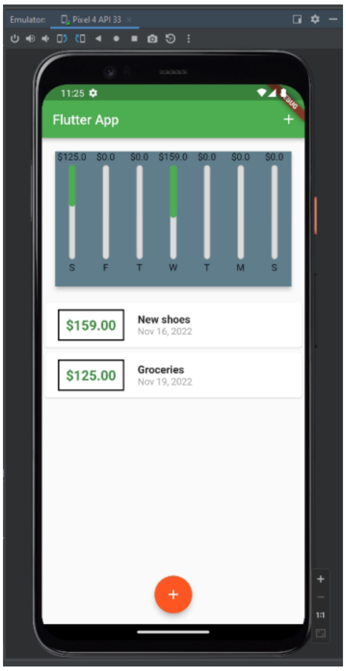
Week 7

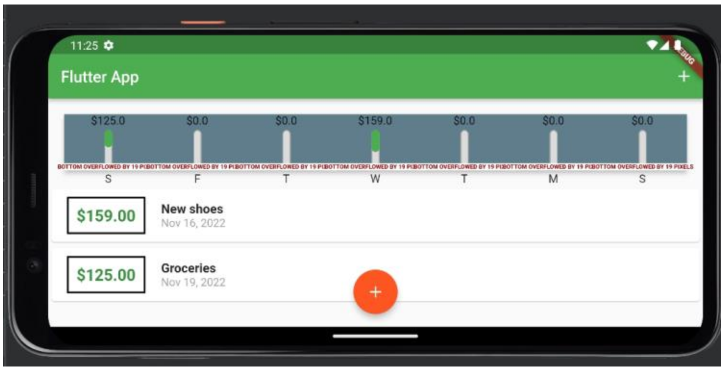
Responsive and Adaptive UIs

- What is responsive? Users can use the mobile devices in Portrait mode or Landscape-mode. Or, they use Tablets or PCs.
- What is adaptive? You want that your app may look different in Android or iOS by using a single code base.
  - For Android: Material design look, Android animations and fonts
  - For iOS: Cupertino look, iOS animations and fonts
- We have one root widget-tree but by using **if Platform.isIOS** we can create a different sub-tree.
- Open Week6 demo and change the height of the contaion of listview to **height: MediaQuery.of(context).size.height*0.6,**
- Add a height parameter to the chart card: **height: MediaQuery.of(context).size.height*0.4,**
- However, you can scroll the page either why? Because we did not count the AppBar and StatusBar.
- You can spare 0.1 percent for the AppBar height and redistribute the chart height as 0.3 and listview to 0.6 percent.
- If you want to measure the exact heights of the appbar and the status bar you can use these:
  - appBar.preferedSize.height
  - MediaQuery.of(context).padding.top
- If you want to get the textScaleFactor of the mobile device, you can use:
  - Final curScaleFactor = MediaQuery.of(context).textScaleFactor;
  - Text('Hello World!', style: TextStyle(fontSize:20*curScaleFactor));
- To adjust the bar height dynamically wrap the Row of the chart by a LayoutBuilder:

```
height: MediaQuery.of(context).size.height*0.3,
child: Card(
  child: Container(
    child: LayoutBuilder(
      builder: (ctx,constraints) {
        return Row(
          children: grouppedTransactionValues.map((data) {
```

- Adjust the height of the bar by **height: constraints.maxHeight*0.7,**
- Try to run this app on Android Studio with the Android emulator and rotate the device (from portrait mode to landscape mode) then see the result.





- How to enforce the app not to work on Landscape mode?
- import 'package:flutter/services.dart';
- Change the main function like this:

```
void main() {
  WidgetsFlutterBinding.ensureInitialized();

SystemChrome.setPreferredOrientations([DeviceOrientation.portraitUp,DeviceOrientation.portraitDown]);
  runApp(MyApp());
}
```

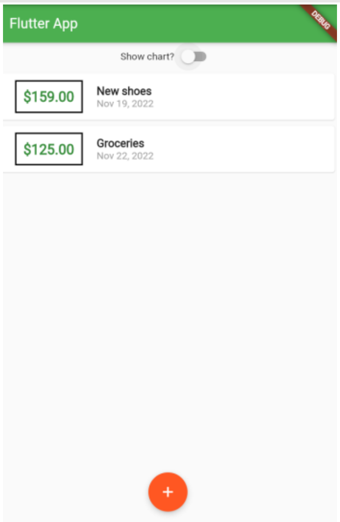
- Now, the user can only use the portrait-mode.
- How can we show proper content in landscape-mode? We can use Switch widget.
- Add a bool variable above the build method: **bool _showChart = false;**
- Add a Switch before the chart, invoke the **setState** inside the **onChanged** parameter and modify the bool variable:

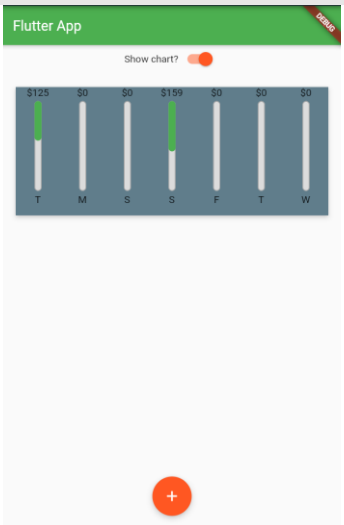
```
            Row(
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                Text('Show chart?'),
                Switch(
                  value: _showChart,
                  onChanged: (val) {
                    setState(() {
                      _showChart = val;
                    });
                  },
                ),
              ],
            ),
```

- Use this bool variable to show the chart or the list:

```
_showChart
 ?
```

```
//chart
Container(
…
)
 :
//list
Column(
…
),
```





- We will show the Switch only in the landscape-mode by adding this definition inside the build method: **final isLandscape= MediaQuery.of(context).orientation == Orientation.landscape;**
- Add an if statement before the Row of Switch: **if (isLandscape) Row …**
- And add extra if statements before the chart and list:

```
         if ((isLandscape && _showChart) || (!isLandscape))
          //chart
          Container(
          …
           ),
          if ((isLandscape && !_showChart) || (!isLandscape))
          //list
          Column(
          …
           ),
```

- Please don’t forget to comment the codes in main function that enforces users not to enter the landscape-mode:

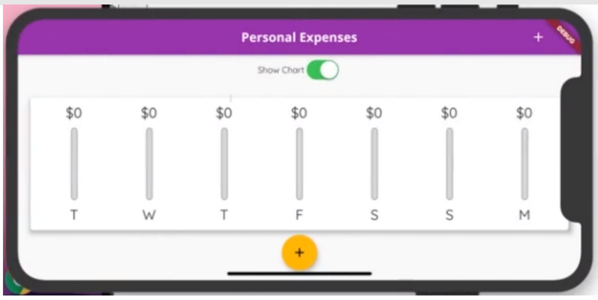
```
 /*
 WidgetsFlutterBinding.ensureInitialized();
 SystemChrome.setPreferredOrientations(
    [DeviceOrientation.portraitUp, DeviceOrientation.portraitDown]);
  */
```

- If you run that app again (in Android Studio), you will see that in portrait-mode both chart and list are visible and the Switch button is invisible. However, in landscape-mode the Switch is visible and user can select to see the chart or list.
- Image running your app in a device with a small display. While entering data in the Modal Bottom Sheet, the soft keyboard may fill the screen and user cannot enter data.
- To overcome this issue, change the padding of the Container of the Card in _startAddNewTransaction method. And wrap the Card with a SingleChildScrollView.

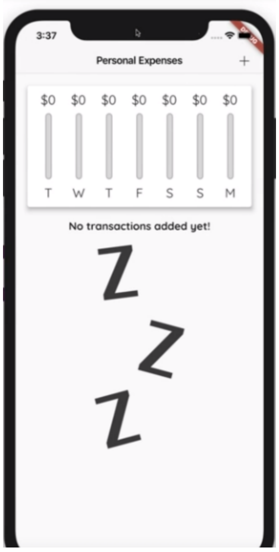
```
child: SingleChildScrollView(
          child: Card(
            elevation: 5,
            child: Container(
              padding: EdgeInsets.only(
                top: 10,
                left: 10,
                right: 10,
                bottom: MediaQuery.of(ctx).viewInsets.bottom +10,
              ),
```

- If you want to check the available width to show extra content, you can use: **if (MediaQuery.of(context).size.width>360) …**
- If you are using MediaQuery widget so many times in a project, assign it in a final variable in the build method and then call this variable: **final mediaQ= MediaQuery.of(context);**
- If your app is running on iOS, your app until now will work fine. However, some widgets may be weird for iOS users for example FloatingActionButton and Switch widgets. You can use this constructor of Switch to look iOS or Android-like style adaptive to the environment that your app is running:

```
              Switch.adaptive(
                value: _showChart,
                onChanged: (val) {
                  setState(() {
                    _showChart = val;
                  });
                },
              ),
```



- If you don't want to show the floating action button in iOS, first import this library above the other imports: **import 'dart:io';**
- Change the definition of the button like this: **floatingActionButton: Platform.isIOS ? Container() : FloatingActionButton(…**
- If you are willing to create a more native look for iOS, you can consider using Cupertino style widgets: https://docs.flutter.dev/development/ui/widgets/cupertino
- First you need to import the proper library: **import 'package:flutter/cupertino.dart';**
- You can check the platform then use a Material or Cupertino Scaffold: **return Platform.isIOS ? CupertinoPageScaffold (…) : Scaffold (…);**



Personal Expenses

No transactions added yet!

```
import 'dart:io';

import 'package:flutter/material.dart';
import 'package:intl/intl.dart';
import 'package:flutter/services.dart';

void main() {
  /*
  WidgetsFlutterBinding.ensureInitialized();
  SystemChrome.setPreferredOrientations(
      [DeviceOrientation.portraitUp, DeviceOrientation.portraitDown]);
   */
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter App',
      theme: ThemeData(
        primarySwatch: Colors.green,
        accentColor: Colors.deepOrange,
        fontFamily: 'Consolas',
      ),
      home: TransactionList(),
    );
  }
}

class TransactionList extends StatefulWidget {
  @override
  _TransactionListState createState() => _TransactionListState();
}

class _TransactionListState extends State<TransactionList> {
  final List<Transaction> transactions = [
    Transaction(
        id: 't1',
        title: 'New shoes',
        amount: 159.0,
        date: DateTime.now().subtract(Duration(days: 3))),
    Transaction(
      id: 't2', title: 'Groceries', amount: 125.0, date: DateTime.now()),
];
final titleContoller = TextEditingController();
final amountController = TextEditingController();
void addTransaction(String txtitle, double txamount) {
  final newTx = Transaction(
      title: txtitle, amount: txamount, id: 'tx', date: DateTime.now());
  setState(() {
    transactions.add(newTx);
  });
  Navigator.of(context).pop();
}

void _startAddNewTransaction(BuildContext ctx) {
  showModalBottomSheet(
    context: ctx,
    builder: (ctx) {
      return GestureDetector(
        behavior: HitTestBehavior.opaque,
        onTap:
            () {}, //use this with gesturedetector to prevent close the bottomsheet when you click inside the sheet
        child: SingleChildScrollView(
          child: Card(
            elevation: 5,
            child: Container(
              padding: EdgeInsets.only(
                top: 10,
                left: 10,
                right: 10,
                bottom: MediaQuery.of(ctx).viewInsets.bottom +10,
              ),
              child: Column(
                crossAxisAlignment: CrossAxisAlignment.end,
                children: [
                  TextField(
                    decoration: InputDecoration(labelText: 'Title'),
                    controller: titleContoller,
                  ),
                  TextField(
                    decoration: InputDecoration(labelText: 'Amount'),
                    controller: amountController,
                    keyboardType: TextInputType.number,
                  ),
                  TextButton(
                    child: Text('Add Transaction'),
                    onPressed: () {
                      print(titleContoller.text);
                      print(amountController.text);
                      addTransaction(titleContoller.text,
                          double.parse(amountController.text));
                    },
                  ), //textbutton
                ], //column children
              ), //column
            ), //container
          ),
        ), //card
      ); //gesturedetector return
    }, //showModalBottomSheet builder
  ); //showModalBottomSheet function
} //_startAddNewTransaction function

List<Map<String, Object>> get grouppedTransactionValues {
  return List.generate(7, (index) {
    final weekDay = DateTime.now().subtract(Duration(days: index));
    double totalSum = 0;
    for (var i = 0; i < transactions.length; i++) {
      if (transactions[i].date.day == weekDay.day &&
          transactions[i].date.month == weekDay.month &&
          transactions[i].date.year == weekDay.year) {
        totalSum += transactions[i].amount;
      }
    }
    return {
      'day': DateFormat.E().format(weekDay).substring(0, 1),
      'amount': totalSum,
    };
  });
}

double get totalSpending {
  return grouppedTransactionValues.fold(0.0, (sum, item) {
    return sum + (item['amount'] as double);
  });
}

bool _showChart = false;

@override
Widget build(BuildContext context) {
  final mediaQ= MediaQuery.of(context);
  final isLandscape= mediaQ.orientation == Orientation.landscape;
  return Scaffold(
    appBar: AppBar(
      title: Text('Flutter App'),
      actions: [
        IconButton(
          icon: Icon(Icons.add),
          onPressed: () {
            _startAddNewTransaction(context);
          },
        ),
      ],
    ),
    floatingActionButtonLocation: FloatingActionButtonLocation.centerFloat,
    floatingActionButton: Platform.isIOS ? Container() : FloatingActionButton(
      child: Icon(Icons.add),
      onPressed: () {
        _startAddNewTransaction(context);
      },
    ),
    body: SingleChildScrollView(
      child: Column(
        mainAxisAlignment: MainAxisAlignment.start,
        crossAxisAlignment: CrossAxisAlignment.stretch,
        children: [
          //switch
          if (isLandscape) Row(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              Text('Show chart?'),
              Switch.adaptive(
                value: _showChart,
                onChanged: (val) {
                  setState(() {
                    _showChart = val;
    });
   },
  ),
 ],
),

if ((isLandscape && _showChart) || (!isLandscape))
//chart
 Container(
  height: mediaQ.size.height * 0.3,
  child: Card(
   child: Container(
    child: LayoutBuilder(builder: (ctx, constraints) {
     return Row(
      children: grouppedTransactionValues.map((data) {
       final spendingPctOfTotal = totalSpending == 0.0
         ? 0.0
         : (data['amount'] as double) / totalSpending;
       //return Text(data['day'].toString() + ' ' + data['amount'].toString());
       return Flexible(
        fit: FlexFit.tight,
        child: Column(
         children: <Widget>[
          FittedBox(
           child: Text('\$${data['amount']}'),
          ),
          SizedBox(
           height: 4,
          ),
          Container(
           height: constraints.maxHeight * 0.7,
           width: 10,
           child: Stack(
            children: <Widget>[
             Container(
              decoration: BoxDecoration(
               border: Border.all(
                 color: Colors.grey,
                 width: 1.0),
               color: Color.fromRGBO(
                 220, 220, 220, 1),
               borderRadius:
               BorderRadius.circular(10),
              ),
             ),
             FractionallySizedBox(
              heightFactor: spendingPctOfTotal,
              child: Container(
               decoration: BoxDecoration(
                color: Theme.of(context)
                  .primaryColor,
                borderRadius:
                BorderRadius.circular(10),
               ),
              ),
             ),
            ],
           ),
          ),
          SizedBox(
           height: 4,
          ),
          Text(data['day'].toString()),
               ],
              ),
             );
            }).toList(),
           );
          }),
          width: double.infinity,
          color: Colors.blueGrey,
         ),
         elevation: 5,
         margin: EdgeInsets.all(20),
        ),
       ),
      if ((isLandscape && !_showChart) || (!isLandscape))
      //list
       Column(
        children: [
         Container(
          height: mediaQ.size.height * 0.6,
          child: transactions.isEmpty
            ? Column(
           children: [
            Text('No transacations yet'),
            SizedBox(
             height: 20,
            ),
            Image.network(
             'https://static.thenounproject.com/png/194055-200.png',
             fit: BoxFit.cover,
            ),
           ],
          )
            : ListView.builder(
           itemCount: transactions.length,
           itemBuilder: (bc, index) {
            return Card(
             child: Row(
              children: [
               Container(
                margin: EdgeInsets.symmetric(
                  vertical: 10, horizontal: 20),
                decoration: BoxDecoration(
                  border: Border.all(
                   color: Colors.black,
                   width: 2,
                  )),
                padding: EdgeInsets.all(10),
                child: Text(
                 '\$' +
                   transactions[index]
                     .amount
                     .toStringAsFixed(2),
                 style: TextStyle(
                  fontWeight: FontWeight.bold,
                  fontSize: 20,
                  color: Theme.of(context)
                    .primaryColorDark,
                 ),
                ),
               ),
               Column(
                crossAxisAlignment:
                CrossAxisAlignment.start,
                children: [
                 Text(
                  transactions[index].title,
                  style: TextStyle(
                   fontSize: 16,
                   fontWeight: FontWeight.bold,
                  ),
                 ),
                 Text(
                  DateFormat.yMMMd().format(
                    transactions[index].date),
                  style: TextStyle(
                   color: Colors.grey,
                  ),
                 ),
                ], //column children
               ), //column
              ], //row children
             ), //row
            ); //card
           }, //itemBuilder
          ), //listview builder
         ), //container of the list view
        ], //colum children
       ), //column,
     ], //column children
    ), //column
   ), //SingleChildScrollView
  ); //scaffold
 }
}

class Transaction {
 final String id;
 final String title;
 final double amount;
 final DateTime date;
 Transaction(
   {required this.id,
    required this.title,
    required this.amount,
    required this.date});
}
```

References

- https://www.udemy.com/course/learn-flutter-dart-to-build-ios-android-apps/