Flutter providers:

<https://blog.codemagic.io/flutter-state-management-with-riverpod/>

flutter\_riverpod: ^1.0.3

Main.dart:

import 'package:flutter/material.dart';

import 'package:flutter\_riverpod/flutter\_riverpod.dart';

import 'home.dart';

void main() {

runApp(ProviderScope(

child: MyHomePage(),

));

}

**Stateless widget:**

import 'package:flutter/material.dart';

import 'package:flutter\_riverpod/flutter\_riverpod.dart';

final valueProvider = **Provider**<int>((ref) {

return 95;

});

Or

final valueProvider = Provider<String>((ref) {

return "Stalin";

});

Method1:

class MyHomePage extends **StatelessWidget** {

@override

Widget build(BuildContext context) {

return MaterialApp(

home: Scaffold(

body: Center(

*// 1. Add a Consumer*

child: Consumer(

*// 2. specify the builder and obtain a WidgetRef*

**builder: (context, ref, \_) {**

*// 3. use ref.watch() to get the value of the provider*

final value = ref.watch(valueProvider);

return Text(

'Value: $value',

style: Theme.*of*(context).textTheme.headline4,

);

},

),

),

),

);

}

}

Method 2:

class MyHomePage extends **ConsumerWidget** {

@override

*// 2. build() method has an extra [WidgetRef] argument*

**Widget build(BuildContext context, WidgetRef ref) {**

*// 3. use ref.watch() to get the value of the provider*

final value = ref.watch(valueProvider);

return MaterialApp(

home: Scaffold(

body: Center(

child: Text(

'Value: $value',

style: Theme.*of*(context).textTheme.headline4,

),

),

),

);

}

}

**Statefull:**

import 'package:flutter/material.dart';

import 'package:flutter\_riverpod/flutter\_riverpod.dart';

final valueProvider = Provider<int>((ref) {

return 95;

});

class MyHomePage extends **ConsumerStatefulWidget** {

const MyHomePage({Key? key}) : super(key: key);

@override

\_MyHomePageState createState() => \_MyHomePageState();

}

// 2. Extend [ConsumerState]

class \_MyHomePageState extends ConsumerState<MyHomePage> {

@override

void initState() {

super.initState();

// 3. use ref.read() in the widget life-cycle methods

final value = ref.read(valueProvider);

print(value);

}

@override

Widget build(BuildContext context) {

// 3. use ref.watch() to get the value of the provider

final value = ref.watch(valueProvider);

return MaterialApp(

home: Scaffold(

body: Center(

child: Text(

'Value: $value',

style: Theme.of(context).textTheme.headline4,

),

),

),

);

}

}

**StateProvider:**

import 'package:flutter/material.dart';

import 'package:flutter\_riverpod/flutter\_riverpod.dart';

final valueProvider = StateProvider<int>((ref) {

return 100;

});

class MyHomePage extends ConsumerStatefulWidget {

const MyHomePage({Key? key}) : super(key: key);

@override

\_MyHomePageState createState() => \_MyHomePageState();

}

// 2. Extend [ConsumerState]

class \_MyHomePageState extends ConsumerState<MyHomePage> {

@override

void initState() {

super.initState();

// 3. use ref.read() in the widget life-cycle methods

final value = ref.read(valueProvider);

print(value);

}

@override

Widget build(BuildContext context) {

// 3. use ref.watch() to get the value of the provider

final value = ref.watch(valueProvider);

return MaterialApp(

home: Scaffold(

body: Center(

child: Text(

'Value: $value',

style: Theme.of(context).textTheme.headline4,

),

),

floatingActionButton: FloatingActionButton(

// access the provider via ref.read(), then increment its state.

onPressed: () => ref.read(valueProvider.state).state++,

child: Icon(Icons.add),

),

),

);

}

}

**Change notifier:**

import 'package:flutter/material.dart';

import 'package:flutter\_riverpod/flutter\_riverpod.dart';

import 'home.dart';

void main() {

runApp(ProviderScope(

child: MyHomePage(),

));

}

Home.dart:

import 'package:flutter/material.dart';

import 'package:flutter\_riverpod/flutter\_riverpod.dart';

import 'model/counternotifier.dart';

final counterProvider = ChangeNotifierProvider((ref) => CounterNotifier());

class MyHomePage extends ConsumerStatefulWidget {

const MyHomePage({Key? key}) : super(key: key);

@override

\_MyHomePageState createState() => \_MyHomePageState();

}

*// 2. Extend [ConsumerState]*

class \_MyHomePageState extends ConsumerState<MyHomePage> {

@override

void initState() {

super.initState();

*// 3. use ref.read() in the widget life-cycle methods*

final value = ref.read(counterProvider);

print(value);

}

void \_incrementCounter(BuildContext context, ref) {

ref.read(counterProvider).incrementValue();

}

@override

Widget build(BuildContext context) {

*// 3. use ref.watch() to get the value of the provider*

final counterNotifier = ref.watch(counterProvider);

return MaterialApp(

home: Scaffold(

body: Center(

child: Text(

'${counterNotifier.value}',

style: Theme.*of*(context).textTheme.headline4,

),

),

floatingActionButton: FloatingActionButton(

*// access the provider via ref.read(), then increment its state.*

onPressed: () => \_incrementCounter(context, ref),

child: Icon(Icons.*add*),

),

),

);

}

}

**Counternotifier.dart:**

import 'package:flutter/material.dart';

class CounterNotifier extends ChangeNotifier {

int \_value = 0;

int get value => \_value;

void incrementValue() {

\_value++;

notifyListeners();

}

}

**FutureNotifier:**

import 'package:flutter/material.dart';

import 'package:flutter\_riverpod/flutter\_riverpod.dart';

import 'model/counternotifier.dart';

import 'model/fakeweathercliet.dart';

final counterProvider = ChangeNotifierProvider((ref) => CounterNotifier());

final fakeWeatherClientProvider = Provider((ref) => FakeWeatherClient());

final responseProvider = FutureProvider<int>((ref) async {

final weatherClient = ref.read(fakeWeatherClientProvider);

return weatherClient.get('Mumbai');

});

class MyHomePage extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

home: Scaffold(

body: Center(

*// 1. Add a Consumer*

child: Consumer(

*// 2. specify the builder and obtain a WidgetRef*

builder: (context, ref, \_) {

*// 3. use ref.watch() to get the value of the provider*

final responseValue = ref.watch(responseProvider);

return responseValue.map(

data: (weather) => Text(

'${weather.value}',

style: TextStyle(fontSize: 25),

),

loading: (\_) => CircularProgressIndicator(),

error: (message) => Text("message.error"),

);

},

),

),

),

);

}

}

**fakeweatherclient.dart**

class FakeWeatherClient {

Future<int> get(String cityName) async {

await Future.delayed(Duration(seconds: 2));

return cityName == 'Texus' ? 18 : 21;

}

}

Method 2:

import 'package:flutter/material.dart';

import 'package:flutter\_riverpod/flutter\_riverpod.dart';

import 'model/counternotifier.dart';

import 'model/fakeweathercliet.dart';

final counterProvider = ChangeNotifierProvider((ref) => CounterNotifier());

final fakeWeatherClientProvider = Provider((ref) => FakeWeatherClient());

*// final responseProvider = FutureProvider<int>((ref) async {*

*// final weatherClient = ref.read(fakeWeatherClientProvider);*

*// return weatherClient.get('Mumbai');*

*// });*

final responseProvider =

FutureProvider.*autoDispose*.family<int, String>((ref, cityName) async {

final weatherClient = ref.read(fakeWeatherClientProvider);

return weatherClient.get(cityName);

});

class MyHomePage extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

home: Scaffold(

body: Center(

*// 1. Add a Consumer*

child: Consumer(

*// 2. specify the builder and obtain a WidgetRef*

builder: (context, ref, \_) {

*// 3. use ref.watch() to get the value of the provider*

final responseValue = ref.watch(responseProvider('Kokatta'));

return responseValue.map(

data: (weather) => Text(

'${weather.value}',

style: TextStyle(fontSize: 25),

),

loading: (\_) => CircularProgressIndicator(),

error: (message) => Text("message.error"),

);

},

),

),

),

);

}

}

**Auto\_dispose\_modifier\_page.dart**

**import 'package:flutter/material.dart';**

**import 'package:flutter\_riverpod/flutter\_riverpod.dart';**

**import 'package:river/widget/text\_widget.dart';**

**void main() {**

**runApp(ProviderScope(**

**child: AutoDisposeModifierPage(),**

**));**

**}**

**Future<String> fetchValue() async {**

**await Future.delayed(Duration(seconds: 3));**

**return 'State will be disposed!';**

**}**

**final futureProvider =**

**FutureProvider.*autoDispose*<String>((ref) => fetchValue());**

**class AutoDisposeModifierPage extends ConsumerWidget {**

**@override**

**Widget build(BuildContext context, WidgetRef ref) {**

**final future = ref.watch(futureProvider);**

**return MaterialApp(**

**home: Scaffold(**

**appBar: AppBar(**

**title: Text('AutoDispose Modifier'),**

**),**

**body: Center(**

**child: future.when(**

**data: (value) => Padding(**

**padding: EdgeInsets.all(48),**

**child: TextWidget(value.toString()),**

**),**

**loading: () => CircularProgressIndicator(),**

**error: (e, stack) => TextWidget('Error: $e'),**

**),**

**),**

**),**

**);**

**}**

**}**

**Text\_widget:**

**import 'package:flutter/material.dart';**

**class TextWidget extends StatelessWidget {**

**final String text;**

**const TextWidget(this.text);**

**@override**

**Widget build(BuildContext context) => Text(**

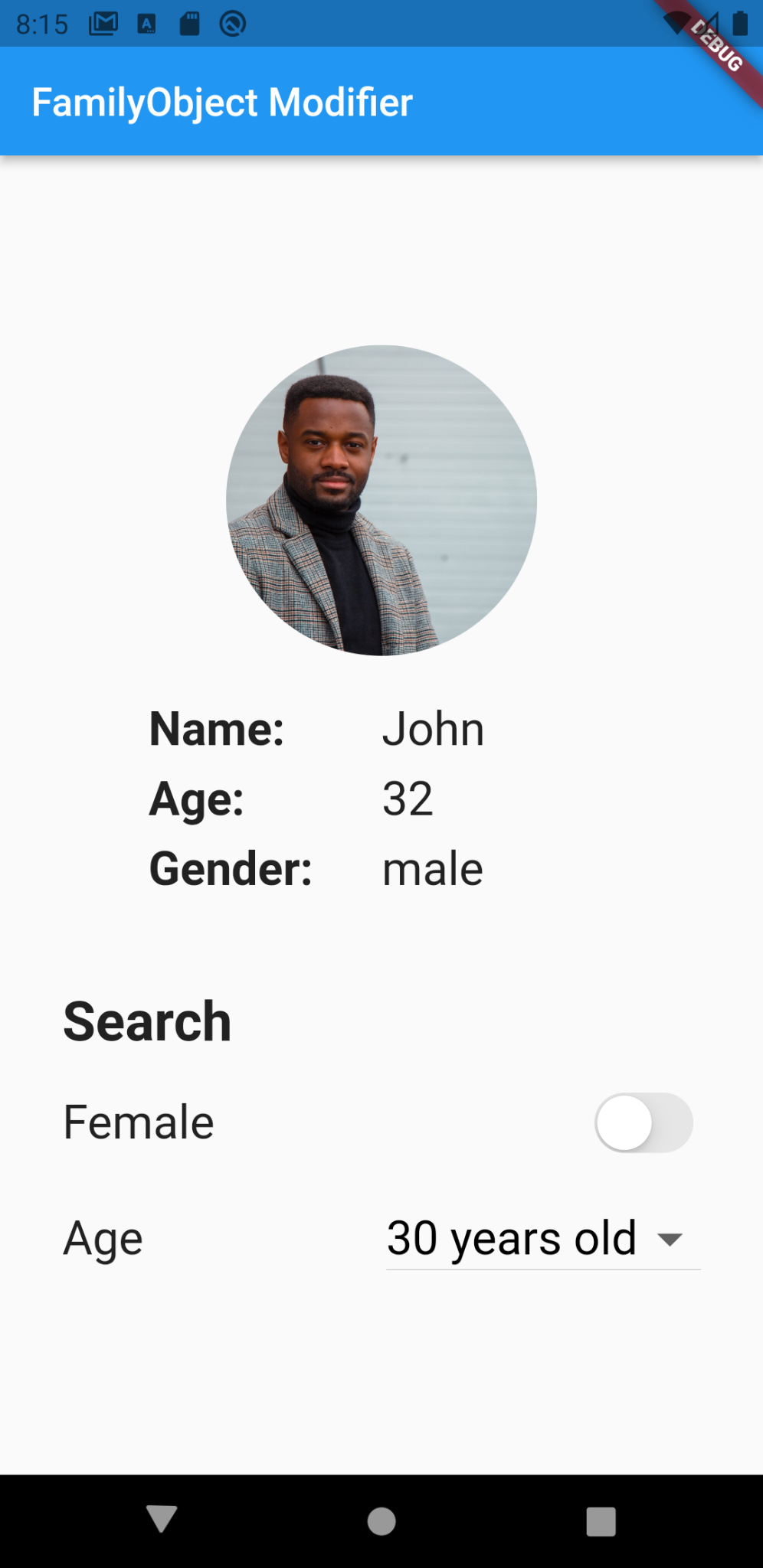
**text,**

**style: TextStyle(fontSize: 62),**

**);**

**}**

**Familyobject Modifier:**

****

import 'package:flutter/cupertino.dart';

import 'package:flutter/material.dart';

import 'package:flutter\_riverpod/flutter\_riverpod.dart';

import 'package:river/widget/text\_widget.dart';

import 'package:river/widget/user\_widget.dart';

import 'modifiers/user\_helper.dart';

void main() {

runApp(ProviderScope(

child: FamilyObjectModifierPage(),

));

}

class UserRequest {

final bool isFemale;

final int minAge;

const UserRequest({

required this.isFemale,

required this.minAge,

});

@override

bool operator ==(Object other) =>

identical(this, other) ||

other is UserRequest &&

runtimeType == other.runtimeType &&

isFemale == other.isFemale &&

minAge == other.minAge;

@override

int get hashCode => isFemale.hashCode ^ minAge.hashCode;

}

Future<User> fetchUser(UserRequest request) async {

await Future.delayed(Duration(milliseconds: 400));

final gender = request.isFemale ? 'female' : 'male';

return users.firstWhere(

(user) => user.gender == gender && user.age >= request.minAge);

}

final userProvider = FutureProvider.*family*<User, UserRequest>(

(ref, userRequest) async => fetchUser(userRequest));

class FamilyObjectModifierPage extends StatefulWidget {

@override

\_FamilyObjectModifierPageState createState() =>

\_FamilyObjectModifierPageState();

}

class \_FamilyObjectModifierPageState extends State<FamilyObjectModifierPage> {

static final *ages* = [18, 25, 30, 40];

bool isFemale = true;

int minAge = *ages*.first;

@override

Widget build(BuildContext context) {

return MaterialApp(

home: Scaffold(

appBar: AppBar(

title: Text('FamilyObject Modifier'),

),

body: Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

Container(

height: 300,

child: Consumer(builder: (context, ref, \_) {

final userRequest =

UserRequest(isFemale: isFemale, minAge: minAge);

final future = ref.watch(userProvider(userRequest));

return future.when(

data: (user) => UserWidget(user: user),

loading: () => Center(child: CircularProgressIndicator()),

error: (e, stack) => Center(child: TextWidget('Not found')),

);

}),

),

const SizedBox(height: 32),

buildSearch(),

],

),

),

),

);

}

Widget buildSearch() => Container(

width: double.*infinity*,

padding: EdgeInsets.symmetric(horizontal: 32),

child: Column(

crossAxisAlignment: CrossAxisAlignment.start,

children: [

Text(

'Search',

style: TextStyle(fontSize: 28, fontWeight: FontWeight.*bold*),

),

const SizedBox(height: 16),

buildGenderSwitch(),

const SizedBox(height: 16),

buildAgeDropdown(),

],

),

);

Widget buildGenderSwitch() => Row(

children: [

Text(

'Female',

style: TextStyle(fontSize: 24),

),

Spacer(),

CupertinoSwitch(

value: isFemale,

onChanged: (value) => setState(() => isFemale = value),

),

],

);

Widget buildAgeDropdown() => Row(

children: [

Text(

'Age',

style: TextStyle(fontSize: 24),

),

Spacer(),

DropdownButton<int>(

value: minAge,

iconSize: 32,

style: TextStyle(fontSize: 24, color: Colors.*black*),

onChanged: (value) => setState(() => minAge = value!),

items: *ages*

.map<DropdownMenuItem<int>>(

(int value) => DropdownMenuItem<int>(

value: value,

child: Text('$value years old'),

))

.toList(),

),

],

);

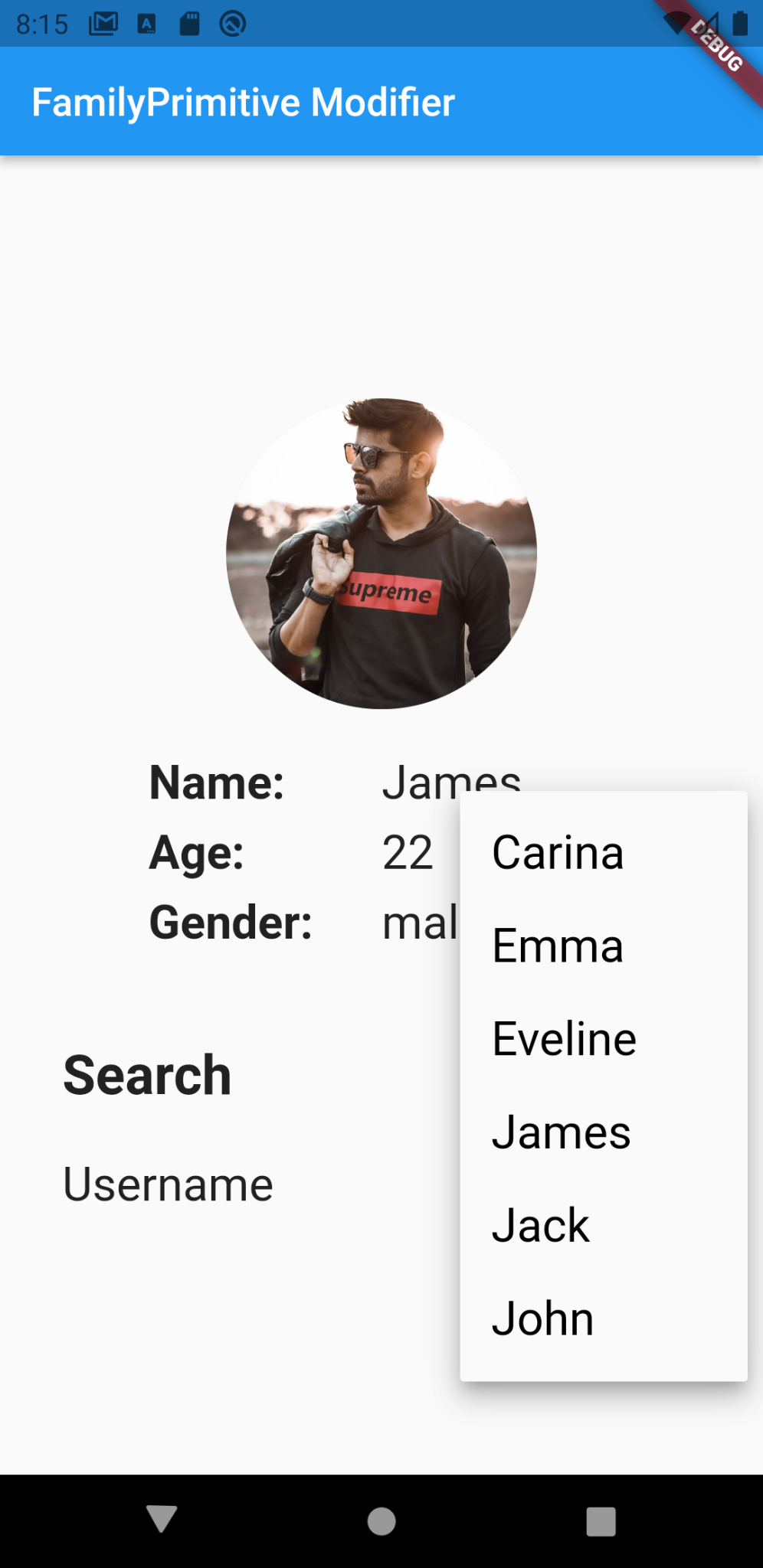
}

1.<https://github.com/JohannesMilke/riverpod_modifiers_example/blob/master/lib/page/modifiers/user_helper.dart>

2.<https://github.com/JohannesMilke/riverpod_modifiers_example/blob/master/lib/widget/user_widget.dart>

3.<https://github.com/JohannesMilke/riverpod_modifiers_example/blob/master/lib/widget/text_widget.dart>

Family primitive:



import 'package:flutter/cupertino.dart';

import 'package:flutter/material.dart';

import 'package:flutter\_riverpod/flutter\_riverpod.dart';

import 'package:river/widget/text\_widget.dart';

import 'package:river/widget/user\_widget.dart';

import 'modifiers/user\_helper.dart';

void main() {

runApp(ProviderScope(

child: FamilyPrimitiveModifierPage(),

));

}

Future<User> fetchUser(String username) async {

await Future.delayed(Duration(milliseconds: 400));

return users.firstWhere((user) => user.name == username);

}

final userProvider = FutureProvider.*family*<User, String>(

(ref, username) async => fetchUser(username));

class FamilyPrimitiveModifierPage extends StatefulWidget {

@override

\_FamilyPrimitiveModifierPageState createState() =>

\_FamilyPrimitiveModifierPageState();

}

class \_FamilyPrimitiveModifierPageState

extends State<FamilyPrimitiveModifierPage> {

String username = users.first.name;

@override

Widget build(BuildContext context) {

return MaterialApp(

home: Scaffold(

appBar: AppBar(

title: Text('FamilyPrimitive Modifier'),

),

body: Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

Container(

height: 300,

child: Consumer(builder: (context, ref, \_) {

final future = ref.watch(userProvider(username));

return future.when(

data: (user) => UserWidget(user: user),

loading: () => Center(child: CircularProgressIndicator()),

error: (e, stack) => Center(child: TextWidget('Not found')),

);

}),

),

const SizedBox(height: 32),

buildSearch(),

],

),

),

),

);

}

Widget buildSearch() => Container(

width: double.*infinity*,

padding: EdgeInsets.symmetric(horizontal: 32),

child: Column(

crossAxisAlignment: CrossAxisAlignment.start,

children: [

Text(

'Search',

style: TextStyle(fontSize: 28, fontWeight: FontWeight.*bold*),

),

const SizedBox(height: 16),

buildUsernameDropdown(),

],

),

);

Widget buildUsernameDropdown() => Row(

children: [

Text(

'Username',

style: TextStyle(fontSize: 24),

),

Spacer(),

DropdownButton<String>(

value: username,

iconSize: 32,

style: TextStyle(fontSize: 24, color: Colors.*black*),

onChanged: (value) => setState(() => username = value!),

items: users

.map((user) => user.name)

.map<DropdownMenuItem<String>>(

(String value) => DropdownMenuItem<String>(

value: value,

child: Text(value),

))

.toList(),

),

],

);

}

Future Provider

import 'package:flutter/cupertino.dart';

import 'package:flutter/material.dart';

import 'package:flutter\_riverpod/flutter\_riverpod.dart';

import 'package:river/widget/text\_widget.dart';

void main() {

runApp(ProviderScope(

child: CombinedProvidersPage(),

));

}

final cityProvider = Provider<String>((ref) => 'Munich2');

Future<int> fetchWeather(String city) async {

await Future.delayed(Duration(seconds: 2));

return city == 'Munich' ? 20 : 15;

}

final futureProvider = FutureProvider<int>((ref) async {

final city = ref.watch(cityProvider);

return fetchWeather(city);

});

class CombinedProvidersPage extends ConsumerWidget {

@override

Widget build(BuildContext context, WidgetRef ref) {

final future = ref.watch(futureProvider);

return MaterialApp(

home: Scaffold(

appBar: AppBar(

title: Text('Combining Providers'),

),

body: Center(

child: future.when(

data: (value) => TextWidget(value.toString()),

loading: () => CircularProgressIndicator(),

error: (e, stack) => TextWidget('Error: $e'),

),

),

),

);

}

}

Another:

import 'package:flutter/cupertino.dart';

import 'package:flutter/material.dart';

import 'package:flutter\_riverpod/flutter\_riverpod.dart';

import 'package:river/widget/text\_widget.dart';

void main() {

runApp(ProviderScope(

child: CombinedProvidersPage(),

));

}

final cityProvider = Provider<String>((ref) => 'Munich2');

Future<int> fetchWeather(String city) async {

await Future.delayed(Duration(seconds: 2));

return city == 'Munich' ? 20 : 15;

}

Future<int> fetchWeatherbyinteger() async {

await Future.delayed(Duration(seconds: 2));

return 20;

}

final futureProvider = FutureProvider<int>((ref) async {

final city = ref.watch(cityProvider);

return fetchWeather(city);

});

final futureProviderbyinteger =

FutureProvider<int>((ref) => fetchWeatherbyinteger());

class CombinedProvidersPage extends ConsumerWidget {

@override

Widget build(BuildContext context, WidgetRef ref) {

final future = ref.watch(futureProviderbyinteger);

return MaterialApp(

home: Scaffold(

appBar: AppBar(

title: Text('Combining Providers'),

),

body: Center(

child: future.when(

data: (value) => TextWidget(value.toString()),

loading: () => CircularProgressIndicator(),

error: (e, stack) => TextWidget('Error: $e'),

),

),

),

);

}

}

Provider:

import 'package:flutter/cupertino.dart';

import 'package:flutter/material.dart';

import 'package:flutter\_riverpod/flutter\_riverpod.dart';

import 'package:river/widget/text\_widget.dart';

void main() {

runApp(ProviderScope(

child: CombinedProvidersPage(),

));

}

final numberProvider = Provider<int>((ref) => 53);

class CombinedProvidersPage extends ConsumerWidget {

@override

Widget build(BuildContext context, WidgetRef ref) {

final value = ref.watch(numberProvider);

return MaterialApp(

home: Scaffold(

appBar: AppBar(

title: Text('Combining Providers'),

),

body: Center(

child: TextWidget(value.toString()),

),

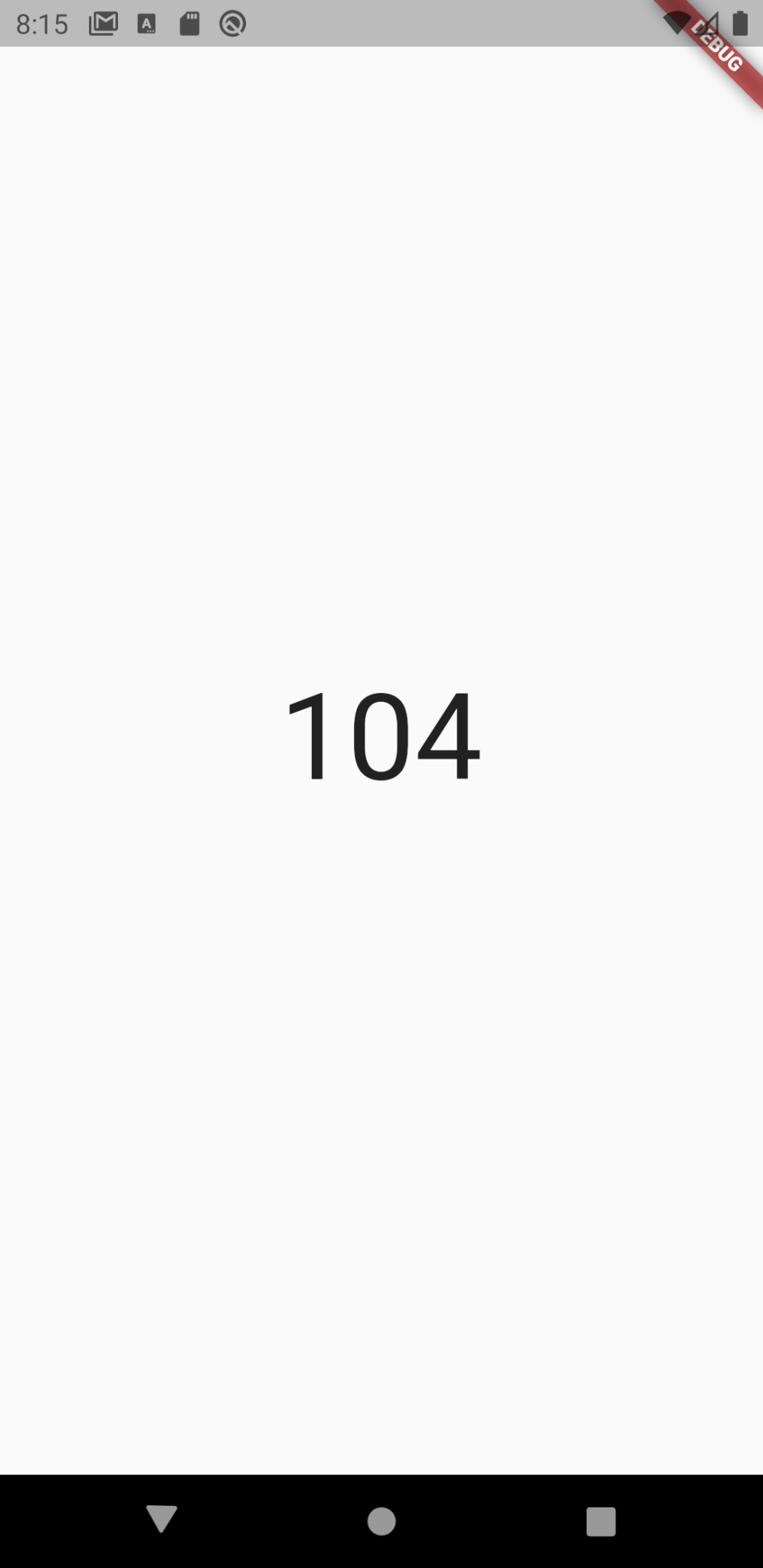
),

);

}

}

Stream builder:



import 'package:flutter/cupertino.dart';

import 'package:flutter/material.dart';

import 'package:flutter\_riverpod/flutter\_riverpod.dart';

import 'package:river/widget/text\_widget.dart';

void main() {

runApp(ProviderScope(

child: MyHomePage(),

));

}

*// StreamProvider.autoDispose: destroys state if no-longer listened*

final streamProvider = StreamProvider<String>((ref) => Stream.periodic(

Duration(milliseconds: 400),

(count) => '$count',

));

class MyHomePage extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

home: Scaffold(

body: Center(

*// 1. Add a Consumer*

child: Consumer(

*// 2. specify the builder and obtain a WidgetRef*

builder: (context, ref, \_) {

*// 3. use ref.watch() to get the value of the provider*

final stream = ref.watch(streamProvider);

return stream.when(

data: (value) => TextWidget(value),

loading: () => CircularProgressIndicator(),

error: (e, stack) => TextWidget('Error: $e'),

);

},

),

),

),

);

}

}

Changenotifier

import 'dart:math';

import 'package:flutter/cupertino.dart';

import 'package:flutter/material.dart';

import 'package:flutter\_riverpod/flutter\_riverpod.dart';

import 'package:river/widget/button\_widget.dart';

import 'package:river/widget/text\_widget.dart';

void main() {

runApp(ProviderScope(

child: ChangeNotifierPage(),

));

}

class CarNotifier extends ChangeNotifier {

int \_speed = 120;

void increase() {

\_speed += 5;

notifyListeners();

}

void hitBreak() {

\_speed = max(0, \_speed - 30);

notifyListeners();

}

*// @override*

*// void dispose() {}*

}

final carProvider = ChangeNotifierProvider<CarNotifier>((ref) => CarNotifier());

class ChangeNotifierPage extends ConsumerWidget {

@override

Widget build(BuildContext context, WidgetRef ref) {

final car = ref.watch(carProvider);

return MaterialApp(

home: Scaffold(

appBar: AppBar(

title: Text('ChangeNotifierProvider'),

),

body: Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

TextWidget('Speed: ${car.\_speed}'),

const SizedBox(height: 8),

Row(

mainAxisAlignment: MainAxisAlignment.center,

children: [

ButtonWidget(

text: 'Increase +5',

onClicked: ref.read(carProvider).increase),

*// car.increase: not efficient*

*// => button rebuilds everytime if car state changes*

const SizedBox(width: 12),

ButtonWidget(

text: 'Hit Brake -30',

onClicked: ref.read(carProvider).hitBreak,

),

],

)

],

),

),

),

);

}

}

Button.dart

import 'package:flutter/material.dart';

class ButtonWidget extends StatelessWidget {

final String text;

final VoidCallback onClicked;

const ButtonWidget({

required this.text,

required this.onClicked,

Key? key,

}) : super(key: key);

@override

Widget build(BuildContext context) => RaisedButton(

child: Text(

text,

style: TextStyle(fontSize: 24),

),

shape: StadiumBorder(),

color: Theme.*of*(context).accentColor,

padding: EdgeInsets.symmetric(horizontal: 16, vertical: 8),

textColor: Colors.*white*,

onPressed: onClicked,

);

}