Flutter

The business value of Flutter

A guide for enterprise application development leaders

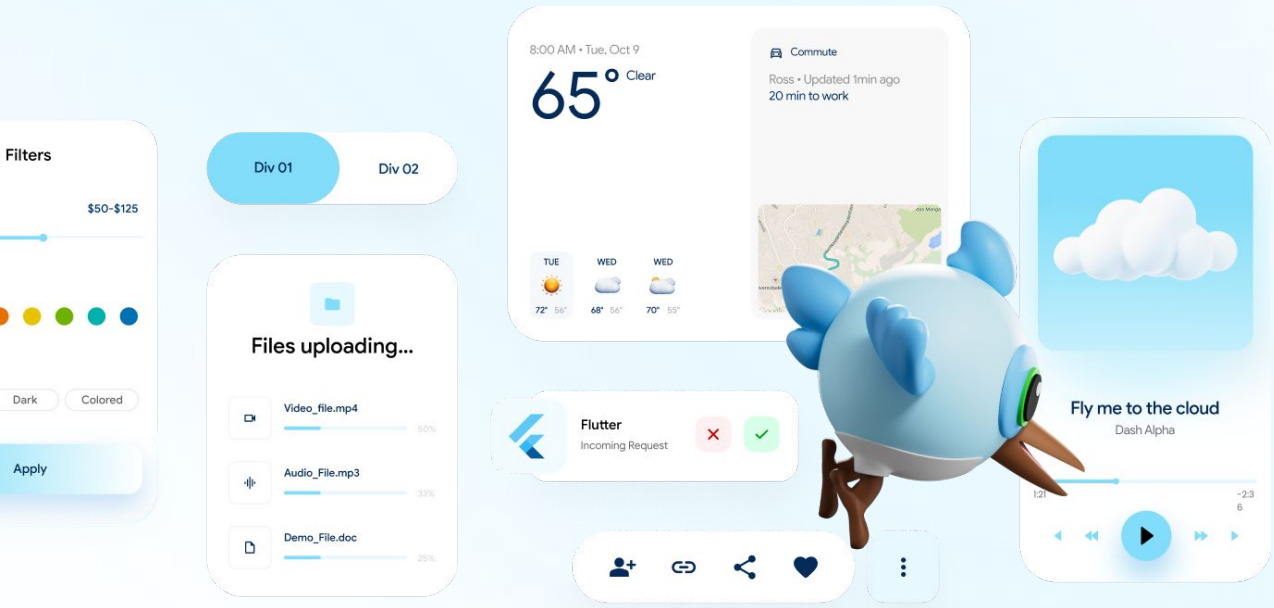
Contents

What is Flutter?

Flutter is an open-source UI framework started by Google

Flutter is a comprehensive, open source, and multi-platform UI framework. Sponsored and developed by Google, it delivers high performance applications anywhere a device paints pixels. It has all the essentials, including a modern programming framework, a rich ecosystem of libraries, robust developer tools and IDE integrations, out of the box support for the Material and Cupertino design languages, and so much more.

Flutter takes a declarative approach to UI composition that is familiar to React, Swift UI, and Jetpack Compose developers. In addition, Flutter ships with rich libraries for layout, animation, text, graphics, and input. This means your teams get up and running quickly building beautiful and feature-rich apps.



Under the hood: The Dart advantage

Flutter and all applications using it are written in Dart.

Dart launched in 2013, featuring an event loop model for asynchrony, object-oriented programming principles, and a clear, typed syntax. For browser-based environments, Dart also compiles to pure JavaScript or Wasm. Taken together, Dart is a high performance language tailor-made to develop, compile, and ultimately execute UI-based applications.

Dart is Flutter's cheat code!

Dart is uniquely optimized to deliver Flutter's needs, including:

JIT (just-in-time) compilation: Powers the "hot reload" experience, enabling you to see code changes instantly without losing state.

AOT (ahead of time) compilation: Produces native machine code for performant release builds.

Sound null safety: Enables you to catch runtime errors at compile-time, increasing app reliability.

Tree-shaking: Removes unused symbols from a finalized Dart binary, ensuring the smallest possible binary size.

Web compilation: Dart code compiles to JavaScript and Wasm, bringing Flutter apps to browsers.

A "batteries included" development experience

Flutter's complete, integrated suite of tools accelerates every stage of the development lifecycle.

Flutter provides a cohesive, officially supported workflow out of the box. From the first line of code to the final release build, the tooling is designed to help you deliver high quality apps while keeping you in a flow state.

Stateful Hot Reload

Powered by Dart's JIT (Just-In-Time) compilation, Hot Reload enables you to inject code changes into a running application instantly, preserving the app's state.

Static Analysis

The Dart Analyzer runs continuously as you type, leveraging sound null safety to catch potential runtime errors before the code even compiles.

Deep Performance Profiling

Flutter's DevTools support performance instrumentation, CPU and memory snapshotting, UI inspection, and full debugging, helping engineers ensure 120 fps performance.

Comprehensive IDE Support

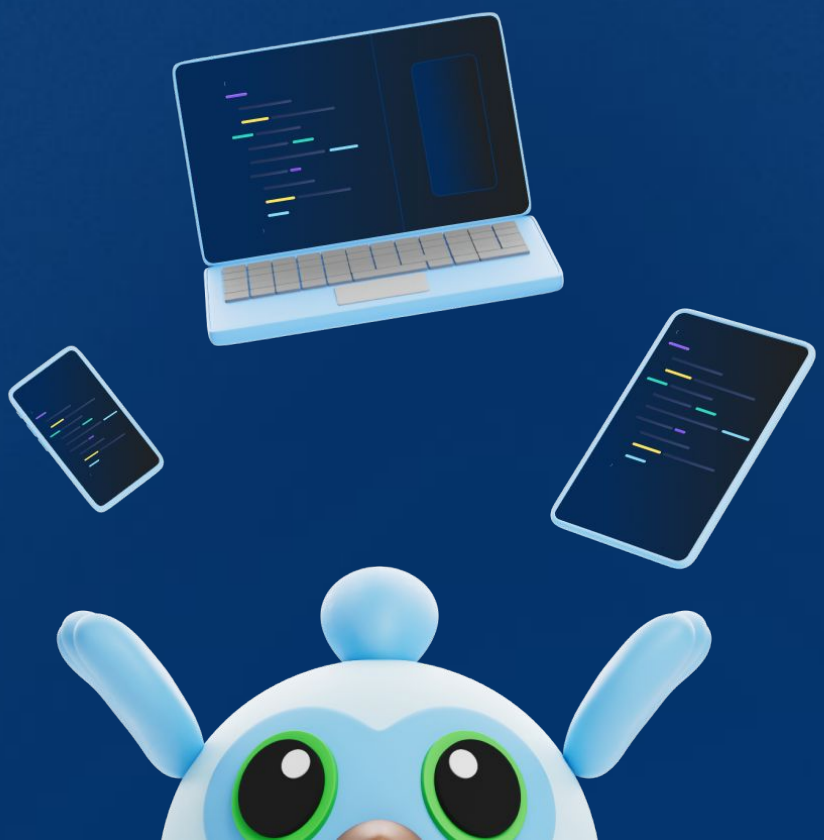
Flutter integrates with the industry's most popular IDEs, including VS Code, Antigravity, Cursor, IntelliJ, Android Studio, and more; providing automated refactoring, dev tools, and intelligent code completion.

Why multi-platform?

Despite code authorship being functionally instantaneous, the old multi-platform challenges of alignment, release schedules, and multiplicative QA burdens remain. Even if your AI agents generate code for multiple platforms, you still have to verify all that code.

"In an era where new code is cheap, trusted code is everything."

With agentic coding, time is money both figuratively and literally. Not only do agents often deliver working code more quickly and cheaply in Flutter, but you can verify it once and deploy it on every platform you target.



Why multi-platform?

Flutter solves fragmentation via consolidation. Flutter apps average 97% code sharing, meaning that business logic, UI layout, and tests are created and verified simultaneously for each platform. Through these and other benefits, Google has consistently measured a **3X productivity increase from teams that adopt Flutter.**

Vibe once, run everywhere.

Whether you vibe code or use LLMs as a skilled pair programmer, Flutter is the only framework that guarantees pixel-perfect consistency across iOS, Android, the web, desktop, and embedded devices from a single codebase.

How can Flutter help you?

Flutter helps your users by putting smooth, accessible, and beautiful apps in their hands across all platforms. Gone are the days where users on certain platforms had to wait for feature parity if one team fell behind or, worse, you had to make hard choices about which platforms to support due to a lack of capacity. With Flutter, your users get the best experience regardless of the device they choose.

Trusted by

LG

TOYOTA

UNIVERSAL

WeChat

SUPERCELL

Flutter helps your business by fundamentally reshaping the economics of software delivery.

Velocity

Google has measured that its teams that adopt Flutter consistently enjoy a 3X productivity increase.

Revenue

By efficiently building features once for every platform, teams that adopt Flutter have seen significant bumps in Google Ads and Google Play revenue.

Proven Scale

Flutter is trusted by engineering teams behind the world's biggest brands, including LG, Universal Studios, WeChat, and Toyota.

How can Flutter help you?

Teams that adopt Flutter are happy with the decision, based on its 93% positive developer satisfaction rating in our surveys.

Faster iteration

Dart's JIT compiler powers "Stateful Hot Reload", enabling you to instantly splice code changes into a running app. This tight feedback loop doesn't just save time; it keeps you and your coding agents in a flow state, boosting productivity.

Code reuse

Flutter apps share an average of 97% of their code across platforms. This means your team writes business logic, UI layouts, and tests once, dramatically reducing the QA burden across individual platforms.

Unified releases

Flutter lowers the effort required to release features to all users at the same time which reduces confusion, simplifies support, and dampens the technical challenges of juggling separate implementations of your app.

Teams that adopt Flutter consistently report that these factors combine to create a healthier engineering culture.

Bring your visions to life

With Flutter, any platform can render any UI element

Most multi-platform frameworks operate by wrapping native platform controls (like an iOS UIButton or an Android Button). This limits development teams to just the overlapping features of the platforms they target.

Flutter completely bypasses this problem by rendering your UI into a custom canvas with absolute freedom.

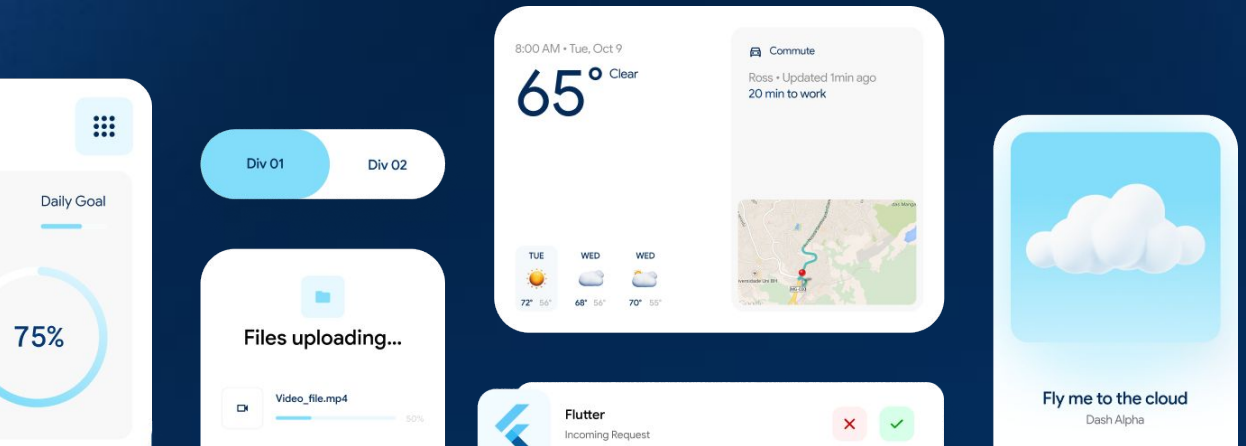
Pixel-perfect control

Because Flutter uses its own rendering engine (Impeller), a button or complex animation can look exactly the same on an iPhone, Pixel tablet, Windows desktop, or a POS kiosk running Linux.

Brand integrity

With Flutter, there is no need to dilute your brand to fit platform constraints. Whether you need a highly stylized, game-like interface or a strictly compliant "native" look, Flutter renders your UI faithfully at 60 or 120 FPS.

With Flutter, your designers are free to dream and your developers are fully equipped to deliver.



Who builds Flutter?

Flutter was started by Google and has thousands of contributors across the globe

Google powers Dart and Flutter with a team of dedicated engineers. Additionally, the ecosystem draws thousands of annual contributions from independent developers and companies.

Beyond the six primary target platforms that Google maintains—Android, iOS, macOS, Windows, Linux, and Web—global hardware giants develop their own custom embedders to run Flutter on specialized devices:

SAMSUNG	**Samsung** maintains the Flutter embedder for Tizen OS.
LG	**LG** maintains the Flutter embedder for webOS.
Canonical	**Canonical** drives Flutter support for Windows, macOS, and Linux desktop apps.
TOYOTA	**Toyota** leads the development of Flutter for Automotive Grade Linux.

From your wrist to your car dashboard, diverse teams using Flutter are united by a single goal: running your code anywhere a screen paints pixels.

Flutter is a thriving open source ecosystem

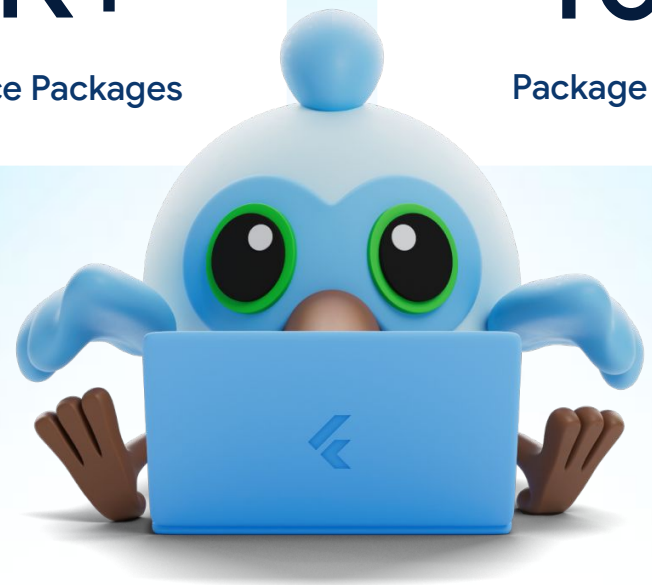
The Flutter and Dart ecosystem are mature. Its official package repository, pub.dev, hosts over 55,000 packages and bridges the gap between multi-platform UI and native functionality. From third-party service integrations to architectural utilities, pub.dev empowers you to build high-performance applications with great speed and reliability.

>1.3B
30-day package downloads

1.7K+
OSS Contributors

55K+
Open Source Packages

10K+
Package publishers



An emerging industry of tooling providers

Flutter is supported by a thriving market of service providers

A healthy ecosystem must support the entire software lifecycle, not just development. Flutter is supported by a rich array of dedicated service providers handling critical operations like CI automation, deployment, and analytics. These solutions integrate directly with Flutter, enabling your teams to offload infrastructure maintenance and focus on delivering business value.

Shorebird provides code-push to Flutter apps, enabling developers to quickly ship important OTA updates without getting bogged down by App and Play Store reviews.

Serverpod provides one-click deployments for its cohesive development framework, including full-stack type-safety.

Widgetbook provides automated UI testing, verifying brand fidelity, UI responsiveness, and accessibility.

Flutterflow provides a world-class WYSIWYG low-code editor, spanning both UI and business logic.

And much more!

Who uses Flutter?

From global brands to startups, developers have voted with their code

Google bets its business on Flutter. Google has invested in Flutter for over a decade, not just as an open-source project, but as a critical framework for its own revenue. To date, over 30 teams across Google build with Flutter, including:

- **Revenue Critical:** Google Pay, Google Ads, and Google One.
- **AI Native:** NotebookLM and the Gemini App.
- **Mass Scale:** YouTube Create, Google Classroom, and Google Earth.

Who uses Flutter?

Flutter is the market leader in multi-platform development

Outside of Google, Flutter has been the world's most used multi-platform framework since 2021.

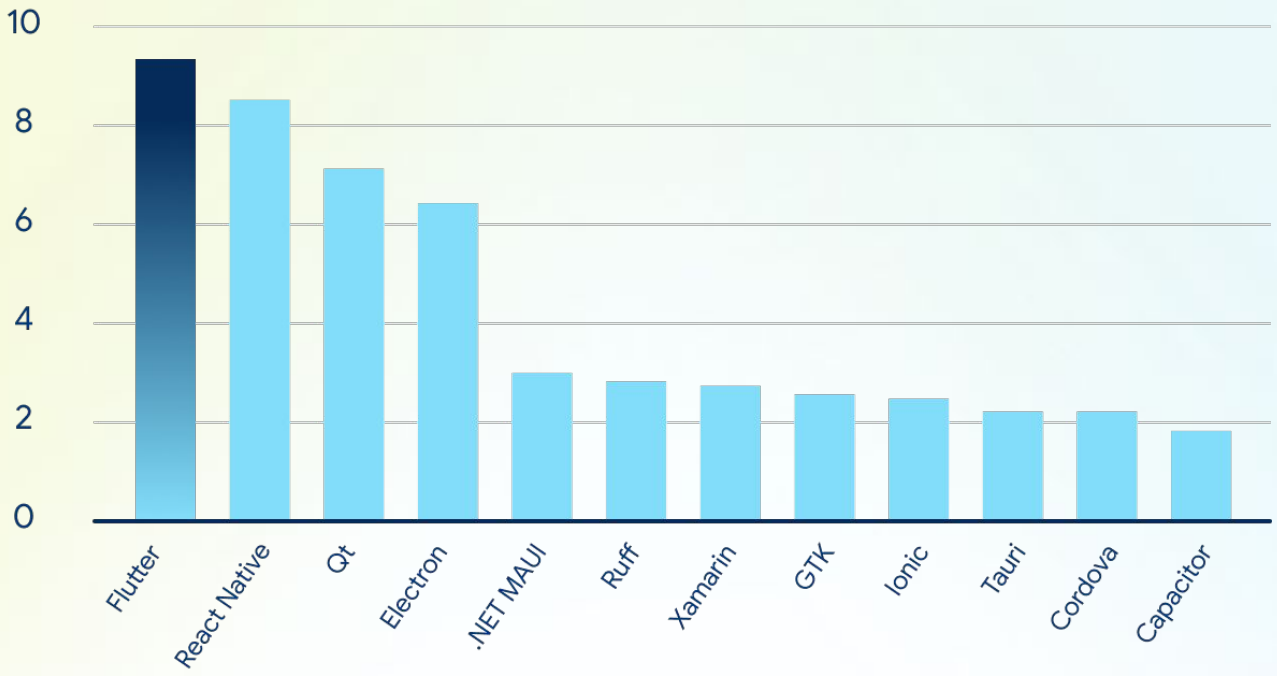
Volume

It powers **1 million+ apps** on the Play Store and accounts for nearly **30% of all new free apps** on the Apple App Store.

Trust

It is the engine of choice for banking (Nubank), automotive (BMW, Toyota), and social giants (WeChat).

Put simply: If you have a smartphone, you already use Flutter every single day.



Source: https://survey.stackoverflow.co/2024/technology/

Full-stack Dart?

Dart is one of few languages capable of handling your entire development stack

Dart enforces consistency and type-safety across your entire stack, from backend APIs, to database schemas, and even critical CI/CD automation scripts. This eliminates context switching and ensures that your business logic, validation rules, and deployment pipelines remain perfectly in sync. The following are some ways in which you can build with Dart for your backend:

- **Serverpod:** A batteries-included solution for cohesive full-stack development, offering a robust Postgres ORM and automatic generation of matching client and server-side code.
- **Dart Frog:** A lightweight, minimal server-side framework designed for speed and flexibility, perfect for building fast, scalable APIs.
- **Cloud Native:** With support for Firebase Cloud Functions, Dart is rapidly becoming a language you can use across your whole stack.

How to evaluate Flutter

There are multiple ways to experience the power of Flutter and Dart, whether you are a fresh startup or a massive enterprise.

Zero-install options

dartpad.dev is Dart's public scratchpad and a great place to immediately tinker with Dart scripts or full Flutter apps. DartPad also makes it easy to share code snippets with GitHub Gists.

Learn declarative UIs

Start with the official learning path for a comprehensive overview to learn about Flutter and Dart. You can also sit back and learn Flutter by following its official YouTube channel.

Ecosystem audit

Explore pub.dev for packages and SDKs for the third party tools you use and download the Flutter extension in popular IDEs like Visual Studio Code [1], IntelliJ, or Android Studio.

Cultural assessment

Engage your engineering team early! Adoption requires buy-in, so gauge their interest in adopting a new stack. While Flutter involves a learning curve, 93% of developers report that the long-term gains in productivity and career satisfaction outweigh the initial cost.

[1] *Or its forks, including Antigravity, Cursor, Windsurf, or others.*

How to adopt Flutter

After evaluating Flutter and deciding to move forward, there are two primary adoption strategies:

1

Add-to-app (brownfield):

Best for: Teams with massive, existing applications. This approach is the industry standard for established apps. Instead of rewriting everything, you embed a Flutter module inside your existing codebases.

2

The fresh start (greenfield):

Best for: New products, major rebrands, or apps carrying heavy technical debt. This is the "clean slate" approach. It involves using Flutter for a brand-new project or seizing a moment of transition to rewrite a legacy flagship app.

How to adopt Flutter

1 Add-to-app (brownfield):

How it works

You delegate a tightly scoped set of upcoming features (e.g., a new "Rewards" dashboard or an in-app customer support chat module) to Flutter.

The Trade-off

This strategy minimizes business risk, but adds some initial complexity to your app as you must sync application state across the native-to-Flutter boundary.

The Outcome

Over time, as your team gains confidence, Flutter's scope naturally expands to cover more of your app. Eventually, this can lead to an app fully written with Flutter.

Resources

For technical guidance, visit docs.flutter.dev/add-to-app. You can also check out this showcase from Talabat of how they incrementally migrated to Flutter.

How to adopt Flutter

2 The fresh start (greenfield):

How it works

You stop developing new features in your existing codebase and begin a new project with Flutter.

The Trade-off

In the case of a rewrite, starting over is often viewed as being risky, but Flutter changes the calculus by enabling you to target multiple platforms with the resources typically required for one.

The Outcome

Teams that choose this path often move faster in the long run because they aren't shackled by legacy code.

Proven Success

Google Classroom, Google Earth recently executed this strategy, successfully rewriting their flagship apps in Flutter to unify their engineering efforts under a single, maintainable codebase. Outside of Google, many teams using Flutter have also found success using this strategy. You can hear more about their migration stories by checking out the following links: Reflectly and Kikoff.

Onboarding in days, not months

Dart and Flutter are easy to learn and upskill existing engineers

New engineers often land code on their first day. Because Flutter uses a declarative UI pattern similar to React or SwiftUI and a familiar C-style syntax, the transition is more of a lateral move than a steep learning curve.

- **For web developers:** Dart's asynchronous patterns (async/await) and collection literals feel like a more structured, "sound" version of TypeScript.
- **For mobile and backend developers:** Dart's object-oriented nature means your Java and C# experts can immediately apply their knowledge of classes, interfaces, and design patterns.
- **For new hires:** Flutter's documentation is world-class, and the "Everything is a Widget" philosophy provides a consistent mental model that eliminates guesswork.

Flutter in the AI era

Code authorship in 2026 does not resemble its past self, even as recently as 2024. Today, coding agents do the bulk of the menial work; leaving the high-level thinking to developers, designers, and product managers. As a staple in mobile and web development for a decade, **Flutter is well understood by all of the world's LLMs;** from Claude, to ChatGPT, to Google's own model, Gemini. Flutter developers enjoy cost savings and improved productivity by incorporating AI tools to plan features, iterate on specs, and ultimately write code once, and reach users everywhere.

But it's so much more than that. Flutter and Dart were early to launch a robust MCP server which provides LLMs with Dart's fully sound static analysis for profiling running apps, proposing helpful dependencies, and debugging compile time or runtime errors.

Flutter's agent skills complement its MCP server by providing your coding agents with best practices that help you build scalable Flutter apps, while reducing your token usage.

Dart and Flutter are also at the forefront of Generative UI, an exciting new paradigm by which LLMs produce bespoke interfaces on-demand from a catalog of trusted UI components. Crucially, this works within Dart's optimized binaries, avoiding any runtime interpretation which hurts performance.

You can read more about our approach to AI in 2026 and beyond.

Flutter in the AI era

How Flutter expedites AI coding experiences

The modern ubiquity of LLMs and agents means intelligence is no longer a differentiator. The competitive advantage now lies in the **user experience of AI.**

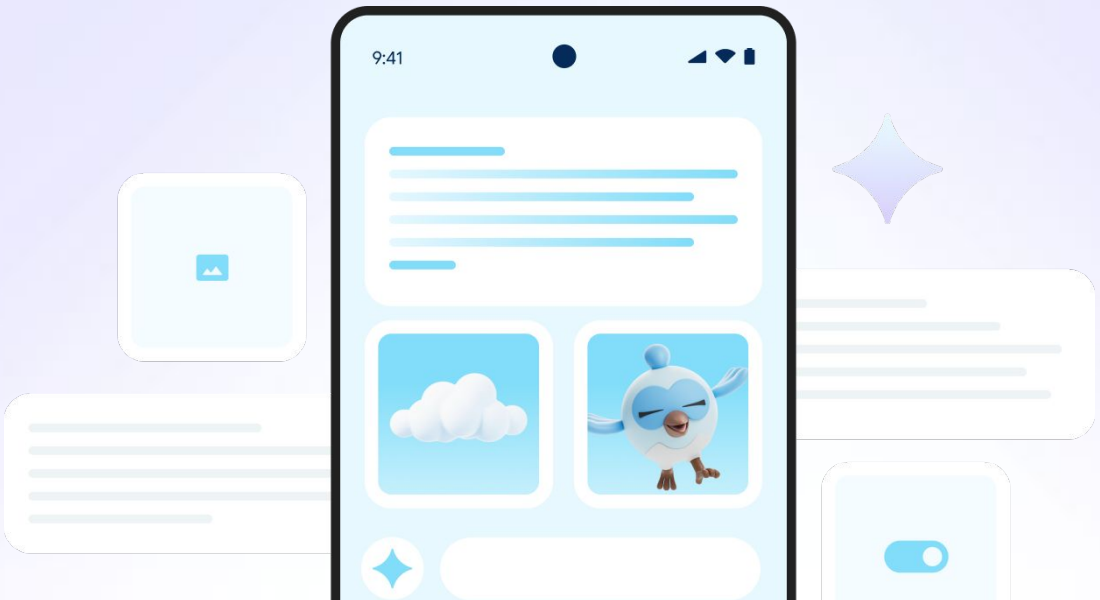
Flutter is uniquely positioned to deliver this experience for multiple reasons.

AI for Business

In your **business logic layer**, Dart's sound type safety acts as a guard rail against AI hallucinations and raises schema mismatch errors during static analysis. This helps your apps consume structured data returned from your server.

AI for Interfaces

In your **interface layer**, generative UI, also known as "GenUI", allows an LLM to generate trusted UI components in novel ways in response to a user's specific context and needs. This helps you guide users through complex, multi-step processes using personalized interfaces.



GenUI with Flutter

Flutter and Generative UI produce rich and dynamic user interfaces.

In the early days of AI, integration was limited to a chat bubble where an LLM simply returned text. Today, GenUI allows your application to assemble its interface in real-time based on the user's intent.

Now, everything has changed. By leveraging the A2UI protocol (developed by Google), Flutter and Dart enable your AI agents to move beyond conversation and become a co-author of your app's frontend.

GenUI with Flutter

How it Works

The foundation of GenUI in Flutter is the Widget Catalog. You define a library of trusted components, your brand's specific buttons, charts, and cards, along with usage instructions. Your AI agent then composes them in to just the right UI for your user at runtime.

Intent recognition:

Your AI agent analyzes the user's query (e.g., "Show me my monthly spending with savings suggestions") and begins composing the necessary UI.

Personalized user interfaces:

Traditionally, building different layouts for different users required manual coding and maintenance. With GenUI, the interface immediately adapts to the individual user's intent and needs.

Branded experiences:

A2UI can only tell your app to render elements out of your widget catalog, meaning the resulting UIs are perfectly on-brand, performant, and secure.

FAQ

Flutter FAQ

How popular is Flutter?

Flutter has >1.5M monthly active developers and powers more than a million apps published to the Google Play Store. On Apple's App Store, 30% of all new free apps are written in Flutter. If you begin to use Flutter, you'll be in good company.

How is Flutter's runtime performance?

Flutter apps compile down to native code, running directly on their host platform without any interpretation or virtualization. This helps Flutter run smoothly at 60+ frames per second.

Do Flutter apps work with screen readers?

Flutter refreshes semantic information with the OS or browser every frame to ensure that screen readers and other accessibility tools work exactly as needed.

Is Flutter viable on iOS?

Yes! Flutter apps have been selected as "Apps We Love" and "Editor's Choice" and accepted to Apple's Foundations Accelerator program within the App Store. Supercell's game "Squad Busters", which won *iPad Game of the Year 2024*, builds its social UI with Flutter. Flutter apps also run beautifully on Apple hardware. You can download the Wonderous app to see how smoothly Flutter handles rich, complex animations on iOS.

Flutter FAQ

Does the Dart ecosystem have a package repository?

Flutter developers have published over 50,000 packages to pub.dev, any of which you can easily add to your projects by configuring your pubsec.yaml file - the central file in any Dart or Flutter application.

Does Flutter have a public roadmap?

Flutter publishes a new roadmap each year. This link is always updated with the latest roadmap details.

When should I not use Flutter?

Flutter is optimized for app-centric experiences. If you are building a text-heavy static website (like a blog or news outlet) where SEO and instant load times are important, then traditional HTML and CSS frameworks are sometimes a better fit. For these scenarios, check out Jaspr, a Dart-based web framework. Similarly, we recommend native toolkits for heavy AR/VR experiences.

Does Flutter impose any design limitations?

It does not. Design freedom is one of Flutter's biggest strengths! Because Flutter paints whatever pixels your heart desires into a native canvas, your developers are able to fully realize your designers' vision. Better yet, nailing the implementation for your custom branding once with Flutter ensures it will always look right on every platform.

Flutter FAQ

Does Flutter include robust developer tools?

Flutter's developer tools help you complete all the typical debugging tasks, including stepping through code, inspecting your UI, profiling CPU and memory usage, tracking network requests, and much more.

Can I develop server-side applications with Dart to share code across my stack?

You most certainly can. Lightweight projects like Dart Frog provide a minimal experience, while Serverpod offers a more batteries-included library with a robust Postgres ORM and matching client and server-side code generation. Firebase users can also use Dart in Cloud Functions for Firebase with the Dart Admin SDK.

Does Flutter integrate with my other tools?

Most third-party tools have strong Flutter libraries, either directly owned by the third-party themselves or by the Flutter community. The best way to assess the coverage of your needs is to visit pub.dev/packages and search for the tools you use.

Are there professional consultancies or firms I can hire to expedite my team's switch?

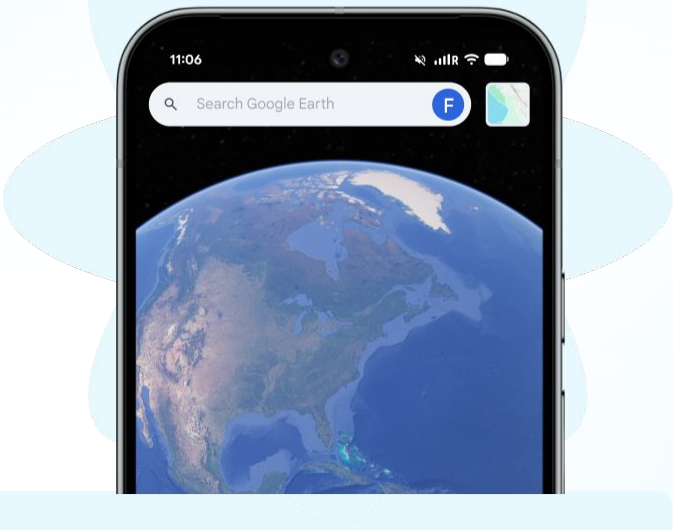
Yes! If you're looking for assistance ramping up on Flutter, you can find over 200 consultancies listed at flutter.dev/consultants.

Case Studies

Tech / Mobile - Web

Google Earth

Using Flutter to deliver powerful and adaptable experiences across web and mobile.



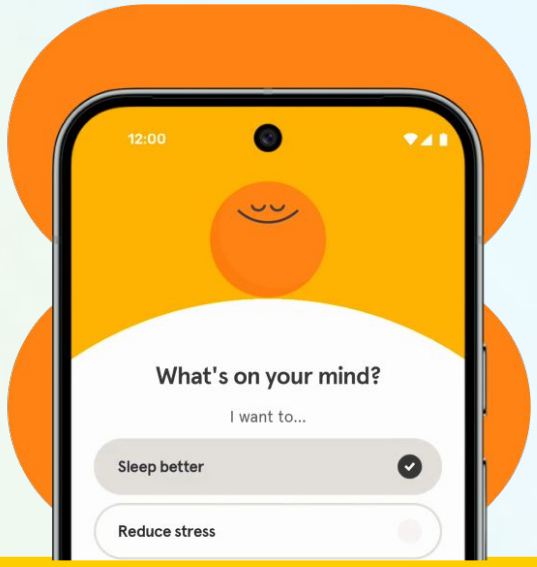
> *Flutter has vastly increased our ability to deliver multi-platform in a timely manner.*

Brian Ellis
Google Earth TL

Health / Mobile - Web

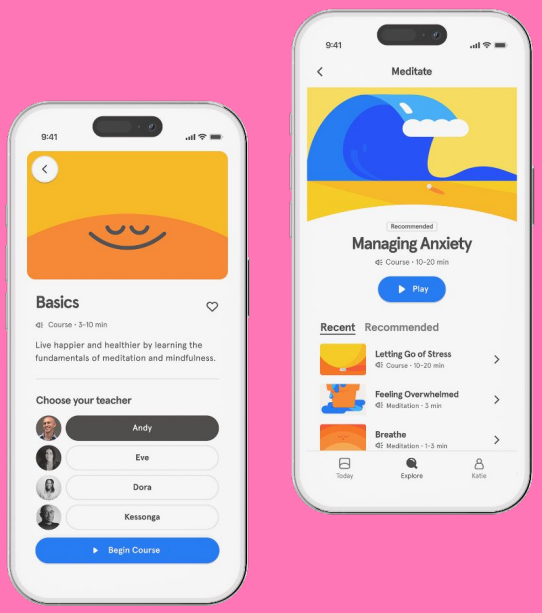
headspace®

By unifying our client architecture and centralizing business logic and analytics, we've seen an estimated 30–40% increase in developer productivity.



> Adopting Flutter allowed us to consolidate previously fragmented platform-specific efforts into a single, cohesive client-side engineering group.

Russell Glass
Headspace CEO



LG webOS

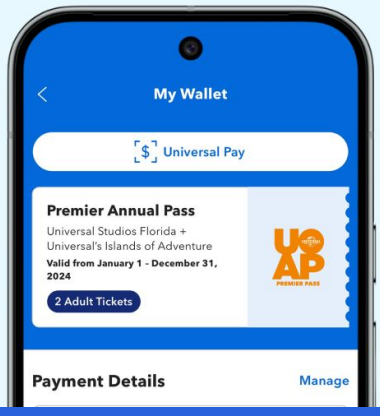
webOS brings Flutter apps to 100s of millions of new devices every year.

> *Without any optimization whatsoever, our Flutter rewrite launched twice as fast as our original app, consumed less runtime memory, and felt more responsive and playful to use.*

Lifestyle / Mobile - Web

UNIVERSAL©

Flutter reduced Universal's codebase size by 45%.



> *When we rolled out our Flutter apps, we quickly saw the number of crashes go down to almost zero.*