بداية ننشئ ملفين

CounterState

بداخلة ننشئ abstract class مثلا باسم CounterState وبداخله نحدد الـ state التي سيتعامل معها التطبيق

part of 'cubits\_cubit.dart';

@immutable

abstract class CounterState {}

class CounterAIncrementState extends CounterState {}

class CounterBIncrementState extends CounterState {}

الملف الأخر نحدد به المتغيرات والدوال مثلا باسم CounterCubit

part 'cubits\_state.dart';

class CounterCubit extends Cubit<CounterState> {

CounterCubit() : super(CounterAIncrementState());

int teamAPoints = 0;

int teamBPoints = 0;

void TeamIncrement(String team, int buttonNumber) {

if (team == 'A') {

teamAPoints += buttonNumber;

emit(CounterAIncrementState());

} else {

teamBPoints += buttonNumber;

emit(CounterBIncrementState());

}

}

}

التعامل مع ui

نقوم بتعريف BlockProvider على مستوى التطبيق، ونحدد به Cubit

class pointsCounter extends StatelessWidget {

@override

Widget build(BuildContext context) {

return BlocProvider(

create: (context) => CounterCubit(),

child: MaterialApp(

debugShowCheckedModeBanner: false,

home: HomePage(),

),

);

}

}

في المكان الذي نريد إستخدام Cubit به نعمل wrap لـ BlocConsumer.

Weather App

weatherState

abstract class WeatherState {}

class WeatherInitial extends WeatherState {}

class WeatherLoading extends WeatherState {}

class WeatherSuccess extends WeatherState {}

class WeatherFailure extends WeatherState {}

WeatherCubit

class WeatherCubit extends Cubit<WeatherState> {

WeatherCubit(

this.weatherModel,

this.weatherService,

) : super(WeatherInitial());

WeatherService weatherService;

WeatherModel? weatherModel;

void getWeather({required String cityName}) async {

emit(WeatherLoading());

try {

weatherModel = await weatherService.getWeather(cityName: cityName);

emit(WeatherSuccess());

} on Exception catch (e) {

emit(WeatherFailure());

}

}

}

Ui

Search

onSubmitted: (data) async {

cityName = data;

BlocProvider.of<WeatherCubit>(context).getWeather(cityName: cityName!);

},