Worldline App Testing Task

Dhineshwaran C
WDGET2024090

Calculator Mobile App Testing Using Appium Tool

GitHub Link:

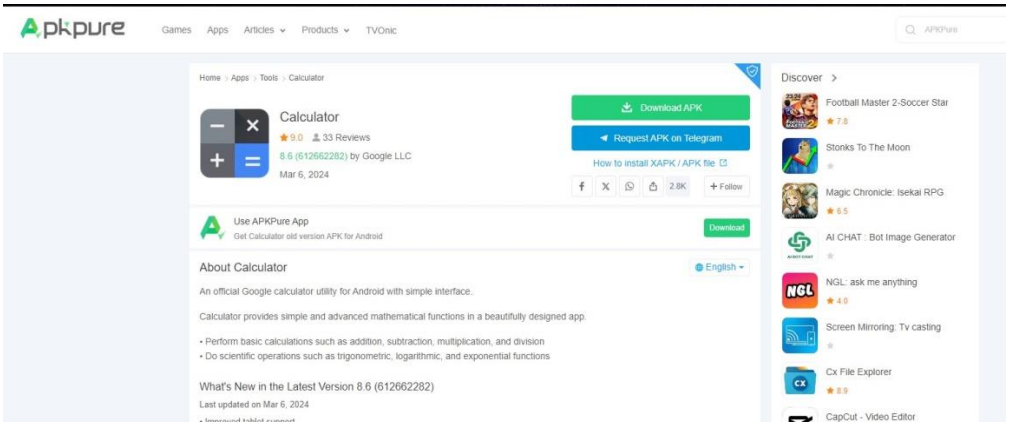
https://github.com/Dhineshwaran-C/CalculatorAppiumTesting.git

Introduction:

This documentation outlines a Appium framework for automating tests on the Android Calculator application. It employs TestNG for organizing test cases and data-driven testing. The script interacts with the calculator app by simulating user inputs and verifies the calculated results against expected values.

Configuring Calculator App in the Mobile Emulator:

To install Calculator app in the Mobile Emulator first we need to download Calculator apk from Apkpure website https://apkpure.com/calculator/com.google.android.calculator/download?utm_content=1008.



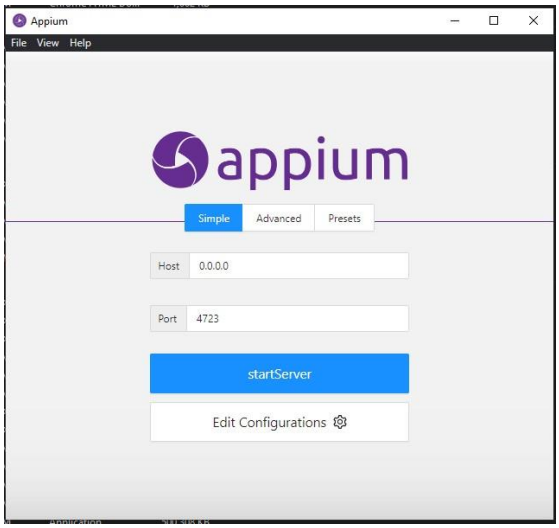
After downloading the APK file, open Android Studio and launch the emulator.

To install calculator APK in the emulator, Open the command prompt and type “adb install G:\worldline\Calculator.apk”. This command will install the Calculator APK located at the specified path onto the emulator.

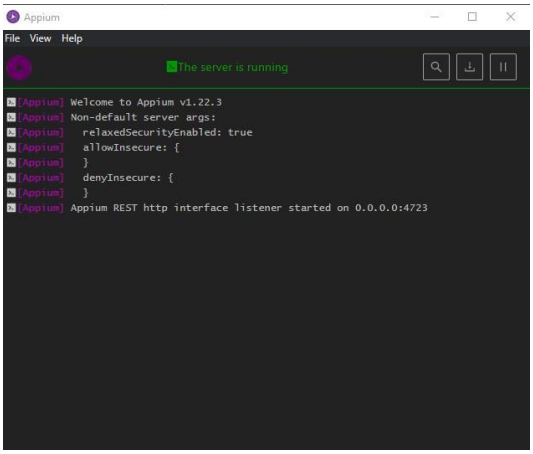
```
C:\Users\Dhineshwaran C>adb install G:\worldline\Calculator.apk
Performing Streamed Install
Success
```

Configuring Mobile Emulator in the Appium Inspector:

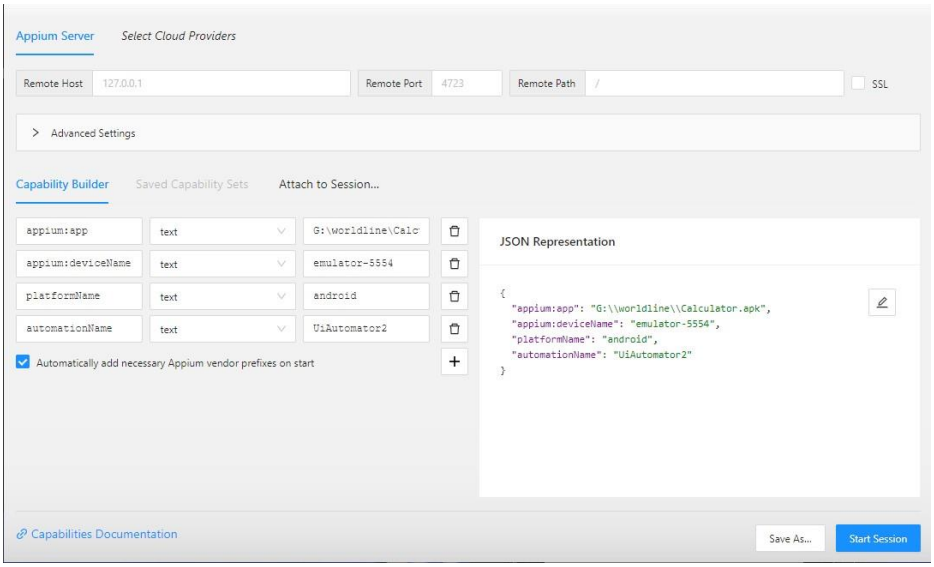
Open the Appim Server GUI and click on “startServer” to initiate the server.



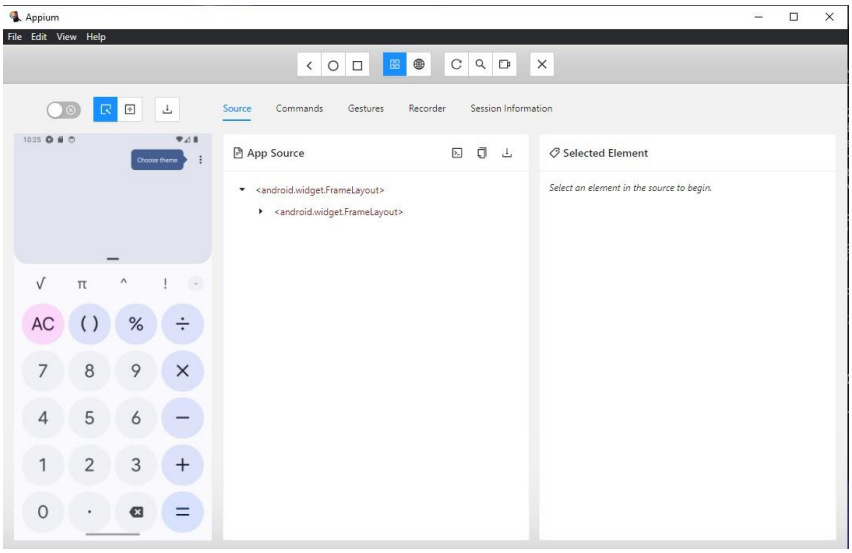
After starting the server, it will appear like this



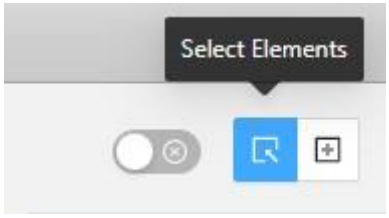
Open the Appium Inspector and provide the mobile credentials and app credentials in the Capability Builder.



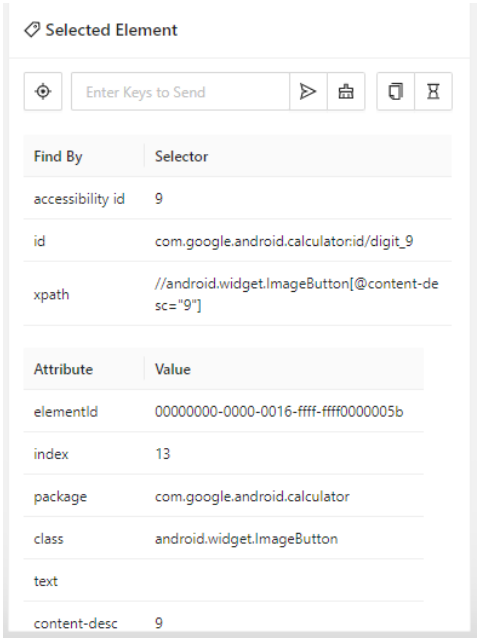
Click on "Start Session" to connect the Calculator app from the mobile emulator to the Appium Inspector.



To obtain the element ID of the Calculator buttons, click on "Select Elements" at the top of the Appium Inspector and then click on the button for which we need the ID.



After clicking the element, we should be able to see the ID of the element on the right side of the Appium Inspector.



By using the ID, we can identify the element and make the necessary changes.

Code Explanation:

Create a Java Maven project and include the dependencies for Appium, Selenium and TestNG in the pom.xml file.

```
<!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
<dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-java</artifactId>
    <version>3.141.5</version>
</dependency>

<!-- https://mvnrepository.com/artifact/io.appium/java-client -->
<dependency>
    <groupId>io.appium</groupId>
    <artifactId>java-client</artifactId>
    <version>7.0.0</version>
</dependency>

<!-- https://mvnrepository.com/artifact/org.testng/testng -->
<dependency>
    <groupId>org.testng</groupId>
    <artifactId>testng</artifactId>
    <version>7.9.0</version>
    <scope>test</scope>
</dependency>
```

Create a class called AppTest in the path src/test/java

Inside the class create the functions called openCalculator, CalculatorTesting, clickElement, calData.

openCalculator() Method:

The openCalculator() method initializes the Appium driver and launches the Android Calculator app. It begins by setting up desired capabilities, including device information and app details. These capabilities specify the configuration for the test session, ensuring compatibility with the targeted device and application. Subsequently, an instance of the AppiumDriver class, specialized for handling mobile elements (MobileElement), is created using the specified URL and capabilities. This establishes a connection to the Appium server and launches the Calculator app on the designated device. Error handling is implemented to catch any potential MalformedURLExceptions that may

occur during the instantiation of the URL object, ensuring robustness of the method.

```java
@Test(priority=1)
Run | Debug
public void openCalculator() throws MalformedURLException {
    DesiredCapabilities cap = new DesiredCapabilities();

    cap.setCapability("deviceName", "sdk_gphone64_x86_64");
    cap.setCapability("udid","emulator-5554");
    cap.setCapability("platformName","Android");
    cap.setCapability("platformVersion","14");
    cap.setCapability("automationName", "UiAutomator2");

    cap.setCapability("appPackage","com.google.android.calculator");
    cap.setCapability("appActivity","com.android.calculator2.Calculator");


    @SuppressWarnings("deprecation")
    URL url = new URL("http://127.0.0.1:4723/wd/hub");
    driver = new AppiumDriver<MobileElement>(url, cap);
    System.out.println("Application Started... ");
}
```

CalculatorTesting() Method:

The CalculatorTesting() method serves as the core function for testing the calculator app. It accepts input parameters representing a mathematical expression (cal) and its expected result (expectedResult) from the data provider. Within this method, the input expression is processed character by character using a for loop. For each character, the clickElement() method is invoked to simulate the corresponding button click on the calculator app, effectively replicating user interaction. After performing the calculation, the actual result is obtained from the calculator app by locating the result element using its ID (result_final). TestNG's assertion mechanism is then utilized to compare the actual result with the expected result, ensuring the correctness of the app's calculations.

```java
@Test(priority=2, dataProvider="testdata")
Run | Debug
public void CalculatorTesting(String cal,String expectedResult) {

    // s = Square root
    // p = pi
    // d = delete
    // c = clear
    for(int i=0;i<cal.length();i++) {
        clickElement(cal.charAt(i));
    }


    MobileElement result = driver.findElement(By.id("com.google.android.calculator:id/result_final"));


    String actualResult = result.getText();

    System.out.println("Ans is : "+ result.getText());

    Assert.assertEquals(expectedResult,actualResult);

}
```

clickElement() Method:

The clickElement() method simulates button clicks on the calculator app based on the input character provided. It employs a switch-case statement to determine the appropriate action for each character. Depending on the character, the method utilizes Appium's findElement(By.id()) method to locate and interact with the corresponding button on the calculator interface.

```
public void clickElement(char value) {
    switch(value) {
    case '0':
        driver.findElement(By.id("com.google.android.calculator:id/digit_0")).click();
        break;

    case '1':
        driver.findElement(By.id("com.google.android.calculator:id/digit_1")).click();
        break;

    case '2':
        driver.findElement(By.id("com.google.android.calculator:id/digit_2")).click();
        break;

    case '3':
        driver.findElement(By.id("com.google.android.calculator:id/digit_3")).click();
        break;

    case '4':
        driver.findElement(By.id("com.google.android.calculator:id/digit_4")).click();
        break;

    case '5':
        driver.findElement(By.id("com.google.android.calculator:id/digit_5")).click();
        break;

    case '6':
        driver.findElement(By.id("com.google.android.calculator:id/digit_6")).click();
        break;

    case '7':
        driver.findElement(By.id("com.google.android.calculator:id/digit_7")).click();
        break;

    case '8':
        driver.findElement(By.id("com.google.android.calculator:id/digit_8")).click();
        break;

    case '9':
        driver.findElement(By.id("com.google.android.calculator:id/digit_9")).click();
        break;

    case '.':
        driver.findElement(By.id("com.google.android.calculator:id/dec_point")).click();
        break;

    case 'd':
        driver.findElement(By.id("com.google.android.calculator:id/del")).click();
        break;

    case '=':
        driver.findElement(By.id("com.google.android.calculator:id/eq")).click();
        break;

    case '+':
        driver.findElement(By.id("com.google.android.calculator:id/op_add")).click();
        break;
```

```
case '-':
    driver.findElement(By.id("com.google.android.calculator:id/op_sub")).click();
    break;

case '*':
    driver.findElement(By.id("com.google.android.calculator:id/op_mul")).click();
    break;

case '/':
    driver.findElement(By.id("com.google.android.calculator:id/op_div")).click();
    break;

case '%':
    driver.findElement(By.id("com.google.android.calculator:id/op_pct")).click();
    break;

case '(':
    driver.findElement(By.id("com.google.android.calculator:id/parens")).click();
    break;

case ')':
    driver.findElement(By.id("com.google.android.calculator:id/parens")).click();
    break;

case 'c':
    driver.findElement(By.id("com.google.android.calculator:id/clr")).click();
    break;

case 's':
    driver.findElement(By.id("com.google.android.calculator:id/op_sqrt")).click();
    break;

case 'p':
    driver.findElement(By.id("com.google.android.calculator:id/const_pi")).click();
    break;

case '^':
    driver.findElement(By.id("com.google.android.calculator:id/op_pow")).click();
    break;

case '!':
    driver.findElement(By.id("com.google.android.calculator:id/op_fact")).click();
    break;
}
```

calData() Method:

The calData() method serves as a data provider for supplying test data to the CalculatorTesting() method. It returns a 2D array where each row represents a test case, consisting of an input mathematical expression and its corresponding expected result. This data-driven approach enables the systematic execution of multiple test scenarios with varying input values.

```
@DataProvider(name = "testdata")
public Object[][] calData(){
    return new Object[][] {
        {"9+(2*3+6)/2=","15"},
        {"s2+s5=","3.650281539872"},
        {"72.65/65.23=","1.113751341407"},
        {"s2+s5d8=","4.242640687119"},
        {"(3*p)+18=","27.42477796076"}
    };
}
```

Test Output:

```
[RemoteTestNG] detected TestNG version 7.9.0
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Mar 27, 2024 12:56:17 PM io.appium.java_client.remote.AppiumCommandExecutor$1 lambda$0
INFO: Detected dialect: W3C
Application Started...
Ans is : 15
Ans is : 3.650281539872
Ans is : 1.113751341407
Ans is : 4.242640687119
Ans is : 27.42477796076
PASSED: AppiumCalTest.AppiumCalTest.AppTest.CalculatorTesting("s2+s5=", "3.650281539872")
PASSED: AppiumCalTest.AppiumCalTest.AppTest.CalculatorTesting("s2+s5d8=", "4.242640687119")
PASSED: AppiumCalTest.AppiumCalTest.AppTest.CalculatorTesting("72.65/65.23=", "1.113751341407")
PASSED: AppiumCalTest.AppiumCalTest.AppTest.CalculatorTesting("9+(2*3+6)/2=", "15")
PASSED: AppiumCalTest.AppiumCalTest.AppTest.CalculatorTesting("(3*p)+18=", "27.42477796076")
PASSED: AppiumCalTest.AppiumCalTest.AppTest.openCalculator

===============================================
    Default test
    Tests run: 2, Failures: 0, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 6, Passes: 6, Failures: 0, Skips: 0
===============================================
```