
</dependency>

<dependency>
        <groupId>org.testng</groupId>
        <artifactId>testng</artifactId>
        <version>6.14.3</version>
        <scope>test</scope>
</dependency>

<dependency>
        <groupId>log4j</groupId>
        <artifactId>log4j</artifactId>
        <version>1.2.17</version>
</dependency>

<dependency>
        <groupId>com.relevantcodes</groupId>
        <artifactId>extentreports</artifactId>
        <version>2.41.2</version>
</dependency>

<dependency>
        <groupId>org.apache.commons</groupId>
        <artifactId>commons-io</artifactId>
        <version>1.3.2</version>
</dependency>
<dependency>
        <groupId>org.apache.poi</groupId>
        <artifactId>poi-ooxml</artifactId>
        <version>3.9</version>
</dependency>
```

Create Folder structure as below-



eclipse-workspace - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Navigator (Deprecated)
Type here to search

```
PracFramework
  src/main/java
    com.mindtree.exception
      ApplicationException.java
      PageObjectException.java
      ReusableException.java
      ScreenShotException.java
      UtilityException.java
    com.mindtree.pageobject
      HomePage.java
      LoginPage.java
    com.mindtree.reusablecomponents
      WebdiverHelper.java
    com.mindtree.runner
      AmazonApp.java
    com.mindtree.uistore
      AppUI.java
    com.mindtree.utility
      Extentlog.java
      ExtentUtility.java
      Log.java
      OpenChrome.java
      ReadPropertyFile.java
      TakeSS.java
  src/test/java
  src/main/resources
    log4j.properties
  JRE System Library [JavaSE-1.8]
  Maven Dependencies
  TestNG
  driver
    chromedriver.exe
  extentreport
  log
  ScreenShot
  src
  target
  test-output
  testdata
    config.properties
  pom.xml
```

Driver folder- In this driver of your browser should be present. E.g.- Chromedriver.exe for chromedriver.

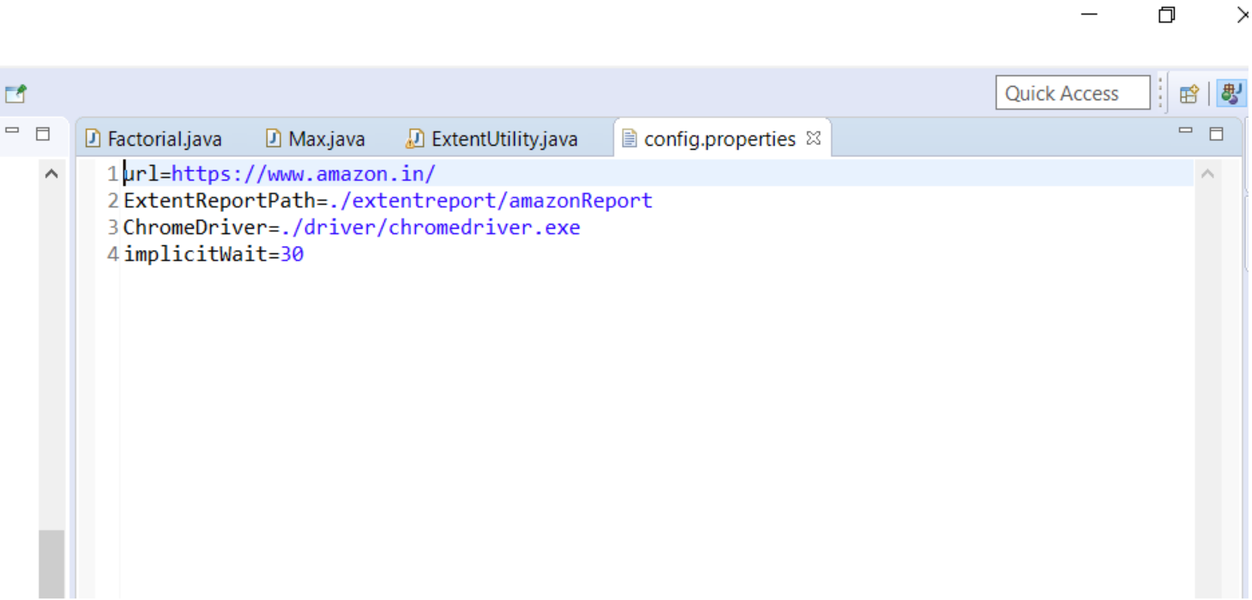
Log Folder- After execution log of your code will be present here.

ExtentReport Folder- After execution generated extentreport will be present here.

Screenshot Folder- At the time of execution if any test fails or if there's any need to take screenshot of certain page or any receipt , that screenshots will be present here.

TestData Folder-

1. Excel sheet– To read data
2. Config.properties- It is used to store data so that we can use it in multiple locations. (Paths(relative) to excel sheet, chromedriver, url, extent report, logs and implicit wait should present here).
   See below image for reference.



```
url=https://www.amazon.in/
ExtentReportPath=./extentreport/amazonReport
ChromeDriver=./driver/chromedriver.exe
implicitWait=30
```

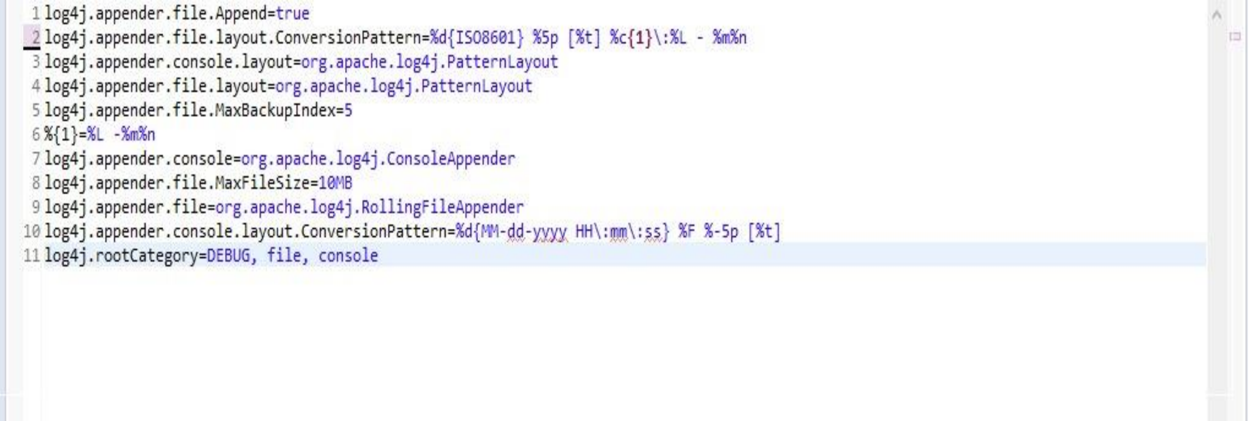
Resource folder-

1. Create a source folder and log4j.properties should present here.

Step-1- Select new package -> name it as src/main/resource.

Step-2- Inside this package create new file and name it as log4j.properties.

Step-3- Inside this file write as below.



```
log4j.appender.file.Append=true
log4j.appender.file.layout.ConversionPattern=%d{ISO8601} %5p [%t] %c{1}\:%L - %m%n
log4j.appender.console.layout=org.apache.log4j.PatternLayout
log4j.appender.file.layout=org.apache.log4j.PatternLayout
log4j.appender.file.MaxBackupIndex=5
%{1}=%L -%m%n
log4j.appender.console=org.apache.log4j.ConsoleAppender
log4j.appender.file.MaxFileSize=10MB
log4j.appender.file=org.apache.log4j.RollingFileAppender
log4j.appender.console.layout.ConversionPattern=%d{MM-dd-yyyy HH\:mm\:ss} %F %-5p [%t]
log4j.rootCategory=DEBUG, file, console
```

Utility package-Common methods that are designed to solve a particular functionality of a framework are included in this package.

1.Log- Log4j is a fast, flexible, and reliable logging framework (APIS) written in Java developed in early 1996. It is distributed under the Apache Software License. Log4J has been ported to the C, C++, C#, Perl, Python, Ruby and Eiffel Languages. It is a tool used for small to large scale Selenium Automation projects.

we use log statements rather than SOPL statements in the code to know the status of a project while it is executing.

In this class we will append timestamp to log file and write all the levels of log.

2.ExtentUtility- ExtentReports is an open-source reporting library useful for test automation. It can be easily integrated with major testing frameworks like JUnit, NUnit, TestNG, etc. These reports are HTML documents that depict results as pie charts. They also allow the generation of custom logs, snapshots, and other customized details.

A Test Status can be indicated by the following values:

PASS

FAIL

SKIP

INFO

The following built-in methods:

startTest: Executes preconditions of a test case

endTest: Executes postconditions of a test case

Log: Logs the status of each test step onto the HTML report being generated

Flush: Erases any previous data on a relevant report and creates a whole new report

Benefits of using Extent Reports-

They can be integrated with TestNG and JUnit

If required, screenshots can be captured and displayed for each step in a test

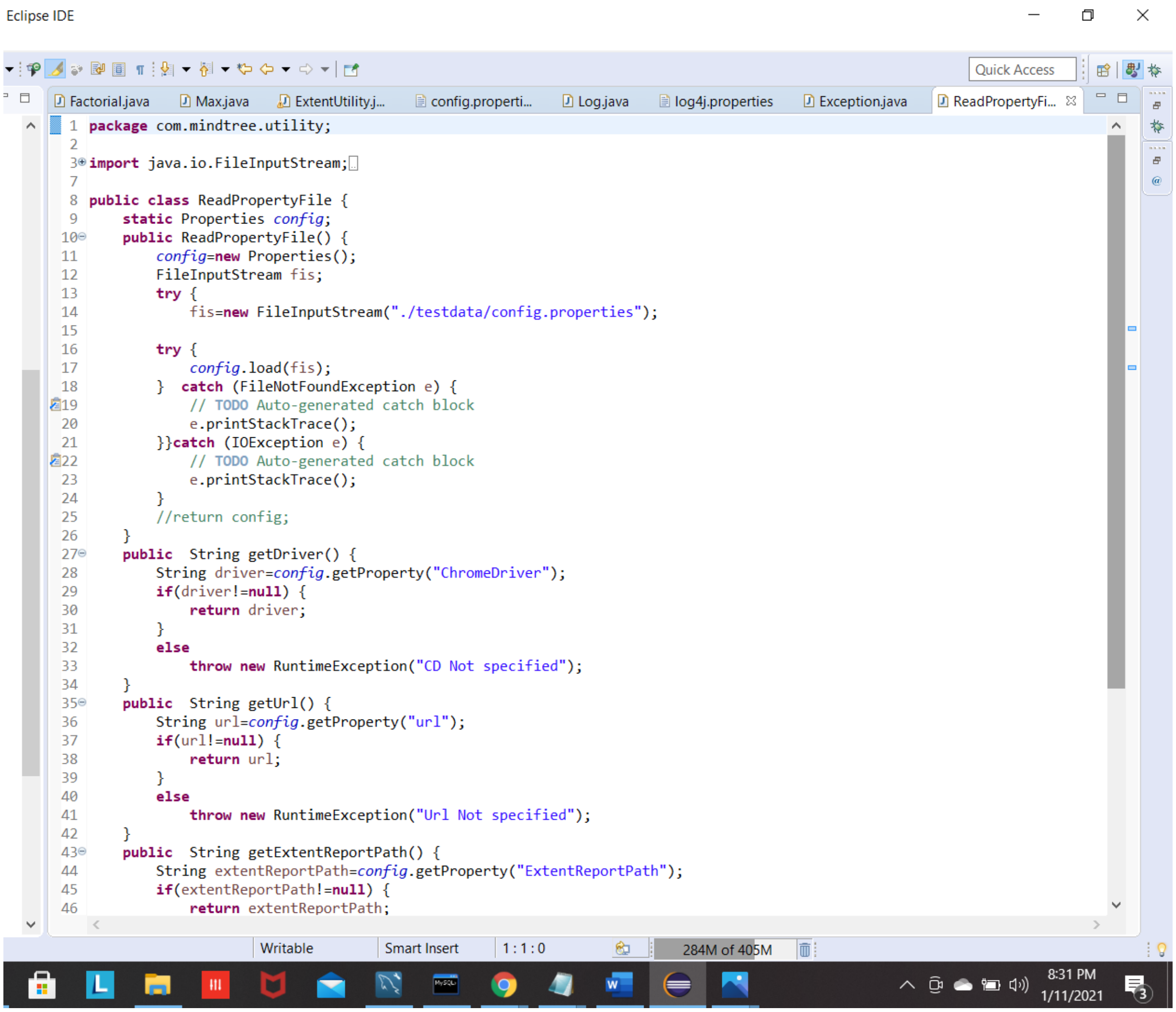
They allow testers to track multiple test case runs in a single test suite

They show the time needed for test execution

They can be customized to graphically represent each step in a test.

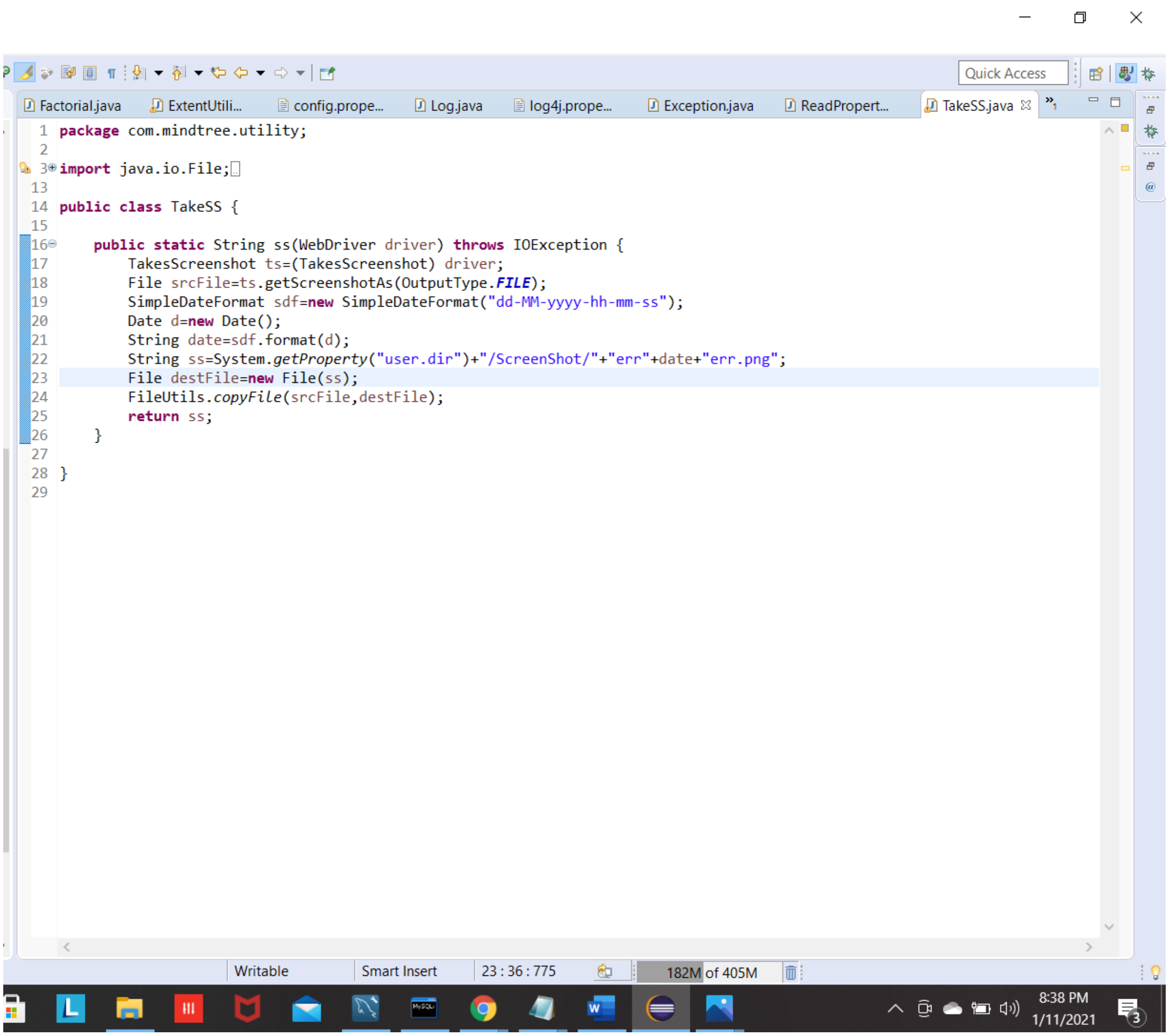
3.OpenBrowser/Connection- It is used to start a driver and to open a particular url.

4.ReadPropertyFile- It is used to get/retrieve the content of config.properties.



```java
package com.mindtree.utility;

import java.io.FileInputStream;

public class ReadPropertyFile {
    static Properties config;
    public ReadPropertyFile() {
        config=new Properties();
        FileInputStream fis;
        try {
            fis=new FileInputStream("./testdata/config.properties");

        try {
            config.load(fis);
        }  catch (FileNotFoundException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }}catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        //return config;
    }
    public  String getDriver() {
        String driver=config.getProperty("ChromeDriver");
        if(driver!=null) {
            return driver;
        }
        else
            throw new RuntimeException("CD Not specified");
    }
    public  String getUrl() {
        String url=config.getProperty("url");
        if(url!=null) {
            return url;
        }
        else
            throw new RuntimeException("Url Not specified");
    }
    public  String getExtentReportPath() {
        String extentReportPath=config.getProperty("ExtentReportPath");
        if(extentReportPath!=null) {
            return extentReportPath;
```

5.TakeSS/Screenshot- By capturing screenshots, testers can better identify what went wrong when the software acted erroneously during a test. Capture screenshots only when a test fails, since they consume a lot of memory.



```
package com.mindtree.utility;

import java.io.File;

public class TakeSS {

    public static String ss(WebDriver driver) throws IOException {
        TakesScreenshot ts=(TakesScreenshot) driver;
        File srcFile=ts.getScreenshotAs(OutputType.FILE);
        SimpleDateFormat sdf=new SimpleDateFormat("dd-MM-yyyy-hh-mm-ss");
        Date d=new Date();
        String date=sdf.format(d);
        String ss=System.getProperty("user.dir")+"/ScreenShot/"+"err"+date+"err.png";
        File destFile=new File(ss);
        FileUtils.copyFile(srcFile,destFile);
        return ss;
    }

}
```

UiStore Package-

It holds all the webelements(locators) as per page object mapping.

Exception Package-

Exception is an abnormal condition which disrupts normal flow of program. These abnormal conditions are taken care of in this class.

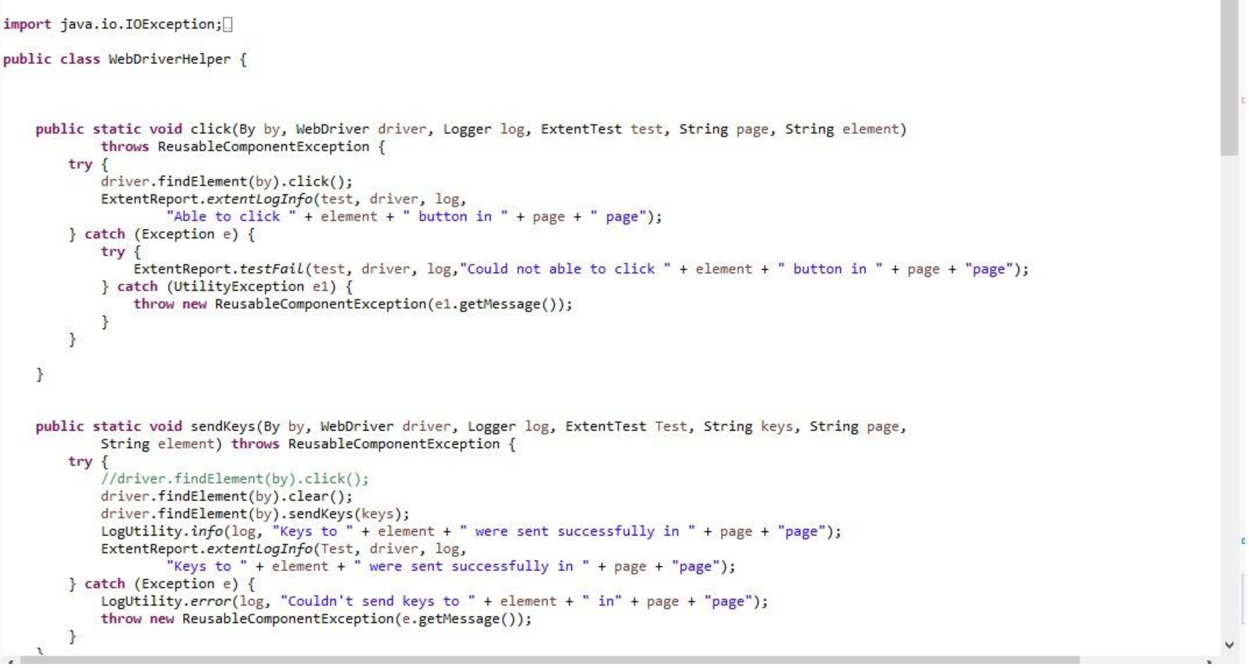
Runner Package-

It holds the class from where the execution is invoked.

1. TestNg annotations should be present in proper order.
2. All the methods of pageobject are invoked from runner.
3. All the exceptions are handled in runner.

Reusable package-

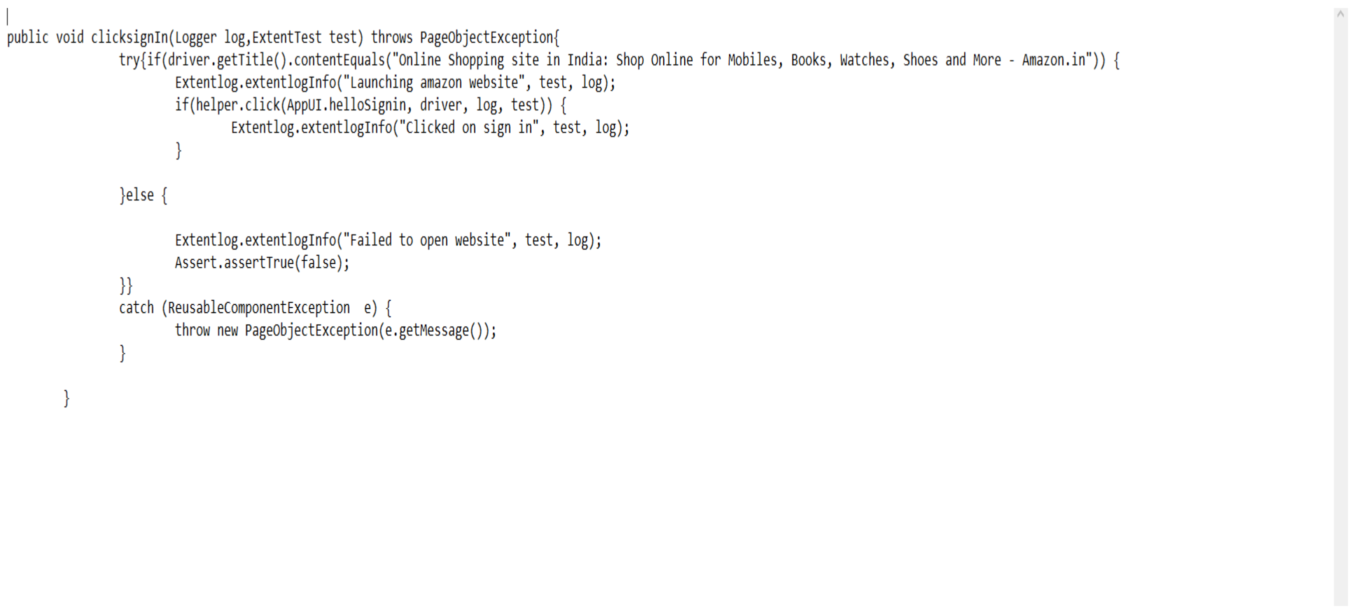
It holds all the reusable common methods for application or framework.



```
import java.io.IOException;

public class WebDriverHelper {


    public static void click(By by, WebDriver driver, Logger log, ExtentTest test, String page, String element)
            throws ReusableComponentException {
        try {
            driver.findElement(by).click();
            ExtentReport.extentLogInfo(test, driver, log,
                    "Able to click " + element + " button in " + page + " page");
        } catch (Exception e) {
            try {
                ExtentReport.testFail(test, driver, log,"Could not able to click " + element + " button in " + page + "page");
            } catch (UtilityException e1) {
                throw new ReusableComponentException(e1.getMessage());
            }
        }

    }


    public static void sendKeys(By by, WebDriver driver, Logger log, ExtentTest Test, String keys, String page,
            String element) throws ReusableComponentException {
        try {
            //driver.findElement(by).click();
            driver.findElement(by).clear();
            driver.findElement(by).sendKeys(keys);
            LogUtility.info(log, "Keys to " + element + " were sent successfully in " + page + "page");
            ExtentReport.extentLogInfo(Test, driver, log,
                    "Keys to " + element + " were sent successfully in " + page + "page");
        } catch (Exception e) {
            LogUtility.error(log, "Couldn't send keys to " + element + " in" + page + "page");
            throw new ReusableComponentException(e.getMessage());
        }
    }
```

PageObject package-

It holds all keywords page-wise.



```
public void clicksignIn(Logger log,ExtentTest test) throws PageObjectException{
        try{if(driver.getTitle().contentEquals("Online Shopping site in India: Shop Online for Mobiles, Books, Watches, Shoes and More - Amazon.in")) {
            Extentlog.extentlogInfo("Launching amazon website", test, log);
            if(helper.click(AppUI.helloSignin, driver, log, test)) {
                Extentlog.extentlogInfo("Clicked on sign in", test, log);
            }

        }else {

            Extentlog.extentlogInfo("Failed to open website", test, log);
            Assert.assertTrue(false);
        }}
        catch (ReusableComponentException  e) {
            throw new PageObjectException(e.getMessage());
        }

    }
```