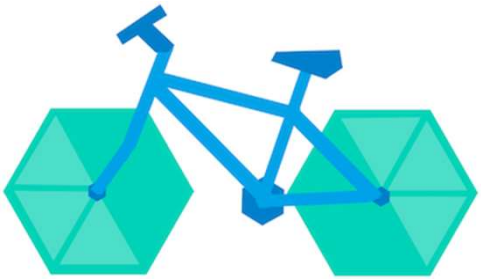
Objectives

- To give a basic understanding about Mobile Application Development Using Google Flutter & Dart .



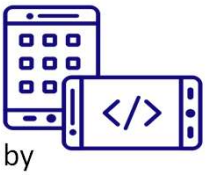
iTelaSoft™ Making IT Simple

Overview

- Evolution of Mobile application Development
- Different Approaches To Mobile Applications Development
- Introduction to Flutter
- Why Does Flutter Matters and how it differentiate from others ?
- Introduction to Dart
- Driving deep on Mobile application development architectures and the approach of flutter.
- Flutter Widgets & Layout
- Comparison of React-Native Vs Flutter
- So, “What’s new and exciting about Flutter?”

iTelaSoft™ Making IT Simple

A brief history of mobile app development



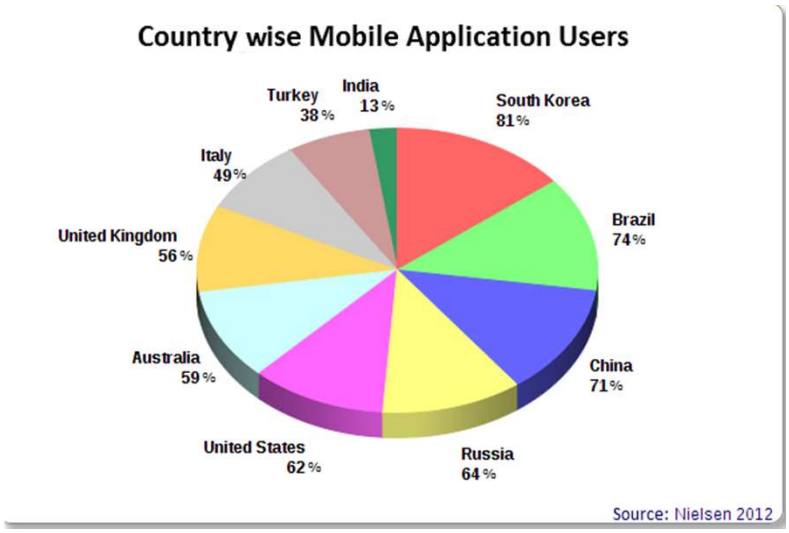
- On the 3rd day of April 1973, the first mobile phone call was made by Martin Cooper of Motorola to Dr. Joel S. Engel of Bell Labs. That device or instrument weighed 1.1 kg and measuring (23 x 13 x 4.45) cm. And, it took two decades of Research and development (R&D) to get first mobile application for smartphones, and the credit goes to IBM Simon, who introduced the world with the first mobile apps for smart phones.
- However, the first smart phone was announced for the general use by IBM in 1993 that was equipped with the features like calculator, world clock, calendar and contact book.
- The announcement of the first iPhone was a giant leap towards the evolution of mobile apps. The digital keyboard, multi-touch display, and finally a functional web browser totally revolutionized the way people were using mobile apps. July 2008 is when everything changed: Apple's App Store went online.

iTelaSoft™ Making IT Simple

Evolution of Mobile - OS

- 1973-1993 – Embedded System based OS
- 1996 – Palm Pilot, Windows CE
- 2000 – Symbian
- 2002 – BlackBerry
- 2007 – iOS, Windows Mobile
- 2008 – Android
- 2009 – webOS, Bada, Palm OS
- 2011 – Tizen
- 2012 – Firefox OS
- 2013 – Ubuntu Touch

World's most popular OS and Smartphone. Till 2010…



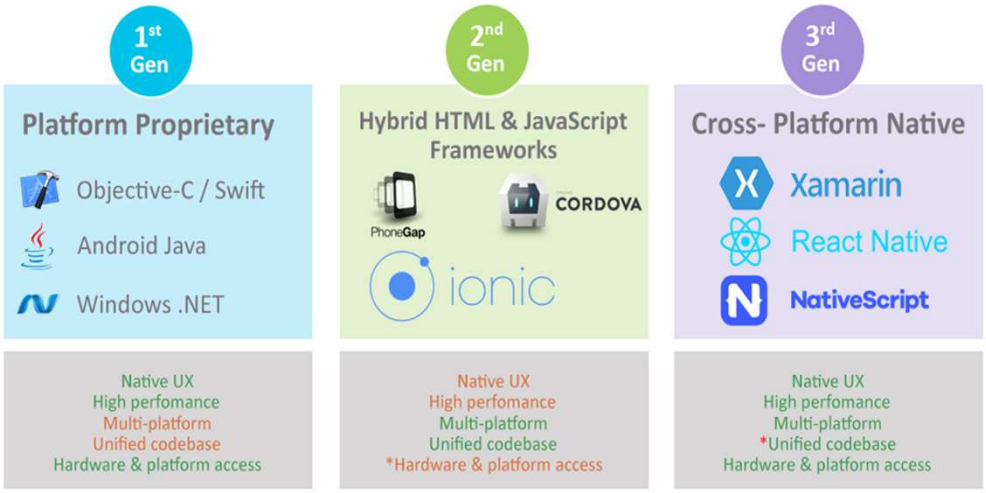
iTelaSoft™ Making IT Simple

Different Approaches To Mobile Applications Development

1st Gen	2nd Gen	3rd Gen
Platform Proprietary	Hybrid HTML & JavaScript Frameworks	Cross- Platform Native
Objective-C / Swift	PhoneGap, CORDOVA, ionic	Xamarin
Android Java		React Native
Windows .NET		NativeScript
Native UX High perfomance Multi-platform Unified codebase Hardware & platform access	Native UX High perfomance Multi-platform Unified codebase *Hardware & platform access	Native UX High perfomance Multi-platform *Unified codebase Hardware & platform access

iTelaSoft™ Making IT Simple

Pros & Cons



iTelaSoft™ Making IT Simple

Flutter!!!

Flutter

iTelaSoft™ Making IT Simple

WHAT is Flutter and WHY succeeds at cross platform development

- Flutter is an open-source cross-platform mobile app development framework which, as Google defines it, "allows you to **build beautiful native apps on iOS and Android from a single codebase**". First alpha version was released on May 2017; and after 1.5 years, Flutter got its first stable 1.0 release on December 4th, 2018.
- Google's development kit allows you to build 2D mobile apps. You can use it to develop full-featured apps including support for cameras, geolocation, network, storage, and many more.
- In addition to support for Android and iOS platforms, Flutter will be the main framework for developing applications for Google's upcoming operating system, **Fuchsia**. It is currently under development and will possibly and gradually replace Android.

iTelaSoft™ Making IT Simple

Why Does Flutter Matters ?

& Why I'm Dropping Everything Else

iTelaSoft™ Making IT Simple

What is Dart? And why?

Dart is the object-oriented, garbage-collected programming language that you use to develop Flutter apps. It was also created by Google, but is open-source, so it has community inside and outside Google.

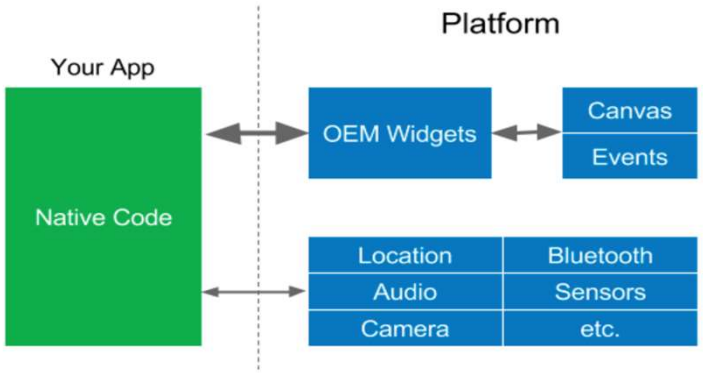
Other than its Google origins, Dart was chosen as the language of Flutter for the following reason: It's one of very few languages that can be compiled both AOT (ahead-of-time) and JIT (just-in-time).

- **JIT compilation** is used during app development process since it enables hot reloads (about which I will talk more in next questions) and fast development cycles by compiling code at runtime on the fly.
- **AOT compilation** is used when you are done with development and ready for release. Code is then compiled AOT to native code, enabling fast startup and performant execution of the app.

iTelaSoft™ Making IT Simple

The Platform SDKs

- The Apple iOS SDK was released in 2008 and the Google Android SDK in 2009. These two SDKs were based on different languages: Objective-C and Java, respectively.

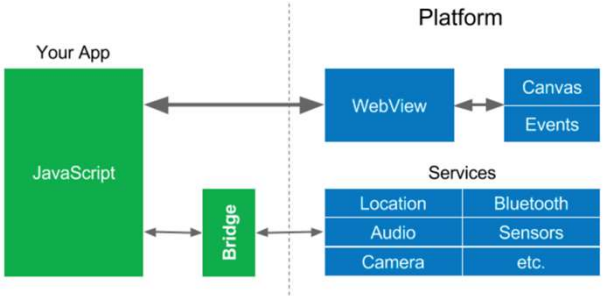


Your app talks to the platform to create widgets, or access services like the camera. The widgets are rendered to a screen canvas, and events are passed back to the widgets. This is a simple architecture, but you pretty much have to create separate apps for each platform because the widgets are different, not to mention the native languages.

iTelaSoft™ Making IT Simple

WebViews

- The first cross-platform frameworks were based on JavaScript and WebViews. Examples include a family of related frameworks: PhoneGap, Apache Cordova, Ionic, and others. Before Apple released their iOS SDK they encouraged third party developers to build webapps for the iPhone, so building cross-platform apps using web technologies was an obvious step.

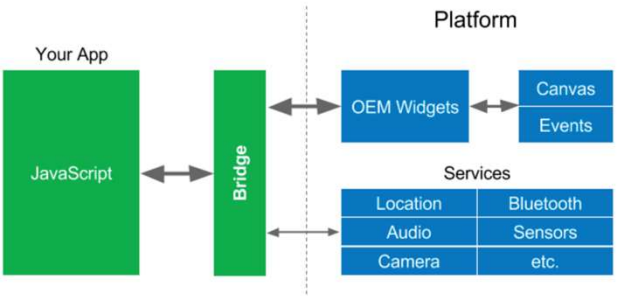


Your app creates HTML and displays it in a WebView on the platform. Note that it is difficult for languages like JavaScript to talk directly to native code (like the services) so they go through a "bridge" that does context switches between the JavaScript realm and the native realm. Because platform services are typically not called all that often, this did not cause too many performance problems.

iTelaSoft™ Making IT Simple

Reactive Views

Reactive web frameworks like ReactJS (and others) have become popular, mainly because they simplify the creation of web views through the use of programming patterns borrowed from reactive programming. In 2015, React Native was created to bring the many benefits of reactive-style views to mobile apps.



React Native is very popular, but because the JavaScript realm accesses the platform widgets in the native realm, it has to go through the bridge for those as well. Widgets are typically accessed quite frequently (up to 60 times a second during animations, transitions, or when the user "swipes" something on the screen with their finger) so this can cause performance problems.

iTelaSoft™ Making IT Simple

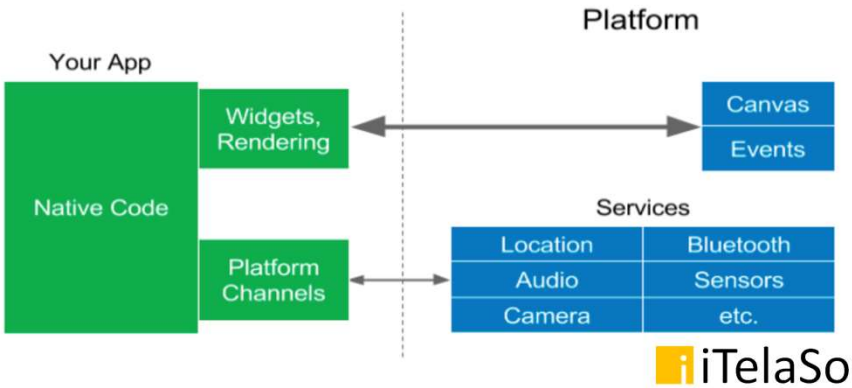
How it happens in Flutter

- Like React Native, Flutter also provides reactive-style views. Flutter takes a different approach to avoiding performance problems caused by the need for a JavaScript bridge by using a compiled programming language, namely Dart.
- Dart is compiled “ahead of time” (AOT) into native code for multiple platforms. This allows Flutter to communicate with the platform without going through a JavaScript bridge that does a context switch. Compiling to native code also improves app startup times.

iTelaSoft™ Making IT Simple

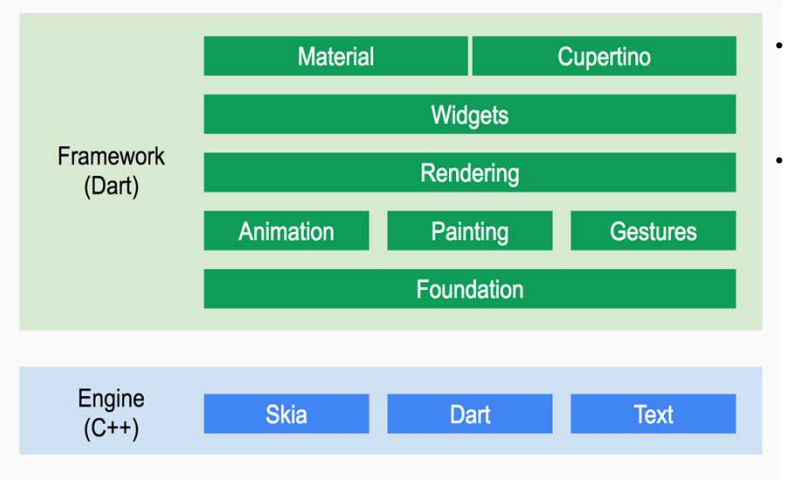
Overview

- There is still an interface between the Dart program (in green) and the native platform code (in blue, for either iOS or Android) that does data encoding and decoding, but this can be orders of magnitude faster than a JavaScript bridge.



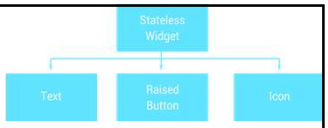
iTelaSoft™ Making IT Simple

Flutter Architecture



- Everything in green in this figure can be customized:
- Most of the engine is written in C++, with Android-specific parts written in Java, and iOS-specific parts written in Objective-C. Some basic classes and functions in dart:ui are written in Dart and mostly serve as bridges between Dart and C++.

Widgets



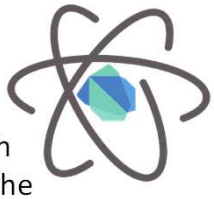
- Widgets are the elements that affect and control the view and interface to an app. It is not an overstatement to say that the widgets are one of the most important parts of a mobile app. In fact, widgets alone can *make* or *break* an app.
- That's right, Flutter does not use the platform widgets (or DOM WebViews), it provides its own widgets.
- All that Flutter requires of the platform is a canvas in which to render the widgets so they can appear on the device screen, and access to events (touches, timers, etc.) and services (location, camera, etc.).
- Flutter comes with rich, customizable widget sets for Android, IOS, and Material Design (in fact, we have been told that Flutter has one of the highest fidelity implementations of Material Design).
- As an added benefit, you can write your app so it uses the new widget even on older OS versions.

Layout

- One of the biggest improvements in Flutter comparatively to other platforms is how it does its layout. Layout determines the size and position of widgets based on a set of rules (also called constraints).
- Instead of having a large set of layout rules that could be applied to any widget, each widget would specify its own simple layout model.
- To simplify layout even further, flutter turned almost everything into a widget.
- Flutter includes quite a few widgets for doing layout, not just columns & rows, but also grids, lists, etc.
- In addition, Flutter has a unique layout model we call the “sliver layout model” which is used for scrolling.

iTelaSoft™ Making IT Simple

React-Native Vs Flutter



- React Native has to communicate with the native widgets through the bridge, so the virtual tree of widgets helps keep passes over the bridge to a minimum, while still allowing the use of native widgets. Finally, once the native widgets are updated, the platform then renders them to the canvas.

- It uses several internal tree structures to render only those widgets that need to be updated on the screen. There are no native platform widgets to manipulate, so what was a virtual widget tree is now the widget tree. Flutter renders the widget tree and paints it to a platform canvas. This is nice and simple (and fast).

iTelaSoft™ Making IT Simple

Flutter vs React Native — A Quick Comparison

	Flutter	React Native
Language	Dart	JavaScript
Developers	Google	Facebook
Initial Release	2017	2015
Native Performances	Superior	Good
Time-to-Market of App	Faster	Slower than Flutter
Documentation	Precise, clear, and up-to-date	Up-to-date and imprecise
Who Uses	Alibaba, Reflectly, Google Greentea, Tencent, Google Ads, App Tree	Facebook, Instagram, Pinterest, Uber, Tesla, Walmart, Wix.com, Baidu Mobile, Artsy, and so on
Competitive Advantage	Simpler, faster, and elegant native apps using easy-to-use Flutter SDK	More than 3 years in the market with the ability to deliver native-like app experiences

So, "What's new and exciting about Flutter?"

- The advantages of reactive views, with no JavaScript bridge
- Comes with beautiful, customizable widgets(eg:- Navigation)
- At last year's MWC, Google announced that their Flutter app development SDK was ready to move from alpha to beta testing. Today at MWC 2019, Flutter is getting its second stable release, version 1.2, combined with the release of version 2.2 of the Dart programming language.
- Another investment the Flutter team has made is into a third debugging tool for Flutter and Dart applications to complement Visual Studio Code and Android Studio, called Dart Dev-Tools. It's actually a web-based application that lets you inspect widgets, view logs, and do full application debugging right from your web browser.
- In-App Purchases and Android App Bundles are supported in Flutter 1.2.
- New Android App Bundles help to reduce app size and enable new features such as dynamic delivery for apps. As exciting as it sounds, the Flutter team has also carefully carried out many bug fixes for maps, video player, and webview.

iTelaSoft™ Making IT Simple

- Hummingbird is the embodiment of the team's experiments to bring Flutter to the web. It is a "web-based implementation of the Flutter runtime that takes advantage of the capability of the Dart platform to compile not just to native ARM code but also to JavaScript. This enables Flutter code to run on the standards-based web without change."
- The project is still at the incipient stage and under heavy development but we do have some prototype concepts that we can go through.

iTelaSoft™ Making IT Simple

Where to start Flutter

Resources

- https://flutter.dev/docs/get-started/install
- https://www.udemy.com/learn-flutter-dart-to-build-ios-android-apps/
- https://www.pluralsight.com/courses/flutter-getting-started
- https://www.youtube.com/results?search_query=%23WidgetoftheWeek

iTelaSoft™ Making IT Simple

Give away and Q & A

Thank you!!!!

iTelaSoft™ Making IT Simple