

Duhok Polytechnic University

Technical College of Zakho

Department of CIS



LAB 4:

Ephemeral State

MOBILE APPLICATION II

Objectives

The Concept of State in Flutter (Static vs Dynamic UI)

The Ephemeral State Dilemma (Why StatelessWidget doesn't redraw on variable change)

Scoped Rebuilding with StatefulWidget

Local setState within Builders and Modals

Interactive Counter & Toggle Architectures

What is State & The Stateless Limitation

In Labs 1 to 3, our UIs responded to touch events by printing messages or pushing routes, but the screen itself remained visually fixed. In Flutter, **State** is any data that changes during the runtime of an app and directly affects what is rendered on screen.

The Problem:

- A StatelessWidget executes its build(BuildContext context) method only once (or when its parent rebuilds it).
- Declaring a normal variable like `int counter = 0;` inside a StatelessWidget and incrementing it inside an onPressed callback will change the variable in memory, but Flutter will never re-execute build(). The screen remains unchanged.

What is StatefulWidget?

- Instead of converting an entire screen into a heavy StatefulWidget, Flutter provides a specialized utility widget called StatefulWidget.
- It introduces a localized mutable state boundary anywhere inside a StatelessWidget.
- It provides a dedicated setState() callback that only rebuilds the subtree inside its builder closure, keeping the surrounding widget tree untouched and efficient.

Anatomy of StatefulWidget

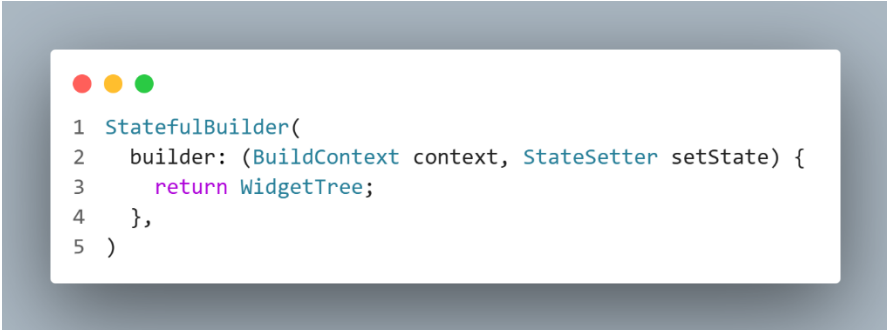
The widget takes a single required parameter: builder.

Key Parameters & Concepts:

- BuildContext context:

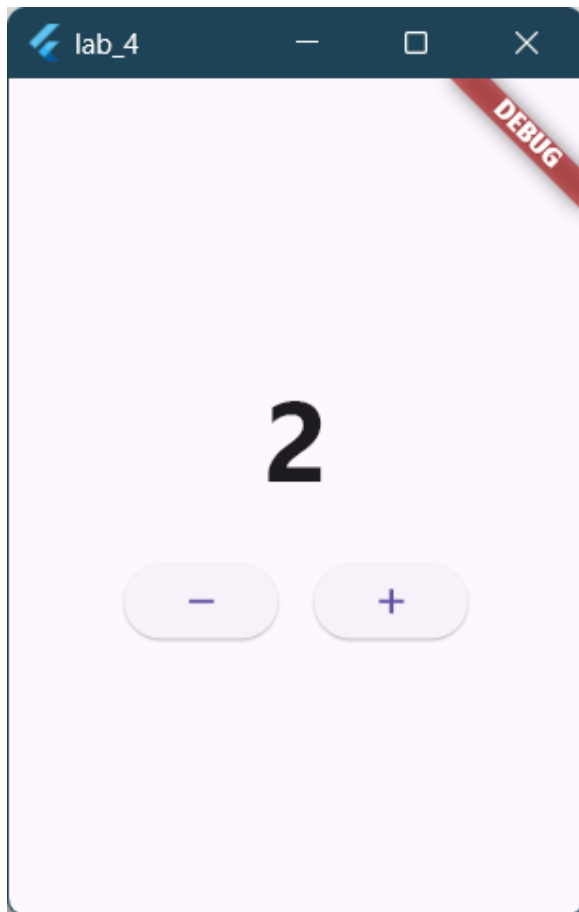
The location anchor of this specific sub-tree in the widget hierarchy.

- StateSetter setState: A custom updater function provided specifically to this builder.
- The Magic of setState((){ ... }): Calling this closure schedules a visual rebuild of only the widgets returned by this builder.
- State Storage: The mutable variable (e.g., int count = 0; or bool isLiked = false;) is declared outside the builder callback so that its value persists across rebuilds.



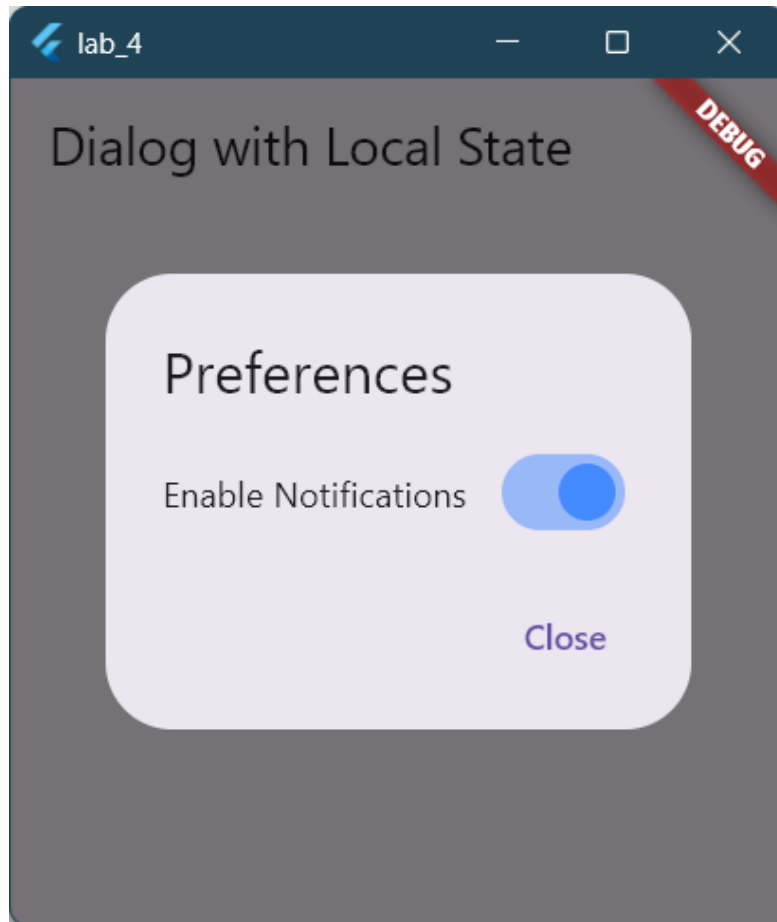
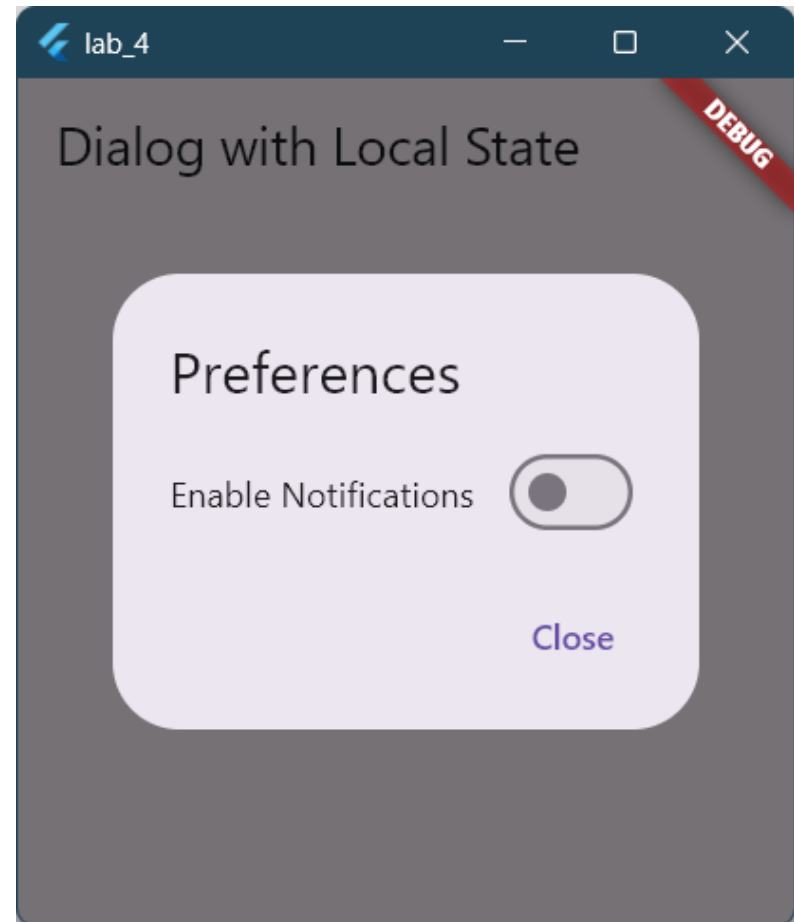
```
1 StatefulWidget(  
2   builder: (BuildContext context, StateSetter setState) {  
3     return WidgetTree;  
4   },  
5 )
```

Example 1



```
1 Widget build(BuildContext context) {
2   int counter = 0;
3
4   return Scaffold(
5     body: Center(
6       child: StatefulBuilder(
7         builder: (BuildContext context, StateSetter setState) {
8           return Column(
9             mainAxisAlignment: MainAxisAlignment.min,
10            children: [
11              Text(
12                '$counter',
13                style: TextStyle(fontSize: 48, fontWeight: FontWeight.bold),
14              ),
15              SizedBox(height: 20),
16              Row(
17                mainAxisAlignment: MainAxisAlignment.center,
18                children: [
19                  ElevatedButton(
20                    child: Icon(Icons.remove),
21                    onPressed: () => setState(() {
22                      counter--;
23                    }),
24                  ),
25                  SizedBox(width: 15),
26                  ElevatedButton(
27                    child: Icon(Icons.add),
28                    onPressed: () => setState(() {
29                      counter++;
30                    }),
31                  ),
32                ],
33              ),
34            ],
35          );
36        },
37      ),
38    ),
39  );
40 }
```

Example 2 Preview



Example 2 Code

```
1 class DialogTogglePage extends StatelessWidget {
2   @override
3   Widget build(BuildContext context) {
4     return Scaffold(
5       appBar: AppBar(title: Text('Dialog with Local State')),
6       body: Center(
7         child: ElevatedButton(
8           child: Text('Adjust Settings'),
9           onPressed: () => _openSettingsDialog(context),
10        ),
11      ),
12    );
13  }
14
15  void _openSettingsDialog(BuildContext context) {
16    bool notificationsEnabled = true;
17
18    showDialog(
19      context: context,
20      builder: (BuildContext dialogContext) {
21        return StatefulBuilder(
22          builder: (BuildContext context, StateSetter setModalState) {
23            return AlertDialog(
24              title: Text('Preferences'),
```

```
25      content: Row(
26        mainAxisAlignment: MainAxisAlignment.spaceBetween,
27        children: [
28          Text('Enable Notifications'),
29          Switch(
30            value: notificationsEnabled,
31            activeThumbColor: Colors.blueAccent,
32            onChanged: (bool newValue) {
33              setModalState(() {
34                notificationsEnabled = newValue;
35              });
36            },
37          ),
38        ],
39      ),
40      actions: [
41        TextButton(
42          onPressed: () => Navigator.pop(context),
43          child: Text('Close'),
44        ),
45      ],
46    );
47  },
48 );
49 },
50 );
51 }
52 }
```

Any Questions?



Lab Work

