前回のレッスンでは、ユーザーがテキストフィールドに入力した値をどのようにキャプチャできるかを見ました。

そして、onChangedプロパティをリッスンすることでそれを行いました。

したがって、テキストフィールドが変更されるたびに、この匿名コールバックがトリガーされ、コンソールに出力できる新しい値が渡されます。

ただし、コンソールに出力する代わりに、プロパティに保存してみましょう。

それでは、cityNameという名前の新しい文字列プロパティを作成して、空から始めましょう。

テキストフィールドが変更されたら、都市名を、ユーザーがテキストフィールドに入力した新しい値と等しくなるように設定します。そして今、私たちがしなければならないことは、ユーザーが天気を取得ボタンをタップしたときに都市名を返してみることです。

そのため、これらの中かっこ内では、何らかの方法でこの都市画面をナビゲーションスタックからポップし、その都市名の値を前の画面に戻す必要があります。

これまでのところ、データを転送して状態オブジェクトに転送するデータの一部を解析する方法についてはすでに見てきました。

ステートフルウィジェットに解析し、状態オブジェクトのウィジェットプロパティを介して取得することで、これを行うことができます。

しかし、ここで、データを逆方向に渡す方法を理解する必要がありますか？

まず、navigator.popメソッドを追加しましょう。このpopメソッドにカーソルを合わせて、ドキュメントを表示します。

2つの値を持っていることがわかります。1つは現在のコンテキストであり、画面をポップオフするために常にそれを渡します。

しかし、結果である2番目のオプション入力もあります。

そして、ここで、戻す値を追加します。

そのため、現在のコンテキストまたはウィジェットツリーのどこでnavigator.pop parseを記述し、その都市名変数を渡します。

これで、さらに多くの解析対象がある場合、たとえばマップやオブジェクトを解析できます。

しかし、今、これを解析し直したので、ロケーション画面でどこにアクセスできますか？プッシュメソッドを見ると、実際にはオプションの出力があることがわかります。

あらゆる種類の動的タイプで未来を出力することができ、その出力はここから来ます。

そのため、ポップしたとき、何かを渡しました。

それがnavigator.pushメソッドの出力になります。

したがって、typedNameと呼ばれる変数を作成することでそれを利用し、それをnavigated.pushの出力と等しくなるように設定できます。

そのため、都市画面がロケーション画面の上にプッシュされると、ユーザーは都市名の値を入力します。画面がポップオフされると、入力した名前の出力として都市名が解析されます。

ただし、ユーザーがいつ都市名を入力するかわからないので、これはもちろん非同期メソッドであることに注意してください。

それが未来を返す理由です。

したがって、この型付けされた名前をどこかで使用する必要がある場合は、少し後で、この結果を待つ必要があります。つまり、onPressedを変更するには非同期を追加する必要があり、このような匿名関数にキーワードasyncを追加できます。または、以前に行ったように、そのような名前の関数とメソッドに。中括弧の前に置く限り、それは実際には問題ではありません。これで、都市画面からのユーザー入力から取得したこの型付きの名前変数にアクセスできるようになったので、使用できるようになりました。

入力した名前をコンソールに出力して、実際に何かが戻ってきたことを確認しましょう。

それでは、これを確認しましょう。

ホットリスタートをクリックして、アプリを最初から起動して位置情報画面をロードします。

そして、これはもちろんnavigator.pushが作成される場所です。

それでは、市のページに移動して、ここで天気を表示したい市の名前を追加し、[天気を取得]をクリックして、もちろん前の画面に戻ります。ここで、入力した名前をコンソールに出力します

入力した名前がここにあるので、実際に現在の天気を取得するためにそれを使用してみましょう。

しかし、最初にそれがnullでないことを確認しましょう。

そして、型名がnullと等しくないかどうかをチェックするIFステートメントを追加することで、それを行うことができます。

その場合、実際に使用できるようになります。

そして、それを使用するために、天気の中でメソッドを呼び出します

.dart。ただし、ユーザーが都市画面に入力した都市の近くにいない可能性がありますが、現在地に基づいて天気を取得することに依存しているため、get location weatherメソッドを使用することはできません。

したがって、ここで新しいメソッドを作成する必要があります。1つ作成し、都市の天気を取得すると呼びますが、これは非常によく似ています。

また、ネットワークヘルパーを使用し、オープンウェザーマップのURLに基​​づくURLを解析します。

それでは、Open Weather Mapのドキュメントを見て、都市名に基づいて天気データを取得する方法を見てみましょう。

最初の方法は、都市名で天気を取得することです。これが、API呼び出しをどのように構成するかがわかります。

したがって、Qというパラメーター名を追加する必要があります。その後、関心のある都市に等しくなるように設定します。そして、既に登録したアプリIDを渡す必要もあります。

ここでURLをフォーマットしましょう。市の天気を取得する中で、urlと呼ばれる変数を作成し、openWeatherMapURLと同じにします。

しかし、その後、他の部分を追加します。

そこで、疑問符を追加してからqを追加し、次に等号を追加します。そしてその後は、私たちの都市名がどこに来るかです。

したがって、都市の天気予報を取得する方法では、都市名を渡す必要があります。

それでは、このメソッドの入力を追加して、都市名と呼びましょう。

そして、それを使用してURLを形成できるようにします。

ドル記号を追加して、この位置に都市名を入力してみましょう。もちろん、アプリIDなどの他のパラメーターにタグがあります。

それでは、アンパサンドまたはandシンボルを追加して、アプリIDパラメーターを追加しましょう。または、単に緑色のこの部分を単にコピーすることもできます。

まったく同じです。

そして、タイプミスがないことを確認してから、APIキーである定数もここに追加します。

そして最後に、温度をケルビンではなく摂氏に戻したいので、最後に単位とメトリックにタグ付けします。これで、セミコロンを追加してURLを完成させることができます。

だから今私がしなければならないことは、場所の天気を取得するためにやったことと非常に似ています。新しいネットワークヘルパーオブジェクトを作成します。これを単にネットワークヘルパーと呼びます。

そして、その場で作成したURLで初期化します。

これで、2行でこのようにするか、ここで行ったことを使用して、このコード行をここに挿入することができます。

両方の方法が機能します。

それはすべて、読みやすく理解しやすい方法に依存します。

データを取得するURLでネットワークヘルパーを初期化したら、networkHelper.getDataメソッドを呼び出して、その結果をweatherDataという新しい変数に保存できます。

したがって、これまでのところ、非常によく似ています。

そして最後に、get city weatherメソッドの出力として気象データを返すことができます。

もちろん、これは、これを動的出力に変更する必要があることを意味しますが、動的出力だけではなく、将来の動的出力になります。再び前とまったく同じです。

そして、getデータは非同期であるため、このメソッドを非同期メソッドとしてマークする必要があります。

それを返すために、これが完了するのを待つ必要があるので、ここにawaitキーワードを追加しましょう。そして、都市名の取得メソッドを設定して、渡した都市名に基づいて天気データを取得できるようにする必要があります。

それでは、ロケーション画面に戻り、タイプ名に値があることを確認するためにすでにチェックしたこの部分の内部に戻りましょう。これで、入力した名前を天気モデルに渡すことができます。天気。

ただし、データ型は、weather.dartから取得した天気モデルです。そして、今作成した都市の天気を取得するメソッドを利用し、テキストフィールドから取得した入力された名前を使用して都市名を渡します。天気データを取得したら、天気データと呼ぶ変数内に保存します。これは非同期メソッドであるため、futureを返すことを忘れないでください。次に、この天気データをupdateUIメソッドに渡してユーザーインターフェイスを更新できるようにするため、awaitキーワードでこれをマークする必要があります。ロケーション画面。

そして、それはここに入り、天気データがnullに等しくない限り、温度、天気アイコン、天気メッセージ、都市名の値を取得しようとします。

それでは、アプリを停止して、寒いところから実行しましょう。それでは、市の画面に移動して、新しい市の名前を入力しましょう。

たぶん北京を試して、天気をクリックしてみましょう。そのため、都市の画面から飛び出してロケーション画面に戻り、現在の北京の天気を伝えます。

もちろん、現在地のアイコンをクリックして現在地の天気を取得することもできますが、都市を考えられる場所に変更することもできます。

現時点でのパリの様子を見てみましょう。涼しい。

したがって、最後に追加する必要があるのは、このボタンが現在何もしていないことです。

そして、それは、キャンセルボタンのようなものです。この画面に戻って、実際にどの都市の天気がいいのか実際にはわからないと判断した場合は、戻ってアクションをキャンセルできます。

そのため、ボタンで必要なことは、何も渡さずに画面をポップオフすることだけです。市の画面上にあるボタンがどこにあるのかを見てみましょう。このボタンはフラットボタンにあります。

そのため、現時点では、onPressedは空ですが、navigator.popに変更し、コンテキストのみを渡します。

この時点で、型名がnullに等しいかどうかを確認しなかった場合、null値で天気を取得しようとすると、潜在的に問題が発生する可能性があることに注意してください。しかし、そのチェックを行っているので、うまくいけば影響を受けないはずです。

もう一度実行して、確認しましょう。それで、戻るボタンをクリックすると、都市を変更せずにここに戻ります。

このボタンをクリックすると、位置データが取得されます。

ここをクリックすると、都市データが取得され、アプリが美しく機能するようになりました。

この天気アプリを私と一緒に楽しんでいただけたことを願っています。

そして、このアプリには多くの改善点があります。

たとえば、現在地の現在の天気を取得するだけでなく、天気予報を取得するのはどうですか？

画面上により多くの値を表示するためにウィジェットを追加するのはどうですか？また、1時間ごとの予報や過去のデータを取得したり、UVインデックスや大気汚染などを取得したりするなど、Open Weather Map APIで実行できるすべてのクールな機能を確認できます。

世界は本当にあなたのカキであり、あなたが望むならあなたの天気アプリをカスタマイズして更新するのはあなた次第です。

しかし、このモジュールについては私からすべてです。次のモジュールでお会いできることを楽しみにしています。