大丈夫。

これで、さまざまな方法で位置データから天気予報を取得することがほぼ完了したので、次のステップは都市画面に移動することです。そして、それはscreensフォルダーの下にあります。そして、ユーザーがこの都市アイコンをタップすると、そこに移動して、都市画面に移動することができます。このアイコンは、アイコンがicons.location\_cityであるこのフラットボタンにあります。ルートとナビゲータを使用して、この現在地画面の上に都市画面をプッシュする方法を覚えているかどうかを確認してください。これを行うために必要なコードを思い出せるように、簡単なグーグルを実行する必要があるかもしれません。しかし、ビデオを一時停止し、それをやってみてください。

大丈夫。

そのため、新しい画面をプッシュするには、ナビゲーターを使用する必要があります。そして、現在のコンテキストを渡すpushメソッドを使用して、マテリアルページルートを構築します。

今、このビルダーはコールバックを期待しているので、入力としてコンテキストを取り、コールバック内で何を返したいのか、つまりプッシュしたい画面を言います。そして、これはもちろん街のスクリーンになります。

そして、現在のファイルを認識させるためには、もちろんそれを一番上にインポートする必要があります。

したがって、city\_screen.dartファイルをインポートすると、都市の画面が何であるかが認識されるはずです。

そして最後に、セミコロンでこれを終了し、コンマを追加して、dartがこのコードを再フォーマットして読みやすくできるようにします。

それでは、保存をクリックして、チェックアウトしてみましょう。

先に進み、市のアイコンをクリックすると、次の画面が表示されます。この画面はごく簡単です。

持っているのは、前の画面に戻るための単一の戻るボタンだけです。

また、天気を取得するボタンもあります。

ここで必要なのは、ユーザーが天気を取得する都市を実際に入力できるテキストフィールドです。

Flutterクックブックには、テキストフィールドに関連する記事がたくさんあります。テキストフィールドは単なるテキスト入力ウィジェットです。したがって、UIのスタイル設定に使用している他のすべてのウィジェットと同様のウィジェットです。

また、装飾やスタイル、境界線などのさまざまなプロパティを使用して、外観を変更できます。

それでは、今それをしましょう。

テキストフィールドウィジェットを作成しましょう。

したがって、コードを編集するときにスタイルをホットリロードおよび更新できるように、アプリを右側に維持します。

市の画面に移動して、現在子としてnullが含まれているこのコンテナ内で、テキストフィールドを含むように変更します。

このテキストフィールドを保存すると、1行だけで表示されます。そして、マテリアルアプリ用に最初に定義したDartスタイルに従っています。

最初にやることは、decorationプロパティを変更することです。そして、これは入力装飾と呼ばれるウィジェットを期待していることがわかります。入力デコレーションには、コンテンツのパディングなどの変更可能なプロパティや、フローティングプレースホルダーやヒントテキストなどが含まれる多数のプロパティがあります。

変更するのは、fillプロパティを変更することです。これにより、コンテナが1行ではなく色で塗りつぶされます。

それでtrueに設定して、保存をクリックします。

そして今、私たちはテキストフィールドを少し満たす必要があります。

そして、塗りつぶしの色を白色に変更しましょう。

そして今、それはこの種の透明な白から完全に不透明な白に変わるはずです。

そして、フィールドにもアイコンを付けます。

そして、これは、Icons.location\_cityになるアイコン作成者を使用します。

そのため、前と同じアイコンが、右にあるテキストフィールドに移動して、ここに何が入るべきかを示します。

次に、現在のテーマに合うように、アイコンの色を白色に変更します。

それでは、colors.whitesで保存してみましょう。これで、アプリの他の部分とのマッチがより一致するようになります。そして、ヒントテキストと呼ばれるものを追加します。

これがプレースホルダーテキストで、ここに入力する内容をユーザーに伝えるためにここに入ります。

また、ヒントテキストには「都市名を入力してください」と表示されます。保存すると、何も変わらないことがわかります。

その理由は、デフォルトではヒントテキストが白だからです。

そのため、色を変更するにはヒントスタイルを変更する必要があります。そして、ヒントスタイルは値としてテキストスタイルウィジェットを取ります。

そして、ここで色をcolors.grayに変更できます。

次に、保存をクリックして、テキストがそのように表示またはヒント表示されるようにします。

そのため、現在、テキストフィールドにこの種の奇妙な形があります。

一番下に1本の線があり、それから長方形になっています。

境界線を追加して、それを締めくくりましょう。

したがって、入力宣言のborderプロパティを利用し、アウトライン入力ボーダーを使用します。

そしてもちろん、これはFlutterのドキュメントでそのborderプロパティのドキュメントを読むことから来ています。

次に、境界線の半径を追加します。この半径は、4つの側面すべてをカバーする半径に設定します。

そして、それは円形の半径になるので、radius.circular。

そして、ここで入力する値は、半径が約10です。そして、コンマを追加して、保存をクリックします。

そのため、実際のテキストフィールドのように見えるようにすべての角が丸くなり、境界線を追加できるようになりました。

そこで、境界線側をnoneに変更して、BorderSide.noneを使用して、その下の行を削除しようとします。そのため、ここでそのボーダー側を削除し、通常のテキストフィールドのように見えます。したがって、最後に修正しなければならないことは、以前はこの白い塗りつぶし色のない暗い背景があったということです。

そのため、入力時に白いテキストが入力された場合、完全に表示されます。

しかし、背景と同じ色であるため、表示されなくなりました。

それでは、テキストフィールドに移動して、そこに入るテキストスタイルを更新しましょう。

そこで、テキストスタイルを追加し、これに黒の色、つまりcolors.blackを設定します。

保存してみましょう。テキストが表示され、完全に表示されるはずです。これらすべてのものを作成したので、テキストフィールドではほぼ一定になりますが、この入力装飾をすべて取得して定数ファイルに移動してみませんか？ユーティリティで定数ファイルを開いて、新しいconst変数を追加します。これはkTextFieldInputDecorationになり、今作成した入力装飾に等しくなるように設定して、それをキャップします。セミコロン付き。

そして、都市の画面に戻って、装飾をkTextFieldInputに変更します

デコレーション、そしてできればセーブをヒットしても何も変わらないでしょう。

ここで、テキストフィールドからここに入力する値を取得する方法を教えてください。ユーザーが入力した都市の場所の天気を取得できるようにしたい場合、それは基本的に必要なものだからです。

たとえば、ロンドンという単語を入力したとしましょう。

どうすればその価値を引き出すことができますか？

テキストフィールドには、onChangedというプロパティもあります。

ボタンにonTap、onPressed、またはonHoldというプロパティがあるように、テキストフィールドにonChangedがあります。これにより、テキストフィールドにある値がトリガーされ、ここに入力したこの匿名コールバックに渡されます。

したがって、値を出力すると、テキストフィールドのテキストが変更されるたびに、onChangedがトリガーされ、値が出力されることがわかります。それでは、Lから始めましょう。Lが印刷されます。 L O L O、それからL N D ON。すべての変更ごとに、printステートメントがトリガーされ、コンソールに印刷されることがわかります。

今、私たちがしなければならないことは、その特定の都市の天気を取得できるように、その値を保存して天気モデルに渡すことです。

そのため、次のレッスンでお会いしましょう。