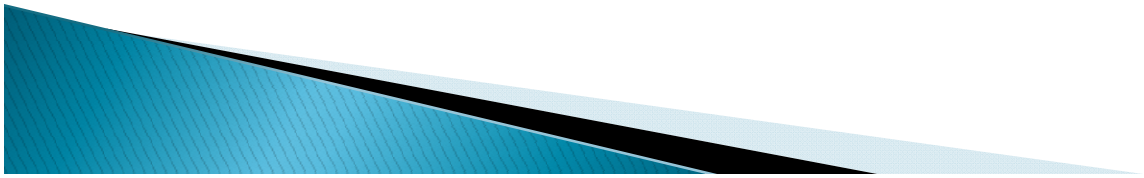
Flutter

พัฒนา Mobile App ด้วย Flutter (Android และ iOS)

โค้ชเอก

Codingthailand.com

แนะนำ Flutter

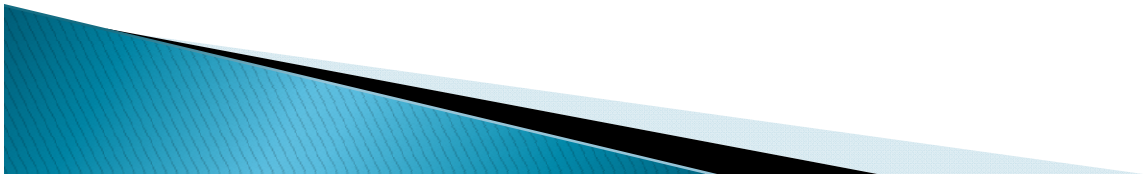


What is flutter?

- Flutter is Google's **UI toolkit for** building beautiful**, natively compiled applications for** mobile**,** web**, and** desktop **from a single codebase.**

- Fast development
- Expressive, beautiful UIs
- Native Performance

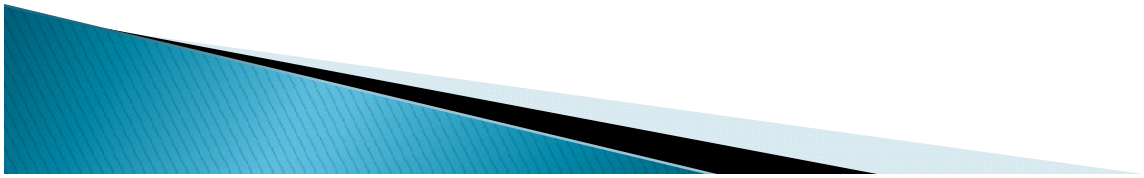
- https://flutter.dev/



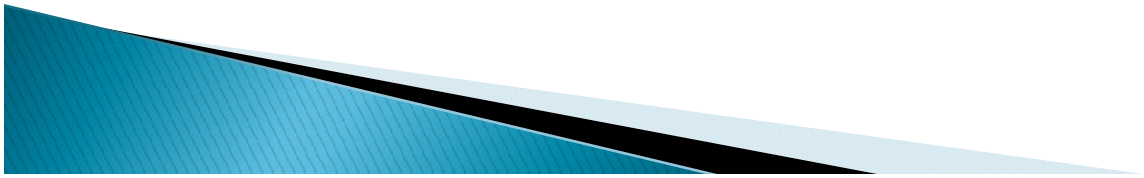
Who's using Flutter?

ถ้าเคยเขียน platform อื่นมาก่อน ดูได้ที่นี่

- https://flutter.dev/docs/get-started/flutter-for

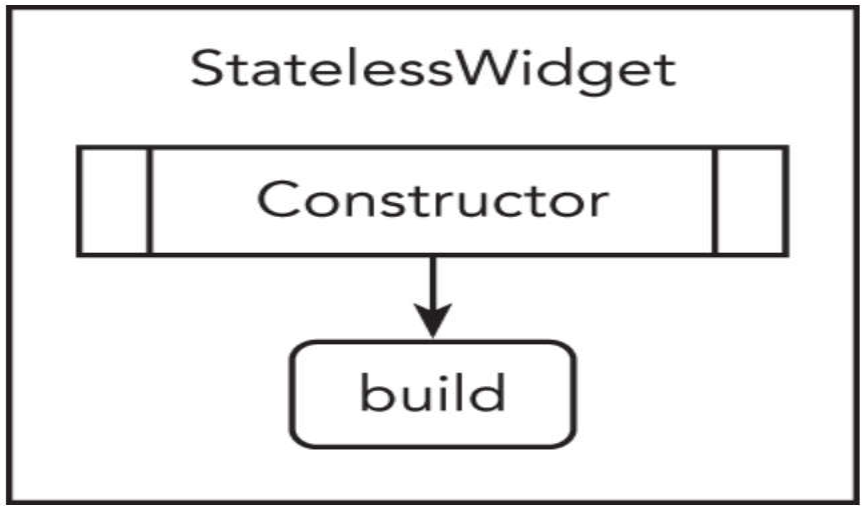


พื้นฐาน Widgets และการใช้งาน



StatelessWidget

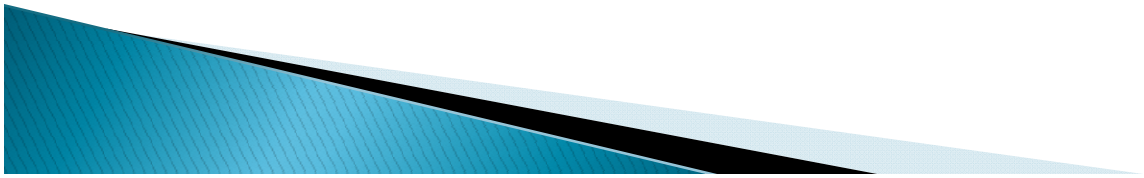
- A StatelessWidget is built based on its own configuration and **does not change dynamically.**



StatelessWidget

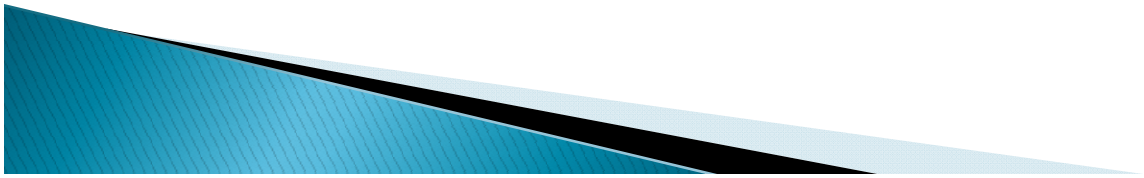
```
class GreenFrog extends StatelessWidget {
  const GreenFrog({ Key key }) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Container(color: const Color(0xFF2DBD3A));
  }
}
```

ทดลองเขียน StatelessWidget

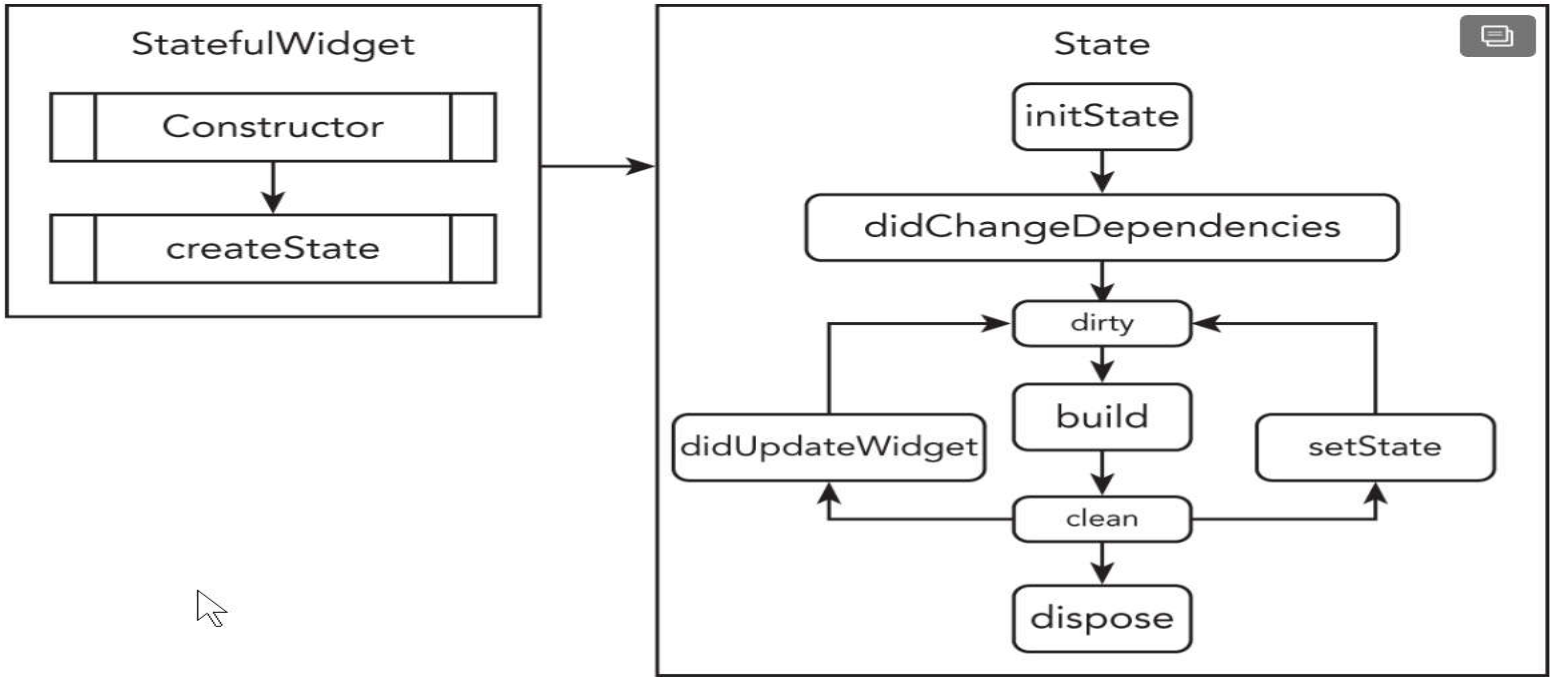


StatefulWidget

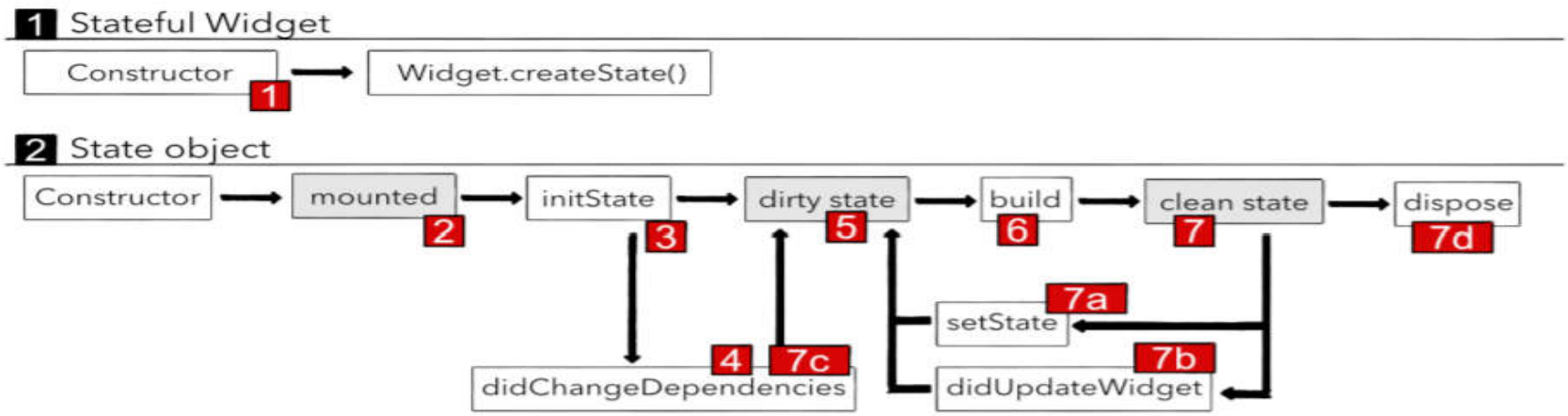
- A StatefulWidget is built based on its own configuration **but can change dynamically.**
- The stateful widget is declared with **two classes,** the **StatefulWidget class and the State class**.
- The StatefulWidget class is rebuilt when the widget's configuration changes, but the State class can persist (remain), enhancing performance.



The StatefulWidget Lifecycle



The StatefulWidget Lifecycle



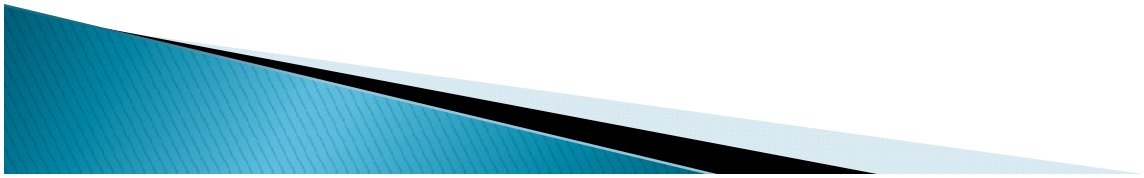
The StatefulWidget Lifecycle

METHOD	DESCRIPTION	SAMPLE CODE

`initState()`	Called once when this object is inserted into the tree.	`@override`
		`void initState() {`
		`  super.initState();`
		`  print('initState');`
		`}`

`dispose()`	Called when this object is removed from the tree permanently.	`@override`
		`void dispose() {`
		`  print('dispose');`
		`  super.dispose();`
		`}`

`didChangeDependencies()`	Called when this `State` object changes.	`@override`
		`void didChangeDependencies() {`
		
		`  super.didChangeDependencies();`
		`  print('didChangeDependencies');`
		`}`



The StatefulWidget Lifecycle

METHOD	DESCRIPTION	SAMPLE CODE

`didUpdateWidget(Contacts oldWidget)`	Called when the widget configuration changes.	`@override`
		`void didUpdateWidget(Contacts oldWidget) {`
		
		`super.didUpdateWidget(oldWidget);`
		`print('didUpdateWidget: $oldWidget');`
		`}`

`deactivate()`	Called when this object is removed from the tree.	`@override`
		`void deactivate() {`
		
		`print('deactivate');`
		`super.deactivate();`
		`}`

`build(BuildContext context)`	Can be called multiple times to build the UI, and the `BuildContext` handles the location of this widget in the widget tree.	`@override`
		`Widget build(BuildContext context)`
		`{`
		`print('build');`
		`return Container();`
		`}`

`setState()`	Notifies the framework that the state of this object has changed to schedule calling the `build` for this `State` object.	`setState(() {`
		`name = _newValue;`
		`});`

setState() method

- The state object marks the element as dirty (has changed) and the build method rebuilds the UI children widgets.

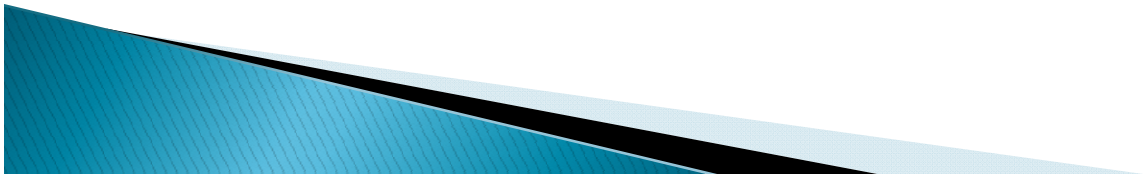
```
class _MyHomePageState extends State<MyHomePage> {
  int _counter = 0;
  String _title = '';

  void _onPressed() {
    setState(() {
      _title = 'สวัสดีนะ';
    });
  }
```

StatefulWidget

```
class YellowBird extends StatefulWidget {
  const YellowBird({ Key key }) : super(key: key);

  @override
  _YellowBirdState createState() => _YellowBirdState();
}

class _YellowBirdState extends State<YellowBird> {
  @override
  Widget build(BuildContext context) {
    return Container(color: const Color(0xFFFFE306));
  }
}
```

ทดลองเขียน StatefulWidget

การใช้งาน MaterialApp และ
การจัดการ Theme



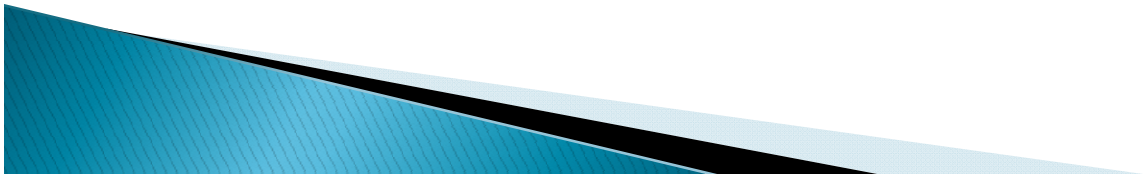
การใช้งาน MaterialApp class

- An application that uses material design.
- https://api.flutter.dev/flutter/material/MaterialApp-class.html

```
MaterialApp(
  home: Scaffold(
    appBar: AppBar(
      title: const Text('Home'),
    ),
  ),
  debugShowCheckedModeBanner: false,
)
```

การจัดการ Theme

- https://api.flutter.dev/flutter/material/ThemeData-class.html



การจัดการ Theme

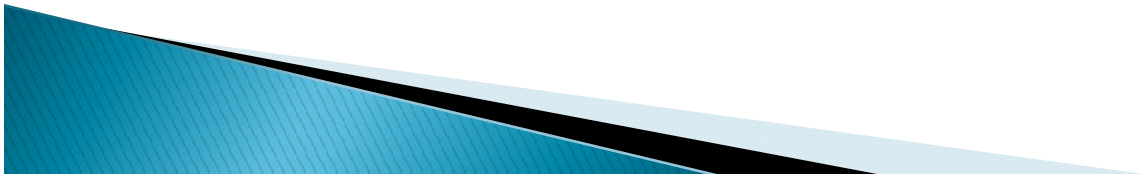
```dart
@override
Widget build(BuildContext context) {
  return MaterialApp(
    title: 'Flutter Demo',
    theme: ThemeData(
      primarySwatch: Colors.lightGreen,
      canvasColor: Colors.lightGreen.shade100,
      platform: TargetPlatform.android
    ), // ThemeData
    home: MyHomePage(title: 'Flutter'),
  ); // MaterialApp
}
```

```dart
return MaterialApp(
 // navigatorKey: mainNavigatorKey,
  title: 'TOA',
  themeMode: ThemeMode.light,
  theme: ThemeData(
    brightness: Brightness.light,
    // primarySwatch: Colors.red,
    primaryColor: Colors.indigo[800],
    accentColor: Colors.redAccent,
    textTheme: TextTheme(
        headline: TextStyle(fontSize: 72.0, fontWeight: FontWeight.bold),
        title: TextStyle(fontSize: 36.0, fontStyle: FontStyle.italic),
        body1: TextStyle(fontSize: 18.0)), // TextTheme
  ), // ThemeData
```

```dart
Widget build(BuildContext context) {
  return MaterialApp(
    title: 'Flutter Demo',
    // theme: ThemeData.light(),
    theme: ThemeData(
      brightness: Brightness.dark,
      primarySwatch: Colors.lightGreen,
      accentColor: Colors.blueAccent
    ), // ThemeData
    home: MyHomePage(title: 'Flutter'),
  ); // MaterialApp
}
```

การจัดการ Theme

- ทดลองแก้ไข และใช้งาน textTheme



การใช้สี ใน Flutter

- https://api.flutter.dev/flutter/material/Colors-class.html
- https://api.flutter.dev/flutter/dart-ui/Color-class.html
- https://material.io/resources/color

```
theme: ThemeData(
  // primarySwatch: Colors.green,
  // primaryColor: Colors.orange[900],
  primaryColor: Color(0xffbf360c),
  accentColor: Colors.orangeAccent,
  textTheme: TextTheme(
      headline: TextStyle(
          fontSize: 62, fontWeight: FontWeight.bold, color: Colors.red)
  ), // TextTheme
), // ThemeData
```

การจัดการ assets และ รูปภาพ

- The Image widget displays an image from a local or URL (web) source.

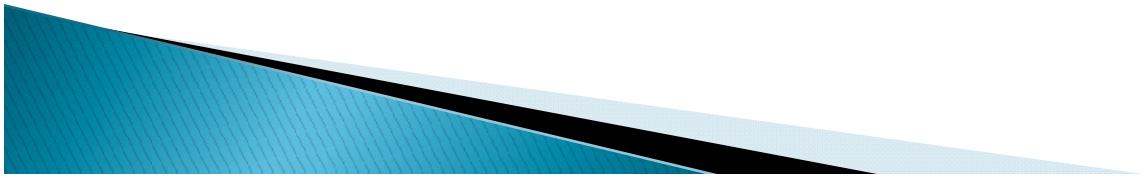
pubspec.yaml

```
41
42  # The following section is specific
43  flutter:
44
45    # The following line ensures that
46    # included with your application,
47    # the material Icons class.
48    uses-material-design: true
49
50    # To add assets to your applicati
51    assets:
52      - assets/images/
53    #   - images/a_dot_ham.jpeg
```

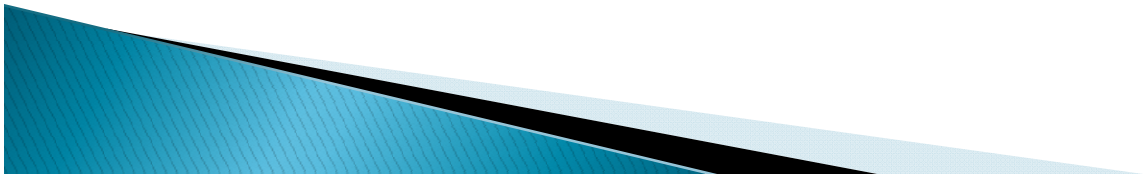
การจัดการ assets และ รูปภาพ

- https://flutter.dev/docs/development/ui/assets-and-images

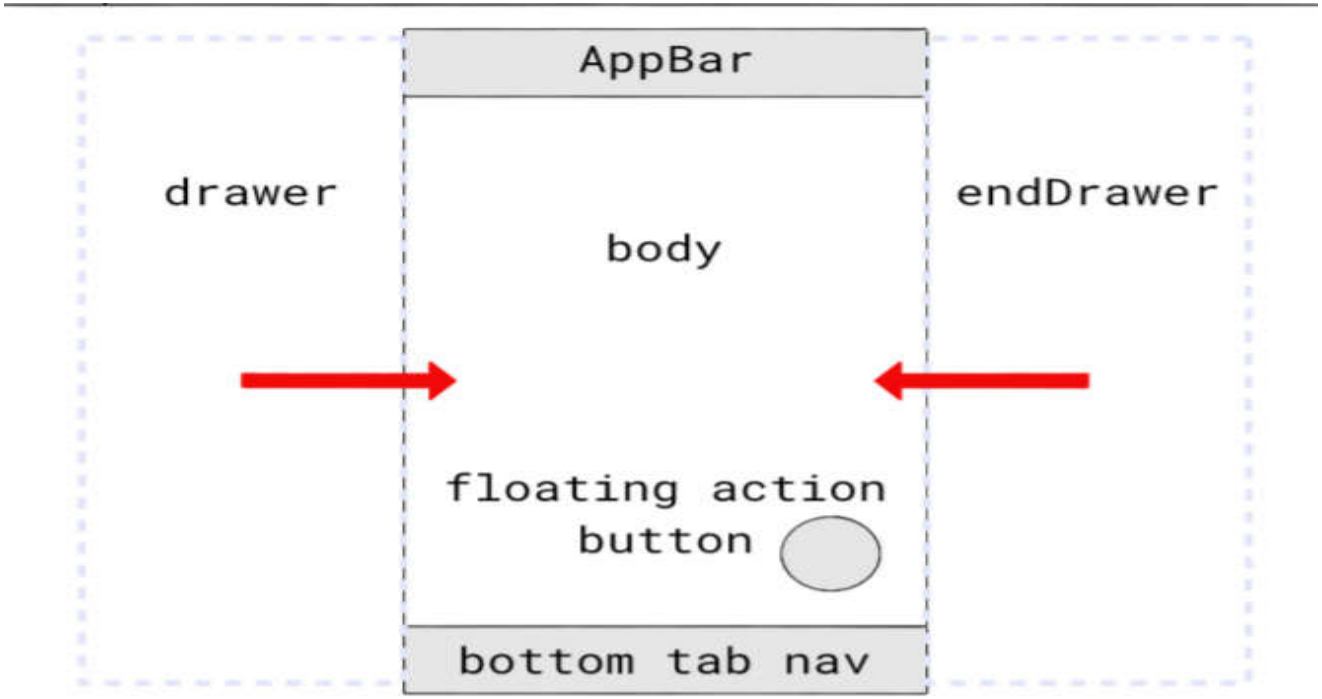
- Image()—Retrieves image from an ImageProvider class
- Image.asset()—Retrieves image from an AssetBundle class using a key
- Image.file()—Retrieves image from a File class
- Image.memory()—Retrieves image from a Uint8List class
- Image.network()—Retrieves image from a URL path



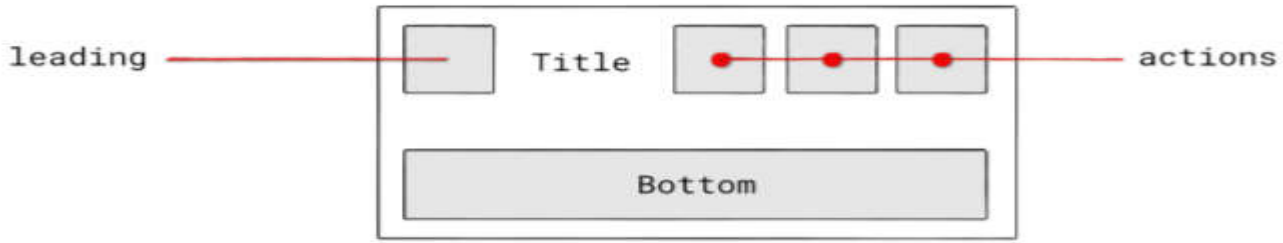
การสร้าง Pages

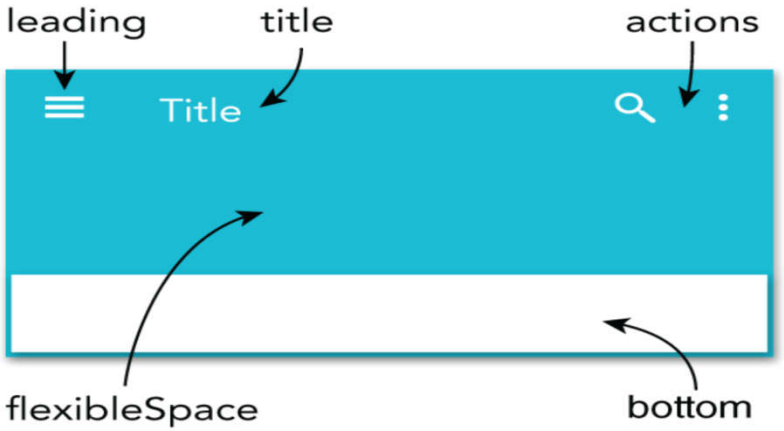


การใช้งาน Scaffold

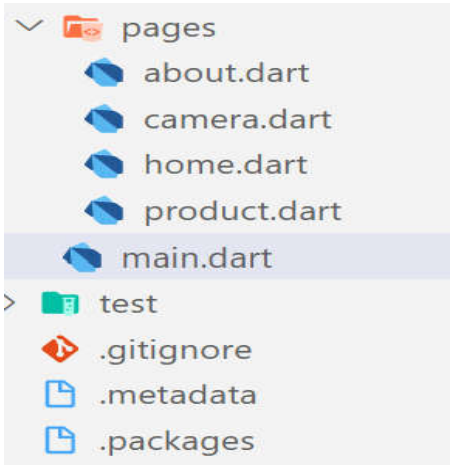


การใช้งาน AppBar





การสร้าง Pages

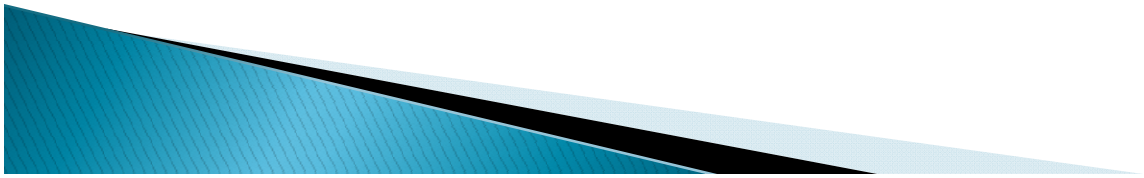



```dart
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Code Sample for Navigator',
      // MaterialApp contains our top-level Navigator
      initialRoute: '/',
      routes: {
        '/': (BuildContext context) => HomePage(),
        '/signup': (BuildContext context) => SignUpPage(),
      },
    );
  }
}
```

ทำแบบฝึกหัด

- ทดลองสร้างเพจ

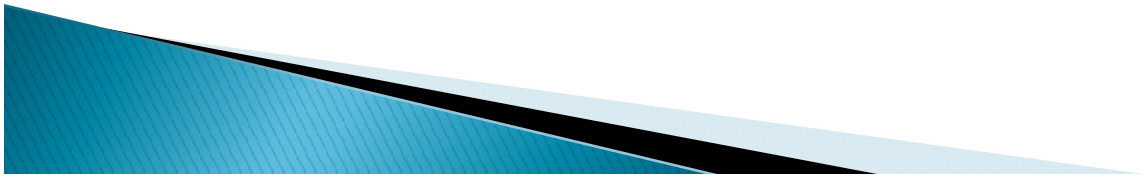
การใช้งาน Navigation และ routing



Navigation (การ push และ pop เพจ)

```
// Within the `FirstRoute` widget
onPressed: () {
  Navigator.push(
    context,
    MaterialPageRoute(builder: (context) => SecondRoute()),
  );
}
```

```
// Within the SecondRoute widget
onPressed: () {
  Navigator.pop(context);
}
```

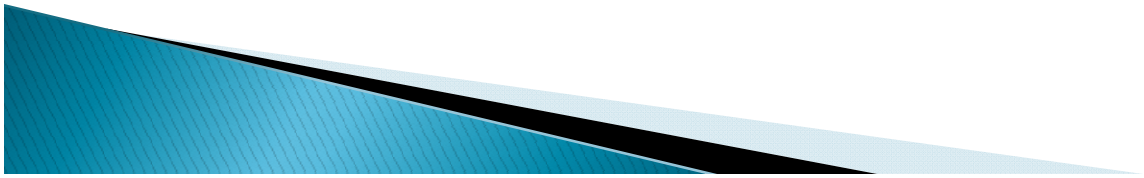


Navigate with named routes

```dart
MaterialApp(
  // Start the app with the "/" named route. In this case, the app starts
  // on the FirstScreen widget.
  initialRoute: '/',
  routes: {
    // When navigating to the "/" route, build the FirstScreen widget.
    '/': (context) => FirstScreen(),
    // When navigating to the "/second" route, build the SecondScreen widget.
    '/second': (context) => SecondScreen(),
  },
);
```

```dart
// Within the `FirstScreen` widget
onPressed: () {
  // Navigate to the second screen using a named route.
  Navigator.pushNamed(context, '/second');
}
```

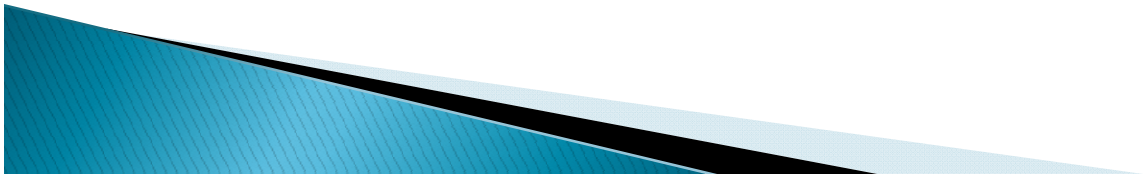
การรับ-ส่งข้อมูลระหว่าง Page



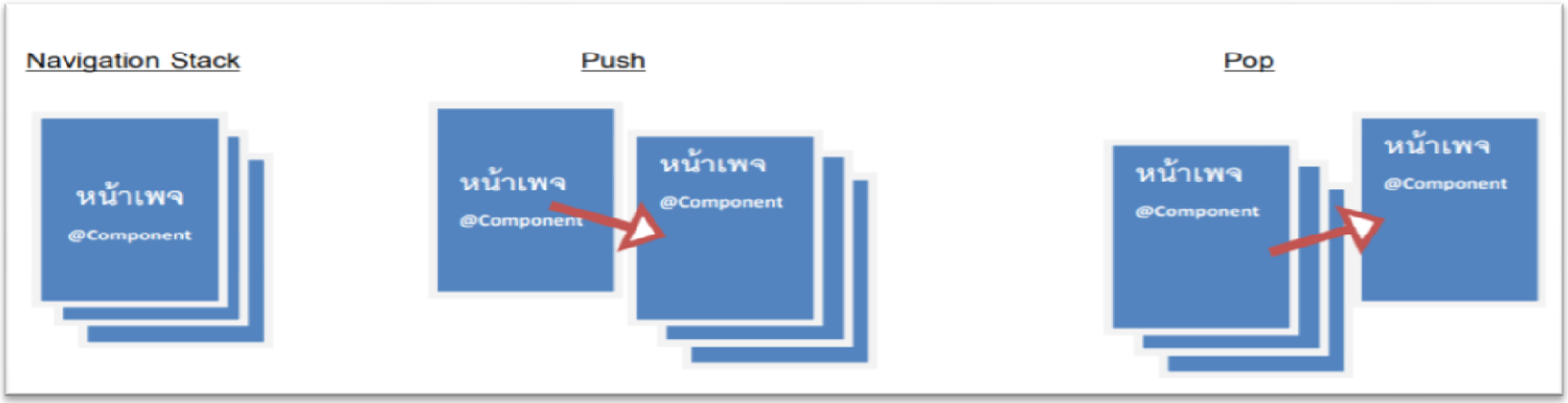
การรับ-ส่งข้อมูลระหว่าง Page

ทำแบบฝึกหัด

```
@override
Widget build(BuildContext context) {

  final Company company = ModalRoute.of(context).settings.arguments;
```



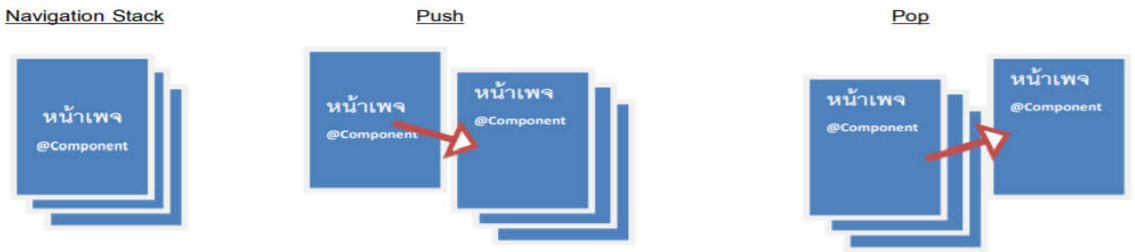
Navigation Stack



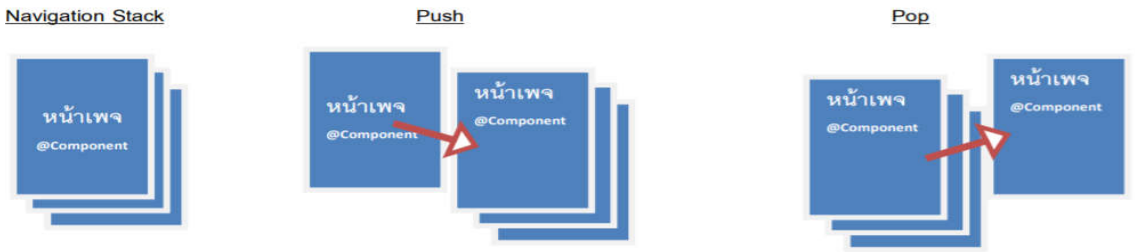
การใช้งาน routing ขั้นสูง (Nesting Navigators)

- https://api.flutter.dev/flutter/widgets/Navigator-class.html

HomeStack

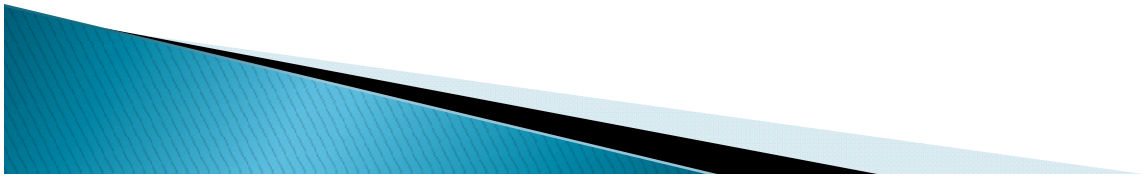


ProductStack



การใช้งาน UI Widgets แบบต่างๆ และการออกแบบ Layout

https://flutter.dev/docs/development/ui/widgets



Text

- **Text :** The Text widget lets you create a run of styled text within your application.

- https://api.flutter.dev/flutter/widgets/Text-class.html

```
Text(
  'Hello, $_name! How are you?',
  textAlign: TextAlign.center,
  overflow: TextOverflow.ellipsis,
  style: TextStyle(fontWeight: FontWeight.bold),
)
```

ตัวอย่างการใช้ Text เพิ่มเติม

```
Text(
 'Flutter World for Mobile',
 style: TextStyle(
  fontSize: 24.0,
  color: Colors.deepPurple,
  decoration: TextDecoration.underline,
  decorationColor: Colors.deepPurpleAccent,
  decorationStyle: TextDecorationStyle.dotted,
  fontStyle: FontStyle.italic,
  fontWeight: FontWeight.bold,
 ),
 maxLines: 4,
 overflow: TextOverflow.ellipsis,
 textAlign: TextAlign.justify,
),
```

การใช้งาน Buttons

https://flutter.dev/docs/development/ui/widgets/material#Buttons

- RaisedButton

https://api.flutter.dev/flutter/material/RaisedButton-class.html

- FlatButton

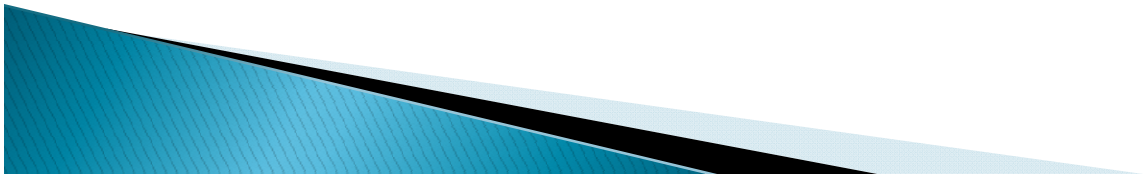
https://api.flutter.dev/flutter/material/FlatButton-class.html

- OutlineButton

https://api.flutter.dev/flutter/material/OutlineButton-class.html

- IconButton

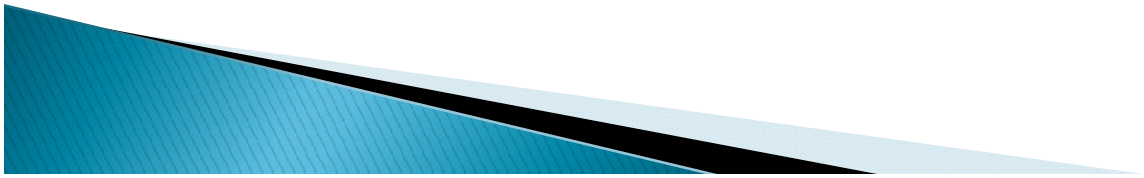
https://api.flutter.dev/flutter/material/IconButton-class.html



Icon

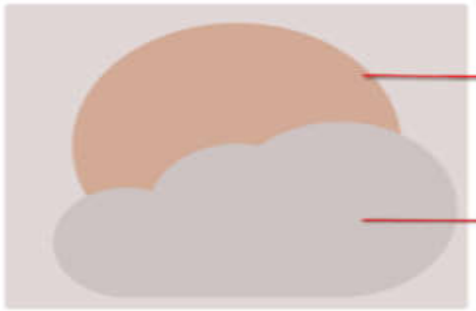
- https://api.flutter.dev/flutter/widgets/Icon-class.html

```
Icon(
 Icons.brush,
 color: Colors.lightBlue,
 size: 48.0,
),
```



Stack

- It’s used to layer widgets (or stack them) on-top of each other.
- https://api.flutter.dev/flutter/widgets/Stack-class.html

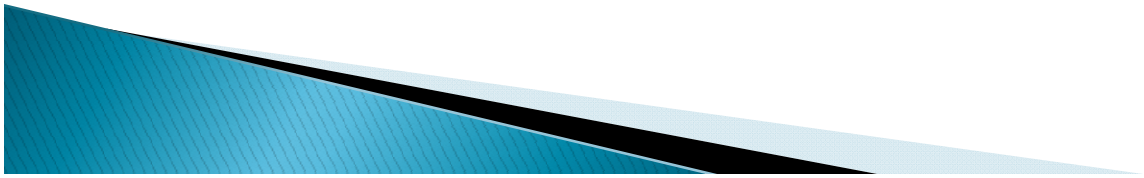


```
body: new Stack(
  children: <Widget>[
    Positioned(
      top: 100.0,
      child: Sun()),
    Positioned(
      top: 200.0,
      child: Clouds()),
  ],
),
```

Containers

- A widget that allows painting, positioning, and sizing.
- https://api.flutter.dev/flutter/widgets/Container-class.html

```
Center(
  child: Container(
    margin: const EdgeInsets.all(10.0),
    color: Colors.amber[600],
    width: 48.0,
    height: 48.0,
  ),
)
```



DECORATORS

- **Decorators** help to convey a message depending on the user's action or **customize the look and feel of a widget.**

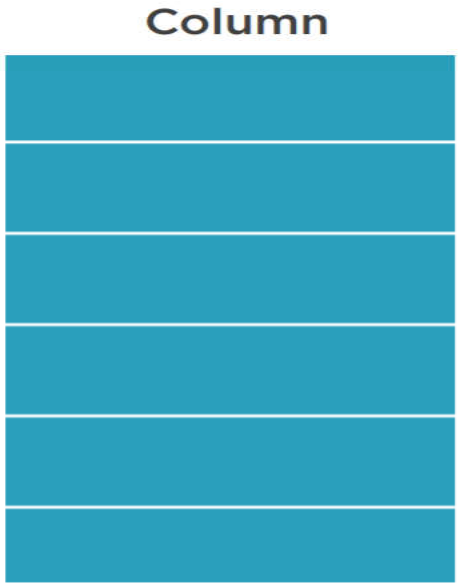
- **Decoration**—The base class to define other decorations.
- **BoxDecoration**—Provides many ways to draw a box with border, body, and boxShadow.
- **InputDecoration**—Used in TextField and TextFormField to customize the border, label, icon, and styles. This is a great way to give the user feedback on data entry, specifying a hint, an error, an alert icon, and more.

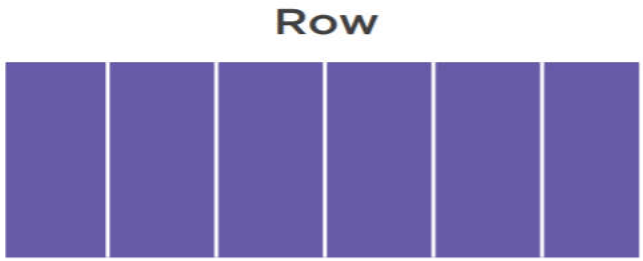
ตัวอย่างการใช้งาน DECORATORS

```
// BoxDecoration
Container(
 height: 100.0,
 width: 100.0,
 decoration: BoxDecoration(
  borderRadius: BorderRadius.all(Radius.circular(20.0)),
  color: Colors.orange,
  boxShadow: [
   BoxShadow(
    color: Colors.grey,
    blurRadius: 10.0,
    offset: Offset(0.0, 10.0),
   )
  ],
 ),
),
```

Rows and Columns

- https://api.flutter.dev/flutter/widgets/Column-class.html
- https://api.flutter.dev/flutter/widgets/Row-class.html





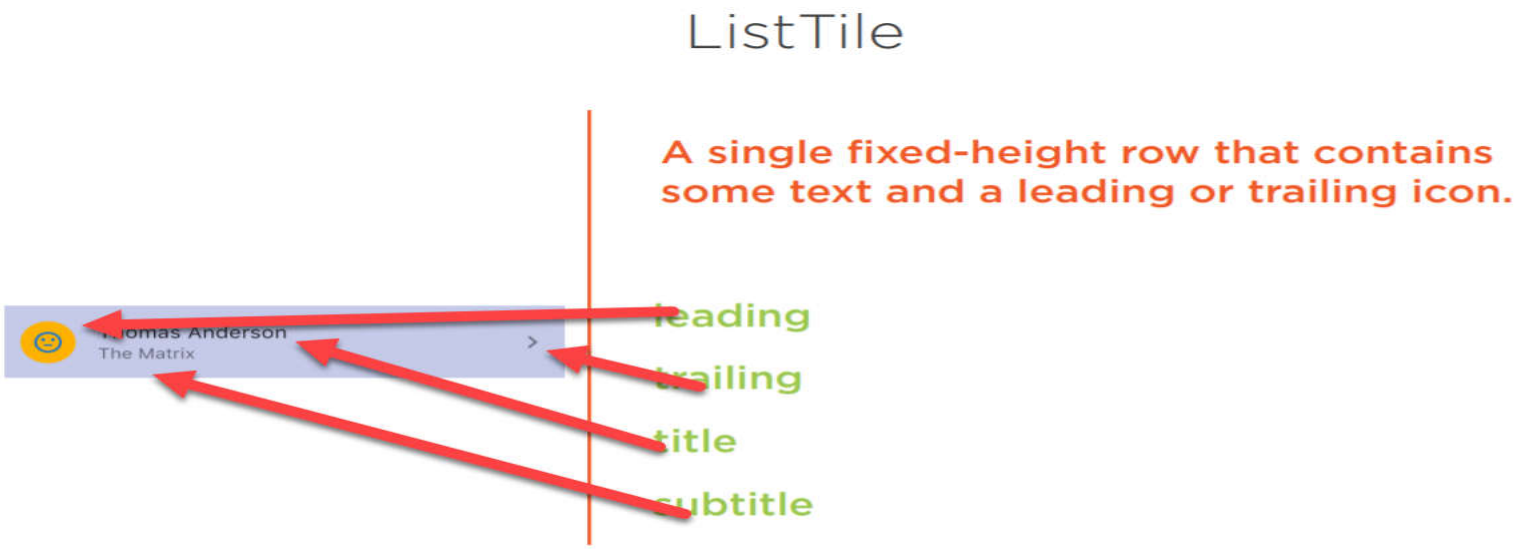
ตัวอย่างการใช้งาน Column และ Row

```
Column(
 crossAxisAlignment: CrossAxisAlignment.center,
 mainAxisAlignment: MainAxisAlignment.spaceEvenly,
 mainAxisSize: MainAxisSize.max,
 children: <Widget>[
  Text('Column 1'),
  Divider(),
  Text('Column 2'),
  Divider(),
  Text('Column 3'),
 ],
),
```

```
Row(
 crossAxisAlignment: CrossAxisAlignment.start,
 mainAxisAlignment: MainAxisAlignment.spaceEvenly,
 mainAxisSize: MainAxisSize.max,
 children: <Widget>[
  Row(
   children: <Widget>[
    Text('Row 1'),
    Padding(padding: EdgeInsets.all(16.0),),
    Text('Row 2'),
    Padding(padding: EdgeInsets.all(16.0),),
    Text('Row 3'),
   ],
  ),
 ],
),
```

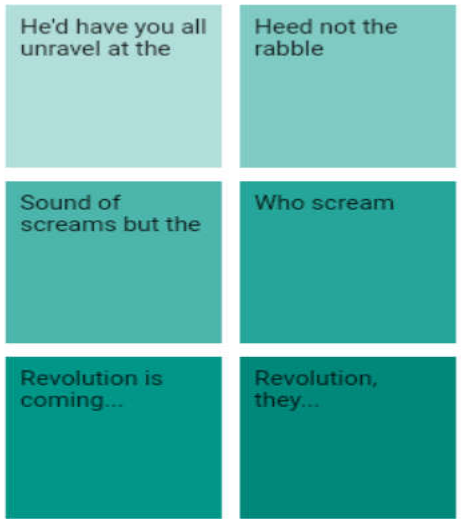
ListView

- A scrollable list or widgets, that you can arrange horizontally or vertically
- https://api.flutter.dev/flutter/widgets/ListView-class.html



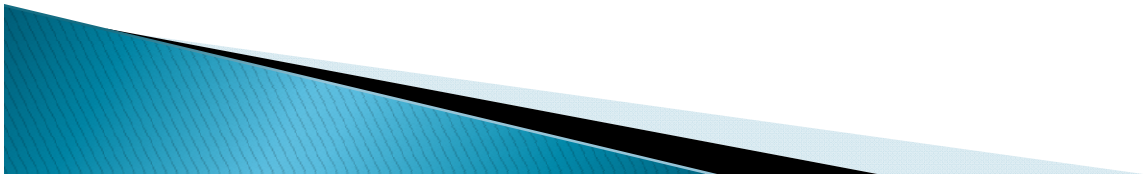
GridView

- A scrollable, 2D array of widgets
- https://api.flutter.dev/flutter/widgets/GridView-class.html



SingleChildScrollView

- https://api.flutter.dev/flutter/widgets/SingleChildScrollView-class.html



SafeArea

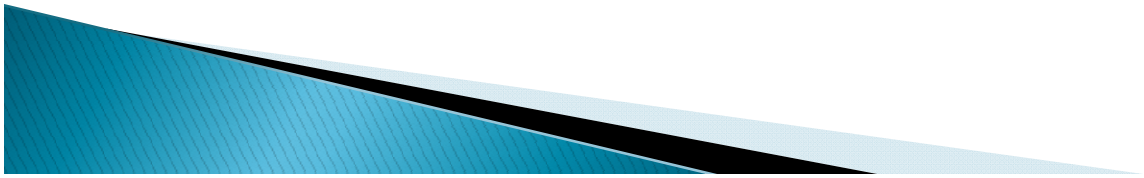
- The SafeArea widget is a must for today's devices such as the iPhone X or Android devices with a notch.

  https://api.flutter.dev/flutter/widgets/SafeArea-class.html

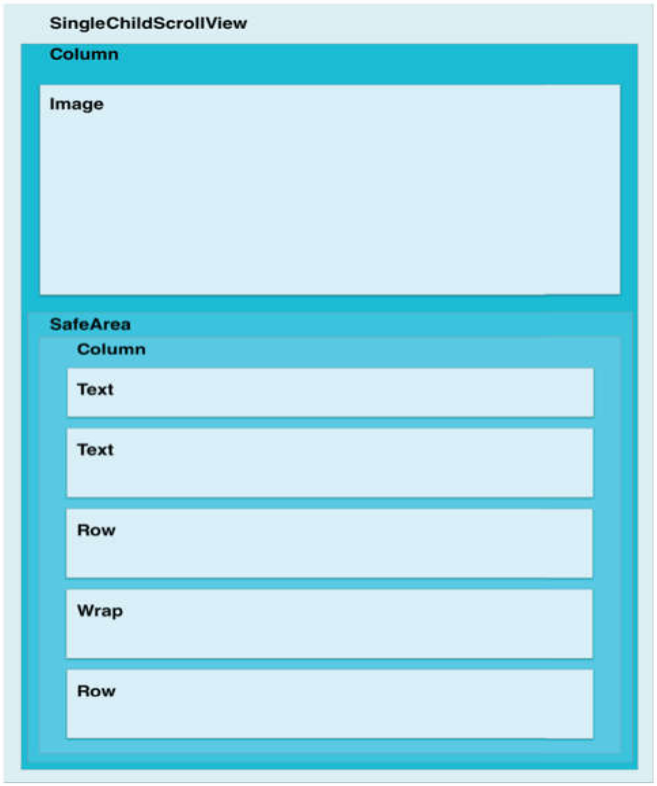
```
body: Padding(
 padding: EdgeInsets.all(16.0),
 child: SafeArea(
  child: SingleChildScrollView(
   child: Column(
    children: <Widget>[

    ],
   ),
  ),
 ),
),
```

แบบฝึกหัด

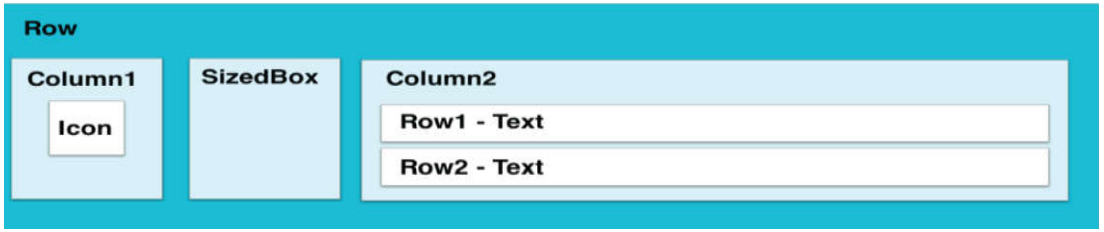
- ทดลองใช้งาน Widgets แบบต่างๆ



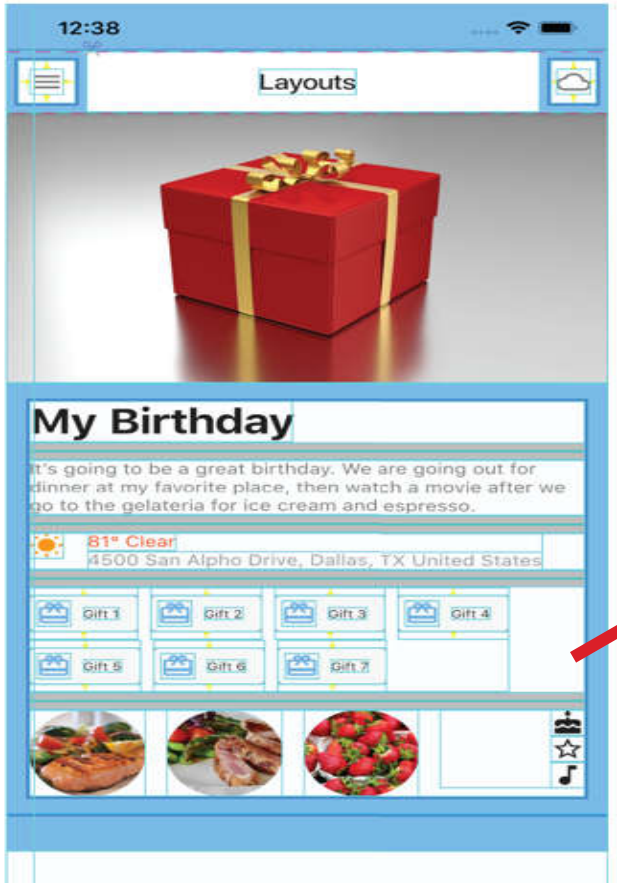
การออกแบบ Layout

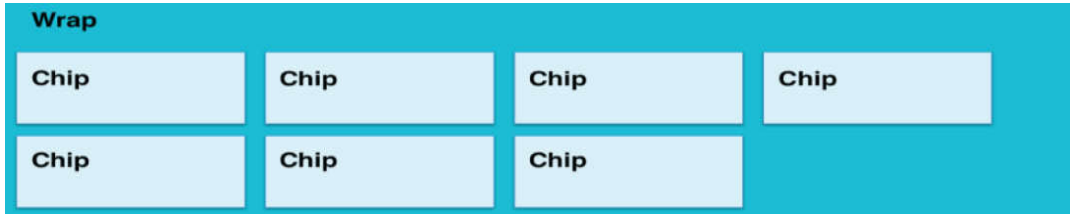


การออกแบบ Layout



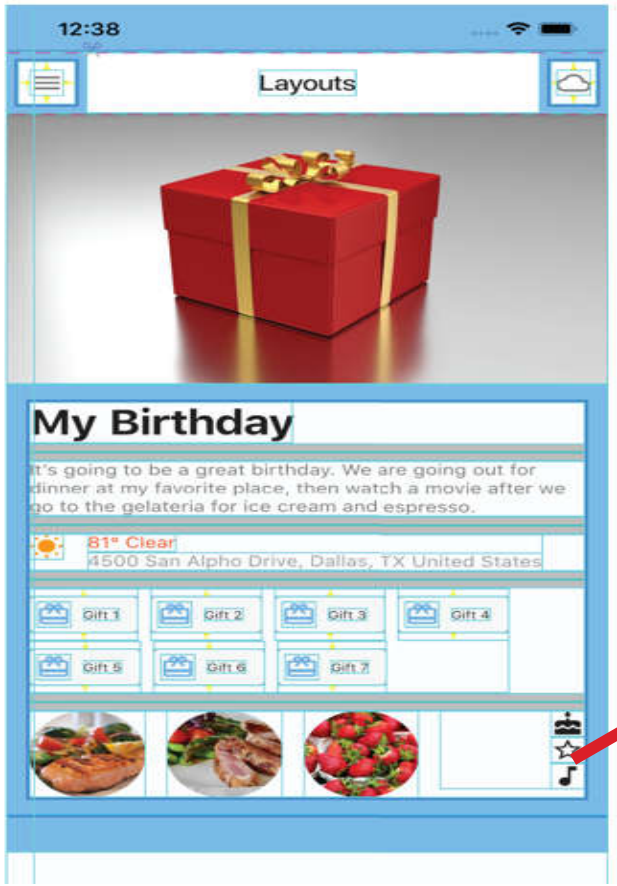
การออกแบบ Layout

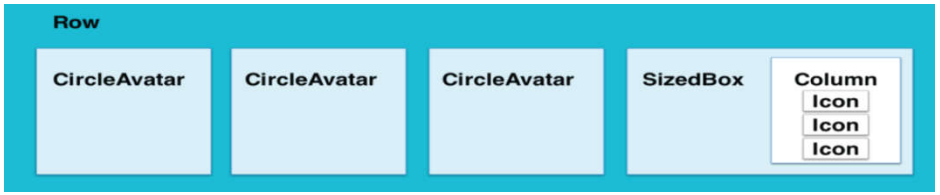




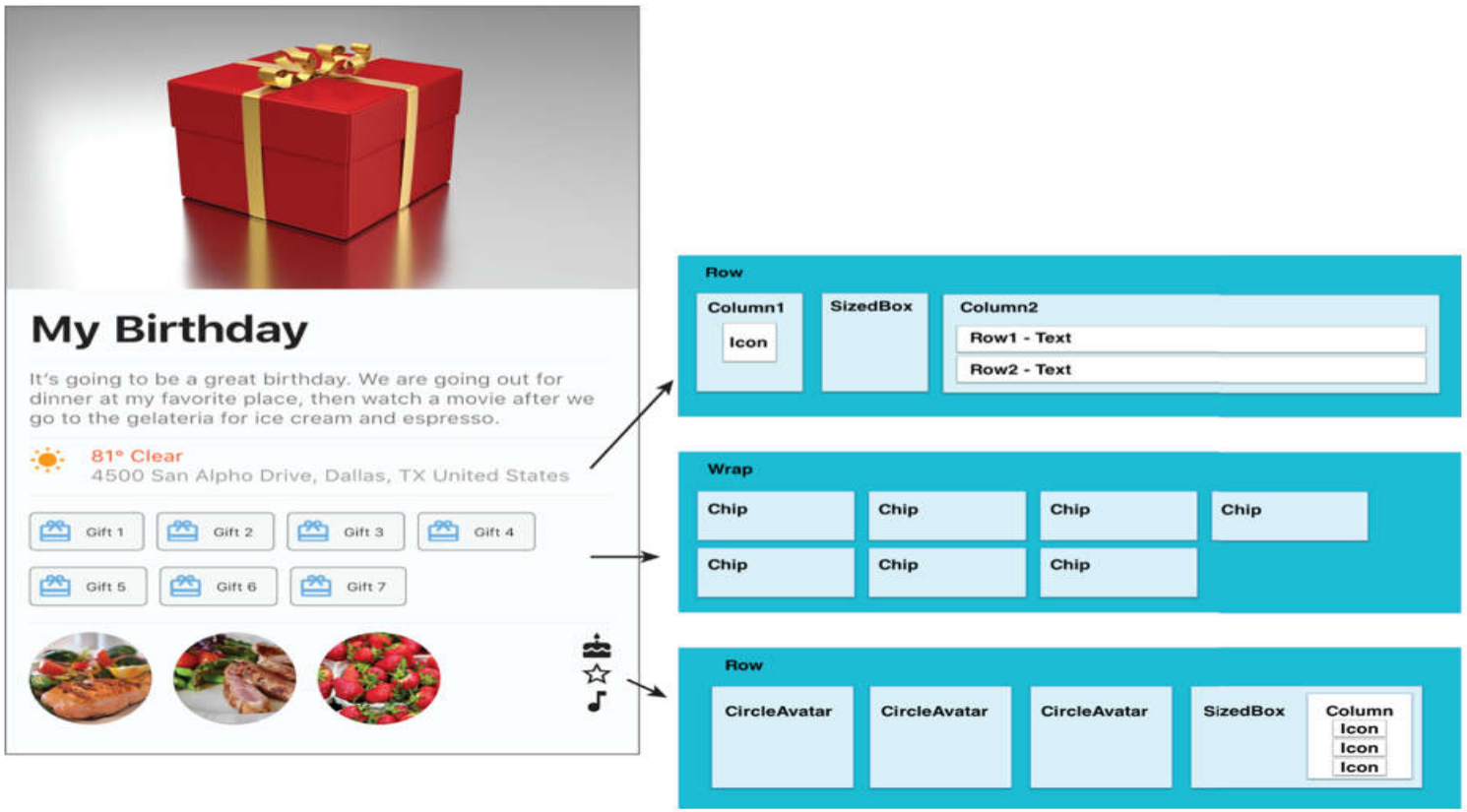

```
Chip(
 label: Text('Vacation'),
 avatar: Icon(Icons.local_airport),
 shape: RoundedRectangleBorder(
  borderRadius: BorderRadius.circular(4.0),
  side: BorderSide(color: Colors.grey),
 ),
 backgroundColor: Colors.grey.shade100,);
```

การออกแบบ Layout

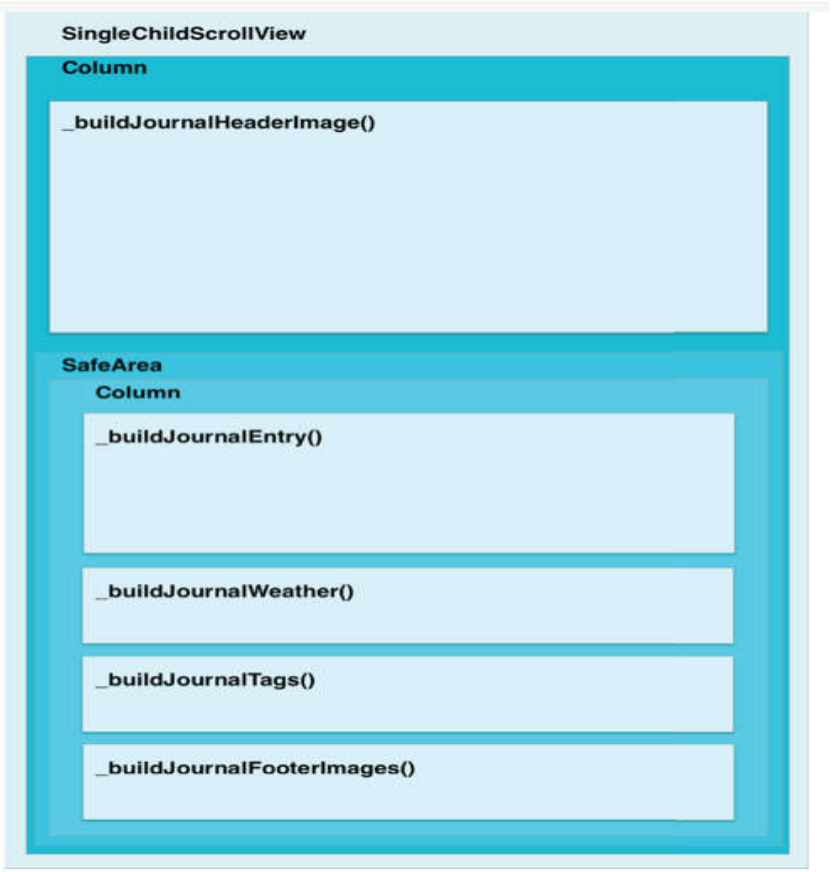




การออกแบบ Layout



การออกแบบ Layout



แบบฝึกหัด

- ทดลองสร้าง Layout

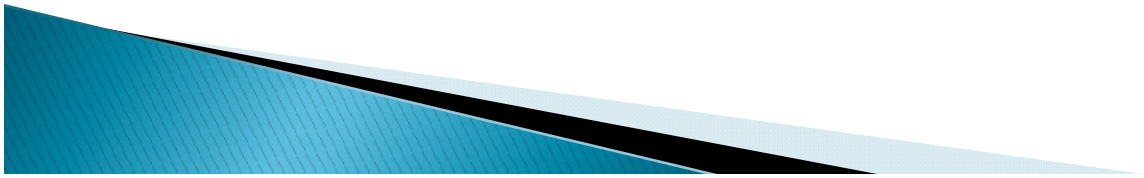
```
Widget _buildBody() {
 return SingleChildScrollView(
  child: Column(
   children: <Widget>[
    _buildJournalHeaderImage(),
    SafeArea(
     child: Padding(
      padding: EdgeInsets.all(16.0),
      child: Column(          crossAxisAlignment: CrossAxisAlignment.start,
       children: <Widget>[
        _buildJournalEntry(),
        Divider(),
        _buildJournalWeather(),
        Divider(),
        _buildJournalTags(),
        Divider(),
        _buildJournalFooterImages(),
       ],
      ),
     ),
    ),
   ],
  )
```

CHECKING ORIENTATION

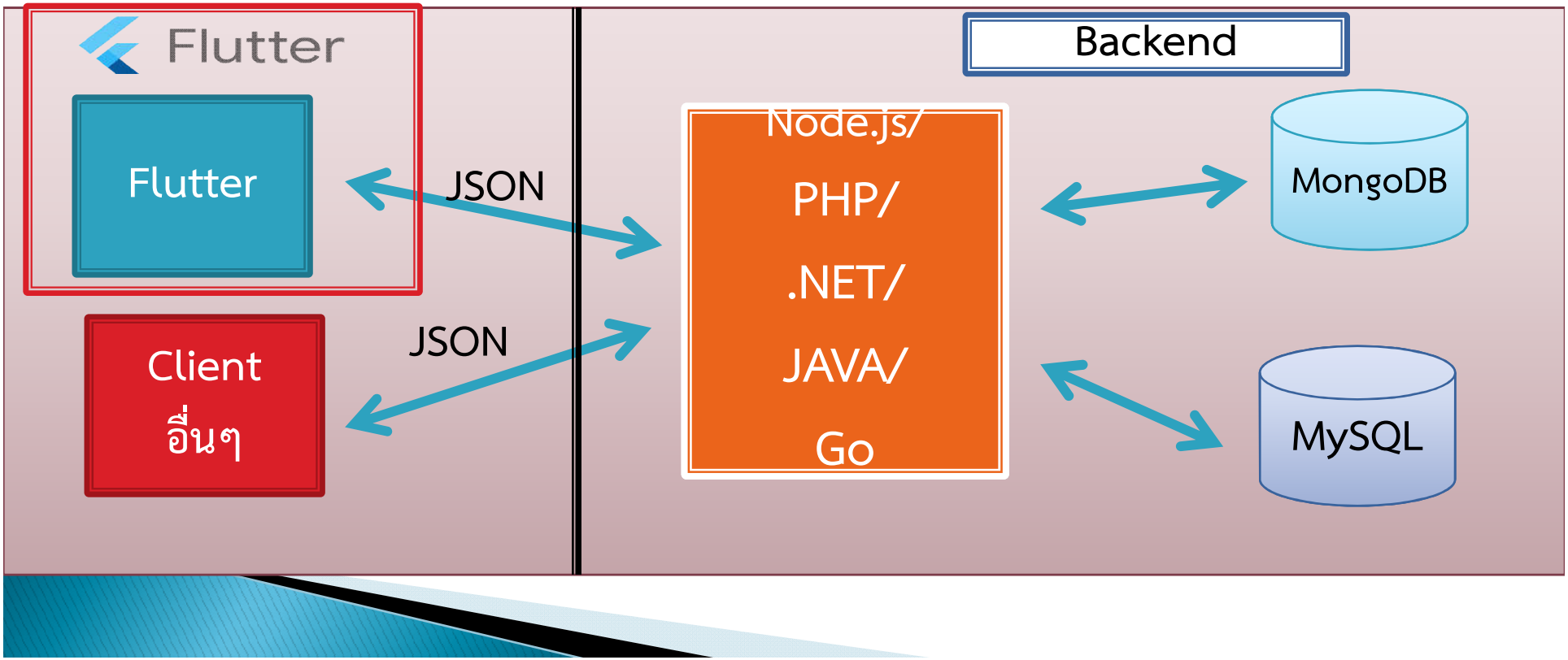
- . There are two ways to figure out orientation, MediaQuery.of(context).orientation and OrientationBuilder.

```
@override
Widget build(BuildContext context) {
 Orientation _orientation = MediaQuery.of(context).orientation;

 return _orientation == Orientation.portrait
   ? Container(
  alignment: Alignment.center,
  color: Colors.yellow,
  height: 100.0,
  width: 100.0,
  child: Text('Portrait'),
 )
   : Container(
  alignment: Alignment.center,
  color: Colors.lightGreen,
  height: 100.0,
  width: 200.0,
  child: Text('Landscape'),
 );
 }
}
```

การทำงานกับ Backend และเชื่อมต่อ REST APIs/Web Services

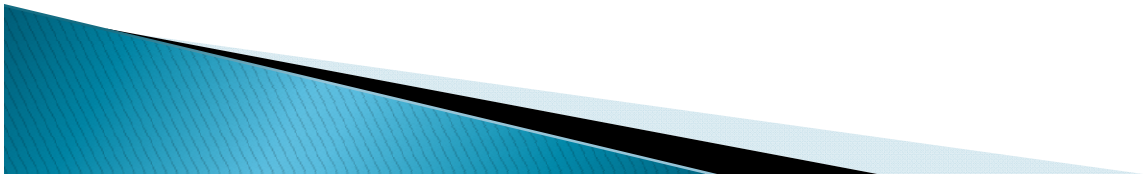


ภาพใหญ่ของหลักสูตรนี้



ขั้นตอนการติดต่อกับ Backend

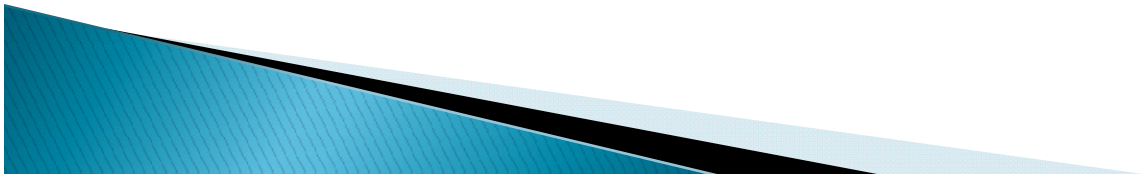
- 1. ติดตั้ง http package ก่อน
- https://pub.dev/packages/http
- 2. สร้าง model เพื่อรับข้อมูลจาก backend (ไม่สร้างก็ได้ใช้ dynamic)
- https://javiercbk.github.io/json_to_dart/
- 3. เขียนเพื่อดึงข้อมูล และแสดงผล



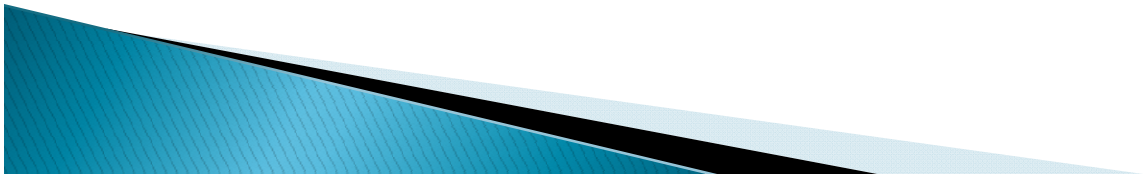
ขั้นตอนการติดต่อกับ Backend

- AndroidManifest.xml file, add the Internet permission.

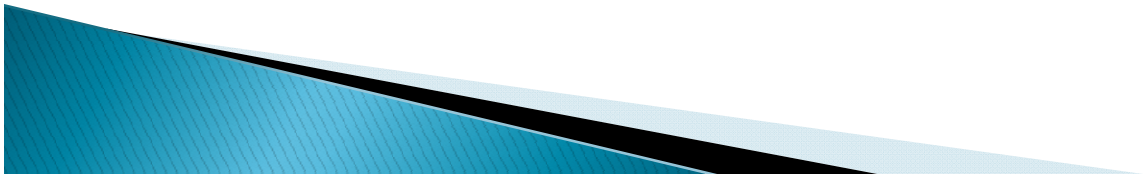
- <uses-permission android:name="android.permission.INTERNET" />



ทดลองแสดงผลข้อมูลจาก Backend



การจัดการฟอร์ม



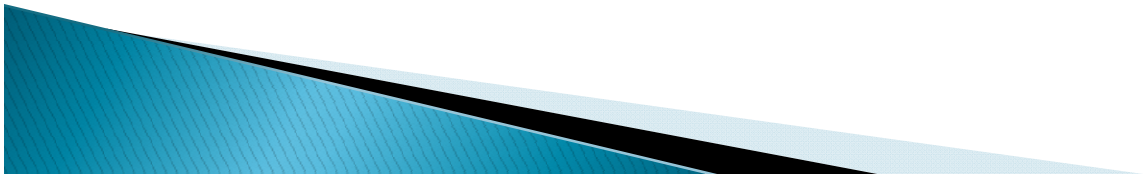
การใช้งานฟอร์ม

การจัดการ และใช้งานฟอร์ม

- การสร้างฟอร์มด้วย และตรวจสอบความถูกต้องของข้อมูล

- **Flutter FormBuilder**
- https://pub.dev/packages/flutter_form_builder

- แสดงผลและแจ้งเตือนข้อมูลให้กับผู้ใช้ (User Notification) ด้วย **Flushbar**
- https://pub.dev/packages/flushbar



The end