DEPARTMENT OF SOFTWARE ENGINEERING MOBILE APPLICATION DEVELOPMENT (SE-487)

**LAB SESSION 10**

**List View, Box decoration & Expanded Widgets**

**Objective:**

To create a Flutter application that demonstrates the usage of ListView, BoxDecoration, and Expanded widget to display a list of items with customized styling.

**Introduction:**

**ListView:**

**ListView** is a commonly used widget in Flutter for displaying a scrollable list of widgets. It can handle a large number of items efficiently and provides multiple constructors for different use cases, such as ListView.builder, ListView.separated, and ListView.custom.List view is scroll enabled.it provides vertical and horizontal scroll direction as well.

**Key Constructors:**

∙ **ListView.builder**: Used when you have a large or infinite list of items, and you want to create widgets only when they are visible.

∙ **ListView.separated**: Similar to ListView.builder but also allows you to specify a separator widget between items.

∙ **ListView.custom**: Provides the most flexibility by allowing you to define a custom list with custom logic.

**Usage Tips:**

∙ Use ListView.builder for large lists to ensure performance efficiency.

∙ Combine with other widgets like Padding and Card to customize the appearance of list items.

∙ Utilize scroll physics to control the scrolling behavior (e.g., NeverScrollableScrollPhysics to disable scrolling). 

∙ import 'package:flutter/cupertino.dart';

import 'package:flutter/material.dart';

void main() {

runApp(const MyApp());

}

class MyApp extends StatelessWidget {

const MyApp({super.key});

@override

Widget build(BuildContext context) {

return MaterialApp(

DEPARTMENT OF SOFTWARE ENGINEERING MOBILE APPLICATION DEVELOPMENT (SE-487)

title:'flutter demo',

debugShowCheckedModeBanner: false,

theme:ThemeData(

primarySwatch:Colors.*lightBlue*,

),

home:const MyHomePage(),

);

}

}

class MyHomePage extends StatefulWidget {

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

@override

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

}

class \_MyHomePageState extends State<MyHomePage>

{

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

backgroundColor: Colors.*orange*,

title: Text('ink well widget',style: TextStyle(fontSize: 26,color:Colors.*white*),),

),

body: ListView(

children: [

Text(

'1',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

),

Text(

'2',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

),

Text(

'3',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

),

Text(

'4',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

),

Text(

'5',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

DEPARTMENT OF SOFTWARE ENGINEERING MOBILE APPLICATION DEVELOPMENT (SE-487)

),

Text(

'6',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

)

],

));

}

}

Steps to implement a horizontal list:

body: ListView(

scrollDirection: Axis.horizontal,

children: [

Padding(

padding: const EdgeInsets.all(8.0),

child: Text(

'1',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

),

),

Padding(

padding: const EdgeInsets.all(8.0),

child: Text(

'2',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

),

),

Padding(

padding: const EdgeInsets.all(8.0),

child: Text(

'3',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

),

),

Padding(

padding: const EdgeInsets.all(8.0),

child: Text(

'4',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

),

),

Padding(

DEPARTMENT OF SOFTWARE ENGINEERING MOBILE APPLICATION DEVELOPMENT (SE-487)

padding: const EdgeInsets.all(8.0),

child: Text(

'5',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

),

),

Padding(

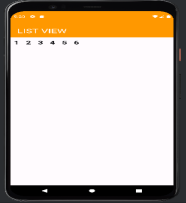
padding: const EdgeInsets.all(8.0),

child: Text(

'6',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

),

) 

],

));

}

}

Steps to reverse a list:

body: ListView(

scrollDirection: Axis.horizontal,

reverse: true,

children: [

Padding(

padding: const EdgeInsets.all(8.0),

child: Text(

'1',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

),

),

Padding(

padding: const EdgeInsets.all(8.0),

child: Text(

'2',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

),

),

DEPARTMENT OF SOFTWARE ENGINEERING MOBILE APPLICATION DEVELOPMENT (SE-487)

Padding(

padding: const EdgeInsets.all(8.0),

child: Text(

'3',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

),

),

Padding(

padding: const EdgeInsets.all(8.0),

child: Text(

'4',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

),

),

Padding(

padding: const EdgeInsets.all(8.0),

child: Text(

'5',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

),

),

Padding(

padding: const EdgeInsets.all(8.0),

child: Text(

'6',

style: TextStyle(fontSize: 22, fontWeight: FontWeight.*bold*),

),

)

],

));

}

}

DEPARTMENT OF SOFTWARE ENGINEERING MOBILE APPLICATION DEVELOPMENT (SE-487) Steps to implement listview.builder class:

body:

ListView.builder(itemBuilder: (context, index) {

return Text('one',

style: TextStyle(fontWeight: FontWeight.*bold*, fontSize: 21));

},

itemCount: 6,

)

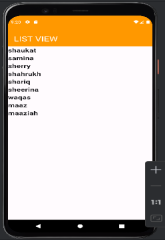
);

}

}

Fetching data from an array

DEPARTMENT OF SOFTWARE ENGINEERING MOBILE APPLICATION DEVELOPMENT (SE-487)

), 

body:

ListView.builder(itemBuilder: (context, index) {

return Text(arrnames[index],

style: TextStyle(fontWeight: FontWeight.*bold*, fontSize: 21));

},

itemCount: arrnames.length,

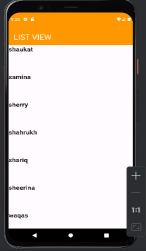
)

);

}

}

Steps required in Customizing a list:



body:

ListView.builder(itemBuilder: (context, index) {

return Text(arrnames[index],

style: TextStyle(fontWeight: FontWeight.*bold*, fontSize: 21));

},

itemCount: arrnames.length,

itemExtent: 100,

)

);

}

}

DEPARTMENT OF SOFTWARE ENGINEERING MOBILE APPLICATION DEVELOPMENT (SE-487)

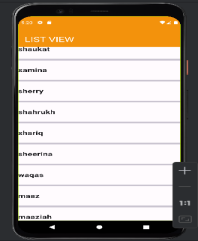
Implementation of listview.separated class:

body:

ListView.separated(itemBuilder: (context, index) {

return Text(arrnames[index],

style: TextStyle(fontWeight: FontWeight.*bold*, fontSize: 21));

}, 

itemCount: arrnames.length,

separatorBuilder: (context,index){

return Divider(height:50,thickness: 6);

},

)

);

}

}

body:

ListView.separated(itemBuilder: (context, index) {

return Row(

children: [

Padding(

padding: const EdgeInsets.all(8.0),

child: Text(arrnames[index],

style: TextStyle(fontWeight: FontWeight.*bold*, fontSize: 21)), ),

Column(

children: [

Padding(

padding: const EdgeInsets.all(8.0),

child: Text(arrnames[index],

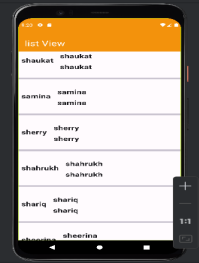
style: TextStyle(fontWeight: FontWeight.*bold*, fontSize: 21)), ),

Text(arrnames[index],

style: TextStyle(fontWeight: FontWeight.*bold*, fontSize: 21)), ],

),

DEPARTMENT OF SOFTWARE ENGINEERING MOBILE APPLICATION DEVELOPMENT (SE-487)

], 

);

},

itemCount: arrnames.length,

separatorBuilder: (context,index){

return Divider(height:50,thickness: 6);

},

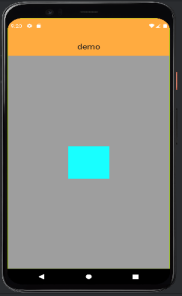
)

);

}

}

**BoxDecoration** is used to paint a box behind a widget. It provides a variety of properties for customizing the appearance of a Container widget, such as background color, border, border radius, gradients, and shadows.

**Key Properties:** 

∙ **color**: The background color of the box.

∙ **borderRadius**: The radius of the box's corners.

∙ **boxShadow**: A list of shadows cast by the box.

∙ **gradient**: A gradient to use for the box's background.

∙ **border**: A border to draw around the box.

body:Container(

width: double.*infinity*,

height: double.*infinity*,

color: Colors.*grey*,

child: Center(

child: Container(

width: 100,

height: 100,

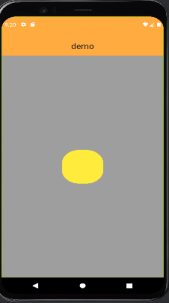
color:Colors.*cyanAccent*,

DEPARTMENT OF SOFTWARE ENGINEERING MOBILE APPLICATION DEVELOPMENT (SE-487)

),

),

),

body:Container( 

width: double.*infinity*,

height: double.*infinity*,

color: Colors.*grey*,

child: Center(

child: Container(

width: 100,

height: 100,

//color:Colors.cyanAccent,

decoration: BoxDecoration(

color:Colors.*yellow*,

borderRadius:BorderRadius.circular(40)

),

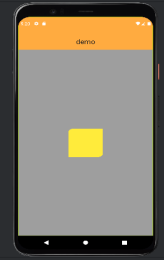
),

),

),

body:Container(

width: double.*infinity*,

height: double.*infinity*, 

color: Colors.*grey*,

child: Center(

child: Container(

width: 100,

height: 100,

decoration: BoxDecoration(

color:Colors.*yellow*,

borderRadius:BorderRadius.only(topLeft:Radius.circular(12),

bottomRight:Radius.circular(12))

),

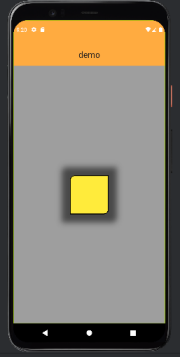
),

),

),

DEPARTMENT OF SOFTWARE ENGINEERING MOBILE APPLICATION DEVELOPMENT (SE-487)

body:Container(

width: double.*infinity*, 

height: double.*infinity*,

color: Colors.*grey*,

child: Center(

child: Container(

width: 100,

height: 100,

//color:Colors.cyanAccent,

decoration: BoxDecoration(

color:Colors.*yellow*,

borderRadius:BorderRadius.only(topLeft:Radius.circular(12),

bottomRight:Radius.circular(12)),

border:Border.all(

width: 2,

color: Colors.*black*,

),

boxShadow: [

BoxShadow(

blurRadius:12,

spreadRadius: 21,

color: Colors.*black54*,

)

]

),

),

),

),

body:Container(

width: double.infinity,

height: double.infinity,

color: Colors.grey,

child: Center(

child: Container(

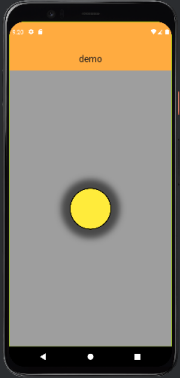
width: 100,

height: 100,

decoration: BoxDecoration(

color:Colors.yellow,

DEPARTMENT OF SOFTWARE ENGINEERING MOBILE APPLICATION DEVELOPMENT (SE-487)



border:Border.all(

width: 2,

color: Colors.black,

),

boxShadow: [

BoxShadow(

blurRadius:12,

spreadRadius: 21,

color: Colors.black54,

),

],

shape:BoxShape.circle,

),

),

),

),

);

}

}

**Expanded**

Expanded is a widget that expands a child of a Row, Column, or Flex to fill the available space. It helps in creating responsive layouts by distributing available space among children.

**Key Properties:**

flex: An integer value that determines how much space the child should take relative to other children.

**body:Row(**

**//mainAxisAlignment:MainAxisAlignment.center,**

**children: [**

**Expanded(**

**child: Container(**

**height:100,**

**width:50,**

DEPARTMENT OF SOFTWARE ENGINEERING MOBILE APPLICATION DEVELOPMENT (SE-487)

**color: Colors.blue,**

**),**

**),**

**Expanded(** 

**child: Container(**

**height: 100,**

**width: 50,**

**color: Colors.pink,**

**),**

**),**

**Expanded(**

**child: Container(**

**height: 100,**

**width: 50,**

**color: Colors.yellow,**

**),**

**),**

**Expanded(**

**child: Container(**

**height: 100,**

**width: 50,**

**color: Colors.green,**

**),**

**)**

**],**

**),**

**body:Column(**

**children: [**

**Expanded(**

**flex:2,**

**child: Container(**

**height:100,**

**color: Colors.blue,**

**),**

**),**

**Expanded(**

**flex:4,**

**child: Container(**

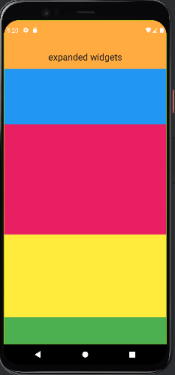
**height: 100,**

**color: Colors.pink,**

DEPARTMENT OF SOFTWARE ENGINEERING MOBILE APPLICATION DEVELOPMENT (SE-487)

**),**

**),**

**Expanded(** 

**flex:3,**

**child: Container(**

**height: 100,**

**color: Colors.yellow,**

**),**

**),**

**Expanded(**

**flex:1,**

**child: Container(**

**height: 100,**

**//width: 50,**

**color: Colors.green,**

**),**

**)**

**],**

**),**

DEPARTMENT OF SOFTWARE ENGINEERING MOBILE APPLICATION DEVELOPMENT (SE-487)

**Exercise**

Create a ListView displaying a list of items where each item contains text and an icon, with the text taking up the remaining space using the Expanded widget. Use a ListView to display a list of items. Each item should be a Row with an Icon and Text widget. Use Expanded to make the text take up the remaining space.

**Code**

//SE-21031 Musadique Hussain

//SE-21045 Muhammad Asim

import 'package:flutter/material.dart';

void main() {

  runApp(MyApp());

}

class MyApp extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return MaterialApp(

      debugShowCheckedModeBanner: false,

      home: Scaffold(

        appBar: AppBar(

          title: Text('ListView with Expaned Example'),

        ),

        body: MyListView(),

      ),

    );

  }

}

class MyListView extends StatelessWidget {

  final List<Map<String, dynamic>> items = [

    {'icon': Icons.arrow\_right, 'text': 'Item 1'},

    {'icon': Icons.arrow\_right, 'text': 'Item 2'},

    {'icon': Icons.arrow\_right, 'text': 'Item 3'},

    {'icon': Icons.arrow\_right, 'text': 'Item 4'},

    {'icon': Icons.arrow\_right, 'text': 'Item 5'},

  ];

  @override

  Widget build(BuildContext context) {

    return ListView.builder(

      itemCount: items.length,

      itemBuilder: (context, index) {

        return Padding(

          padding: const EdgeInsets.all(8.0),

          child: Row(

            children: [

              Icon(items[index]['icon']),

              SizedBox(width: 10),

              Expanded(

                child: Text(items[index]['text']),

              ),

            ],

          ),

        );

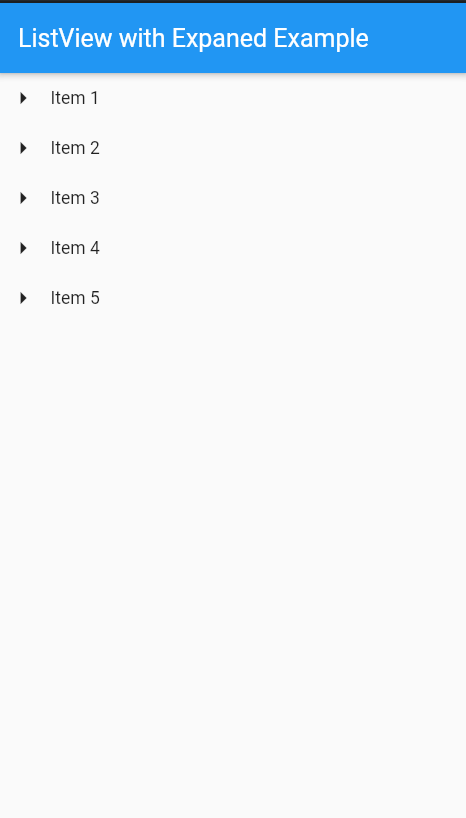
      },

    );

  }

}

**Output**



Create a listView.seperated to display a list of items with custom styling using Box Decoration. Use ListView.separated to display a list of items. Each item should be a container with a Box Decoration that includes a background color and rounded corners. Add a separator between items.

Code

//SE-21031 Musadique Hussain

//SE-21045 Muhammad Asim

import 'package:flutter/material.dart';

void main() {

  runApp(MyApp());

}

class MyApp extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return MaterialApp(

      debugShowCheckedModeBanner: false,

      home: Scaffold(

        appBar: AppBar(

          title: Text('ListView.separated Example'),

        ),

        body: MyListView(),

      ),

    );

  }

}

class MyListView extends StatelessWidget {

  final List<String> arrnames = ['Item 1', 'Item 2', 'Item 3', 'Item 4', 'Item 5'];

  @override

  Widget build(BuildContext context) {

    return ListView.separated(

      itemBuilder: (context, index) {

        return Container(

          margin: const EdgeInsets.all(8.0),

          padding: const EdgeInsets.all(16.0),

          decoration: BoxDecoration(

            color: Colors.blueAccent,

            borderRadius: BorderRadius.circular(12),

            boxShadow: [

              BoxShadow(

                color: Colors.grey.withOpacity(0.5),

                spreadRadius: 5,

                blurRadius: 7,

                offset: Offset(0, 3), // changes position of shadow

              ),

            ],

          ),

          child: Text(

            arrnames[index],

            style: TextStyle(fontWeight: FontWeight.bold, fontSize: 21, color: Colors.white),

          ),

        );

      },

      itemCount: arrnames.length,

      separatorBuilder: (context, index) {

        return Divider(height: 50, thickness: 6);

      },

    );

  }

}

**Output**

