そのため、前回のレッスンでは、Open Weather Mapから取得した実際のデータが表示されるようにロケーション画面を更新しました。

これには、温度、現在の気象条件、およびデータの取得元であるその都市に関するメッセージが含まれます。このレッスンでは、左上隅にこの小さな場所のアイコンがあります。

そして、アイデアは、歩き回ったり他の場所に行ったりして現在の場所の天気を確認したい場合、新しい場所のリクエストをトリガーしてその場所の天気を取得できるようにすることです。

しかし、現時点では、すべての位置情報の作業はローディング画面内で行われます。そして、それはあまり意味がありません。

読み込み画面内だけでなく、アプリ内のどこからでもこれを呼び出すことができます。

それでは、リファクタリングをいくつか行い、これらすべてを独自のファイルに移動しましょう。これらはすべて気象データの取得に関連しているためです。

weather.dartに移動しましょう。

したがって、理論的には、location.dartとnetworking.dartを削除して、代わりにweather.dartファイルに配置できるはずです。

それで、一番上で、ここに行きます。これで、コードをリファクタリングできるようになり、エラーを取り除くことができます。実際の現在位置を取得する位置データ取得の最初の部分を削除し、ネットワークヘルパーを使用して天気データを取得します。次に、この天気データオブジェクトを使用して、必要な画面に送信できるようにします。 。

そのメソッドからそれをカットしましょう。

そして、weather.dart内で、このクラス内に新しいメソッドを作成しましょう。そして、これをgetLocationWeatherと呼びます。そして、コピーしたものをここに貼り付けます。

したがって、最初に気付くのは、APIキーが何であるかがわからなくなったことです。

そして、それはここにあるので、一緒にそれをweather.dartファイルに入れて、天気を取得するためのAPIキーがあることがより理にかなっているからです。今、私が再フォーマットしたいもう一つは、この長いURLです。その最初のパラメータの前のすべて、疑問符の前のすべてを切り取り、定数に追加します。これをopenWeather Map URLと呼び、ここに入力します別の文字列として。

networkHelperで、Open Weather Map URLの値を渡し、最初のパラメーター、緯度、経度、アプリID、最後に選択した単位を追加します。結果はメートル法です。

したがって、私が持っているエラーはawaitに関するものだけです。これは、すべてのメソッドでawaitを使用できるように、このメソッドを非同期としてマークする必要があるためです。

そして最後に、実際に天気データを取得したら、

このメソッドの結果として出力できるようにしたいと思います。

そのため、最後に天気データを返します。天気データは多くの非同期メソッドに依存するため、このメソッドも未来を返すようにマークする必要があり、未来のデータ型は動的な気象データと同じタイプになります。

ですから、将来の動的オブジェクトになるでしょう。また、WeatherModel内に、特定の場所の天気を取得する別のメソッドがあります。

そして、このメソッドを呼び出すと、将来の出力として天気データを取得できます。

それでは、ロード画面に戻り、天気データにアクセスできることを確認してください。

そこで、ここで新しい天気データモデルを作成します。

そのため、そのファイル、services / weather.dartをインポートし、weatherModel equals WeatherModelと呼ばれる新しい天気モデルを作成する必要があります。次に、この天気モデルを使用して場所を取得します天気。そして、このメソッドの出力は変数に保存され、weatherDataと呼ばれます。そして、このコード行を通じて初期化すると、その気象データがロケーション画面に渡されます。

これですべてが長すぎる場合は、単に天気モデルをインラインで初期化することもできます。ここで一度だけ使用するので、ここで切り取って貼り付けることができます。

そして最後に、場所を取得する天気が未来を返すことを知っているので、このメソッドを待機でマークすることを確認する必要がありますか？

したがって、これはいつでも完了できます。

ただし、この内部気象データの結果を使用する必要があります。これは、ロケーション画面に渡すからです。

したがって、これに値がない限り、これは機能しません。

最後に、ロケーション画面に戻って、ロケーションアイコンのあるフラットボタンを見つけると、これがここにあり、押されたときに、現在のロケーションの現在の天気を取得するリクエストをトリガーする必要があります。

そのため、ここでは、weatherDataオブジェクトである新しい変数を作成し、ロケーション画面からロケーション天気を取得するように呼び出します。なぜなら、現在は別のファイルにあり、モジュール化されているからです。

ロード画面で使用したり、ロケーション画面で使用したりできます。

そのため、ここにはすでに気象モデルのインスタンスがあります。

天気と呼びました。したがって、再初期化することなく、ここでそれを利用できます。

つまり、weather.getLocationWeatherと言うと、結果は天気データオブジェクト内に保存されます。そして、再びupdateUIを呼び出し、この呼び出しから天気データを渡します。

位置情報を取得する天気が実際に未来を返すことを覚えておいてください。

したがって、次のコード行でこの気象データを使用する必要がある場合、これがnullでないことを保証する必要があります。

したがって、次の行に進む前に、これが完了していることを確認する必要があります。つまり、再びawaitキーワードが必要になります。

ただし、関数をasyncでマークしない限りawaitキーワードを使用することはできません。他の関数やメソッドなど、他のすべての場所で行ったのと同じ方法でコールバックでそれを行うことができます。それでは、アプリを停止して、ゼロから実行しましょう。そして、iOSシミュレーターで実行することから始めます。そして、場所を変更する方法を紹介します。

現在、私の場所はクパチーノにあるアップル本社に設定されています。そして、それが私がクパチーノの天気を取得している理由です。

ただし、シミュレーターがフォアグラウンドで開かれていて、シミュレーターがここに表示されている間に、デバッグ、ロケーション、およびカスタムロケーションを選択します。

そのため、ここにカスタムの緯度と経度を追加できます。ここでは、ロンドン用に1つを配置しました。

[OK]を選択すると、場所は新しいカスタムの場所に設定されます。

しかし、このボタンをクリックすると、更新された天気データを取得する必要があると予想される奇妙な癖があります。物理的なiPhoneデバイスでこれを実行している場合、それは完全に当​​てはまります。

しかし、iOSシミュレーターはそれほど賢くありません。

ただし、デバイスの場所を強制的に更新する方法があり、iOSでネイティブに開発している場合でもこれを行う必要があります。

トリックは、ホーム画面に移動してマップに移動することです。そしてここ、

自分がいる場所のボタンを選択すると、ここで追加したカスタムの場所に基づいて現在ホランドパークにいます。そしてもし戻ってきたら、マップアプリを終了して天気アプリに戻ってクリックします再びこのアイコンを見ると、そのカスタムの場所から読み込まれていることがわかります。そして、ケンジントンで念のためコートを持って来るように言っており、それは約10度で雨が降っています。これはロンドンのように聞こえます。これをAndroidエミュレーターで実行すると、同様のケースがあります。

そのため、通常はデフォルトでマウンテンビューに設定されますが、これがGoogleのホームです。

また、場所を変更する場合は、これらの3つの点を押して、ここで座標を変更できます。

それでは、51.5に-0.2に変更して、sendを押して、多くのAndroidエミュレーターでも同じことができるようにしましょう。

場所ボタンをクリックしても、現在の場所の天気は実際には更新されません。

ただし、Google Mapsアプリに移動して、これを上にドラッグしてGoogle Mapsを開き、locateをクリックすると、デバイスの場所がトリガーされ、ここでエミュレータに指定した場所に更新されます。

そして、アプリClimaに戻ってこのボタンをもう一度クリックすると、ロンドンにいてこれらの更新された緯度と経度を使用していることが登録されていることがわかります。

さっきも言ったように、これは物理的なデバイスでは起こりません。

AndroidまたはiOSデバイス上にあるコードを使用してアプリを実行し、新しいエリアに登録するのに十分な距離を歩くと、そのボタンを押すと自動的に更新されるはずです。

しかし、エミュレーターには特定の癖があるだけで、エミュレーター内でテストしているときに回避できます。

しかし、実際のデバイスでテストしている場合、これについて心配する必要はありません。

これは、iOSとAndroidの両方、および物理デバイスでもアプリを徹底的にテストする良い機会です。

持っている場合。そして、徹底的にテストすると、この特定のエッジケースがアプリをクラッシュさせることに気付くかもしれません。

たとえば、Androidエミュレーターでアプリの設定に移動し、場所の設定を検索して、先に進み、場所をオフにするだけです。

そして今、場所が無効になっているデバイスでアプリを実行しようとすると、これが起こります。

温度や天気のアイコンなど、表示するように求められているすべての情報はすべて天気データから取得されているため、ウィジェットは何を表示するかわからないため、アプリがクラッシュします。また、現在位置のスイッチがオフになっているデバイスの場所がないため、現在の気象データはヌルに等しくなります。

そのため、存在しない場所の気象データを取得しようとしていますが、気象データのnull値を取得するだけです。

では、どうすればこれをキャッチして、アプリのクラッシュを防ぐことができますか？天気データを使用してウィジェットに表示するものを提供し始めたここでは、天気データがnullに等しいかどうかを確認できます。そして、その場合は、たとえば、サーバーがダウンしていて有効な結果が返されなかった場合、緯度と経度が無効であった場合、または誤って配置した場合など、多くの原因が考えられます間違ったAPIキーを使用した場合、またはインターネットがダウンした場合は、nullの気象データを取得できる理由の多くの理由があります。

そして、これに対処する必要があるポイントは、ウィジェットに値を挿入し始める直前です。これは、画面を混乱させるためです。そのため、失敗する可能性のあるすべての場所で、大量のtry catchを追加することができます。もう1つの方法は、ウィジェット内で使用する気象データを使用する直前に、実際にnullと等しいかどうかを確認することです。

この場合、さまざまな理由で気象データが存在しないため、ユーザーに問題があることを伝えることができます。

そのため、ここで天気データがnullの場合、温度をゼロに設定し、天気アイコンを使用してユーザーにエラーがあることを伝えます。 「天気データを取得できません」と言うように天気メッセージを設定し、最後に都市名について、空の文字列に設定します。

最後に、このifステートメントの最後にreturnステートメントを追加する必要があります。これにより、天気データがnullの場合、これらすべてを実行してからこの関数またはこのメソッドを終了することがアプリに伝えられます。また、このreturnキーワードは、voidを返す場合でも、Dartのすべての関数とメソッドで使用できます。

これにより、メソッドが途中で終了し、次の行に進むことができなくなります。これは、アプリがクラッシュすることがわかっていることです。

ここで、位置情報がまだ無効になっている同じ条件下で、アプリを再度実行しましょう。

このアプリは、エラーが発生し、天気データを取得できないことをユーザーに通知します。

これで、このテキストに由来するキーワード内のキーワードを削除することで、これをもう少し洗練させることができます。そのため、カスタム変数を作成できます。または、ポップアップまたはアラートを追加して、ユーザーにこの情報を提供し、画面を空白のままにすることもできます。

しかし、気象データがヌルになる可能性がある状況を処理する方法はたくさんあります。

したがって、アプリを徹底的にテストし、このバグに対処できるかどうかを確認してください。