大丈夫。これで、入力ページのほとんどが完成し、すべてのコンポーネントが期待どおりにこのページで機能するようになりました。

整頓するためのスタイリングが2、3になりました。

たとえば、ここの一番下には、この計算ボタンがあります。理想的には、テキストを少し見やすくするためにフォーマットすることをお勧めします。

それでは、これにスタイルを追加しましょう。

そしてもちろん、ここで定数として作成します。

それで、kLargeButtonTextスタイルと呼ばれる新しいconstを作成するか、好きな名前で呼び出すことができます。

そして、それはテキストスタイルのウィジェットになります。

そして、おそらくフォントサイズを25に、フォントの太さを太字に設定します。

ここで、この定数をテキストに適用します。

それでkそれはLargeButtonTextでした。

よし、それを見てみましょう。

だからそれはずっと良く見える。

そして今、私が必要なのは、テキストをそのコンテナの中央に配置することです。そうすることで、テキストが左端ではなく、ここの中央にくるようになります。

最後にもう1つあります。計算テキストウィジェットは、一番下に少し近すぎます。

特にこれがインタラクティブなエリアであることがわかっているiPhoneでは、ユーザーのターゲットエリアが少し高くなり、誤って終了しないように、その単語を少し上げるといいでしょうアプリ。

そのため、コンテナに少しパディングを追加できます。そして、下部にのみ追加するパディングを追加します。

そして、ここに20ピクセルのパディングを配置します。

そのため、保存ボタンを押すと、実際にセーフエリア内に適切に配置されるように、上に押し上げるのがわかります。

これで、入力ページはほぼ完成しました。

これで、アプリケーションの次のページである結果ページに進む準備ができました。

このレッスンでは主にこれを行います。ここでデモを行い、カードを作成して結果を得るように構築します。必要に応じて結果を再計算できるように、戻るボタンがもう1つあります。

本当にシンプルに見える結果ページをすでに作成しました。そして、計算ボタンをクリックすることでそれを見ることができます。

そのため、実際にこのテキストは真ん中に必要ありません。

これは、これが2番目の画面であることを確認できるようにするためです。

ただし、必要なのは、このページを構成するすべてのコンポーネントです。

それでは、先に進んで構築しましょう。

まず、上から下への垂直レイアウトになります。

そのため、通常どおりに列を追加し、列の間隔を均等にします。

それで、メイン軸の位置合わせのために均等にスペースを取ります。

そして、それが何をするのかを忘れた場合は、WindowsでCONTROL + JまたはCONTROL + Qを押すだけで、この特定の選択が何をするかが正確にわかります。

これで、列に子​​を追加する準備ができました。そして最初の子は、コンテナだけを含む展開されたウィジェットになります。そして、コンテナーには、少しテキストの子があります。

テキストは「あなたの結果」という言葉を言うだけです。

そのため、保存をクリックすると、ほんの少しの本文として表示されることがわかります。

そのため、少しスタイルを整える必要があります。

2番目のページのユーザーインターフェイス全体を作成している間、私がしていることは、このページに表示されているデザインに従っているだけです。少しのコンポーネントの負荷を追加するために年齢を費やさないように少し簡略化しましたが、デザインの大部分をここで見たものと一致させました。

このデザインをすべて自分で作成したい場合は、この画面をガイドとして使用して、自分でデザインを最初から作成してください。

しかし、あなたが私と一緒にフォローしたい場合は、私たちが同じコードを持ち、お互いに一貫性を保つように、すぐにスタイルを整えます。

テキストを追加したので、少しスタイルを追加します。そして、スタイルを追加するときは、定数を単に定数に追加する方が簡単です。少し後で使用したい場合があるからです。

そのタイトルに少しスタイルを追加しましょう。

したがって、const kTitleTextStyleを追加し、これがテキストスタイルウィジェットになります。そして、私が設定するフォントサイズをかなり大きく、おそらく50にし、フォントの太さを太字にします。大丈夫。

したがって、定数ファイルをインポートする限り、結果ページでこのkTitleTextStyleを使用できます。

それでは、それを追加しましょう。

大丈夫。

これを追加したので、チェックしてみて、このように大きく見えるはずです。先に進んで、次のウィジェットを列内に追加しましょう。

したがって、拡張ウィジェットの後に、別の拡張ウィジェットを追加しますが、このウィジェットはさらに大きくなります。

このコラムのスペースをもっと取って、このコラムまたはこのコラムの5倍のスペースを占めるようにします。

そして、それを行うために、flexプロパティを使用します。

そのため、デフォルトでは、すべての拡張ウィジェットは1のフレックスを取得するため、均等にレイアウトされます。

しかし、この場合、これに5のフレックスを与えて、より多くのスペースを占有するようにします。そして、この展開されたウィジェットに含まれる子は、再利用可能なカードウィジェットをインポートする場合にのみアクセスできる再利用可能なカードウィジェットになります。

したがって、実際に必要なファイルだけを追加しています。

それで、再利用可能なカードを作成してみましょう。それは必須プロパティとして色を持っています。

そこで、アクティブなカードの色である定数を入れます。

それを確認して、必要なだけのスペースを使用していることを確認してください。

そのため、現時点では、これは画面上の使用可能なスペースの1倍を占めており、これはこの高さの5倍を占めています。

これで、カードの子を再利用可能なカードに追加する準備ができました。

そして、これには別の列が含まれます。

それで、列の中に列ができます。

この列には、再び等間隔に配置される主軸があります。

また、すべてを水平軸の中央に配置するクロスアライメントを追加します。

そして、テキストウィジェットになる子を追加する準備が整いました。

したがって、最初のテキストウィジェットはBMIの結果になるので、太りすぎ、普通、または減量と言います。

そして、2番目のテキストウィジェットは、小数点以下1桁に丸められた実際の読み取り値になります。

そして最後に、ユーザーがこのBMIの意味を実際に理解できるように、体重の解釈があります。

先に進んで、最初のテキストウィジェットを追加してみましょう。これは単に結果になります。

だから私はここで多分普通の文字列から始めます。そして、比較的大きなスタイルを指定し、resultTextStyleと呼びます。

この特定のデザインから選んだので、これは緑色になります。

そのため、色はカスタム16進コードになり、0xFFになるため、完全に不透明になり、24D876になります。そして、あなたはそれが緑の素敵な明るい石灰として現れているのを見ることができます。

これで色が落ちたので、フォントサイズを追加します。約22にしましょう。

そして、フォントの太さが再び太字になります。

ここにkを追加するのを忘れたので、kResultTextStyleに変更します。

大丈夫。

これで、ここで結果ページにこれを適用できます。

したがって、kResultTextStyleになります。

それを確認しましょう。

大丈夫。

だから、かなり見栄えがいい。しかし、私たちのカードは現在、非常に小さなサイズに縮小されています。

したがって、それを引き伸ばしたい場合は、クロス軸の配置プロパティを追加して、画面全体に引き伸ばすだけです。

そのため、カードは可能な限り幅を占有します。

そして、中央には通常の解釈があります。

これが最初のテキストウィジェットであり、2番目が実際のBMI読み取り値になるため、列内の別のテキストウィジェットになります。そして、18.3、またはあなたの頭に浮かぶどんな数字でも、これから始めましょう。

また、テキストウィジェット内で必要なものであるため、数値ではなく文字列として表示されます。そして、これも同様にスタイリングします、そして、それはここの一番下に行きます。

したがって、これはkBMITextStyleになり、非常に大きくなります。

そのため、フォントサイズを約100に変更して、それを超大きくし、フォントの太さも太字に変更して、画面に最も重点を置くようにします。これがkBMITextStyleです。そして、あなたはそれが絶対に巨大であることを見ることができます。

あなたが画面に向かうとき、あなたはそれを見逃すことはできません。

ここで必要なのは、最終的なテキストウィジェットだけで、これが解釈になります。

この場合、「BMIの結果が非常に低い。

もっと食べるべきだ」

もちろん、これもスタイルを整えます。

kBodyTextStyleを追加してみましょう。これは少し小さいスタイルになるので、フォントサイズは約22になりますが、それ以外は変更しません。それでは、先に進んでKBodyTextStyleを追加してみましょう。今すぐ保存をクリックすると、デフォルトですべてのテキストが左に揃えられていることがわかります。

したがって、テキストの配置を変更する場合は、テキストウィジェット内で変更する必要があります。

したがって、テキストの配置を確認し、中央に配置します。そして、このように、それは私たちのデザインとより一貫して見えるように真ん中に行きます。ここで最後に必要なことは、ここの下部にあるボタンです。ユーザーが何か間違ったことを入力したと思う場合、BMI結果を再計算できることをユーザーに伝えます。

これは、以前の計算ボタンであるボタンとまったく同じになります。

したがって、最初から作成するのではなく、抽出したいウィジェットを選択してFlutterアウトラインに移動します。ここでも強調表示され、列のツリーの最後に表示されます。

それで、それを右クリックしてウィジェットを抽出しましょう。

単に下のボタンと呼びます。

好きなものを呼び出すことができます。

そして、リファクタリングをクリックします。

これで、下のボタンが表示され、ポップアップされた場合は新しいキーワードを削除できます。

そうでない場合は、Flutterチームがツールを更新し、以前のバージョンのDartを使用していないことを意味します。

したがって、最新バージョンのDartには新しいものはありません。

そして、一番下のボタンのコンストラクターを削除し、カスタムのonTapといくつかのカスタムテキストを使用してセットアップします。これらは最初のページと2番目のページで異なる部分だからです。

そこで、ここにいくつかのプロパティを追加しましょう。

したがって、onTapと呼ばれる最終関数があり、buttonTitleとなる最終文字列もあるので、bottomButtonとなるカスタムコンストラクターを作成します。 2つの入力を取得します。ボタンであるため、両方とも入力する必要があります。

おそらくいくつかのテキストが必要であり、おそらくタップを処理する必要があります。

このため、this.onTapおよび必要なthis.buttonTitleを追加します。

これで、ジェスチャ検出器に入ることができます。少し後で必要になるので、これらすべてを単純に切り取ります。代わりに、bottomButtonに渡されるonTapを使用します。また、buttonTitleから渡されたテキストも使用しますが、これは完全にリファクタリングされました。

ここに入り、これらのプロパティを追加できます。

つまり、すべての大文字でCALCULATEという単語になるボタンタイトルがあり、それからonTapがあります。これは、以前持っていたすべてのものを貼り付ける場所です。

そのため、次の画面（結果ページ）をナビゲーションスタックにプッシュするのは、そのコールバックです。

また、便利な場合は、コンマを追加してDartが適切に再フォーマットできるようにすることができます。

そこで、これをすべて入力ページから削除します。そして、新しいDartファイルを作成して、bottom\_buttonという名前を付けて、クラス全体に貼り付ける必要があります。

それをここに追加しましょう。

そして再び、すべてが機能するようにマテリアルパッケージをインポートし、これらの色とサイズとスタイルが何であるかを知るために定数もインポートします。

そして、私がそれをしている間に、他の場所でも必要な場合に備えて、丸いアイコンボタンを分離することもできます。

入力ページ内に保持し、そこで保持することは意味がありません。

round\_icon\_buttonを作成しましょう。ここでこれを切り取り、この新しいファイルに貼り付けます。

これで、マテリアルパッケージを再びインポートできるようになりました。それだけで十分です。

インポートページに移動し、bottom\_button.dartをインポートして、下のボタンが何であるかを認識できるようにし、また、round\_icon\_button.dartファイルをインポートして、丸いボタンが何であるかを認識できるようにします。

そのため、このページにはエラーがなくなり、このページ全体がユーザーインターフェイスとアプリの入力画面の機能専用になりました。

必要に応じて、実際に非常に簡単にフォルダーを追加してlibを整理できます。想像できるように、より多くのカスタムコンポーネントとより多くの画面を作成すると、少し面倒になります。したがって、新規ファイルに移動し、新しいディレクトリを選択できます。これをコンポーネントと呼ぶこともできますし、画面を保持する新しいものを作成して、これを画面と呼ぶこともできます。

そして、入力ページと結果ページを画面に移動し、参照の検索を常にチェックして、すべてが正しく移動するようにします。

また、コンポーネントを移動し、下のボタン、アイコンのコンテンツ、再利用可能なカード、丸いアイコンボタンをコンポーネントフォルダーに移動して、すべてがきれいに見えるようにします。

コンポーネントや画面はありますが、エラーもあります。そして、エラーは同じフォルダ内の定数を正しく見ることができなくなったためです？

以前、bottom\_button.dartと定数は、すべて同じフォルダー内にありました。

現在、実際には異なるレベルにあります。

したがって、あなたがしなければならないのは、それを削除して再入力することです。

したがって、定数から始めて、それを検索して、パッケージbmi\_calculator / constants内で見つけることができるようになり、すべてのエラーがなくなります。

それでは、これをコピーして、定数ファイルを見つけるためのエラーが発生しているすべての場所でコピーし、画面でも同じ問題が発生します。

コンポーネントフォルダ内に移動されたアイコンコンテンツがあります。移動された再利用可能なカードなので、これらすべてを再構築し、コンポーネントを使用している他のすべての場所でも同じようにする必要があります。

そのため、インポートでエラーがなくなり、すべてのコードが正常に機能します。

最後は結果ページです。

それでは、定数の場所と再利用可能なカードを追加してみましょう。

大丈夫。

すべてをリファクタリングして整理したので、今度は実際にその最終コンポーネントをここの列に追加します。

そのため、最後に展開されたウィジェットの後に、下のボタンを追加します。

コンポーネントフォルダー内にある下ボタンをインポートする必要があります。ここで下ボタンウィジェットを追加するために使用できます。

そして、「再計算」と言うボタンのタイトルを追加します。

少し違うので、戻ったということです。そして、コールバックを行うonTapを作成します。

ユーザーがこの下のボタンをタップすると、この2番目の画面であるこの結果画面から飛び出し、この前の画面に戻って新しい入力を追加して再計算できるようになります。

そのため、ナビゲーションとルートについて学習したときに見たように、一番上にある画面を破壊してポップオフするのは本当に簡単です。

Navigator.popを作成して、現在のコンテキストを渡すだけです。

したがって、保存をクリックしてBMIを計算したら、再計算をクリックして、入力画面に戻ることができます。最後に行うことは、単にこの上部のテキストウィジェットを微調整して、カードに少し近づき、左から少しマージンがあるようにすることです。

それでは、そのコンテナウィジェットに進み、いくつかのパディングを追加してみましょう。パディングを15前後ほど追加して、左上から少し離れるようにします。それから、左下に少し下がるようにアライメントを追加します。

それでは、保存してみましょう。

そして今、それははるかに一貫性があります。カードに近く、上部から離れているため、すべてがうまくグループ化されているように見えます。

入力ページと結果ページの両方でほとんどすべての設計が完了したので、欠落しているのはたった1つで、実際の機能です。 BMIを計算する実際の能力。

それで、それ以上については、次のレッスンで見ます。