

**Intermediate Flutter**

**Course Number:** FLTR-102  
**Duration:** 3 days

**Overview**

This Intermediate Flutter training teaches attendees to build more sophisticated, robust Flutter applications. Attendees learn how to implement responsive design, style widgets, manage state, make RESTful API calls with HTTP/HTTPS, and more.

**Prerequisites**

Experience in another object-oriented programming language like Java, C#, or C++. Students should have also taken Accelebrate's [Introduction to Flutter](file:////training/flutter-introduction) course or have equivalent experience.

**Materials**

All Flutter training attendees receive comprehensive courseware.

**Software Needed on Each Student PC**

* Google Chrome
* Other modern browsers as desired
* IDE/development environment of your choice
* Other free software and lab files that Accelebrate would specify

**Objectives**

* Precisely control the layout of apps in a responsive way
* Handle form data entry from users
* Make multiscreen apps with navigation, menus, and tabs
* Use Flutter to read and write data from an online RESTful API
* Find and include 3rd party libraries

**Outline**

* Introduction
* Laying Out Your Widgets
  + Laying out the whole scene
  + MaterialApp widget
  + The Scaffold widget
  + The AppBar widget
  + SafeArea widget
  + SnackBar widget
  + How Flutter decides on a widget’s size
  + The dreaded “unbounded height” error
  + Flutter’s layout algorithm
  + Putting widgets next to or below others
  + Your widgets will never fit!
  + What if there’s extra space left over?
  + mainAxisAlignment
  + crossAxisAlignment
  + Expanded widget
  + What if there’s not enough space?
  + The ListView widget
  + Container widget and the box model
  + Alignment and positioning within a Container
  + So how do you determine the size of a Container?
  + Special layout widgets
  + Stack widget
  + GridView widget
  + The Table widget
* Navigation and Routing
  + Stack navigation
  + Navigating forward and back
  + Get result after a scene is closed
  + Drawer navigation
  + The Drawer widget
  + Filling the drawer
  + Tab Navigation
  + TabController
  + TabBar and Tabs
  + The Dialog widget
  + showDialog( ) and AlertDialog
  + Responses with a Dialog
  + Navigation methods can be combined
* Styling Your Widgets
  + Thinking in Flutter Styles
  + A word about colors
  + Styling Text
  + TextStyle
  + Custom fonts
  + Container decorations
  + Border
  + BorderRadius
  + BoxShape
  + Stacking widgets
  + Positioned widget
  + Card widget
  + Themes
  + Applying theme properties
* Managing State
  + What is state?
  + What goes in a StatefulWidget?
  + The most important rule about state!
  + Passing statedown
  + Lifting state backup
  + An example of state management
  + When should we use state?
  + Advanced state management
  + InheritedWidget
  + BLoC
  + ScopedModel
  + Hooks
  + Provider
  + Redux
* Your Flutter App Can Work with Files
  + Including libraries in your Flutter app
  + Finding a library
  + Adding it to pubspec.yaml
  + Importing the library
  + Using the library
  + Futures, async, and await
  + Why would it wait?
  + await
  + async
  + Including a file with your app
  + Writing a file
  + And reading it!
  + Using JSON
  + Writing your app’s memory to JSON
  + Reading JSON into memory
  + Shared preferences
  + To write preferences
  + To read preferences
* Making RESTful API Calls with HTTP
  + The flavors of API requests
  + Making an HTTP GET or DELETE request
  + Making an HTTP PUT, POST, or PATCH request
  + HTTP responses to widgets
  + Brute force – The easy way
  + FutureBuilder – The clean way
  + Strongly typed classes
  + Create a business class
  + Write a fromJSON( ) method
  + Use fromJSON( ) to hydrate the object
  + One big example
  + A GET request in Flutter
  + A DELETE request in Flutter
  + A POST and PUT request in Flutter
* Using Firebase with Flutter (time permitting)
  + Introducing Firebase
  + Cloud Firestore
  + Cloud Functions
  + Authentication
  + Setting up Firebase itself
    - Creating a Firebase project
    - Creating the database
    - Creating an iOS app
    - Creating an Android app
    - Adding FlutterFire plugins
  + Using Firestore
  + To get a collection
  + To query
  + To upsert
  + To delete
  + Where to go from here
* Conclusion