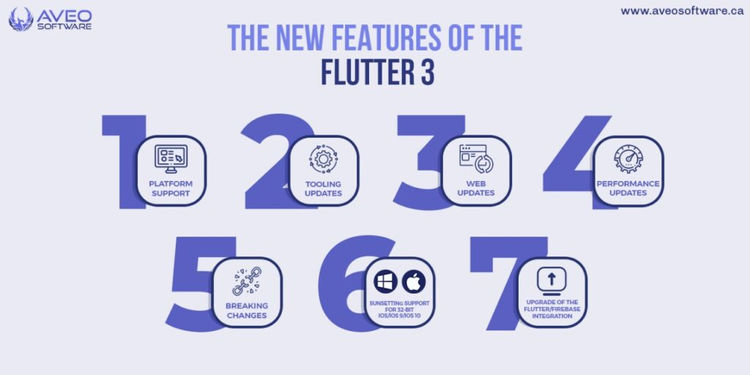
**Google Unveiled Flutter 3 – Here Are All the New Features**



The release of [Flutter 3](https://docs.flutter.dev/development/tools/sdk/releases) by Google increased platform compatibility, added more capabilities to enhance the developer experience, and eventually increased the market reach of the entrepreneurs' apps.

Earlier iterations of Flutter supported Android, iOS, the web, and most recently Windows. These were significant improvements, but Flutter 3 now supports Linux and macOS as well. The most important upgrade included numerous other outstanding features in addition to platform support.

Do you want to know the updates that Flutter 3 brought? Now let's find out!

**The new features of the Flutter 3**

We'll talk about a lot of the changes that Flutter brought about, so let's get to the upgrades and get to business.

**Platform support**

A cross-platform framework gains power as it expands to include additional devices. Furthermore, Flutter currently provides robust support for Linux, macOS, Android, iOS, web, and Windows (supported on Feb 2022). With the help of this functionality, you may produce apps that run on all six platforms using a single codebase.

With the latest Flutter 3, which supports Apple Silicon and Linux, your apps may now run on the six major platforms with natively generated apps.

You can run native apps on both Apple and Intel silicon thanks to the consistent support for macOS, which provides compatibility for both architectures. The ability to support Universal Binary makes this support possible.

The shift to Apple Signal is made possible by the use of [Universal Binary](https://en.wikipedia.org/wiki/Universal_binary), Apple's format for executable files, with apps that run on both Intel machines with 32/64-bit architectures and computers with a 64-bit extension of ARM.

**Tooling updates**

**Multilingual support**

With the revised Flutter localizations package, Flutter 3 made it feasible to localize your apps for different markets. By making your Flutter applications available to users who prefer other languages, you can now expand your user base internationally.

The localizations widget makes it simpler to add additional language support even if US English remains Flutter's default language. If the [list of accepted languages](https://api.flutter.dev/flutter/flutter_localizations/GlobalMaterialLocalizations-class.html) does not contain the language you prefer, you can easily add new ones. To learn how to [add support to your new language](https://docs.flutter.dev/development/accessibility-and-localization/internationalization#adding-support-for-a-new-language) in Flutter, click this page.

Additionally, all three desktop platforms for the new edition fully support the international text [Input Method Editors](https://docs.microsoft.com/en-us/windows/apps/design/input/input-method-editors) (IMEs), including Chinese, Japanese, and Korean.

**Support for accessibility services**

With the latest update for desktop platforms, accessibility features including accessible navigation, screen-readers, and inverted colors are now completely supported.

**Lint package updates**

Dart drives the Flutter. Additionally, this version added a number of new Dart capabilities that boost developer productivity by improving code readability and cutting down on boilerplate sourcing time.

The sophisticated **Dart linter** tool analyzes source code using the available linter rules to assist developers spot potential code problems. It checks for typing errors, coding conventions, and formats, saving you the many hours you would otherwise need to spend attempting to find them manually.

With Flutter 3, Dart and Flutter lints version 2 was made available. As a result, whenever you use Flutter create to create a new app, these versions will now be used automatically.

If you don't switch to the new lint updates, you'll still be utilizing the old lint versions for your current apps. You should upgrade to version 2 using the flutter pub upgrade -major-versions flutter lints command to adhere to the most recent Flutter best practices.

**Impeller engine**

The Flutter team fixed a number of iOS jank issues in 2021, although these fixes still needed to be reconsidered. Thanks to the Impeller engine in this latest Flutter release, it is now possible to preview an experimental rendering backend for iOS. That demonstrates good work from the group responsible for creating the graphics backend.

Impeller prevents compilation while the application is running by precompiling a smaller or simpler set of shaders during engine build time. This significantly reduces jank and boosts speed by more than 20 times.

**Material Design 3 Support**

Flutter now supports opting into Material Design 3 with the introduction of Flutter 3, which also includes support for Material You. You can use this codelab to play around with the new capabilities in Material Design 3 in Flutter: [Bring Beauty to Your Boring Flutter App.](https://io.google/2022/program/7508cd2e-a0a3-4190-adf0-961d21b2424e/)

**Web updates**

**ImageDecoder API Support**

With the release of Flutter 3, image decoding is now accomplished in half the time. Flutter web has been updated to automatically detect the ImageDecoder API in the browser and use it when supported, which has resulted in a speed boost. Thankfully, the API is supported by practically all Chromium-based browsers.

Using the ImageDecoder API has the advantage of asynchronously decoding your images using a built-in image code in the browser. As a result, it perfectly timed off the main thread and removes all picture janks during the procedure without interfering with the operations. As a result, the entire process of decoding images gets quicker.

**Flutter web app initialization**

The lifecycle API is another modification to the Flutter web updates. With this new release, Flutter app developers now have more freedom and control over the bootstrapping process. Monitoring and analyzing a Flutter app's performance on the host HTML page will now be simpler. Now, before your Flutter app has finished loading, you may use an interactive landing page, a splash screen, and a loading indicator.

**Performance Updates**

**Engine’s raster and UI thread prioritization**

Raster and UI threads for the Flutter engine for Android and iOS are now given precedence. The average frame processing time decreases with this upgrade, improving the speed of your program.

The prioritized schedules of the UI threads and the Dart VM garbage collection threads are excellent illustrations of prioritization during multitasking. The UI threads are scheduled to run at a higher level when they are operating concurrently because they have been given a higher priority.

**Opacity animations**

Small-scale opacity animation performance enhancements are included in Flutter 3. The rasterization time on a benchmark's approximate comparisons metric improved, demonstrating the benefits of these enhancements. An illustration of this improvement is the removal of the save layer method when opacity calls it in order to speed up rasterization and enhance the functionality of the program.

**Raster cache**

A new method for deciding whether to render or cache a picture was introduced with Flutter 3. This technique determines if an image is suitable for caching by estimating the cost of its draw operations. Previously, the admission policy performed the identical task using only the draw operations shown in the illustration. Thus, graphics that could render more quickly would use little memory.

**Frame scheduling**

Additionally, there is a bug fix for iOS's dropped frames. There had been numerous problems with this bug reported. The issue with the animation dropping has been resolved by this update, which rectified the frame scheduling error. Additionally, this upgrade was a relief for consumers whose apps rely on Flutter for development.

**Breaking changes**

Here are some of the changes that might prevent other Flutter features from functioning properly or at all.

* Deprecated API that was no longer supported after version 2.10 has been eliminated.
* Except for on macOS and iOS, ZoomPageTransitionsBuilder replaces FadeUpwardsPageTransitionsBuilder as the default page transition builder.
* Change the deleteButtonTooltipMessage of Chips on widgets like the Chip, InputChip, and RawChip from useDeleteButtonTooltip.

**Sunsetting support for 32-bit iOS/iOS 9/iOS 10**

The final stable version of Flutter that supported 32-bit iOS devices running iOS 9 and 10 was Flutter 3. After this update, the iPhone 4S, iPhone 5, iPhone 5C, and the second, third, and fourth-generation iPad devices will all be affected.

**Upgrade of the Flutter/Firebase integration**

As a major component of Firebase's offering, Flutter integration is now fully supported. You can be sure that this integration will continue to receive thoughtful attention as the source code and documentation are transferred to the main Firebase repository and website.

**Conclusion**

With the addition of these key features, Flutter 3 has become a lot better, quicker, and more powerful framework. Platform support for Linux and Mac, the launch of additional engines like Impeller, and ongoing optimizations like UI threads optimizations are some of these aspects.

Flutter 3 also enhanced developer experiences with tools including updated lints, support for Material Design 3, and broader platform compatibility.

While many additions attempt to speed up the engine and enhance development experiences, others have enhanced the performance of the app, which has dramatically enhanced user experiences.

Flutter will become an essential tool for cross-platform programming as more capabilities are added to it. So, if you're planning to develop an app soon, think about using Flutter not just to save money and time, but also because it will enhance the performance of your app.