package com.ixigo;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.ie.InternetExplorerDriver;

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

import org.testng.annotations.AfterClass;

import org.testng.annotations.BeforeClass;

import org.testng.asserts.Assertion;

import com.ixigo.base.Base;

import com.ixigo.base.Page;

import com.ixigo.files.ConfigFileReader;

import io.github.bonigarcia.wdm.WebDriverManager;

/\*\*

\* @author Subhajit

\*

\*/

public class BaseTest {

private static WebDriver driver;

private static WebDriverWait wait;

private static Page base;

public static Assertion hardAssert;

public static void initialization() {

openBrowser(ConfigFileReader.getConfigPropObject().getProperty("browser"));

// Navigate to a URL

try {

driver.get(ConfigFileReader.getConfigPropObject().getProperty("url"));

} catch (Exception e) {

e.printStackTrace();

}

try {

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

driver.manage().deleteAllCookies();

wait = new WebDriverWait(driver, 50);

base = new Base(driver, wait);

hardAssert = new Assertion();

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

} catch (Exception e) {

e.printStackTrace();

}

}

public static void openBrowser(String browserType) {

try {

switch (browserType) {

case "firefox":

WebDriverManager.firefoxdriver().clearPreferences();

WebDriverManager.firefoxdriver().setup();

driver = new FirefoxDriver();

break;

case "chrome":

WebDriverManager.chromedriver().setup();

System.setProperty("webdriver.chrome.silentOutput", "true");

driver = new ChromeDriver();

break;

case "IE":

WebDriverManager.iedriver().setup();

driver = new InternetExplorerDriver();

break;

default:

System.out

.println("browser : " + browserType + " is invalid, Launching Firefox as browser of choice..");

WebDriverManager.chromedriver().setup();

driver = new ChromeDriver();

}

} catch (Exception e) {

e.printStackTrace();

}

}

@BeforeClass

public void setup() {

initialization();

}

/\*\*

\* Getter and Setter

\*

\*/

public static Page getBase() {

return base;

}

public static void setBase(Page base) {

BaseTest.base = base;

}

@AfterClass

public void closeBrowser() {

driver.close();

}

}