現在、最後のいくつかのレッスンでは、コードをリファクタリングし、よりシンプルで読みやすく、より簡潔にしています。そしてその過程で、列挙型やconst、最終的な解析関数、プログラミングに関する多くの知識など、これを可能にするDartの基礎の多くを学びました。

しかし、このレッスンでは、設計に戻り、次のカードを作成します。

次のカードはスライダーになり、Flutterスライダーを使用しますが、好みに合わせてカスタマイズします。そしてその過程で、Flutterウィジェットのカスタマイズについて多くを学びます。

まず最初に、次の再利用可能なカードに向かいましょう。これはまさにここにあります。そして現時点では、実際には何もしません。それは完全に空です。

それではcardChildを与えましょう。 cardChildプロパティは、多くのものを保持します。垂直に積み上げられるものがかなりあるため、列ウィジェットを作成するのは理にかなっています。そして、それはたくさんの子供を持つことになります。その最初のものはテキストウィジェットになります。そして、テキストは単に単語の高さを言います。これは高さの入力になるからです。そして、私はそのテキストが男性と女性のテキストと同じスタイルを持っていることを本当に望んでいます。

したがって、同じフォントサイズ、同じ色、同じフォントです。そして現在、そのスタイルは、icon\_content.dartファイルで定義されている定数から来ています。そして、入力ページにあるラベルを定義できるようにアイコンコンテンツファイルにアクセスすることは、実際には意味がありません。

それは本当に面倒になりそうです。

専用の定数ファイルを作成することについて話をする良い機会です。デザインとユーザーインターフェイスを作成するとき、デザインを共有したり、異なる画面間で設定を共有したりすることが非常に頻繁にあります。

したがって、特定のファイルまたは特定のコンポーネント内に配置するのではなく、Dartファイルになり、定数と呼ばれる新しいファイルを作成することは理にかなっています。

そして、これはアプリ全体で一定のすべてのものを保持します。

そのため、たとえばこのlabelTextStyleが含まれます。そこで、ここから切り取って、代わりにここに貼り付けます。

そしてもちろん、マテリアルコンポーネントを使用するときはいつでも、マテリアルライブラリをインポートする必要があります。

これらのテキストスタイルに加えて、ここからすべての色定数を追加することもできます。それを代わりにこのファイルに貼り付けることができます。

そして今、私はすべての定数を1か所にまとめており、定数をインポートするだけで、任意のファイルからそれを参照できます。

そのため、現時点では、labelTextStyleを移動したため、もうlabelTextStyleを認識していないため、ここにいくつかのエラーがあることがわかります。

しかし、インポートを押してconstants.dartファイルをインポートするとすぐに、そのエラーは消え、ここを調べて、そのラベルに適用するテキストスタイルを確認できます。

入力ページでも同様に、このactiveCardcolorとinactiveCardColorがわからないため、あちこちでエラーが発生しています。

それでは、このファイルに定数をインポートしてみましょう。

ですから、これらの定数の命名規則は通常、Kで始まることに言及するのがよいタイミングです。

そのため、そのスタイルに合わせてコードをリファクタリングできます。したがって、各定数を右クリックして、リファクタリングと名前変更を行うことができます。これは、この定数を使用するすべての場所を検索し、すべての名前を一度に変更する友人です。

そのため、bottomContainerHeightを呼び出すのではなく、定数はK、Kから始めます。Cは他の多くのことに使用されるためです。

したがって、Kは定数であり、次に大文字のBから始めます。

K定数、bottomContainerHeightをリファクタリングし、それを使用しているすべての場所を見ることができるようになり、自動的にリファクタリングされました。

だから私は先に進み、他のすべての定数について同じことをするつもりです。そして、constants.dartで新しい定数を作成するときに、この命名規則を使用します。

大丈夫。

したがって、ここにすべての定数の名前が変更されており、使用されているすべての場所で名前が変更されます。

なぜ私はすべての定数の名前を変更するのに苦労したのですか？

定数ファイルから色またはテキストスタイルを選択しようとするときはいつでも、Kを入力することから始めます。そして、Android Studioは、これで使用したい関連定数をすべて表示できるほどインテリジェントです。ポイント。そのため、kBottomContainerColor、kBottom ContainerHeightがあり、kMaterialEdges、kMaterialListPaddingなどのFlutterで定義されている定数もあるため、この方法ですべての定数を簡単に検索できます。

それをkActiveCardColorに戻しましょう。そして、あなたが選択する定数を探しているたびに、ドロップダウンリストを選択することができます。これは、ファイルの上部にある定数の名前を覚えようとするよりもはるかに簡単です。アプリはより複雑になります。

大丈夫。

テキストウィジェットが追加されたreusableCardに戻り、kLabelTextStyleになる追加スタイルを指定できるようになったので、それを選択するだけです。

そして今、あなたはアプリでその高さのテキストに対して同じスタイルを持っているのを見ることができるはずです。しかし、私たちのカードは現在、その子のサイズにサイズ変更されています。

そのため、コンテナが機能する方法は、子がない場合、親が許可する大きさになるように自分自身のサイズを調整しようとすることであることに注意してください。この場合、列の幅全体に引き伸ばされます。ただし、子を受け取ると、子のサイズに合わせてサイズを変更しようとします。

そのため、私のカードの幅が縮小しました。

しかし、これは比較的簡単に修正できます。

一番上のメインカラムに移動するだけで、crossAxisAlignmentを友人のcrossAxisAlignment.stretchに変更できます。そして今では、子のサイズに関係なく、すべての子を引き伸ばして列の幅全体を占有します。高さラベルに加えて、選択した高さの値を追加して、ユーザーがスライダーを切り替えたときにスライダーが移動するときに変化するようにします。

そこで、数字と、cmまたはcmというテキストを少し追加する必要があります。そのために、ここでテキストウィジェットの下に行を追加します。したがって、私の行ウィジェットには2つの子のみがあり、どちらも選択された高さを表示し、2番目はcmという単語を表示するだけなので、両方ともテキストウィジェットになります。私のテキストウィジェットにはいくつかの数字が含まれます。

だから、おそらく文字列180で始まるとしましょう。そして、それはいくつかのスタイルを持つことになります。そして、これを非常に大きくしたいので、私のテキスタイルのフォントサイズは50になり、フォントの重量も非常に大きくなります。それで、おそらく最も重い重量であるw900に設定します。

私のフォントのヘビー級チャンピオン。

そのため、デザインを見てみると、数字が私のアイコンとほぼ同じくらい厚く、その一貫性が非常に気に入っていることがわかります。そのため、たとえば、体重カードや年齢カードを作成するときなど、画面上の他の数字にこの数字スタイルを使用することになります。したがって、これを定数に変えるのは理にかなっています。

ビデオを一時停止し、このテキストスタイルから定数を作成し、ここでこのコードをすべて置き換えます。

大丈夫。

したがって、もちろん、新しいconstants.dartファイルに定数を作成します。

キーワードconstから始めて、K規則を使用して名前を付けます。

したがって、これはkになり、numberTextStyleと呼びます。

あなたが望むものは何でも呼び出すことができます。

そして、ここにあるものをテキストスタイルに置き換えます。

そして、ここでNumberTextStyleと呼びます。

そして今、それはここで私たちのファイルを参照する必要があります。

そして、以前のテキストスタイルで貼り付けます。

そのため、これとこれを多くの設計コンポーネントで再利用できます。

また、メインページでコードが比較的シンプルで簡単になります。番号を作成したので、行に2番目のテキストウィジェットを追加します。これは単にcmと言うだけです。

あなたがアメリカから来て、あなたのユニットとして帝国を持ちたいなら、あなたが望むように番号またはユニットを自由に変更してください。

これで、そのcmには、男性、女性、または身長ラベルと同じテキストスタイルが設定されます。

十分にシンプルで、単にスタイルプロパティをタップし、定数にkを追加して、ラベルテキストスタイルを選択できます。

だから今私は180を持っていると私は私のcmを持っています。

今、あなたはすべてがあらゆる場所にあるということに気づくでしょう。私の身長ラベルはすべて右上に押し込まれ、私の番号列は一番左にあります。

そして、それはあまりよく見えません。

それでは、列のmainAxisAlignmentをcenterに変更して、すべてのコンポーネントを列の中心に移動させましょう。

これにより、すべてが列の中央に配置されます。

また、行に移動してmainAxisAlignmentをcenterに変更することにより、行をメイン軸に沿って中央に配置して中央に移動させます。

そして今、すべてが少なくとも中間にあります。

しかし、私が本当に欲しいのはもう1つあります。cm、cm、または選択した単位が、ここでの私の数字と同じベースラインにあることです。

そのため、実際にはユニットが真ん中のどこかに浮かんでいるのではなく、180 cmのように見えます。

そして、最終的なデザインでは、まさに私が持っているものを見ることができ、同じベースラインにすべて揃えると、より見栄えがよくなります。

これで、crossAxisAlignmentを行に追加するだけで、これを非常に簡単に行うことができます。

したがって、主軸は行の長さに沿っており、交差軸は短辺に沿っているため、行の幅に沿っていることに注意してください。

そして、crossAxisAlignmentを、希望するものと非常によく似たオプションの1つである.baselineに設定します。

そしてこの場合、子を交差軸に沿って配置し、ベースラインが一致するようにします。

先に進んでそれを選択しましょう。

この時点で、保存をクリックしてデザインを確認しようとすると、破損することがわかります。

そして、それが私たちに言う理由は、ちょっと不可解なようです。クロス軸の配置をベースラインに変更する場合、実際にはテキストベースラインプロパティを何かに設定する必要があり、nullにすることはできません。

これをどうやって知るのですか？

実行パネルからこのエラーメッセージを取得して、もちろんGoogleを持っている友人に渡すことができます。そして、これが他の誰かのためにすでに起こっているかどうかを調べるために検索することができます。

そのため、Flutterでエラーコードを取得すると、Googleでそれを検索でき、Stack Overflowまたは実際のFlutter GitHubリポジトリの問題領域に表示されます。そして、ここで実際にFlutterチームの誰かがここで解決策を提供しました。ベースラインを揃えたい場合はtextBaseline引数を指定する必要があるため、彼らはどのベースラインに沿って、そしておそらく将来的にはそのメッセージをより良くすることを計画するために。

そして、これはFlutterコミュニティの素晴らしい点の1つです。開発中ですが、あなたの意見や他のすべての開発者の経験が最終製品に貢献しています。

そして、日々の修正が行われ、変更が行われ、さらに良くなります。

それで、それを試して実装しましょう。

したがって、彼らはtextBaseline引数と呼ばれるものについて話している。

したがって、行にcrossAxisAlignmentがある場合、先に進むと、WindowsでCONTROL + JまたはCONTROL + Qをクリックすると、そのプロパティの1つがまさにそのテキストベースラインであることがわかります。

それでは、crossAxisAlignment.baselineが機能するように設定してみましょう。

それで、textBaselineを追加して、大文字のTでTextBaselineの入力を開始できます。そして、アルファベット文字用のアルファベット文字または基本的にグラフィカル文字用の表意文字の2つのオプションがあることがわかります。

したがって、アルファベットが必要です。保存をクリックすると、エラーがなくなり、行がベースライン全体にきれいに配置されていることがわかります。きれいです。

Web、モバイル、Flutterのいずれかの種類の開発に取り組んでいる場合、エラーや警告、問題が必ず発生します。しかし、あなたを助けることができる強力なコミュニティがあなたの後ろにあり、あなたが一人ではないことを知ってください。

そして、ほとんどの場合、あなたが経験したこと、他の人々も経験しており、おそらく彼らはすでに解決策を見つけました。大丈夫。

ラベルと数値を解決したので、今度はスライダーを追加します。 Flutterにはスライダーと呼ばれるウィジェットがあり、マテリアルデザインスライダーを実装します。そして、それを検索すると、これがおおよそそのように見えることがわかります。それはあなたが動き回ることができる小さな丸い親指を持っています、そして、それは1つのアクティブ側と1つの非アクティブ側を持っています。

クリックすると、このようなアウトラインがポップアップ表示されます。そして、それは私たちが設計に実装したいものと非常によく似ています。

それでは、この設計を作成できるように、先に進んで実装しましょう。ここで、ここで行が終わる場所のすぐ下に、

スライダーウィジェットを追加します。また、スライダーウィジェットには、onChangedコールバックなどの多くのプロパティがあります。

したがって、入力としてdoubleを取り、voidを出力するメソッドが必要です。

そのため、プロパティの1つとして矢印が表示されるときはいつでも、これが機能を期待していることを常に知っています。

また、変更の開始時または終了時も検出できます。

つまり、ユーザーがドラッグを開始し、ドラッグを終了するときに、最小位置と最大位置を設定することもできます。

したがって、温度を設定する場合は0〜30になりますが、他のユーザーの場合はスケールが異なるため、それを判断できます。

そして、あなたが望むかもしれない分割数、スライダーのラベル、そしてアクティブな色と非アクティブな色があります。

ドキュメントによると、アクティブな色は、アクティブなスライダートラックの割合に使用する色です。

クリックすると、もう少し詳細が表示されます。

そして、アクティブな側は実際には親指、つまり中央の円と最小値の間の側であることを示しています。

基本的には、ここの左側の色になります。

そして、非アクティブな色は右側の色です。

それでは、これらのプロパティのいくつかを実装して、スライダーに命を吹き込みましょう。まず最初に、値を提供します。これで、スライダーの値は、ユーザーが選択した高さになります。

そして、この番号で、この中に表示されることも必要です。

それでは先に進み、クラスの別のプロパティを作成します。このプロパティは整数になり、高さと呼ばれます。そして、値180から始めます。

これは、ユーザーがそのスライダーをスライドするたびに常に変化するため、これは最終的でも一定でもありえません。ここに180をハードコーディングする代わりに、ここに高さの値を入れます。

しかし、テキストウィジェットはデータとして文字列を必要とするため、ここに単に数字を入れることはできません。そして、タイプstringに割り当てることができないタイプを示しています。

では、height.toStringを記述して変換してみましょう。

これで、数値から変換された文字列ができました。数値をいつでも文字列に変換したり、intをdoubleに変換したりできます。しかし、反対の方法でやりたい場合は、A BとCが直接数値にマッピングされないため、少し注意が必要です。

しかし、これは簡単です。

そのため、スライダーの値は高さになります。そして、ここで別の互換性のない問題を取得します。

引数型intをパラメーター型doubleに割り当てることはできません。これは、値がdoubleが値として入力されることを期待しているためです。

したがって、できることは、信頼できる.notationを再度使用してdoubleに変更することです。

とても簡単に変換でき、簡単に実行できます。

そのため、スライダーの値として高さが表示されます。

それが基本的に開始値です。さて、次に身長のスケールを定義します。ほとんどの人は身長0センチではないことを願っています。

単細胞生物のようなものでない限り、私は推測しますが、おそらく私たちのアプリを使用することはないでしょう。

最小値120から始めて、最大値220から始めましょう。

センチメートル単位であるため、1メートル20と2メートル20について話しています。これが意味をなさない場合は、明らかにスケールを変更できます。

あなたが住んでいるすべての人の身長が3メートルの場合、気軽にスケールを変更してください。

繰り返しますが、これらは定数として非常にうまく機能します。

したがって、ここで定数に追加することは理にかなっているかもしれません。

ただし、ここでは、スライダーがどのように構築されているかを簡単に確認できるように、ここに残しておきます。

ただし、定数を作成しておけば、1つのファイルを簡単に見つけて、必要に応じてそれらの定数を更新できます。

スライダーが黄色で強調表示されていることがわかります。その場合は、警告が何であるかを示す右の行も表示されます。

また、問題があるか警告が見つかったことを示す小さな黄色のボックスが表示されます。

そのため、この場合、カーソルを合わせると、パラメーターまたはプロパティonChangedが必要であることがわかります。

そのため、現時点では、onChangedというもう1つのプロパティがまだありません。

ここで入力しているように、入力としてdoubleを取り、voidを返す関数を期待していることがわかります。

そのため、reusableCardの場合と同じように、スライダーに関数を渡す必要があります。 onChangedでは、入力のない無名関数を使用する代わりに、実際にはそこにdouble値が必要です。

そして、その値はスライダーの新しい値になります。

そのため、ドキュメントで、ユーザーがドラッグして新しい値を選択しているときに、ドラッグ中にonChangedが呼び出されることがわかります。

さらにクリックして詳細を確認すると、スライダーが新しい値をコールバックに渡し、匿名関数内でその新しい値を利用できることがわかります。

これが、onChangedプロパティの使用を推奨している方法です。

それでは先に進んでそれをしましょう。 onChangedプロパティでは、2倍になります。

そして、newValueと呼びます。

そして、これは、ユーザーがスライダーを変更したときにスライダーから渡される値になります。

そして、この中括弧内で、そのnewValueを利用して、高さを変更する目的で使用することができます。

さて、保存して実行すると、画面上で漠然と表示されますが、実際には背景と区別できないほど暗くなっています。

それでは、色を追加してみましょう。

通常どおりカスタムカラーになるactiveColorを追加しましょう。

この場合、0xFFEB1555になり、左側がピンク色になります。

先に進み、右側にも色を追加します

したがって、inactiveColorおよびこれは0xFF8D8E98になります。

これで、右側にグレーが、左側にピンクが表示されます。

この段階で、非常に奇妙なことが起こります。スライダーのハンドルをクリックしてドラッグすると、スライダーの新しい値が印刷されるため、コンソール内に表示されます。

そのため、値の変更が表示され、一番下の120まで下がって、220まで戻ることができます。

しかし、スライダー自体は実際には画面上で何もしていませんか？アニメーションではなく、まったく動きません。完全にスタックしているようです。新しい値を出力することでこれらの測定値のすべてが得られなかった場合、スライダーが完全に機能していないと思いました。

どうしてこれなの？

ドキュメントに目を通し、onChangedプロパティを見ると、スライダーがコールバックに新しい値を渡すことを示していることがわかります。

そのため、トグルを動かすと、この入力を介してコールバックに新しい値が渡されます。

そのため、コンソール内に出力されている新しい値にアクセスできます。しかし、親ウィジェットが新しい値でスライダーを再構築するまで、スライダーの状態は実際には変わりません。

では、親スライダーを新しい値でどのように再構築するのでしょうか？

さて、コールバックは、これらの中括弧の間にあるすべてのものが、セット状態を使用して親のステータスを更新するはずです。

したがって、親は、ここで示した方法で再構築されます。

したがって、たとえばonChangedの場合、ここでこのコールバックに渡される新しい値があり、その新しい値はここにあるスライダーの値を更新するために使用されます。

これが、スライダーの外観を更新する方法です。

コードでも試してみて、実際にスライダーを動かしてアニメートしてみましょう。新しい値を出力する代わりに、ここにある可変プロパティまたは可変プロパティであるheightプロパティを設定します。

また、180から始まる整数です。

スライダーを変更したら、高さを新しい値と等しくなるように設定します。

しかし、問題は、新しい値はdoubleですが、高さは整数であるということです。

それでは、最も近い整数に変換できるように、整数に切り上げてみましょう。

そして、設定状態を呼び出し、スライダーの状態がこの高さに基づいて更新されることを確認するだけです。

保存ボタンを押して手仕事を確認すると、スライダーの動きと高さを表示するテキストも更新されていることがわかります。

パーフェクト。

スライダーカードのすべての機能が有効になりました。