Scrollviews in Flutter

<u>SingleChildScrollView</u>

A box in which a single widget can be scrolled.A box in which a single widget can be scrolled.

For more: https://api.flutter.dev/flutter/widgets/SingleChildScrollView-class.html

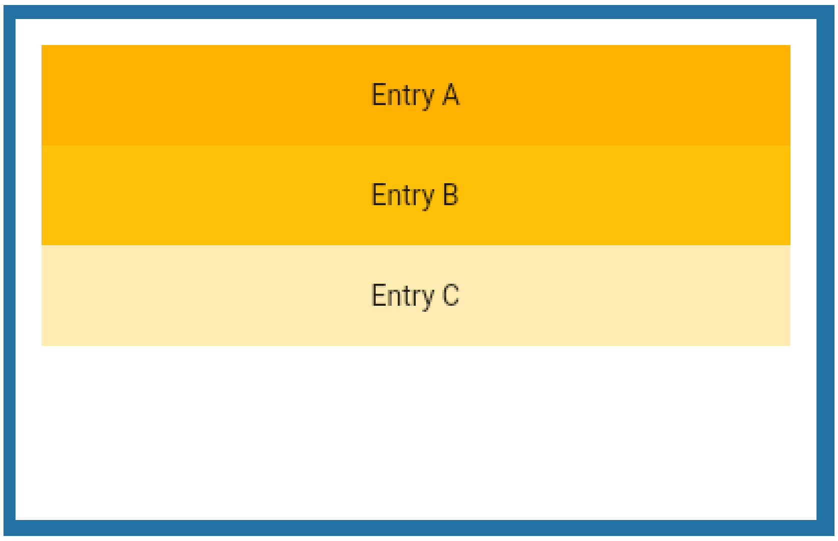
<u>ListView</u>

ListView is the most commonly used scrolling widget. It displays its children one after another in the scroll direction.

Types of listviews

- The default constructor takes an explicit List<Widget> of children. This constructor is appropriate for list views with a small number of children because constructing the List requires doing work for every child that could possibly be displayed in the list view instead of just those children that are actually visible.

```
ListView(
  padding: const EdgeInsets.all(8),
  children: <Widget>[
    Container(
      height: 50,
      color: Colors.amber[600],
      child: const Center(child: Text('Entry A')),
    ),
    Container(
      height: 50,
      color: Colors.amber[500],
      child: const Center(child: Text('Entry B')),
    ),
    Container(
      height: 50,
      color: Colors.amber[100],
      child: const Center(child: Text('Entry C')),
    ),
  ],
)
```

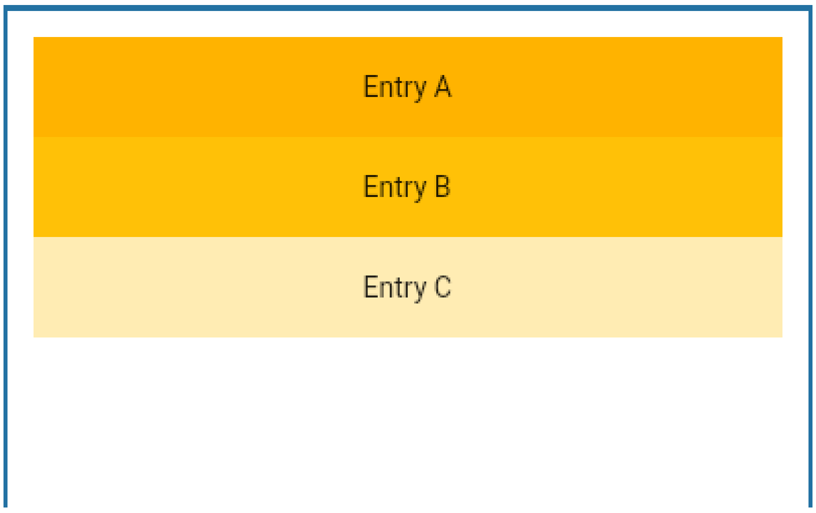


- The ListView.builder constructor takes an IndexedWidgetBuilder, which builds the children on demand. This constructor is appropriate for list views with a large (or infinite) number of children because the builder is called only for those children that are actually visible.

```
final List<String> entries = <String>['A', 'B', 'C'];
final List<int> colorCodes = <int>[600, 500, 100];

ListView.builder(
  padding: const EdgeInsets.all(8),
  itemCount: entries.length,
  itemBuilder: (BuildContext context, int index) {
    return Container(
      height: 50,
      color: Colors.amber[colorCodes[index]],
      child: Center(child: Text('Entry ${entries[index]}')),
    );
  }
);
```

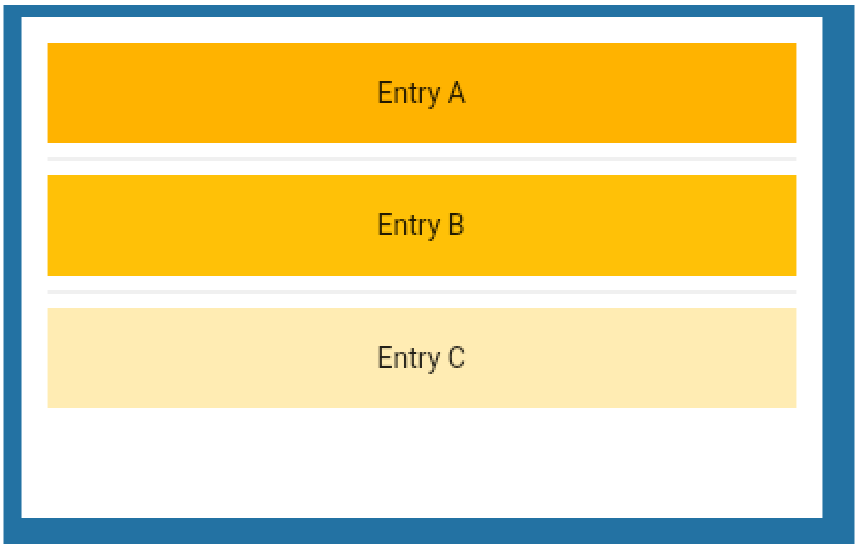
-



- The ListView.separated constructor takes two IndexedWidgetBuilders: itemBuilder builds child items on demand, and separatorBuilder similarly builds separator children which appear in between the child items. This constructor is appropriate for list views with a fixed number of children.

```
final List<String> entries = <String>['A', 'B', 'C'];
final List<int> colorCodes = <int>[600, 500, 100];

ListView.separated(
  padding: const EdgeInsets.all(8),
  itemCount: entries.length,
  itemBuilder: (BuildContext context, int index) {
    return Container(
      height: 50,
      color: Colors.amber[colorCodes[index]],
      child: Center(child: Text('Entry ${entries[index]}')),
    );
  },
  separatorBuilder: (BuildContext context, int index) => const Divider(),
);
```



For more: https://api.flutter.dev/flutter/widgets/ListView-class.html

Navigate to another Screen and Back

use the Navigator.push() method. The push() method adds a Route to the stack of routes managed by the Navigator. Where does the Route come from? You can create your own, or use a MaterialPageRoute, which is useful because it transitions to the new route using a platform-specific animation.

```
onPressed: () {
  Navigator.push(
    context,
    MaterialPageRoute(builder: (context) => SecondRoute()),
  );
}
```

Return to the first route Return to the first route

```
onPressed: () {
  Navigator.pop(context);
}
```

Geo J Vallavancottu

+91 8281782564

geo.j@sjcetpalai.ac.in