大丈夫。

したがって、GitHubからスケルトンプロジェクトをチェックアウトし、Android Studioでパッケージを取得して開くという通常の手順をすべて完了すると、次のような画面が表示されます。

また、現在、main.dartファイルは1つしかありませんが、今回は開始コードが少ししかありません。 BMI計算機は、入力ページをホームページとするマテリアルアプリを返すステートレスウィジェットとして用意されています。また、入力ページはステートフルウィジェットであるため、入力ページは変更および更新できます。そして、入力ページの状態の内部には、BMI計算機のみを示すアプリバー、「本文」のみを示す中央のテキスト、およびダミーのフローティングアクションボタンを備えた足場があります。

そして、これらのものがここにある理由は、私があなたに何かを示すことができるようにするためです。そして、この最初のレッスンで話したいのはFlutterテーマです。

これは、単に、アプリのルックアンドフィールに合わせてカスタムカラーパレットとカスタムスタイリングを定義できるようにする方法です。

つまり、ブランド、会社、またはスタートアップ向けにアプリを作成する場合、さまざまな画面で一貫性を保ち、独自のカスタムスタイルと独自の色を定義して、全体で一貫性を保つことができます。別のページ。

Flutterテーマについての背景を最初に読むことに興味がある場合は、コースリソースのこのページにリンクします。

そして、これはクックブックと呼ばれるFlutterのドキュメントのセクションから来ています。そして、クックブックは基本的に彼らがどのように聞こえるかです。基本的には、特定のことを達成する方法を説明するレシピが満載の本です。

たとえば、サウンドを再生する場合は、まずこれを実行する必要があり、次にステップ2でそれを実行する必要があります。

そして最後に、必要なすべてのコードを書き終えたら、最後にポテトチップレシピに似た最終目標を達成できるはずです。 Flutterチームはクックブックに大量のレシピを作成しました。あとはflutter.dev/docs/cookbookにアクセスするだけで、できることをすべて確認できます。

そして、ここには絶えず新しい追加と更新があります。

ですから、カスタムアプリで何か特別なことをしたいときにチェックアウトするのは本当にクールな場所です。

テーマに戻りましょう。アプリのテーマを作成します。

アプリ全体でスタイルと色を共有できるようになります。

そのためには、マテリアルアプリにThemeDataウィジェットを提供する必要があります。

そのため、マテリアルアプリを作成するときに、タイトルや家を持つことができることに加えて、テーマを指定することもできます。

そして、これはThemeDataウィジェットであり、多数のプロパティがあります。

また、チェックアウトすると、それらのプロパティが何であるかを確認できます。

そして私が言ったように、あなたが通過するためのそれらの負荷があります。アプリバーのテーマ、アプリバーの下部の色、明るさのボタンの色など、新しいボタンを作成するときやアプリを作成するときにデフォルトの色を使用できるように設定できるものすべてバー。

そして、これはアプリのすべてのページで共有されます。

このモジュールで作成しているアプリでは、初めて複数のページが実際にあるため、これは非常に重要です。

複数の画面があります。

したがって、テーマでハードワークを行うと、アプリ全体でそれを使用できるようになります。 GitHubからチェックアウトした後すぐにスケルトンプロジェクトを実行すると、まだ実装されているスタイリングがないことがわかります。

そのため、アプリバー、テキスト、floatingActionボタンなど、画面に表示されるすべてのものは、デフォルトのFlutterテーマを使用してスタイル設定されます。つまり、アプリバーのマテリアルは青色で、アプリバーのテキストは白ですが、足場の本文のテキストは黒です。

そして、floatingActionButtonも同じ青色です。また、フローティングアクションボタンの内側にあるもの、その子はすべて通常は白です。

これはデフォルトのフラッターライトテーマです。

このテーマを使用する場合は、何もする必要はありません。

マテリアルアプリのテーマプロパティに何も指定する必要はありません。ただし、アプリのテーマをカスタマイズする場合は、themeという新しいプロパティを追加するだけです。ご覧のとおり、これはテーマデータウィジェットであるウィジェットを想定しています。また、テーマデータは、このアプリ内に存在するすべての色、フォント、形状などの視覚的プロパティを定義できます。

テーマを追加してみましょう。テーマの入力を開始すると、あらかじめパッケージ化されたデフォルトのテーマがいくつかあることがわかります。それはとにかくすべてのマテリアルアプリのデフォルト値であるため、通常選択する必要のない軽いものです。しかし、暗いものもあります。

ENTERキーを押して、ダークテーマをマテリアルアプリに追加し、保存ボタンを押すと、どのように見えるかを確認できます。

そのため、背景は暗くなり、アプリバーは非常に暗くなり、フローティングアクションボタンには、ターコイズグリーンのような素晴らしいアクセント色が付きます。

この特定のテーマが必要な場合は、テーマをダークテーマに変更するのと同じくらい簡単です。

しかし、もっとカスタムにしたい場合はどうしますか？

背景色を何にするか、アプリの色を何にするか、アクセントトーンを何にするかを定義したい場合はどうでしょうか。

それでは、独自のテーマデータウィジェットを作成する必要があります。

それが今私たちがやろうとしていることです。前述したように、テーマデータウィジェットには、変更可能なプロパティが多数あります。

そして、あなたが変更できるすべてのものと、これらの各プロパティが実際に何をするかを見ることができるようにするために、このドキュメントを実際に読んでおくのは良い考えです。

そのため、たとえばアクセントカラーは、ノブ、テキスト、オーバースクロールエッジエフェクト、そしてもちろんフローティングアクションボタンなどのウィジェットの前景色を変更します。

そして、ボタンの色など、他にもあります。ボタンの色は、上げられたボタンで使用されるマテリアルのデフォルトのフルカラーです。

そのため、これらの各プロパティを変更するとどうなるかについての詳細な説明があります。

始める前にそれを読んで、回すことができるすべての異なるノブが何であるかを理解したら、変更できるものとアプリをカスタマイズする方法を知るのがはるかに簡単です。今、私たちの場合、私が変更しようとしているプロパティは、これがおそらく最も頻繁に変更されるプロパティであり、原色になります。

プライマリカラーは、ツールバーやタブバーなどのアプリの主要部分の背景に与えられる色です。

しかし、これには、例えば足場の本体が含まれていないことに注意してください。

先に進んで原色をaに変更したら、わかりません。デフォルトとは違うものが赤色と言って、保存してみましょう。

そして、最初にテーマのデフォルトがFlutter lightテーマになっていることがわかります。変更したのは、実際にトップアップバーを変更して、背景色を赤にすることだけです。

これで、アプリ内にタブバーや下部アプリバーなどの他のものがある場合、それらも同じ原色を持つように変更されます。

ここでアクセントトーンを変更する場合はどうでしょうか。

もちろん、ここに入ってアクセントの色を変更することもできます。

たとえば、これを紫色に変更しましょう。

その様子をご覧ください。

そのため、すべてのフローティングアクションボタンの塗りつぶし色が、ここで選択したアクセント色に変更されました。

そして、ブランドや会社、アプリに必要なルックアンドフィールを実現するまで、ここやここに表示されるすべてのプロパティをいじることができます。そして、あなたができるカスタマイズの程度は本当に非常に広範囲です。

これまでは、マテリアルパレットのデフォルトのマテリアルカラーを常に使用してきました。そして、それを使用するために、私たちは常に色をタップしてから、パレットで特定の色合いまたは特定の色調を単に選択しました。しかし、マテリアルパレットの色よりも多くの色があるため、独自のカスタム色を使用したい場合はどうでしょうか。

その場合、16進コードを取得することから始めたいと思うかもしれません。

したがって、たとえば、ここで複製しているデザインには、この素敵な暗い黒がかった紫の色が背景としてあります。それは正確に黒ではなく、正確に紫ではありませんが、その間の素敵な色合いです。

今、私が非常に頻繁に使用するクールなツールの1つは、ColorZillaと呼ばれる無料のツールです。

ChromeとFirefoxの両方で動作します。そして、それはあなたのブラウザに小さなカラードロッパーを追加します。

つまり、クリックすると、ページから色を選択できます。

たとえば、このページからこの黄色の色合いを取得したいので、アプリやウェブサイトで使用したい場合は、それをクリックするだけで、その色の16進コードがクリップボードにコピーされますその後、実装できます。

これをインストールする場合は、これをクリックしてChromeまたはFirefoxにインストールするだけです。

そして、それは非常に使いやすいです。

それを実行してみましょう。

カラードロッパーを使用して、ページから色を選択します。そして、このアプリの背景である色を選択してみます。

そして、それは0Aの何かが出てきます。ここに貼り付けて、原色を変更します。

ここから取得するデフォルトのマテリアルカラーパレットカラーではないものを使用するには、別のものを使用する必要があります。

それを削除して色を書き始めると、原色として使用する色を作成できるさまざまな方法がたくさんあることがわかります。

そして、16進コードを使用している場合に使用する最も一般的なものは、単語colorを単純に記述し、括弧内に値を追加する最初のものです。

そして、前からコピーした16進コードを貼り付けるだけです。

Flutterドキュメントでカラークラスを見ると、ARGB形式で32ビットのカラー値を期待していることがわかります。それはどういう意味ですか？

Aは色のアルファまたは透明度です。

そのため、この緑の色などの色を選択すると、完全に不透明から完全に透明になり、色の大小を表示できます。

また、カラークラスを使用するときに指定することもできます。ほとんどの場合、完全に不透明な色を使用します。

そして、完全に不透明な値は0xFFです。これは、多くの色で見られる値です。

そのため、16進コードをコピーするときは常に、Rのコードのみが含まれるので、赤、G、緑、最後の2つの値は青であるBを表します。

したがって、赤、緑、青の3つのチャネルすべてで、各色に異なる量を指定すると、特定の色になります。

したがって、たとえば、赤はたくさんあるが緑はあまり多くなく、青もたくさんある場合、紫色のような色になります。通常、色はハッシュ表記を使用して表され、ここに16進コードがあります。 16進数のカラーコードを使用したことがない場合は、このスタックへのリンクを追加します

16進数を使用してさまざまな値を表す方法を説明する上で非常に優れた機能を果たすオーバーフローページ。

そして、これらすべての文字と数字がどこから来たのかを理解し始めるのに良い場所です。しかし、本質的には、それは特定の量の赤、緑、青をエンコードし、それらを一緒に混合すると特定の色が作成されます。 16進コードをコピーすると、通常は次のようになります。

6つの値があり、先頭にハッシュタグまたはポンド記号が付いています。

Dartで使用するための最も簡単な方法は、16進コードを貼り付け、パンド記号を削除して0xFFを追加することです。これにより、色が不透明色になり、残りの6文字がそのまま残ります。実際の色の色相をエンコードするためです。

では、保存してから、アプリがどのように見えるかを確認しましょう。

原色によって決定されるアプリバーが表示され、ドリブルデザインで見たものとまったく同じ色に変更されました。

それでは、足場の本体の背景色も変更したい場合はどうでしょうか？

さて、ここに課題があります。

テーマデータクラスのドキュメントを使用して、変更できるプロパティを調べて、ここで背景に影響を与える可能性のあるものを見つけてみてください。

ドキュメントを巡回する最も簡単な方法の1つは、COMMAND + FまたはCONTROL +

Fキーを押して、関連するキーワードを見つけます。

したがって、このケースでは、足場本体の背景を変更します。

そして、背景の検索を開始すると、表示されるすべての場所が強調表示され、背景色のプロパティがあることがわかります。次に、ダイアログの背景色がありますが、足場の背景色もあります。

そして、これが説明するように、これは典型的なマテリアルアプリの背景色です。

そして、これはまさに私たちが変えたいものです。

ここにそのプロパティを追加しましょう。

したがって、原色に加えて、足場の背景色を変更し、以前の値とまったく同じになるようにします。

それでは、保存してみましょう。足場の背景を同じ紫がかった黒っぽい色に変更しました。しかし、今ではページの中央にあったテキストが中央に完全に消えてしまったため、少し問題があります。

そして、その理由は、テキストのデフォルトの色が黒であり、黒が黒の隣にあるとき、それはちょっと見にくいからです。

したがって、白色に変更する必要があります。そのためには、テーマデータのテキストの色を単に変更することはできません。なぜなら、本文テキストや見出しテキストを変更できるさまざまな種類のテキストがあるからです。

したがって、アクセスできるプロパティはテキストテーマと呼ばれるものです。これには、テキストテーマウィジェットと呼ばれる特定のウィジェットが必要です。ここで、変更するテキストの部分を指定できます。

キャプションですか？ボタンですか？

見出しですか？

私たちの場合、実際にはbody1です。

したがって、これはマテリアルアプリ本体のデフォルトのテキストスタイルです。

body1プロパティを選択してみましょう。これはテキストスタイルを想定しています。アプリ内でテキストのスタイルを変更する方法と同様に、テーマ内でも同じことができます。そこで、色を白色に変更します。

ここではマテリアルカラーパレットを使用します。これは、はるかに簡単だからです。その後、保存をクリックします。

そして、テキストが表示されるようになったはずです。テキストが白になり、コントラストがあります。

カラーパレットを使用したくない場合のもう1つのオプションは、以前に行ったことである色を使用することもできます。これは、白い色にするために必要なことを覚えておくのは良いことですdoは8つのFを追加するだけです。 1 2 3 4 5 6 7 8

そして、保存をクリックすると、colors.whiteを使用するのとまったく同じことです。

読みやすいと思う方はあなたの選択です。

これまでは、暗い背景と暗いアプリも使用しているため、暗いテーマにはテーマに必要なものがほとんど揃っていることがわかりました。

では、暗いテーマで既に指定されているすべてのものを使用して、特定の背景色や特定のアプリバーの色など、特定の部分を好みに合わせて更新するにはどうすればよいでしょうか？

さて、copyWithと呼ばれる、本当にすてきなトリックがあります。

私たちにできることをお見せしましょう。

先に進み、テーマデータウィジェットにあるすべてのものを切り取ります。そしてテーマとして、私はその暗いテーマに戻ります。

ダークテーマを指定した後、ドットを記述し、copyWithメソッドを選択します。

そして、ここでは、ダークテーマから実際に変更したいすべてのものを指定できます。

そのため、私たちの原色である足場の背景色が含まれます。

私は実際にあなたが暗いテーマから得たものを非常に気に入っているので、アクセントの色を削除します。

また、暗いテーマがすでにすべてのテキストを白に変更していることがわかっているため、テキストテーマのすべてを削除します。

したがって、これにより、テーマデータに含まれるコードが大幅に簡素化されます。

そして、保存をクリックすると、デフォルトのダークテーマのアクセントトーンがあるという事実以外は、結果がほとんど同じに見えることがわかります。テーマプロパティを変更することで、マテリアルアプリレベルでアプリ全体のテーマを作成できるようになりましたが、実際には、特定のウィジェットのテーマをより細かく制御することもできます。

たとえば、この特定のページでフローティングアクションボタンのテーマを変更する場合、このフローティングアクションボタンだけにカスタムテーマを実際に指定できます。したがって、このフローティングアクションボタンを埋め込むことができると言えます。

したがって、意図アクションを使用し、フローティングアクションボタンを新しいウィジェット内にラップします。このウィジェットは、常にデータのプロパティを持つテーマウィジェットになります。そして、これにはいくつかのテーマデータが必要です。

ここで、テーマを変更できます。そのため、アクションボタンが青色になるライトモードに変更したい場合があります。または、アクセントカラーをカスタムカラーに変更するカスタムテーマが必要な場合もありますが、そのパープルカラーに戻ることができます。そのため、この場合、アプリ全体のテーマはカスタムの原色とカスタムの背景色を持つ暗いテーマですが、実際にはウィジェットをテーマウィジェット内にラップするだけで、ウィジェットのテーマを変更できます。

これにより、特定のページや特定のウィジェットをカスタマイズするためのより詳細な制御が可能になります。

ただし、実際にはアプリにフローティングアクションボタンは必要ありません。

したがって、テーマウィジェットを含むフローティングアクションボタンプロパティにあるすべてのものを削除します。これで、実際に電卓のデザインを作成する次のステップに進む準備ができました。

しかし、その直前に、メインページと入力ページを分けたいと思います。

以前は、1つのページにすべてを常に作成してきましたが、これは一種のマンボジャンボです。

しかし、今は、複数のルート、複数の画面を作成する方法を示したいと思います。また、コードをもう少し整理しておく必要もあります。したがって、コードが大きくなると、ページ内の多くのスペースを占有します。

そのため、通常、個々のコンポーネントを個別のDartファイルに実際に分離することをお勧めします。

その方法は、libフォルダーを右クリックして、新しいDartファイルを作成することです。そして、このinput\_pageを呼び出します。次に、[OK]をクリックして、input\_page.dartを作成します。また、Gitに追加するかどうかを尋ねられた場合は、[はい]または[いいえ]をクリックできます。

この場合、実際には問題ではありません。

ここにinput\_page.dartがあり、ステートフルウィジェットと入力ページ状態クラスをmain.dartから切り取り、ここに貼り付けます。

ですから、最初に目にするのはエラーの束です。これは、新しいスケルトンプロジェクトを開くときと非常によく似ているため、少し馴染みがあるかもしれません。それを解決するために必要なのは、material.dartパッケージをインポートすることだけです。ステートフルウィジェットが何であるか、足場が何であるか、その他すべてを把握できるようにします。

しかし、今では入力ページが何であるかわかりません。

さて、この特定のページにどのようにアクセスできますか？繰り返しますが、秘密は輸入品にあります。

インポートの新しい行を入力し、このページにinput\_page.dartをインポートして、ウィジェットの入力ページ状態にアクセスできるようにします。 Aそして、マテリアルアプリのホームとして作成できます。

Flutterアプリに出くわすと非常に頻繁に、このように構成されていることがわかります。

通常、最初のmain.dartファイルには多くのテーマが含まれています。

それでは、カスタムフォントとは何ですか？

カスタムカラーとは何ですか？

名前が付けられるメインアプリは何ですか？

しかし、その後、最初の画面に向けられ、その後のすべての画面に対して、個別のDartファイルが必要になります。そして、この現在のモジュールでより多くの画面を作成するにつれて、それを見ることになります。

しかし、今ではアプリを2つの別個のページに分割し、アプリのテーマは既に作成されています。

したがって、このページをそのままにして、はるかに短くて簡潔なinput\_page .dartファイルに集中できます。それが次のレッスンで継続するところです。

それで、そのすべてについて、次のレッスンでお会いしましょう。