Get started
Samples & tutorials
Development
- User interface
  - Introduction to widgets
  - Building layouts
    - Layouts in Flutter
    - Tutorial
    - Creating responsive apps
    - Box constraints
  - Splash screens
  - Adding interactivity
  - Assets and images
  - Navigation & routing
  - Animations
  - Advanced UI
  - Widget catalog
- Data & backend
- Accessibility & internationalization
- Platform integration
- Packages & plugins
- Add Flutter to existing app
- Tools & techniques
- Migration notes

Testing & debugging
Performance & optimization
Deployment
Resources
Reference
- Widget index
- API reference
- Package site

Creating responsive apps

A responsive app lays out its UI according to the size and shape of the screen or window. This is especially necessary when the s
app can run on a variety of devices, from a watch, phone, tablet, to a laptop or desktop computer. When the user resizes the wind
on a laptop or desktop, or changes the orientation of the phone or tablet, the app should respond by rearranging the UI accordingl

Flutter allows you to create apps that self-adapt to the device's screen size and orientation.

There are two basic approaches to creating Flutter apps with responsive design:

Use the `LayoutBuilder` class
From its `builder` property, you get a `BoxConstraints` object. Examine the constraint's properties to decide what to display. For example, if your `maxWidth` is greater than your width breakpoint, return a `Scaffold` object with a row that has a list on the left. If it narrower, return a `Scaffold` object with a drawer containing that list. You can also adjust your display based on the device's heigh the aspect ratio, or some other property. When the constraints change (for example, the user rotates the phone, or puts your app a tile UI in Nougat), the build function runs.

Use the `MediaQuery.of()` method in your build functions
This gives you the size, orientation, etc, of your current app. This is more useful if you want to make decisions based on the comp context rather than on just the size of your particular widget. Again, if you use this, then your build function automatically runs if t user somehow changes the app's size.

Other useful widgets and classes for creating a responsive UI:

- `AspectRatio`
- `CustomSingleChildLayout`
- `CustomMultiChildLayout`
- `FittedBox`
- `FractionallySizedBox`
- `LayoutBuilder`
- `MediaQuery`
- `MediaQueryData`
- `OrientationBuilder`

For more information, here are a few resources, including contributions from the Flutter community:

- Developing for Multiple Screen Sizes and Orientations in Flutter by Deven Joshi
- Build Responsive UIs in Flutter by Raouf Rahiche
- Making Cross-platform Flutter Landing Page Responsive by Priyanka Tyagi
- How to make flutter app responsive according to different screen size?, a question on StackOverflow