



# GetX State Management Class Notes

এই নোটগুলো আজকের ক্লাসে শেখানো কোড ও কনসেপ্টের উপর ভিত্তি করে তৈরি করা হয়েছে। এখানে আমরা GetX ব্যবহারের মাধ্যমে Flutter অ্যাপে State Management এবং Routing কীভাবে কাজ করে, তা বিস্তারিত ব্যাখ্যা করব।



## ১. GetX কী?

GetX হলো একটি lightweight এবং powerful Flutter framework, যা বিভিন্ন কাজের জন্য ব্যবহৃত হয়:

- State Management
- Route Management (Navigation)
- Dependency Injection

→ সহজে, কম কোডে এবং performance-friendly অ্যাপ বানাতে GetX অনেক সাহায্য করে।



## ২. কেন GetX State Management এর জন্য ভালো?

- Boilerplate কোড কম লাগে
- Reactive এবং Simple State Management
- Controller তৈরি করে UI সহজে update করা যায়
- Performance friendly (minimum rebuilds)
- Navigation ও Dependency Injection একই ফ্রেমওয়ার্কে পাওয়া যায়



## ৩. কোড স্ট্রাকচার:

### ◆ main.dart

```
void main() {  
  runApp(const MyApp());  
}  
  
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return GetMaterialApp(  
      home: FirstPage(),  
    );  
  }  
}
```

```
}  
}
```

 ব্যাখ্যা:

- `GetMaterialApp` ব্যবহার করলে `GetX` এর সমস্ত সুবিধা (state management, routing, snackbar, dialog ইত্যাদি) পাওয়া যায়।

## 8. View & Controller কিভাবে কানেক্ট করা হয়?

### ◆ View: FirstPage

```
return GetBuilder<FirstPageController>(  
  init: FirstPageController(),  
  builder: (controller) {  
    // UI Code  
  },  
);
```

 ব্যাখ্যা:

- `GetBuilder` হল এক ধরনের widget যা controller এর সাথে যুক্ত থাকে।
- `init:` দিয়ে কোন controller initialize করা হবে তা নির্ধারণ করা হয়।
- `controller.update()` করলে UI rebuild হয়।

## ৫. State Management উদাহরণ (Counter Example)

### ◆ Controller: FirstPageController

```
class FirstPageController extends GetxController {  
  int _counter = 0;  
  int get counter => _counter;  
  set counter(int value) {  
    _counter = value;  
  }  
}
```

### ◆ UI থেকে State Change

```
controller.counter++;
```

```
controller.update(); // UI Update
```

 ব্যাখ্যা:

- Controller এ থাকা ভ্যারিয়েবল UI থেকে পরিবর্তন করে update() দিলে UI তে reflect করে।

## ৬. Page Navigation

### ◆ Navigation with Data Pass

```
Get.to(() => SecondPage(counter: controller.counter));
```

 ব্যাখ্যা:

- `Get.to()` দিয়ে নতুন পেইজে যাওয়া যায় (push) এবং data pass করা যায়।

### ◆ Named Routing (Extra Info)

#### ✓ `Get.toNamed()`

```
Get.toNamed('/second');
```

#### ✓ `Get.offNamed()`

```
Get.offNamed('/second');
```

#### ✓ `Get.offAllNamed()`

```
Get.offAllNamed('/home');
```


 ব্যাখ্যা:

- `Get.toNamed` : নতুন পেইজে যায়
- `Get.offNamed` : আগের পেইজ remove করে নতুনটাতে যায়
- `Get.offAllNamed` : সব পেইজ remove করে নতুনটাতে যায়

## Second Page Example

```
class SecondPage extends StatelessWidget {
  final int counter;

  @override
  Widget build(BuildContext context) {
    return GetBuilder<SecondPageController>(
      init: SecondPageController(),
      builder: (controller) {
        return Scaffold(
          body: Text('Counter: $counter'),
        );
      },
    );
  }
}
```

 ব্যাখ্যা:

- Constructor দিয়ে ডাটা নিচ্ছে (counter)।

## Extra Controller Example

```
class SecondPageController extends GetxController {
  int counter = 0;

  void incrementCounter() {
    counter++;
  }
}
```

## Summary Table

বিষয়	ব্যাখ্যা
GetMaterialApp	GetX এর সুবিধা পেতে ব্যবহার করা হয়
GetBuilder	Controller এর সাথে Widget কানেক্ট করে
controller.update()	UI update করে
Get.to()	নতুন পেইজে যায়
Get.toNamed()	নাম দিয়ে route করে

বিষয়	ব্যাখ্যা
<code>Get.offNamed()</code>	আগের route replace করে
<code>Get.offAllNamed()</code>	সব কিছু replace করে নতুন পেইজে যায়

---

## Tips

---

- Always separate `Controller` and `View` (MVC Pattern)
- Controller এ logic রাখো, View শুধু UI দিক
- Routing এবং State Management GetX দিয়ে অনেক clean হয়