**Form**

Um contêiner opcional para agrupar vários widgets de FormField (por exemplo, widgets TextField).

Cada campo do formulário deve ser agrupado em um widget FormField, com o widget Form como um ancestral comum a todos eles. Chame métodos no FormState para salvar, redefinir ou validar cada FormField que é um descendente deste formulário. Para obter o FormState, você pode usar Form.of com um context cujo ancestral é o Form, ou passar um GlobalKey para o construtor Form e chamar GlobalKey.currentState.

**TextFormField**

Um FormField que contém um TextField. Esse é um widget que por conveniência envolve um widget TextField em um FormField.

Um ancestral Form não é obrigatório. O formulário simplesmente torna mais fácil salvar, redefinir ou validar vários campos de uma só vez. Para usar sem um formulário, passe um GlobalKey para o construtor e use GlobalKey.currentState para salvar ou redefinir o campo de formulário.

**RaisedButton**

Um botão baseado em um widget de MaterialDesign cuja elevação aumenta quando o botão é pressionado.

Use RaisedButtons para adicionar dimensão a layouts geralmente simples, por exemplo, em longas listas de conteúdo.

Se o retorno de chamada onPressed for null, o botão será desativado e, por padrão, será semelhante a um botão plano no disabledColor.

**CheckBox**

A checkbox não mantém nenhum estado. Em vez disso, quando o estado da checkbox é alterado, o widget chama o onChanged. A maioria dos widgets que usam uma checkbox , a partir do onChanged, reconstruirá a checkbox com um novo valor para atualizar a aparência visual da checkbox.

A checkbox pode exibir, opcionalmente, três valores - true, false e null - se o estado triplo for verdadeiro. Quando o valor é null, um traço é exibido. Por padrão, o estado triplo é false e o valor da checkbox deve ser true ou false.

**RadioButton**

Usado para selecionar entre um número de valores mutuamente exclusivos. Quando um RadioButton de um grupo é selecionado, os outros RadioButton do grupo deixam de ser selecionados.

O RadioButton em si não mantém nenhum estado. Se groupValue e value corresponderem, esse rádio será selecionado. A maioria dos widgets responderá ao onChanged chamando State.setState para atualizar o groupValue do RadioButton.

**SnackBar**

Uma mensagem estilo pop-up com uma ação opcional que é exibida rapidamente na parte inferior da tela.

Para exibir uma SnackBar, chame Scaffold.of(context).showSnackBar(), passando uma instância de SnackBar que descreva a mensagem.

Para controlar por quanto tempo a SnackBar permanece visível, especifique uma duração.

**BottomNavigationBar**

O BottomNavigationBar consiste em vários itens na forma de rótulos de texto, ícones ou ambos, dispostos em um widget. Ele fornece navegação rápida entre as visualizações de nível superior ou inferior de um aplicativo. Para telas maiores, a navegação lateral pode ser mais adequada.

Um BottomNavigationBar é geralmente usada em conjunto com um Scaffold, onde é fornecido como o argumento Scaffold.bottomNavigationBar.

O tipo do BottomNavigationbar altera a maneira como seus itens são exibidos. Se não for especificado, ele será definido automaticamente como BottomNavigationBarType.fixed quando houver menos de quatro itens e BottomNavigationBarType.shifting, caso contrário.

**DropdownMenu**

Um widget exibido na parte inferior de um aplicativo para selecionar entre um pequeno número de visualizações, geralmente entre três e cinco.

Um DropdownMenu permite ao usuário selecionar um número de itens. O botão mostra o item atualmente selecionado, bem como uma seta que abre um menu para selecionar outro item.

Todas as entradas em um determinado menu devem representar valores com tipos consistentes. Normalmente, um enum é usado. Cada DropdownMenuItem nos itens deve ser especializado com o mesmo argumento de tipo.

O retorno de chamada onChanged deve atualizar uma variável de estado que define o valor da lista. Ele também deve chamar State.setState para recriar o DropdownMenu com o novo valor.

**SingleChildScrollView**

Permite deixar a tela “rolável” evitando assim o estouro de pixel, por exemplo.

Também é útil se você precisar encolher em ambos os eixos (a direção de rolagem principal e o eixo cruzado), como em um diálogo ou menu pop-up.

**CarouselSlider**

A carousel slider widget, support infinite scroll and custom child widget, with auto play feature.

**Hero**

Quando uma PageRoute é enviada ou exibida com o Navigator, todo o conteúdo da tela é substituído. Uma rota antiga desaparece e uma nova rota aparece. Se houver um recurso visual comum em ambas as rotas, pode ser útil orientar o usuário para que o recurso se mova fisicamente de uma página para outra durante a transição das rotas. Essa animação é chamada de animação Hero. Os widgets Heros "voam" na sobreposição do Navigator durante a transição e, enquanto estão em voo, por padrão, eles não são exibidos em seus locais originais nas rotas antigas e novas.

Para rotular um widget como um recurso, coloque-o em um widget Hero. Quando a navegação acontece, os widgets Hero em cada rota são identificados pelo HeroController. Para cada par de widgets Hero que possuem a mesma tag, uma animação de Hero é acionada.