Cross-platform Mobile App Development

Desenvolvimento de Software e Sistemas Móveis (DSSMV)

Licenciatura em Engenharia de Telecomunicações e Informática

LETI/ISEP

2025/26

Paulo Baltarejo Sousa

pbs@isep.ipp.pt

Instituto Superior de Engenharia do Porto

P.PORTO

Disclaimer

Material and Slides

Some of the material/slides are adapted from various:

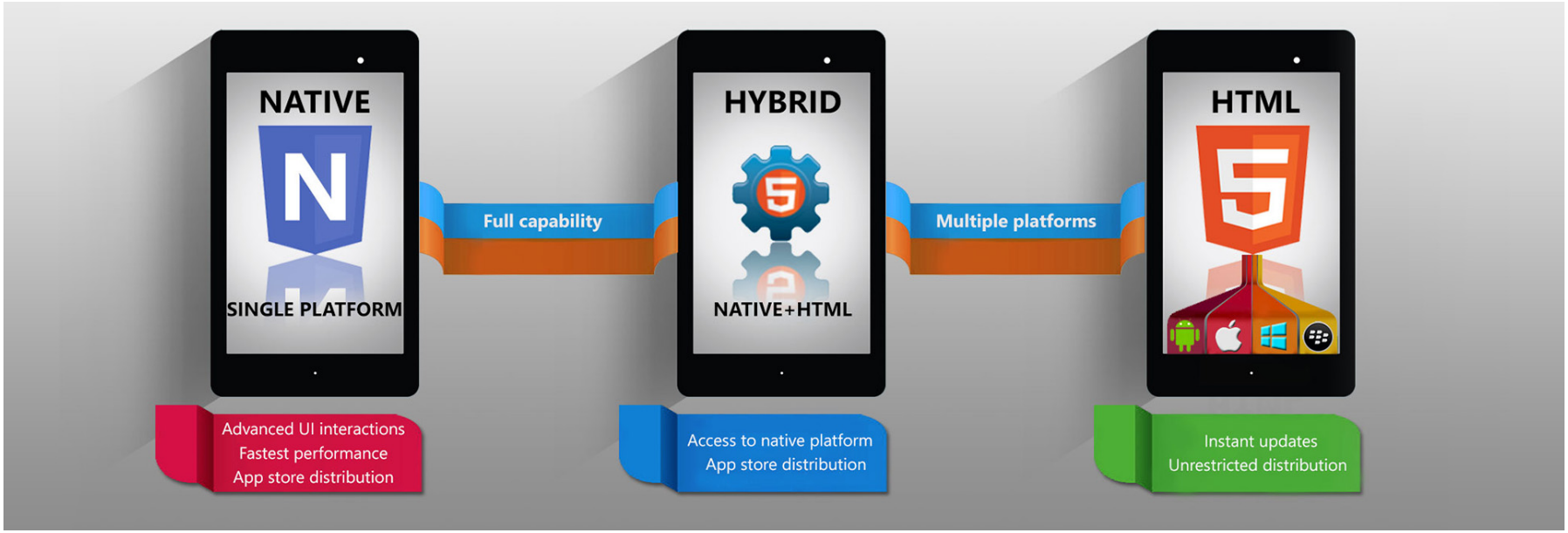
- Presentations found on the internet;
- Books;
- Web sites;
- ...

Outline

Mobile Apps Development

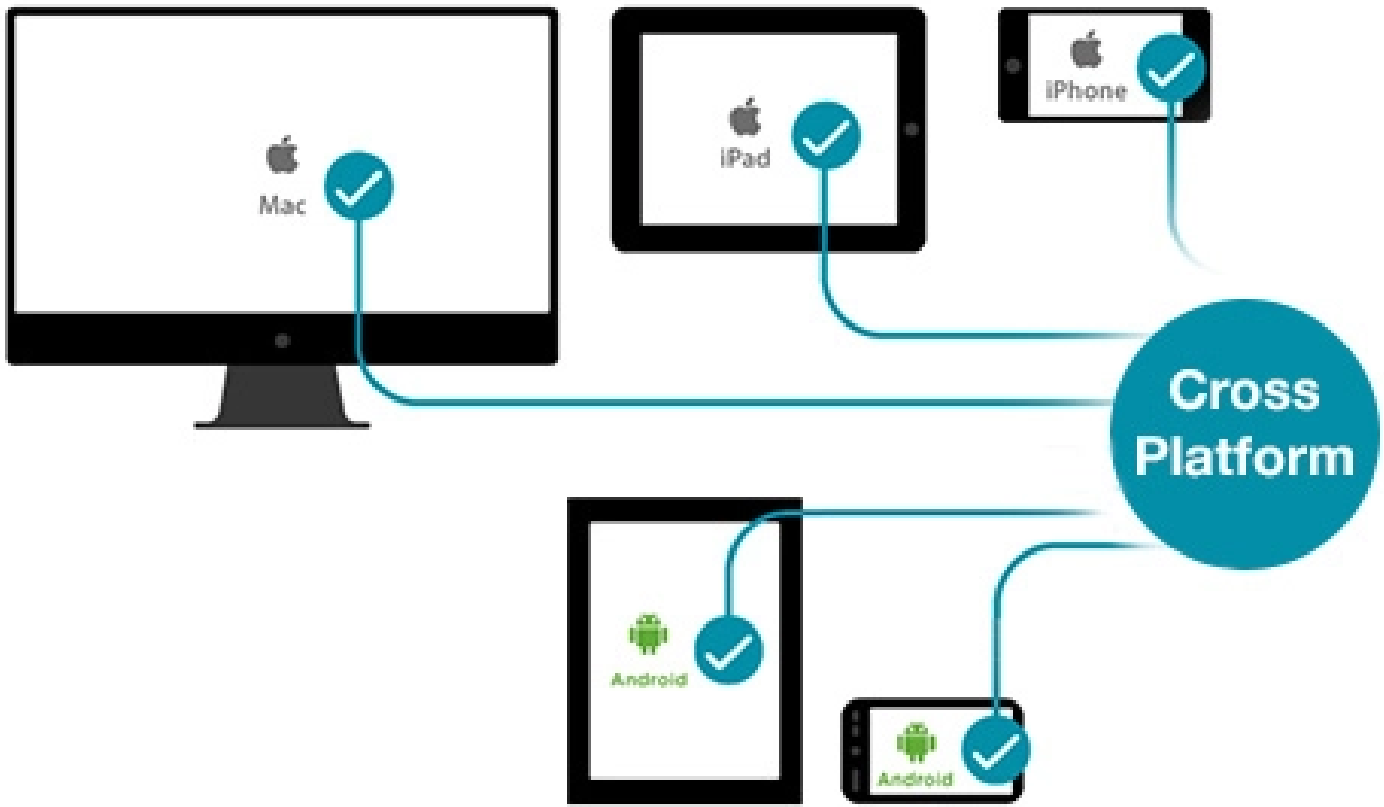
Mobile Apps Development (I)

- **Native Mobile App**: Native mobile apps are developed to be specific to a single platform or mobile O.S.
- **Web Mobile App**: Execute in a web browser - while requiring an internet connection - encompass mobile web applications.
- **Hybrid Mobile App**: It is a blend of a native app and a web app.



Mobile Apps Development (II)

- **Cross-Platform Mobile Applications**: Cross-platform mobile apps are designed to run on various mobile platforms and Operating Systems without developers needing to recode for specific, native platforms.

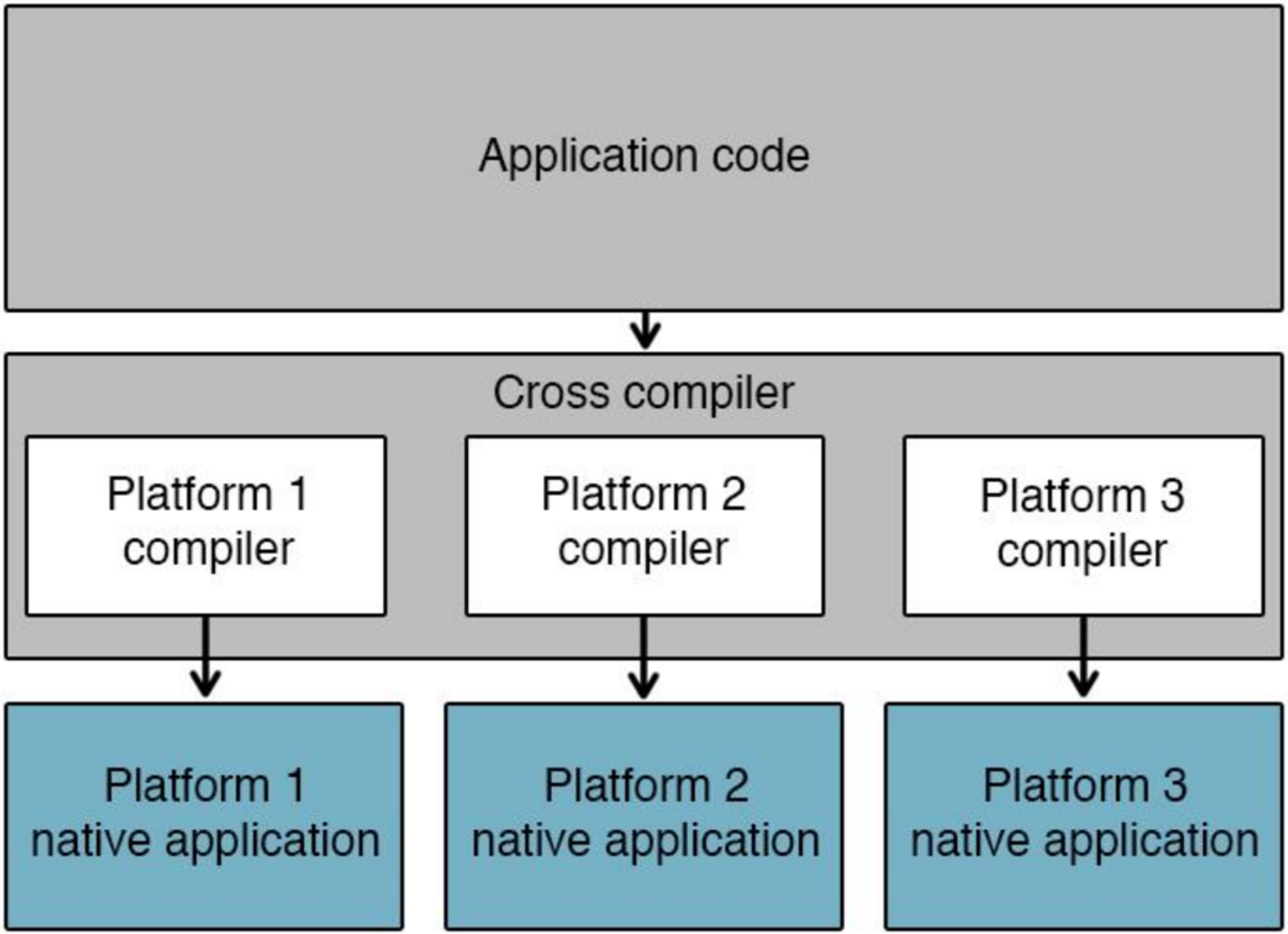


What Is Cross-Platform Development?

Cross-Platform Development (I)

- The term **cross-platform mobile app development** refers to an approach that allows developers to build a mobile solution that is simultaneously compatible with several operating systems and platforms (Android, iOS, Windows).
- Cross-platform app have a native look and feel due to the combination of native code with independent code that is suitable for multiple platforms.
- Developers write the **code once and then reuse it**, thus making it possible to release a product quickly.
- To code cross-platform software, **developers use intermediate programming languages**, JavaScript, C#, Dart and others, not native to devices and OSs.
- Then, apps are packaged into native containers and integrated into platforms.

Cross-Platform Development (II)



Native vs. Cross-Platform Apps (I)

- Native development **relies on tools and programming languages designed** specifically for one platform.
  - Objective-C and Swift are used for iOS apps;
  - Java, C/C++ and Kotlin are used for Android apps;
- When creating native applications, developers are guided by specific OS requirements.
- Installed on a targeted computing device, **a native app can fully use the available features and capabilities**.
- The drawback is that **a native app is not compatible with other platforms**.

Native vs. Cross-Platform Apps (II)

- **To reach wider audiences**, you would need **to develop several solutions** (one app for each targeted platform), which is time-consuming and cost-intensive.
- **Cross-platform development aims at creating a single application that runs identically on several platforms**.
  - It uses platform-agnostic technologies such as HTML and CSS and helps businesses cover many end devices at lower costs.

Native vs. Cross-Platform Apps(II)

	Native	Cross-Platform
Code	For a single platform	Sharable
Access to device capabilities	Full	Limited
Performance	Seamless	High, but lags and compatibility issues may occur
Development cost	High	Relatively low

Pros and Cons of Cross-Platform Mobile Development (I)

- Cross-Platform development is definitely the way to go for companies that want **to attract users of various mobile devices and release the product to market quicker at a lower cost**.
- The more complicated code of hybrid solutions combines native and non-native components, **which may affect the performance**.
- Cross-platform **apps cannot support all native-only functions and features of mobile devices**, such as advanced graphics and animation or 3D effects.
- This results in limited functionality and poorer app design.

Pros and Cons of Cross-Platform Mobile Development (II)

Frameworks

Frameworks

- According to Statista website [1] in 2023 **Flutter** and **React Native** are the most used cross-platform mobile frameworks with 46% and 35% of market share, respectively.

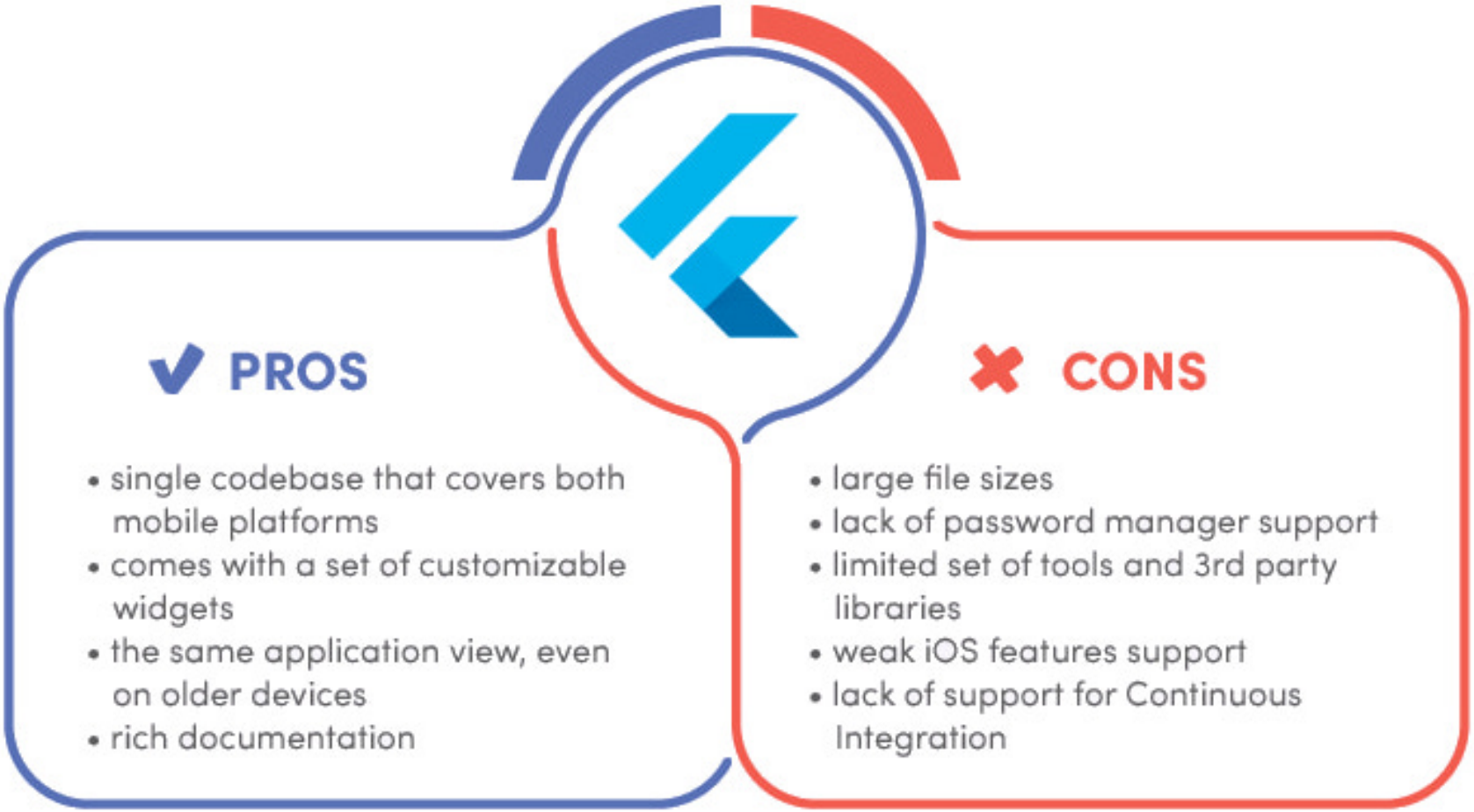
[1] https://www.statista.com/statistics/869224/worldwide-software-developer-working-hours/

Flutter [2]

- Flutter is an open-source mobile framework launched by Google (2017).
- Flutter apps are written in the Dart language.
- Dart is compiled Ahead-Of-Time (AOT), also called precompilation, into native code for several other platforms.
- **Hot Reload**: This feature updates your app immediately when you make any change to the code.
- **Widgets**: User Interface based on widgets.
- **Code integration**: Flutter sports multiple mechanisms that facilitate interoperability not only with external code but also APIs.
  - For example, you could reuse native languages such as Android's native Kotlin or iOS's Swift.

[2] `https://flutter.dev/`

Flutter: Pros and Cons

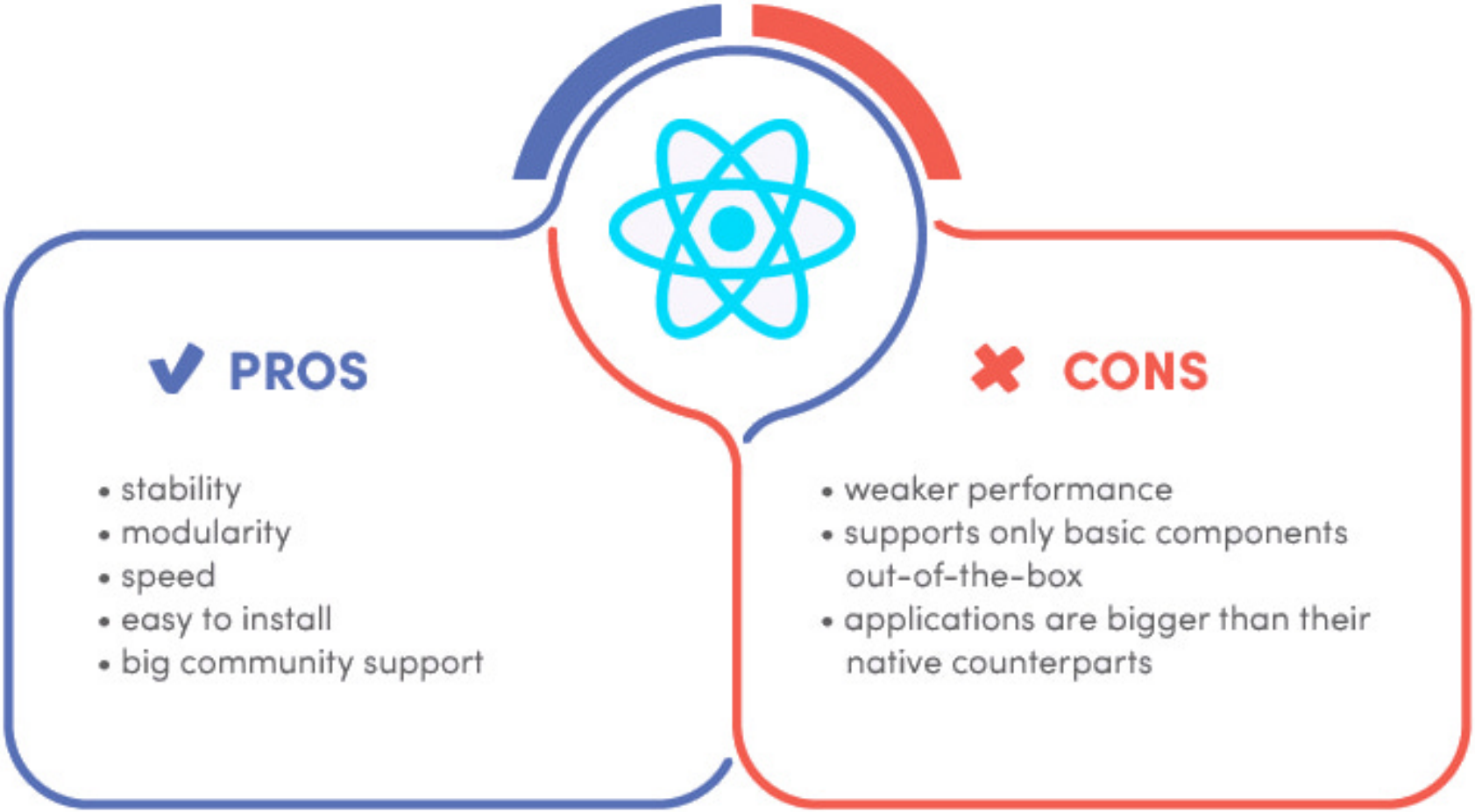


React Native [3]

- React Native is a JavaScript framework for writing real, natively rendering mobile applications for iOS and Android.
- It's based on React, Facebook's JavaScript library (called ReactJS) for building user interfaces, but instead of targeting the browser, it targets mobile platforms.
- React Native applications are written using a mixture of **JavaScript and XML**, known as JSX.
- React apps render real mobile UI components.
- **Hot Reload**: This feature updates your app immediately when you make any change to the code.

[3] `https://reactnative.dev/`

React Native: Pros and Cons

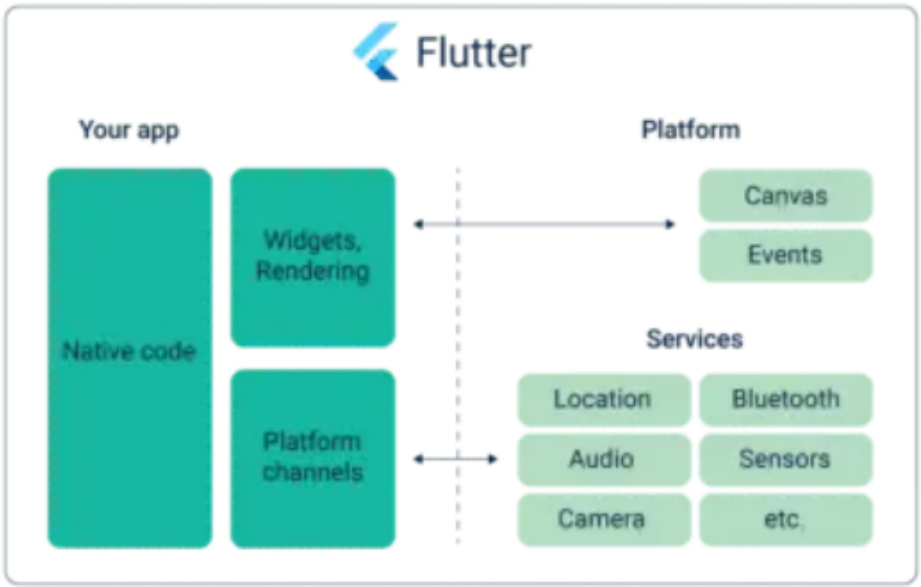


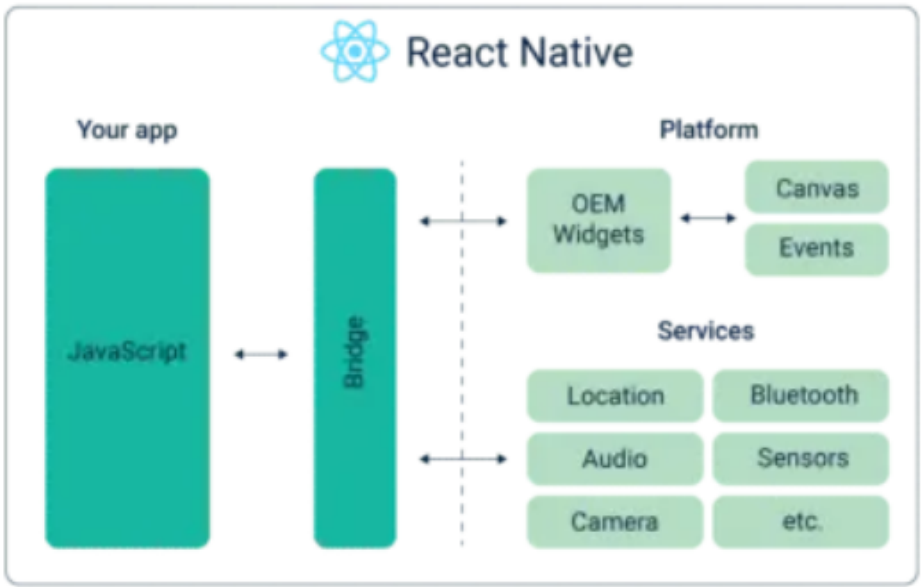
Flutter *vs* React Native (I)

	Flutter	React Native
Initial release	May 2017	March 2015
Backed by	Google	Facebook
Programming language	Dart	JavaScript
Platform support	Android, iOS	Android, iOS, Web apps
Application performance	Fairly robust, 60 fps	Close to native
IDE support	Android Studio, Visual Studio Code, IntelliJ IDEA	Range of IDE's and tolls with JS support
Native appearance	Better. Has access to device core functionalities	Lower. Dependency on third party apps
Hot reloading	Yes	Yes
GUI	Use proprietary widgets and deliver UI	Use native UI controller
Time to market	Faster	Slower than Flutter
Code reusability	50-90%	90%
Testing	Mobile device or emulator	Mobile device or emulator
Community & support	Limited, Fast growing	Extensive

Flutter *vs* React Native (II)

- React Native uses the JavaScript bridge, which is the JavaScript runtime environment.
  - This provides a pathway to communicate with the native modules.
- Flutter contains most of the required components within itself.





Bibliography

Resources

- David Flanagan, "JavaScript: The Definitive Guide", O'Reilly Media, Inc., 2020
- Adam Boduch and Roy Derks, "React and React Native, Third Edition,A complete hands-on guide to modern web and mobile development with React.js"
- Dan Ward, "React Native Cookbook Second Edition Step-by-step recipes for solving common React Native development problems"
- `https://reactnative.dev/`
- `https://reactjs.org/`
- `https://reactnavigation.org/`