Tus Primeros pasos en el desarrollo Móvil con Flutter

Ana Cecilia Castillo | @zezzi
Mobile Engineer

April 2022

Aficionada a correr, a la bici, leer, la naturaleza, los perros y viajar.

Women Techmakers

- Software Developer at PayPal
- GDG Organizer
- WTM ambassador
- Android & iOS

@zezzi

@zezzi

Agenda

- Native vs Multiplataforma
- Desafíos del desarrollo Móvil
- Declarative UI
- Por que Flutter
- Framework
- Widgets
- WorkShop

https://github.com/zezzi/flutter-wwcode-choose-your-adventure

Empecemos con un ejemplo sencillo

https://dartpad.dev/

Starter

https://gist.github.com/zezzi/3271c37fb8df77c1858ff000f1496ac2

Terminado

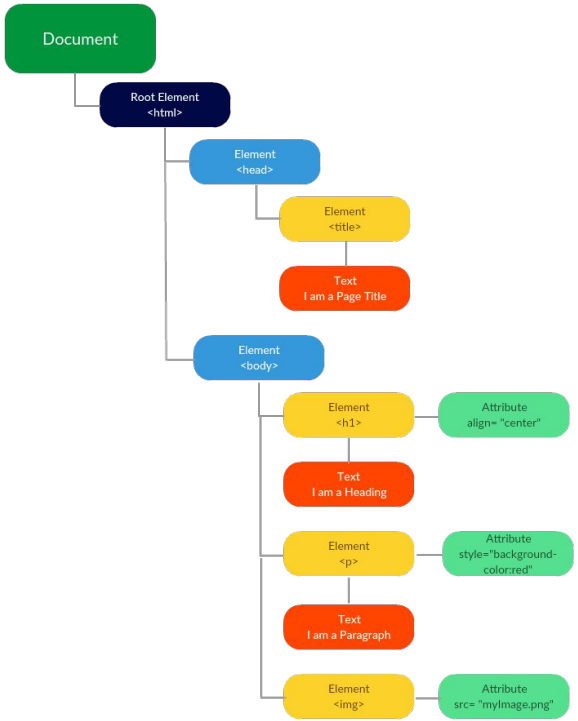
https://gist.github.com/zezzi/4befe1fefb882bcab960e83329ff8517

Tree of <Element>

Tree of <UIViews>

Tree of <Views>

Tree of <Composables>



https://mrvirk.com/html-dom-diagram-and-explanation.html

Por qué desarrollar para mobile

Native VS Multiplatform

Promesas:

- Costo
- Reusabilidad
- Tecnologías nuevas
- Misma funcionalidad para Android y iOS

- Asumir que es mucho más sencillo contratar o re-entrenar
- Hot reload

Native VS Multiplatform

Trade Offs

- Performance
- Experiencia para los desarrolladores
- Tooling
- Device Support
- UI/UX “native” look and feel/

- Cross-platform siempre nos van a limitar un poco mas.
- Build performance
- Siempre un Paso atrás
- Código Nativo en la mayoría de casos se necesita

desafíos debido a la naturaleza del desarrollo móvil

UI	Construcción del UI distintos Devices and Screen orientation	Performance
	Componentes Reusables	Threads (Main Thread vs Background Threads)
	Imperativo o Declarativo	Accesibilidad/ Localización
	Manejo del Estado	Fragmentación de Dispositivos/ Soporte de API antiguos
Arquitectura	Inyección de Dependendencias	S.O.L.I.D/ Design Patterns
	MVVM/ Clean Architecture/ BLoC/ Repository Pattern	Data Storage (database, user preferences, cache)
	Manejo de Dependencias (Third Party Libraries)	Modularization of Code
	Manejo de Assets	Navigation and Sharing Information between screens

@zezzi

Network/ Memory/ Data	JSON/ GraphQL	Serialize and deserialize
	Métodos para manejar las llamadas Asíncronas	Push Notifications/ Deep Links
	Updating the UI	Memory Management
	Threads	Caching/ Offline Support/ Security
Infraestructura	Publishing	Lint, Code Organization, Style of Code,Code formatting
	Certificates and Provisioning Profiles	CI / CD
	Unit Testing/ UI Testing	Analytics
	Manual Testing/ Automated Testing	Performance/ Crash Management

@zezzi

En que nos enfocaremos en el taller

UI	**Construcción del UI**	Performance
	Componentes Reusables	Threads (Main Thread vs Background Threads)
	Imperativo o Declarativo	Accesibilidad/ Localización
	Manejo del Estado	Fragmentación de Dispositivos/ Soporte de API antiguos
Arquitectura	Inyección de Dependendencias	S.O.L.I.D
	MVVM/ Clean Architecture/ BLoC/ Repository Pattern	Data Storage (database, user preferences, cache)
	Manejo de Dependencias (Third Party Libraries)	Modularization of Code
	Manejo de Assets	Navigation and Sharing Information between screens

@zezzi

WHAT IS...

Declarative v. Imperative

Directions	Distance
1. Start out going South on N AKARD ST toward FEDERAL ST.	0.1 miles
2. Turn RIGHT onto ELM ST.	0.5 miles
3. Turn SLIGHT LEFT to stay on ELM ST.	0.1 miles
4. ELM ST becomes COMMERCE ST.	<0.1 miles
5. Merge onto I-35E S.	57.4 miles
6. Merge onto I-35 S/US-81 S via the exit- on the left.	137.6 miles
7. Take the exit- exit number 234A- toward 1ST ST/HOLLY ST.	<0.1 miles
8. Turn SLIGHT LEFT onto I-35 N.	<0.1 miles
9. Turn RIGHT onto CESAR CHAVEZ ST E/E 1ST ST/TX-343 LOOP.	0.4 miles
10. Turn RIGHT onto BRAZOS ST.	<0.1 miles
11. Turn LEFT onto E 2ND ST.	<0.1 miles
12. Turn LEFT onto CONGRESS AVE S.	0.3 miles
13. Turn RIGHT onto BARTON SPRINGS RD.	0.1 miles

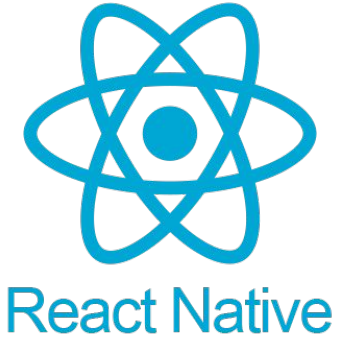
UI = f (state)

The layout on the screen

Your build methods

The application State

UI Declarativos El Inicio

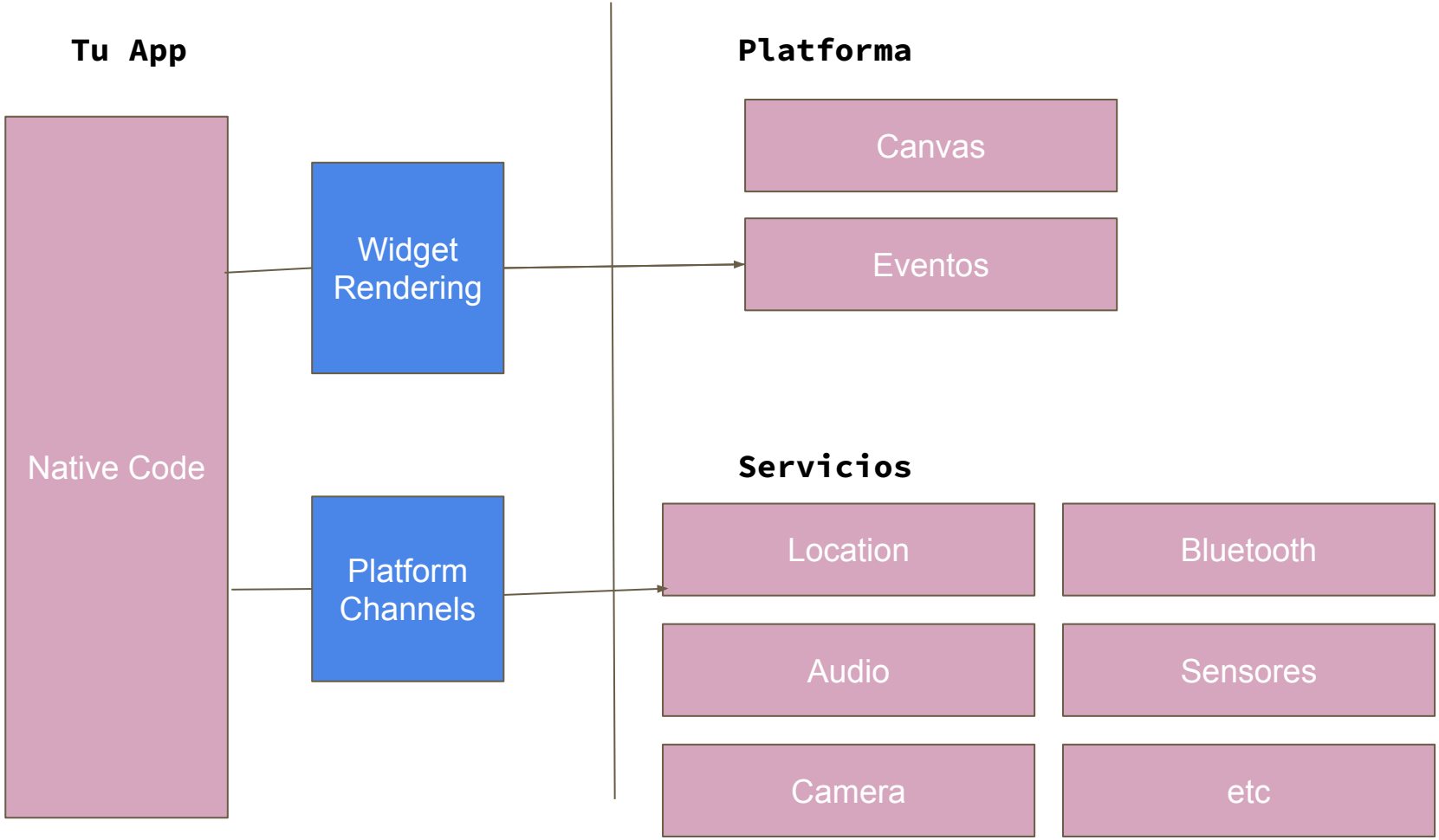


“Learn once, write anywhere,” as the mantra goes.

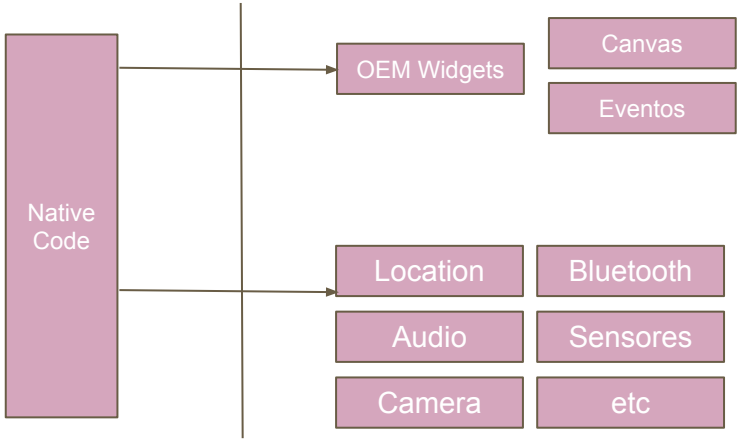
“120 Hz or bust” interface and a fast development cycle with hot reload.

Swift UI y JetPack Compose

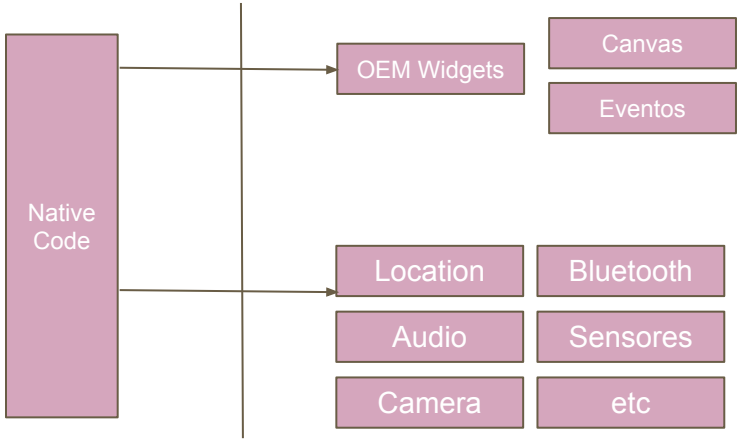
Por que Flutter?


Tu App
Platforma
Canvas
Widget Rendering
Eventos
Native Code
Servicios
Location
Bluetooth
Platform Channels
Audio
Sensores
Camera
etc

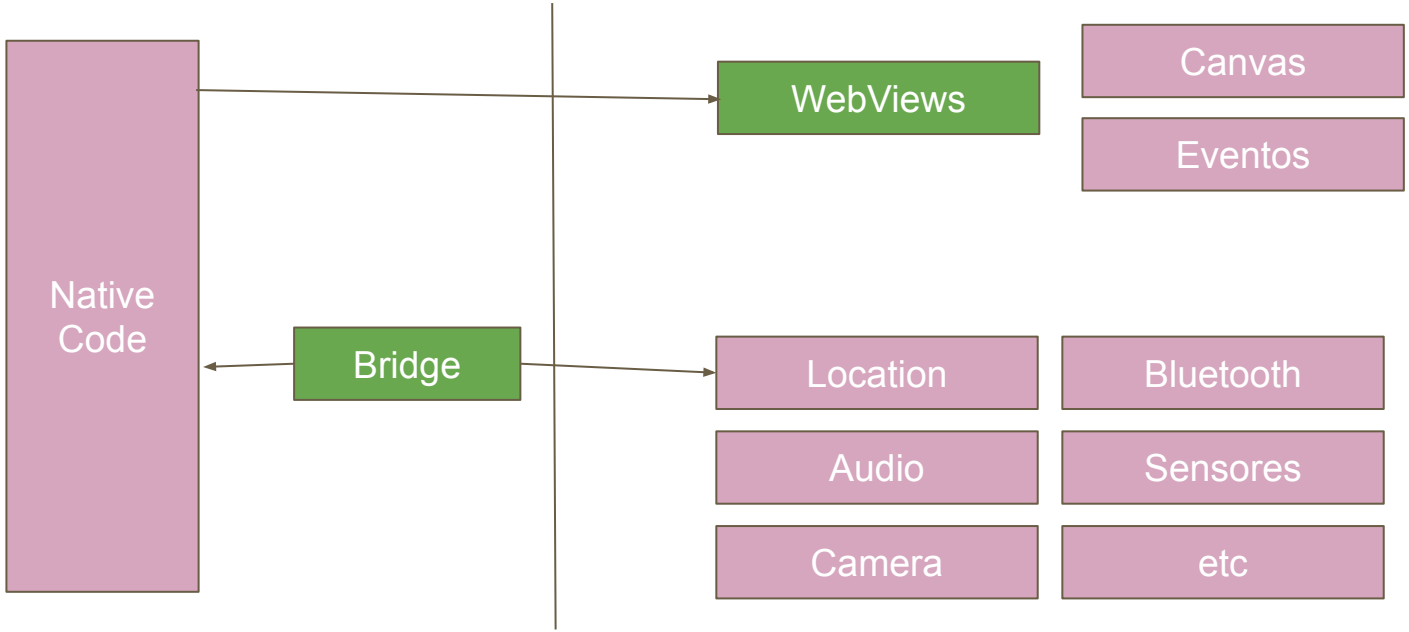
Android

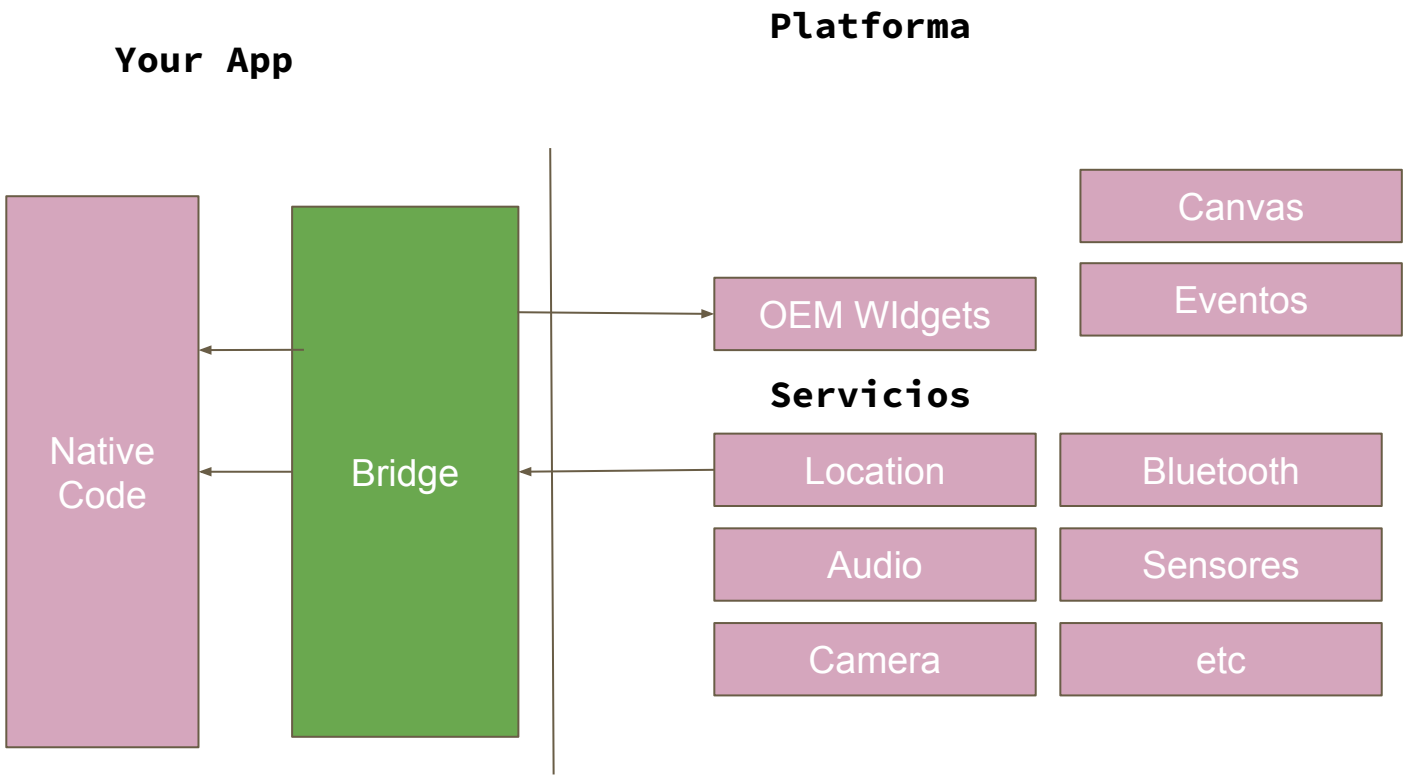

Native Code
OEM Widgets
Canvas
Eventos
Location
Bluetooth
Audio
Sensores
Camera
etc

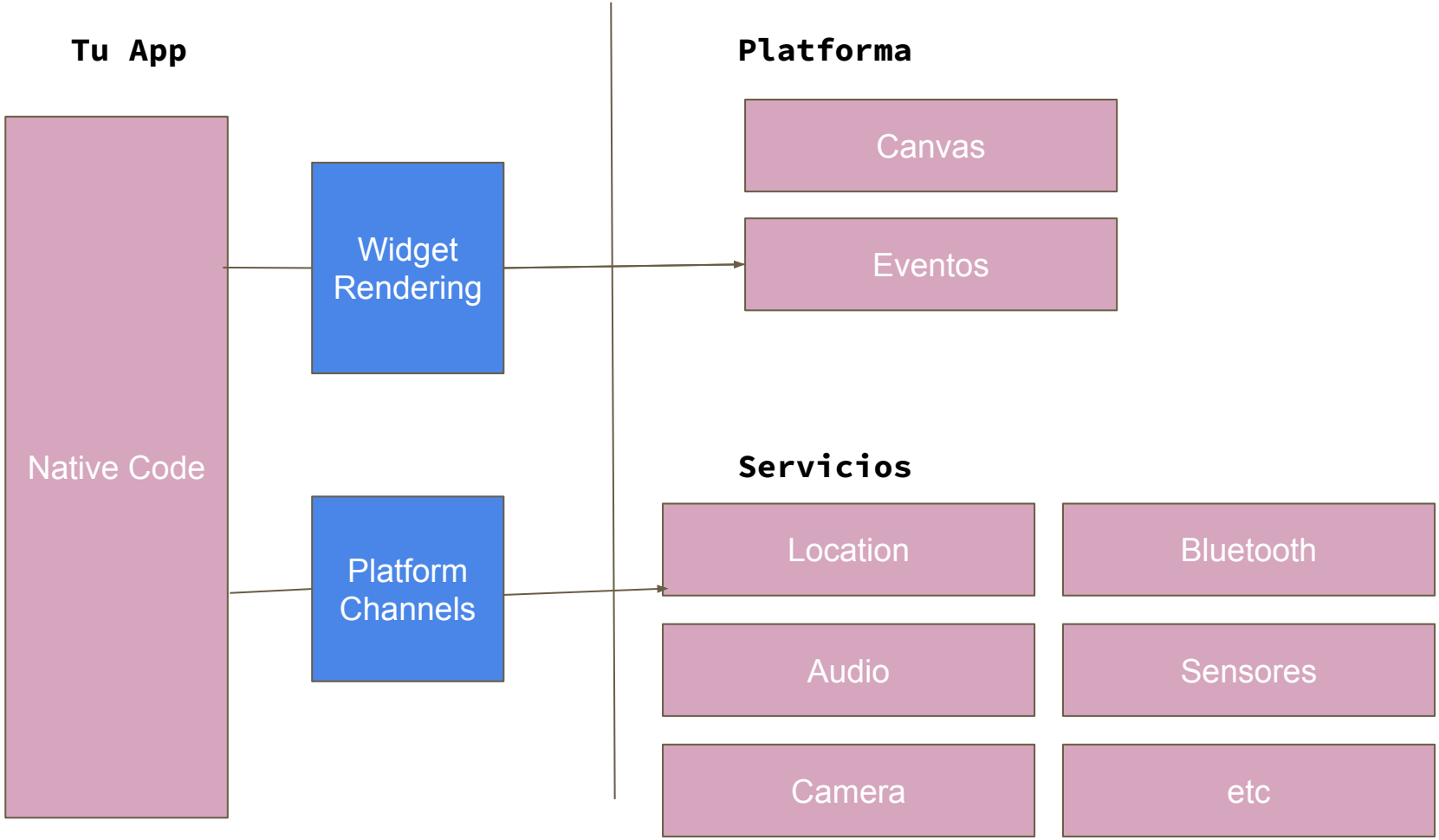
iOS


Native Code
OEM Widgets
Canvas
Eventos
Location
Bluetooth
Audio
Sensores
Camera
etc

Your App

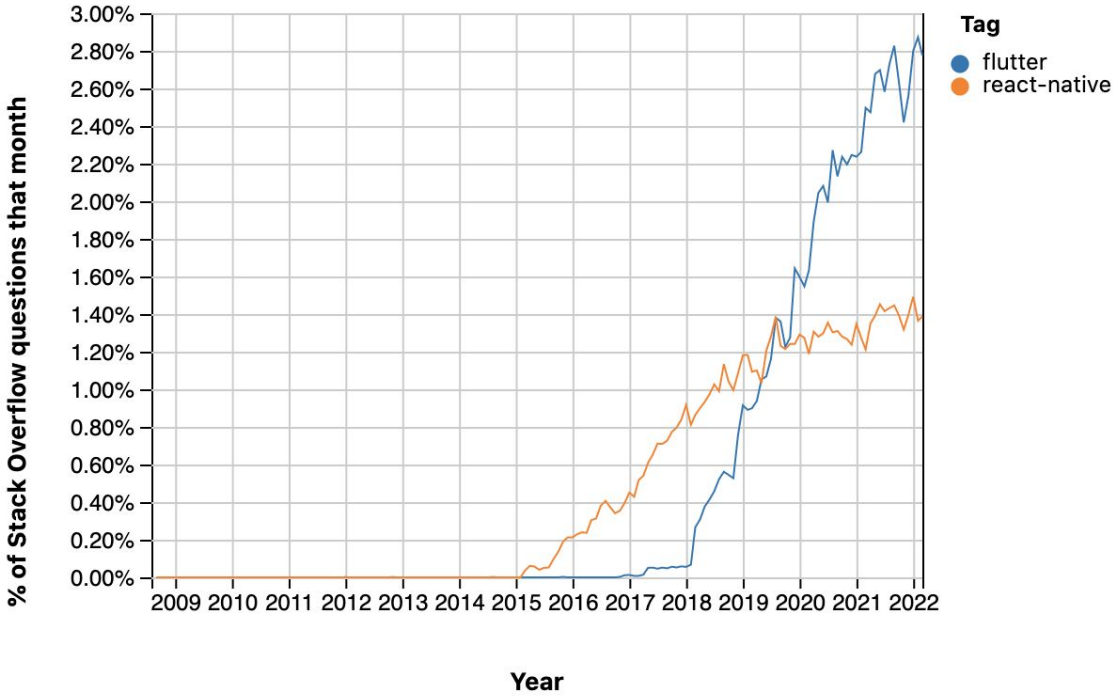

Native Code
Bridge
WebViews
Canvas
Eventos
Location
Bluetooth
Audio
Sensores
Camera
etc


Your App
Platforma
Native Code
Bridge
OEM WIdgets
Canvas
Eventos
Servicios
Location
Bluetooth
Audio
Sensores
Camera
etc


Tu App
Platforma
Canvas
Widget Rendering
Eventos
Native Code
Servicios
Location
Bluetooth
Platform Channels
Audio
Sensores
Camera
etc

Por que Flutter?

- Control sobre el rendering stack
- Reactive Views sin utilizar un bridge
- Hot Reload (JIT)
- UI predecible y rapido (AOT)
- Múltiples plataformas con un código fuente

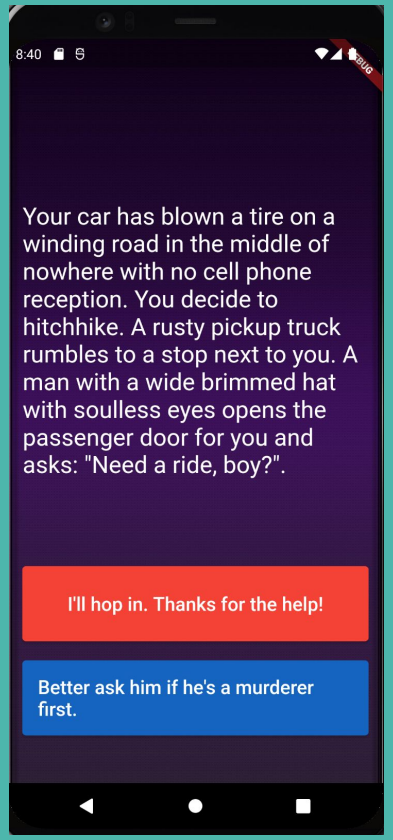


https://insights.stackoverflow.com/trends?tags=flutter%2Creact-native

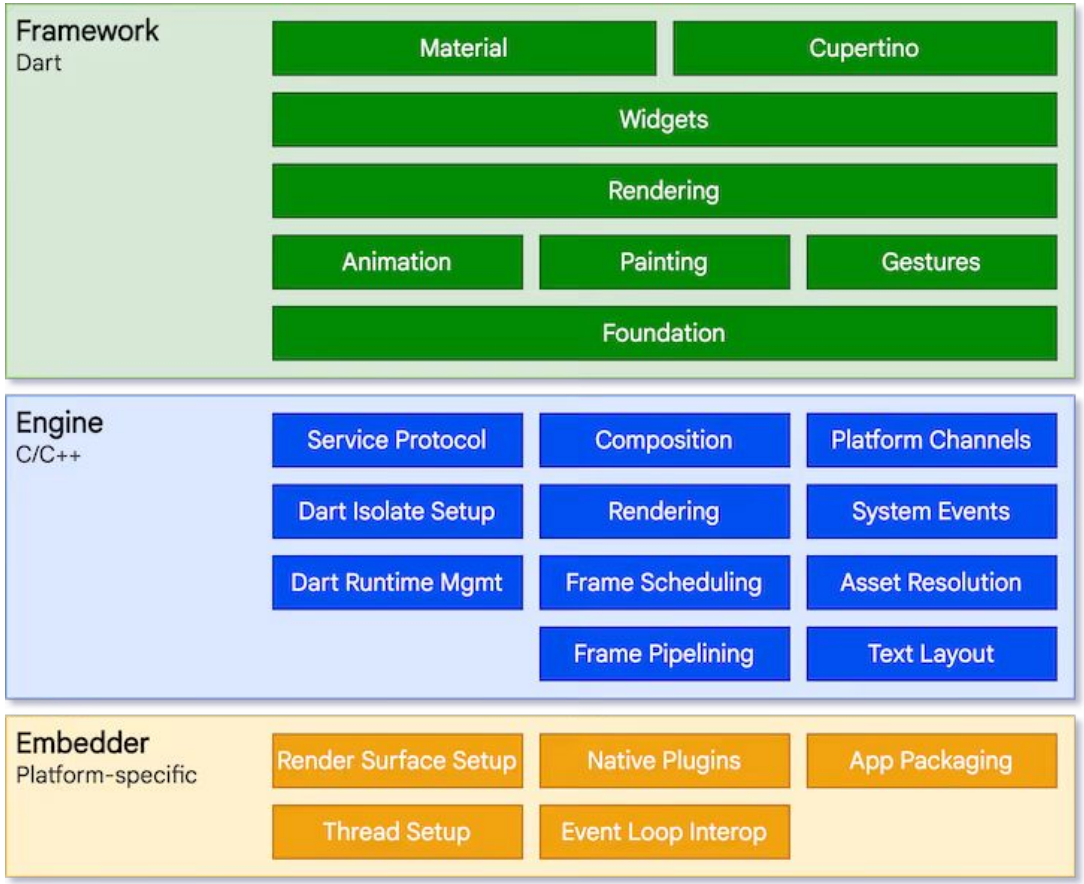
Empecemos

Corriendo nuestro código de git

https://github.com/zezzi/flutter-wwcode-choose-your-adventure



Tooling Y Organización del proyecto


Framework
Dart
Material
Cupertino
Widgets
Rendering
Animation
Painting
Gestures
Foundation
Engine
C/C++
Service Protocol
Composition
Platform Channels
Dart Isolate Setup
Rendering
System Events
Dart Runtime Mgmt
Frame Scheduling
Asset Resolution
Frame Pipelining
Text Layout
Embedder
Platform-specific
Render Surface Setup
Native Plugins
App Packaging
Thread Setup
Event Loop Interop

Widgets

Widget Class

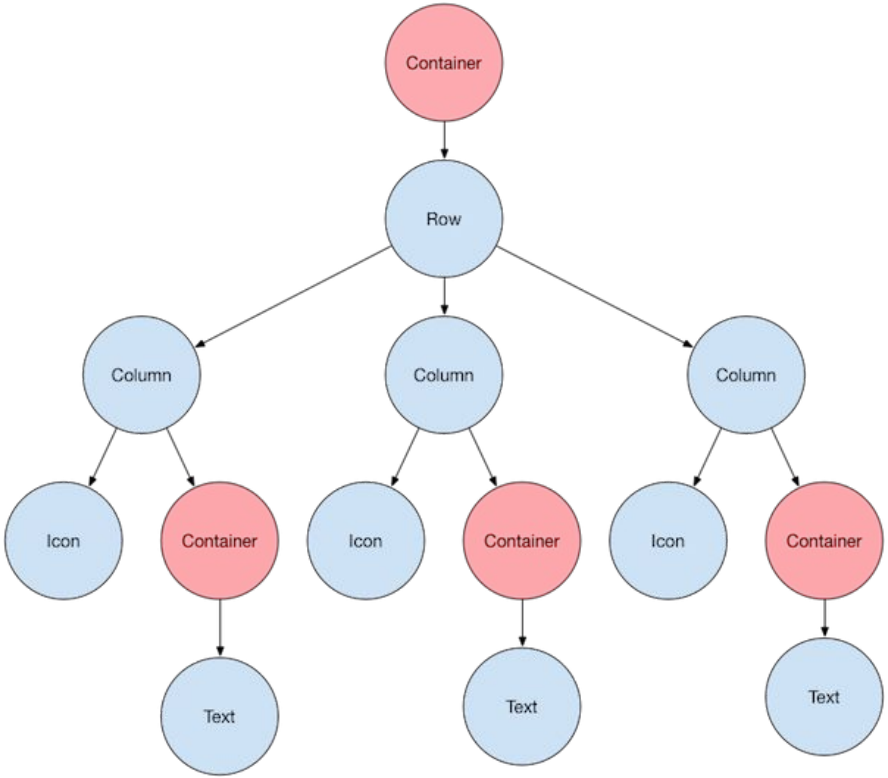
“Widgets are classes used to build UIs”

“A widget is an immutable description of part of a user interface.”

Widgets

“The key property controls how one widget replaces another widget in the tree. “

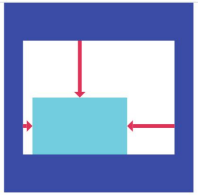
Tree of <Object> en Flutter Widgets



https://docs.flutter.dev/development/ui/layout

Layout

Single-child layout widgets

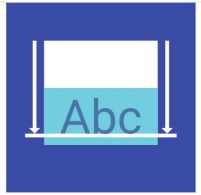


Align

A widget that aligns its child within itself and optionally sizes itself based on the child's size.

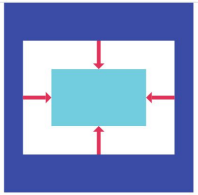
AspectRatio

A widget that attempts to size the child to a specific aspect ratio.



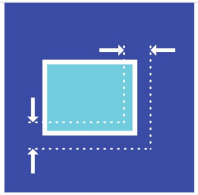
Baseline

A widget that positions its child according to the child's baseline.



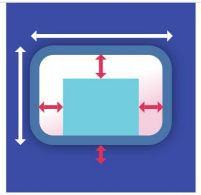
Center

A widget that centers its child within itself.



ConstrainedBox

A widget that imposes additional constraints on its child.



Container

A convenience widget that combines common painting, positioning, and sizing widgets.

Layout

Multi-child layout widgets

Column

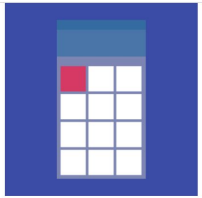
Layout a list of child widgets in the vertical direction.

CustomMultiChildLayout

A widget that uses a delegate to size and position multiple children.

Flow

A widget that implements the flow layout algorithm.



GridView

A grid list consists of a repeated pattern of cells arrayed in a vertical and horizontal layout. The GridView widget implements this component.

IndexedStack

A Stack that shows a single child from a list of children.

LayoutBuilder

Builds a widget tree that can depend on the parent widget's size.

Packages

- Esto permite construir rápidamente una aplicación sin tener que desarrollar todo desde cero.
- .yaml
- https://pub.dev/

Material / Cupertino

Material es un sistema de diseño creado por Google para ayudar a los equipos a crear experiencias digitales de alta calidad para Android, iOS, Flutter y la web.

Material

Cupertino

Assets

Material Icons

Event Handling

Dart Basics

Variables

Functions

Loops/ Conditionals

String Interpolation

Clases

Constructors

Stateful vs Stateless Widgets

It has three trees.

- Widget Tree
- Element Tree
- Render Tree

- Aprende sobre los Trees que utiliza Flutter para manejar el UI

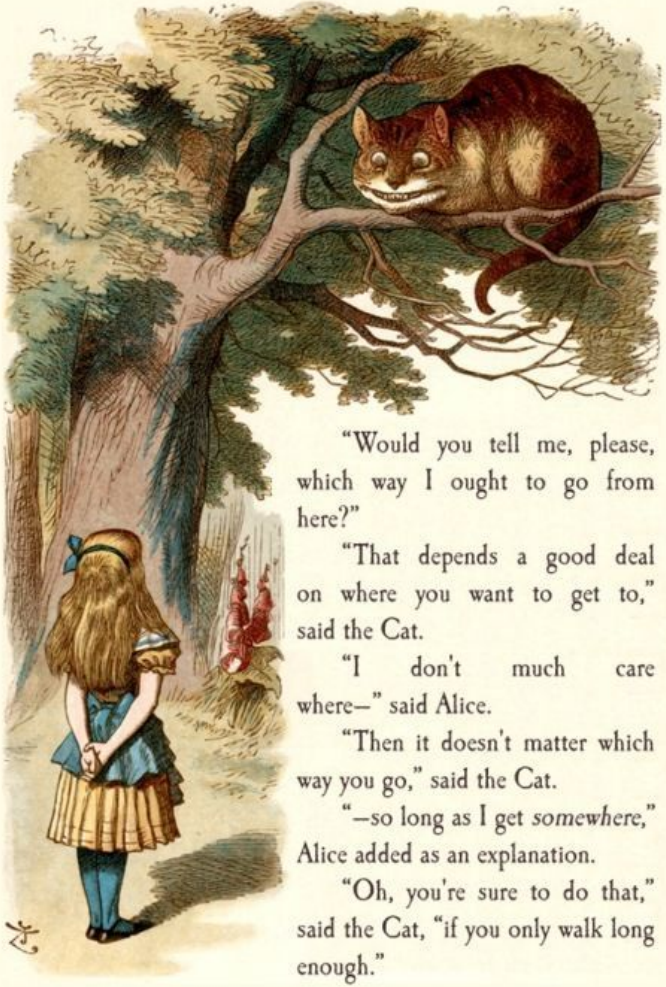
Managing State

Comprensible y de fácil lectura

Testeable

Performance

Lifting state UP (React inspiration)



"Would you tell me, please, which way I ought to go from here?"

"That depends a good deal on where you want to get to," said the Cat.

"I don't much care where—" said Alice.

"Then it doesn't matter which way you go," said the Cat.

"—so long as I get *somewhere*," Alice added as an explanation.

"Oh, you're sure to do that," said the Cat, "if you only walk long enough."

Women Techmakers

Tus Primeros pasos en el desarrollo Móvil con Flutter

GRACIAS

@zezzi

@zezzi

Links

Repositorio

https://github.com/zezzi/flutter-wwcode-choose-your-adventure

Comunidad en Guatemala

https://www.facebook.com/fluttergt

https://www.facebook.com/androidgt

Roadmap de Flutter Subjetivo pero hay muy buenos links

https://github.com/olexale/flutter_roadmap

Desarrollo

https://dartpad.dev/

https://pub.dev/