MTR105 Assinment

Step 1: Set Up

Install Software

- **Install Java**: Make sure you have Java installed. You can check by running java -version in the terminal.
- **Install Node.js**: Appium is built on Node.js, so you need to install it.
- **Install Appium**: Run the following command to install Appium globally using npm:

  bash
  Copy code
  npm install -g appium

Install Android SDK and ADB

- **Install Android Studio**
- **Set up ADB**

  adb version

- **Add Android SDK tools** to the system path.

  Set Up Appium Client (Java in this case)

- **Install Appium Java Client**: Add the Appium Java Client to your project's dependencies (Maven or Gradle).
- **Create a Maven project** (for example, with the following dependencies in pom.xml):

  xml
  Copy code
  <dependency>
  <groupId>io.appium</groupId>
  <artifactId>java-client</artifactId>
  <version>8.0.0</version>
  </dependency>

- **Create the AppiumDriver** for Android.

Start Appium Server

- Open a terminal and start Appium server with:

  appium

Step 2: Validate the Apps

Verify App Name and Version

The app's name is **API Demos APK**. You can verify the app version by inspecting the app details in the app store or using ADB commands.

Install the Application on Emulator

Use the following ADB command to install the **API Demos APK** on Android emulator:

download the APK from a trusted source or from the official **API Demos** website.

Launch Application

Once installed, launch the application using Appium:

```
DesiredCapabilities capabilities = new DesiredCapabilities();
capabilities.setCapability("platformName", "Android");
capabilities.setCapability("platformVersion", "11");  // Adjust based on your emulator
capabilities.setCapability("deviceName", "emulator-5554");  // Adjust based on your emulator
capabilities.setCapability("appPackage", "com.mxtech.videoplayer.ad");  // API Demos package name
capabilities.setCapability("appActivity", "com.mxtech.videoplayer.ad.ActivityIntro");  // Launch Activity

// Initialize Appium Driver
AndroidDriver<MobileElement> driver = new AndroidDriver<>(new URL("http://127.0.0.1:4723/wd/hub"), capabilities);
```

Step 3: Test (Tap, Scroll, Click, Drag and Drop, Send Keys) Actions

For each of the actions (touch, tap, scroll, click, drag and drop, send keys), we will automate the interactions with the **API Demos** app.

Touch

You can use Appium's touch actions to simulate touch events:

```
TouchAction touchAction = new TouchAction(driver);
touchAction.tap(PointOption.point(300, 400)).perform();
```

Tap

A tap is a simple click on a specific element:

```
MobileElement playButton = driver.findElement(By.id("com.mxtech..ad:id/play_button"));
playButton.click();
```

Scroll

Scroll within the app:

```
MobileElement element = driver.findElement
new TouchAction(driver)
    .longPress(PointOption.point(element.getLocation()))
    .moveTo(PointOption.point(0, 100))
    .release().perform();
```

Click

Click on specific elements (e.g., buttons, icons):

```
MobileElement settingsButton = driver.findElement(By.id("com..ad:id/settings_button"));
settingsButton.click();
```

Drag and Drop

Simulate dragging and dropping:

```
MobileElement sourceElement = driver.findElement(By.id("source_element_id"));
MobileElement targetElement = driver.findElement(By.id("target_element_id"));
new TouchAction(driver)
    .longPress(ElementOption.element(sourceElement))
    .moveTo(ElementOption.element(targetElement))
    .release()
    .perform();
```

Send Keys

Send keyboard input (e.g., search text):

```
MobileElement searchBox = driver.findElement(By.id("com.mxtech.ad:id/search_box"));
searchBox.sendKeys("Movie Name");
```

```
import io.appium.java_client.MobileElement;
import io.appium.java_client.android.AndroidDriver;
import io.appium.java_client.TouchAction;
import io.appium.java_client.touch.offset.PointOption;
import org.openqa.selenium.By;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.WebElement;

import java.net.URL;

public class MXPlayerTest {
    public static void main(String[] args) throws Exception {
        DesiredCapabilities capabilities = new DesiredCapabilities();
        capabilities.setCapability("platformName", "Android");
        capabilities.setCapability("platformVersion", "11");
        capabilities.setCapability("deviceName", "emulator-5554");
        capabilities.setCapability("appPackage", "com.mxtech.videoplayer.ad");
        capabilities.setCapability("appActivity", "com.mxtech..ad.ActivityIntro");

        AndroidDriver<MobileElement> driver = new AndroidDriver<>(new URL("http://127.0.0.1:4723/wd/hub"), capabilities);

        // Tap on Play Button
        MobileElement playButton = driver.findElement(By.id("com.mxtech..ad:id/play_button"));
        playButton.click();

        // Scroll Action
        MobileElement scrollableView = driver.findElement(By.id("com.mxtech..ad:id/scrollable_view"));
        new TouchAction(driver)
            .longPress(PointOption.point(scrollableView.getLocation()))
            .moveTo(PointOption.point(0, 100))
            .release()
            .perform();

        // Send text input
        MobileElement searchBox = driver.findElement(By.id("com.mxtech..ad:id/search_box"));
        searchBox.sendKeys("Movie Name");

        // Close the driver session
        driver.quit();
    }
}
```

Step 4: Execute the Automation Script

Once you've set up the environment and the script, you can execute the test using a test runner or directly via your IDE