import 'package:english\_words/english\_words.dart';

import 'package:flutter/material.dart';

import 'package:provider/provider.dart';

// starts with main function

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

const MyApp({super.key});

// build: What does this widget contain

@override

Widget build(BuildContext context) {

return ChangeNotifierProvider(

// MyAppState created here in myApp

create: (context) => MyAppState(),

child: MaterialApp(

title: 'Namer App',

theme: ThemeData(

// Buttons look a certain way

useMaterial3: true,

// Button trying to be this colour

colorScheme: ColorScheme.fromSeed(

seedColor: Color.fromARGB(255, 161, 76, 179)),

),

home: MyHomePage(),

),

);

}

}

// ChangeNotifier: "Oh i have changed!", should take note

class MyAppState extends ChangeNotifier {

var current = WordPair.random();

void getNext() {

current = WordPair.random();

notifyListeners(); // Hey notice me!

}

// List of wordPair

var favorites = <WordPair>[];

void toggleFavorite() {

if (favorites.contains(current)) {

favorites.remove(current);

} else {

favorites.add(current);

}

notifyListeners(); // Rmb to notify anyone whos intested to MyAppState (Incase important)

}

}

// New MyHomePage (Navigation part)

class MyHomePage extends StatefulWidget {

@override

State<MyHomePage> createState() => \_MyHomePageState();

}

class \_MyHomePageState extends State<MyHomePage> {

// New property on \_MyHomePageState not the widget

var selectedIndex = 0;

@override

Widget build(BuildContext context) {

// Im Defininf a new varaible called page (a widget)

Widget page;

// Based on selectedIndex, call these...

switch (selectedIndex) {

case 0:

page = GeneratorPage();

break;

case 1:

// Draw a little box lol

// Nice when we working on the app and not yet ready it!

// page = Placeholder();

page = FavoritesPage();

break;

// HMMMM if didnt have default, Dart would know page could be NULL, need to be something LOL

default:

throw UnimplementedError('no widget for $selectedIndex');

}

return Scaffold(

body: Row(

// A new Row with 2 children

children: [

// SafeArea lol

SafeArea(

child: NavigationRail(

// Extended view something like that

extended: false,

destinations: [

NavigationRailDestination(

icon: Icon(Icons.home),

label: Text('Home'),

),

NavigationRailDestination(

icon: Icon(Icons.favorite),

label: Text('Favorites'),

),

],

// 0 was the home (old code lol)

// selectedIndex: 0;

selectedIndex: selectedIndex,

// Call back to navigation rail, when user selects a new navi

onDestinationSelected: (value) {

// Whenever any code that changes the code, we must wrap it in setState

// Telling something important happens and should rebuild it!!

setState(() {

selectedIndex = value;

});

},

),

),

// Special widget used in rows/ columns

// "Hey give me as much space as possible, please!!"

Expanded(

child: Container(

color: Theme.of(context).colorScheme.primaryContainer,

child: page,

),

),

],

),

);

}

}

class GeneratorPage extends StatelessWidget {

@override

Widget build(BuildContext context) {

var appState = context.watch<MyAppState>();

var pair = appState.current;

IconData icon;

if (appState.favorites.contains(pair)) {

icon = Icons.favorite;

} else {

icon = Icons.favorite\_border;

}

return Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

BigCard(pair: pair),

SizedBox(height: 20),

Row(

mainAxisSize: MainAxisSize.min,

children: [

ElevatedButton.icon(

onPressed: () {

appState.toggleFavorite();

},

icon: Icon(icon),

label: Text('Like'),

),

SizedBox(width: 10),

ElevatedButton(

onPressed: () {

appState.getNext();

},

child: Text('Next'),

),

],

),

],

),

);

}

}

// class MyHomePage extends StatelessWidget {

// @override

// Widget build(BuildContext context) {

// // It watches MyAppState

// // "Hey i want to rebuild everytime this MyAppState change"

// var appState = context.watch<MyAppState>();

// var pair = appState.current;

// IconData icon;

// // If current pair in flavoured, this icon

// if (appState.favorites.contains(pair)) {

// icon = Icons.favorite;

// } else {

// // Or not this icon

// icon = Icons.favorite\_border;

// }

// return Scaffold(

// body: Center(

// // Center horizontally, if just a widget not in a column or row ...

// child: Column(

// // Column, didnt know how to deal with children, put them all at top

// mainAxisAlignment: MainAxisAlignment.center,

// // Centering vertically

// children: [

// // Text(appState.current.asPascalCase), replaced to pair (depends on a single variable but not on all of appState)

// // Make sure our widget only that the thing they need

// // Text(pair.asLowerCase); [Extract widget becomes BigCard] Refactor

// BigCard(pair: pair),

// SizedBox(

// height: 20,

// ),

// Row(

// mainAxisSize: MainAxisSize.min,

// // Hey row! Dont take everything (whole verticle axis) !, Default: .max

// children: [

// // ElevatedButton(

// // onPressed: () {

// // appState.toggleFavorite();

// // },

// // child: Text("Flavourite")),

// ElevatedButton.icon(

// onPressed: () {

// appState.toggleFavorite();

// },

// icon: Icon(icon),

// label: Text('Like'),

// ),

// SizedBox(width: 20,),

// ElevatedButton(

// onPressed: () {

// appState.getNext();

// },

// child: Text('Next'),

// ),

// ],

// ),

// ],

// ),

// ),

// );

// }

// }

class BigCard extends StatelessWidget {

const BigCard({

super.key,

required this.pair,

});

final WordPair pair;

@override

Widget build(BuildContext context) {

// Similar: var appState = context.watch<MyAppState>();

// Watches Theme data

var theme = Theme.of(context);

// textTheme: has font definitions for all text in app

// display: Huge chunk of text

final style = theme.textTheme.displayMedium!.copyWith(

// onPrimary: colorScheme automatically choose a clear colour (dynamic)

color: theme.colorScheme.onPrimary,

);

return Card(

// Widget: Material design part

// Bold colour from the theme

color: theme.colorScheme.primary,

child: Padding(

padding: const EdgeInsets.all(20.0),

child: Text(

pair.asLowerCase,

style: style,

// For screen readers

semanticsLabel: pair.asPascalCase,

),

),

);

}

}

class FavoritesPage extends StatelessWidget {

@override

Widget build(BuildContext context) {

var appState = context.watch<MyAppState>();

if (appState.favorites.isEmpty) {

return Center(

child: Text('No favorites yet.'),

);

}

return ListView(

children: [

Padding(

padding: const EdgeInsets.all(20),

child: Text('You have '

'${appState.favorites.length} favorites:'),

),

for (var pair in appState.favorites)

ListTile(

leading: Icon(Icons.favorite),

title: Text(pair.asLowerCase),

),

],

);

}

}