1. Download Maven Dependency 4.00 version for extent report .Copy dependency files and paste into pom.xml

Extent report used for advance report generation

In extent report generation there are 3 classes

1. Extent Html Reporter: How it looks, what kind of things you want to do, what types of resources you want to use and maintained etc

2. Extent Reports: helps for look and feel

3. Extent Test: helps to generate logs

Create instance of all extent reporter

If you could not flush the report it is not possible to generate report.

Supermost class of exception is getThrowable

Program File = **demoExtentReports.java**

package com.demoExtentReports;

import java.io.File;

import java.io.IOException;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.TakesScreenshot;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.io.FileHandler;

import org.testng.Assert;

import org.testng.ITestResult;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.AfterTest;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.BeforeTest;

import org.testng.annotations.Test;

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.Theme;

public class OrangeHrmTest {

// create instance of three extent reports at class level

public ExtentHtmlReporter htmlReporter;

public ExtentReports report;

public ExtentTest logger;

public static WebDriver driver;

public ExtentTest test;

@BeforeTest

public void setupExtent() {

// where you want to use extent reports specify the path//location of file

File fs = new File("./Reports/orange\_hrm.html");

// instantiate html reporter class

// by using object creation

htmlReporter = new ExtentHtmlReporter(fs);

// set the title of documentation report

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

htmlReporter.config().setDocumentTitle("Functional Report");

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

// Attach Html report

report = new ExtentReports();

report.attachReporter(htmlReporter);

report.setSystemInfo("Hostname", "LocalHost");

// set system information for which operating system you are used.

report.setSystemInfo("OS", "Windows");

report.setSystemInfo("Tester name", "Amruta");

report.setSystemInfo("Browser", "Chrome");

}

// If you could not flush the report it is not possible to generate report.

@AfterTest

public void endReport() {

report.flush();

}

@BeforeMethod

public void setup() {

// set the system variable path

System.setProperty("webdriver.chrome.driver", "./Drivers/chromedriver.exe");

// create the instance of chrome driver

driver = new ChromeDriver();

// navigate to the application

driver.get("https://opensource-demo.orangehrmlive.com/");

// manage the window

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

//driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

}

@Test

public void verifyTitleTest() {

test = report.createTest("verifyTitleTest");

String actualTitle = driver.getTitle();

String ExpectedTitle = "OrangeHRM";

Assert.assertEquals(actualTitle, ExpectedTitle);

}

@Test

public void verifyLogoTest() {

test = report.createTest("verifyTitleTest");

boolean status = driver.findElement(By.xpath("//div[@id ='divLogo']/img")).isDisplayed();

Assert.assertTrue(status);

}

@AfterMethod

public void tearDown(ITestResult result) throws IOException {

if (result.getStatus() == ITestResult.FAILURE) {

test.log(Status.FAIL, "Test Case failed" + result.getName());

test.log(Status.FAIL, "Test Case failed" + result.getThrowable());

String screenshotpath = captureScreenshot(result.getName());

test.addScreenCaptureFromPath(screenshotpath);

} else if (result.getStatus() == ITestResult.SUCCESS) {

test.log(Status.PASS, "Test Case Passed" + result.getName());

} else if (result.getStatus() == ITestResult.SKIP) {

test.log(Status.SKIP, "Test Case Passed" + result.getName());

}

driver.quit();

}

// create method for capture screenshot

public static String captureScreenshot(String screenshotname) throws IOException {

String dateFormat = new SimpleDateFormat("yyyyMMddhhmmss").format(new Date());

// take screenshot interface

TakesScreenshot ts = (TakesScreenshot) driver;

// capture the screenshot ..output is in form of file, it will return you file

// object in buffer memory

File srcFile = ts.getScreenshotAs(OutputType.FILE);

String screenshotPath = "./Screenshots/\"+screenshotname+dateFormat+\".png";

File destFile = new File("screenshotPath");

FileHandler.copy(srcFile, destFile);

return screenshotPath;

}

}

Output

