Milestone 2: Run the tools

After integrating both the Xamarin.UITest tool and Exerciser Monkey tool, I will run the unit tests from the first tool, and then run the second tool for a set amount of time and/or runs. The results of the tests will be tracked.

I decided to run these tools on two projects; one being my own mobile application with lots of buttons and user input, and the other being a floppy bird clone.

Tool #1: Monkey Exerciser

I ran the following command for the Monkey Exerciser tool on my mobile application:

```
adb shell monkey -p com.companyname.petinsights_all -v 1000 --pct-
syskeys 0 --throttle 500 --ignore-crashes > petinsightsapp_log.txt
```

`-v 1000` signifies that I want to send 1000 commands (tap, drag, hold, etc.) to my application.

`--pct-syskeys 0` signifies that I want 0% of system key events, such as clicking the Home, Back, Start Call, End Call, or Volume controls.

`-- throttle 500` signifies that I want there to be a 500 millisecond gap between each command.

`--ignore-crashes` signifies that if a touch sequence results in a crash, the Monkey Exerciser tool continues to run.

Finally, the results of the run are stored in `petinsightsapp_log.txt`. This file is included in the repository, and a sample is included on the next page.

Thoughts:

Drawbacks

Even though the –pct-syskeys 0 was set to 0, some system key commands were still run. Also, the results in the log file were very difficult to interpret.

Advantages

This tool appears very useful in stress-testing an application. Although difficult to interpret, the log file did state a few Permission Denied errors and helped me look into that issue in my application.

```
:Sending Touch (ACTION_UP): 0:(717.4599,1916.0681)
:Sending Touch (ACTION_DOWN): 0:(445.0,1172.0)
:Sending Touch (ACTION_UP): 0:(447.83023,1163.1597)
:Sending Touch (ACTION_DOWN): 0:(881.0,419.0)
    //[calendar_time:2021-03-24 21:28:32.714  system_uptime:478453876]
    // Sending event #300
:Sending Touch (ACTION_UP): 0:(856.4057,293.59918)
:Sending Touch (ACTION_DOWN): 0:(283.0,492.0)
:Sending Touch (ACTION_UP): 0:(274.662,483.917)
:Sending Trackball (ACTION_MOVE): 0:(3.0,-5.0)
:Sending Trackball (ACTION_MOVE): 0:(0.0,2.0)
:Sending Touch (ACTION_DOWN): 0:(384.0,1436.0)
:Sending Touch (ACTION_UP): 0:(386.80634,1436.0634)
:Sending Trackball (ACTION_MOVE): 0:(-4.0,1.0)
:Sending Trackball (ACTION_UP): 0:(0.0,0.0)
:Sending Trackball (ACTION_MOVE): 0:(1.0,2.0)
:Sending Trackball (ACTION_MOVE): 0:(-2.0,4.0)
:Sending Trackball (ACTION_MOVE): 0:(1.0,-4.0)
:Sending Touch (ACTION_DOWN): 0:(243.0,1282.0)
:Sending Touch (ACTION_UP): 0:(378.479,1374.6163)
:Sending Flip keyboardOpen=true
Got IOException performing flipjava.io.FileNotFoundException: /dev/input/event0: open failed: EACCES (Permission denied)
    // Injection Failed
:Sending Trackball (ACTION_MOVE): 0:(-2.0,-2.0)
    //[calendar_time:2021-03-24 21:28:32.910  system_uptime:478454071]
    // Sending event #400
:Sending Touch (ACTION_DOWN): 0:(154.0,706.0)
:Sending Touch (ACTION_UP): 0:(49.338985,699.31726)
:Sending Touch (ACTION_DOWN): 0:(636.0,602.0)
:Sending Touch (ACTION_UP): 0:(629.96844,589.7561)
:Sending Touch (ACTION_DOWN): 0:(870.0,1715.0)
:Sending Touch (ACTION_UP): 0:(868.57544,1718.9716)
:Sending Touch (ACTION_DOWN): 0:(707.0,69.0)
:Sending Touch (ACTION_UP): 0:(681.95447,69.0223)
:Sending Touch (ACTION_DOWN): 0:(398.0,1845.0)
:Sending Touch (ACTION_UP): 0:(414.20798,1851.4485)
:Sending Touch (ACTION_DOWN): 0:(356.0,690.0)
:Sending Touch (ACTION_UP): 0:(355.65338,694.9497)
:Sending Touch (ACTION_DOWN): 0:(321.0,1140.0)
:Sending Touch (ACTION_UP): 0:(323.33472,1140.0557)
```

Tool #2: Xamarin.UITests

I intended to base my Xamarin.UITests test cases on the results form Monkey Exerciser, but after running the exerciser tool I did not find this to be a practical approach anymore. At the very least, the buttons pressed in the monkey tool could be replicated in the manually-written tests.

I found it to be incredibly time-consuming to write individual test cases for my Xamarin application, so I have included just a sample test below.

Thoughts:

Drawbacks

Although more precise than the Monkey Exerciser tool, the effort in building test cases is extraordinary. It would be most effective to add the test cases as you are building your application, since stepping back into the multiple layers of the application to add the proper labels to the UI components takes more time. Due to this, I found that Xamarin.UITests to not be a good addition for a project especially if the project is already near completion.

Advantages

As mobile apps now must be compatible with all sorts of devices (Samsung Galaxy, Google Pixel, iPhone, etc.), it becomes unreasonable to manually test your app on every device that exists. However, Xamarin.UITests is very beneficial in this scenario as it works with an external tool called AppCenter [1] which allows you to submit your project and test cases, and runs the test cases simultaneously on the selected devices. AppCenter also keeps screenshots of the test results for manual analysis.

[1] https://appcenter.ms/

```
using System;
using System.IO;
using System.Linq;
using NUnit.Framework;
using Xamarin.UITest;
using Xamarin.UITest.Queries;

namespace UITestDemo.UITest
{
    [TestFixture(Platform.Android)]
    [TestFixture(Platform.iOS)]
    public class Tests
    {
        IApp app;
        Platform platform;

        public Tests(Platform platform)
        {
            this.platform = platform;
        }

        [SetUp]
        public void BeforeEachTest()
        {
            app = AppInitializer.StartApp(platform);
        }

        [Test]
        public void AppLaunches()
        {
            app.Screenshot("First screen.");
        }

        [Test]
        public void ClearTextDemo()
        {
            app.Tap(x => x.Marked("Add"));
            app.Tap(x => x.Text("Item name"));

           app.Screenshot("Before calling ClearText");
           app.ClearText();
           app.EnterText("The test worked!");
           app.Screenshot("Text cleared & replaced");
           app.Back();
        }
    }
}
```