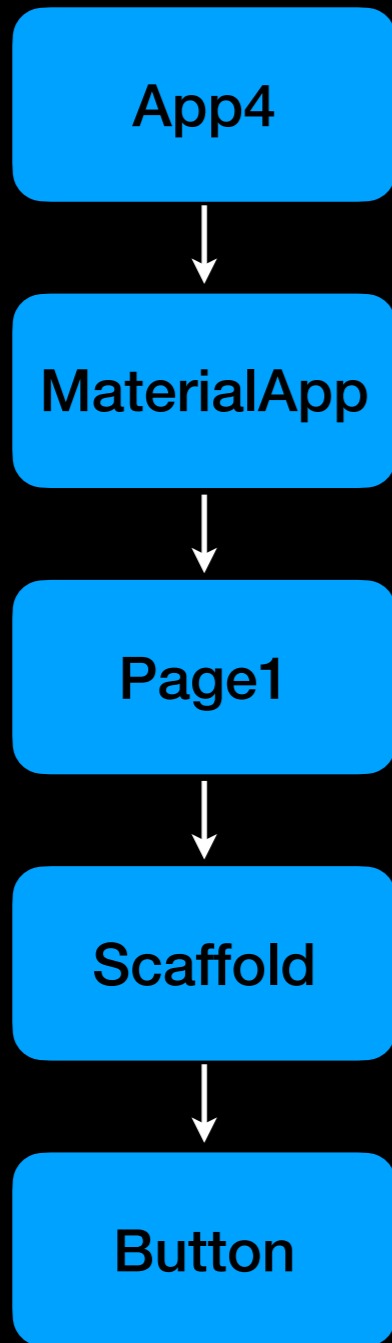
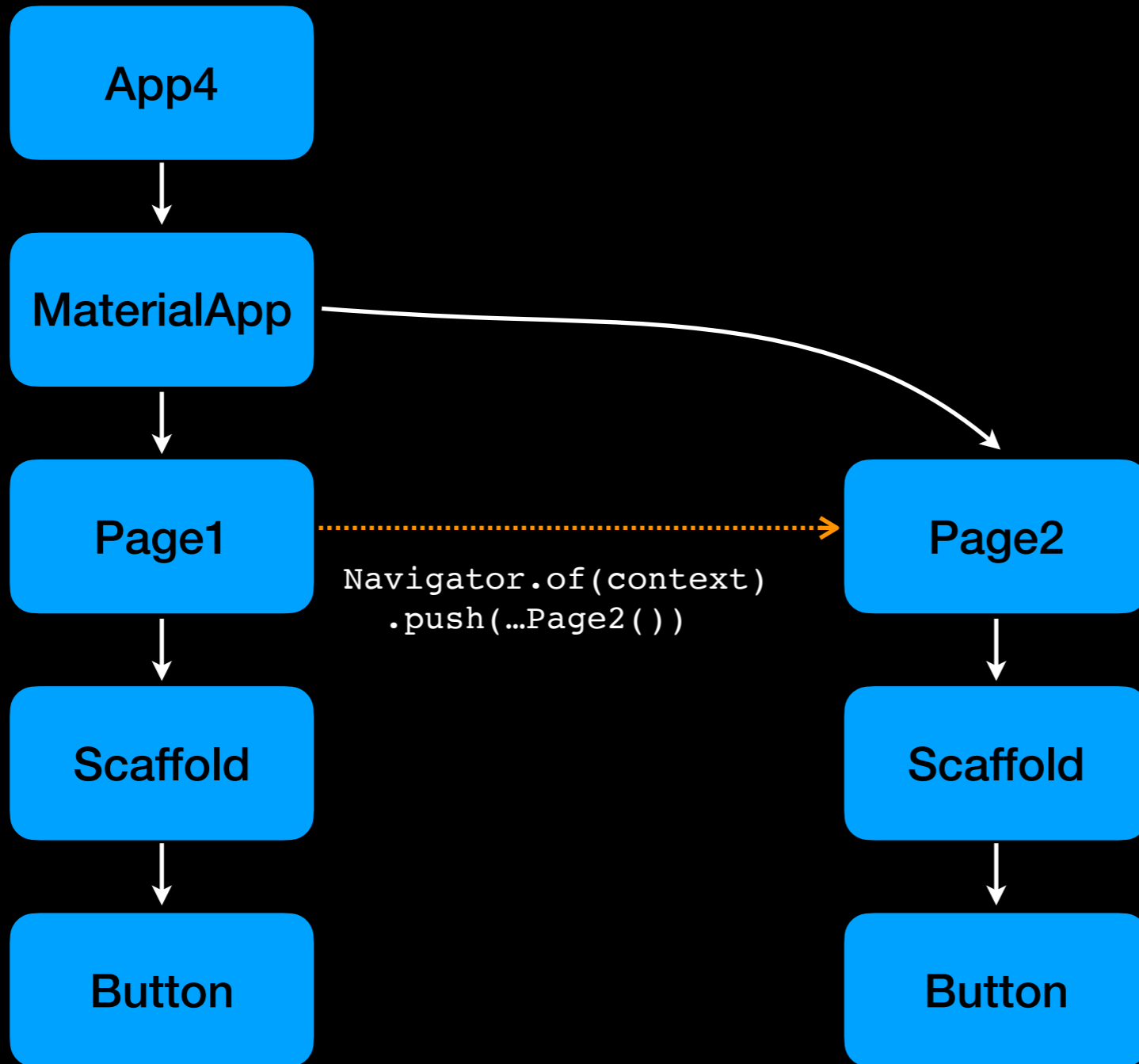


**03\_flutter\_nav\_route\_eg1**

# Widget Tree



Widget Tree



Widget Tree

App4



MaterialApp



Page1



Scaffold



Button

Page2



Scaffold

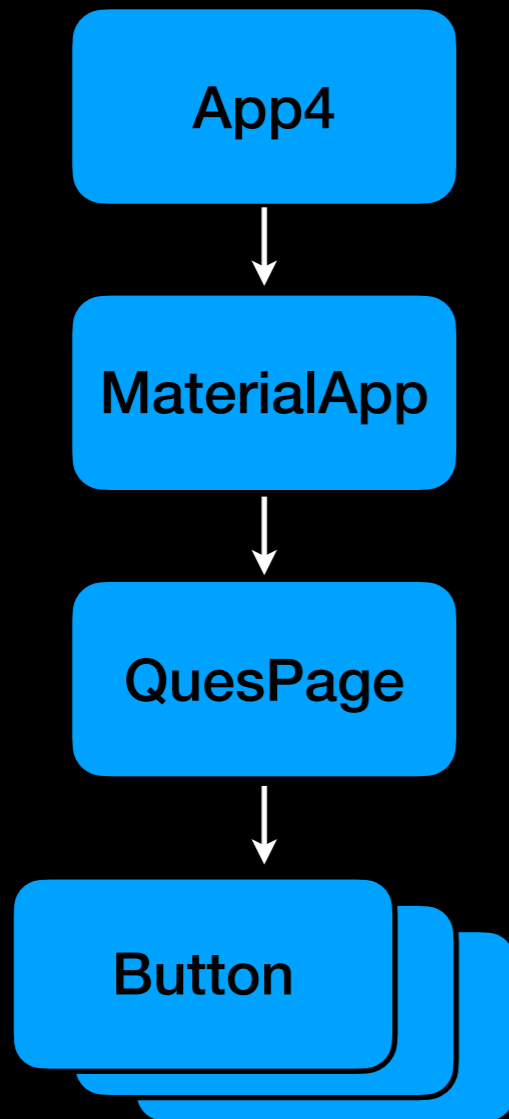


Button

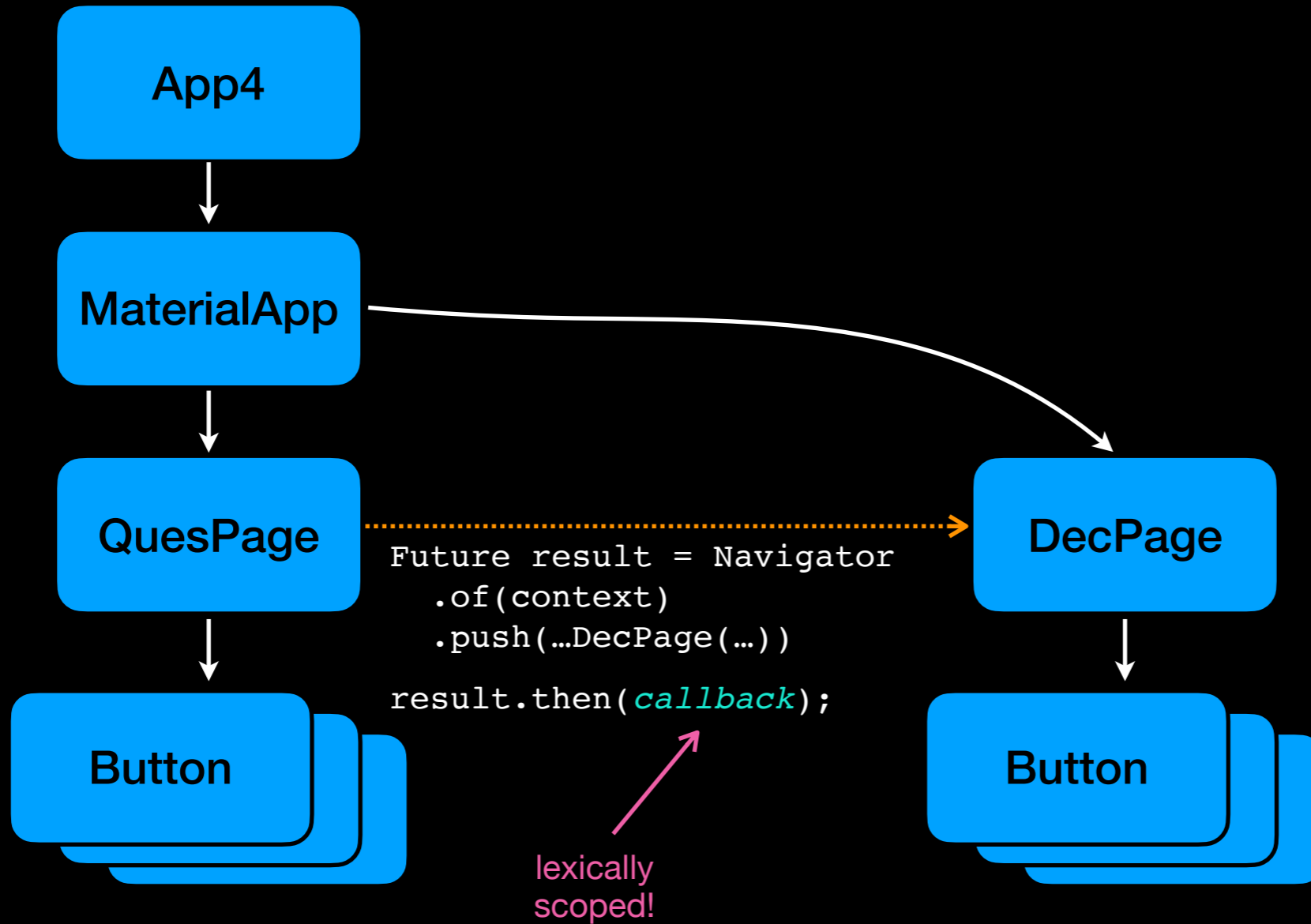
`Navigator.of(context)  
.pop()`

**03\_flutter\_nav\_route\_eg2**

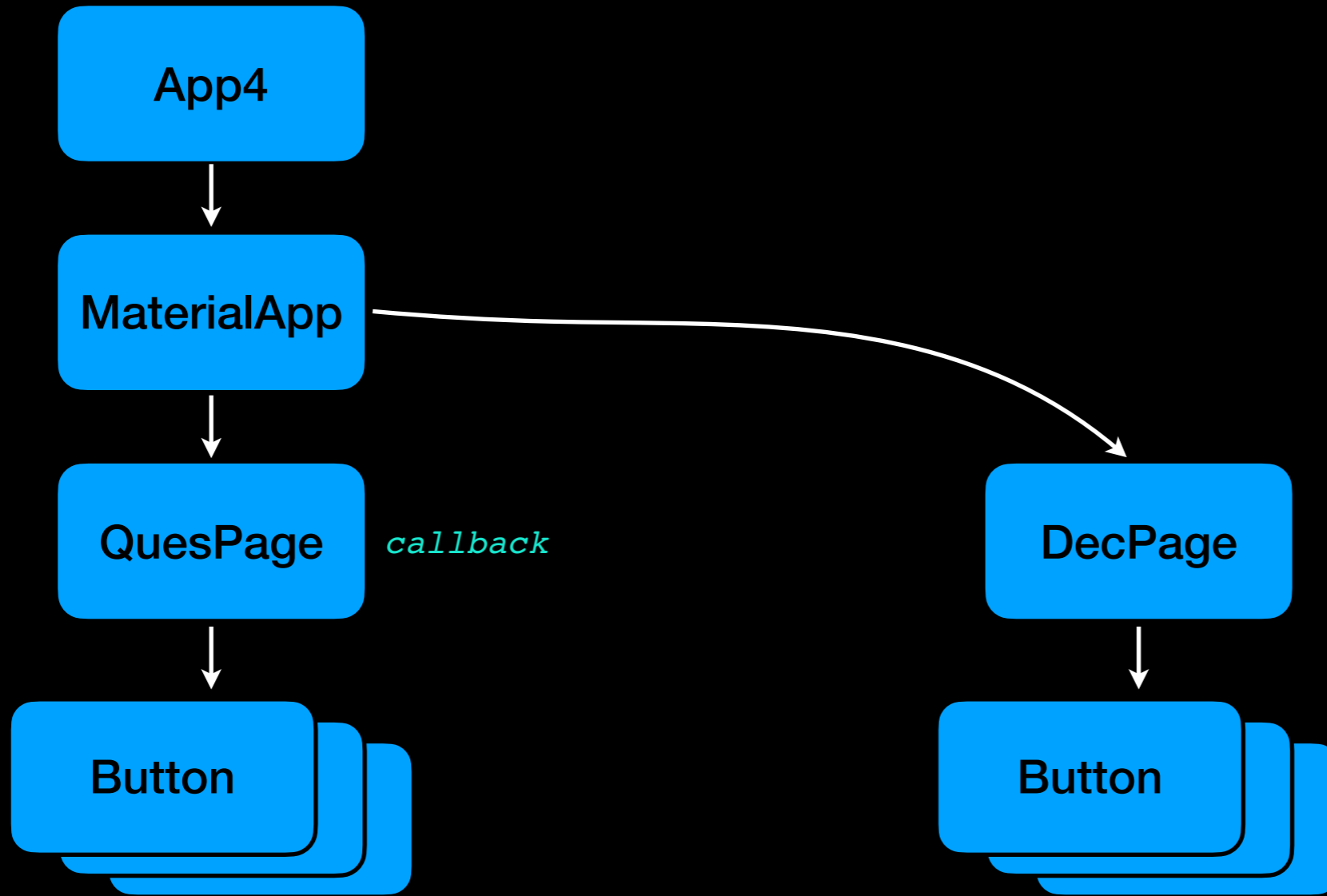
# Widget Tree



Widget Tree

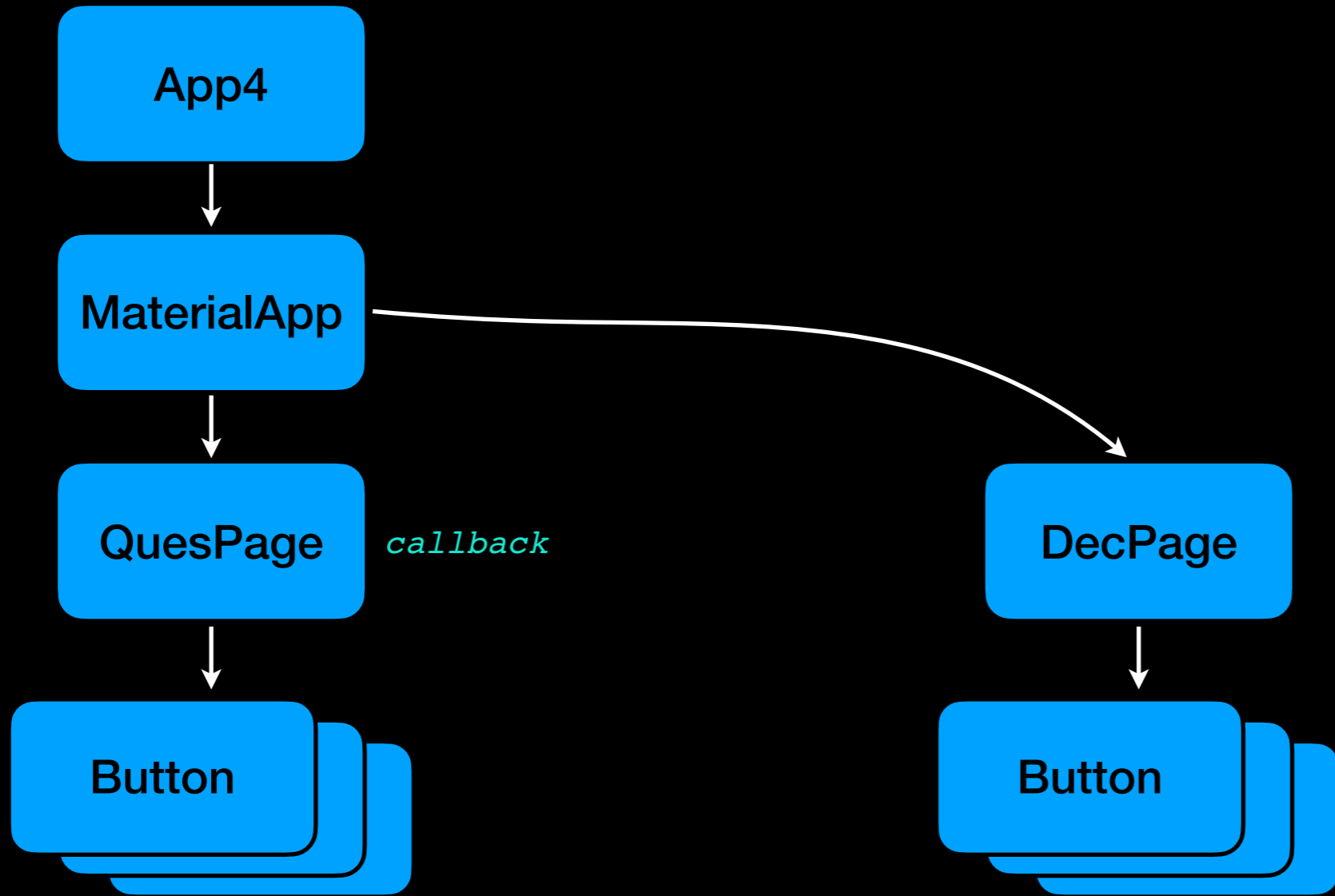


Widget Tree



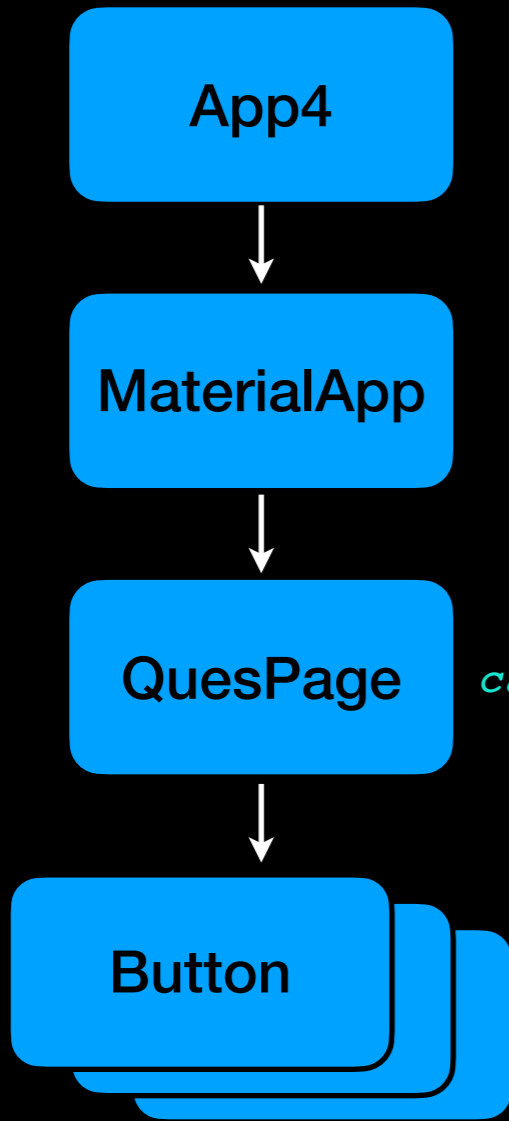


# Widget Tree

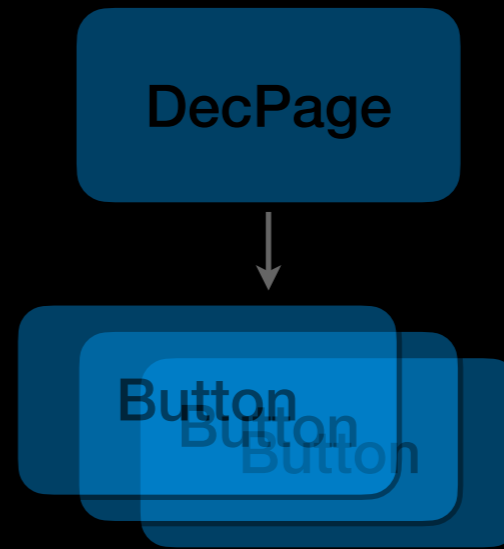


```
Navigator.of(context)  
.pop(answer)
```

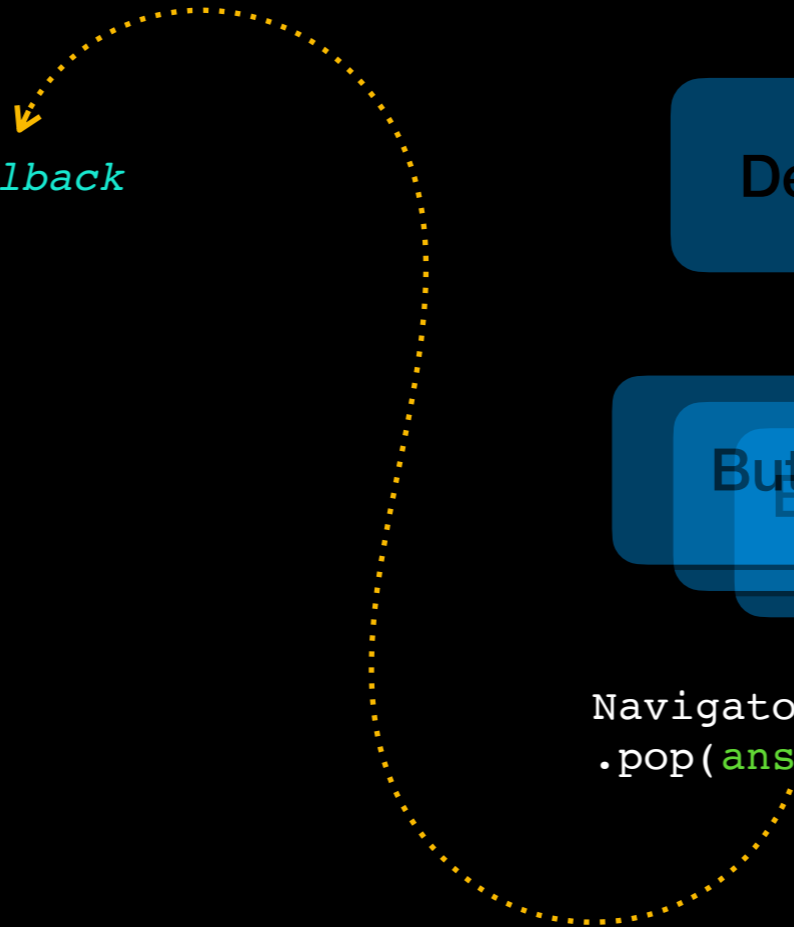
Widget Tree



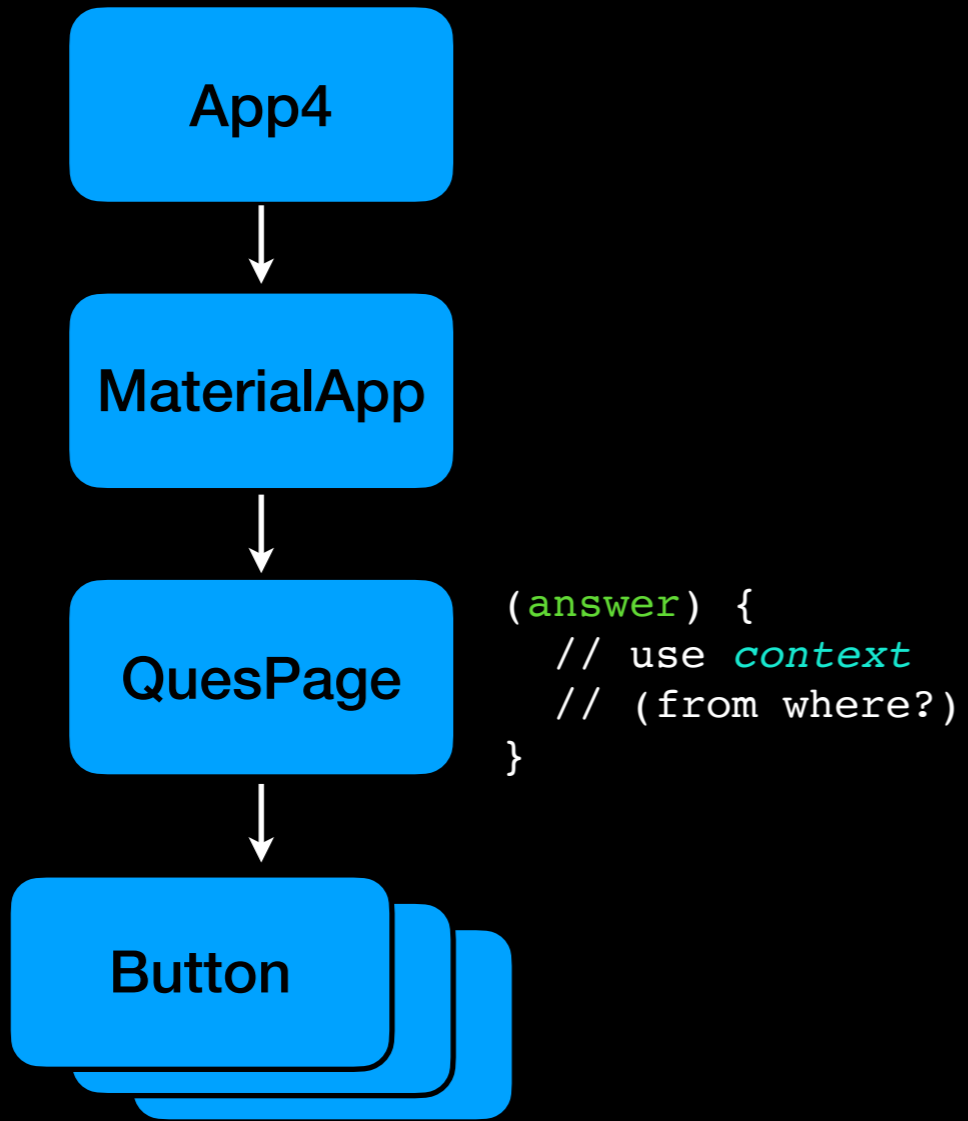
*callback*



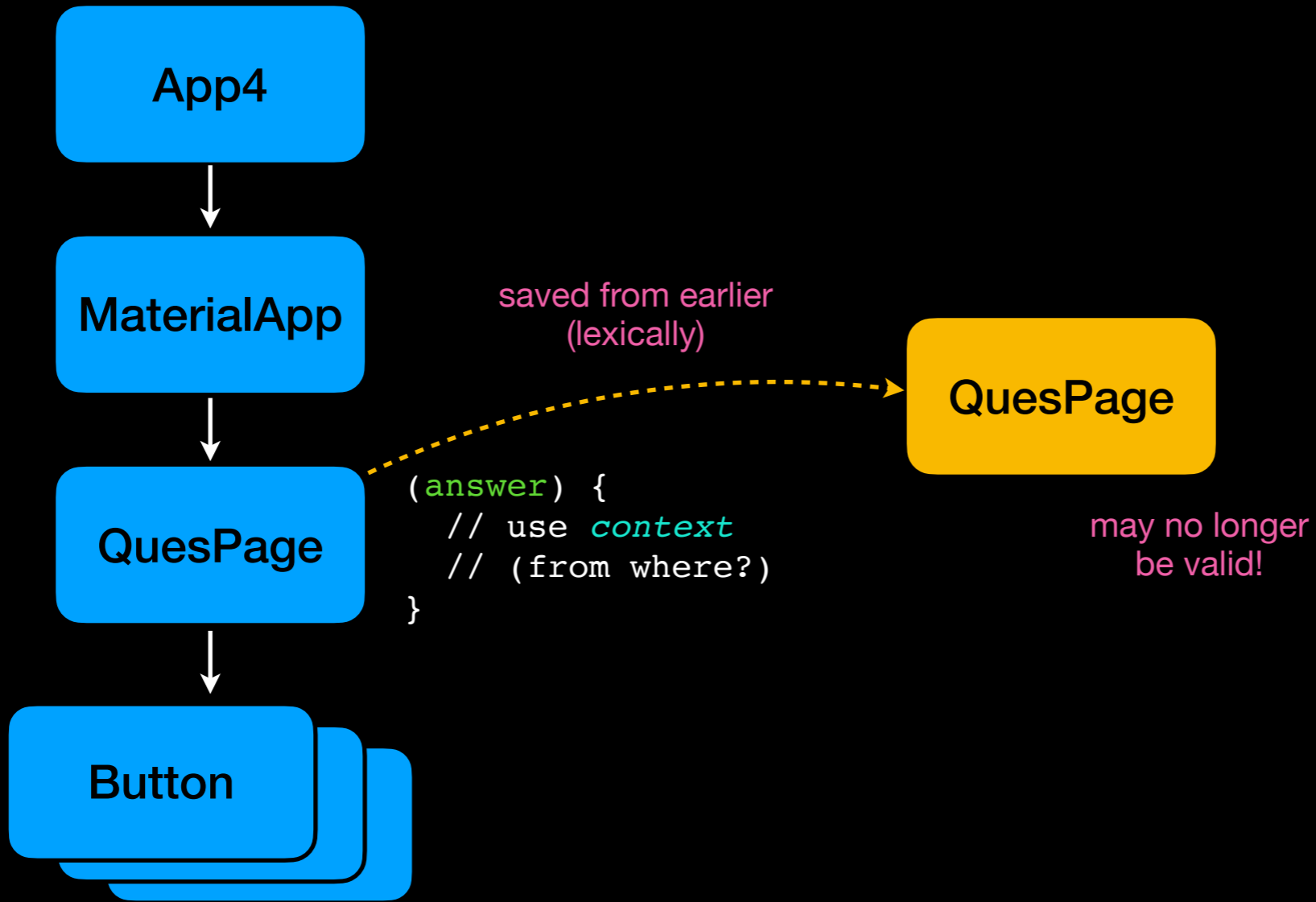
```
Navigator.of(context)  
.pop(answer)
```



# Widget Tree



Widget Tree



“Async Gap”

**futures (aka promises)  
& async/await**

```
abstract class Future<T> {  
    Future<R> then<R>(R Function (T));  
    Future<T> catchError(Function onError);  
}
```

```
abstract class FullOfPromises {  
    Future<String> longOperation(String input);  
}  
  
void consumer(FullOfPromises fop) {  
    Future<String> future = fop.longOperation('input');  
}
```

```
abstract class FullOfPromises {
    Future<String> longOperation(String input);
}

void consumer(FullOfPromises fop) {
    Future<String> future = fop.longOperation('input');
    future.then((result) {
        print('Got result "$result"');
    });
}
```



```
abstract class FullOfPromises {
    Future<String> longOperation(String input);
}

void consumer(FullOfPromises fop) {
    Future<String> future = fop.longOperation('input');
    future.then((result) {
        print('Got result "$result"');
    });
}

void main() {
    consumer(...);
    print('After consumer call');
}
```

```
abstract class FullOfPromises {  
    Future<String> longOperation(String input);  
}
```

```
void consumer(FullOfPromises fop) {  
    Future<String> future = fop.longOperation('input');  
    future.then((result) {  
        print('Got result "$result"');  
    });  
}
```


```
• void main() {  
    consumer(...);  
    > print('After consumer call');  
}
```

The diagram consists of yellow dotted lines. A line starts at the call to `consumer(...)` in the `main` function, loops around to the left, and ends with a dot at the start of the `consumer` function. Another line starts at the closing brace of the `consumer` function, loops around to the left, and ends with an arrowhead pointing to the `print` statement in the `main` function.

```
abstract class FullOfPromises {  
    Future<String> longOperation(String input);  
}
```

```
void consumer(FullOfPromises fop) {  
    Future<String> future = fop.longOperation('input');  
    future.then((result) {  
        print('Got result "$result"'); ←..... called later!  
    });  
}
```

```
• void main() {  
    consumer(...);  
    print('After consumer call');  
}
```

A diagram consisting of a large dotted line that starts from the left side of the code, loops around the 'consumer' method call in 'main', and then points to the 'print' statement in 'main'. A small black dot is placed to the left of the 'void main()' line, with a dotted arrow pointing from it to the 'consumer(...)' call. Another dotted arrow points from the 'print' statement in 'main' back to the 'print' statement in the 'consumer' method, indicating the return of control.


```
abstract class FullOfPromises {
    Future<String> longOperation(String input);
}

void consumer(FullOfPromises fop) async {
    var result = await fop.longOperation('input');

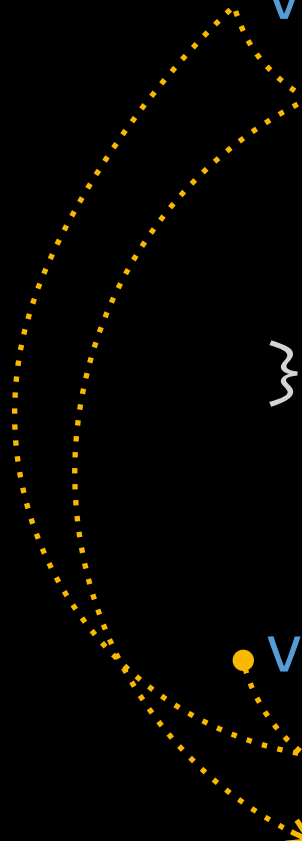
    print('Got result "$result"');
}

void main() {
    consumer(...);
    print('After consumer call');
}
```

```
abstract class FullOfPromises {  
    Future<String> longOperation(String input);  
}
```

```
void consumer(FullOfPromises fop) async {  
    var result = await fop.longOperation('input');  
  
    print('Got result "$result"');  ..... called later!  
}
```

```
• void main() {  
    consumer(...);  
    print('After consumer call');  
}
```



```
void consumer(FullOfPromises fop) {
  fop.longOperation('input')
    .then((result) {
      fop.nextLongOperation(result)
        .then((result2) {
          print('Got result2 "$result2"');
        })
      .catchError((err) {
        print('Got error: "$err"');
      });
    })
    .catchError((err) {
      print('Got error: "$err"');
    });
}
```

```
void consumer(FullOfPromises fop) async {
  try {
    var result = await fop.longOperation('input');
    print('Got result "$result"');

    var result2 = await fop.nextLongOperation(result);
    print('Got result2 "$result2"');
  } catch (err) {
    print('Got error: "$err"');
  }
}
```

**creating futures**



```
void longComputation(void Function(String) callback) {  
    Timer(const Duration(seconds: 1), () {  
        callback('result');  
    });  
}
```

```
Future<String> longComputation2() {  
    final completer = Completer<String>();  
    Timer(const Duration(seconds: 1), () {  
        completer.complete('result');  
    });  
    return completer.future;  
}
```

```
abstract class Future<T> {  
    factory Future.delayed(Duration duration,  
                           T Function ());  
    factory Future.value(T value);  
}
```

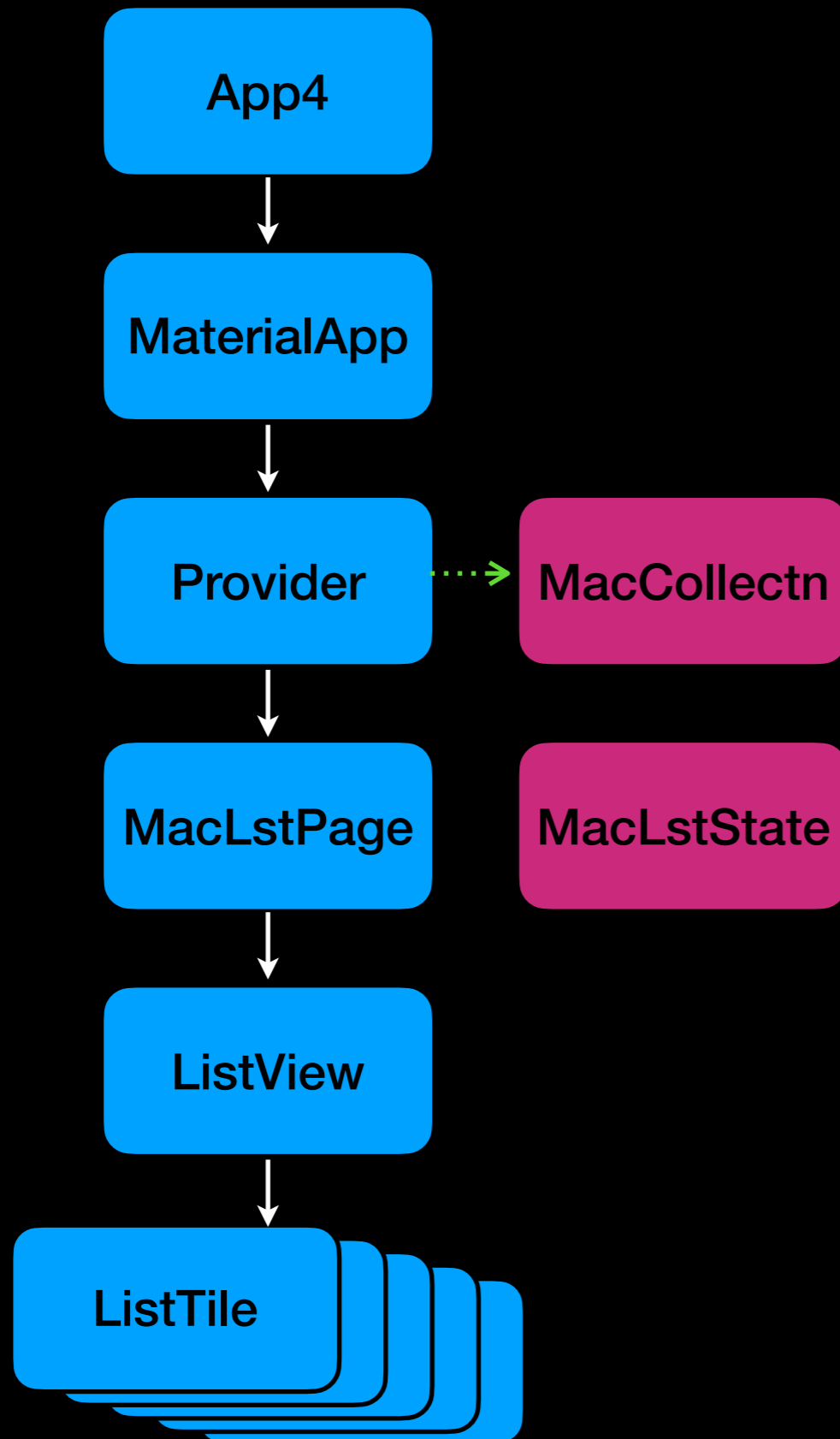
```
Future<String> longComputation3() {  
    return Future.delayed(const Duration(seconds: 1),  
                          () => 'result');  
}
```

```
Future<String> longComputation4() async {  
    await Future.delayed(const Duration(seconds: 1));  
    return 'result';  
}
```

```
Future<String> shortComputation() {  
    return Future.value('Hello');  
}
```

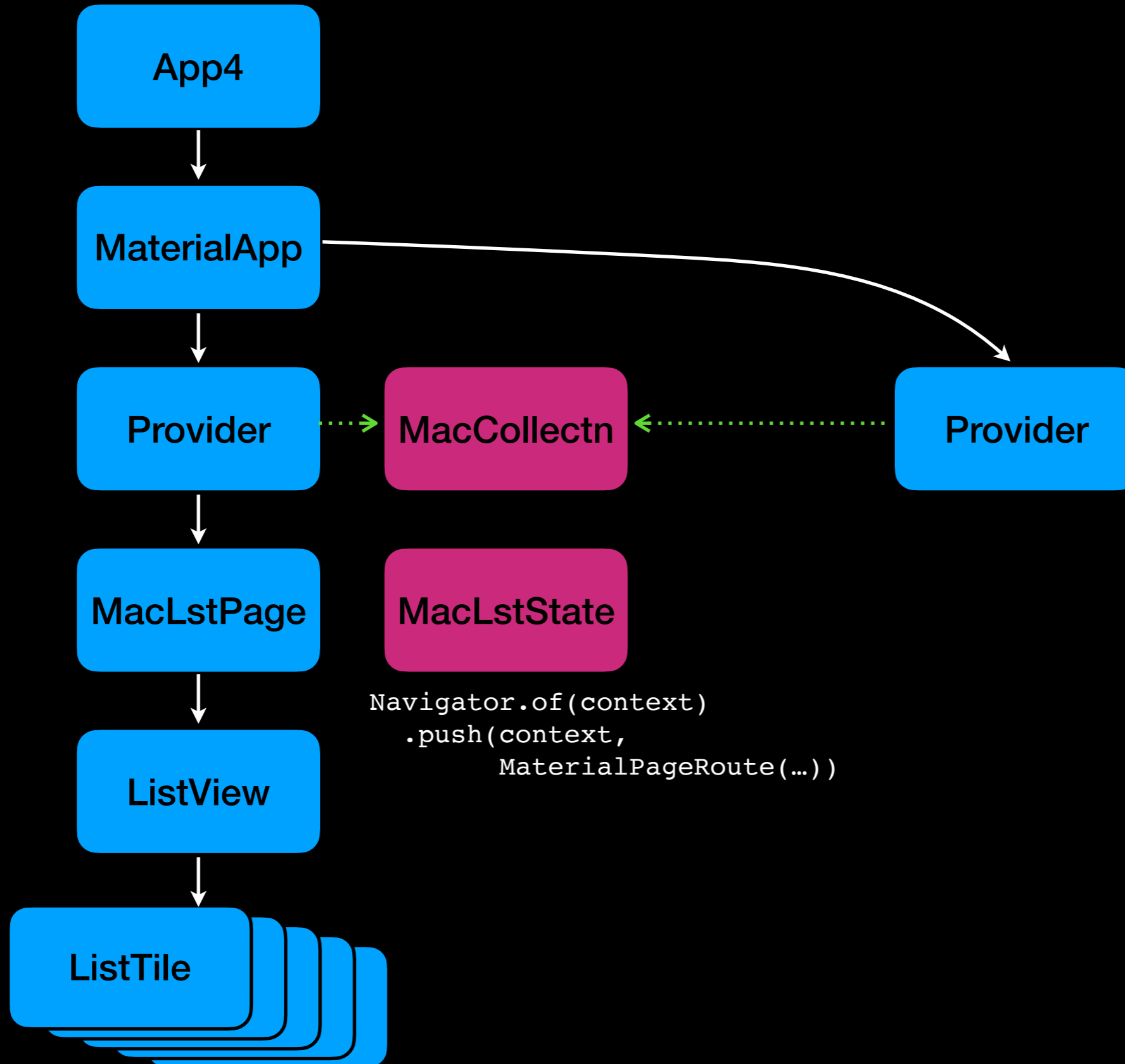
**03\_flutter\_nav\_route\_eg4**

Widget Tree





Widget Tree



Widget Tree

