Flutter

Comprehensive Flutter Development Course

Unlock the power of Flutter and Dart. Build beautiful, natively compiled applications for mobile, web, and desktop from a single codebase.

by Rittik Soni

YouTube - @king_rittik

Community - https://discord.gg/Tmn6BKwSnr

Course Modules Overview

Explore the comprehensive curriculum designed to take you from Flutter fundamentals to advanced application development.

Module 1
Introduction to Flutter & Dart

Module 2
Widgets & Layouts

Module 3
Navigation & Routing

Module 4
State Management

Module 5
Assets, Forms & User Input

Module 6
Async & Networking

Module 7
Data Persistence

Module 8
Advanced Concepts

Module 9
Building & Deploying

Module 10
Project & Next Steps

Module 1: Intro to Flutter & Dart

What is Flutter?

UI toolkit for building natively compiled applications. It offers a single codebase for multiple platforms. Created by Google and open-source, Flutter uses the Dart programming language.

Why Choose Flutter?

- **Hot Reload:** Instant code changes for rapid development.
- **Expressive UI:** Create beautiful, custom interfaces.
- **Native Performance:** Compiled to native code for optimal speed.
- **Cross-Platform:** Reach mobile, web, and desktop from one codebase.

Dart Language Basics

Learn the fundamentals of Dart syntax.

```
// Variables and data types
dynamic name = 'Flutter';
String course = 'Development';
int modules = 10;
bool isAwesome = true;

// Functions
void main() {
 print('Hello Flutter!');
}
```

Module 2: Widgets & Layouts

Understanding Widgets

Everything in Flutter is a widget - the building blocks of your UI

StatelessWidget
Immutable, no internal state

StatefulWidget
Mutable, maintains state

InheritedWidget
Propagates data down

Layout Widgets

Row
Horizontal arrangement

Column
Vertical arrangement

Stack
Overlapping widgets

Expanded
Fill available space

ListView & GridView
Scrollable collections of widgets

Basic Widgets

Text

Image

Icon

Container

Button

Card

Widget Example

```
class MyWidget   extends StatelessWidget {
  const MyWidget({Key? key}) :    super(key: key);

  @override
Widget build(BuildContext context) {
    return Container(
padding: EdgeInsets.all(16.0),
decoration: BoxDecoration(
color: Colors.blue,
borderRadius: BorderRadius.circular(8.0),
),
child: Column(
children: [
Text(           'Hello Flutter!'  ),
```

Module 3: Navigation & Routing

Master how to move between screens and manage application flow in Flutter.

1

Basic Navigation

Push to a new screen. Return to the previous screen using pop.

```
Navigator.push(context,
MaterialPageRoute(builder: (context) => SecondScreen(),
  ),
);

// Returns to previous screen
Navigator.pop(context);
```

2

Passing Data Between Screens

Forward data via constructors or return data via pop results.

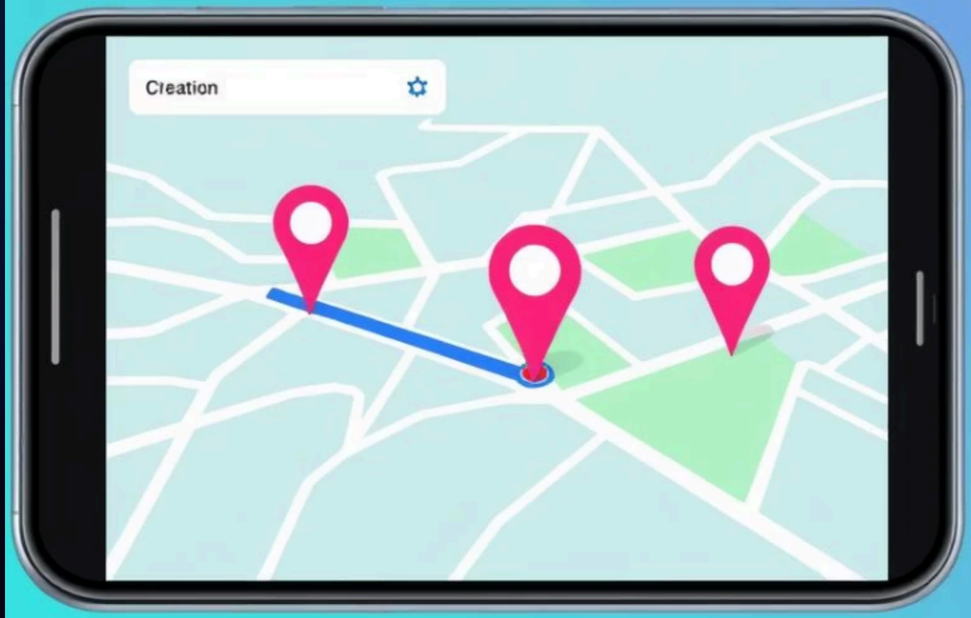
```
Navigator.push(context,
MaterialPageRoute(builder: (context) => DetailScreen(data: 'Hello from Home!',
   ),
  ),
);
```

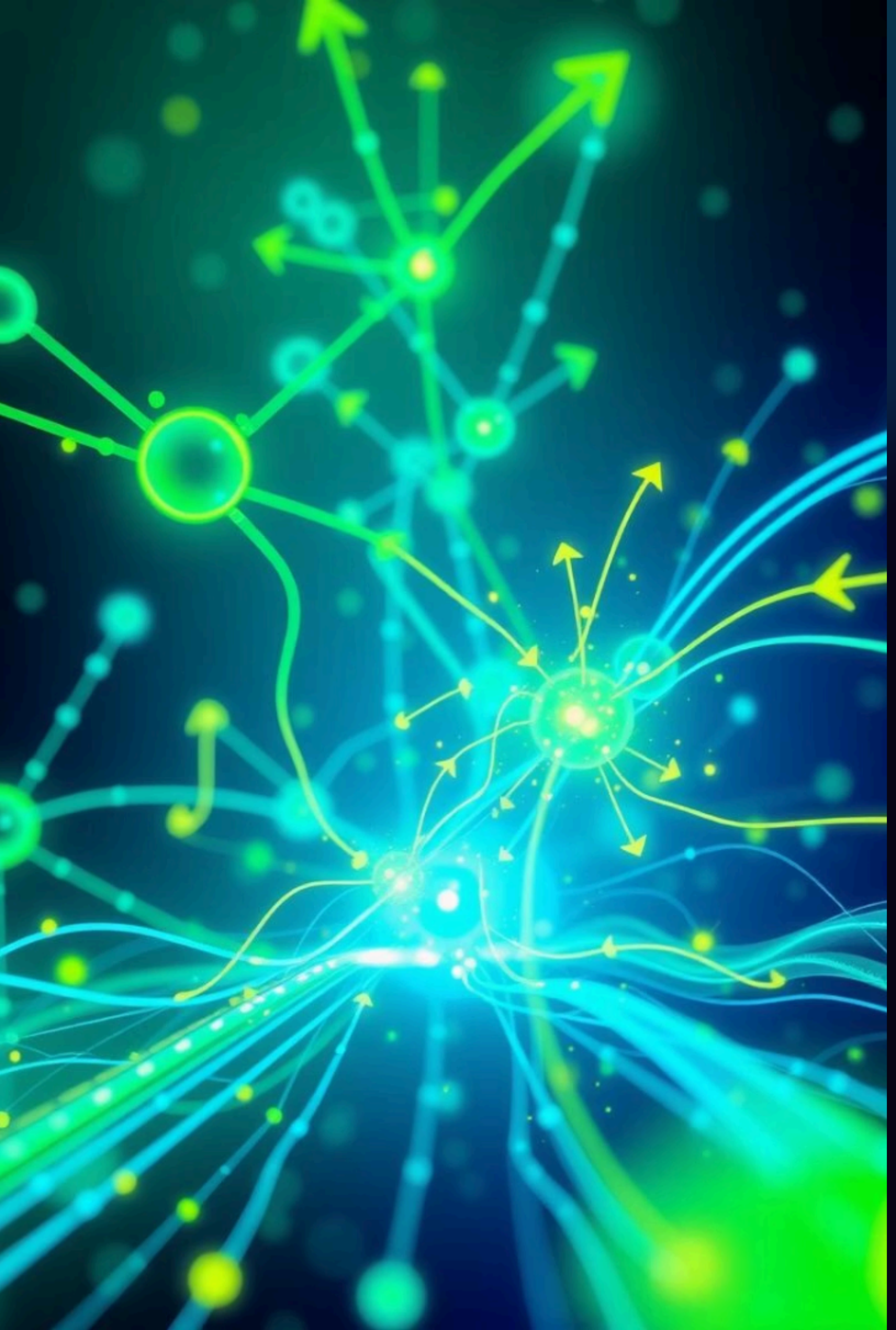
3

Named Routes

Define routes in MaterialApp for cleaner navigation.

```
MaterialApp(
  routes: {
    '/': (context) => HomeScreen(),
    '/details': (context) => DetailScreen(),
  },
);
```





Module 4: State Management

Learn to manage data that changes during your app's lifetime.

What is State?

State is data that can change during the lifetime of the app.

Local State

Belongs to a single widget, like form input values.

Global State

Shared across multiple widgets, like user authentication.

setState & StatefulWidget

Manage mutable state within a StatefulWidget using setState.

setState & StatefulWidget

```
classCounterWidget  extendsStatefulWidget {
  @override
_CounterWidgetState createState() =>
_CounterWidgetState();
}

class_CounterWidgetState  extends
State<CounterWidget> {
  int_counter = 0;

  void_incrementCounter() {
setState(() {
_counter++;
});
}

  @override
Widget build(BuildContext context) {
    returnColumn(
children: [
Text(         'Count: $_counter'),
ElevatedButton(
onPressed: _incrementCounter,
child: Text(           'Increment'),
),
],
);
}
}
```

Module 4: State Management Example

YouTube - @king_rittik

Provider Example

```
// 1. Define a model
classCounterModel  extendsChangeNotifier {
  int_count = 0;
  intget count => _count;

  voidincrement() {
_count++;
notifyListeners();
}
}
```

```
// 2. Provide the model
ChangeNotifierProvider(
create: (context) => CounterModel(),
child: MyApp(),
);

// 3. Consume the model
finalcounter = Provider.of<CounterModel>(context);
// or
Consumer<CounterModel>(
builder: (context, counter, child) =>
Text('${counter.count}'),
);
```

Module 5: Assets, Forms & Input

Integrate external resources and handle user interactions effectively.

Working with Assets

Declare and use images and custom fonts in your Flutter app.

```
# pubspec.yaml
flutter:
assets:
- assets/images/
- assets/icons/logo.png  fonts:
- family: Raleway
fonts:
- asset: fonts/Raleway-Regular.ttf
```

Images

Display images from your declared assets.

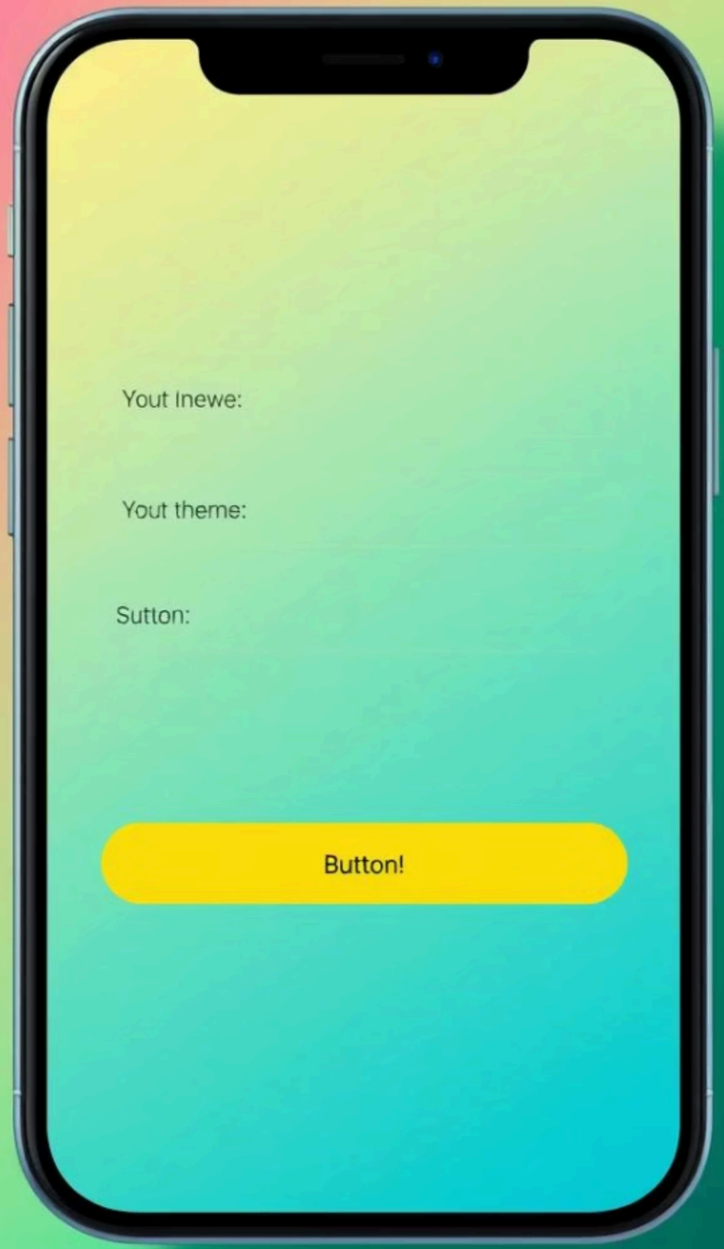
```
Image.asset(
'assets/images/pic.png'
);
```

User Input & Forms

Create interactive forms with text fields and buttons.

Use **TextFormField** for input. Handle submissions with **ElevatedButton**.

YouTube - @king_rittik



Module 6: Async & Networking

Handle asynchronous operations and network requests for dynamic data.

1 Asynchronous Programming

Learn to work with operations that don't block the UI.

2 Futures

A Future represents a value available later.

```
Future fetchData() async {
 await Future.delayed(Duration(seconds: 2));
 return 'Data loaded';
}
```

3 Streams

A Stream is a sequence of asynchronous events.

```
Stream countStream() async* {

for (int i = 1; i <= 5; i++) {
   await Future.delayed(Duration(seconds: 1));
    yield i;
  }

}
```

Module 7: Data Persistence

Store and retrieve data locally within your Flutter applications.

Shared Preferences

Simple key-value storage for small data.

```
final prefs = await SharedPreferences.getInstance();

await prefs.setString('username', 'king_rittik');
```

SQLite Database

Relational database for structured data. Use for complex local storage.

Integrate with **sqflite** package. Perform CRUD operations.

Cloud Databases

Explore options like Firebase Firestore. Real-time data synchronization.

Build scalable, connected applications.

YouTube - @king_rittik

Module 8: Advanced Concepts

Elevate your Flutter skills with advanced styling and optimization techniques.

Themes & Styling

Ensure consistent UI with **ThemeData**.

```
MaterialApp( theme: ThemeData( primarySwatch: Colors.blue, brightness: Brightness.light, ),);
```

Performance Optimization

Techniques for smooth app performance.

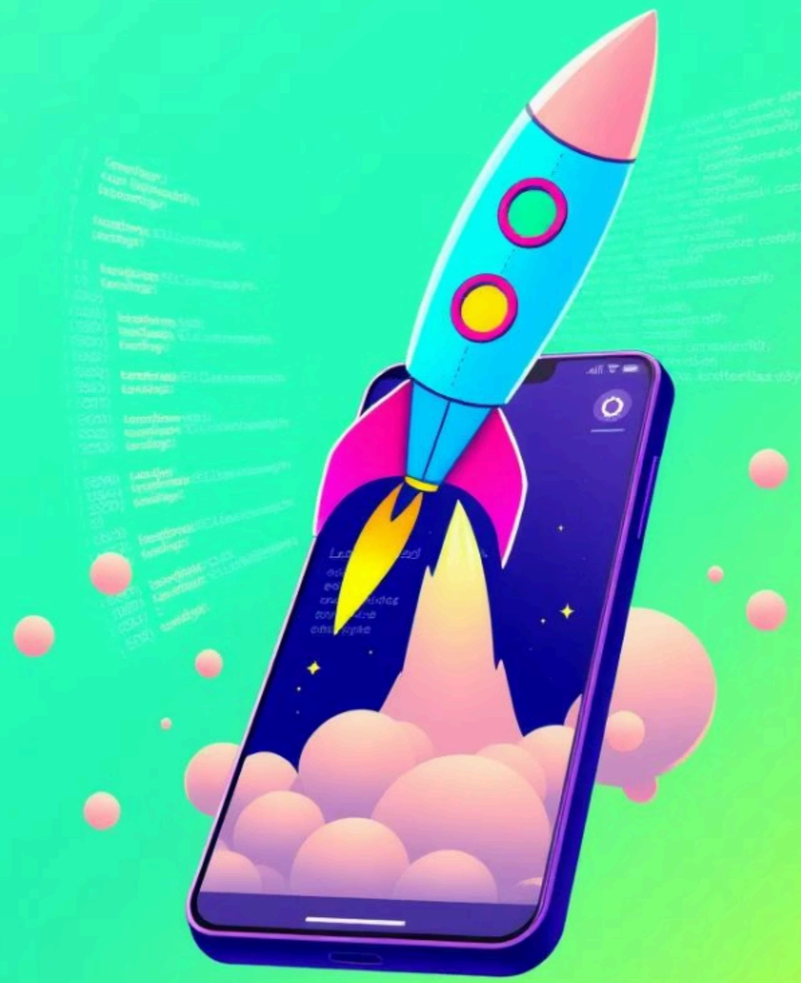
Use **const** widgets.

Security Best Practices

Protect user data and app integrity.

Secure API keys.

YouTube - @king_rittik



Module 9: Building & Deploying

Prepare for Release

Optimize performance. Update app icons and versions. Securely sign your application for distribution.

Build Release Packages

Generate platform-specific binaries. Create Android App Bundles (AAB) or iOS archives for submission.

Deploy to Stores

Upload to Google Play Store or Apple App Store. Complete all store listing details for review.

Module 8: Advanced Concepts

Themes & Styling

Ensure consistent UI with ThemeData. Apply cohesive visual designs across your application.

Comprehensive Testing

Ensure app quality and stability. Write robust unit, widget, and integration tests.

Platform Channels

Communicate with native iOS or Android code. Access device-specific APIs and functionalities.

Animations

Bring your UI to life. Implement smooth transitions using implicit or explicit animations.

Course Conclusion & Beyond

Official Flutter Docs

Explore comprehensive guides and API references. Stay updated with new features.

Join Community

Connect with fellow developers on Discord. Share knowledge and collaborate on projects. https://discord.gg/Tmn6BKwSnr

YouTube Tutorials

Continue learning with @king_rittik. Subscribe for more in-depth Flutter content.

Keep Building!

Apply your new skills to create amazing apps. Your Flutter journey has just begun.

Thank You!

Search me on Google - Rittik Soni

By Rittik Soni
YouTube - @king_rittik
https://discord.gg/Tmn6BKwSnr
contact.kingrittik@gmail.com

Your Dedication

We appreciate your hard work and commitment throughout the Flutter course.

Keep Growing

Continue exploring new concepts and mastering your development skills.

Build Amazing Apps

Now, apply your knowledge to create innovative and impactful mobile experiences.