



FLUTTER

FLUTTER #6

Copyright © 2023 Accenture. All rights reserved.

FLUTTER



FLUTTER WIDGETS

PARTE 2



FLUTTER



FLUTTER

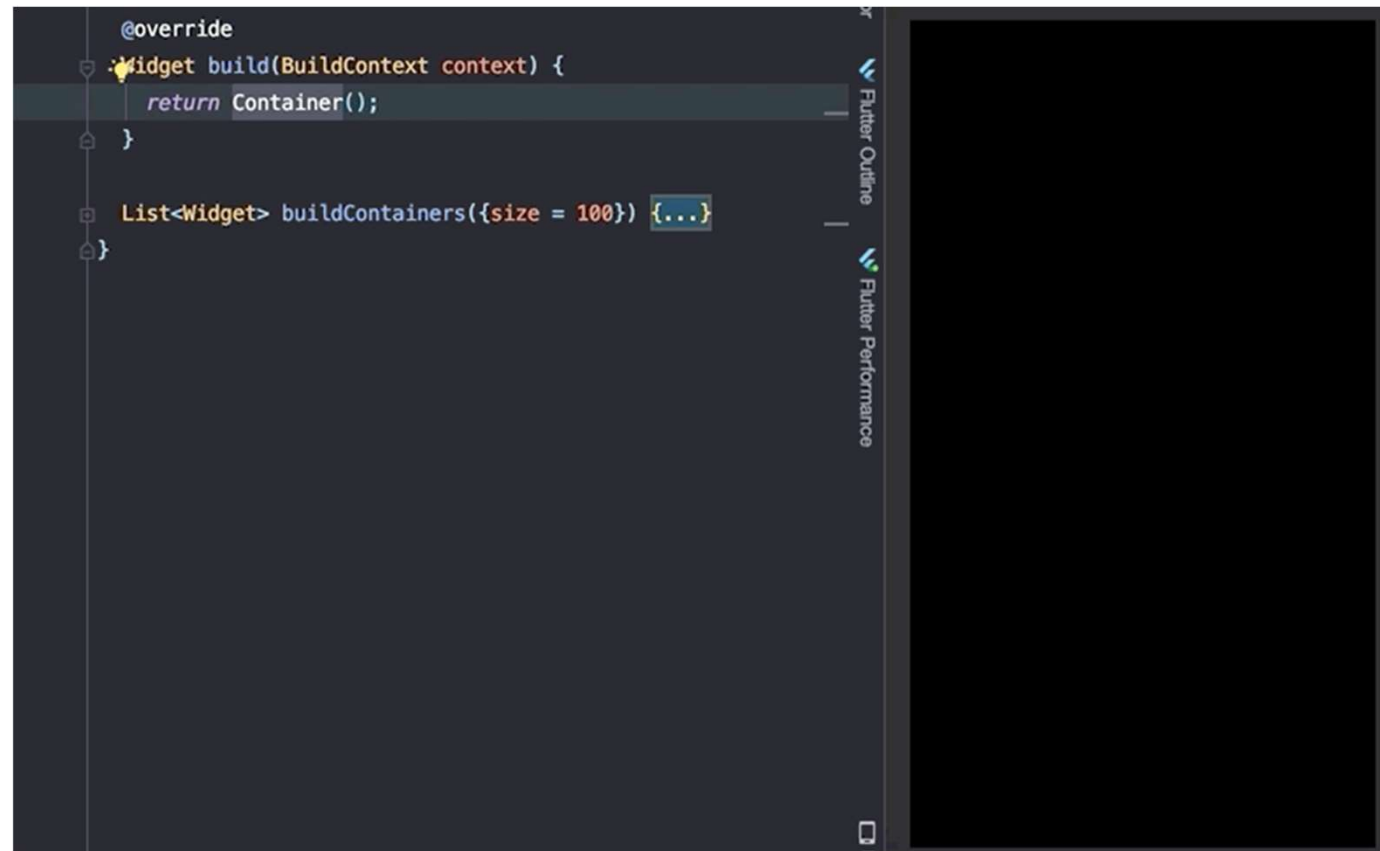


Listview

O Widget **ListView** serve para mostrarmos listas e podemos utilizar o poder do scroll.

Ele é bem parecido com o **Column** ou a **Row**, porém precisa ter um tamanho delimitado, já que esse Widget irá tentar expandir ao máximo na tela para delimitar o início e fim do scroll.

O Flutter automaticamente configura as ações de chegar ao topo ou ao final da lista de acordo com o aparelho.



FLUTTER



Listview

Podemos utilizar a **ListView** na horizontal, fazendo um efeito parecido com um carousel.



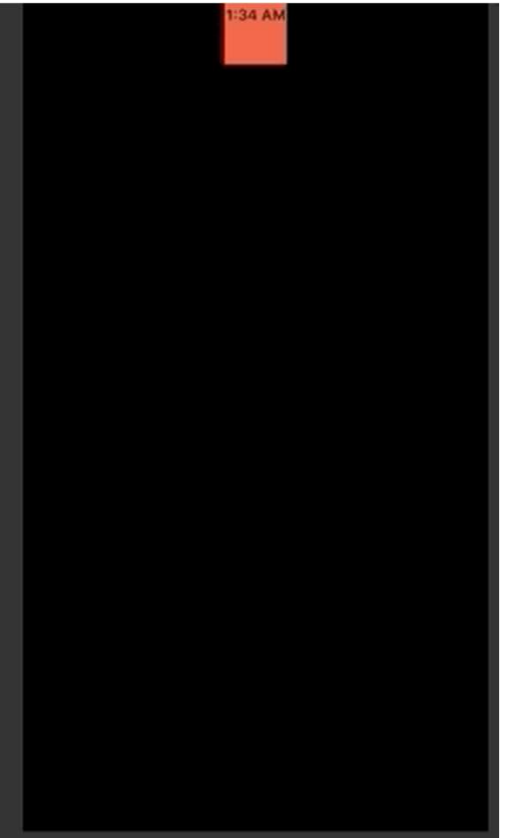
FLUTTER



Expanded

Para conseguir uma **ListView** dentro de uma **Column**, precisamos de um Widget chamado **Expanded**, que expande ao máximo que puder e passa o limite para seu Child fazendo o seguinte efeito:

```
return Column(  
  children: <Widget>[  
    Container(height: 50, width: 50, color: Colors.red),  
    ListView(  
      children: buildContainers(),  
    ), // ListView  
  ], // <Widget>[]  
); // Column  
  
List<Widget> buildContainers({size = 100}) {...}
```



FLUTTER



Expanded

```
Row(  
  crossAxisAlignment: CrossAxisAlignment.center,  
  children: [  
    Expanded(  
      child: Image.asset('images/pic1.jpg'),  
    ),  
    Expanded(  
      child: Image.asset('images/pic2.jpg'),  
    ),  
    Expanded(  
      child: Image.asset('images/pic3.jpg'),  
    ),  
  ],  
);
```

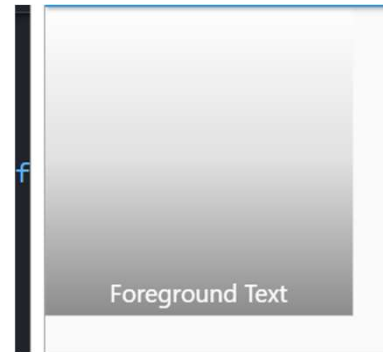
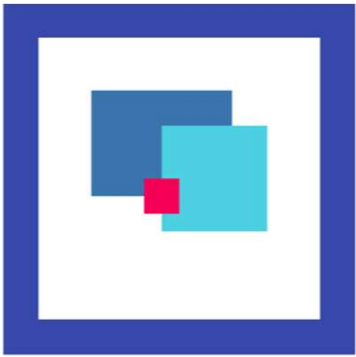


FLUTTER

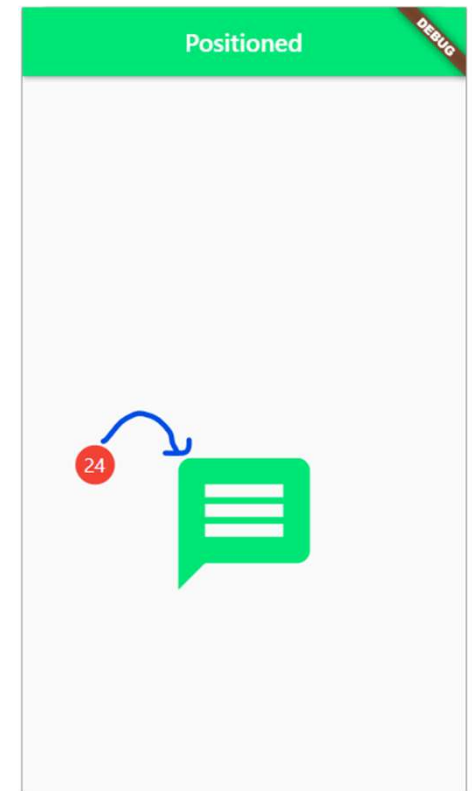


Stack

Usando uma **stack** você pode posicionar widgets uns sobre os outros



Esta classe é útil se você deseja sobrepor vários filhos de uma forma simples, por exemplo tendo algum texto e uma imagem, sobrepostos com...



<https://docs.flutter.dev/ui/widgets/layout>



FLUTTER



Table

Usando uma **table** você pode posicionar widgets como se fossem uma listview porém sem scroll

```
child: Table(  
  children: [  
    TableRow(children: [...]), // TableRow  
    TableRow(children: [...]), // TableRow  
    TableRow(children: [...]), // TableRow  
    TableRow(children: [...]), // TableRow  
  ],  
)
```

GeeksforGeeks		
Table		
Education	Institution	University
B.Tech	ABESEC	AKTU
12th	Delhi Public School	CBSE
High School	SFS	ICSE





Data Table

Usando uma **datatable** você posiciona os widgets em colunas de dados e linhas de dados. O dataTable dimensiona as colunas automaticamente.

```
25 @override
26 Widget build(BuildContext context) {
27   return DataTable(
28 >     columns: const <DataColumn>[ ...
47   ], // <DataColumn>[]
48 >     rows: const <DataRow>[ ...
70   ], // <DataRow>[]
71   ); // DataTable
72 }
```

main.dart M x

```
27 return DataTable(
28   columns: const <DataColumn>[
29 >     DataColumn( ...
34   ), // DataColumn
35 >     DataColumn( ...
40   ), // DataColumn
41 >     DataColumn( ...
46   ), // DataColumn
47   ], // <DataColumn>[]
48   rows: const <DataRow>[
49 >     DataRow( ...
55   ), // DataRow
56 >     DataRow( ...
62   ), // DataRow
63 >     DataRow( ...
69   ), // DataRow
70   ], // <DataRow>[]
71   ); // DataTable
```

Flutter Code Sample

Name	Age	Designation
Mohit	23	Associate Software Developer
Akshay	25	Software Developer
Deepak	29	Team Lead



Paginated Data Table

Usando uma **PaginatedDatatable** você pagina os todos os dados. Para isso é preciso definir a fonte de dados.

```
25 @override
26 Widget build(BuildContext context) {
27   return DataTable(
28     > columns: const <DataColumn>[ ...
47   ], // <DataColumn>[]
48   > rows: const <DataRow>[ ...
70   ], // <DataRow>[]
71   ); // DataTable
72 }
```

main.dart M x

```
27 return DataTable(
28   columns: const <DataColumn>[
29     > DataColumn( ...
34   ), // DataColumn
35     > DataColumn( ...
40   ), // DataColumn
41     > DataColumn( ...
46   ), // DataColumn
47   ], // <DataColumn>[]
48   rows: const <DataRow>[
49     > DataRow( ...
55   ), // DataRow
56     > DataRow( ...
62   ), // DataRow
63     > DataRow( ...
69   ), // DataRow
70   ], // <DataRow>[]
71   ); // DataTable
```

Flutter Code Sample

Name	Age	Designation
Mohit	23	Associate Software Developer
Akshay	25	Software Developer
Deepak	29	Team Lead



Paginated Data Table

Definindo a fonte de dados.

```
main.dart M Main Menu
66 class MyData extends DataTableSource {
67   // Generate some made-up data
68 > final List<Map<String, dynamic>> _data = List.generate(...
76   @override
77   bool get isRowCountApproximate => false;
78   @override
79   int get rowCount => _data.length;
80   @override
81   int get selectedRowCount => 0;
82   @override
83   DataRow getRow(int index) {
84     return DataRow(cells: [
85       DataCell(Text(_data[index]['id'].toString())),
86       DataCell(Text(_data[index]["title"])),
87       DataCell(Text(_data[index]["price"].toString())),
88     ]); // DataRow
89   }
```

Kindacode.com

My Products

ID	Name	Price
40	Item 40	6984
41	Item 41	2317
42	Item 42	3075
43	Item 43	5838
44	Item 44	392
45	Item 45	7422
46	Item 46	5910
47	Item 47	7213

11:48 AM 2023

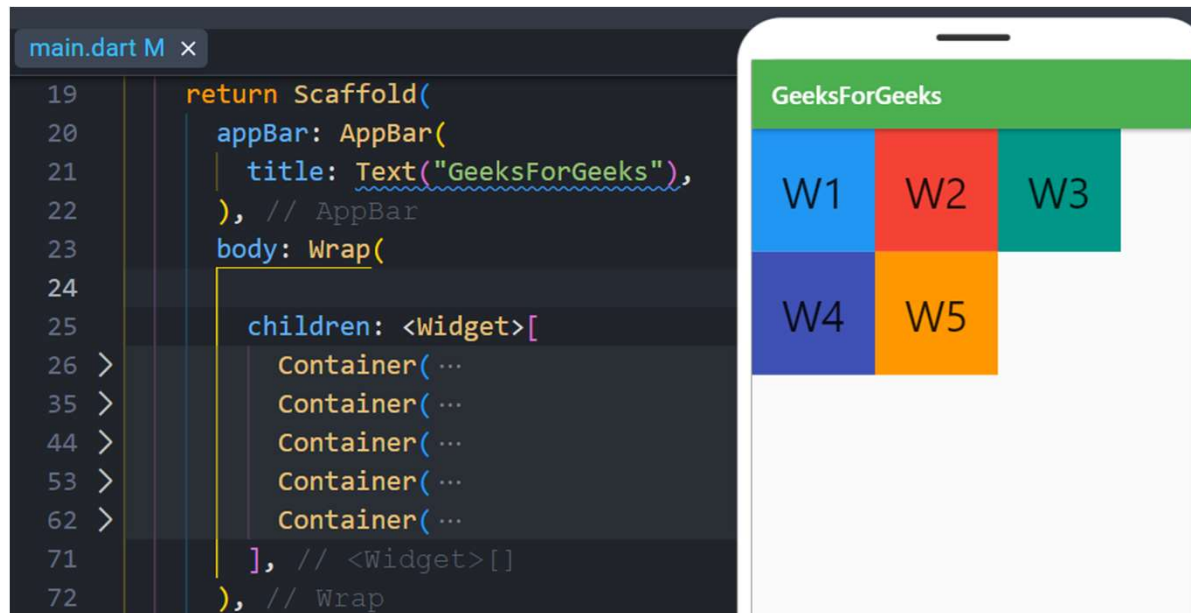


FLUTTER



WRAP

Um **Wrap** dispõe cada child e tenta colocar o child adjacente ao child anterior no eixo principal, deixando espaço entre eles. Se não houver espaço suficiente para acomodar o child, **Wrap** criará um novo trecho adjacente aos children existentes no eixo.



FLUTTER



FloatingActionButton

Um **floating action button (FAB)** executa a ação principal ou mais comum em uma tela. Ele aparece na frente de todo o conteúdo da tela, normalmente como uma forma circular com um ícone no centro.



(FAB) são mais comumente usados no Scaffold.

(FAB) é único.

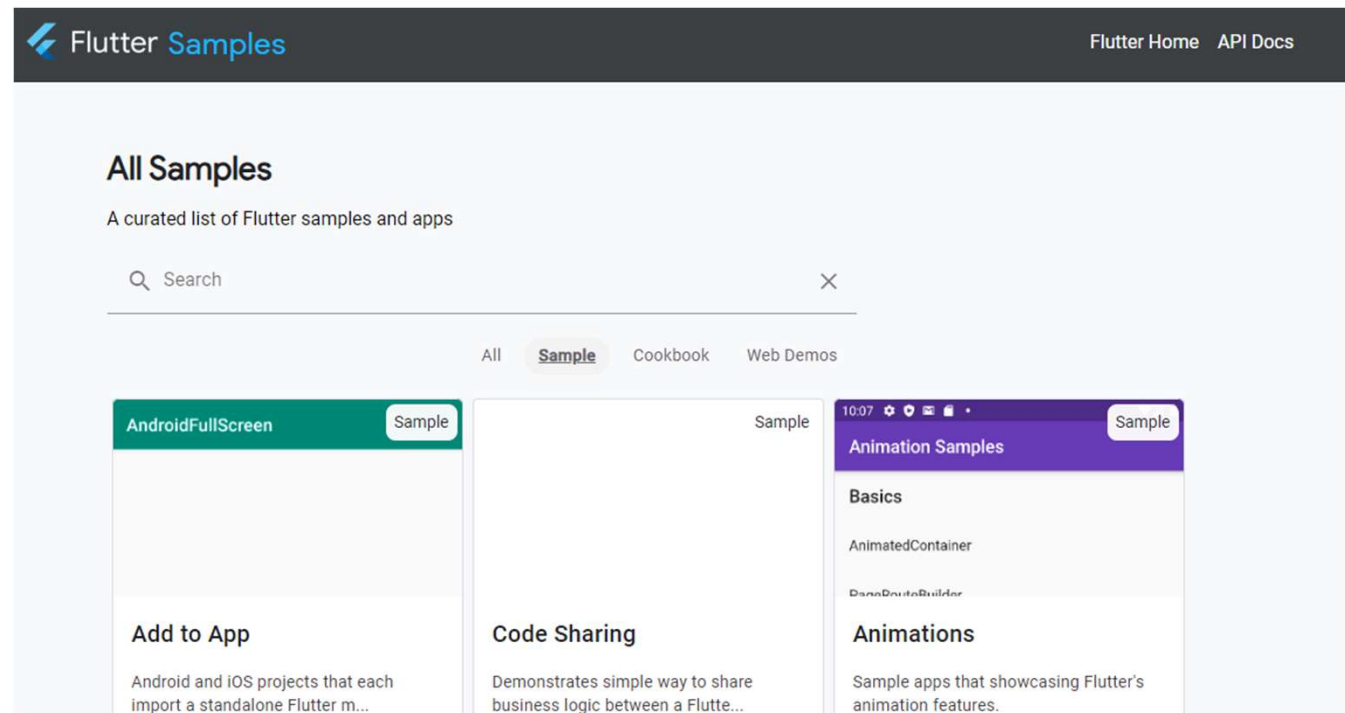


FloatingActionButton.01 e 02



FLUTTER

Exemplos



<https://flutter.github.io/samples/#?type=sample>

