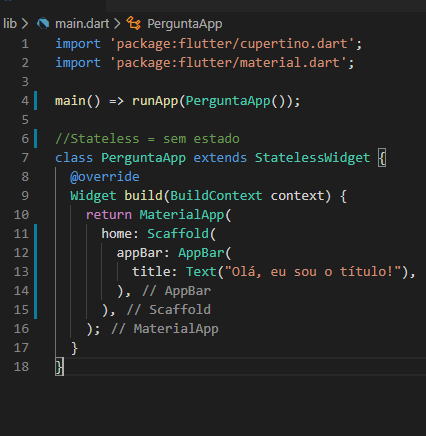
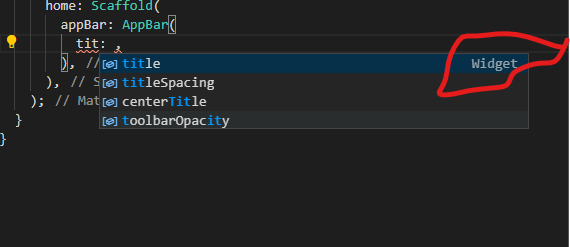
|  |
| --- |
| Imports de libs usadas no projeto, onde cupertino.dart é o padrão do iOS e material.dart padrão do Android. |
| Função main(), principal função, ela que inicia seu projeto. |
| Class do tipo Stateless é uma class sem estado e uma Class Stateful é com estado. |
| Todo Widget tem o construtor build que você deve passar o contexto. |
| MaterialApp geralmente usado para criar rotas, mas não só para isso. |
| Scaffold é a estrutura do app. |





|  |
| --- |
| Uma maneira de identificar se você deve informar um widget ou outro tipo de elemento é ao digitar o argumento vc vê o tipo que o argumento espera, nesse caso widget. |

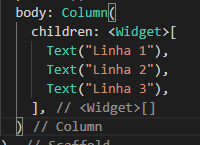


**Usando Column para organizar aplicação**

Conceito de flexBox.

Recebe vários widget’s como filhos, nesse caso uma lista.

Ex de implementação:



**Passando funções como parâmetro:**

Observe que passar uma função como parâmetro é diferente de chamar uma função, vamos ao ex:

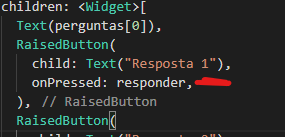
Chamando uma função:

funcao();

Repare que passo os parênteses normalmente.

Chamando uma função como parâmetro:

Apenas escreva -> responder, veja o ex onde passo a função responder como parâmetro, pois onPressed espera isso.



Se quiser passar como função, responder(), você pode fazer uma função que retorna um função, vejamos:

    void Function() funcaoQueRetornaUmaOutraFuncao () {

      return () {

        print("Pergunta Respondida!");

      };

    }

Agora, você pode chamar no onPressed: funcaoQueRetornaUmaOutraFuncao()

Temos então, 3 formas diferentes para passar uma função para o argumento onPressed do RaisedButton, vejamos:

            RaisedButton(

              child: Text("Resposta 1"),

              onPressed: responder,

            ),

            RaisedButton(

              child: Text("Resposta 2"),

              onPressed: () {

                print("Função do tipo normal")

              },

            ),

            RaisedButton(

              child: Text("Resposta 3"),

              onPressed: () => {

                print("Função do tipo Arrow Function")

              },

Algo importante a ser anotado é que quando passamos null para onPressed o botão fica desabilitado.

Widgets do tipo Stateless são sem estado e por isso você não deve fazer alterações de estado/layout no mesmo, pois não irão funcionar e não tente burlar isso.

Use o Widget do tipo Stateful para isso.



Stateless só muda de estado com ações exeternas.

Stateful muda de estado com ações externas e internas dele próprio.

