







O Flutter possui diversos widgets, para diversas situações. Veremos alguns exemplos.

Crie uma nova aplicação Flutter com o nome widgets.

> flutter create widgets



Altere main.dart para:

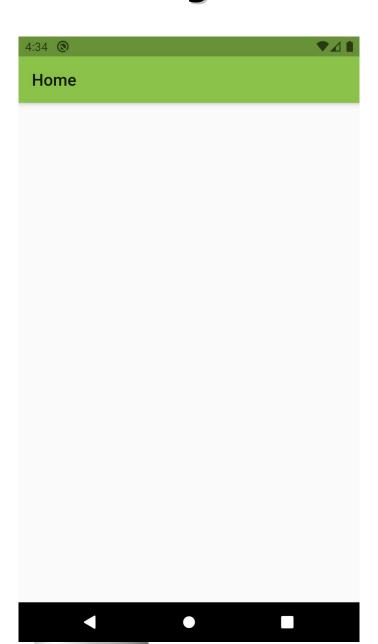
```
import 'package:flutter/material.dart';
import 'package:widgets/pages/home.dart';
void main() => runApp(MyApp());
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context)
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Starter Template',
      theme: ThemeData(
        primarySwatch: Colors.lightGreen,
        visualDensity:
VisualDensity.adaptivePlatformDensity,
      home: Home(),
```



Crie o subdiretório lib/pages e o arquivo home.dart nesse subdiretório, com o seguinte conteúdo:

```
import 'package:flutter/material.dart';
class Home extends StatefulWidget {
  @override
  _HomeState createState() => _HomeState();
class HomeState extends State<Home>/
  @override
  Widget build(BuildContext context)/{/
    return Scaffold(
      appBar: AppBar(
        title: Text('Home'),
      body: Container(),
```







Nessa aplicação temos alguns widgets muito utilizados:

MaterialApp – é uma classe de conveniência para envolver os widgets de uma aplicação, fornecendo um layout orientado ao Material Design do Google.

ThemeData - define a configuração visual do tema geral para um MaterialApp ou uma subárvore de widgets.

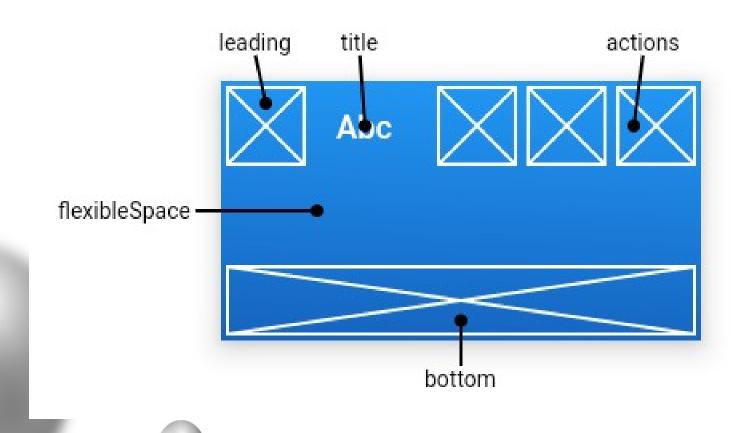
Scaffold – implementa o layout básico do visual do Material Design, permitindo a adição de outros widgets como AppBar.

AppBar – provê uma toolbar, título e outros elementos para a aplicação.

Text - exibe um texto

Container – permite a adição de outros widgets dentro do container, possibilitando organizar esses widgets.





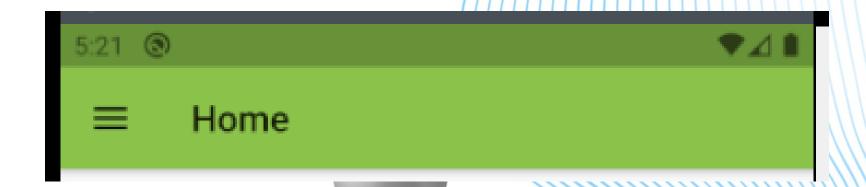


No leading da AppBar, pode ser adicionado um menu para a aplicação.

IconButton – botão que exibe um ícone e aciona uma ação.
Icon – widget não interativo que exibe um ícone.

Adicione o seguinte código após o titulo da aplicação:

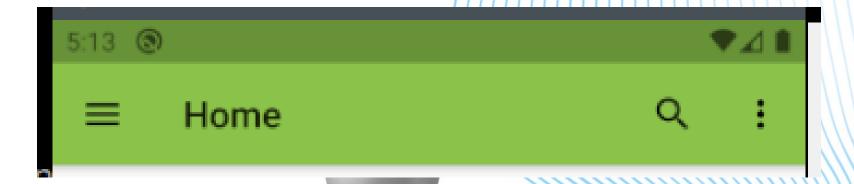
```
leading: IconButton(
  icon: Icon(Icons.menu),
  onPressed: () {
    print("Chamar menu da AppBar");
  },
},
```





No actions da AppBar, podem ser adicionados ícones para ações da aplicação, Adicione o seguinte código após o leading da aplicação:

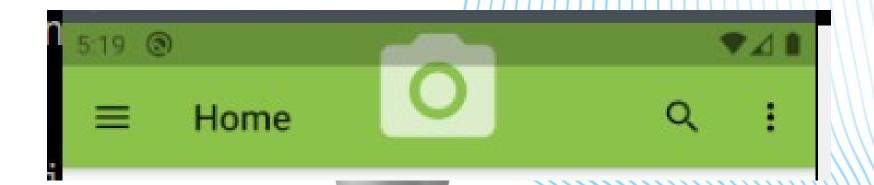
```
actions: <Widget>[
   IconButton(
      icon: Icon(Icons.search),
      onPressed: () {},
   ),
   IconButton(
      icon: Icon(Icons.more_vert),
      onPressed: () {},
   ),
   ],
}
```





No flexibeSpace da AppBar, pode ser adicionado qualquer widget, usualmente é utilizada uma imagem. Adicione o seguinte código após o actions da aplicação:

```
flexibleSpace: Icon(
    Icons.photo_camera,
    size: 75.0,
    color: Colors.white70,
),
```





Com os novos celulares utilizando notch para a câmara frontal, o conteúdo da flexibeSpace pode ser colocado em uma SafeArea para ser deslocado, evitando o notch.

SafeArea – desloca o conteúdo para evitar interferência de elementos do hardware ou sistema operacional.

Altere o código do flexibleSpace para;



Para incluir diversos widgets no corpo da página da aplicação, primeiro inclua alguns widgets para gerenciar a exibição dos widgets.

Padding – ajusta o conteúdo com um deslocamento.

SingleChildScrollView – uma caixa para único widget que permite o scroll se o widget não ficar inteiramente visível.

Column – exibe os widgets em um array vertical.

Substitua o corpo da página por:



Inclua um container com decoração e um texto formatado.

BoxDecoration – determina a decoração de uma caixa.

LinearGradient – define o gradiente de cor de um **BoxDecoration**.

BoxShadow – define o sombreamento de um **BoxDecoration**.

Center – centraliza o conteúdo.

RichText – exibe um texto com múltiplos formatos, o texto deve ser divido em uma árvore de TextSpan.

TextSpan – um pedaço do texto a ser formatado.

TextStyle – definição do estilo de um texto.



Acrescente a seguinte linha na lista de widgets do corpo da aplicação:

```
children: <Widget>[
  const ContainerWithBoxDecorationWidget(),
Insira a seguinte classe no arquivo home.dart:
class ContainerWithBoxDecorationWidget extends
StatelessWidget {
  const ContainerWithBoxDecorationWidget({
    Key key,
  }) : super(key: key);
  @override
  Widget build(BuildContext context) {
    return Column(
      children: <Widget>[
        Container(
          height: 100.0,
```



```
decoration: BoxDecoration(
  borderRadius: BorderRadius.only(
    bottomLeft: Radius.circular(100.0),
    bottomRight: Radius.circular(10.0),
  gradient: LinearGradient(
    begin: Alignment.topCenter,
    end: Alignment.bottomCenter,
    colors: [
      Colors.white,
      Colors.lightGreen.shade500,
    ],
  boxShadow:
    BoxShadow(
      color: Colors.grey,
      blurRadius: 10.0,
      offset: Offset(0.0, 10.0),
```

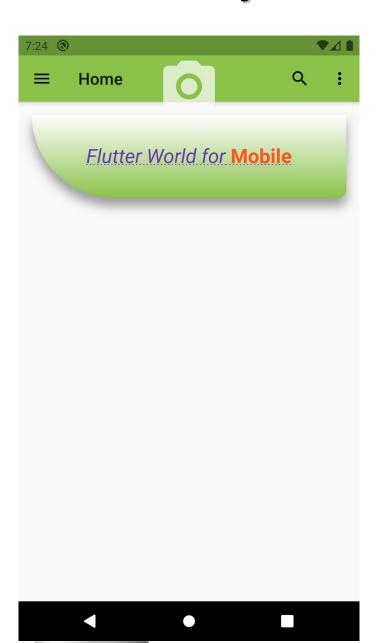


```
child: Center(
            child: RichText(
              text: TextSpan(
                text: 'Flutter World',
                style: TextStyle(
                  fontSize: 24.0,
                  color: Colors.deepPurple,
                  decoration: TextDecoration.underline,
                  decorationColor: Colors.deepPurpleAccent,
                  decorationStyle:
TextDecorationStyle.dotted,
                  fontStyle: FontStyle italic,
                  fontWeight: FontWeight.normal,
                ),
```



```
children: <TextSpan>[
                 TextSpan(
                   text: 'for',
                 TextSpan(
                   text: '
                          Mobile',
                   style: TextStyle(
                       color: Colors.deepOrange,
                       fontStyle: FontStyle.normal,
                       fontWeight: FontWeight.bold),
```





Colunas



Divider – cria uma linha horizontal entre os widgets.

Para um exemplo de exibição de widgets em coluna, acrescente as seguintes linhas na lista de widgets do corpo da aplicação:

```
children: <Widget>[
  const ContainerWithBoxDecorationWidget(),
  Divider(),
  const ColumnWidget(),
]
```

Insira a seguinte classe no arquivo home.dart:

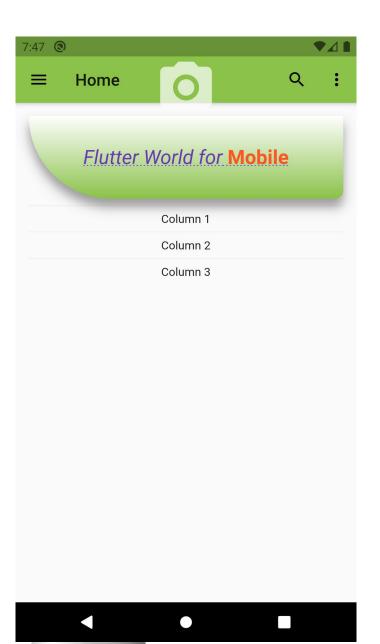
Colunas



```
class ColumnWidget extends StatelessWidget {
  const ColumnWidget({
    Key key,
  }) : super(key: key);
  @override
  Widget build(BuildContext context) {
    return Column(
      crossAxisAlignment: CrossAxisAlignment.center,
      mainAxisAlignment: MainAxisAlignment.spaceEvenly,
      mainAxisSize: MainAxisSize.max,
      children: <Widget>[
        Text('Column 1'),
        Divider(),
        Text('Column 2'),
        Divider(),
        Text('Column 3'),
```

Colunas





Linhas



Row – exibe os widgets em um array horizontal.

Para um exemplo de exibição de widgets em linha, acrescente as seguintes linhas na lista de widgets do corpo da aplicação:

```
Divider(),
const RowWidget(),

Insira a seguinte classe no arquivo home.dart:

class RowWidget extends StatelessWidget {
   const RowWidget({
     Key key,
   }) : super(key: key);
```

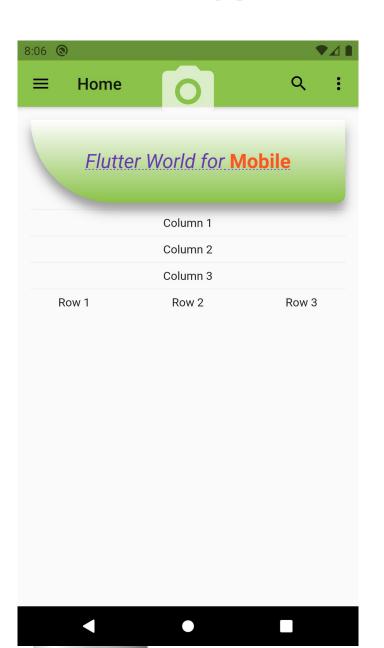
Linhas



```
@override
Widget build(BuildContext context) {
  return Row(
    crossAxisAlignment: CrossAxisAlignment.start,
    mainAxisAlignment: MainAxisAlignment.spaceEvenly,
    mainAxisSize: MainAxisSize.max,
    children: <Widget>[
      Text('Row 1'),
      Padding(
        padding: EdgeInsets.all(16.0),
      Text('Row 2'),
      Padding(
        padding: EdgeInsets.all(16.0),
      Text('Row 3'),
```

Linhas





Linhas e Colunas



Linhas e colunas podem ser combinadas. Acrescente as seguintes linhas na lista de widgets do corpo da aplicação:

```
Divider(),
const ColumnAndRowNestingWidget(),
Insira a seguinte classe no arquivo home.dart:
class ColumnAndRowNestingWidget extends StatelessWidget {
  const ColumnAndRowNestingWidget({
    Key key,
  }) : super(key: key);
  @override
  Widget build(BuildContext context)/{/
    return Column(
      crossAxisAlignment: CrossAxisAlignment.center,
      mainAxisAlignment: MainAxisAlignment.spaceEvenly,
      mainAxisSize: MainAxisSize.max,
```

Linhas e Colunas



```
children: <Widget>[
  Text('Célula 1 1'),
  Row (
    mainAxisAlignment: MainAxisAlignment.spaceEvenly,
    children: <Widget>[
      Text('Célula 2 1'),
      Text('Célula 2 2'),
  Row (
    mainAxisAlignment: MainAxisAlignment.spaceEvenly,
    children: <Widget>[
      Text('Célula 3 1'),
      Text('Célula 3 2'),
      Text('Célula 3 3'),
```

Linhas e Colunas



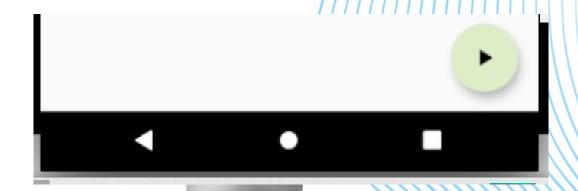


FloatingActionButton



FloatingActionButton é um botão que pode flutuar sobre outros elementos. Adicione as seguintes linhas após o Padding no corpo da página da aplicação:

```
floatingActionButton: FloatingActionButton(
  onPressed: () {
    print("Acionando Floating Action");
  },
  child: Icon(Icons.play_arrow),
  backgroundColor: Colors.lightGreen.shade100,
),
```



BottomAppBar



BottomAppBar é um container usualmente utilizado na bottomNavigationBar do Scaffold e pode possuir um notch para acomodar o FloatingActionButton.

GestureDetector – detecta gestos.

Acrescente as seguintes linhas após o FloatingActionButton:

BottomAppBar



```
floatingActionButtonLocation:
FloatingActionButtonLocation.endDocked,
bottomNavigationBar: BottomAppBar(
  color: Colors.lightGreen.shade100,
  child: Row(
    mainAxisAlignment: MainAxisAlignment.spaceEvenly,
    children: <Widget>[
      GestureDetector(
        child: Icon(Icons.pause),
        onTap: () => print("Paused")
      Icon(Icons.stop),
      Icon(Icons.access_time),
      Padding(padding: EdgeInsets.all(32.0)),
                   II
```



FlatButtom é um botão minimalista sem borda ou elevação.

RaiseButton é um botão com elevação.

Acrescente as seguintes linhas na lista de widgets do corpo da aplicação:

```
Divider(),
const ButtonsWidget(),

Insira a seguinte classe no arquivo home.dart:

class ButtonsWidget extends StatelessWidget {
   const ButtonsWidget({
      Key key,
   }) : super(key: key);
```

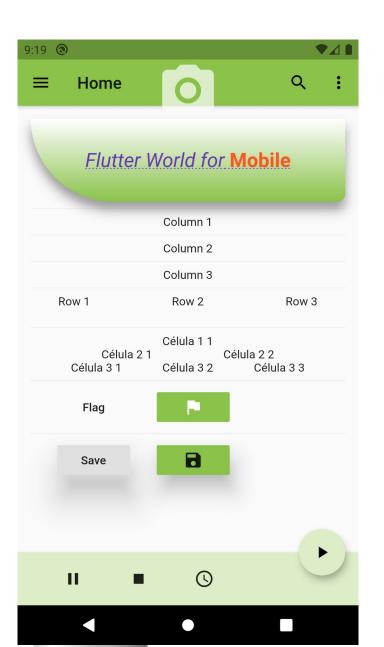


```
@override
Widget build(BuildContext context) {
  return Column(
    children: <Widget>[
      Row (
        children: <Widget>[
          Padding(padding: EdgeInsets.all(16.0)),
          FlatButton(
            onPressed: () => print("Flag Texto"),
            child: Text('Flag'),
          Padding(padding: EdgeInsets.all(16.0)),
          FlatButton(
            onPressed: () => print("Flag Icon"),
            child: Icon(Icons.flag),
            color: Colors.lightGreen,
            textColor: Colors.white,
      Divider()
```



```
Row (
  children: <Widget>[
    Padding(padding: EdgeInsets.all(16.0)),
    RaisedButton(
      onPressed: () => print("Save Texto"),
      elevation: 29,
      child: Text('Save'),
    Padding(padding: EdgeInsets.all(16.0)),
    RaisedButton(
      onPressed: () => print("Save Icon"),
      elevation: 20,
      child: Icon(Icons.save),
      color: Colors.lightGreen,
```









ButtomBar alinha horizontalmente um grupo de botões. Se não houver espaço suficiente, os botões serão colocados em uma coluna.

Acrescente as seguintes linhas na lista de widgets do corpo da aplicação:

```
Divider(),
const ButtonBarWidget(),
Insira a seguinte classe no arquivo home.dart:
class ButtonBarWidget extends StatelessWidget {
  const ButtonBarWidget({
    Key key,
  }) : super(key: key);
```

Botões



```
@override
Widget build(BuildContext context) {
  return Container(
    color: Colors.white70,
    child: ButtonBar(
      alignment: MainAxisAlignment.spaceEvenly,
      children: <Widget>[
        IconButton(
          onPressed: () => print("Inserir"),
          icon: Icon(Icons.add), ),
        IconButton(
          onPressed: () => print("Salvar"),
          icon: Icon(Icons.save),//),
        IconButton(
          onPressed: () => print("Deletar"),
          icon: Icon(Icons.delete),
          highlightColor: Colors red, ),
```

Botões









PopupMenuButton exibe um menu com uma lista de itens. Ao ser selecionado um valor, esse valor é passado para onSelected. Uma das formas de adicionar os itens do menu é criando uma lista de itens e utilizando itemBuilder.

Acrescente em home.dart uma classe para os itens do menu:

```
class MenuItem {
  final String title;
  final Icon icon;
  MenuItem({this.title, this.icon});
}
Acrescente a lista de itens do menu:
List<MenuItem> MenuList = [
```

.ist<MenuItem> MenuList = [
 MenuItem(title: 'Fast Food', icon: Icon(Icons.fastfood)),
 MenuItem(title: 'Remind Me', icon: Icon(Icons.add_alarm)),
 MenuItem(title: 'Flight', icon: Icon(Icons.flight)),
 MenuItem(title: 'Music', icon: Icon(Icons.audiotrack)),
:



Insira a seguinte classe no arquivo home.dart:

```
class PopupMenuButtonWidget extends StatelessWidget
    implements PreferredSizeWidget {
  const PopupMenuButtonWidget({
    Key key,
  }) : super(key: key);
  @override
  Widget build(BuildContext context)
    return Container(
      color: Colors.lightGreen.shade100,
      height: preferredSize.height,
      width: double.infinity,
      child: Center(
        child: PopupMenuButton<MenuItem>(
          icon: Icon(Icons.view_list),
          onSelected: ((valueSelected) {
            print('valueSelected: ${valueSelected.title}');
          }),
```



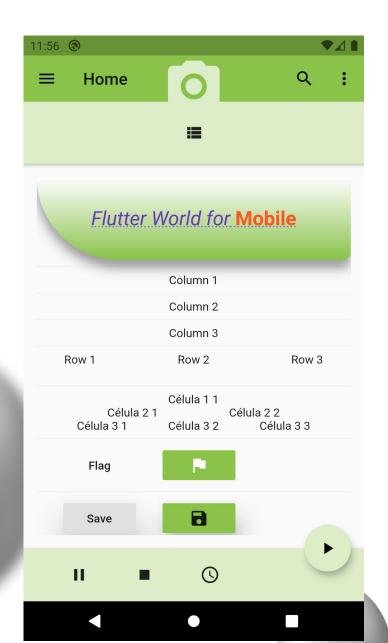
```
itemBuilder: (BuildContext context) {
          return MenuList.map((MenuItem menuItem) {
            return PopupMenuItem<MenuItem>(
              value: menuItem,
              child: Row(
                children: <Widget>[
                  Icon(menuItem.icon.icon),
                  Padding(
                    padding: EdgeInsets.all(8.0), ),
                  Text(menuItem.title),
                ],
          }).toList();
@override
Size get preferredSize => Size.fromHeight(75.0);
```



Acrescente a seguinte linha após o flexibleSpace da AppBar da aplicação:

bottom: const PopupMenuButtonWidget(),

Titulo do Tópico



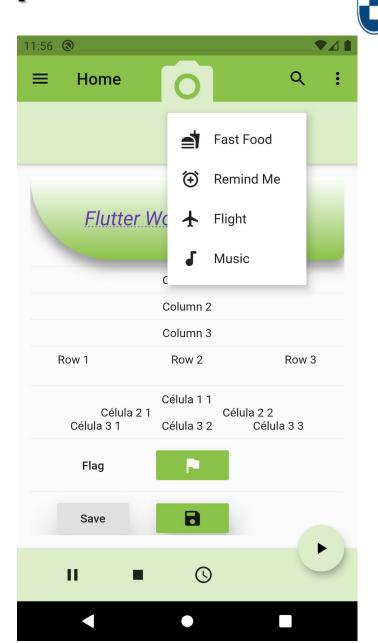




Image exibe uma imagem. A imagem pode ser obtida de várias fontes, dependendo do construtor usado para criar o objeto:

- Image() obtém a imagem da classe ImageProvider.
- ■Image.asset() obtém a imagens da classe AssetBunble (coleção de assets da aplicação).
- Image.network() obtém a imagem de uma URL.
- Image.file() obtém a imagem da classe File.
- Image.memory() obtém a imagem da classe Uint8List.



Para incluir uma imagem na coleção de assets da aplicação, crie o subdiretórios assets e assets/images.

Copie os arquivos tux.jpg, bmw_m3.jpg, nissan_gt-r.jpg e nissan_sentra.jpg, dawn.jpg, eagle.jpg. para o subdiretório assets/images.

Acrescente as linhas a seguir no arquivo pubspec.yaml: assets:

- assets/images/tux.jpg
- assets/images/bmw_m3.jpg
- assets/images/nissan_gt-r.jpg
- assets/images/nissan_sentra.jpg
- assets/images/dawn.jpg
- assets/images/eagle.jpg

Também pode ser usada a linha a seguir para incluir todos os arquivos no subdiretório assets/images:

assets:

- assets/images/



Acrescente as seguintes linhas na lista de widgets do corpo da aplicação:

```
Divider(),
const ImagesWidget(),

Insira a seguinte classe no arquivo home.dart:

class ImagesWidget extends StatelessWidget {
   const ImagesWidget({
     Key key,
   }) : super(key: key);
```



```
@override
  Widget build(BuildContext context) {
    return Row(
      mainAxisAlignment: MainAxisAlignment.spaceEvenly,
      children: <Widget>[
        Image(
          image: AssetImage("assets/images/tux.jpg"),
          fit: BoxFit.cover,
          width: MediaQuery.of(context).size.width / 3,
        Image.network(
          'https://flutter.io/images/catalog-widget-
placeholder.png',
          width: MediaQuery.of(context).size.width / 3,
```





ListView



ListView exibe uma lista de widgets em uma lista horizontal ou vertical. A lista pode ser criada estaticamente ou dinamicamente.

Acrescente as seguintes linhas na lista de widgets do corpo da aplicação:

```
Divider(),
const ListViewWidget(),
Insira a seguinte classe no arquivo home.dart:
class ListViewWidget extends StatelessWidget {
   const ListViewWidget({
     Key key,
   }) : super(key: key);
```

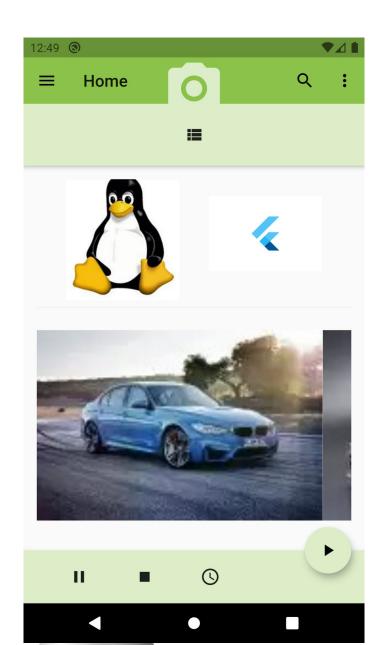
ListView



```
@override
  Widget build(BuildContext context) {
    return Container(
      margin: EdgeInsets.symmetric(vertical: 20.0),
      height: 230.0,
      child: ListView(
        scrollDirection: Axis.horizontal,
        children: <Widget>[
          Image(fit: BoxFit.cover,
            image: AssetImage("assets/images/bmw_m3.jpg"),
          Image(fit: BoxFit.cover,
            image: AssetImage("assets/images/nissan_gt-
r.jpg"), ),
          Image(fit: BoxFit.cover,
            image:
AssetImage("assets/images/nissan_sentra.jpg"),
```

ListView







Stack exibe widgets posicionados dentro de uma região, permitindo a sobreposição dos widgets.

Positioned – controla a posição do widget dentro do Stack.

FractionalTranslation - aplica uma translação no widget.

CircleAvatar – um circulo que representa um usuário.

Acrescente as seguintes linhas na lista de widgets do corpo da aplicação:

```
Divider(),
const StackWidget(),
```

Insira a seguinte classe no arquivo home.dart:

```
class StackWidget extends StatelessWidget {
  const StackWidget({
    Key key,
  }) : super(key: key);
```





```
Positioned(
  top: 0.0,
  right: 0.0,
  child: FractionalTranslation(
    translation: Offset(0.3, -0.3),
    child: CircleAvatar(
      radius: 24.0,
      backgroundColor: Colors.white30,
      child: Icon(
        Icons.star,
        size: 24.0,
        color: Colors.white,
```

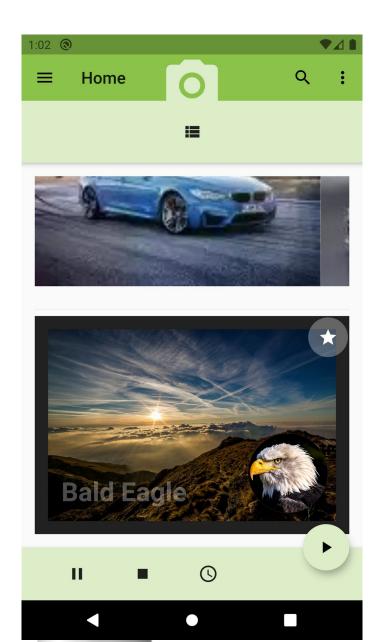
```
Positioned(
bottom: 10.0,
right: 10.0,
child: CircleAvatar(
radius: 48.0,
backgroundImage:
AssetImage('assets/images/eagle.jpg'),
),
),
```





```
Positioned(
  bottom: 16.0,
  left: 16.0,
  child: Text(
    'Bald Eagle',
    style: TextStyle(
      fontSize: 32.0,
      color: Colors.white30,
      fontWeight: FontWeight.bold,
```







Wrap exibe uma lista de widgets ajustando em linhas e colunas.

Chip – elemento compacto para representar um atributo, texto, entidade, ou ação.

Acrescente as seguintes linhas na lista de widgets do corpo da aplicação:

```
Divider(),
const WrapWidget(),
Insira a seguinte classe no arquivo home.dart:
class WrapWidget extends StatelessWidget {
   const WrapWidget({
     Key key,
   }) : super(key: key);
```

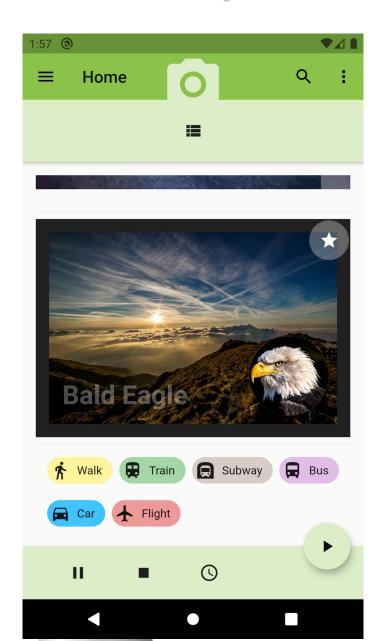


```
@override
Widget build(BuildContext context) {
  return Wrap(
    spacing: 8.0, // gap between adjacent chips
    runSpacing: 4.0, // gap between lines
    children: <Widget>[
      Chip(
        avatar: Icon(Icons.directions_walk),
        backgroundColor: Colors.yellow[200],
        label: Text('Walk'),
      Chip(
        avatar: Icon(Icons.train)
        backgroundColor: Colors.green[200],
        label: Text('Train'),
      Chip(
        avatar: Icon(Icons.subway),
        backgroundColor: Colors.brown[100],
        label: Text('Subway'),
```



```
Chip(
  avatar: Icon(Icons.directions_bus),
  backgroundColor: Colors.purple[100],
  label: Text('Bus'),
Chip(
  avatar: Icon(Icons.directions_car),
  backgroundColor: Colors.lightBlueAccent,
  label: Text('Car'),
Chip(
  avatar: Icon(Icons.flight),
  backgroundColor: Colors.red[200],
  label: Text('Flight'),
```







Drawer exibe um painel que desliza a partir da borda do Scaffold. Se não for definido um leading na AppBar, o Drawer pode ser aberto pelo botão que aparecerá no local do leading.

DrawHeader é o cabeçalho do painel do Drawer.

ListTile é um elemento para uma lista que permite incluir um botão, de uma a três linhas de texto ou outros widgets.

Acrescente o seguinte código após o body do Scaffold da aplicação:

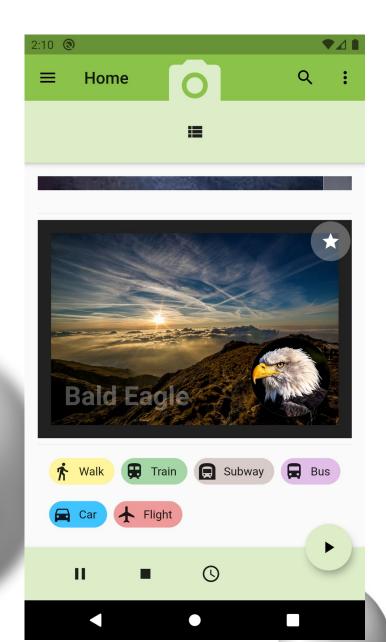


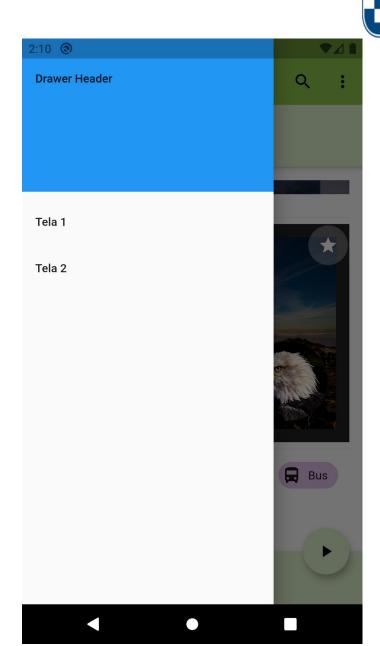
```
drawer: Drawer(
  child: ListView(
    padding: EdgeInsets.zero,
    children: <Widget>[
      DrawerHeader(
        child: Text('Drawer Header'),
        decoration: BoxDecoration(
          color: Colors.blue,
      ListTile(
        title: Text('Tela 1'
        onTap: () {
          print("Abrir tela/1");
          Navigator.pop(context);
        },
```

```
ListTile(
    title: Text('Tela 2'),
    onTap: () {
       print("Abrir tela 2");
       Navigator.pop(context);
     },
    ),
    ]),
}
```

Comente as seguintes linhas da AppBar:

```
/*
leading: IconButton(
  icon: Icon(Icons.menu),
  onPressed: () {
    print("Chamar menu da AppBar");
  },
),
*/
```







A aplicação pode ter um Drawer na borda esquerda, na borda direita ou um em cada borda. Se não for definido um action na AppBar, o Drawer da borda direita pode ser aberto pelo botão que aparecerá no local do action.

Altere a seguinte linha do Scaffold:

drawer: new Drawer(

Para:

endDrawer: new Drawer(



Comente as seguintes linhas da AppBar:

```
/*
actions: <Widget>[
   IconButton(
     icon: Icon(Icons.search),
     onPressed: () {},
   ),
   IconButton(
     icon: Icon(Icons.more_vert),
     onPressed: () {},
   ),
   ],
   ],
},
```

