Paolo Forgia

SSSE  SIG

Manuale tecnico

Indice

[1 Requisiti 2](#_Toc113382619)

[2 Analisi tecnologie 2](#_Toc113382620)

[2.1 App ibrida vs nativa 2](#_Toc113382621)

[2.1.1 Nativa 2](#_Toc113382622)

[2.1.2 Ibrida 2](#_Toc113382623)

[2.2 Analisi tecnologie 2](#_Toc113382624)

[2.2.1 Xamarin 2](#_Toc113382625)

[2.2.2 Flutter 2](#_Toc113382626)

[2.2.3 Ionic 2](#_Toc113382627)

[2.2.4 Cordova 3](#_Toc113382628)

[2.2.5 React Native 3](#_Toc113382629)

[2.3 Comparazione 3](#_Toc113382630)

[3 Analisi approfondita 3](#_Toc113382631)

[3.1 Xamarin 4](#_Toc113382632)

[3.1.1 Sviluppo 4](#_Toc113382633)

[3.1.2 Performance 4](#_Toc113382634)

[3.1.3 Orientamento dispositivo 4](#_Toc113382635)

[3.1.4 Dimensioni dispositivo 4](#_Toc113382636)

[3.1.5 Bluetooth 4](#_Toc113382637)

[3.1.6 Emulazione 4](#_Toc113382638)

[3.2 React Native 4](#_Toc113382639)

[3.2.1 Sviluppo 4](#_Toc113382640)

[3.2.2 Orientamento dispositivo 4](#_Toc113382641)

[3.2.3 Dimensioni dispositivo 5](#_Toc113382642)

[3.2.4 Bluetooth 5](#_Toc113382643)

[3.2.5 Emulazione 5](#_Toc113382644)

[3.3 Scelta finale: // Da confermare 5](#_Toc113382645)

[4 Fonti 6](#_Toc113382646)

# Requisiti

# Analisi tecnologie

## App ibrida vs nativa

### Nativa

Un app nativa è compilata in uno specifico linguaggio per una specifica piattaforma. Queste applicazioni non sono veramente native, perché questo vorrebbe dire sviluppare un’applicazione diversa per ogni sistema, ma ci si avvicinano molto.

Le applicazioni native possono avere accesso a funzionalità legate all’hardware del dispositivo, come fotocamera, GPS, contatti, ecc.

### Ibrida

Un app ibrida è usa HTML, CSS e JavaScript per creare una versione web dell’applicativo e poi viene visualizzato all’interno del dispositivo.

Al contrario delle applicazioni native, quelle ibride non hanno accesso ad alcune funzionalità del dispositivo. Questo non vuol dire necessariamente che non sia sempre possibile accedergli: Ionic, ad esempio, usa dei plugin di Cordova per integrare le funzionalità native richieste per il funzionamento dei componenti hardware.

## Analisi tecnologie

Sono stati scelti esclusivamente frameworks che supportano lo sviluppo cross-platform, siccome l’applicativo deve funzionare sia su dispositive Android che iOS, e la possibilità di sviluppare due sistemi distinti è stata scartata.

### Xamarin

|  |  |
| --- | --- |
|  |  |
| Sviluppatore | Microsoft |
| Rilascio | 2011 |
| Linguaggio | C#, F# |
| Open source | Si |
| IDE | Visual Studio |

### Flutter

|  |  |
| --- | --- |
|  |  |
| Sviluppatore | Google |
| Rilascio | 2017 |
| Linguaggio | Dart |
| Open Source | Si |
| IDE | Indipendente |

### Ionic

|  |  |
| --- | --- |
|  |  |
| Sviluppatore | Drifty |
| Rilascio | 2013 |
| Linguaggio | JavaScript con Angular, React o Vue |
| Open Source | Si |
| IDE | Indipendente |

### Cordova

|  |  |
| --- | --- |
|  |  |
| Sviluppatore | Adobe |
| Rilascio | 2019 |
| Linguaggio | HTML5, CSS3, and JavaScript |
| Open Source | Si |
| IDE | Indipendente |

### React Native

|  |  |
| --- | --- |
|  |  |
| Sviluppatore | Meta (Facebook) |
| Rilascio | 2015 |
| Linguaggio | JavaScript con React |
| Open Source | Si |
| IDE | Indipendente |

## Comparazione

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Xamarin | Flutter | Ionic | Cordova | React Native |
| Sviluppatore | Microsoft | Google | Drifty | Adobe | Facebook |
| Rilascio | 2011 | 2017 | 2013 | 2009 | 2015 |
| Linguaggio | C#, F# | Dart | JavaScript con Angular, React o Vue | HTML5, CSS3, and JavaScript | JavaScript con React |
| Nativo/Ibrido | Nativo | Nativo | Ibrido | Ibrido | Nativo |
| Open source | Si | Si | Si | Si | Si |
| IDE | Visual Studio | Indipendente | Indipendente | Indipendente | Indipendente |

# Analisi approfondita

Ho deciso di scartare Flutter per via del nuovo linguaggio da dover imparare, il quale rallenterebbe notevolmente lo sviluppo iniziale.

// DA RIVEDERE ------------------

Tra Cordova e Ionic ho scelto di scartare Cordova siccome sviluppare per Ionic risulta più semplice ed immediato e ci si può

Ho deciso di scartare anche le varianti ibride (Ionic e Cordova) perché dovendo interagire con le funzionalità Bluetooth preferisco sviluppare in modo nativo.

## Xamarin

### Sviluppo

Gli strumenti all’interno di Visual Studio velocizzano il processo di sviluppo con controlli e layout predefiniti. La funzione di “Hot Reloading”, ovvero la possibilità di vedere l’applicazione aggiornata immediatamente senza bisogno di ricompilarla, accellera molto la parte di sviluppo.

### Performance

A livello di performance è quasi alla pari con le applicazioni sviluppate nativamente per iOS o Android.

### Orientamento dispositivo

La possibilità di avere schermate diverse a dipendenza se il dispositivo è in orizzontale o in verticale è direttamente integrata nel framework.

### Dimensioni dispositivo

Supporta vari tipi di dispositivo e si possono configurare facilmente delle viste per determinati dispositivi.

### Bluetooth

Per connettersi ad un dispositivo che usa Bluetooth Classic, si può aggiungere un plugin chiamato “Plugin.BluetoothClassic[[1]](#footnote-1)”.

### Emulazione

Visual Studio racchiude un emulatore Android tramite il quale si può vedere l’applicativo e viene aggiornato in tempo reale. L’emulatore gira senza problemi su una macchina dotata di almeno 16GB di RAM[[2]](#footnote-2) e con installato Hyper-V o altri software che assistono alla virtualizzazione, come Intel HAXM.

## React Native

### Sviluppo

Come per Xamarin, anche React Native supporta la funzione di ‘Hot Reloading’. Performance

A livello di performance è quasi alla pari con le applicazioni sviluppate nativamente per iOS o Android.

### Orientamento dispositivo

Non è predisposto ad avere la gestione comoda dell’interfaccia a dipendenza dell’orientamento del dispositivo. Per fare ciò è necessario avvalersi di una libreria esterna (react-native-orientation), la quale richiede una configurazione manuale scritta nel linguaggio nativo per il dispositivo (Objective C per iOS o Java per Android).

Tutto questo rende il tutto impossibile da testare senza compilare il tutto e installarlo su un dispositivo.

### Dimensioni dispositivo

React Native usa un sistema senza unità per definire le dimensioni dei vari elementi, il quale si basa sulla densità di pixel del dispositivo. Questa è una soluzione che può creare dei problemi quando si passa su dispositivi con uno schermo e una densità di pixel molto alti. Ci sono però delle librerie che danno più flessibilità e aiutano a dare un risultato migliore (ad esempio “react-native-size-matters”).

È anche possibile implementare due schermate diverse a dipendenza della dimensione del dispositivo.

### Bluetooth

Al contrario di Xamarin non c’è una soluzione per la connessione Bluetooth direttamente integrate ma esistono delle soluzioni sviluppate da terzi.

Un esempio è la libreria “react-native-bluetooth-classic[[3]](#footnote-3)” offre un implementazione altrettanto semplice e veloce quanto quella di Xamarin.

### Emulazione

React Native non necessita però di un emulatore ma l’applicativo può essere visualizzato su un qualsiasi browser moderno.

## Scelta finale: // Da confermare

//

Ho personalmente esperienza con React nell’ambito dello sviluppo web, per questo motivo tenderei per React Native, siccome il passaggio è molto semplice.

Ma data la probabile necessità di dover implementare delle schermate diverse a dipendenza dell’orientamento del dispositivo ho deciso di andare su Xamarin.

# Fonti

* <https://blog.logrocket.com/react-native-vs-ionic/>
* <https://blog.back4app.com/xamarin-vs-ionic/>
* <https://devcount.com/xamarin-vs-flutter/>
* <https://www.simform.com/blog/xamarin-vs-ionic/>
* <https://blog.logrocket.com/react-native-vs-ionic/>
* <https://www.waldo.com/blog/cordova-vs-react-native>
* <https://brainhub.eu/library/react-native-vs-xamarin>
* <https://medium.com/@stenalferd/flutter-vs-xamarin-vs-react-native-let-the-battle-begin-d3e783bb4bf1>
* <https://github.com/yamill/react-native-orientation>
* <https://medium.com/enigmamx/how-to-integrate-bluetooth-le-in-xamarin-forms-2dcdf974703a>
* <https://goshacmd.com/different-mobile-desktop-tablet-layouts-react/>
* <https://medium.com/react-native-training/scaling-react-native-apps-for-tablets-211de8399cf1>
* <https://stackoverflow.com/a/12201785/1685157>
* <https://support.apple.com/en-us/HT204387>
* <https://mfi.apple.com/en/who-should-join.html>

1. https://github.com/rostislav-nikitin/Plugin.BluetoothClassic [↑](#footnote-ref-1)
2. Riferimento personale in base alle risorse utilizzate avendo l’ambiente di sviluppo e la macchina virtuale avviata. [↑](#footnote-ref-2)
3. https://www.npmjs.com/package/react-native-bluetooth-classic [↑](#footnote-ref-3)