

Mobile Flutter

1. Flutter คืออะไร

- Native: iOS = Swift / Android = Kotlin
- Cross-platform: เขียนครั้งเดียว ได้ 2 ระบบ

2. Flutter Concept ที่ต้องรู้

- Everything is Widget

```
Text('Hello Flutter');
```

- Widget Tree

```
Scaffold(  
  appBar: AppBar(title: Text('Home')),  
  body: Center(  
    child: Text('Hello'),  
  ),  
);
```

- Build flow (Widget → Build → Frame)

```
Widget → build() → UI Frame
```

- Hot reload vs Hot restart

Hot reload: UI เปลี่ยน / state ยังอยู่

Hot restart: reset ใหม่หมด

3. Widget พื้นฐาน

- Text
- Icon
- Image
- Scaffold
- AppBar

```
Scaffold(  
  appBar: AppBar(title: Text('My App')),  
  body: Column(  
    children: [  
      Text('Hello'),  
      Icon(Icons.star),  
      Image.network('https://picsum.photos/200'),  
    ],  
  ));
```

Dart พื้นฐาน + Layout ที่ใช้จริง

เป้าหมาย

- อ่าน Dart ออก
- เข้าใจโครงสร้าง layout

Dart ที่จำเป็น

- var / final

```
var name = 'John';  
final age = 20;
```

- int / double / String / bool

```
int a = 10;  
double b = 10.5;  
String c = 'Flutter';  
bool isActive = true;
```

- List / Map

```
List<int> numbers = [1, 2, 3];  
Map<String, dynamic> user = {  
  'name': 'John',  
  'age': 20,  
};
```

- function

```
int sum(int a, int b) {  
  return a + b;  
}
```

- if / for

```
if (age > 18) {  
  print('Adult');  
}
```

```
for (var n in numbers) {  
  print(n);  
}
```

- class

```
class User {  
  final String name;  
  final int age;  
  User({required this.name, required this.age});  
}
```

Flutter Layout

- Row / Column

```
Row(  
  children: [  
    Text('A'),  
    Text('B'),  
  ],  
);
```

- Container
- Padding
- SizedBox

```
Container(  
  padding: EdgeInsets.all(16),  
  color: Colors.blue,  
  child: Text('Box'),  
);
```

- Expanded

```
Row(  
  children: [  
    Expanded(child: Text('Left')),  
    Text('Right'),  
  ],  
);
```

UI จริง + Interaction

Widgets

- TextField

```
TextField(  
  decoration: InputDecoration(  
    labelText: 'Email',  
  ),  
);
```

- ElevatedButton

```
ElevatedButton(  
  onPressed: () {  
    print('Clicked');  
  },  
  child: Text('Login'),  
);
```

- InkWell
- GestureDetector

```
InkWell(  
  onTap: () {  
    print('Tapped');  
  },  
  child: Text('Click me'),  
);
```

- ListView

```
ListView.builder(  
  itemCount: 3,  
  itemBuilder: (_, i) {  
    return ListTile(  
      title: Text('Item $i'),  
    );  
  },  
);
```

State Management ด้วย GetX

เป้าหมาย

- เข้าใจ Controller
- รู้จัก Rx / Obx
- อ่านโค้ด GetX ได้

สอนแบบ “ไม่ทฤษฎี”

1. ปัญหาถ้าไม่มี State

- setState วุ่นวาย
- ส่งค่าข้าม widget ยาก

2. GetX Concept

- Controller = logic
- View = UI
- Rx = state

Controller

```
class CounterController extends GetxController {  
  final count = 0.obs;  
  
  void increment() {  
    count.value++;  
  }  
}
```

View + Obx

```
final c = Get.put(CounterController());
```

```
Obx(() => Text('${c.count.value}'));
```

Button

```
ElevatedButton(  
  onPressed: c.increment,  
  child: Text('Add'),  
);
```

โครงสร้างโปรเจกต์จริง + API

```
lib/  
└─ modules/  
    └─ user/  
        ├── controller/  
        ├── view/  
        └─ model/
```

View → วาด UI

Controller → คำนวณ state + call API

Model → แปลง JSON

การเชื่อมต่อ API ด้วย Dio (Basic)

Model (แปลง JSON → Object) แปลงในเว็บได้ json to dart

```
class UserModel {  
  final int id;  
  final String name;  
  final String email;  
  
  UserModel({  
    required this.id,  
    required this.name,  
    required this.email,  
  });  
  
  factory UserModel.fromJson(Map<String, dynamic> json) {  
    return UserModel(  
      id: json['id'],  
      name: json['name'],  
      email: json['email'],  
    );  
  }  
}
```

Controller (เรียก API)

```
class UserController extends GetxController {  
  final users = <UserModel>[].obs;  
  final isLoading = false.obs;  
  
  final dio = Dio();  
  
  @override  
  void onInit() {
```

```

loadUsers();
super.onInit();
}

Future<void> loadUsers() async {
  try {
    isLoading.value = true;

    final response = await dio.get(
      'https://jsonplaceholder.typicode.com/users',
    );

    final data = response.data as List;

    users.value = data
      .map((e) => UserModel.fromJson(e))
      .toList();
  } catch (e) {
    Get.snackbar('Error', 'โหลดข้อมูลไม่สำเร็จ');
  } finally {
    isLoading.value = false;
  }
}
}

```

View (แสดงผล)

```

class UserView extends StatelessWidget {
  UserView({super.key});

  final controller = Get.put(UserController());

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: const Text('Users')),
      body: Obx(() {
        if (controller.isLoading.value) {
          return const Center(
            child: CircularProgressIndicator(),
          );
        }

        return ListView.builder(
          itemCount: controller.users.length,
          itemBuilder: (_, index) {
            final user = controller.users[index];

```

```
        return ListTile(  
            title: Text(user.name),  
            subtitle: Text(user.email),  
        );  
    },  
);  
}),  
);  
}  
}
```