

# FUTURE DECODED

6-7 OTT '16 / MILANO

IN PARTNERSHIP WITH:



CommunityDays.it

[www.futuredecoded.it](http://www.futuredecoded.it)



#FutureDecoded



# Automated UI Testing for iOS and Android Mobile Apps

Massimo Bonanni

*Microsoft Sr. Consultant - EMEA Modern App Domain*

✉ [massimo.bonanni@microsoft.com](mailto:massimo.bonanni@microsoft.com)

🐦 [@massimobonanni](https://twitter.com/massimobonanni)

[www.futuredecoded.it](http://www.futuredecoded.it)

🐦 [#FutureDecoded](https://twitter.com/FutureDecoded)



# Agenda

Perchè test!!!

Unit vs Coded UI Test

Xamarin Test Cloud

Xamarin.UITest, REPL, Test Recorder

Continuous Integration

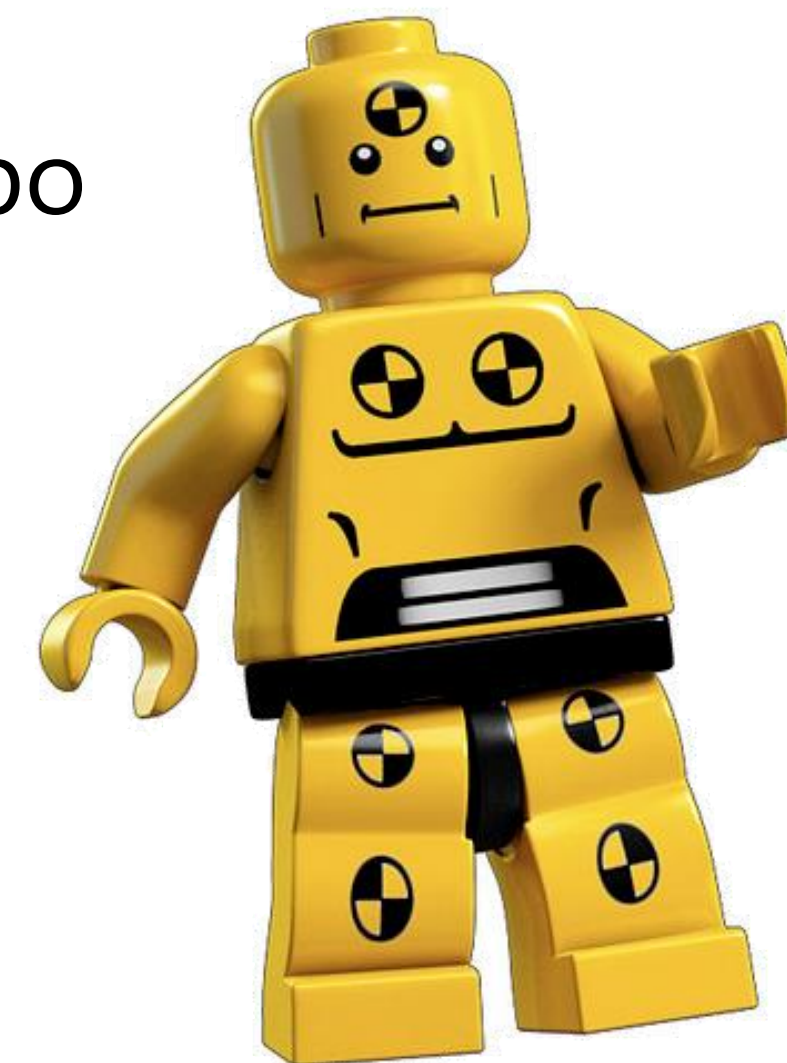
Quanto mi costi.....



# Perché test!!!!

- + Individuazione dei bug o defect prima del rilascio
  - + Incremento della qualità
  - + Garanzia di non regressione a seguito di refactoring o nuove funzionalità
  - + Se unito alle build automatiche, utilissimo in cicli rapidi di sviluppo
- 
- Più codice da scrivere
  - Maggiori skill da parte del team
  - Maggiori costi del prodotto

**Nell'ambito mobile, cross-plattform, c'è un'altra cosa che ci spinge ad adottare politiche di test.....**





# Tutta una questione di numeri...



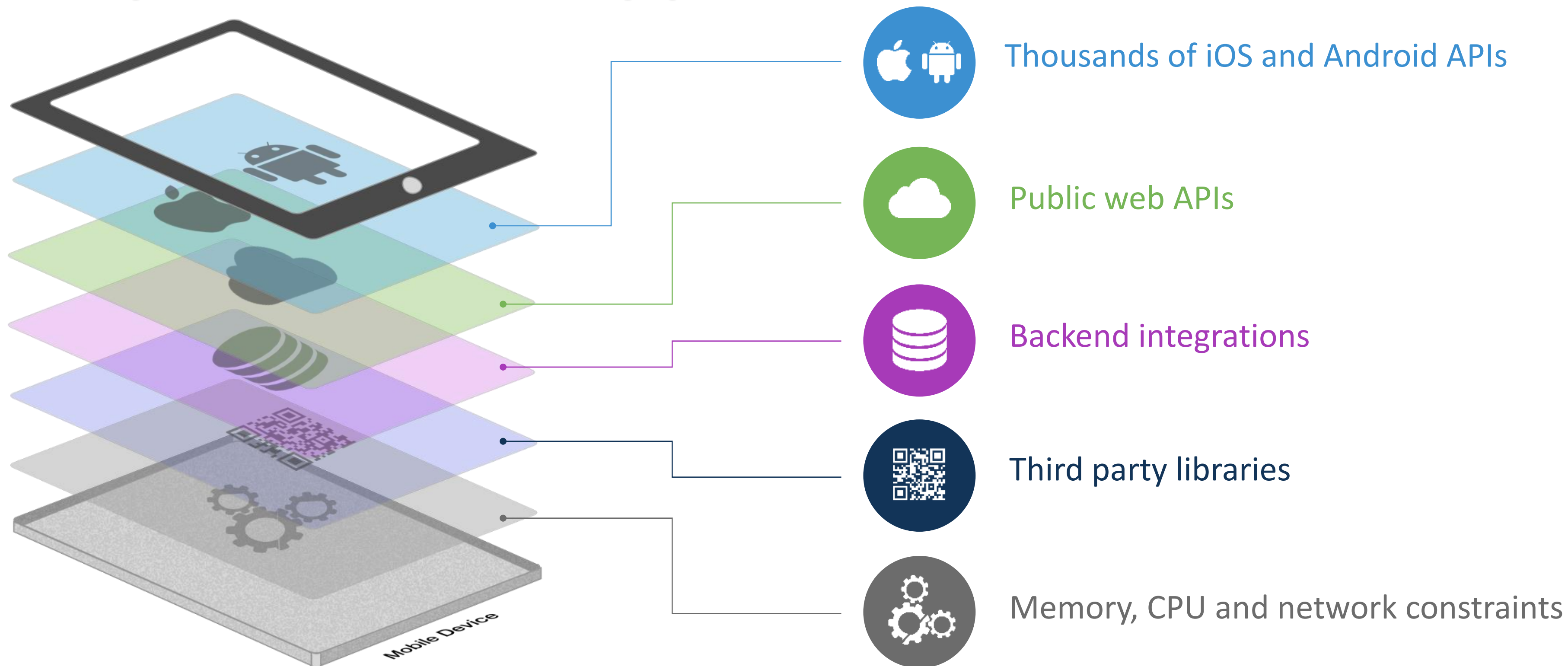
5 Versioni  
20 Devices  
20 Lingue  
35 Localizzazioni  
6 Formati  
schermo



9 Versioni  
24K+ Differenti Device  
39 Lingue  
57 Localizzazioni  
27 Formati schermo  
1,294 Produttori  
6 Configurazioni  
Schermo



# Complessità delle app



# Unit Test....non basta!!

Disaccoppiare la UI dalla logica restrostante (es. MVVM) e testare i ViewModel con degli Unit Test non basta a garantire che la User Experience sia delle migliori:

- I tempi di reazione dell'interfaccia variano in base alla potenza del dispositivo
- La corretta visualizzazione dei controlli della UI dipendono dalla dimensione e dal form factor dello schermo
- I tempi di interazione con i sistemi esterni possono creare problemi non previsti in sede di unit test
- ....





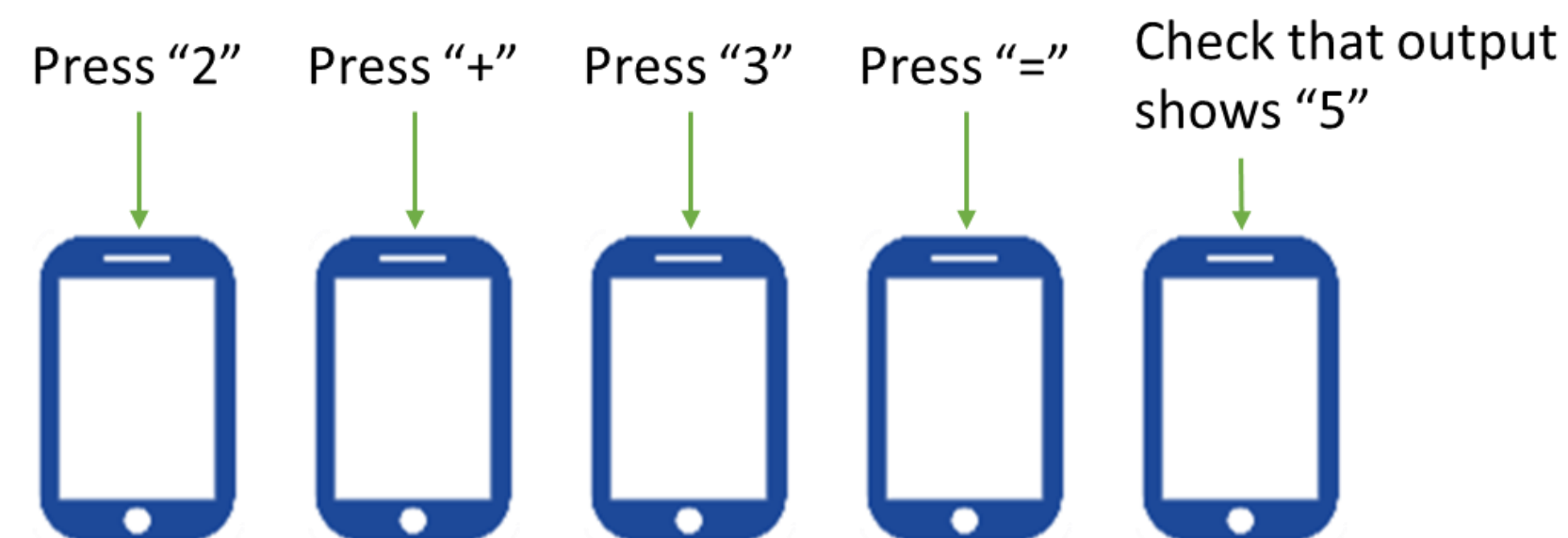
# E....allora?

Il modo migliore per validare il comportamento di una applicazione è utilizzarla.

Se l'applicazione ha il comportamento atteso senza errori o eccezioni, allora può essere rilasciata.

Il processo di test del comportamento della UI prende il nome di UI Acceptance Test o Coded UI Test (quando sono «codificati»).

UI test script:





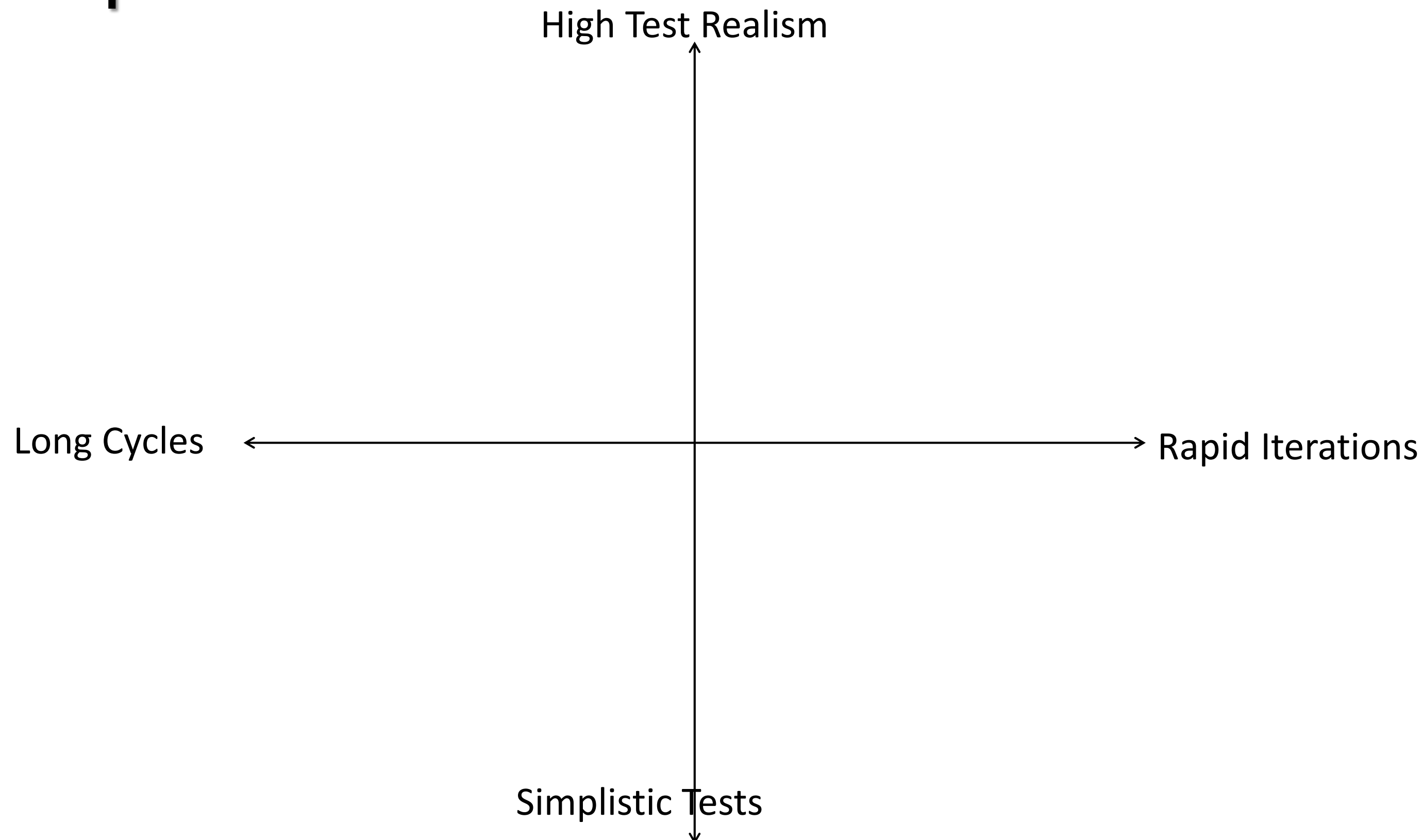
# UI Acceptance Test

Questo processo ha i seguenti limiti:

- E' costoso perché necessita dei dispositivi hardware;
- E' costoso perché debbono essere impiegate persone per il test manuale;
- E' dispendioso a livello di tempo perché il tester è un essere umano (processo poco adatto a rilasci rapidi)
- E' prone di errori perché l'uomo tende a distrarsi e sbagliare quando esegue operazioni ripetitive e meccaniche
- E' di difficile rendicontazione (il tester può semplicemente dire se il test è andato bene o andato male ma senza l'utilizzo di tecnologie accessorie non è in grado di ottenere altre informazioni come memoria o cpu occupata)



# Il magic quadrant dei test





# Cosa facciamo?

Soluzioni:

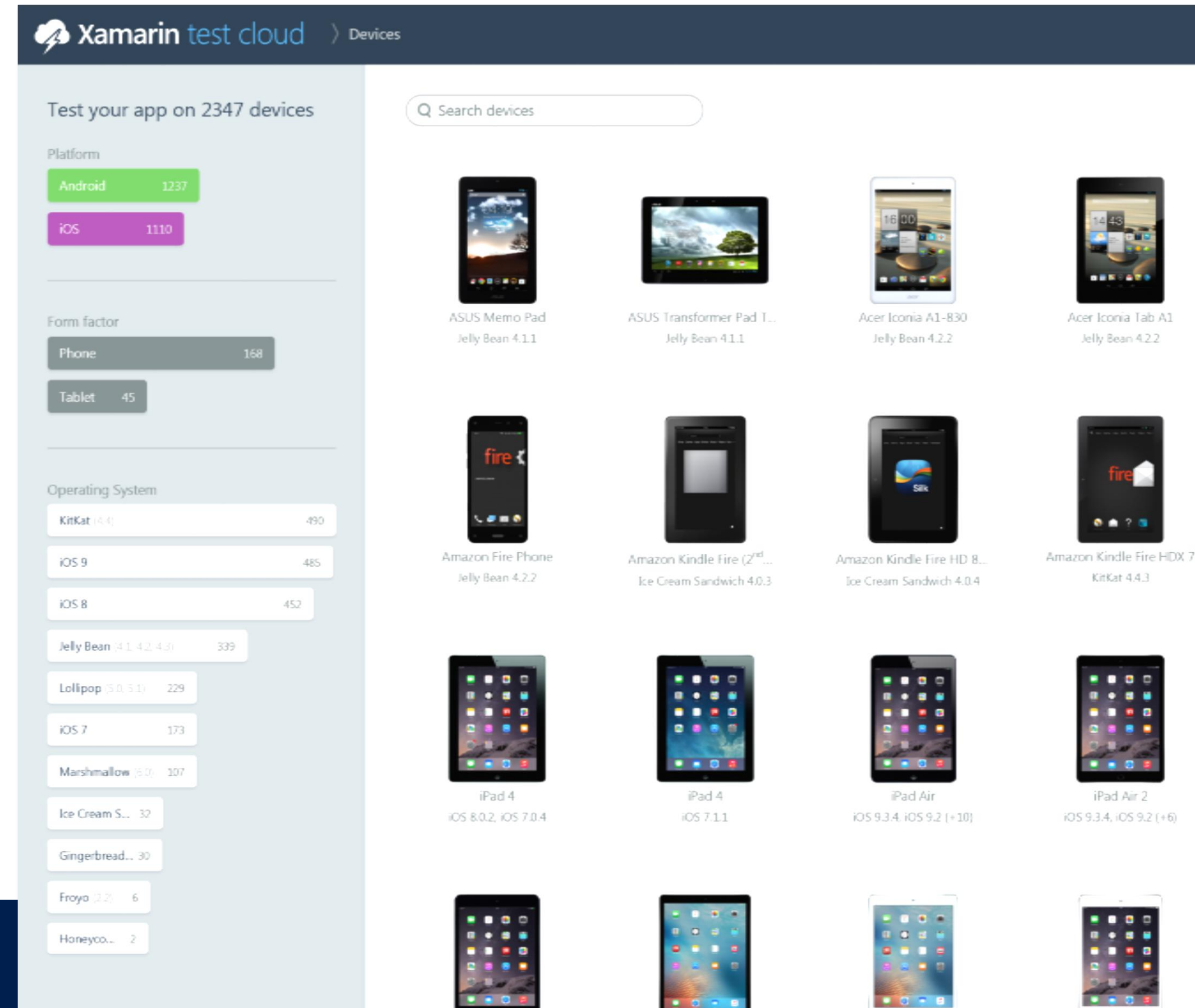
- Non si fa UI Test !!! 😞 😞
- Si spera che tutto funzioni al meglio!!! 😞 😞 😞
- Si sfrutta il concetto di beta
- ...
- Si utilizza una piattaforma di testing automatica, scalabile e veloce: **Xamarin Test Cloud**





# Cosa è Xamarin Test Cloud?

- Xamarin Test Cloud è un servizio basato su cloud che fornisce una piattaforma di test per la user interface automatizzata
- Fornisce centinaia di dispositivi diversi di differenti brand e sistemi operativi
- Consente a chiunque di verificare che il comportamento della UI sia conforme ai requisiti attraverso una varietà di dispositivi con il minimo sforzo.
- La manutenzione e l'approvvigionamento di nuovi dispositivi viene semplificata di molto dalla natura Cloud del sistema





# Cosa è Xamarin Test Cloud?

Xamarin Test Cloud supporta i seguenti framework di test:

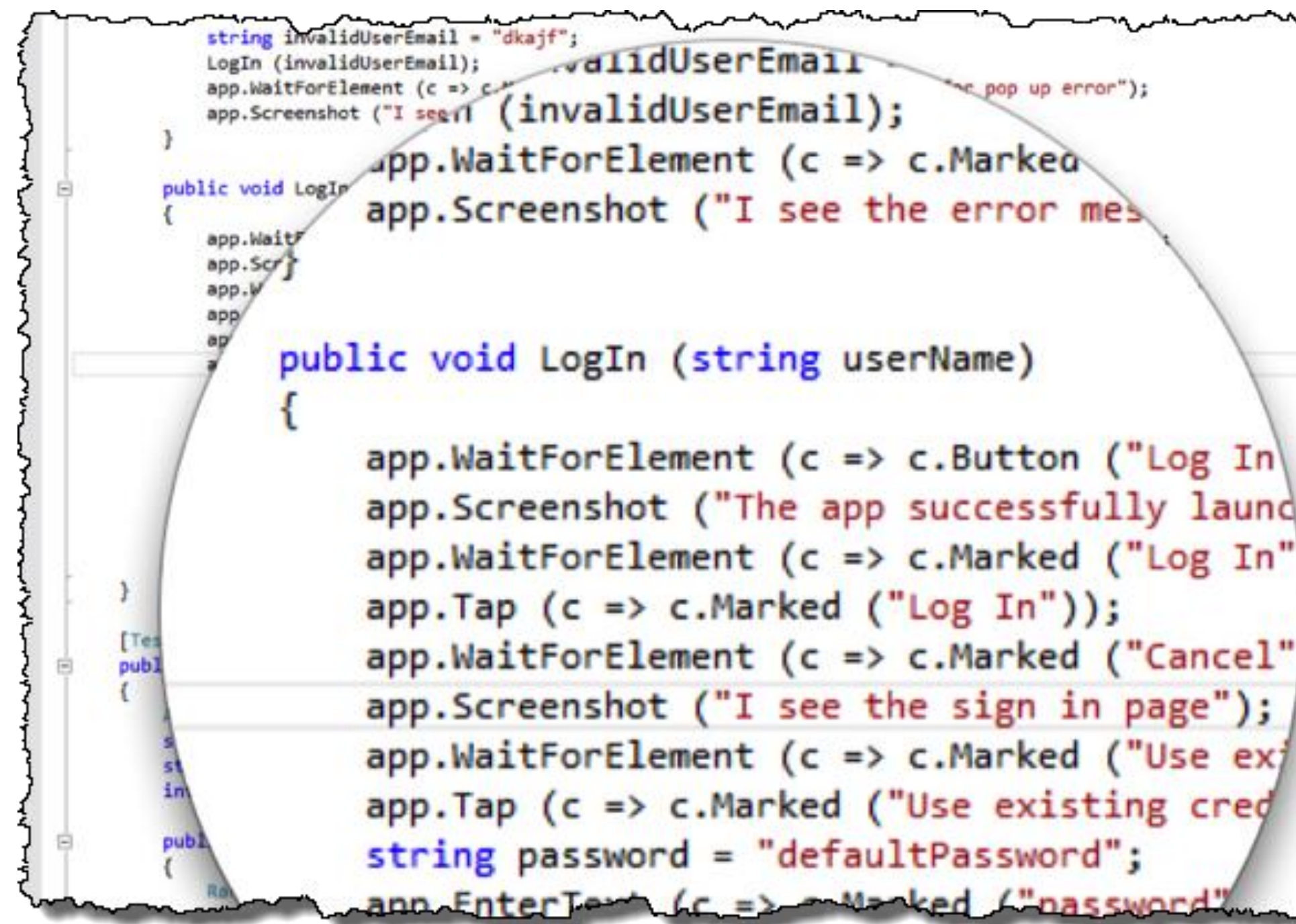
**Xamarin.UITest:** framework che permette di scrivere test in C#. Basato su Nunit. Utilizzabile nelle applicazioni cross-plattform scritte con Xamarin (sia native che Forms)

**Calabash:** framework che permette di scrivere test in Ruby utilizzando Cucumber. Permette di scrivere test utilizzando «la lingua del business» sfruttando le regole grammaticali di Cucumber. Utile per le applicazioni native scritte senza l'uso di Xamarin



# Xamarin.UITest

- Basato su Nunit
- Visual Studio e Xamarin Studio forniscono template di progetto per UI Test
- Fornisce supporto alle gesture
- Struttura simile agli unit test
- Modalità per eseguire Xamarin.UITests:
  - Eseguire l'upload dei test (e dell'app) su Xamarin Test Cloud.
  - Eseguire i test localmente utilizzando un device, un emulatore (Android), o un simulatore (iOS) e sfruttando il Test Runner in Xamarin Studio (iOS o Android) o Visual Studio (Android).

A screenshot of a code editor showing Xamarin.UITest code. The code is written in C# and includes comments in Italian. It defines a test method 'LogIn' that interacts with an application. The code includes steps for logging in with an invalid email, checking for an error message, and logging in with valid credentials. The code is partially obscured by a circular graphic.

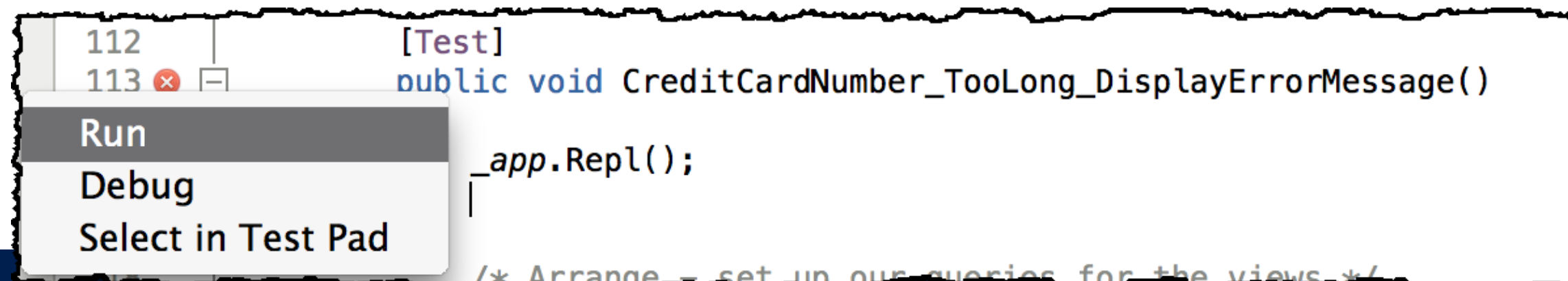
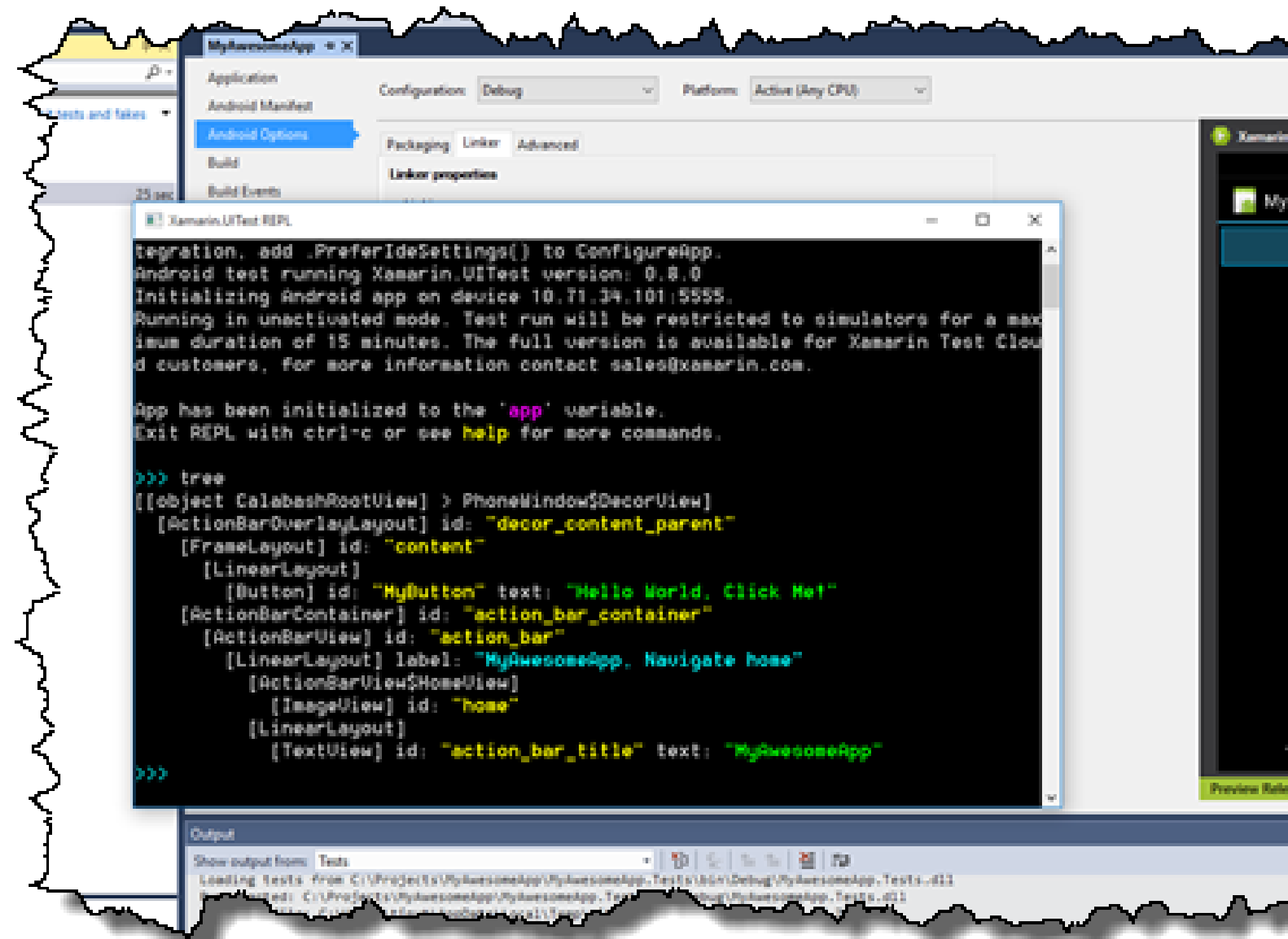
```
string invalidUserEmail = "dkajf";
LogIn (invalidUserEmail);
app.WaitForElement (c => c.Text ("pop up error"));
app.Screenshot ("I see an error message");

public void LogIn (string userName)
{
    app.WaitForElement (c => c.Button ("Log In"));
    app.Screenshot ("The app successfully launched");
    app.WaitForElement (c => c.Marked ("Log In"));
    app.Tap (c => c.Marked ("Log In"));
    app.WaitForElement (c => c.Marked ("Cancel"));
    app.Screenshot ("I see the sign in page");
    app.WaitForElement (c => c.Marked ("Use existing credentials"));
    app.Tap (c => c.Marked ("Use existing credentials"));
    string password = "defaultPassword";
    app.EnterText (c => c.Marked ("password"), password);
}
```



# REPL

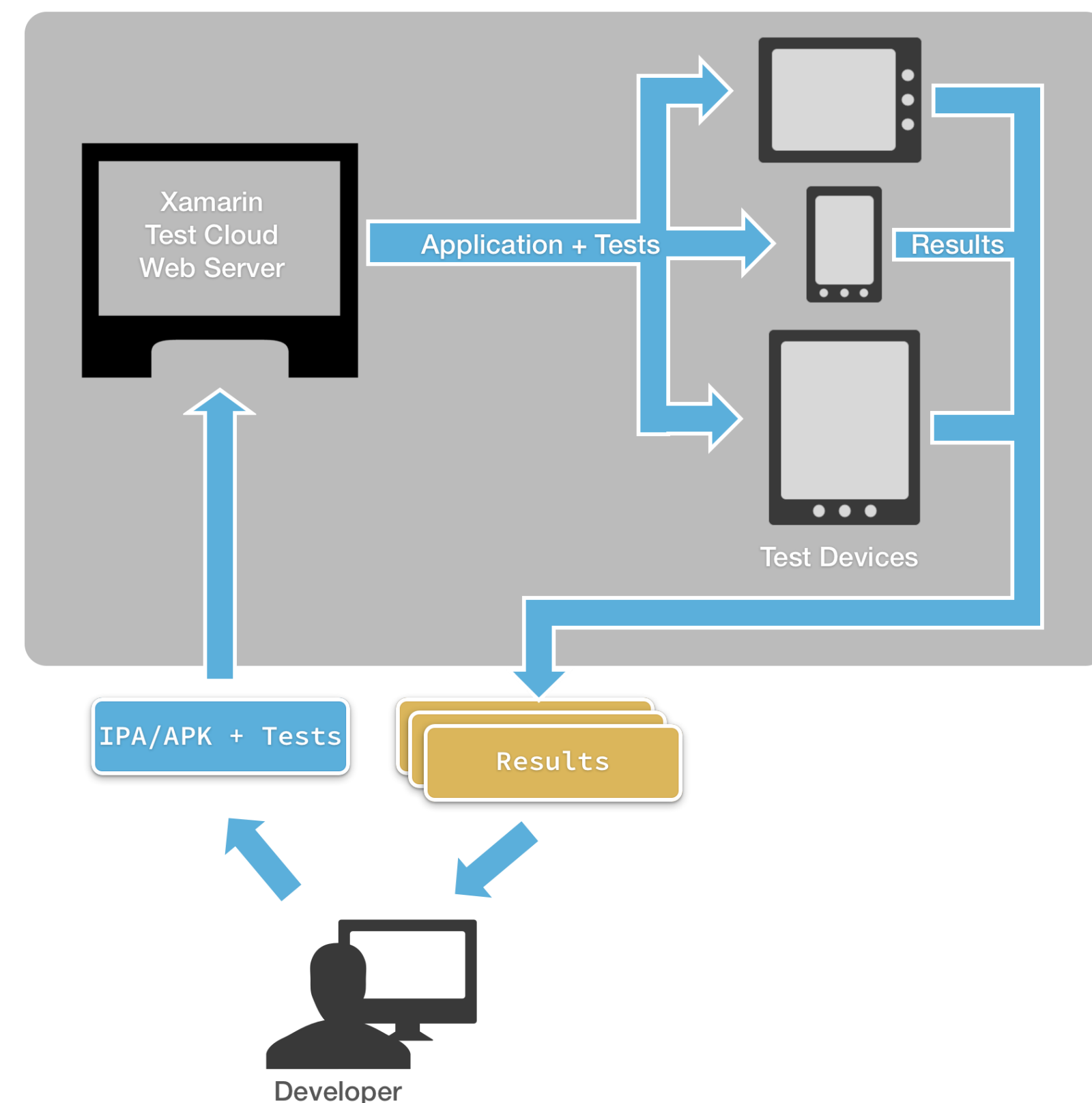
- REPL (**read-eval-print-loop**) è uno dei più importanti tools per la creazione di UI Test.
- Tool a riga di comando con cui gli sviluppatori possono eseguire espressioni e comandi.
- Permette di “esplorare” l’interfaccia utente ed interagire in tempo reale con essa.
- I comandi utilizzati possono essere esportati in Visual Studio (o Xamarin Studio)





# Come funziona Xamarin Test Cloud

1. I test vengono creati localmente (in un progetto Visual Studio di test)
2. L'applicazione sotto test e i relativi test vengono inviati a Xamarin Test Cloud
3. Applicazione e test vengono installati ed eseguiti sui device selezionati dall'utente
4. Al termine di tutti i test, Test Cloud colleziona i risultati e li invia all'utente





# DEMO – My first UI.Test

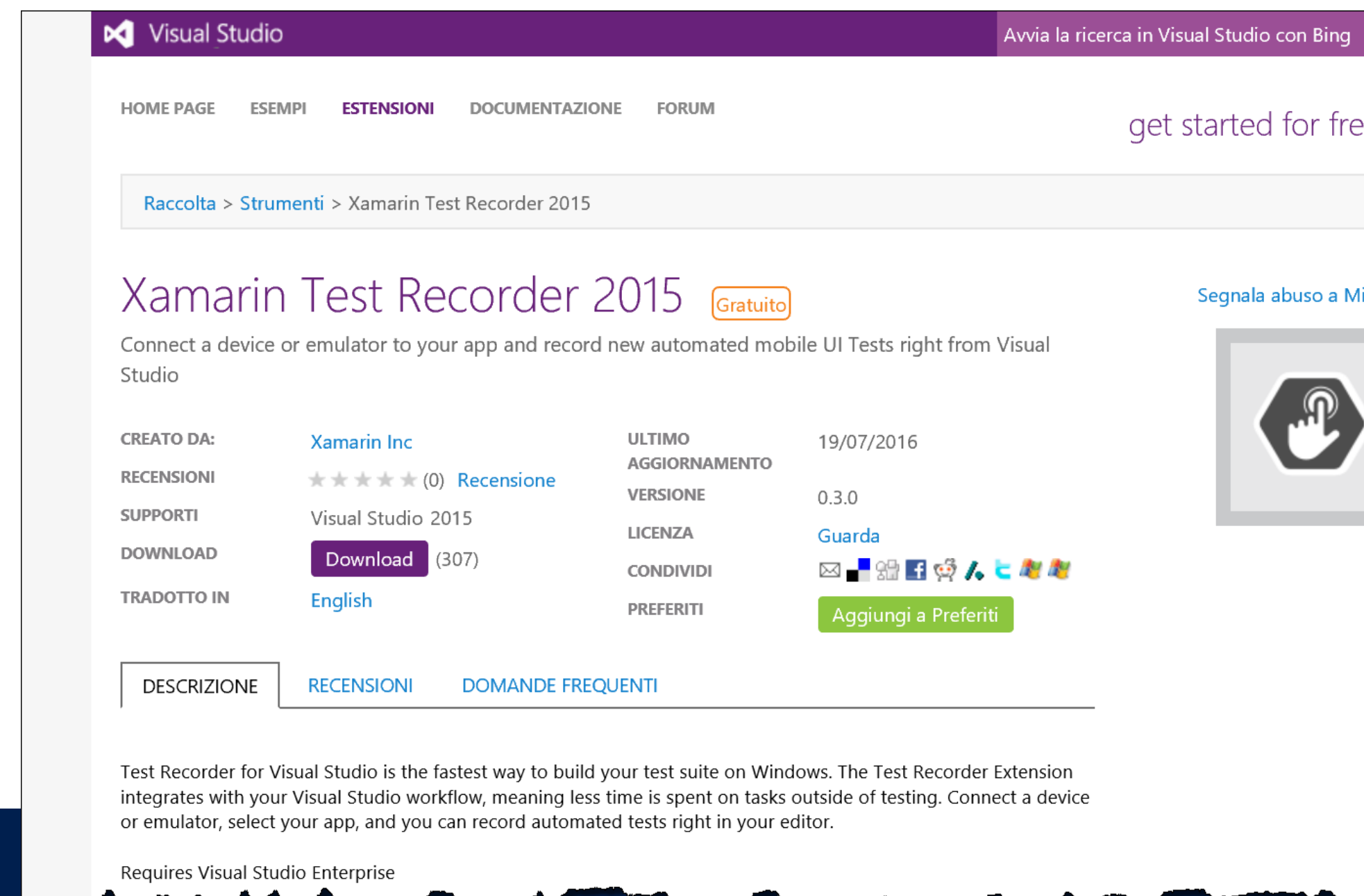
[www.futuredecoded.it](http://www.futuredecoded.it)

 #FutureDecoded



# Xamarin Test Recorder for Visual Studio

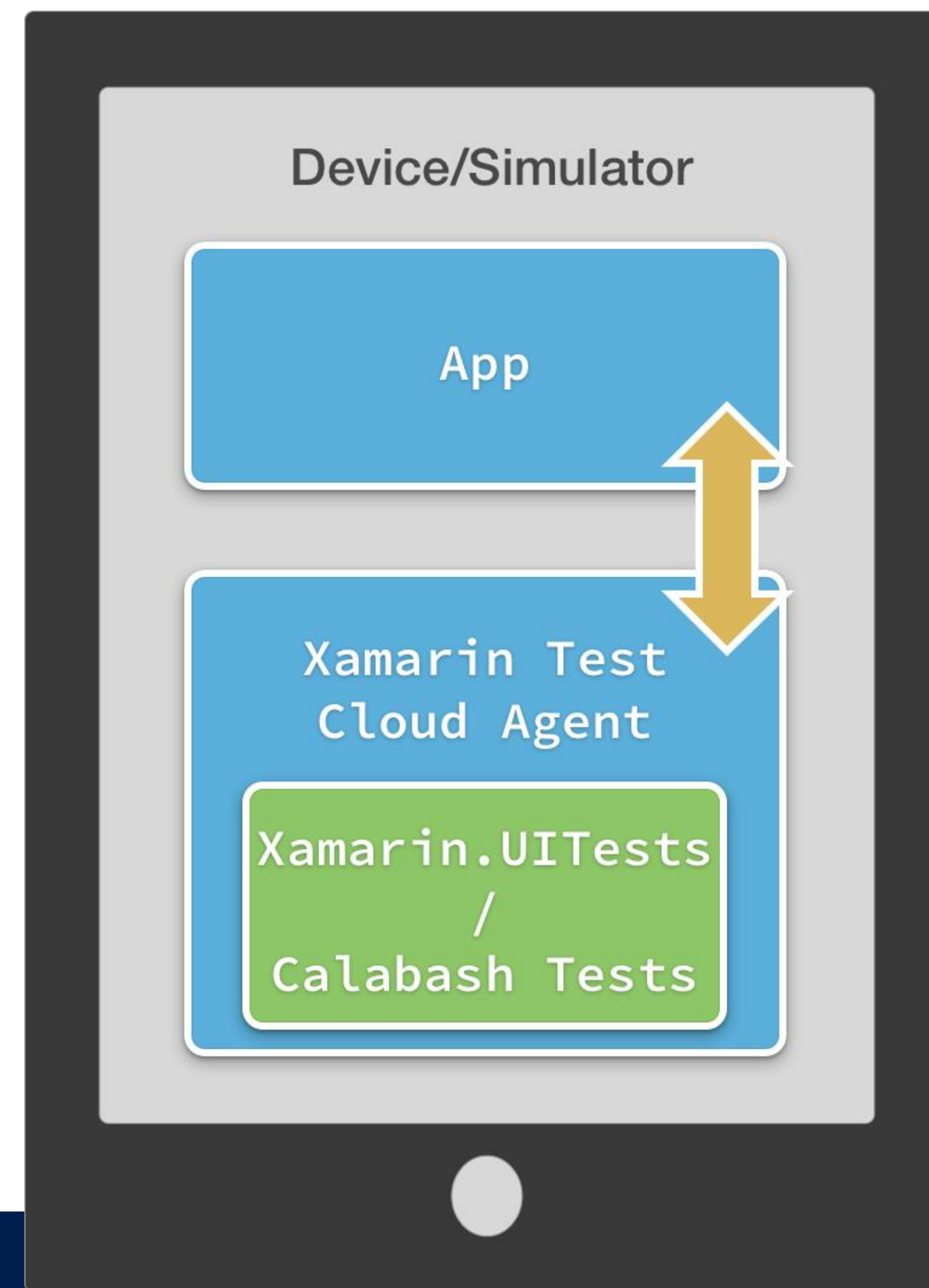
- Consente di registrare coded UI test utilizzando un device fisico (via USB) o un emulatore;
- Solo con Visual Studio Enterprise;
- Al momento supporta solo Android (per iOS Xamarin Test Recorder for Mac OSX)
- La registrazione necessita di Android 4.3 (API Level 23) o superiori ma il test generato può essere eseguito anche su versioni precedenti





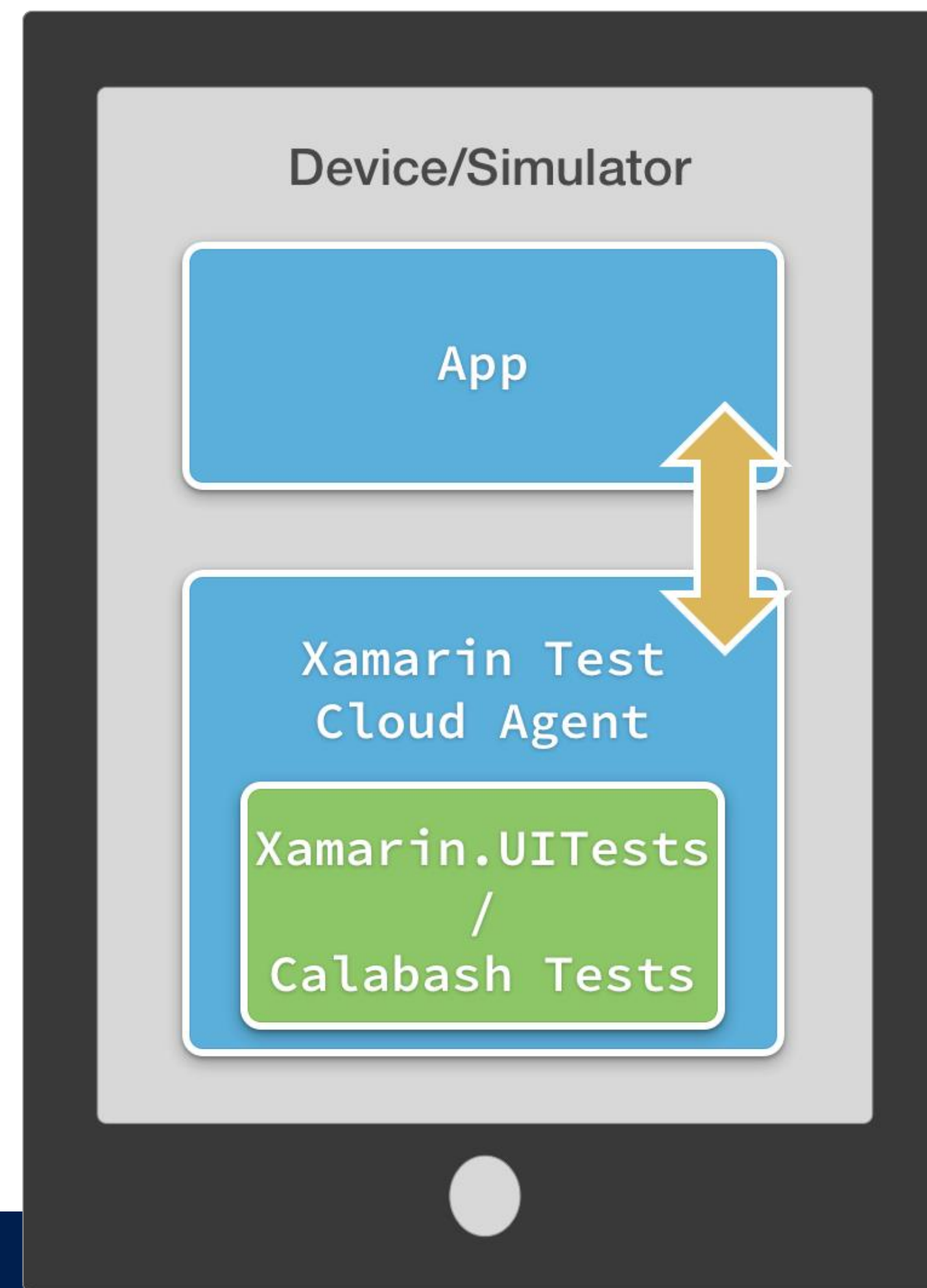
# Anatomia di un test

- Calabash e Xamarin.UITest da soli non sono in grado di interagire con l'interfaccia dell'applicazione (semplicemente perché non ne fanno parte)
- Hanno la necessità di una «libreria di automazione» che effettivamente si occupa di eseguire il test sul device e si occupa dell'interazione con la UI
- Xamarin Test Cloud Agent si occupa di «hostare» i test e di eseguirli nel device



# Xamarin Test Cloud Agent

- Xamarin Test Cloud Agent è installato da Test Cloud assieme all'applicazione
- Implementa un'architettura client/server per far in modo che la piattaforma Test Cloud possa effettivamente comandare l'esecuzione dei test e il recupero dei risultati
- La parte server è, di fatto un web server (molto semplice), che viene installato sul device ed è in ascolto per rispondere ai comandi (JSON over HTTP) inviati dal client
- Il client si trova all'interno della piattaforma Test Cloud

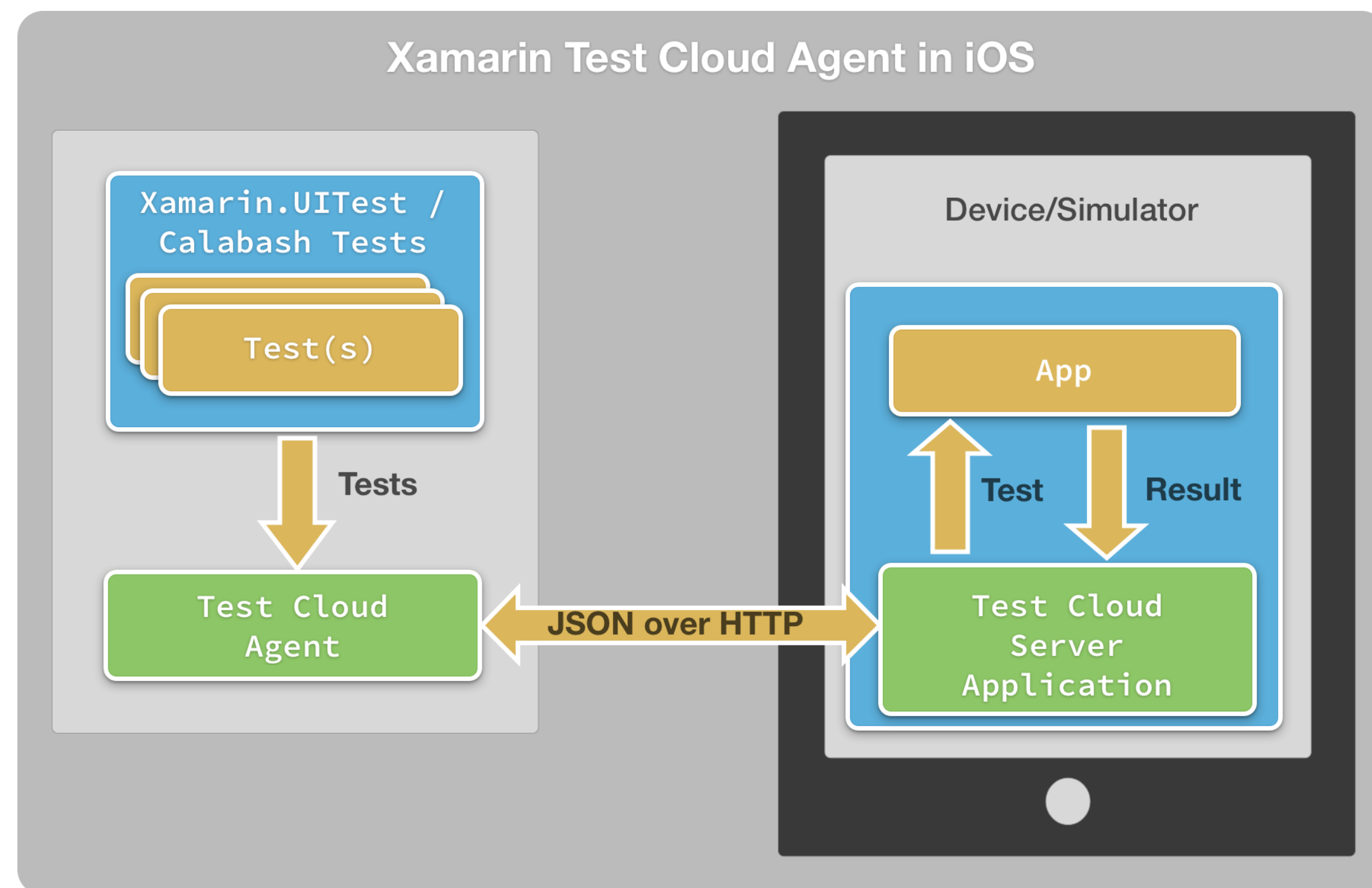




# Xamarin Test Cloud Agent su iOS

A causa delle limitazioni imposte dalla piattaforma Apple, il Test Cloud Agent è compilato assieme all'applicazione.

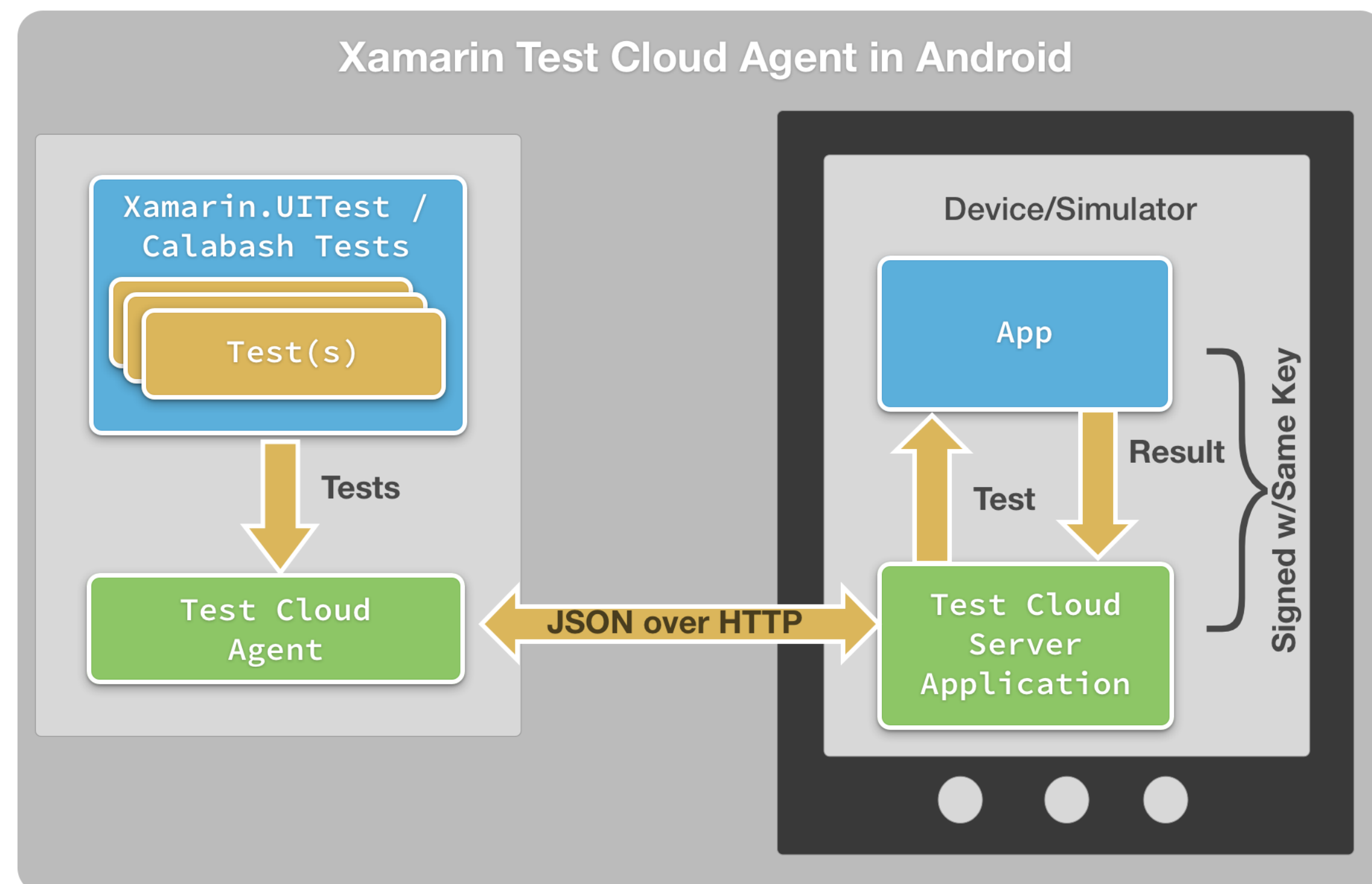
Viene utilizzato un package NuGet per referenziare l'agent nell'applicazione



# Xamarin Test Cloud Agent su Android

Test Cloud Agent è un'applicazione a se stante che viene distribuita con i permessi per instrumentare l'applicazione che stiamo testando.

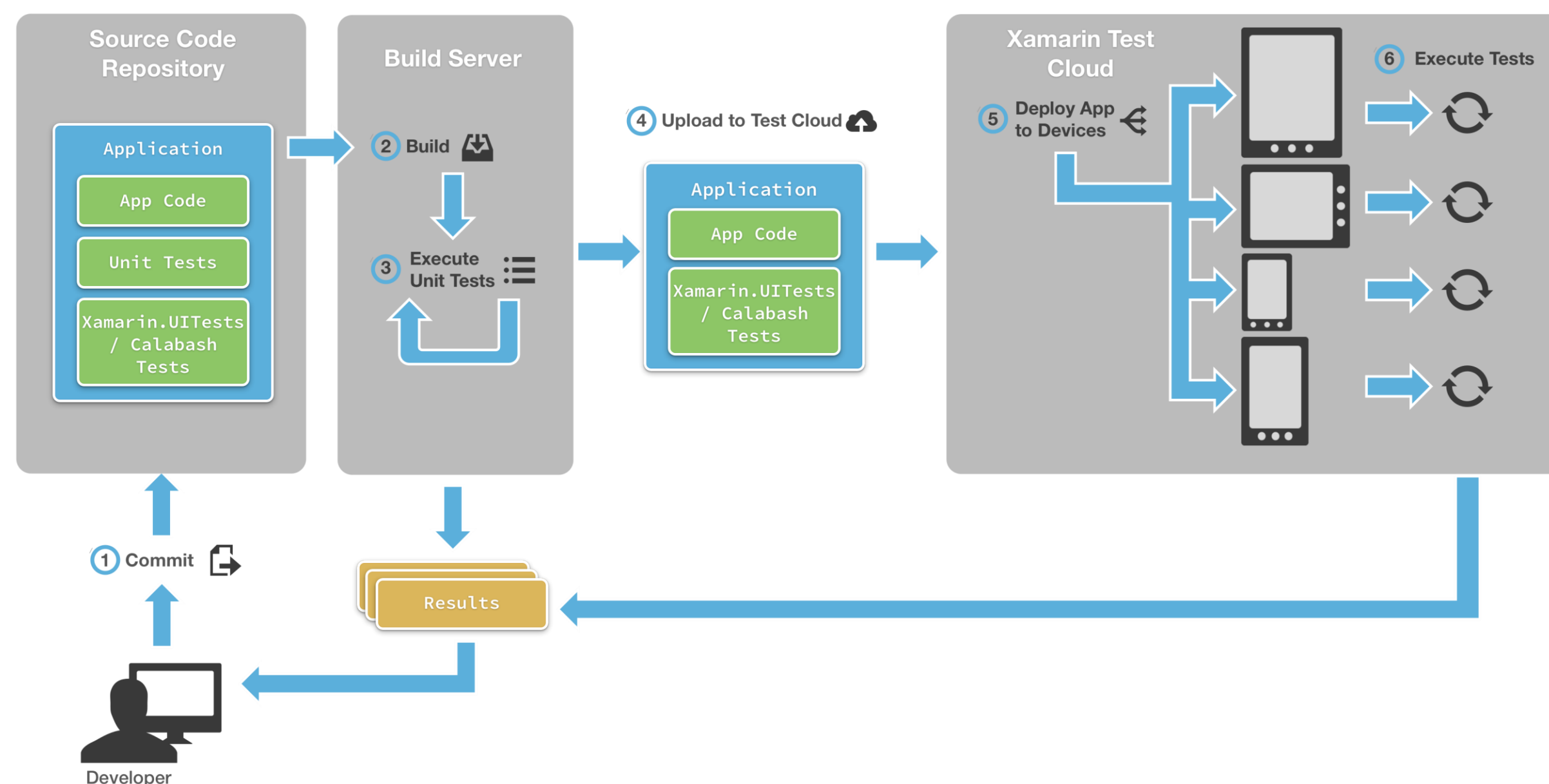
L'applicazione da testare e il Test Agent sono firmati con la stessa chiave in modo da poter tra loro comunicare.





# Xamarin Test Cloud Continuous Integration

Il meccanismo di pubblicazione dell'app e dei test può avvenire anche in un contesto di Continuous Integration (CI)



# Xamarin Test Cloud Continuous Integration

### Create new build definition

Select a template

**Build** Deployment

**Universal Windows Platform**  
Build Universal Windows Platform applications using Visual Studio. This template requires that Visual Studio and the Universal templates are installed on the build agent.

**Visual Studio**  
Build and run tests using Visual Studio. This template requires that Visual Studio be installed on the build agent.

**Xamarin.Android**  
Build an Android app and Xamarin.UITest assembly. Test with Xamarin Test Cloud.

**Xamarin.iOS**  
Build a Xamarin.iOS app and Xamarin.UITest assembly. Test with Xamarin Test Cloud. This template requires a Mac OS build agent.

**Xcode**  
Build and test an Xcode workspace. This template requires a Mac OS build agent.

**Empty**  
Start with a definition that has no steps.

Next > Cancel

Team Services / **Xamarin Test Cloud**

HOME CODE WORK **BUILD** TEST RELEASE

Definitions **Explorer**

Definitions / Xamarin Test Cloud CI | Builds

**Build** Options Repository Variables Triggers General Retention History

Save Queue build... Undo

+ Add build step...

- NuGet restore **\*\*/\*.sln**  
*NuGet Installer*
- Activate Xamarin license  
*Xamarin License*
- Build Xamarin.Android Project **\*\*/\*Droid\*.csproj**  
*Xamarin.Android*
- Build solution **\*\*/\*test\*.csproj**  
*MSBuild*
- Test \$(build.binariesdirectory)/\$(BuildConfiguration)/\*.apk with Xamarin.UITest in Xamarin Test Cloud**
- Signing and aligning APK file(s) **\$(build.binariesdirectory)/\$(BuildConfiguration)/\*.apk**  
*Android Signing*
- Deactivate Xamarin license  
*Xamarin License*
- Publish Artifact: drop  
*Publish Build Artifacts*

**Test \$(build.binariesdirectory)/\$(BuildConfiguration)/\*.apk with Xamarin.UITest in Xamarin Test Cloud**

App File \$(build.binariesdirectory)/\$(BuildConfiguration)/\*.apk

dSYM File (iOS only)

Team API Key

User Email mabonann@microsoft.com

Devices e37764e4

Series master

Test Assembly Directory \$(build.binariesdirectory)/\$(BuildConfiguration)/test-assembly

**Advanced**

Parallelization None

System Language English (United States)

test-cloud.exe Location **\*\*/packages/\*\*/tools/test-cloud.exe**

Optional Arguments

Publish results to TFS/Team Services ☒

**Control Options**

Enabled ☒

Continue on error ☐

Always run ☐

Timeout 0

More Information



# Ma quanto mi costi....

×

START

**\$99**/month  
or \$1,009.80 billed annually

1 concurrent device   1 device hour per day

✓ Unlimited apps  
✓ Forum support

Start a 30-day free trial

×

SCALE

**\$379**/month  
or \$3,865.80 billed annually

3 concurrent devices   5 device hours per day

✓ Unlimited apps  
✓ Email support

msdn subscriptions  
**-25%**

×

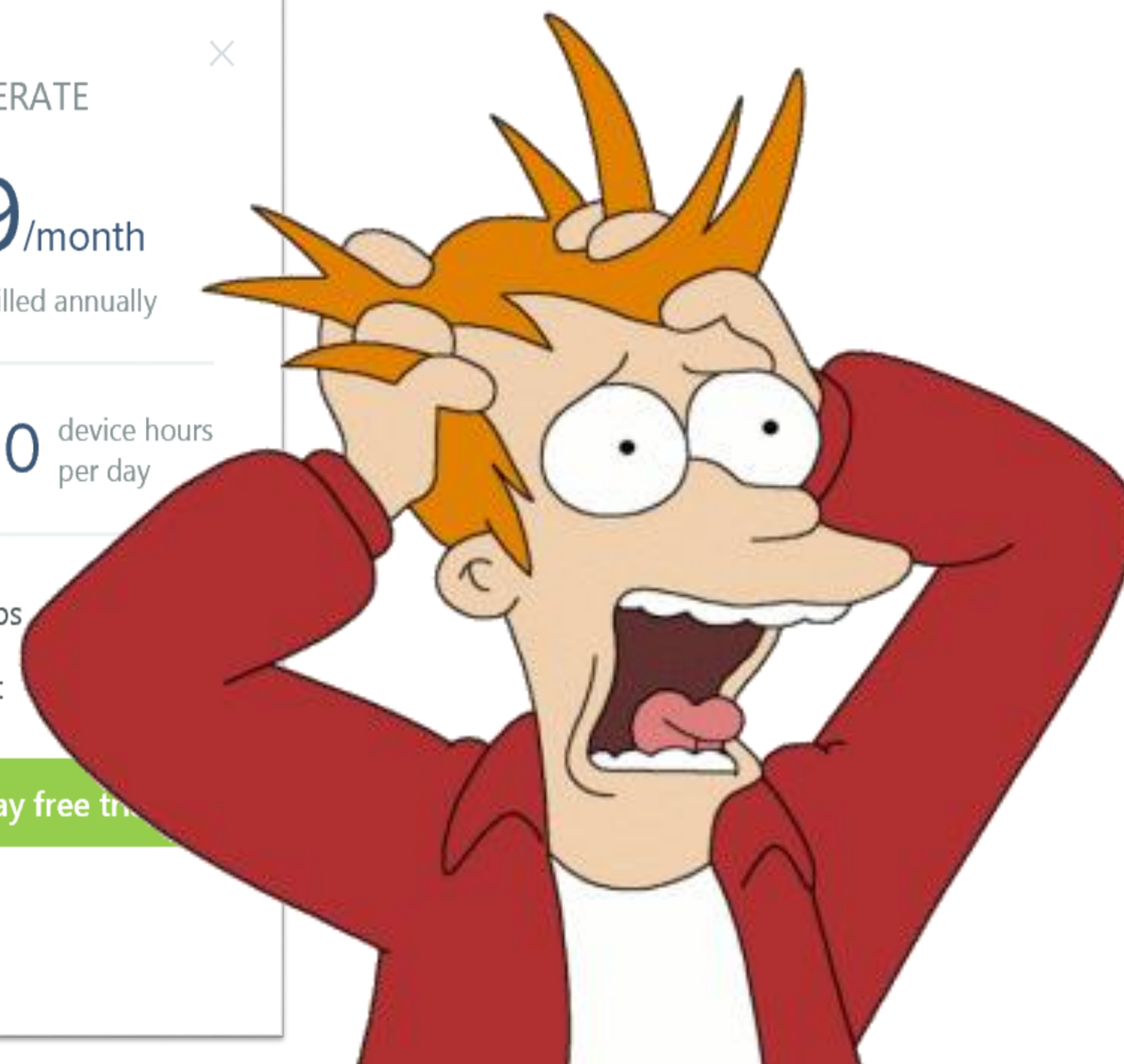
ACCELERATE

**\$799**/month  
or \$8,149.80 billed annually

5 concurrent devices   10 device hours per day

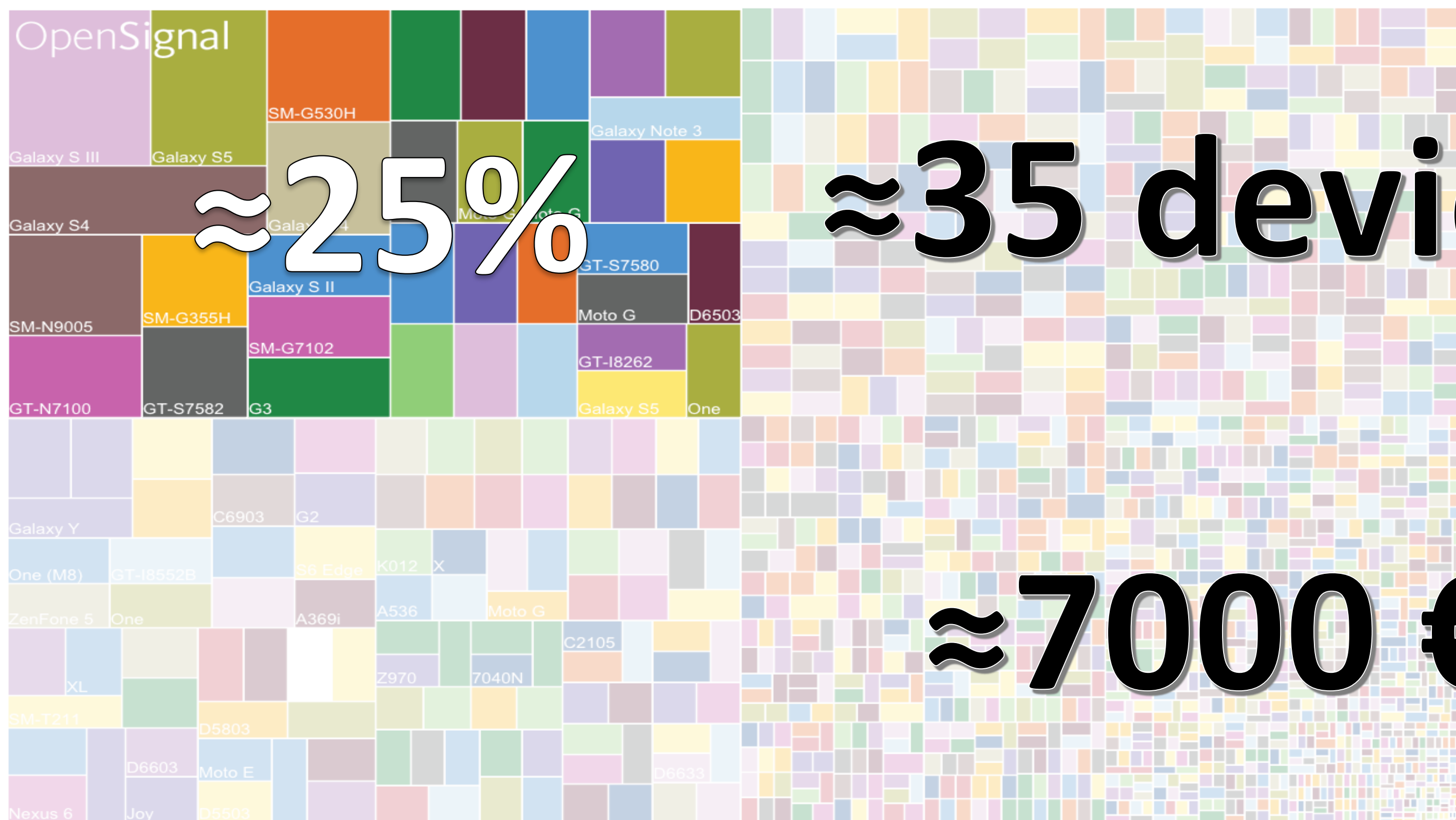
✓ Unlimited apps  
✓ Email support

Start a 30-day free trial





# Ma quanto mi costi....



# Take Away

- L'utilizzo di Xamarin Test Cloud va valutato in base ai risultati desiderati
- Xamarin Test Cloud va bene anche se non usate Xamarin (app native iOS o Android)
- Migliaia di device a disposizione + report su utilizzo di risorse



# Reference

## Xamarin Test Cloud Web Site

<https://testcloud.xamarin.com>

## Documentazione ufficiale

<https://developer.xamarin.com/guides/testcloud/>

## Test Recorder for Visual Studio

<https://developer.xamarin.com/guides/testcloud/testrecorder/visual-studio/>

## Devices

<https://testcloud.xamarin.com/devices>

## Demo

<https://github.com/massimobonanni/DDN-XamarinTestCloudDemo>

# Q&A

