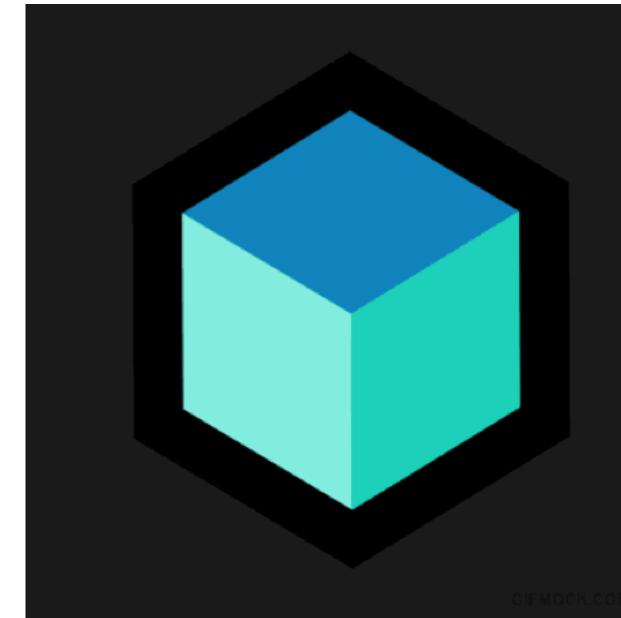


Cubit and Beyond



Still the new kid on the bloc



Felix Angelov @ Very Good Ventures

Very Good Ventures, Chicago



Screenshot of a GitHub profile page for Felix Angelov (@felangelov). The profile picture shows a smiling man with a beard. The bio reads:

Hi there! I'm a software engineer at Very Good Ventures in Chicago, IL. I'm currently working on the bloc library. I'm learning how to make latte art. I'm looking to collaborate on open source projects. Ask me about Flutter and Dart. Pronouns: he/him/his. Fun fact: I have a pet bunny named Coco.

The pinned repositories are:

- bloc: A predictable state management library that helps implement the BLoC design pattern. (Dart, 5.3k stars, 1.2k forks)
- bloc.js: A predictable state management library that helps implement the BLoC design pattern in JavaScript. (TypeScript, 67 stars, 8 forks)
- equatable: Simplify Equality Comparisons | A Dart abstract class that helps to implement equality without needing to explicitly override == and hashCode. (Dart, 377 stars, 37 forks)
- fresh: A token refresh library for Dart. (Dart, 92 stars, 7 forks)
- flutter: Forked from flutter/flutter. Flutter makes it easy and fast to build beautiful mobile apps. (Dart, 1 star)
- sealed_flutter_bloc: flutter_bloc state management extension that integrates sealed_unions. (Dart, 45 stars, 4 forks)

Follow, Sponsor, and other profile details are also visible.



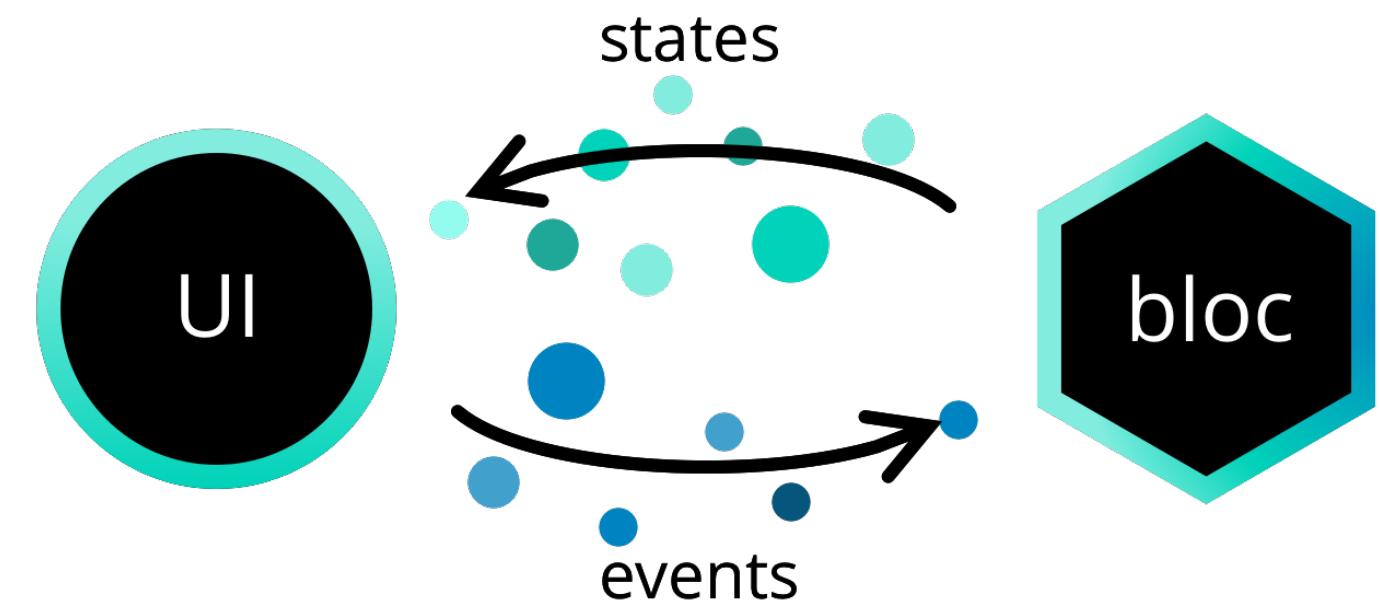
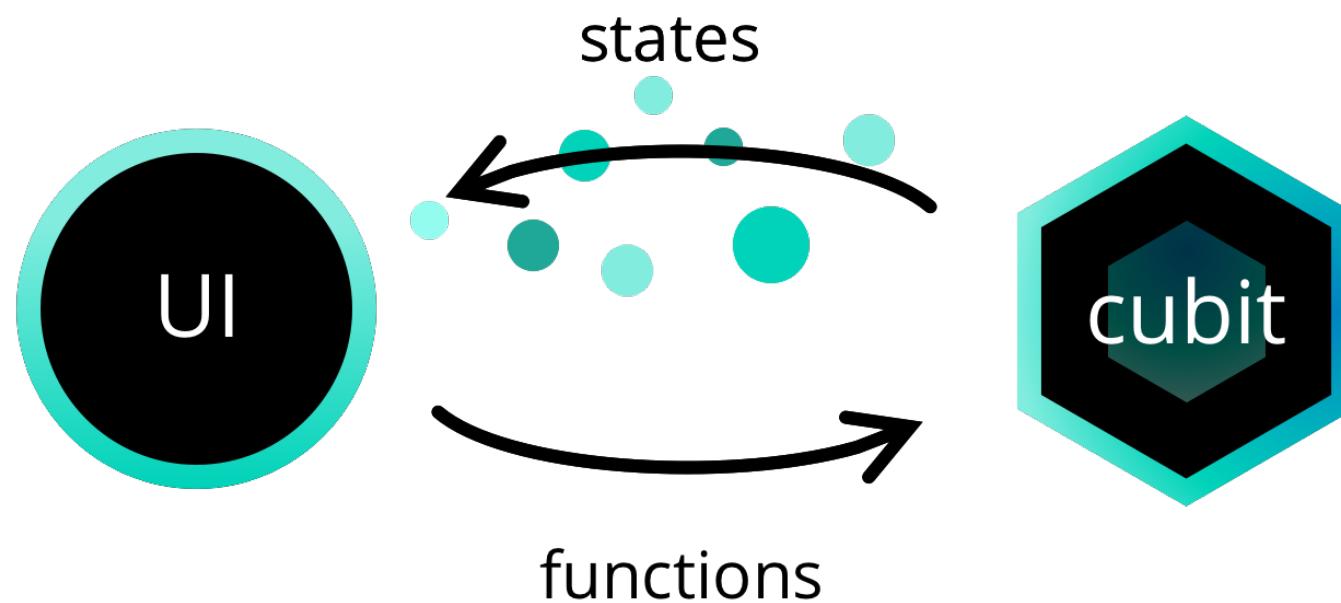
<https://verygood.ventures>

Agenda

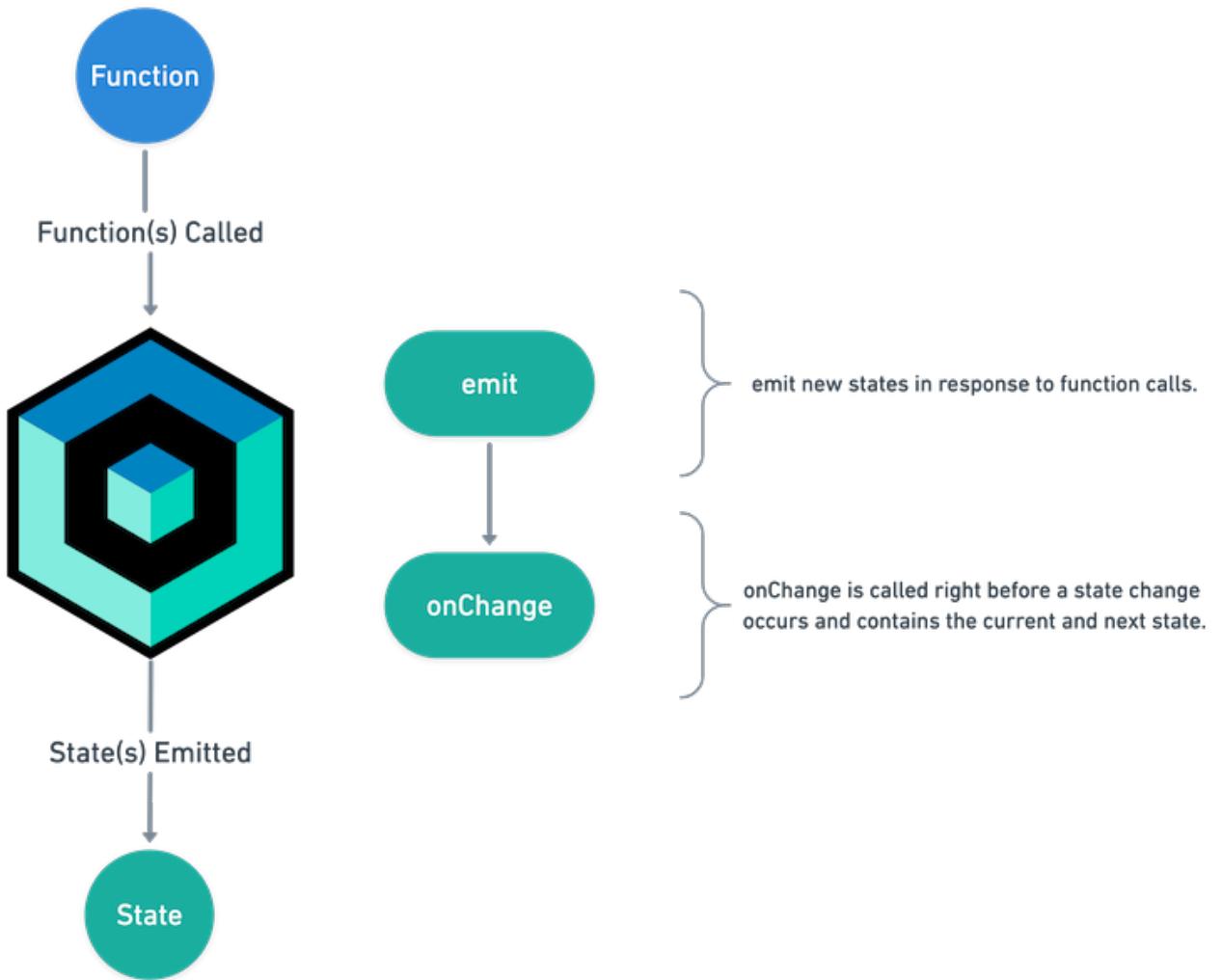
- Meet Cubit
- Cubit vs. Bloc
- What's new in v6.1.0
- Live Coding

Meet Cubit

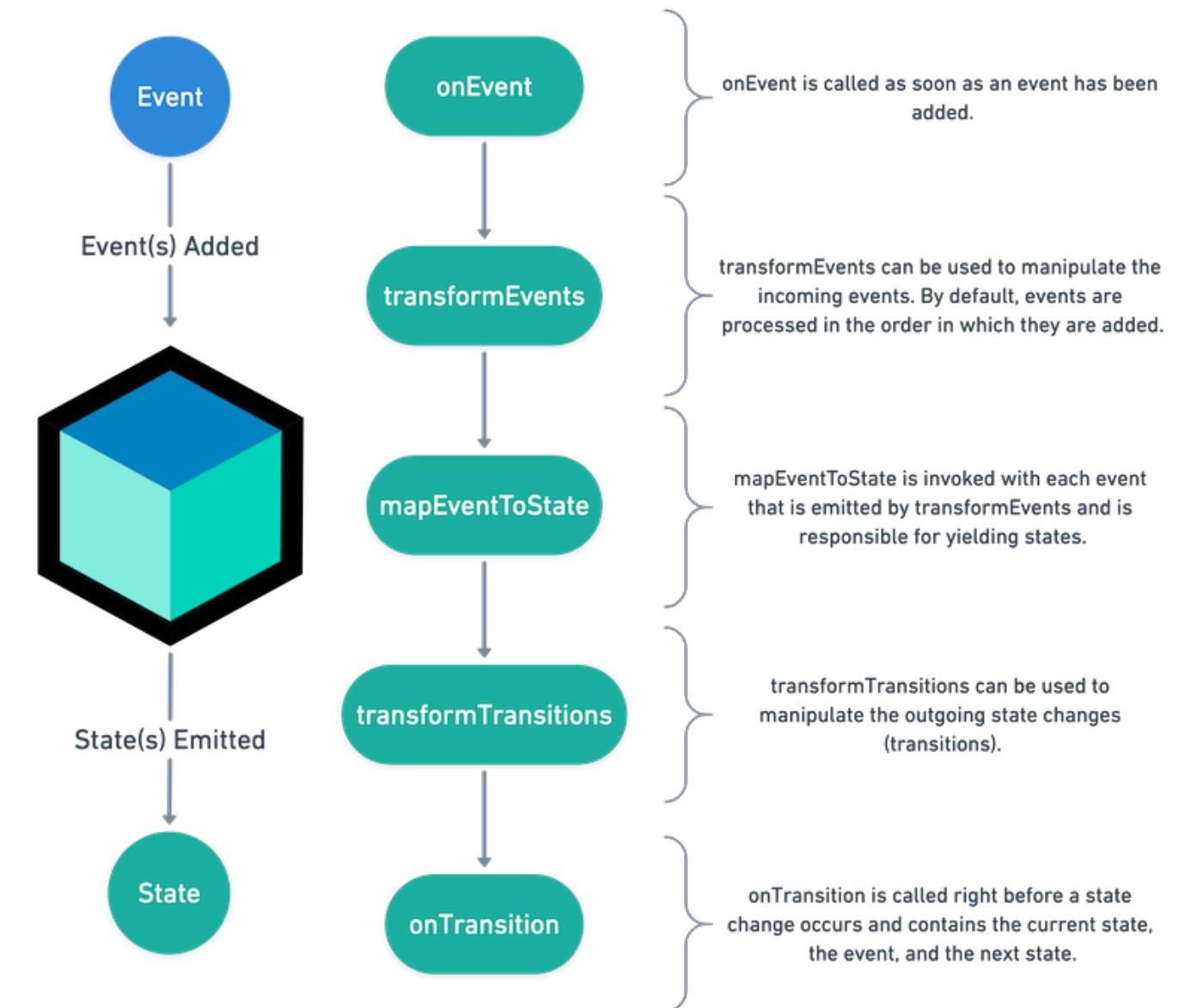
Cubit = Bloc - Events



Cubit



Bloc



CounterCubit

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

class CounterCubit extends Cubit<int> {
    CounterCubit() : super(0);

    void increment() => emit(state + 1);
}
```

Cubit vs. Bloc

Cubit

```
class CounterCubit extends Cubit<int> {  
    CounterCubit() : super(0);  
  
    void increment() => emit(state + 1);  
}
```

Bloc

```
enum CounterEvent { increment }  
  
class CounterBloc extends Bloc<CounterEvent, int> {  
    CounterBloc() : super(0);  
  
    @override  
    Stream<int> mapEventToState(CounterEvent event) async* {  
        switch (event) {  
            case CounterEvent.increment:  
                yield state + 1;  
                break;  
        }  
    }  
}
```

Cubit vs. Bloc

	Cubit	Bloc
Simple	✓	
Concise	✓	
Traceable		✓
ReactiveX Operators		✓
Testable	✓	✓
Scalable	✓	✓
Tooling Support	✓	✓

What's New in v6.1.0?



context.watch

```
@override  
Widget build(BuildContext context) {  
    final state = context.watch<MyBloc>().state;  
    return Text('$state');  
}
```

- Rebuilds when the state of MyBloc changes.
- Only allowed directly in build.
- Equivalent to top level BlocBuilder.

context.watch basics

```
// 😊  
override  
Widget build(BuildContext context) {  
  return BlocBuilder<MyBloc, MyState>(  
    builder: (context, state) => Text('$state'),  
  );  
}
```

```
// 😎  
override  
Widget build(BuildContext context) {  
  final state = context.watch<MyBloc>().state;  
  return Text('$state');  
}
```

context.watch gotchas

```
// 😬  
@override  
Widget build(BuildContext context) {  
    final state = context.watch<MyBloc>().state;  
    return Scaffold(  
        appBar: ...  
        body: Text('$state'),  
        floatingActionButton: ...  
    );  
}
```

```
// 😎  
@override  
Widget build(BuildContext context) {  
    return Scaffold(  
        appBar: ...  
        body: Builder(  
            builder: (context, state) {  
                final state = context.watch<MyBloc>().state;  
                return Text('$state');  
            },  
            floatingActionButton: ...  
        );  
    }  
}
```

context.watch recap

```
// BEFORE (bad) 😬  
@override  
Widget build(BuildContext context) {  
    final state = context.bloc<MyBloc>().state;  
    return Text('$state');  
}
```

```
// NOW (good) 😊  
@override  
Widget build(BuildContext context) {  
    return BlocBuilder<MyBloc, MyState>(  
        builder: (context, state) => Text('$state'),  
    );  
}
```

```
// NOW (good) 😎  
@override  
Widget build(BuildContext context) {  
    final state = context.watch<MyBloc>().state;  
    return Text('$state');  
}
```

 Closed malithkuruppu opened this issue on Sep 13 · 1 comment

mohammadne opened this issue on Jun 27 · 1 comment



malithkuruppu commented on Sep 13

Are there any plans to implement `MultiBlocBuilder` and `MultiBlocProvider`?

MultiBlocBuilder #42

 Closed

amreniouinnovent

BlocBuilder nest hell #714



amreniouinnovent c

Is your feature request related to a problem? I want better readability for

MultiBlocBuilder? #841

 Closed edsonbonfim opened this issue on Feb 2 · 6 comments



edsonbonfim commented on Feb 2

A case of use when this feature is helpful is when we have a widget that responds at two or more independent external events. In this case, the widget is wrapped with two or more bloc builder that managed these events. Nested blocs can't be a real problem, but it's expressive to write, like that as create another bloc which has dependencies on the two blocs.

A example where this feature can be helpful.

```
return BlocBuilder<BlocA, bool>(
  builder: (context, stateA) {
    return BlocBuilder<BlocB, int>(
      builder: (context, stateB) {
        if (stateA || stateB == 1) {
          // do something
        }
        // do another thing
      });
}, @felangelov - Flutter Vikings 2020
);
```

Some help?

[Feature Request] MultiBlocBuilder #538

 Closed

ine commented on Jun 27

`MultiBlocBuilder` like `MultiBlocProvider`.



ffpetrovic opened this issue on Sep 25, 2019 · 67 comments

ffpetrovic commented on Dec 9, 2019 · edited

Is your feature request related to a problem? bloclibrary.dev shows that we are supposed to use `MultiBlocProvider`. Consider blocs `Bloc1` `Bloc2` `Bloc3` ...
pattern is to have a ...`Loading` ...`Loaded` ...
want to check whether ANY of these blocs are in ...
even 4) for every Bloc (because, honestly, I feel like
use case for me), would be really appreciated. Sorry if ...
Describe alternatives you've considered
Nesting BlocBuilders in the root of the build method.



13



4



1

Nested bloc builders in flutter

 Closed

francipvb opened this issue on Aug 20 · 2 comments



francipvb commented on Aug 20

...

Hello,

Can I construct a nested bloc builder widget tree? What I have to be concerned about?

Furter explanation

I have a list loaded in a bloc state, but I want to track the changes for the items individually and avoid any unnecessary rebuild of the parent bloc builder. I want it to be rebuilt only if the length of the list changes. At the deeper level, I want to rebuild the list item when the individual item changes its value.

Is it possible to do without major issues?

Thanks,

MultiBlocBuilder and MultiBlocConsumer #1724

Closed malithkuruppu opened this issue on Sep 13 · 1 comment



malithkuruppu commented on Sep 13

Are there any plans to implement MultiBlocBuilder and MultiBl...

MultiBlocBuilder #422

Closed

amreniouinnov...



amreniouinnovent c

Is your feature request i
I want better readability for

MultiBlocBuilder? #841

Closed edsonbonfim opened this issue on Feb 2 · 6 comments



edsonbonfim commented on Feb 2

A case of use when this feature is helpful is when we have a widget that responds at two or more independent external events. In this case, the widget is wrapped with two or more bloc builder that managed these events. Nested blocs can't be a really problem, but is expressive to write, like that as create another bloc which has dependencies on the two blocs.

A example where this feature can be helpful.

```
return BlocBuilder<BlocA, bool>(  
  builder: (context, stateA) {  
    return BlocBuilder<BlocB, int>(  
      builder: (context, stateB) {  
        if (stateA || stateB == 1) {  
          // do something  
        }  
        // do another thing  
      }  
    );  
  },  
);
```

Adding multiBlocBuilder interface #1347

Closed mohammadne opened this issue on Jun 27 · 1 comment

...

[Feature Request] MultiBlocBuilder #538

Closed

pczn0327 commented on Jun 27

MultiBlocBuilder like MultiBlocProvider .



pczn0327 commented on Sep 25, 2019 · 67 comments

pczn0327 commented on Sep 25, 2019

Sometimes I would need to access different blocs in the same widget, and the code goes pretty messy. I am thinking maybe we can have something like MultiBlocBuilder which serve the similar purpose of MultiRepositoryProvider and



ffpetrovic

commented on Sep 9, 2019 · edited

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

MultiBlocBuilder 🎉 (sort of)

```
Builder(  
  builder: (context) {  
    final stateA = context.watch<BlocA>().state;  
    final stateB = context.watch<BlocB>().state;  
    final stateC = context.watch<BlocC>().state;  
  
    // return a Widget which depends on the state of BlocA, BlocB, and BlocC  
  }  
);
```

context.read

```
@override  
Widget build(BuildContext context) {  
  return RaisedButton(  
    onPressed: () {  
      context.read<MyBloc>().add(MyEvent());  
    },  
    ...  
  )  
}
```

- Does not rebuild when the state of MyBloc changes.
- Not allowed directly in build.

context.read recap

```
// BEFORE (bad) 😬  
@override  
Widget build(BuildContext context) {  
    final bloc = context.bloc<MyBloc>();  
    return RaisedButton(  
        onPressed: () {  
            bloc.add(MyEvent());  
        },  
        ...  
    )  
}
```

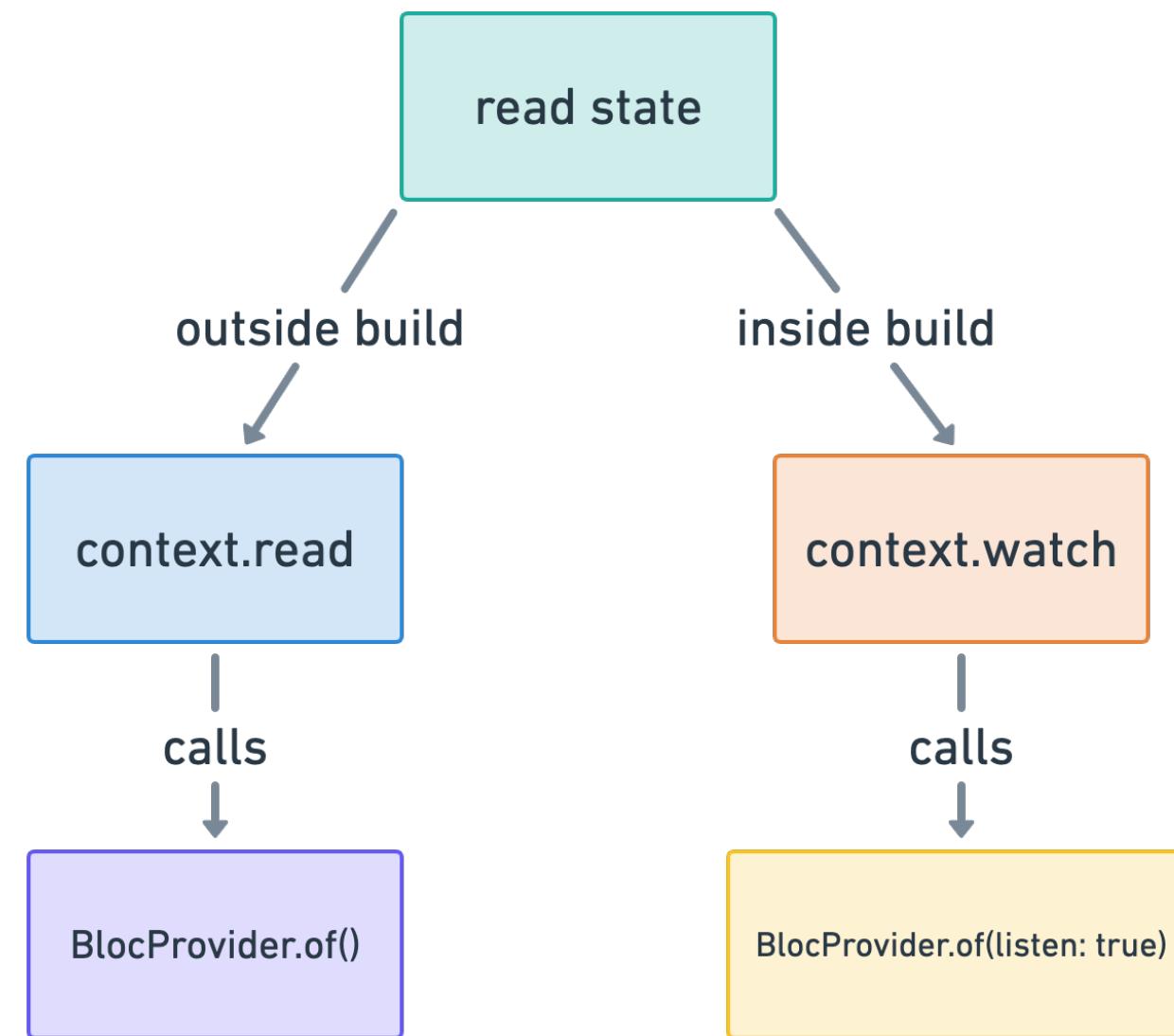
```
// NOW 😎  
@override  
Widget build(BuildContext context) {  
    return RaisedButton(  
        onPressed: () {  
            context.read<MyBloc>().add(MyEvent());  
        },  
        ...  
    )  
}
```

context.select

```
@override  
Widget build(BuildContext context) {  
  final name = context.select(  
    (UserBloc bloc) => bloc.state.user.name,  
  );  
  return Text('Hello $name!');  
}
```

- Rebuild in response to part of a bloc's state.

Decision Tree



Bloc Test Scenario

```
blocTest<CounterBloc, int>(  
  'emits 100 when increment is called and state is 99',  
  build: () => CounterBloc(),  
  act: (bloc) {  
    bloc  
      ..add(CounterEvent.increment)  
      ..add(CounterEvent.increment)  
      ..add(CounterEvent.increment)  
      ..add(CounterEvent.increment)  
      ..add(CounterEvent.increment)  
    ...  
  },  
  expect: const <int>[1, 2, 3, ..., 99, 100],  
);
```

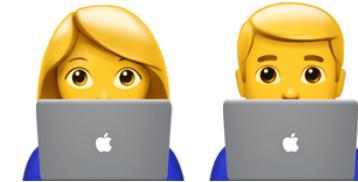
Bloc Test Seed State

```
blocTest<CounterBloc, int>(  
  'emits 100 when increment is called and state is 99',  
  build: () => CounterBloc(),  
  seed: 99,  
  act: (bloc) => bloc.add(CounterEvent.increment),  
  expect: const <int>[100],  
);
```

Complete migration guide

<https://bloclibrary.dev/#/migration>

Live Coding



https://github.com/felangel/cubit_and_beyond/tree/main/example

Thanks! 🙏

Twitter @felangelov

Github @felangel