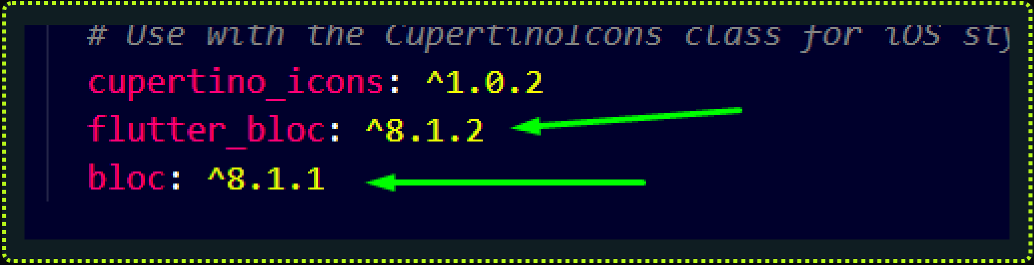
How to use BLOC PATTERN in FLUTTER

Obaid Ullah
Expert Flutter Developer

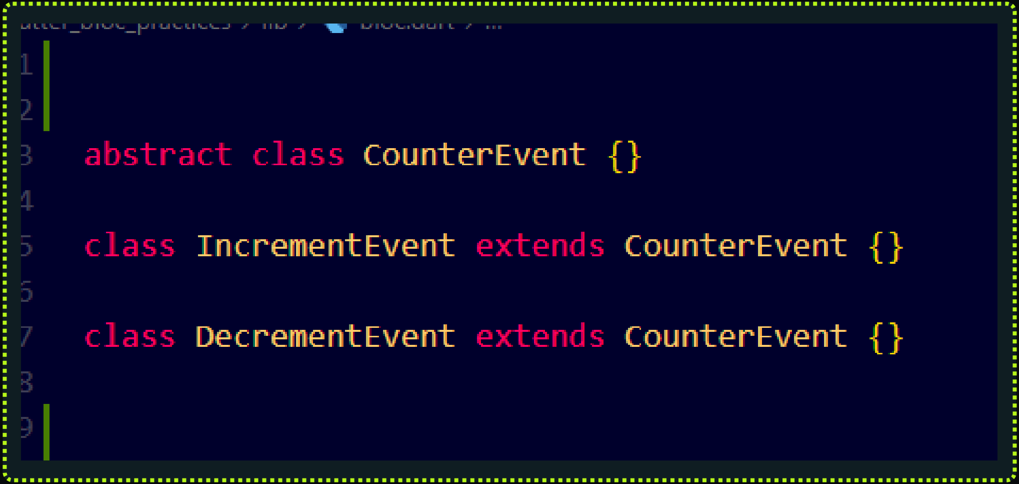
SWIPE

Add the Bloc packages (bloc) & (flutter_bloc) to your project's pubspec.yaml file.



```
# Use with the CupertinoIcons class for iOS sty
cupertino_icons: ^1.0.2
flutter_bloc: ^8.1.2
bloc: ^8.1.1
```

Define the increment and Decrement events that extends the counter event class to update the value.

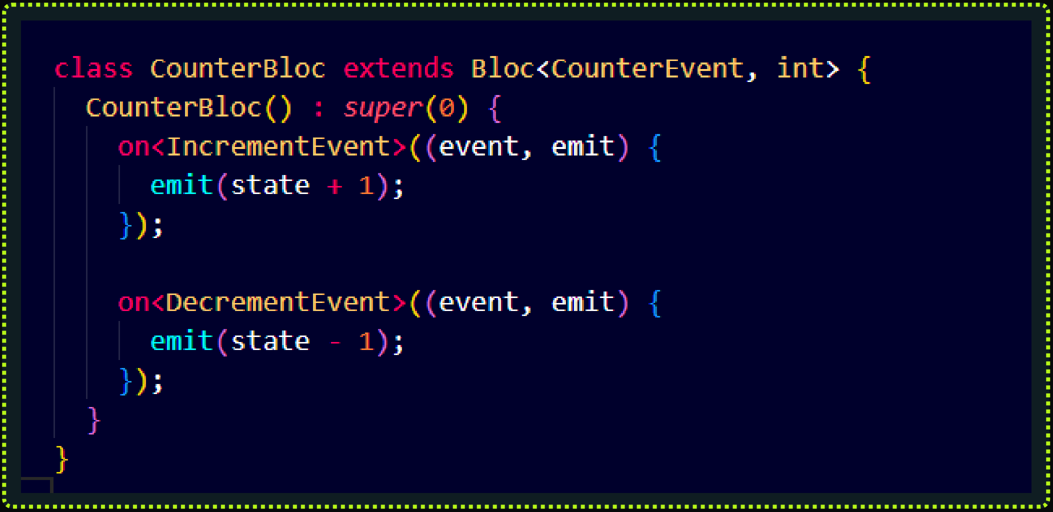


```
abstract class CounterEvent {}

class IncrementEvent extends CounterEvent {}

class DecrementEvent extends CounterEvent {}
```

Now, you need to create a bloc counter class to handle the state and events to change its state (increament or decremnet the counter value).



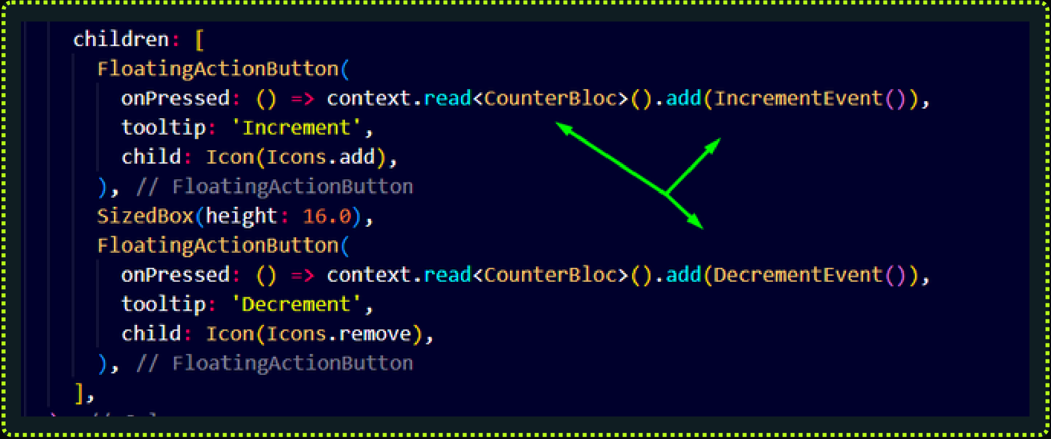
```
class CounterBloc extends Bloc<CounterEvent, int> {
  CounterBloc() : super(0) {
    on<IncrementEvent>((event, emit) {
      emit(state + 1);
    });

    on<DecrementEvent>((event, emit) {
      emit(state - 1);
    });
  }
}
```

To update the bloc's state, you need to send events to it. In this counter example, we want to increment/decrement the counter when the user presses the "+", "-" button, so we call

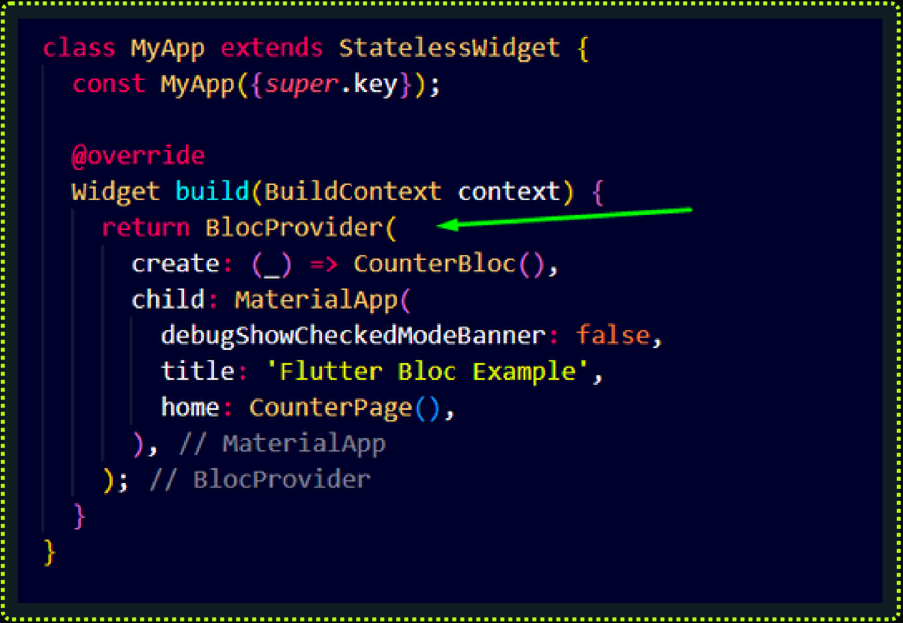
context.read<CounterBloc>().add(IncrementEvent())

context.read<CounterBloc>().add(DecrementEvent())



```
children: [
  FloatingActionButton(
    onPressed: () => context.read<CounterBloc>().add(IncrementEvent()),
    tooltip: 'Increment',
    child: Icon(Icons.add),
  ), // FloatingActionButton
  SizedBox(height: 16.0),
  FloatingActionButton(
    onPressed: () => context.read<CounterBloc>().add(DecrementEvent()),
    tooltip: 'Decrement',
    child: Icon(Icons.remove),
  ), // FloatingActionButton
],
```

Wrap your material widget tree with BlocProvider and pass it to the bloc instance you want to provide.



```
class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return BlocProvider(
      create: (_) => CounterBloc(),
      child: MaterialApp(
        debugShowCheckedModeBanner: false,
        title: 'Flutter Bloc Example',
        home: CounterPage(),
      ), // MaterialApp
    ); // BlocProvider
  }
}
```

I hope this example helps you understand how to use the BLoC pattern in Flutter.

Keep in mind that this is just one way to use BLoC, and there are many other approaches and variations that you can use depending on your app's needs.

Obaid Ullah
Expert Flutter Developer

SWIPE

If you want

FLUTTER

APP DEVELOPMENT

to execute your ideas into real-time mobile apps

DM ME for a Free Consultation Call

OBAID ULLAH

SAVE

FOR LATER