Flutter Random Guess App - Qual é o número?

Este projeto foi desenvolvido em Flutter, framework de UI desenvolvido pelo Google, e utiliza linguagem de programação Dart. O app possui apenas uma tela onde o jogador deverá tentar descobrir o número sorteado, inserindo seu palpite no campo de texto e clicando em enviar. O app dirá se o palpite foi maior ou menor que o número sorteado até que o jogador acerte.

Executando o projeto

Com um dispositivo ou emulador conectados, rode os seguintes comandos na raiz do projeto:

```
flutter pub get
flutter run
```

Detalhes de implementação

Para possibilitar a reutilização de widgets, serviços e outras classes o diretório `lib` foi estruturado da seguinte forma:

```
lib
|- components
|- models
|- repositories
|- screens
|- services
main.dart
```

O arquivo **`main.dart`** aplica a estilização, propriedades do layout, o estado global **`GlobalState`** e renderiza a página **`HomePage`**. Essa, por sua vez, renderiza os componentes **`DisplayContainer`** e **`FormRow`**, respectivamentes responsáveis por o palpite do usuário junto à resposta do sistema e receber o input do usuário. A HomePage é um **`StatefulWidget`** que armazena os principais dados da aplicação. Por meio da função assíncrona **`_fetchRandomNumber`**, que é chamada ao iniciar do componente, obtém-se da API um **`NumberResponse`** cuja propriedade **`value`** é salva na variável de estado **`_randomNumber`**.

As chamadas à API são feitas pelo **`randomNumberRepository`** que por sua vez chama o **`ApiProvider`**. Esse processo é envolvido por blocos try-catch de modo que eventuais erros sejam tratados. Tanto erros enviados pela API, como falhar de internet e outros enviam a exceção **`CustomException`** que armazena o **`StatusCode`** e a mensagem de erro, para serem exibidos na HomePage.

No **`FormRow`**, o input do usuário é um **`TextField`** numérico e não negativo. O botão 'Enviar' aciona a função **`onSubmit`** que faz as verificações de "Acerto", "Maior" ou "Menor".

Em caso de o usuário acertar ou ocorrer erro na API, a variável **`_showPlayAgainButton`** do estado da HomePage é setada como true e então exibido o botão "Nova Partida". Além disso, sempre que essa variável ou **`_isLoading`** for true, o botão enviar fica desabilitado. Ambos os botões são instâncias do mesmo widget **`CustomButton`**, visto que o layout e comportamento são muito similares.

Por fim, o **`CustomSegmentDisplay`** foi desenvolvido da seguinte forma:

1. Receber um valor inteiro e separa as unidades, dezenas e centenas, passando-as cada uma para um **`SingleDisplay`** organizados em linha.
2. O **`SingleDisplay`** recebe o algarismo e cria um **`ledMap`**, indicando quais segmentos devem acender para que o número seja exibido e envia-o para um **`DisplayPainter`**.
3. Este usa a função paint do **`CustomPainter`** para criar retângulos com os devidos tamanhos e posições(*), para formar um display de sete segmentos. Inicialmente, pinta-se o fundo de todos os retângulos e, posteriormente, de vermelho aqueles de devem estar ligados.

(*) Os tamanhos e posições inicialmente foram implementados de forma estática, como abaixo:

```
canvas.drawRect(Offset(10, 0) & Size(40, 10), paintOff);
canvas.drawRect(Offset(10, 55) & Size(40, 10), paintOff);
// ...
```

No entanto, ao implementar a alteração dinâmica do tamanho e cores, foram utilizadas as variáveis: *hSize*, *vSize*, *aOffset*, *bOffset*, *cOffset* etc. Estas variáveis são calculadas a partir de um *size*, proveniente do **`GlobalState`**: trata-se de um **`StatefulWidget`** que armazena *size* e *color*, utilizando um **`InheritedWidget`**, tornando assim possível acessar e manipular o estado de forma global.

Essas e outras estratégias de implementação e design patterns adotados visam organização, coesão, robustez, flexibilidade e escalabilidade do código. Vale ressaltar também que o projeto foi desenvolvido tendo multiplas fontes de informação como referência e deixo listadas abaixo algumas delas:

- https://flutter.dev/docs/
- https://api.flutter.dev/flutter/dart-ui/dart-ui-library.html
- https://itnext.io/solid-principles-explanation-and-examples-715b975dcad4
- https://itnext.io/flutter-handling-your-network-api-calls-like-a-boss-936eef296547
- https://stackoverflow.com/questions/53019061/how-to-implement-a-custom-dialog-box-in-flutter
- https://stackoverflow.com/questions/49491860/flutter-how-to-correctly-use-an-inherited-widget/49492495