AN INTRODUCTION TO MOBILE DEVELOPMENT

To understand the mobile development scene, we must first know who are the present industry leaders and who are the new contenders for 2013/14.

In Q4 2012, the most used mobile operating systems were **Android** (with the 68.4% of market share) **and iOS** (with the 19.4% of market share). Both together grabbed the 92% of the global Smartphone shipments. These results make Java (Android) and Objective-C (iOS) two of the most used programming languages for mobile development, but that's only when we are talking about apps built using the Android SDK or the iOS SDK.

** Please note that HTML5 web apps running in a mobile browser cannot be downloaded from the Android market or App Store. This kind of apps are not strictly mobile native, instead they are web apps optimized to be executed in a mobile browser like Chrome or Firefox. These web apps can be built using frameworks like jQuery Mobile, M-Project, Sencha Touch, Zepto, DHTMLX, LimeJS, jQTouch, Kendo UI or Dojo Mobile.*

The other mobile operating systems already in the market are Windows Phone and Blackberry 10, but their market share is still very small. In the following months we can expect the following releases:

- Firefox OS from Mozilla
- Ubuntu Mobile from Ubuntu
- Tizen from Linux Foundation, Samsung, Intel, Tizen Community

You can read more about mobile operating systems here and here

THE PROS AND CONS OF NATIVE CODING

The benefits of building native apps are all related to maximum flexibility and performance. All programmers know that using the lowest level programming languages give you the possibility to tweak the apps, customize them and optimize the code to the maximum level. This is possible because you are working almost directly with the hardware; there are no unnecessary layers of complexity in between your code and the device.

The apps that rely heavily on 3D graphics and need to process a lot of data or make intensive use of the CPU are the best candidates for the native code approach.

But flexibility and performance have a price and that is complexity and duplicate codebases. If you want your native app to be compatible with Android and iOS, then you need to have two different codebases, which means you have to work double. Java and Objective-C are two very different programming languages, they require completely different programming environments and maintaining the code (refactoring) can become expensive for individuals or organizations that don't have a team of developers working full time in mobile apps.

WHAT ARE THE ALTERNATIVES?

Luckily or not, there are several companies with products that promise to solve the hassle of having two or more different codebases for the same app. These products allow developers to use common APIs to build the apps and then “export” them to native Java (Android), Objective-C code (iOS) or even other platforms like Kindle Fire, Blackberry, Windows Phone, etc:

- Appcelerator and its free JavaScript driven Titanium SDK and Cloud services. They advertise their success with more than **50.000 apps deployed** in the market, **419,000 developers** and customers like eBay, Merck, Mitsubishi Electric, NBC, and PayPal. Showcase of apps.
  After having tested the platform for several months, I can say that the product certainly works and the learning curve is fairly smooth, which makes the development process quite fun. Although not everything is as marvellous as they say.

- Marmalade is one of the best game development frameworks out there. Whether you choose to code natively (C++) or take a hybrid (HTML5-native) approach, with Marmalade you can deploy to iOS, Android, BlackBerry, Windows and Mac, as well as selected Smart TVs and (shortly) set top box platforms as well. Clients: **Electronic Arts, Nokia, Apple, Konami, Google, Square Enix, nVidia, Samsung**, etc. App showcase.

- Adobe PhoneGap is a free and open source framework that allows you to create mobile apps using standardized web APIs like HTML5, CSS and JavaScript. They have **400,000 developers** and contributors from IBM, RIM and Microsoft. App showcase.

- Corona SDK is a well known framework for mobile game developers. From a single codebase you can deploy to iOS, Android, Kindle Fire and NOOK. App showcase.

- Xamarin is another company/product to develop apps for iOS, Android and Mac using C#. They have **230,245 developers** and customers like 3M, Microsoft, VMWare, Accenture, Cisco, AT&T, Aol, Monster and HP. App showcase.

- Shiva3D and Unity3D are options more focussed in high level multiplatform 3D games. Shiva showcase. Unity showcase.

- MoSync SDK is a less known option that makes it easy to build and compile apps for up to nine different platforms at once, using C/C++ or HTML5/JavaScript, or a combination of both. App showcase.

- Biznessapps and gamesalad are fast and easy solutions for small business and simple games with low requirements of custom programming and design. These solutions are **not recommended** for the vast majority of projects due to the limitations of the editors used to build the apps and their lack of flexibility.

- For an extended comparison chart of mobile frameworks, please visit this link, this wiki page or this other one.

WHAT ARE THE CHALLENGES FOR BOTH, NATIVE AND NON-NATIVE APPS?

Programming mobile apps is harder than building desktop or web apps. The platform is very new and is getting more and more fragmented, which doesn't help developers.
In example, Android has hundreds of different devices in the market. Some of them have a 3.4'' screen, others 7'' or 10''. Some of them run Android 2.1 and others run Android 4.3. Some of them have 3G, GPS and a 300dpi (dots per inch / pixel density), HD Ready or Full HD screen. The possible combinations are really intimidating and the truth is that every single user expects that your app will work flawlessly in his device.

Some important aspects to bear in mind are:

- Deciding which tool/framework/approach to use is a critical decision that should be carefully thought before starting any development work
- Performance in old devices… the bar has to be set at some point.
- Responding quickly and successfully to complaints/bugs/feature requests in the Android market and App store. Remember about having duplicate codebases…
- Finding the right marketing approach and value for an app (freemium scheme or ads based). Some marketing strategies require a lot of customization in the code. Make sure to plan everything in advance.