


SQA

AUTOMATION

TESTING

# Automating Mobile Game Testing

You don't really hear a lot about automating mobile games. The general practice for most game companies is to play test games rather than automate them. Difficult sure but not impossible. Let's find out in this blog.

Sumana Ghimire  
Nov 10, 2023 · 4 min read

## Introduction

Automating mobile game testing is critical as they evolve and new features are added regularly. It enables us to detect faults early in the development process and maintain a high level of gameplay for our users. We should consider both functional and non-functional automated testing to guarantee the quality and stability of our games. Here are some methods and approaches for automating mobile game testing:

### Frameworks For Test Automation:

There are several specialized frameworks for automating tests in mobile games, each with its strengths. Appium is a versatile option suitable for various platforms, while the Unity Test Framework is tailored specifically for games developed using Unity. However, some frameworks like Calabash, designed for Ruby, Espresso and XC Test, optimized for specific platforms (Android and iOS, respectively), might be less favoured due to their limited support for different programming languages and cross-platform testing.

### Unity Test Framework:

Unity has a Test Runner tool that allows us to run and view the results of our tests. The Unity Editor displays test results, making spotting test failures and errors simple. It uses the UnTest framework, a bespoke testing framework created by Unity. It also includes a Test Runner tool, which allows us to run tests and evaluate the results.

### Appium:

Appium, on the other hand, is a popular open-source test automation framework for automating mobile gaming applications across multiple platforms, including Android and iOS. It enables developers and testers to create and run automated tests for mobile apps written in mainstream programming languages such as Java, Python, Ruby, and others. It is popular because of its versatility, cross-platform compatibility, and active community. Setting up the testing environment, developing test scripts, and running tests on real devices or emulators are all phases of automating mobile app testing with Appium.

While Appium presents a limitation in directly inspecting elements using its inspector tool for mobile apps, this challenge had previously constrained mobile game testing to primarily rely on mass testing conducted by game players to identify defects. AltUnityTester emerged as a solution to address this limitation, enabling comprehensive inspection of games within and outside the Unity editor. This tool significantly broadened the possibilities for in-depth game testing, providing a more detailed approach beyond the conventional limitations of Appium.

### AltUnityTester:

AltUnityTester offers versatile applications through AltTester Unity SDK and AltTester Desktop. AltTester Desktop facilitates game inspection outside the Unity editor, functioning like the Appium Inspector or the DOM Inspector in web browsers. On the other hand, AltTester Unity SDK empowers the creation of test cases directly within the Unity editor. Users can opt for the tool they are most familiar and comfortable with, enhancing their confidence in the testing process.

💡 Secret tip: For those less experienced in Unity coding, the AltTester Desktop is an excellent choice. Simply download and connect it to the game build. This connection can be established through the built-in AltServer or a Remote AltServer using an IP or reverse port forwarding, adapting to the specific device in use.

If you are already developing games in Unity, I suppose you will find AltTester Unity SDK better than AltTester Desktop as it requires knowledge of other programming languages, ide, setup and dependencies. For this, we need to import the AltTester package into Unity Editor,

Resolve the dependencies by adding these dependencies to your `manifest.json` :

```
{
  "dependencies": {
    "com.unity.nuget.newtonsoft-json": "3.0.1"
  },
  "dependencies": {
    "com.unity.editorcoroutines": "1.0.0"
  }
},
{
  "testables": ["com.unity.inputsystem"]
}
```

Dependencies to be added in your manifest.json file

To utilize the AltTester Unity SDK, begin by instrumenting the app and initiating the AltServer Module. Subsequently, run your app within Unity or on your preferred platform. This setup enables the creation and execution of initial tests for the targeted application, streamlining the testing process.

Our first test case starts like this:

```
using NUnit.Framework;
using AltTester.AltTesterUnitySDK.Driver;

public class MyFirstTest
{
    private AltDriver altDriver;

    [OneTimeSetUp]
    public void Setup()
    {
        altDriver = new AltDriver();
    }

    [OneTimeTearDown]
    public void TearDown()
    {
        altDriver.Stop();
    }

    [Test]
    public void TestStartGame()
    {
        altDriver.LoadScene("Scene 2 Draggable Panel");

        altDriver.FindObject(By.NAME, "Close Button").Tap();
        altDriver.FindObject(By.NAME, "Button").Tap();

        var panelElement = altDriver.WaitForObject(By.NAME, "Panel");
        Assert.IsTrue(panelElement.enabled);
    }
}
```

Sample test script using AltTester Unity SDK in Unity with NUnit Framework

Run your test file from the command line by using the following command:

```
<UnityPath>/Unity -projectPath $PROJECT_DIR -executeMethod AltTestRunner.RunTestFromCommandLine -test:
```

Command to Run in terminal

### Continuous Integration (CI):

Mobile game testing can also be incorporated into our CI/CD pipeline. This ensures that tests are run automatically whenever a new build or code update is made. CI tools that support mobile game testing include Jenkins, Travis CI, and CircleCI. Our specific demands, project objectives, and preferences influence the choice of tools. Jenkins is used for platform support, extensibility, community, and ecosystem. Similarly, for SAAS, quick setup, integration, and version control, we favour Travis CI and CircleCI.

### Load Testing:

We can automate load testing to determine how the game performs under heavy user loads. Apache JMeter and LoadRunner, for example, can simulate a huge number of concurrent users.

### Performance Testing:

GameBench, Unity Performance Testing, and Appium with Performance Plugins each offer distinct ways to measure performance indicators such as frame rates, battery consumption, CPU utilization, and memory usage. Unity Performance Testing seamlessly records and analyzes performance data directly within the Unity Editor, while Appium integrates performance testing plugins or libraries for comprehensive analysis.

Moreover, security testing becomes crucial for games handling user data or transactions. This process identifies vulnerabilities and safeguards user information, ensuring a secure gaming experience.

### Conclusion:

As we peer into the future, it's evident that more sophisticated automated game-testing models will emerge to tackle complex testing scenarios. The idea that game testing can't be automated has significantly expanded since reading the blog. We now have a wider range of options, reshaping how we approach quality assurance beyond solely relying on manual mass game testing to spot bugs.

With automated testing gaining ground, there's a chance we'll discover new bugs that traditional human testing might have missed. This hints at a future where automated testing becomes a key player in ensuring game quality and uncovering issues that might have slipped through the cracks earlier.

Thank you for reading this article. See you in the next one.

Receive Next Article on Email

If you liked this article, feel free to share this post on

Facebook, Twitter or LinkedIn.

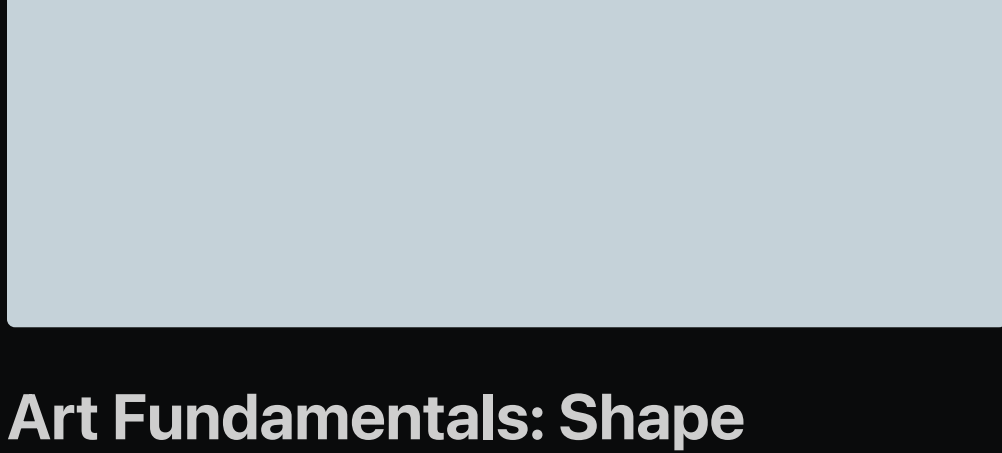
Follow Yarsa Labs on LinkedIn



### Device Farming in the QA Process


Discover Device Farming: A crucial technique for testing apps across multiple devices. Learn its benefits and impact on software development.

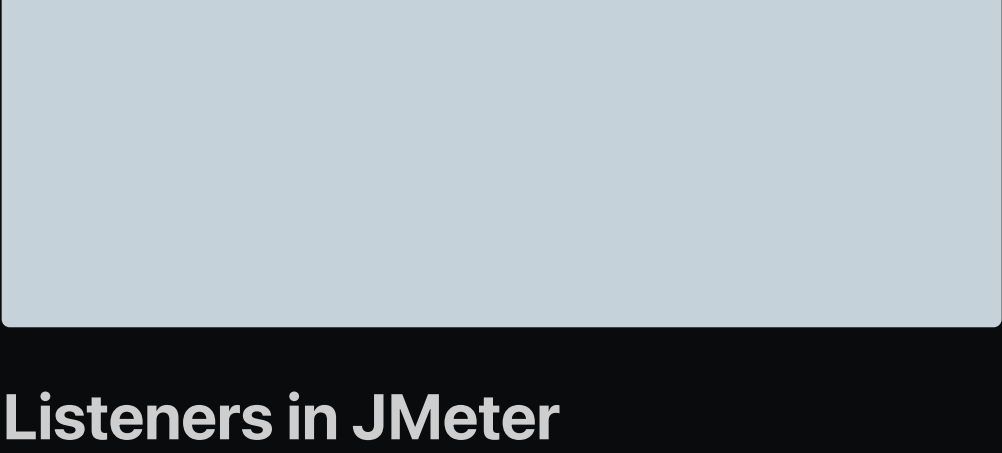
Ayya Pathak  
Sep 13, 2024 · 4 min read



### Art Fundamentals: Shape Language in Character Design


Learn how Shape Language influences character and object design by using common shapes like circles, squares, and triangles to convey personali...

Aisha Lama  
Sep 13, 2024 · 5 min read



### Listeners in JMeter

Discover JMeter listeners: essential tools for visualizing and analyzing performance test results. Learn about these 14 types to enhance your testin...

Sabina Bajracharya  
Sep 9, 2024 · 7 min read