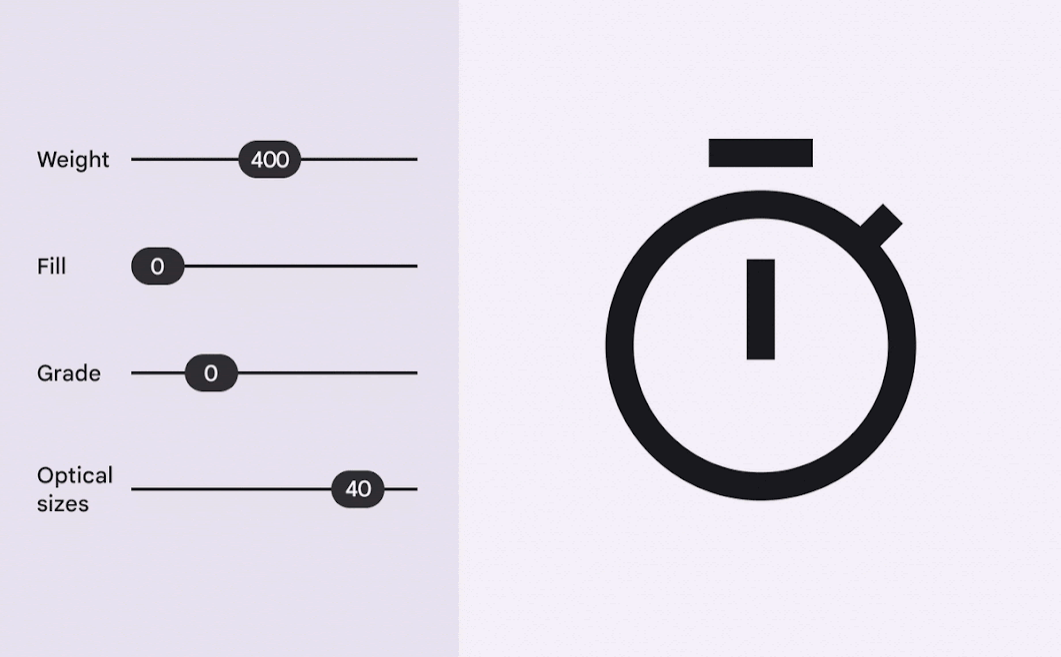
Flutter Material Symbols

SUMMARY

Proposal for adding support for Material Symbols.

Author: Pierre-Louis Guidez (guidezpl)
Go Link: flutter.dev/go/material-symbols
Created: 05/2022 / **Last updated:** 01/2025



WHAT PROBLEM IS THIS SOLVING?

Make **Material Symbols** (replacement for Material Icons) available to all Flutter developers while addressing pain points with the current Material Icons offering around availability, maintenance, and IDE integration.

Tracking issue: [Support Material Symbols · Issue #102560 · flutter/flutter · GitHub]

BACKGROUND

Material Symbols

Material Symbols (developer guide, introductory blog post) is Material's newest set of icons, consolidating over 2,500 glyphs in 3 font files with a wide range of design variants. Symbols are available in three styles (outlined, rounded, sharp) and four adjustable variable font axes (fill, weight, grade, and optical size). Over time, it is anticipated Material Symbols will deprecate Material Icons.

Glossary

- **Variable font** - a font file that is able to store a continuous range of design variants
- **Variable font axis** - an input to a variable font, controlling some stylistic aspect. Axes are defined by 4 letters and are relatively standardized. One commonly found axis is weight (*WGHT*):

- `FontVariation` - an axis tag and value pair that can be used to customize variable fonts, e.g. `FontVariation('wght', 800.0)`
- `IconData` - a description of an icon fulfilled by a font glyph, including a codepoint and a font family, e.g. `IconData(0xe701, fontFamily: 'MaterialIcons')`
- **Material Icons** - a set of icons backed by a single font containing 4 styles (filled, outlined, sharp, round), currently totalling 8695
- **Material Symbols** - a new set of icons, backed by 3 variable fonts (outlined, sharp, round), each offering much greater customization options through the use of 4 variable font axes: fill, weight, grade, optical size
- **Icon font tree shaking** - removing unused data from a font file, or chucking it altogether

Material `Icons`

Current Flutter's support for Material Icons is found in:

- the Flutter tool itself
  - `uses-material-design: true` in an app's `pubspec.yaml` signals to the tool that the app requires Material fonts and icons, i.e. **Roboto** and the **Material Icons** fonts
- the `material` library

- `Icons` is an interface of `static const`s, listing all available icons https://api.flutter.dev/flutter/material/Icons-class.html

Pain point: Delayed availability

When Google Fonts introduces new icons, the time to availability in a stable release is dictated by the Flutter release cycle. In other words, the time to notice an icon is missing + time to create and land a PR + time to stable (3+ months). This can take **upwards of 6 months**.

Pain point: Maintenance

While the process to add/update icons has been vastly improved through the use of Google-internal scripts (go/effortless-flutter-font-updates), it must be initiated by a Google engineer, after icons are noticed to be missing.

Pain point: IDE integration

Both IDE plugins rely on special tooling in https://github.com/flutter/tools_metadata to generate PNGs for each icon. Read more background in the following comments: VS Code, IntelliJ / Android Studio.

`CupertinoIcons`

`CupertinoIcons` is an interface similar to `Icons` for providing Apple icons to developers. However, while it is also defined in the framework, it differs from Material Icons in that the **icon font** is provided in a **pub package**, `cupertino_icons`. See Export fonts from a package | Flutter for details.

Icon

`Icon` is the Flutter API (in `widgets`) for displaying an icon. For example, `Icon(icon: Icons.airplane)` or `Icon(icon: CupertinoIcons.airplane)`. Its members for visual customizations are `size`, `color`, and `shadows`.

Tree-shaking

In release builds for some platforms, Flutter tree-shakes icon fonts such that the font only includes used icons. However, if no icons from a font can be detected as used, Flutter **does not tree shake the file** at all**,** as it could actually be a text font. Usually, this isn't a problem because it is highly likely that an app which has set `uses-material-design: true` uses at least 1 icon.

~~Flutter **does not support tree shaking for the web**, thereby including the entire font file. See https://github.com/flutter/flutter/issues/57181. This has important bundle size implications currently, even more so with Material Symbols, whose font files are ~5MB each (15MB total), untreeshaken.~~

OVERVIEW

This section covers framework support, API, location, IDE integration, tree-shaking, and maintenance.

Non-goals

- Deprecation of Material Icons
- Replacing the Roboto font with Roboto Flex (variable font)

These will be covered in future design documents.

DETAILED DESIGN/DISCUSSION

Framework support

- ☑ ~~Add 4 members to `Icon`: `fill`, `weight`, `grade`, and `opticalSize`. Under the hood, pass these to the engine using the `FontVariation` API. `fill` gets a default value of true. No guarantees are provided:~~
  - If used with a font which does not support these axes => no effect
  - No bounds checking for values => no effect outside supported range
    - Fill: 0-1, weight: 100-700, grade: -25-200, optical size: 20-48

- ☑ ~~Add 4 corresponding members to `IconThemeData` to customize all icons in a given widget subtree.~~

API

Function:
`Symbols.get(String name, style)`
E.g. Symbols.get('airplane', SymbolStyle.rounded))

Statically:
`Symbols.x, Symbols.x_rounded, Symbols.x_sharp`
E.g. `Symbols.airplane`, `Symbols.airplane_rounded`, `Symbols.airplane_sharp`

Location

As previously alluded to, support for icons comes in two parts:

- The Dart API defining `IconData`s
- The icon font file

	Option	**Interface location**	**Icon font location**	**Decision**
1	*New*	Pub package	Pub package (same)	**Recommended**: Simplifies adding icons

	Option	Interface location	Icon font location	Decision
2	`CupertinoIcons`-style	`material`	Pub package	
3	`Icons`-style	`material`	`material`	**Rejected**: doesn't solve the problem of delayed availability of icons

Maintenance

The GitHub hosted package will rely on a GitHub action which will:

1. Regularly check https://github.com/google/material-design-icons/tree/master/variablefont for updated icons
2. Run https://github.com/flutter/flutter/blob/master/dev/tools/update_icons.dart to update the Dart interface
3. Create a PR for review which also bumps up the version
4. (Optional) Publish new versions to pub

IDE integration

TBD

Tree-shaking

~~https://github.com/flutter/flutter/issues/57181 is blocking for web adoption of Material Symbols and will need to be prioritized.~~

OPEN QUESTIONS

- Is there an advantage to do what `CupertinoIcons` did and provide an interface in the framework rather than in the pub package?

TESTING PLAN

Testing will come in the following forms:

- Improved tree-shaking tests, since we now care about 3 fonts instead of 1
- New golden tests for a sample of icons in various valid and invalid configurations

DOCUMENTATION PLAN

A migration guide will be provided. It is yet to be determined whether transitioning from Material Icons to Material Symbols is a visual breaking change, but there are likely to be small pixel differences.

Code samples will showcase best practices for using the different variable font axes, either individually and in combinations.

MIGRATION PLAN

Developers should be able to migrate from Material Icons to Material Symbols with relative ease.

Material Icons (before)	Material Symbols (after)

`Icon(`
`  // fill of 0 implied`
`  Icons.ten_mp,`
`)`
	`Icon(`
	`  // fill style by default`
	`  Symbols.ten_mp,`
	`)`

`Icon(`
`  Icons.ten_mp_outlined,`
`)`
	`Icon(`
	`  Symbols.ten_mp,`
	`  fill: 1.0,`
	`)`

`Icon(`
`  Icons.ten_mp_rounded,`
`)`
	`Icon(`
	`  Symbols.ten_mp_rounded,`
	`)`

`Icon(`
`  Icons.ten_mp_sharp`
`)`
	`Icon(`
	`  Symbols.ten_mp_sharp,`
	`)`

When possible, Dart fixes will be provided.