Bloc

Código

Main.dart

```
import 'package:flutter/material.dart';
import 'bloc/counter_bloc.dart';
//ejemplo de repository

//https://www.youtube.com/watch?v=4oAwnbZOMKE&t=702s

//hacer otro ejemplo
//libreria bloc
//https://github.com/Vikkybliz/counter/blob/master/lib/main.dart
// Otro ejemplo con visualCOde
//https://www.dbestech.com/tutorials/flutter-bloc-
example  https://www.youtube.com/watch?v=Auh7fVk_CX4
void main() {
  runApp(const MyApp());
}

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

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        // This is the theme of your application.
        //
        // Try running your application with "flutter run". You'll see
the
        // application has a blue toolbar. Then, without quitting the
app, try
        // changing the primarySwatch below to Colors.green and then
invoke
        // "hot reload" (press "r" in the console where you ran "flutter
run",
        // or simply save your changes to "hot reload" in a Flutter IDE).
        // Notice that the counter didn't reset back to zero; the
application
```

Nombre: Alex Reyes

```
        // is not restarted.
        primarySwatch: Colors.blue,
      ),
      home: const MyHomePage(title: 'Flutter Demo Home Page'),
    );
  }
}

class MyHomePage extends StatefulWidget {
  const MyHomePage({super.key, required this.title});

  // This widget is the home page of your application. It is stateful,
meaning
  // that it has a State object (defined below) that contains fields that
affect
  // how it looks.

  // This class is the configuration for the state. It holds the values
(in this
  // case the title) provided by the parent (in this case the App widget)
and
  // used by the build method of the State. Fields in a Widget subclass
are
  // always marked "final".

  final String title;

  @override
  State<MyHomePage> createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  //int _counter = 0;
  CounterBloc _bloc = CounterBloc();
  @override
  void dispose() {
    // TODO: implement dispose
    _bloc.dispose();
    super.dispose();
  }
/*
  void _incrementCounter() {
    setState(() {
      // This call to setState tells the Flutter framework that something
has
      // changed in this State, which causes it to rerun the build method
below
      // so that the display can reflect the updated values. If we
changed
```

Nombre: Alex Reyes

```
      // _counter without calling setState(), then the build method would
not be
      // called again, and so nothing would appear to happen.
      _counter++;
    });
  }

  void _clearCounter() {
    setState(() {
      _counter = 0;
    });
  }*/

  @override
  Widget build(BuildContext context) {
    // This method is rerun every time setState is called, for instance
as done
    // by the _incrementCounter method above.
    //
    // The Flutter framework has been optimized to make rerunning build
methods
    // fast, so that you can just rebuild anything that needs updating
rather
    // than having to individually change instances of widgets.
    return Scaffold(
      appBar: AppBar(
        // Here we take the value from the MyHomePage object that was
created by
        // the App.build method, and use it to set our appbar title.
        title: Text(widget.title),
      ),
      body: Center(
        // Center is a layout widget. It takes a single child and
positions it
        // in the middle of the parent.
        child: Column(
          // Column is also a layout widget. It takes a list of children
and
          // arranges them vertically. By default, it sizes itself to fit
its
          // children horizontally, and tries to be as tall as its
parent.
          //
          // Invoke "debug painting" (press "p" in the console, choose
the
          // "Toggle Debug Paint" action from the Flutter Inspector in
Android
          // Studio, or the "Toggle Debug Paint" command in Visual Studio
Code)
```

Nombre: Alex Reyes

```
          // to see the wireframe for each widget.
          //
          // Column has various properties to control how it sizes itself
and
          // how it positions its children. Here we use mainAxisAlignment
to
          // center the children vertically; the main axis here is the
vertical
          // axis because Columns are vertical (the cross axis would be
          // horizontal).
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            const Text(
              'You have pushed the button this many times:',
            ),
            StreamBuilder<int>(
                stream: _bloc.counterStream,
                initialData: 0,
                builder: (context, snapshot) {
                  return Text(
                    '${snapshot.data}',
                    style: Theme.of(context).textTheme.headline4,
                  );
                }),
            /*Text(
              '$_counter',
              style: Theme.of(context).textTheme.headline4,
            ),*/
          ],
        ),
      ),
      floatingActionButton: Row(
        mainAxisAlignment: MainAxisAlignment.end,
        children: [
          FloatingActionButton(
            heroTag: "Button1",
            //onPressed: _clearCounter,
            onPressed: () {
              _bloc.sendEvent.add(DecrementCounter());
            },
            tooltip: 'Clear',
            child: const Icon(Icons.clear),
          ),
          const SizedBox(
            width: 50,
          ),
          FloatingActionButton(
            heroTag: "Button2",
            //onPressed: _incrementCounter,
```

Nombre: Alex Reyes

```
            onPressed: () {
              _bloc.sendEvent.add(IncrementCounter());
            },
            tooltip: 'Increment',
            child: const Icon(Icons.add),
          ),
          const SizedBox(
            width: 50,
          ),
          FloatingActionButton(
            heroTag: "Button3",
            //onPressed: _incrementCounter,
            onPressed: () {
              _bloc.sendEvent.add(PotCounter());
            },
            tooltip: 'Increment',
            child: const Icon(Icons.add),
          ),
          const SizedBox(
            width: 50,
          ),
          FloatingActionButton(
            heroTag: "Button4",
            //onPressed: _incrementCounter,
            onPressed: () {
              _bloc.sendEvent.add(MultCounter());
            },
            tooltip: 'Increment',
            child: const Icon(Icons.add),
          ),
        ],
      ), // This trailing comma makes auto-formatting nicer for build
methods.
    );
  }
}
```

Nombre: Alex Reyes

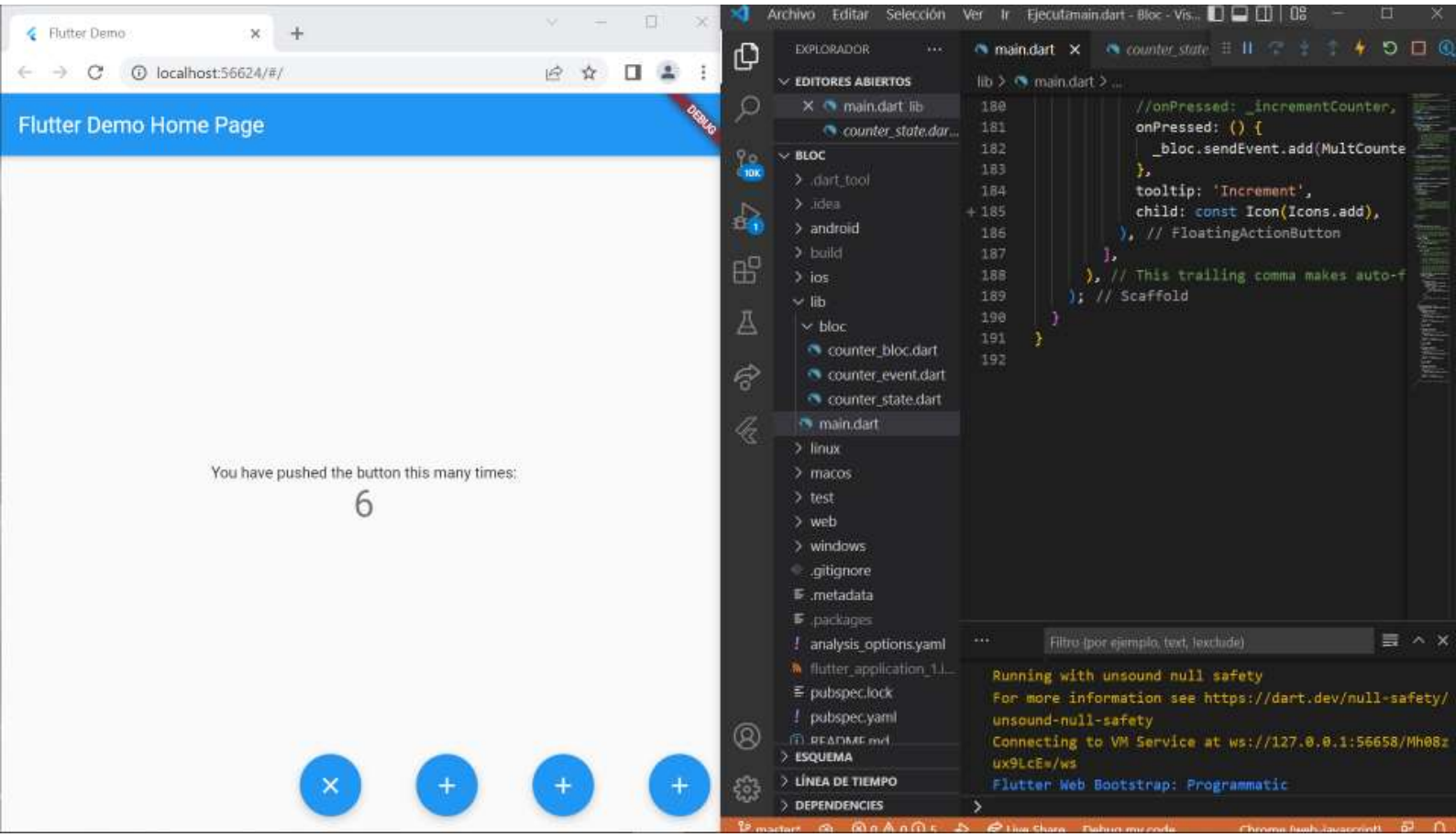
counter_bloc.dart

```dart
/*import 'package:bloc/bloc.dart';
import 'package:meta/meta.dart';

part 'counter_event.dart';
part 'counter_state.dart';

class CounterBloc2 extends Bloc<CounterEvent, CounterState> {
  CounterBloc2() : super(CounterInitial()) {
    on<CounterEvent>((event, emit) {
      // TODO: implement event handler
    });
  }
}*/
import "dart:async";

class CounterBase {}

class IncrementCounter extends CounterBase {}

class DecrementCounter extends CounterBase {}

class MultCounter extends CounterBase {}

class PotCounter extends CounterBase {}

class CounterBloc {
  int _count = 0;
  StreamController<CounterBase> _input = StreamController();
  StreamController<int> _output = StreamController();
  Stream<int> get counterStream => _output.stream;
  StreamSink<CounterBase> get sendEvent => _input.sink;

  CounterBloc() {
    _input.stream.listen(_onEvent);
  }

  void dispose() {
    _input.close();
    _output.close();
  }

  void _onEvent(CounterBase event) {
    if (event is IncrementCounter) {
      _count++;
    }
    if (event is MultCounter) {
      _count = (_count * 2);
```

Nombre: Alex Reyes

```
    }
    if (event is PotCounter) {
      _count = (_count * _count);
    } else if (event is DecrementCounter) {
      _count--;
    }
    _output.add(_count);
    /*if(event is IncrementCounter){
        _output.add(++_counter);
      }else if(event is ClearCounter){
        _counter = 0;
        _output.add(0);
      }*/
  }
}
```

Resultados



Link: https://github.com/AlexReyes9725/GPL-3.2-Bloc.git

Nombre: Alex Reyes