

Challenges with Automation for Games

Altom

Nice to
meet you!



What do we do?

- We are a company offering testing services
- We teach through our BBST courses
- We are building tools that help with test automation (AltTester®, AltWalker, AltTap)



How about you?

- How familiar are you with game development?
- Have you implemented apps using Unity or other engines?
- What challenges did you face while testing the app?

Game testing world



Game World

- Game Engines (like Unity, Unreal, Custom Engines)
 - develop once, export to many platforms
 - a lot is common, but each platform has its own quirks

Traditional software development vs. Game software development



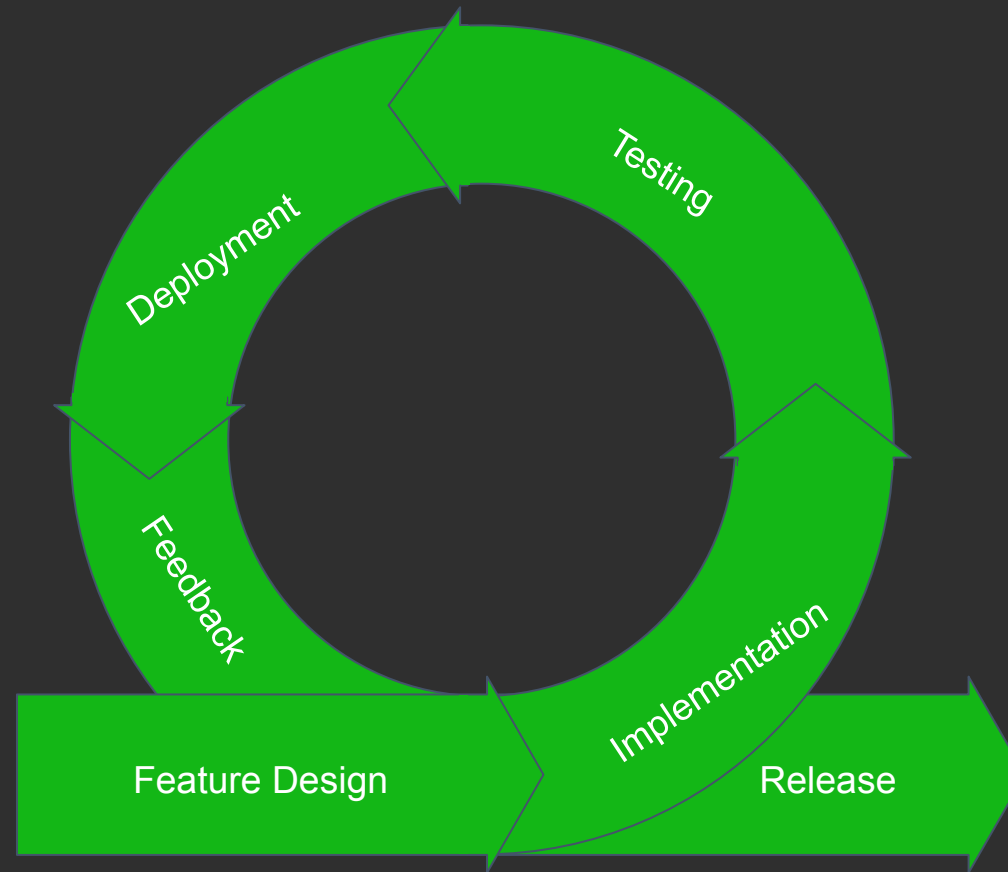
Traditional / Agile Software Development

- Every change in the code is tested
 - functional tests as automated scripts
 - give **continuous feedback** on how other parts of the software are affected by the change

Game Software Development

- QA team performs functional testing once features are completed
- Playtesting, Alpha testing, Beta testing

Traditional / Agile Software Development



Challenges with non-automated approach

- Games are generally complex and testing all functionalities manually becomes too time consuming
- Games are frequently updated and you have to retest the same thing all over again
- You have to test the same functionalities on multiple platforms
- Make sure nothing broke from the existing features

Options for Game Test Automation



Many programming languages

Supports iOS and Android,
Mac, Windows

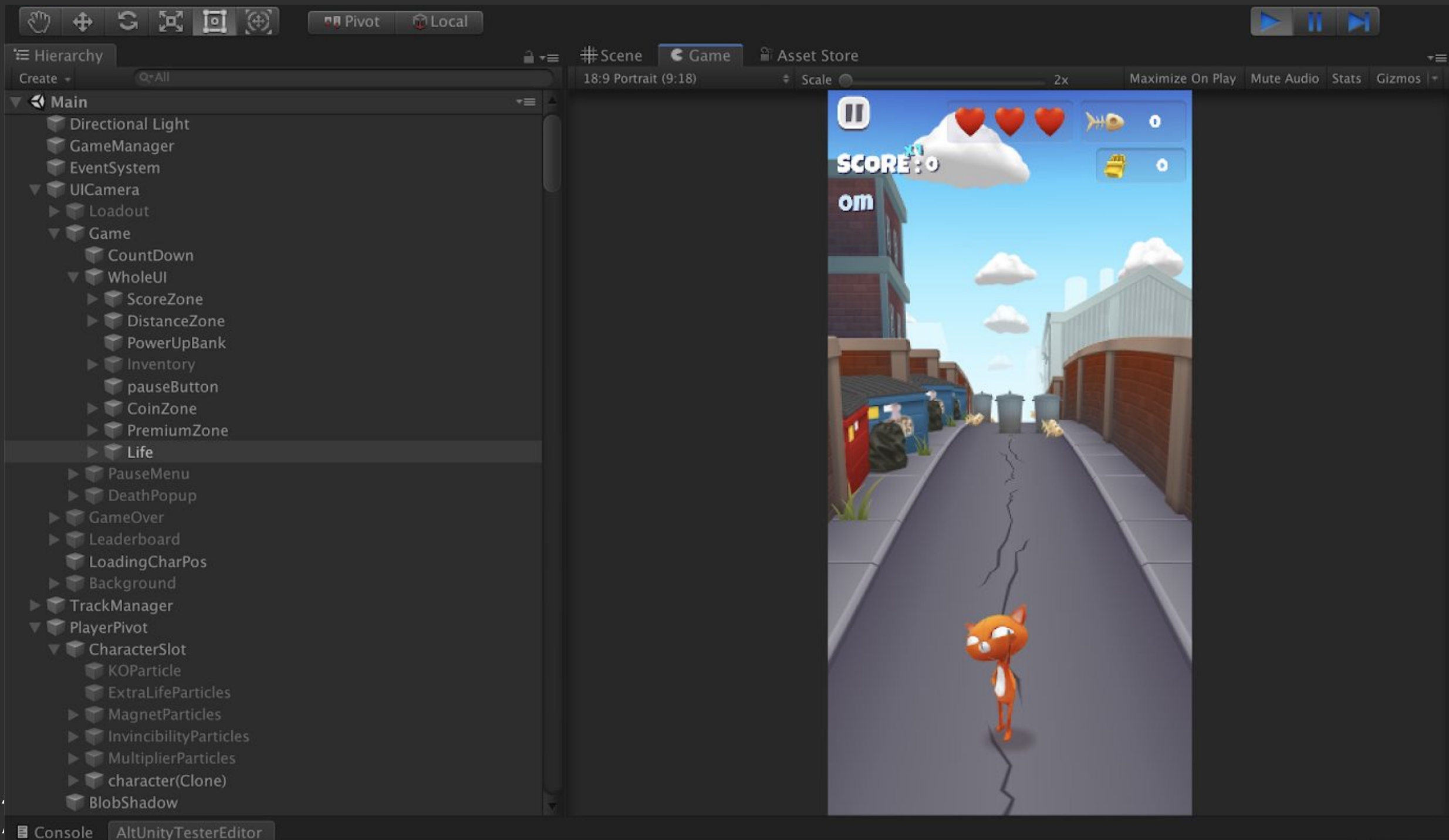
Offers great interaction with
the target platform



APPIUM

Works with Browserstack, Bitbar Cloud,
AWS and other Device Cloud platforms

Testers who are familiar with
Selenium find it easy



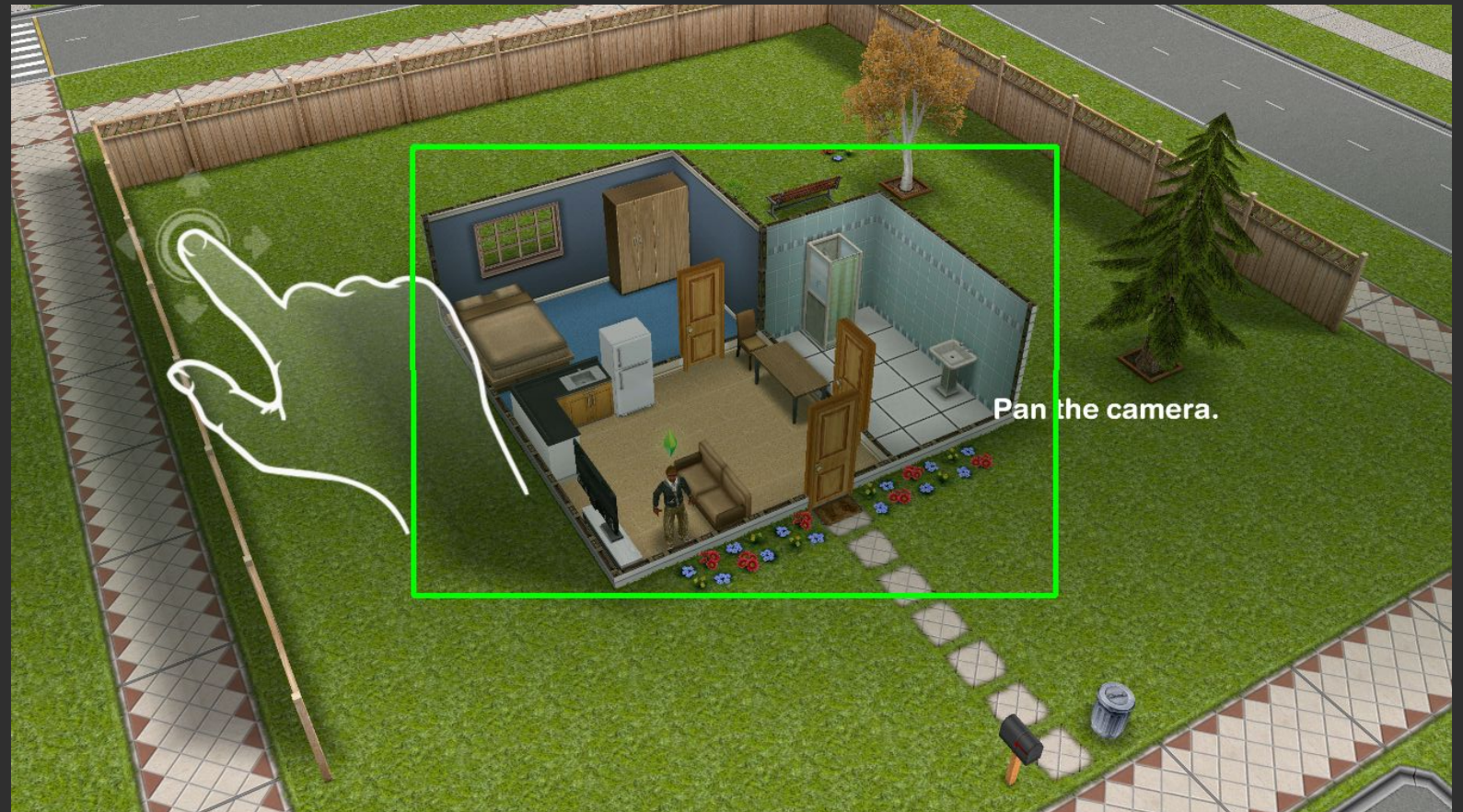


—Click at coordinates—



Image Recognition

Define “House” element using an image



Define “House” element using an image



OpenCV - Feature detection and matching



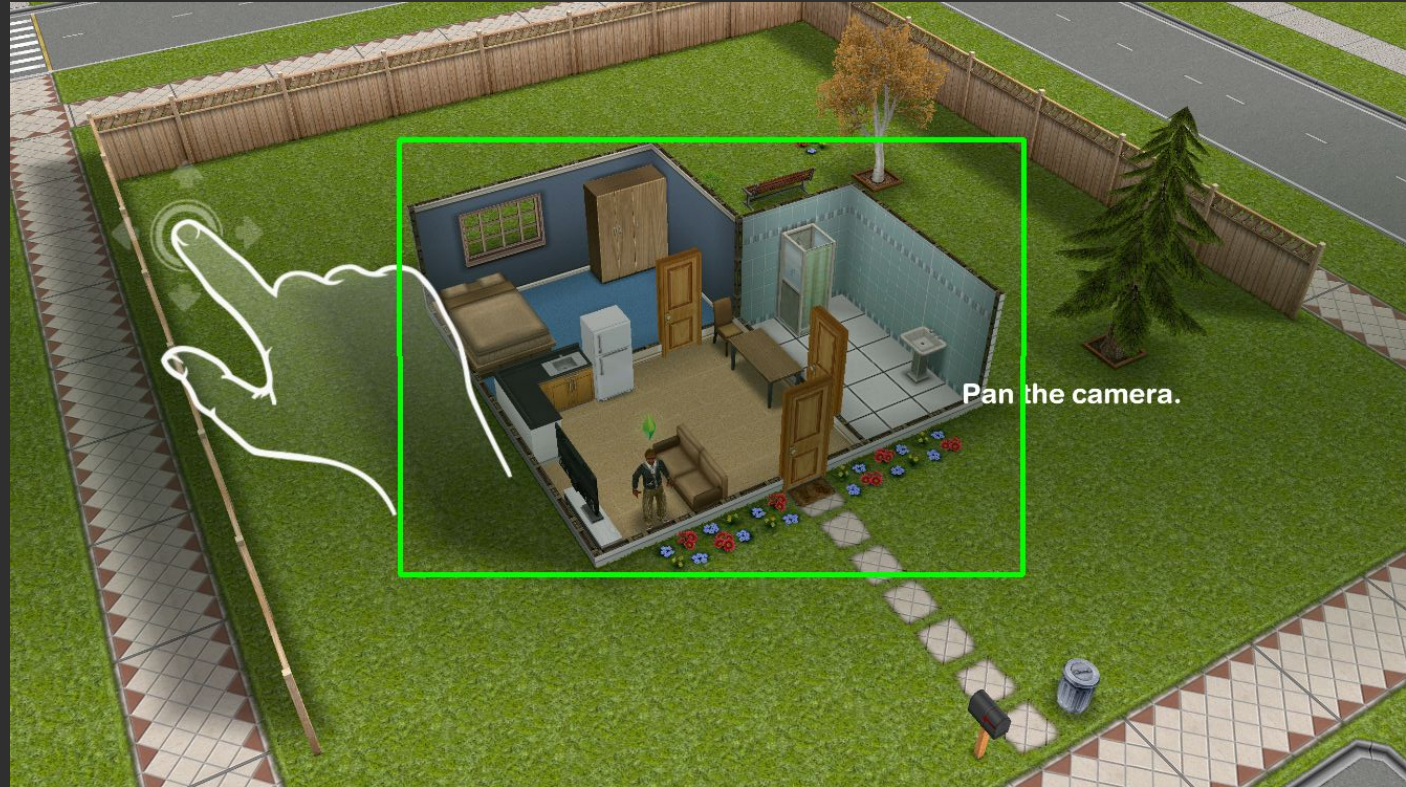
OpenCV - Feature detection and matching



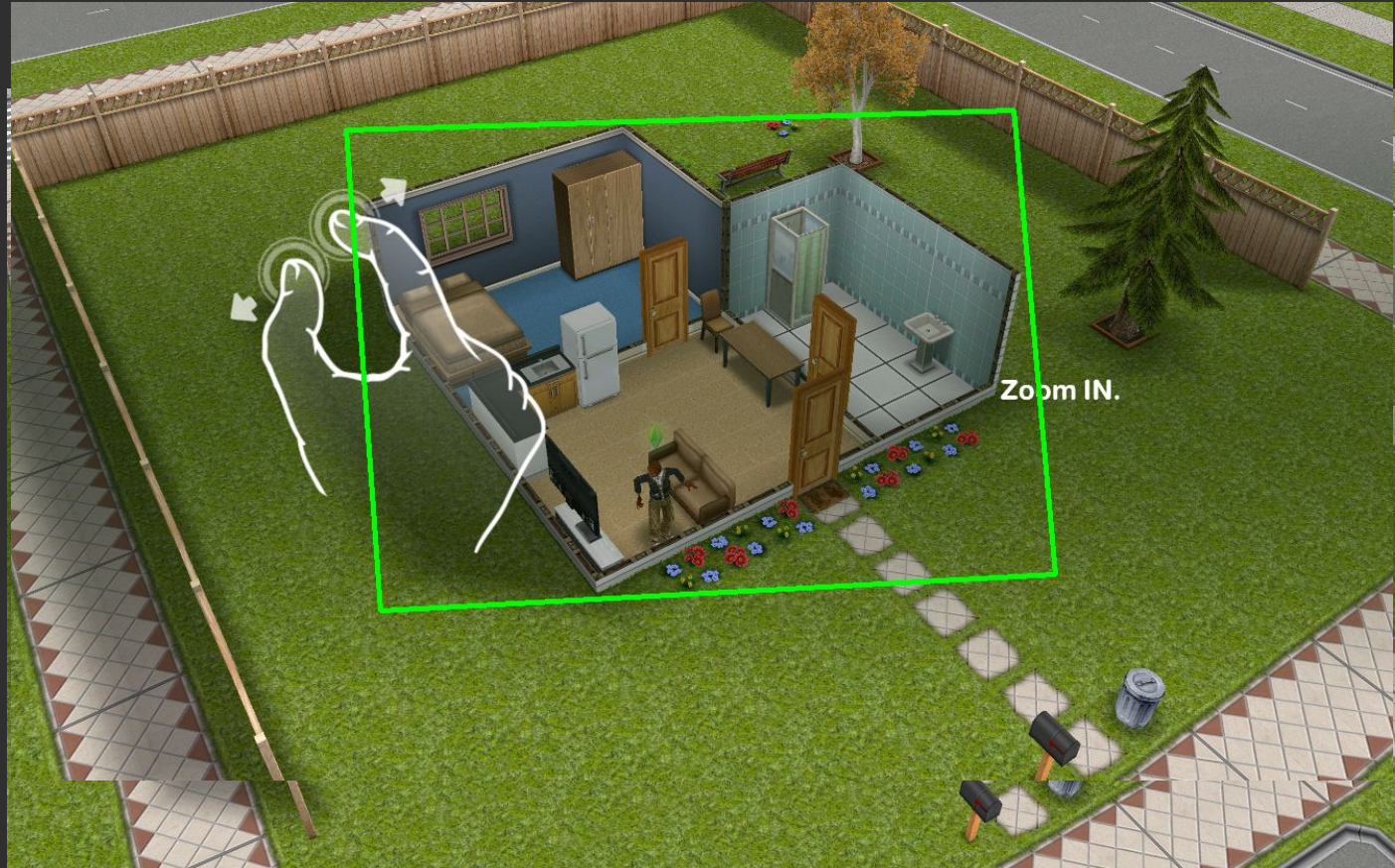
OpenCV - Feature detection and matching



Resolution Agnostic, works rotated and stretched items



Resolution Agnostic, works rotated and stretched items



Resolution Agnostic, works rotated and stretched items



Resolution Agnostic, works rotated and stretched items



Image Recognition + Appium

<https://github.com/bitbar/testdroid-samples/tree/master/image-recognition>

- worked well for simple scenarios
- didn't require any changes to the game
- found really useful issues
 - out of memory problems
 - crashes
 - graphics missing/not displayed correctly

Image Recognition + Appium

<https://github.com/bitbar/testdroid-samples/tree/master/image-recognition>

- was not fast enough
- graphics change all the time in games
- didn't work well for apps that had a lot of text
- didn't give enough granularity when it came to identifying objects



Struggles with Test Automation in Games



Struggles with UI Automation in Games

- Small teams => no dedicated testers, developers don't have time to invest in automated tests
- Large teams => large teams of “manual” testers with not enough coding skills; manual testing takes a lot of time

Struggles with Test Automation in Games

- Make the tools more accessible to testers that have no coding skills
- Cover multiple engines (Unity, Unreal, Godot, o3de)
- Cover multiple platforms
- Keep the development engagement

Beyond Image Recognition

AltTester[®]



Altom

▼ Main :
Main Camera
Directional Light
GameManager
▶ Explosion
EventSystem
▶ UICamera
▶ TrackManager
▶ PlayerPivot
▼ DontDestroyOnLoad :
▶ MusicPlayer
CoroutineHandler
AssetBundleManager

TRASH CAT



Run!



LEADERBOARD



STORE



MISSIONS



SETTINGS



Altom

▼ Main :
Main Camera
Directional Light
GameManager
Explosion
EventSystem
UICamera
TrackManager
PlayerPivot
DontDestroyOnLoad :
AltTesterPrefab
MusicPlayer
CoroutineHandler
AssetBundleManager

▼ Main
Main Camera
Directional Light
Explosion
GameManager
EventSystem
UICamera
TrackManager
PlayerPivot
DontDestroyOnLoad
AltTesterPrefab
MusicPlayer
CoroutineHandler
AssetBundleManager

TRASH CAT



Run!



LEADERBOARD



STORE



MISSIONS



SETTINGS



Altom



Game/App

GamePlay Scene

Obstacle

Cat

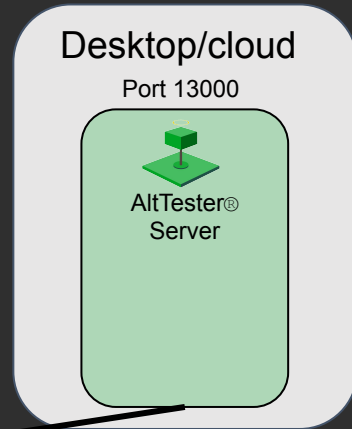
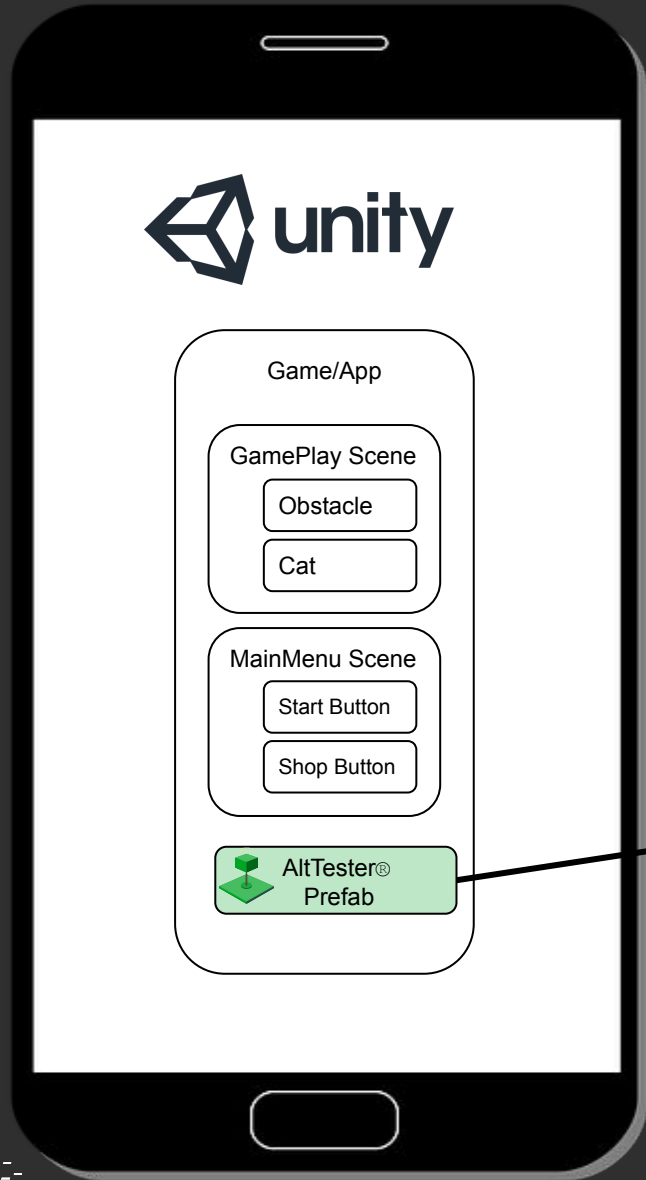
MainMenu Scene

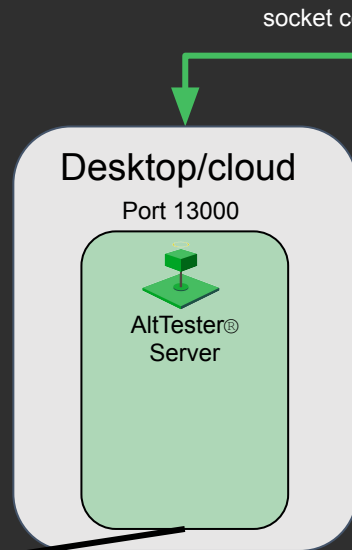
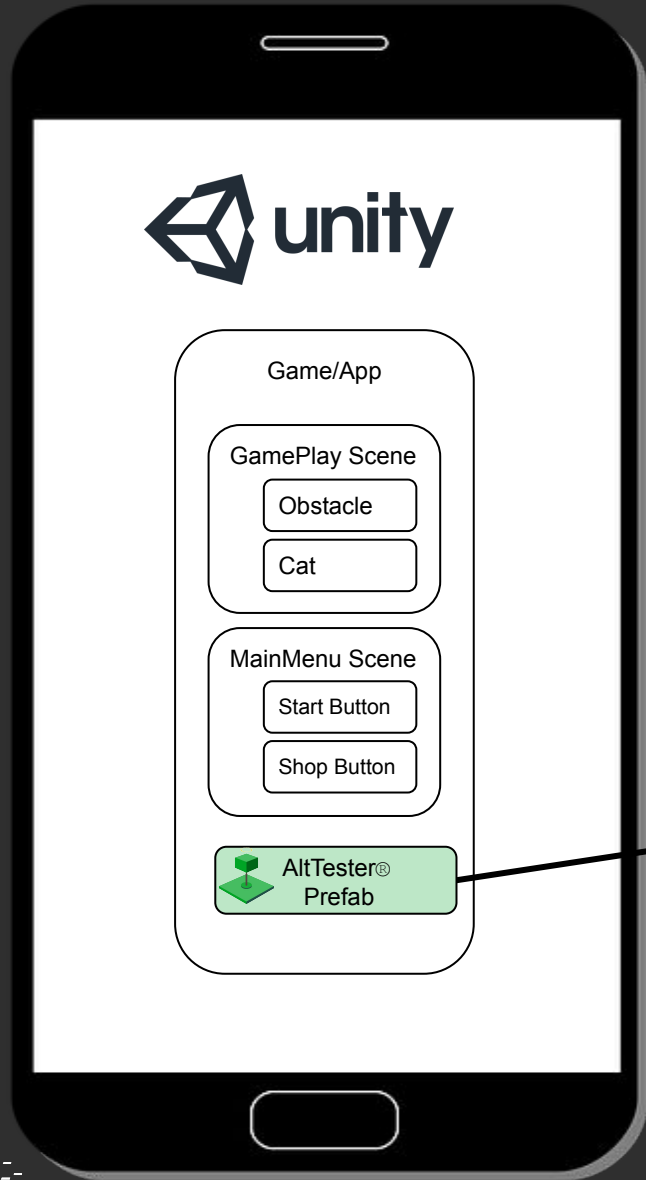
Start Button

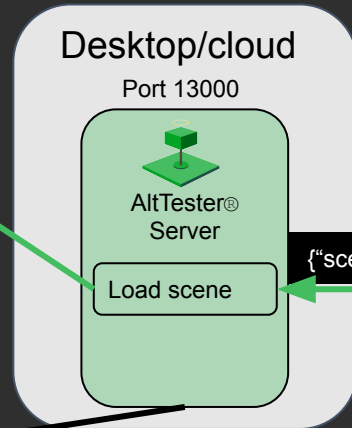
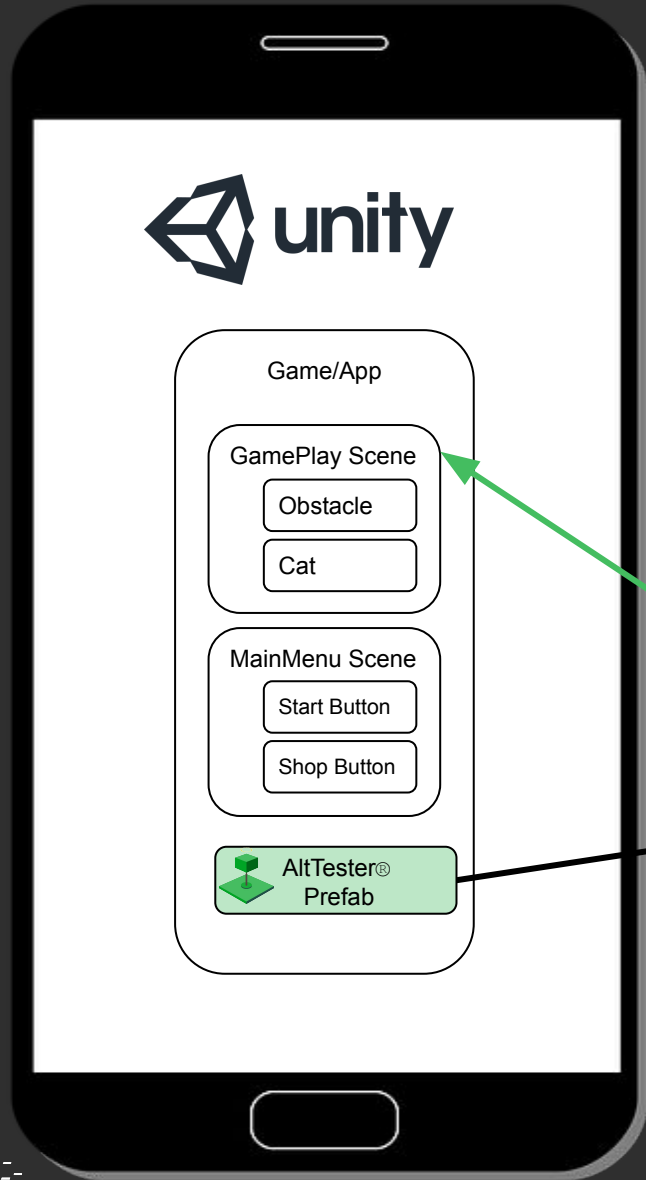
Shop Button

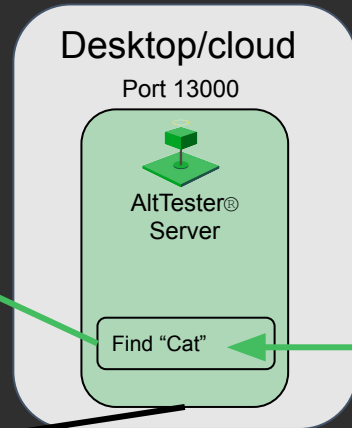
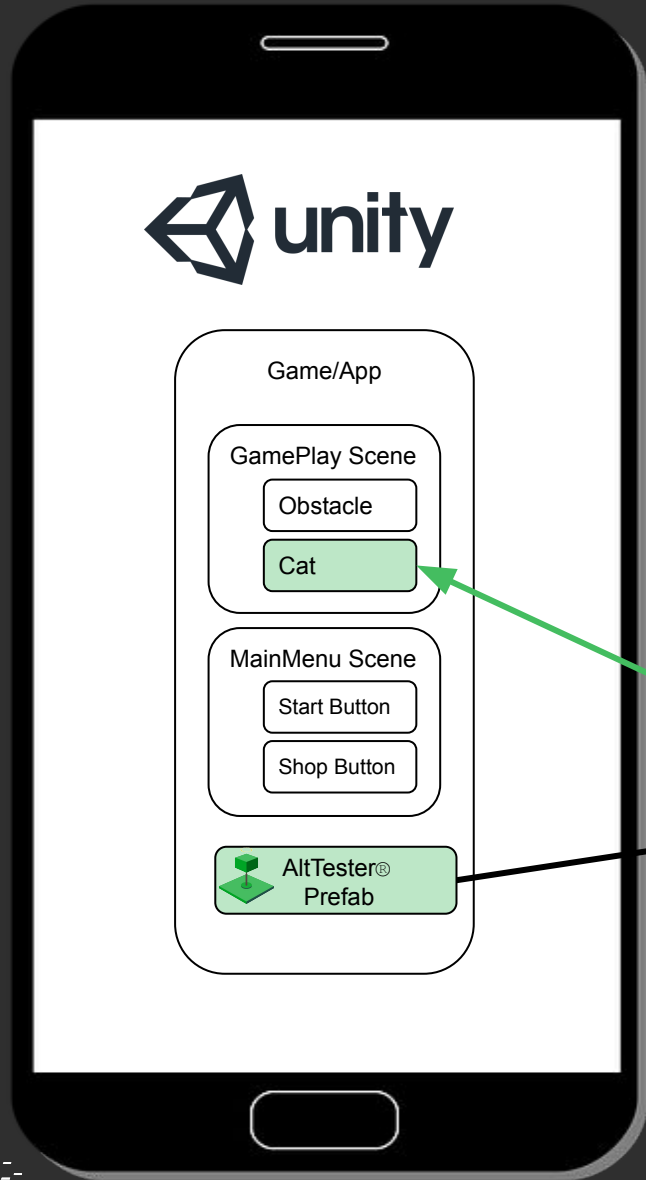


Altom









AltTester[®]

<https://github.com/alttester>

- easier access to objects from the scene
- reduces the time taken for testing - automated tests give fast feedback
- concurrent test execution on multiple devices

AltTester®

- test execution in the cloud (AWS, BitBar, BrowserStack)
- easy to learn by non-technical people (Inspect, Record)
- people skilled in web testing or mobile testing (apps, not games)
could learn faster



AltTester[®]

- worked well even for complex scenarios
- really fast and worked for all types of games
- tests are easier to maintain than with image recognition



Challenges implementing AltTester[®]

- A more technical person is needed to create the instrumented build. What are the consequences?
- Is click on coordinates reliable?
- Unity has multiple types of objects (objects from the scene, menus, UI toolkit).
- What if the game uses meshes and not objects?

Challenges implementing AltTester[®]

- Unity has an old and a new input system
- What if the game to be tested uses a custom input system?
- Recorder - what actions do we record?
- Running the same set of tests on 2 standalone builds on the same machine. What happens with the resources?

- Manual testing or automation?
 - ex1. : VuCity - add floors to a newly created building. Has the building the expected height?
 - ex2. : test fonts

Keep the conversation going



Thank you!