



# INTRODUZIONE A FLUTTER

---

Il framework multiplatforma definitivo è qui?



# Bonacina Alberto

@polilluminato



By day...

- Software Engineer
- Backend Developer
- Flutter Developer
- Presidente Commissione ICT



By night...

- **Flutter enthusiast**
- Free Software activist
- Linux User

# Obiettivi

- Introduzione a Flutter
- Caratteristiche, vantaggi e limiti
- Applicazione Demo

# Parole Chiave

Multipiattaforma	Widget	Hot Reload
Package	Documentazione	Dart
IDE	Adaptive	Responsive
Monetizzazione	CI/CD	Full Stack



# Applicazione Multiplatforma

Un'applicazione multiplatforma è un'applicazione progettata per **essere eseguita su più piattaforme**, come Android, iOS, il Web e Desktop. Può essere sviluppata utilizzando diversi linguaggi di programmazione, strumenti e framework che consentono agli sviluppatori di **creare una sola volta e distribuire su più piattaforme**.



# App - Web App - Software - Applicazione

- **App**: abbreviazione di "applicazione", si riferisce a qualsiasi software progettato per essere eseguito su un **dispositivo mobile**, come uno smartphone o un tablet
- **Web App**: applicazione a cui si accede tramite un **browser** web e che viene eseguita su un server
- **Software**: programma o insieme di programmi progettati per svolgere una funzione o un insieme di funzioni specifiche
- **Applicazione**: sinonimo di app, software, programma progettato per svolgere funzioni specifiche su vari dispositivi e piattaforme



Framework open source di Google per la creazione di  
**applicazioni multiplatforma** compilate in modo nativo da  
**un'unica base di codice**



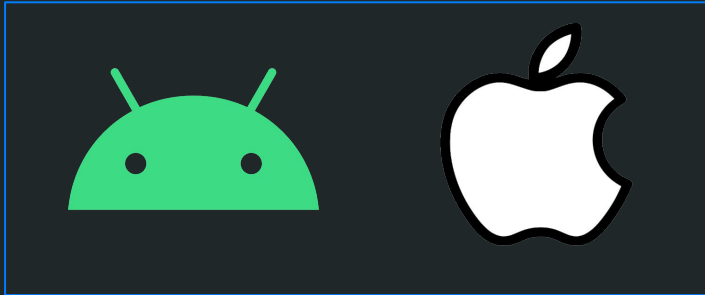
# Quali applicazioni?

	
Interazione diretta con l'utente	API (?)
Visuali	Microservizi (?)
Touch - Click - Tastiera	Server (?)



# 🤔 Quali piattaforme?

mobile



web



desktop



# Challenge sviluppo multipiattaforma

- Molteplici team/sviluppatori (1 X piattaforma)
- Molteplici linguaggi di programmazione (1 X piattaforma)
- Allineamento funzionalità
- Molteplici stili nella UI
- Aumento costi e tempi di sviluppo (costo X piattaforma)
- Manutenzione per piattaforma

## □ Vantaggi con Flutter

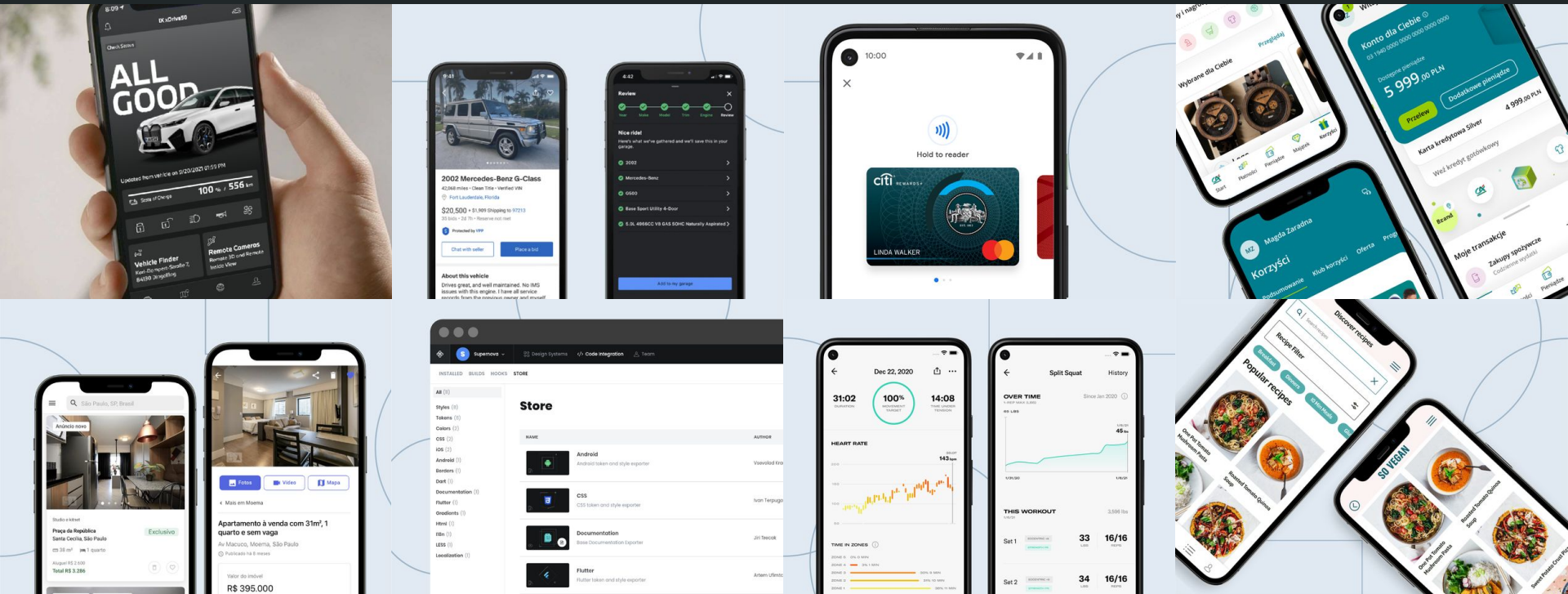
- Unica base di codice scritta in Dart
- Compilazione nativa sulle piattaforme supportate
- UI consistente e feature parity
- Maggiore velocità nello sviluppo di nuove funzionalità
- Diminuzione tempi di sviluppo
- Assistenza e bugfix più rapidi

# Piattaforme di sviluppo e target

Sviluppo\Target	Android	iOS	Web	Windows	macOS	Linux
macOS	✓	✓	✓	✗	✓	✗
Windows	✓	✗	✓	✓	✗	✗
Linux	✓	✗	✓	✗	✗	✓

😲 Chi lo usa?

BMW, Google Pay, Alibaba, eBay, Nubank, Tencent, Toyota, Crédit Agricole, Abbey Road Studios...





# Perchè lo usano?

98.3%

codice condiviso  
(Ebay)

35%

riduzione linee di codice  
(Google Pay)

90%

codice multiplatforma  
(Tencent)

20%

riduzione costi  
di manutenzione (Push)

100%

miglioramento tempo di sviluppo  
(Crowdsourcing)

80%

miglioramento efficienza nel debug  
(Tencent)

# Parole chiave: **Widget**

- Tutto è un widget
- Componibili
- Estendibili
- Più di 215 widget



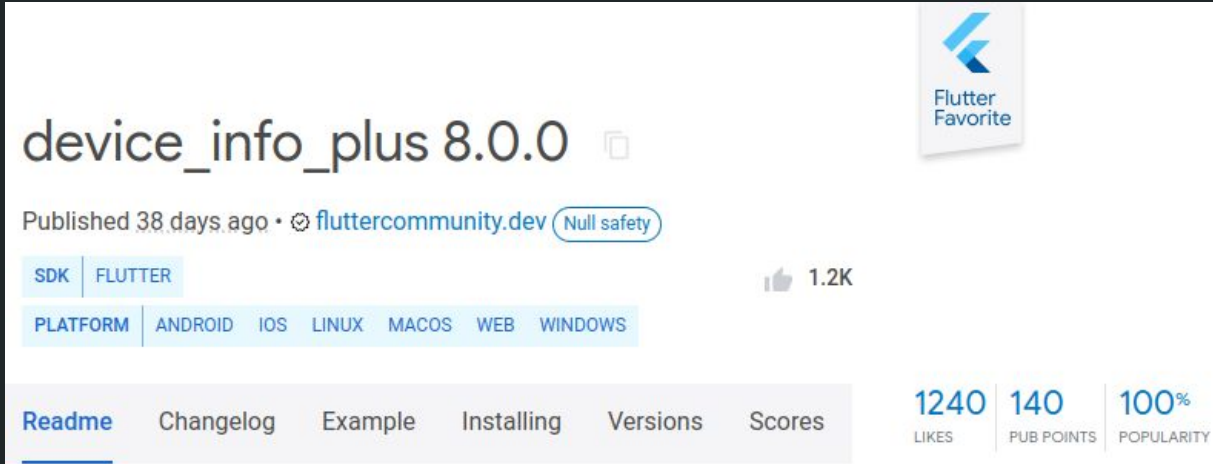
Parole chiave: **Hot Reload** 





# Parole chiave: **Package**

- Estendere le funzionalità
- Pub.dev
- Flutter Favourite
- Piattaforme supportate



The screenshot shows the Flutter package page for `device_info_plus 8.0.0` on Pub.dev. The package is published by `fluttercommunity.dev` and is marked as "Null safety". It has 1.2K likes. The page includes tabs for SDK (Flutter), Platform (Android, iOS, Linux, MacOS, Web, Windows), and a bottom navigation bar with links to Readme, Changelog, Example, Installing, Versions, and Scores. On the right, there is a "Flutter Favorite" badge and a summary of metrics: 1240 Likes, 140 Pub Points, and 100% Popularity.

Metric	Value
LIKES	1240
PUB POINTS	140
POPULARITY	100%

# Parole chiave: Documentazione

- Getting Started, Layouts, Animazioni, Navigazione e Routing
- Catalogo Widget
- Deployment e Pubblicazione Store
- DartPad

## Flutter documentation

### Get started

Set up your environment and start building.

### Widgets catalog

Dip into the rich set of Flutter widgets available in the SDK.

### API docs

Bookmark the API reference docs for the Flutter framework.

### Cookbook

Browse the cookbook for many easy Flutter recipes.

### Samples

Check out the Flutter examples.

### Videos

View the many videos on the Flutter YouTube channel.

## Dart documentation

Welcome to the Dart documentation! For a list of changes to this site—new pages, new guidelines, and more—see the [What's new page](#).

Here are some of this site's most visited pages:

### Language samples

A brief, example-based introduction to the Dart language.

### Language tour

A more thorough (yet still example-based) introduction to the Dart language.

### Effective Dart

Best practices for building consistent, maintainable, efficient Dart code.

### Library tour

An example-based introduction to the major features in the Dart SDK's core libraries.

### Dart SDK

What's in the SDK, and how to install it.

### Futures, async, await

How to write asynchronous Dart code that uses futures and the `async` and `await` keywords.

# Parole chiave: Dart 🧐

- Linguaggio tipizzato ad oggetti
- Compila per ARM, x64 e JavaScript
- Ereditarietà, Interfacce, Abstract, Mixins
- Null Safe
- Async-Await

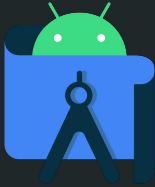
```
class Animal {  
  
    String name;  
    int legs;  
    bool canFly;  
  
    Animal({required this.name,  
           required this.legs,  
           this.canFly = false});  
}
```

```
void main() {  
  
    List<Animal> animals = [  
        Animal(name: "Lion", legs: 4),  
        Animal(name: "Penguin", legs: 2),  
        Animal(name: "Eagle", legs: 2, canFly: true),  
    ];  
  
    for (Animal animal in animals) {  
        print('I\'m a ${animal.name}, ' +  
              'I\'ve got ${animal.legs} legs' +  
              ' and I ${animal.canFly ? "can" : "cannot"} fly');  
    }  
}
```

Parole chiave: **IDE** 



Visual Studio Code



Android Studio



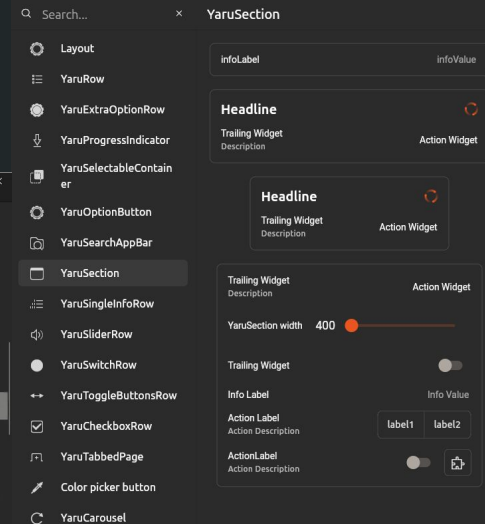
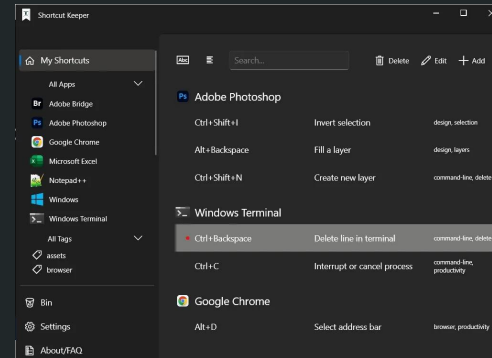
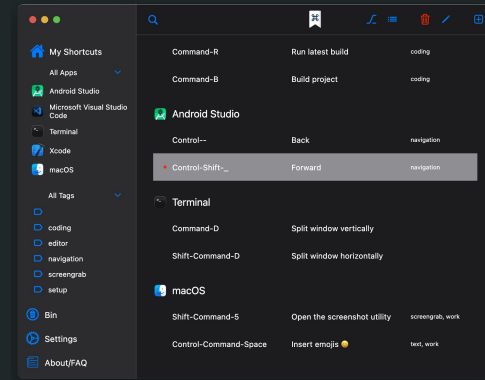
IntelliJ IDEA



Editor + Terminale

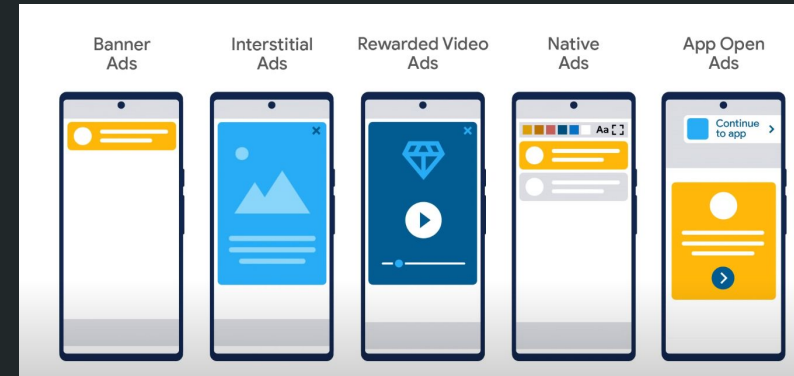
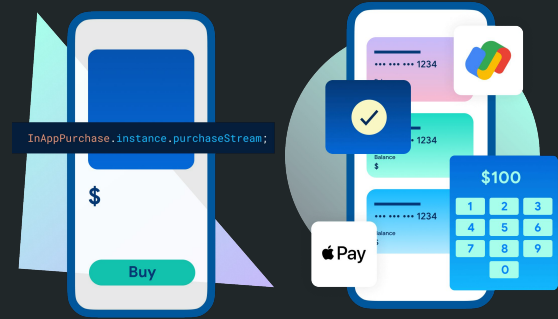
# Parole chiave: **Adaptive e Responsive** 🦄

- Adaptive: adattarsi allo stile e alle aspettative degli utenti di una piattaforma specifica
- Responsive: la disposizione degli elementi cambia in base alle dimensioni o all'orientamento dello schermo



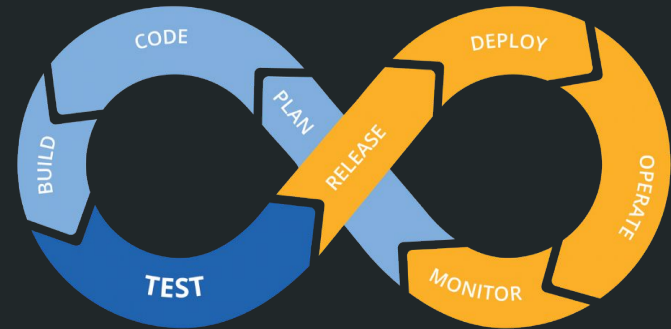
# Parole chiave: **Monetizzazione** 💰

- Google Mobile Ads e Ad Manager
- Banner, interstitial, native e rewarded Ads
- In-app purchase
- Google and Apple Pay
- Stripe & RevenueCat



# Parole chiave: CI/CD 🤖

- Develop -> Test -> Release
- Build automatiche basate su tag
- Test e analisi del codice
- Build sulle piattaforme supportate
- Code signing e pubblicazione su Store



# Full Stack Dart? 🙌

- Applicazione scritta in Dart lato server
- Condivisione model, classi
- Minori problemi di serializzazione
- API, servizi REST, microservizi



Dart Frog



Serverpod





Stop talking!  
Show me the code

# Domande?



[albertobonacina.com](http://albertobonacina.com)



[linkedin.com/in/bonacinaalberto/](https://linkedin.com/in/bonacinaalberto/)



[bonacina.alberto@gmail.com](mailto:bonacina.alberto@gmail.com)



[@polilluminato](https://twitter.com/polilluminato)



[@polilluminato](https://github.com/polilluminato)



[@polilluminato@fluttercommunity.social](https://fluttercommunity.social/@polilluminato)