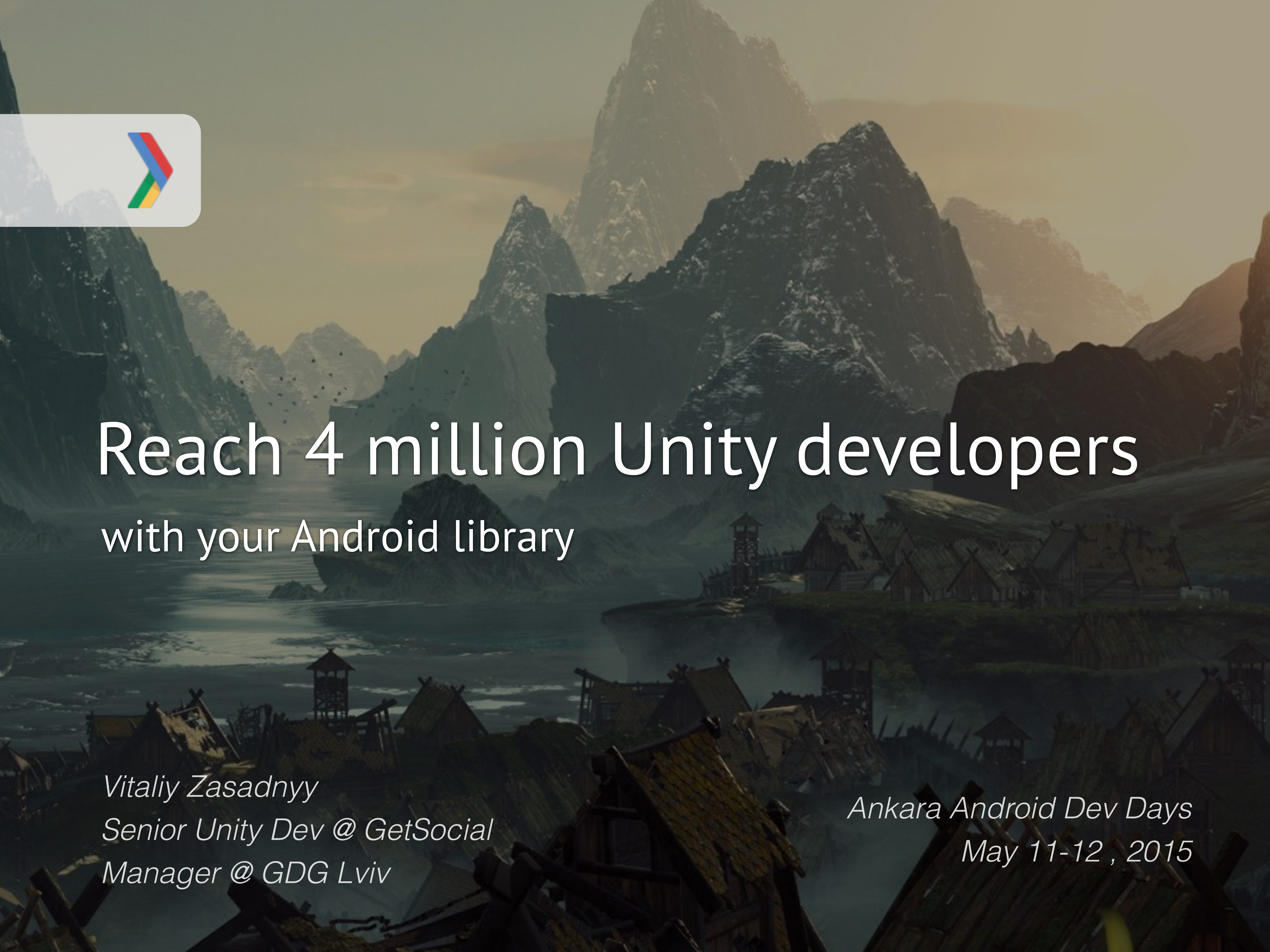


Reach 4 million Unity developers

with your Android library

Vitaliy Zasadnyy
Senior Unity Dev @ GetSocial
Manager @ GDG Lviv

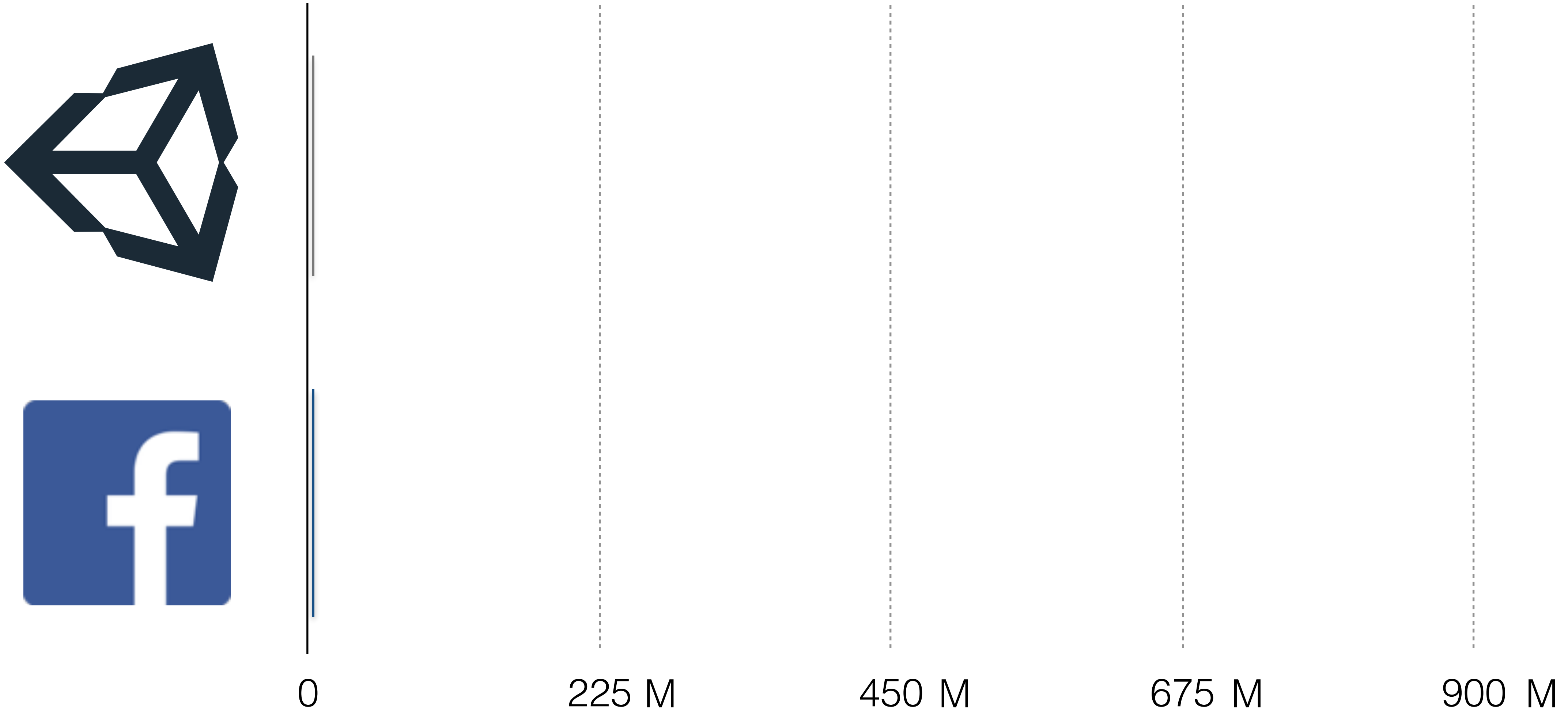
Ankara Android Dev Days
May 11-12 , 2015

Why Unity?

unity

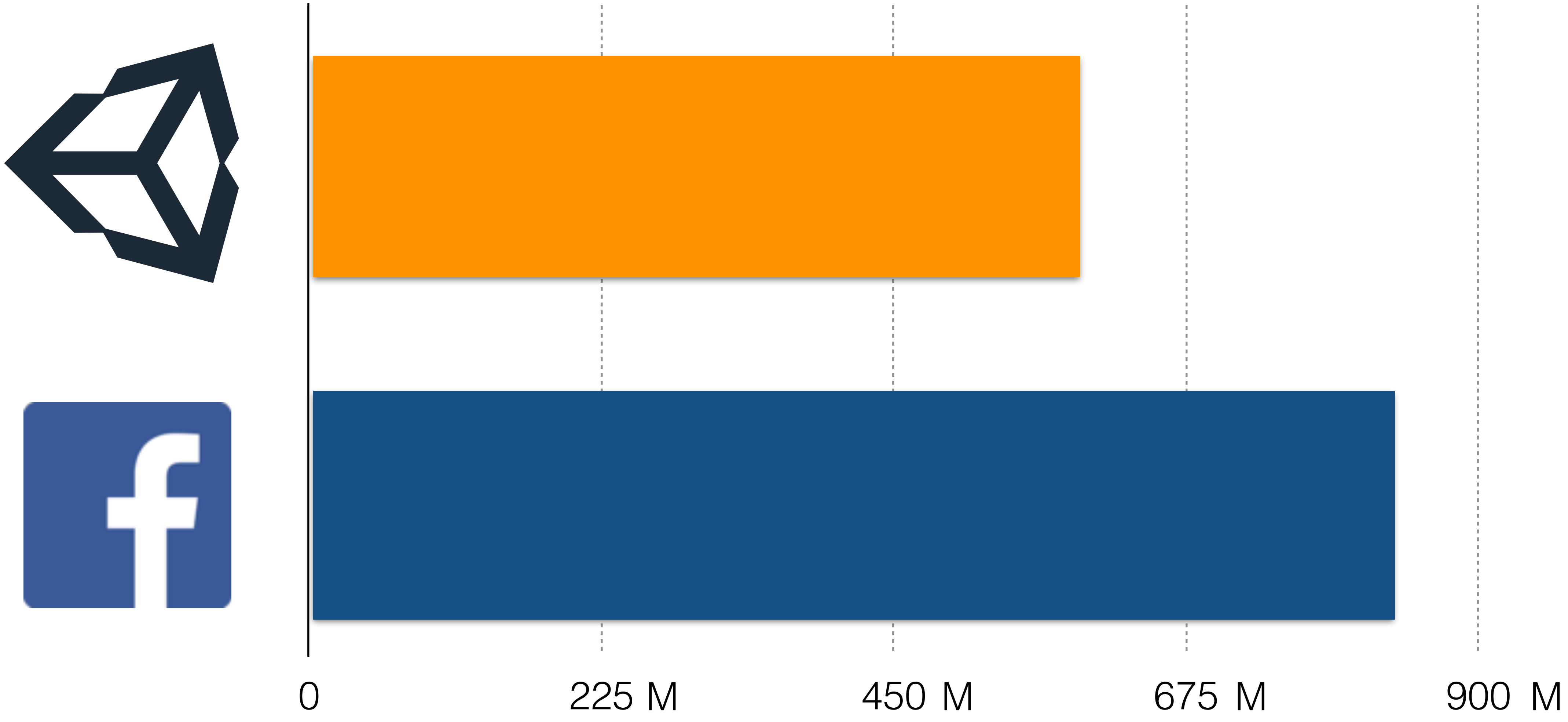
THE ROOM
TOTAL WAR BATTLES SHOGUN
HTC
SPLICE
Beat Sneak Bandit
Kingdoms of Camelot
Battle of the North

Daily Users



Source: Unreleased McKinsey report; Facebook http://newsroom.fb.com/company-info/

Daily Users



Source: *Unreleased McKinsey report; Facebook http://newsroom.fb.com/company-info/*

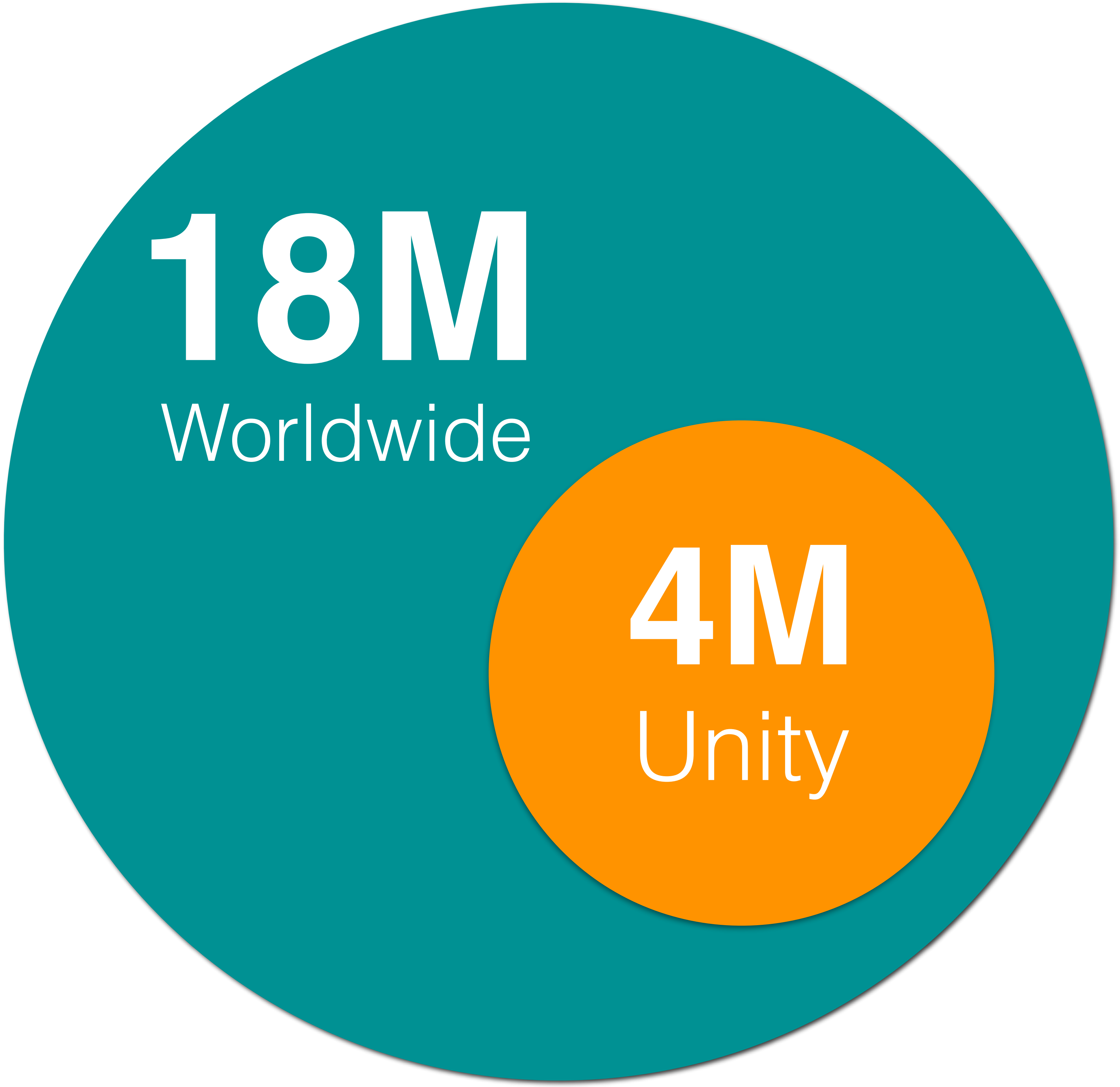
Unity Developers

4M

Unity

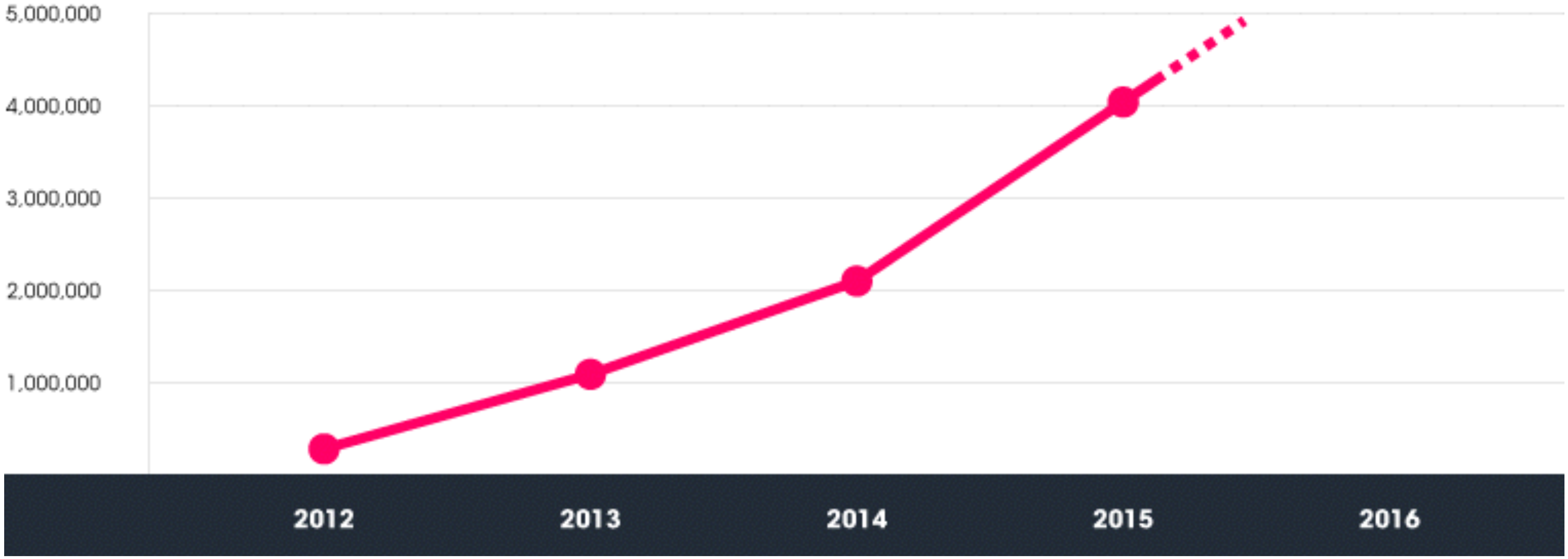
Source: Unity Technologies

Unity Developers



Source: Global Developer Population and Demographic Study 2014

Unity Developers

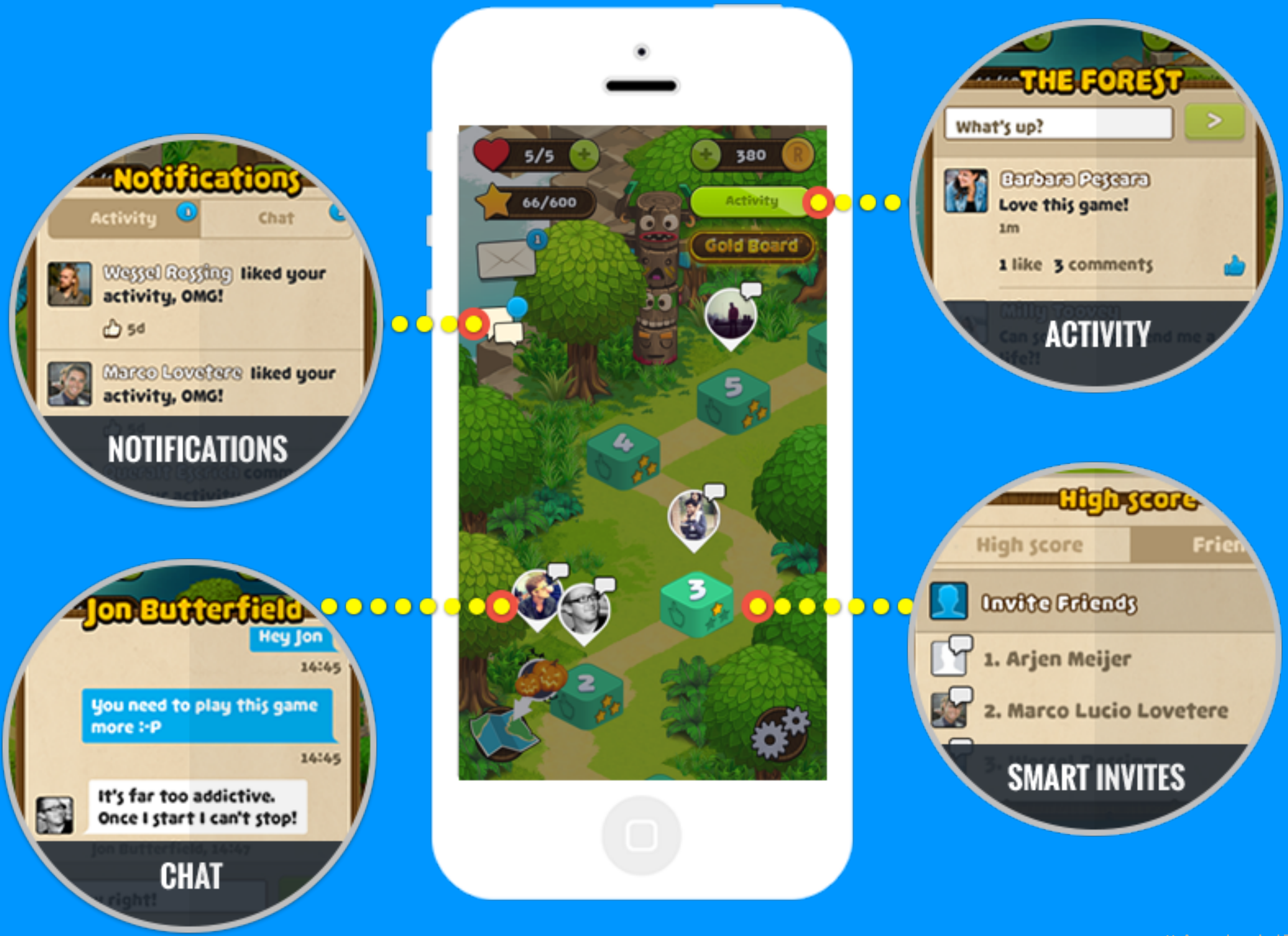


Source: Unity Technologies

Unity is Android

in game engine world

GetSocial



Wessel Rossing liked your activity, OMG!
5d

Marco Lovetere liked your activity, OMG!

What's up?

Barbara Pescara
Love this game!
1m
1 like 3 comments

Hey Jon
14:45

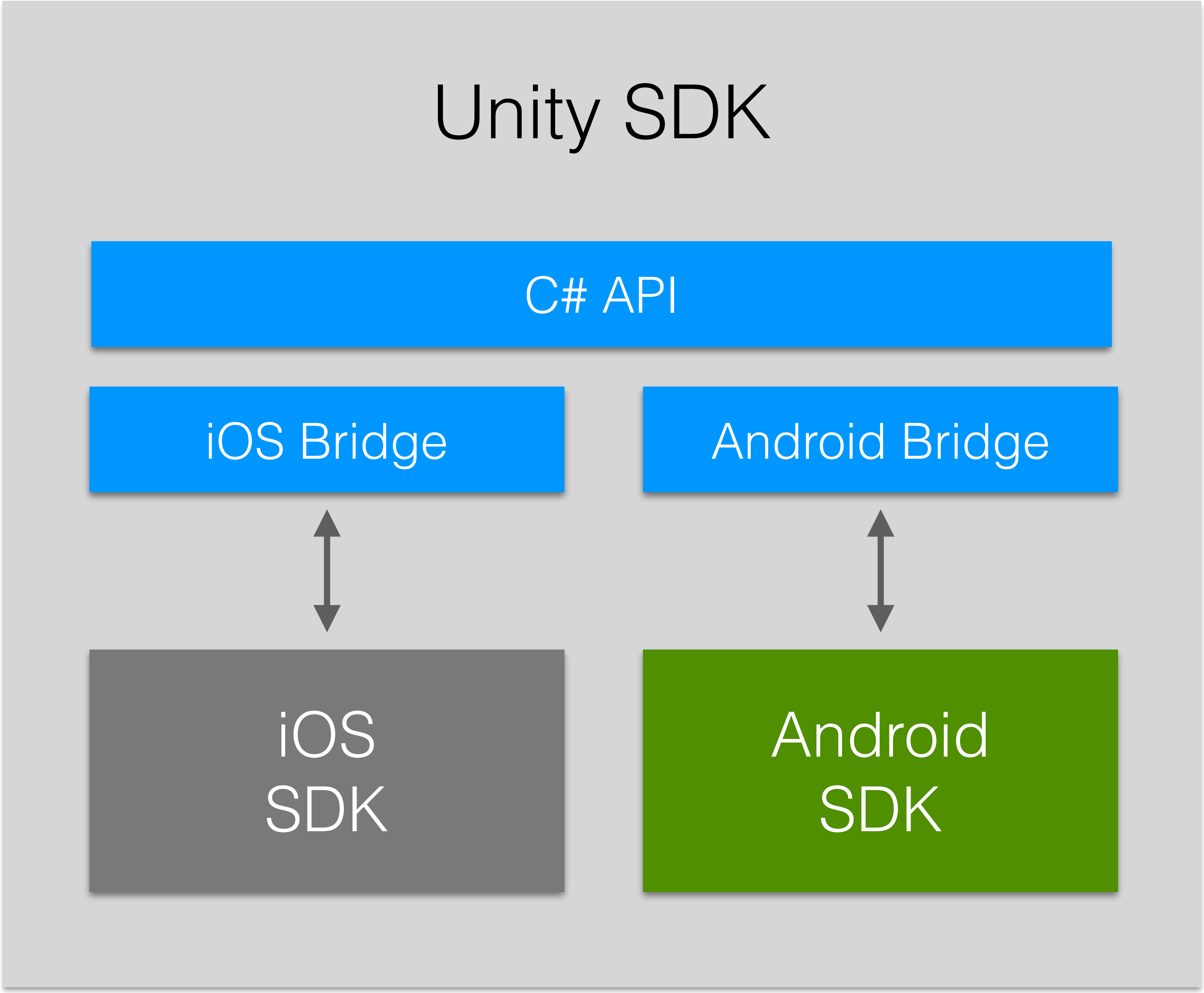
You need to play this game more :-P
14:45

It's far too addictive. Once I start I can't stop!

Invite Friends
1. Arjen Meijer
2. Marco Lucio Lovetere

#AndroidDevDays

GetSocial Unity SDK Architecture



Unity SDK

C# API

iOS Bridge

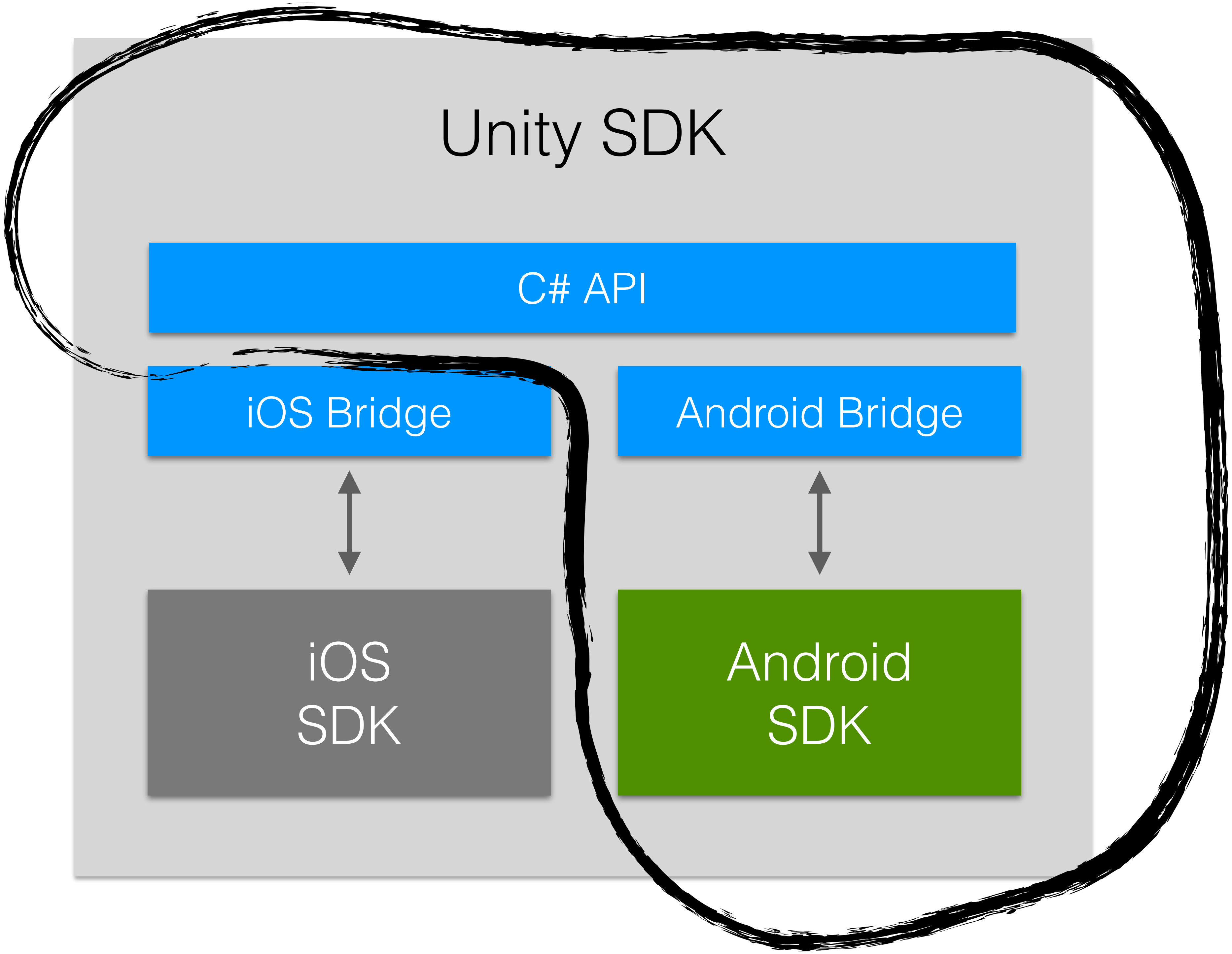
Android Bridge

iOS SDK

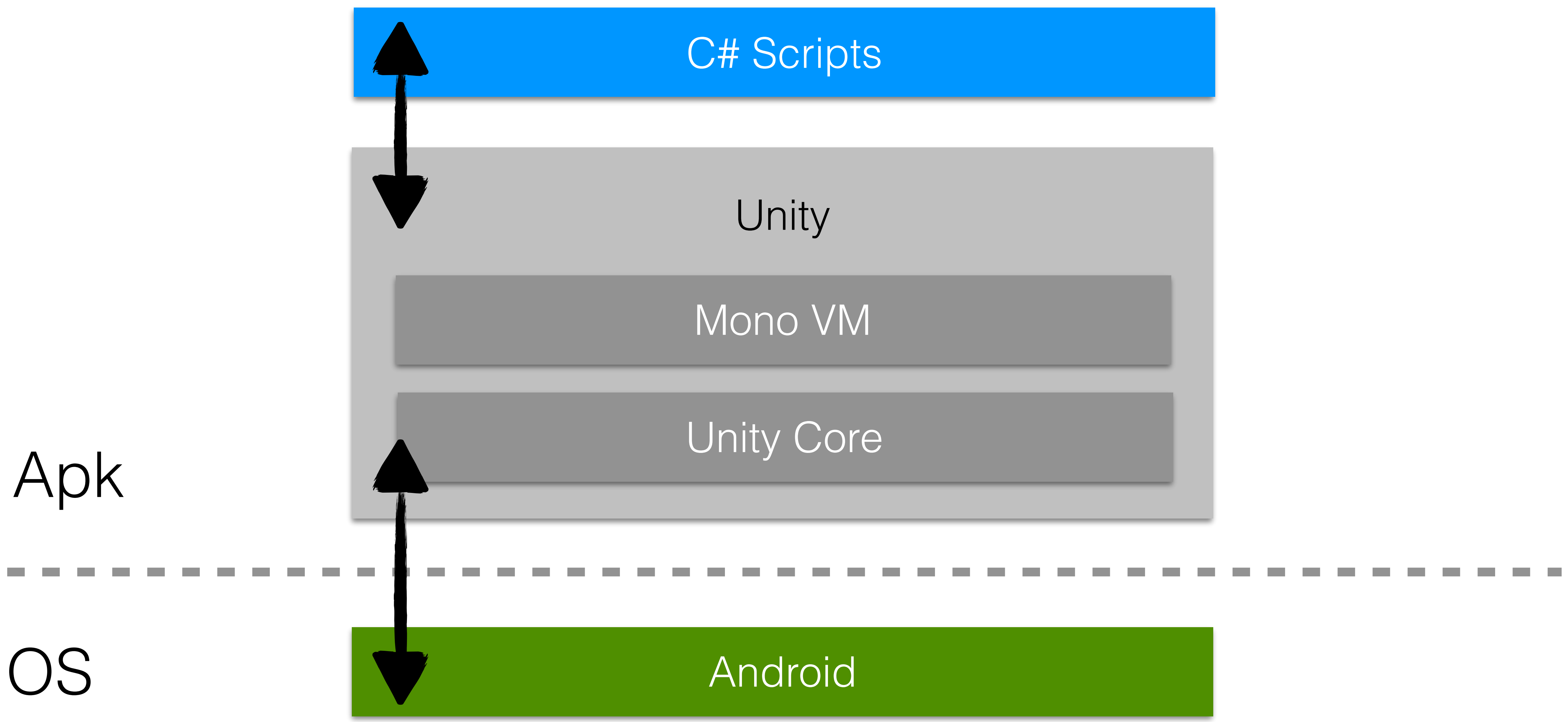
Android SDK

🟩 Java ⬛ Objective C 🟦 C#

During Android Build

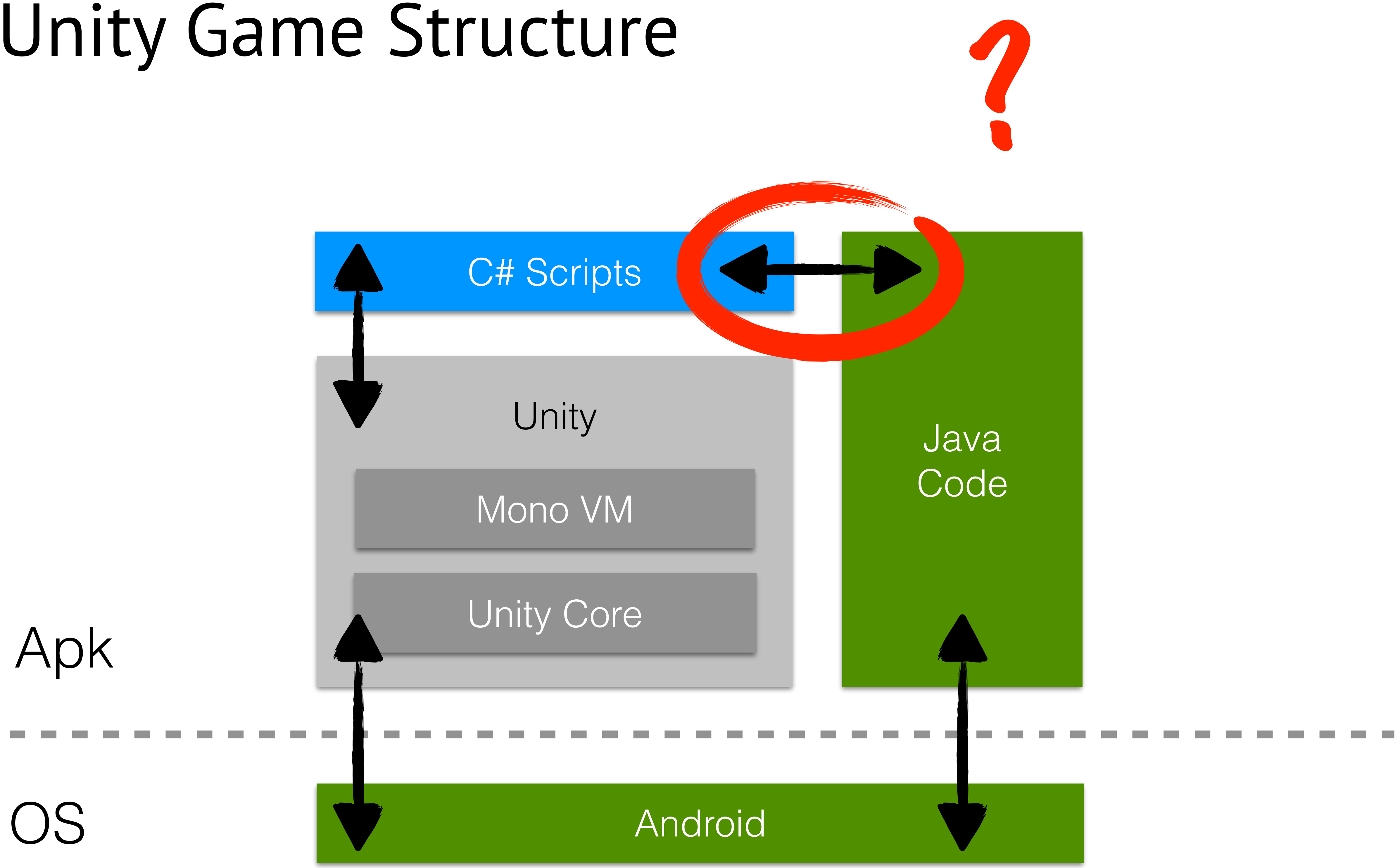


Unity Game Structure



#AndroidDevDays

Unity Game Structure



Provided Wrappers

Android / Java	Unity / C#
primitive types	primitive types
java.lang.String	string
java.lang.Object	AndroidJavaObject
java.lang.Class	AndroidJavaClass
java.lang.Runnable	AndroidJavaRunnable
java.lang.Proxy	AndroidJavaProxy

Provided Wrappers

Android / Java	Unity / C#
primitive types	primitive types
java.lang.String	string
java.lang.Object	AndroidJavaObject
java.lang.Class	AndroidJavaClass
java.lang.Runnable	AndroidJavaRunnable
java.lang.Proxy	AndroidJavaProxy

Two-way communication

C# ↔ Java

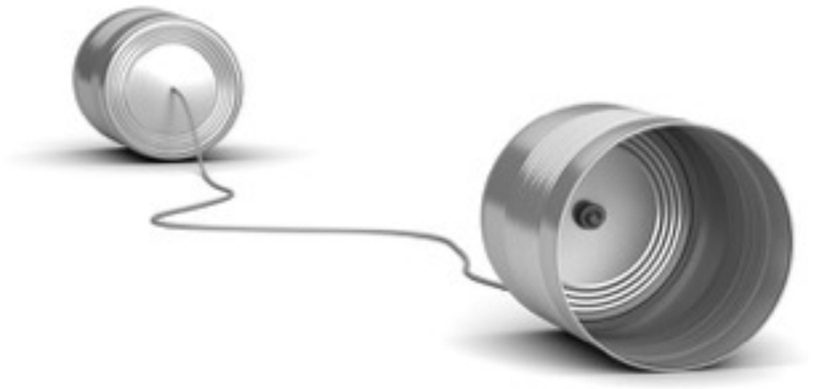
Two Way Communication

C# → Java

- AndroidJavaObject
- Pure JNI (Java Native Interface)

Java → C#

- Unity Messaging
- AndroidJavaProxy



Sample Java class

Java

```
public class Speaker {
  private String name;

  public Speaker(String name) { … }

  public static Speaker copy(Speaker speaker) { … }
  public String askQuestion(String question) { … }
}
```

AndroidJavaObject

- Wrapper around `java.lang.Object`

C#

```
var jSpeaker = new AndroidJavaObject(
                        “com.zasadnyy.Speaker”, “Vitaliy”);

var jClone = jSpeaker.CallStatic<AndroidJavaObject>(
                                          “copy”, jSpeaker);

var answer = jSpeaker.Call<string>(“askQuestion”, “HDYD?”);
```

Pure JNI

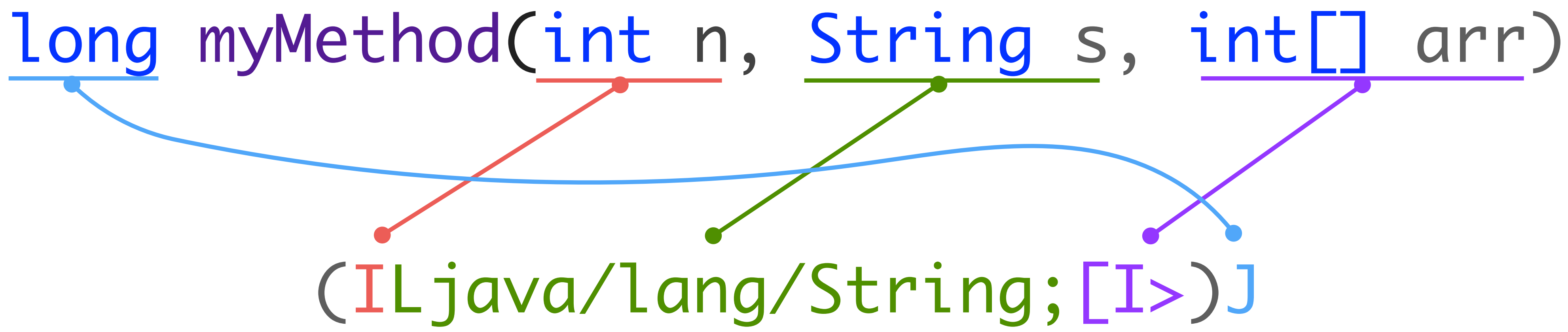
C#

```
// var jSpeaker = new AndroidJavaObject(
//                        “com.zasadnyy.Speaker”, “Vitaliy”);

IntPtr classID = jni.FindClass("com.zasadnyy.Speaker");

IntPtr constructorID = jni.GetMethodID(classID,
                          "<init>", "(Ljava/lang/String;)V");

IntPtr javaString = jni.NewStringUTF("Vitaliy");

IntPtr objectID = jni.NewObject(classID,
                                  constructorID, javaString);
```

JNI Cheatsheet

B	byte	S	short
C	char	V	void
F	float	Z	boolean
I	int	Lclass;	class
J	long	[type>	array of type

JNI Cheatsheet / Example



```
long myMethod(int n, String s, int[] arr)
```

```
(ILjava/lang/String;[I>)J
```

B	byte	S	short
C	char	V	void
F	float	Z	boolean
I	int	Lclass;	class
J	long	[type>	array of type

JNI is slooooow!

Respect 16ms rule

Conclusion

Use `AndroidJavaObject`:

- Less boilerplate
- Less error prone
- Faster (caching)

But:

- JNI is more flexible
- `AndroidJavaObject` has bugs

AndroidJavaObject Bugs *

C#

```
var result = javaObject.Call<TResult>(TParam param);
```

- we can’t pass `string[]` as `TParam` (`NoSuchMethodException`)

- we can’t return null objects from Java (`Exception: JNI: Init'd AndroidJavaObject with null ptr!`)

** Tested on Unity 5.0.1f1*

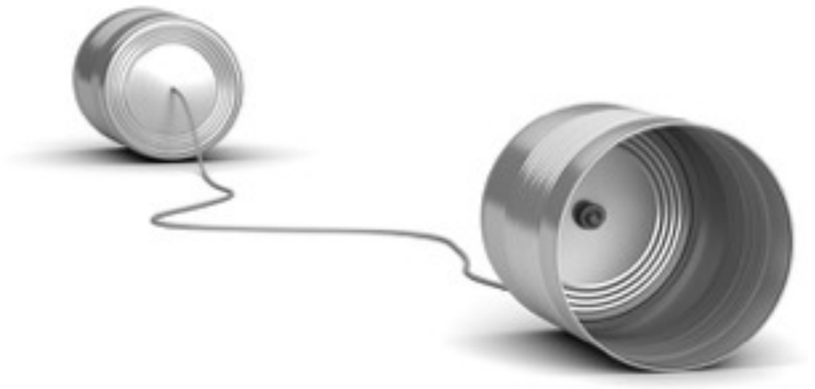
Two Way Communication

C#

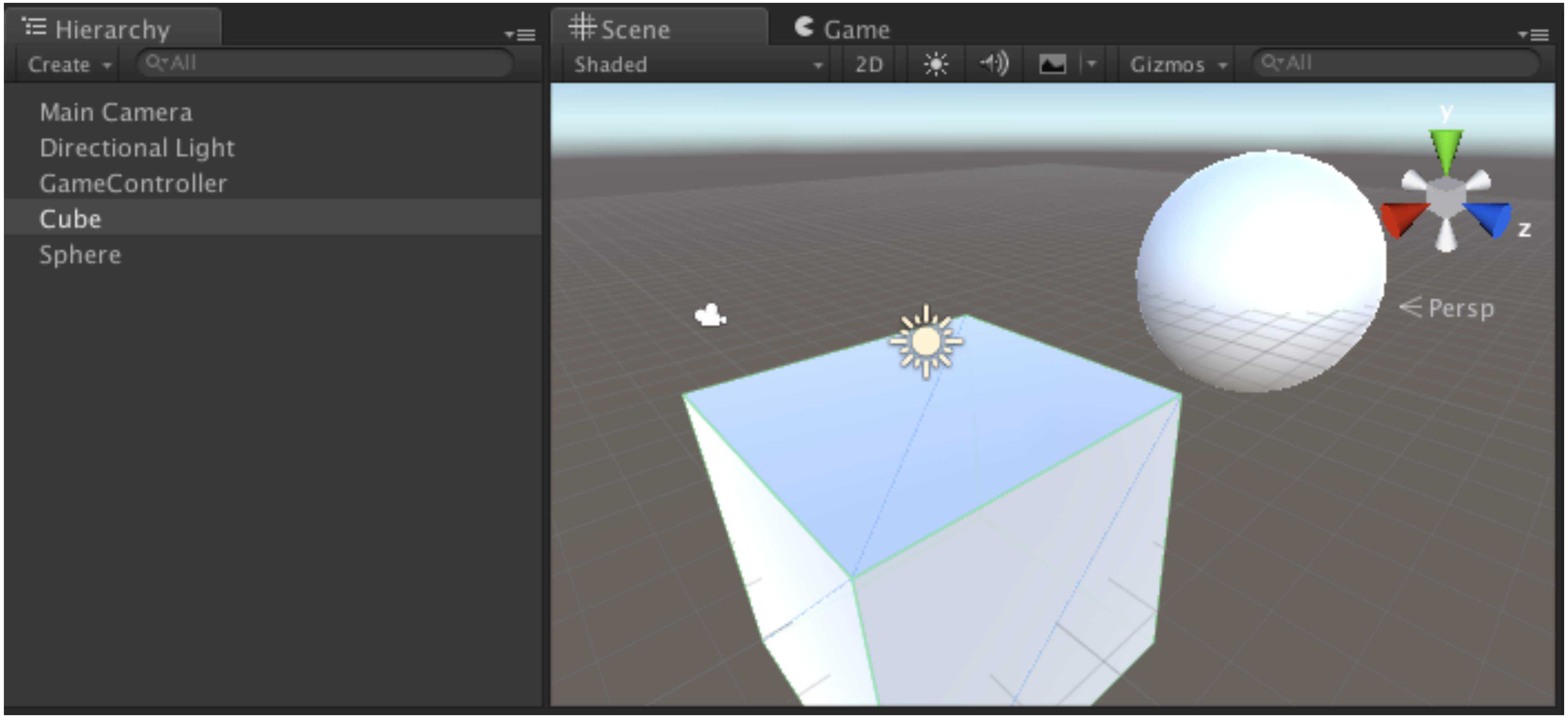
- AndroidJavaObject
- Pure JNI (Java Native Interface)

Java → C#

- Unity Messaging
- AndroidJavaProxy



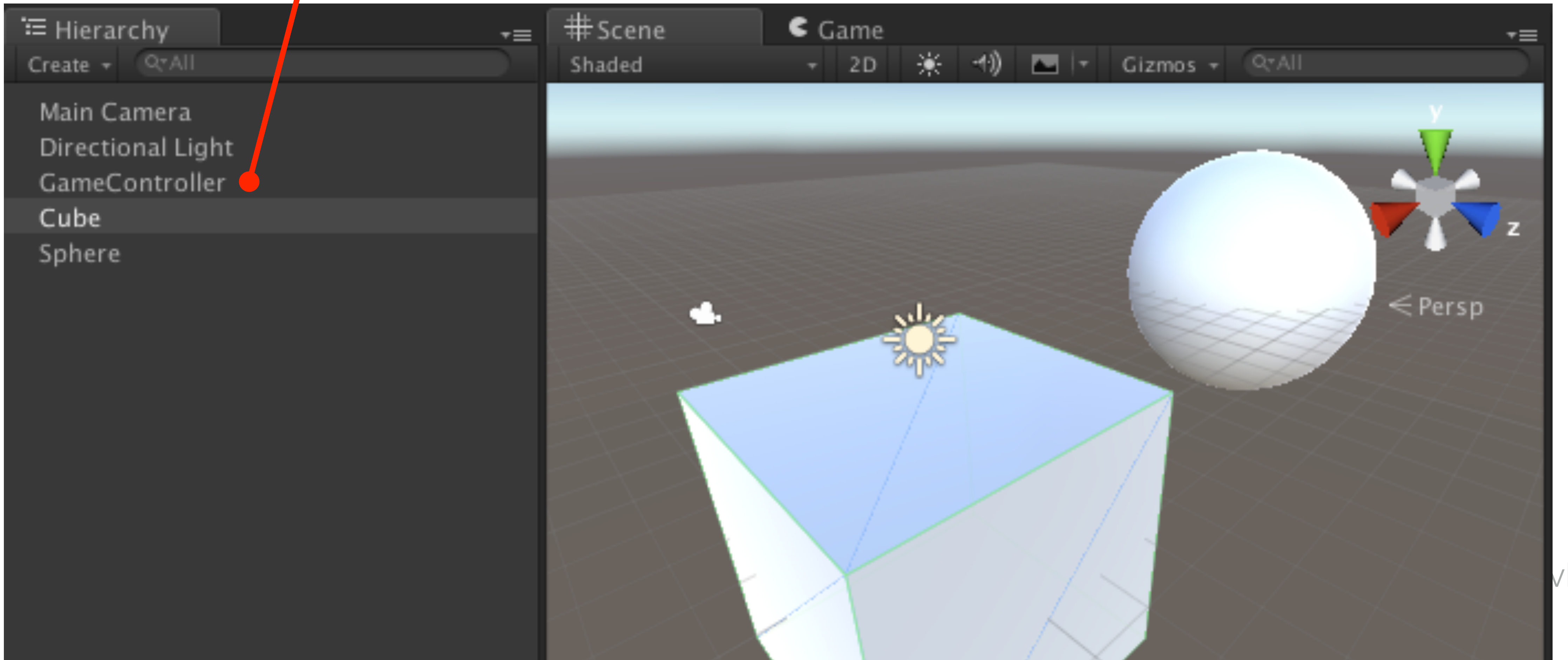
Unity Messaging



Unity Messaging

Java

```
UnityPlayer.UnitySendMessage(
        "GameController", "OnPushMessage", "New gift");
```



Unity Messaging

Pros:

- Quick to implement

Cons:

- Always 1 frame late
- Only string parameters
- Requires `GameObject`
- Requires `unity-classes.jar` in classpath

AndroidJavaProxy

- Wrapper around `java.lang.Proxy`

Java

```java
public interface IPushMessageListener {
    void onPushMessage(String message);
}
```

AndroidJavaProxy

Java

```java
public interface IPushMessageListener {
    void onPushMessage(String message);
}
```

C#

```csharp
public class PushMessageListener : AndroidJavaProxy {
    public PushMessageListener() :
          base("com.zasadnyy.IPushMessageListener") { … }

    void onPushMessage(string message) { … }
}
```

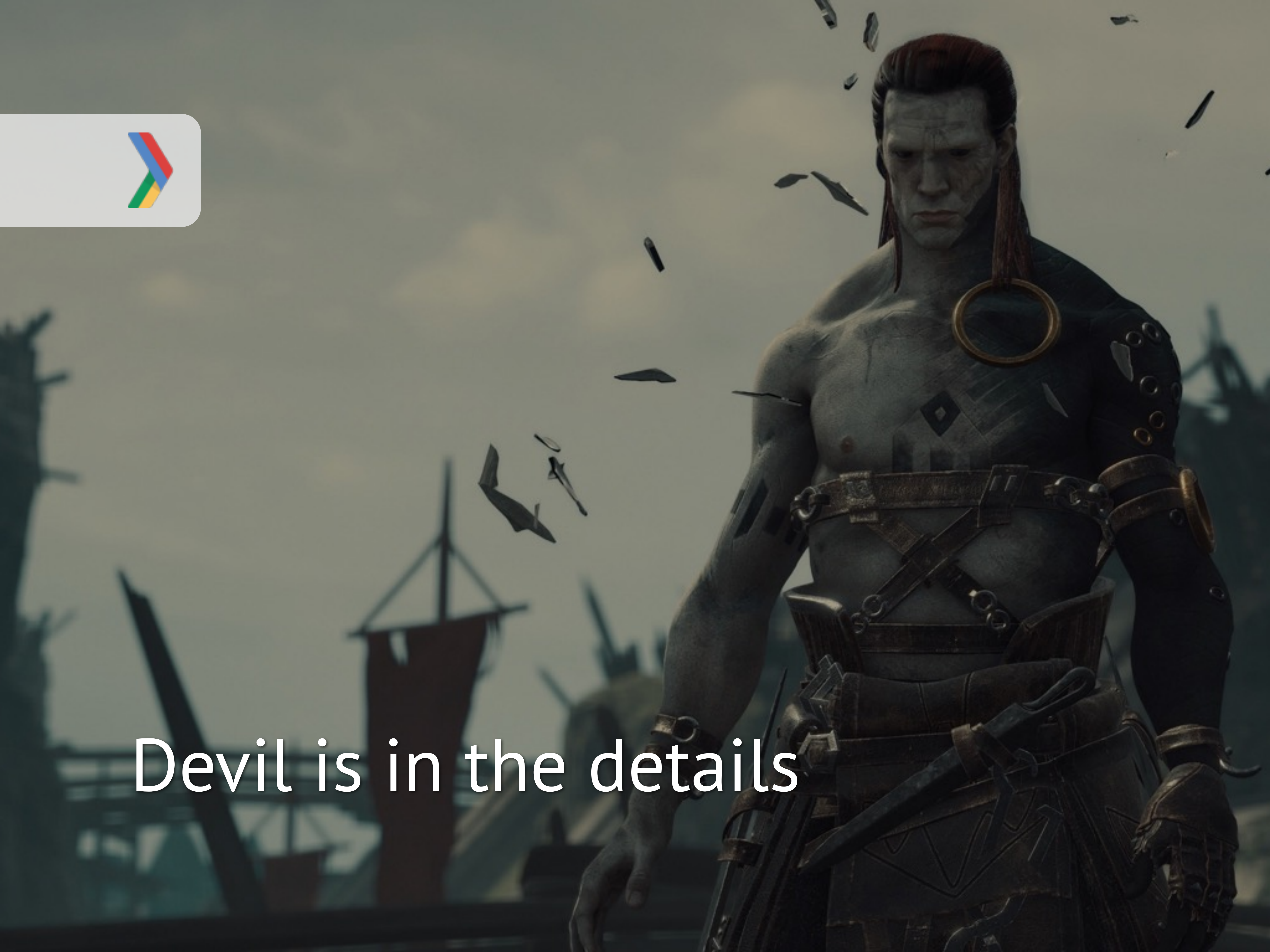
AndroidJavaProxy

Pros:

- Immediate result
- No type restriction on parameters
- No extra dependencies on Java side

Cons:

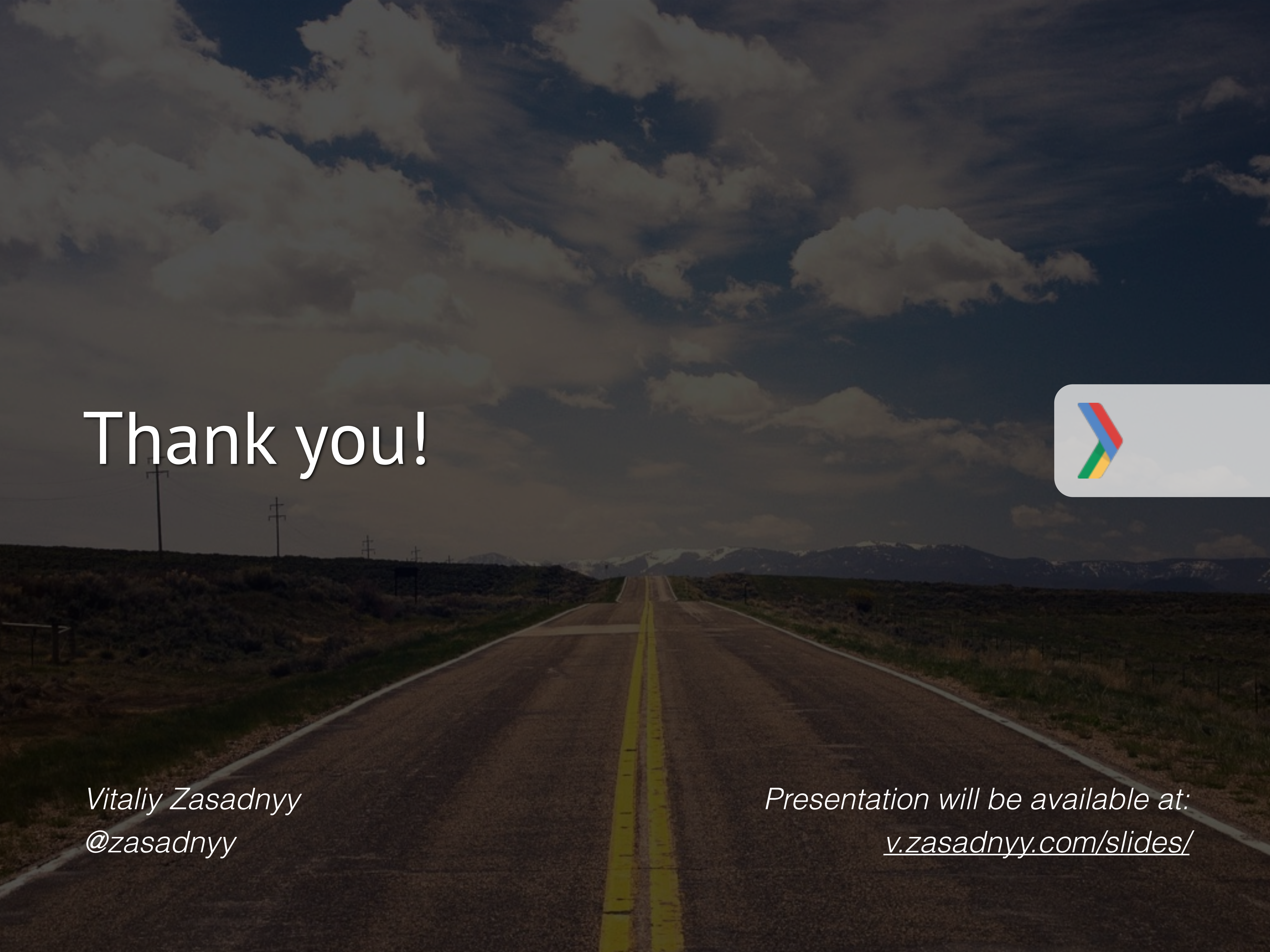
- Bug with null string parameter
- Local reference table is limited to 512 entries



Devil is in the details

Details

- Unity is single threaded
- Unity main thread != Android main thread
- Implement `equals()` in all `AndroidJavaProxy`'ies
- Differences between `GameObject` and `Activity` lifecycle
- Keep calm and `signal 11`



Thank you!

Vitaliy Zasadnyy
@zasadnyy

Presentation will be available at:
v.zasadnyy.com/slides/