Flutter: Lists and List View Widget

An indexable collection of objects with a length.

Subclasses of this class implement different kinds of lists. The most common kinds of lists are:

- Fixed-length list. An error occurs when attempting to use operations that can change the length of the list.
- Growable list. Full implementation of the API defined in this class.

```
var fixedLengthList = List<int>.filled(5, 0);
fixedLengthList.length = 0;  // Error
fixedLengthList.add(499);    // Error
fixedLengthList[0] = 87;
var growableList = [1, 2];
growableList.length = 0;
growableList.add(499);
growableList[0] = 87;
```

Lists are the most popular elements of every web or mobile application. They are made up of multiple rows of items, which include text, buttons, toggles, icons, thumbnails, and many more. We can use it for displaying various information such as menus, tabs, or to break the monotony of pure text files.

In this section, we are going to learn how we can work with Lists in the Flutter. Flutter allows you to work with Lists in different ways, which are given below:

- Basic Lists
- Long Lists
- Grid Lists
- Horizontal Lists

Let us see all the above lists one by one.

Basic Lists:

- Flutter includes a **ListView** widget for working with Lists, which is the fundamental concept of displaying data in the mobile apps. The ListView is a perfect standard for displaying lists that contains only a few items. ListView also includes **ListTitle** widget, which gives more properties for the visual structure to a list of data.

The following example displays a basic list in the Flutter application.

```
body: ListView(
 children: <Widget>[
  ListTile(
   leading: Icon(Icons.map),
   title: Text('Map'),
  ),
  ListTile(
   leading: Icon(Icons.photo_album),
   title: Text('Album'),
  ),
  ListTile(
   leading: Icon(Icons.phone),
   title: Text('Phone'),
  ),
  ListTile(
   leading: Icon(Icons.contacts),
   title: Text('Contact'),
  ),
  ListTile(
   leading: Icon(Icons.settings),
   title: Text('Setting'),
  ),
```

Basic lists (Static lists)-> all the elements are created once. Used when you have limited option or short list.

Long Lists:

- Sometimes you want to display a very long list in a single screen of your app, in that case, the above method for displaying the lists is not perfect.

- To work with a list that contains a very large number of items.

- Use **ListView.builder()** constructor.

- Use **ListView.seperated()** constructor in-order to generate separated listview using divider.

- ListView.builder creates items when they are scrolled onto the screen. Unlike, ListView which builds all at once. So, it is memory efficient.

  GitHub link: https://github.com/samsunk/list_view.git

```
body: ListView.builder(
 itemCount: products.length,
 itemBuilder: (context, index) {
  return ListTile(
   title: Text('${products[index]}'),
  );
```

Grid lists:

Sometimes we want to display the items in a grid layout rather than the normal list that comes one after next. A **GridView** widget allows you to create a grid list in Flutter. The simplest way to create a grid is by using the **GridView.count()** constructor, which specifies the number of rows and columns in a grid.

GitHub link: https://github.com/samsunk/grid_view.git

```dart
void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);
  final _title = 'Grid view example';
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text(_title),
        ), // AppBar
        body: GridView.count(
          crossAxisCount: 3,
          children: List.generate(100, (index) {
            return Center(
              child: Container(
                width: 200,
                height: 100,
                decoration: BoxDecoration(
                  color: Colors.teal,
                  border: Border.all(width: 10, color: Colors.black),
                ), // BoxDecoration
                child: Text(
                  'Item $index',
                  style: Theme.of(context).textTheme.headline5,
                ), // Text
              ), // Container
            ); // Center
          }), // List.generate
        ), // GridView.count
      ), // Scaffold
    ); // MaterialApp
  }
}

```

Horizontal List:

- The listview widget supports horizontal lists.
- Use the standard ListView constructor, passing in a horizontal ScrollDirection.

```dart
body: Container(
  margin: EdgeInsets.symmetric(vertical: 25.0),
  height: 150.0,
  child: ListView(
    scrollDirection: Axis.horizontal,
    children: <Widget>[
      Container(
        width: 150.0,
        color: Colors.blue,
        child: new Stack(
          children: <Widget>[
            ListTile(
              leading: Icon(Icons.home),
              title: Text('Home'),
            ),
          ],
        ),
      ),
```