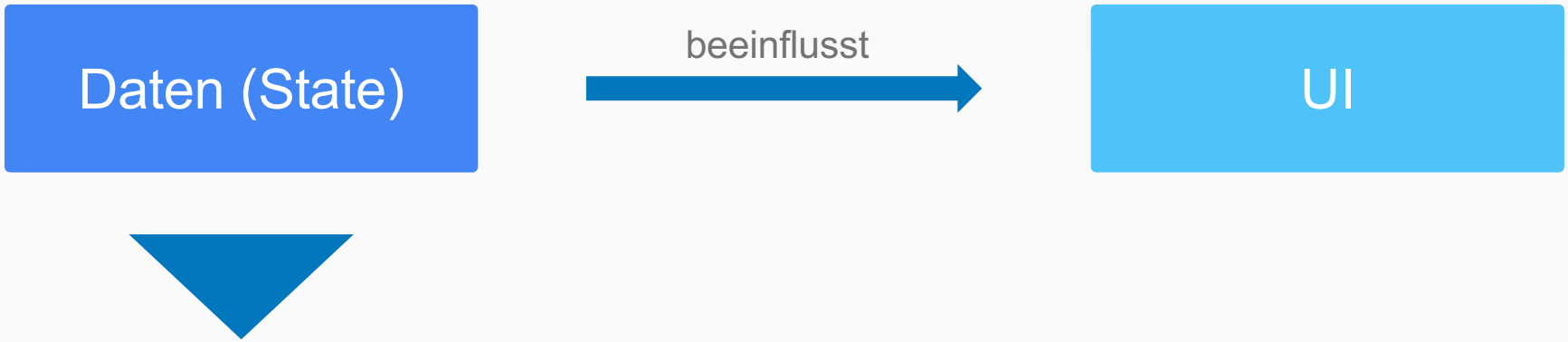
Mobile Apps Workshop

Flutter: State Management

State



Betrifft gesamte App oder grosse Teile davon (z.B. “Ist der User eingeloggt?”)

Betrifft nur ein Widget (z.B. “Soll eine Ladeanimation angezeigt werden?”)

Stateless Widgets

- Benötigen keinen State, sind also unveränderlich

  → werden während der Laufzeit nicht neu gerendert, ***build()*** Methode wird nur **einmal** ausgeführt

- Werte werden über Constructor gesetzt, Variablen können als *final* markiert werden

```
1  class TitleWidget extends StatelessWidget {
2    TitleWidget({this.title});
3
4    final String title;
5
6    @override
7    Widget build(BuildContext context) {
8      return new Center(child: new Text(this.title));
9    }
10 }
```

Stateful Widgets

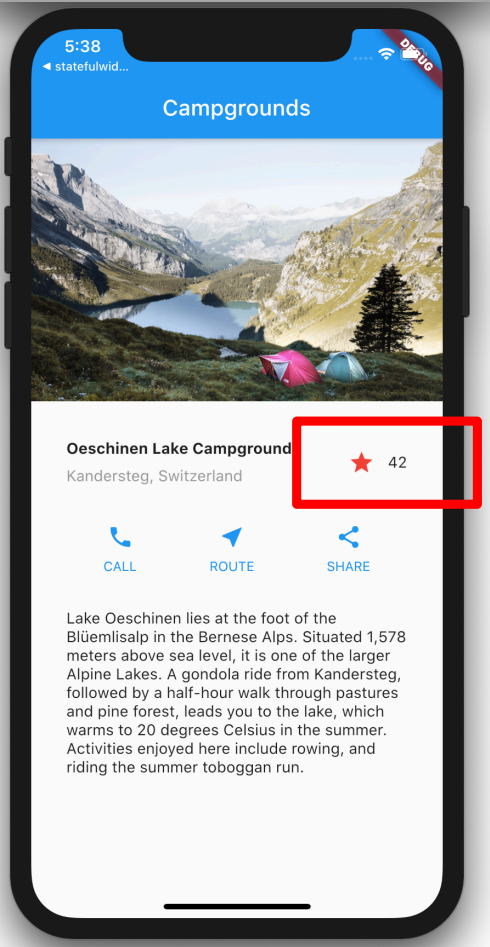
- Beinhalten veränderbaren State, der sich während der Laufzeit ändern kann
- Können während der Laufzeit beliebig oft neu gerendert werden
- Sobald im State Objekt die ***setState()*** Methode aufgerufen wird, wird ***build()*** ausgeführt und das Widget neu gezeichnet

```
1 class HomeScreen extends StatefulWidget {
2   @override
3   _HomeScreenState createState() => _HomeScreenState();
4 }
5
6 class _HomeScreenState extends State<HomeScreen> {
7   @override
8   Widget build(BuildContext context) {
9    // UI
10  }
11 }
12
```

Übung

- Code auschecken
  - > StatefulWidgetExercise
- TODOs bearbeiten, um ein **StatefulWidget** zu erstellen, mit dem der Favorite Button und Counter verwaltet wird

20 Min



Provider

- Provider: Empfohlen vom Flutter Team zum State Management
- Globaler **Datencontainer** (Provider), der zu einem Widget hinzugefügt werden kann → alle Child Widgets haben Zugriff und können **Listener** registrieren
- Sobald State ändert, wird *build()* ausgeführt, aber **nur** in den Widgets, die Listener auf den entsprechenden Provider registriert haben

Provider

Providers

- Stellen die benötigten Objekte/Daten zur Verfügung
- Platzierung im Widget Tree oberhalb der Widgets, die die Daten brauchen
- z.B. ChangeNotifierProvider oder MultiProvider

Notifiers

- Benachrichtigen Provider über Änderungen
- z.B. ChangeNotifier oder ValueNotifier

Consumers

- Werden dort registriert, wo die Daten benötigt werden und wo auf Änderungen der Daten reagiert werden muss
- z.B. Provider.of oder Consumer

Übung

- Code auschecken
  - > ProviderExercise
- TODOs bearbeiten, um das **Provider** Package zum State Management zu verwenden
- Doku zum Nachlesen

40 Min