そのため、現時点のアプリでは、気象データから温度を取得できます。しかし、残りのすべては、実際にはまだハードコーディングされたテキストです。

weather.dartファイルを見ると、これらの条件チェックをすべて入力してelse ifを追加する必要がないように、ここにコードが含まれていることがわかります。一つずつ。ここにあるのは、WeatherModelと呼ばれるクラスを作成し、現在2つのメソッドがあることです。

条件を入力として取得するgetWeatherIconと呼ばれるものは、整数になります。そして、私たちはその状態を取り、それをあらゆるものに対してチェックします。

したがって、Open Weather MapのAPIドキュメントを見ると、条件コードのリストにセクションがあることがわかります。また、その天気IDから取得した条件コードをテーブルと照合して、天気の状態を判断できることがわかります。

300のコードを取得した場合、それは光強度の霧雨を意味し、その数値が上がると、霧雨はどんどん重くなり、シャワーを浴びてから500の範囲で雨が降ります。そのため、そのテーブルを見て、その条件コードをチェックして対応する絵文字を返すif elseステートメントのセットをコンパイルしました。これは、静的なサンシャイン絵文字ではなく、このテキストウィジェット内に表示するものです。

また、温度を入力として受け取り、その温度と照合してメッセージを返すgetMessageメソッドもあります。

たとえば、気温が摂氏25度を超えている場合、「アイスクリームタイム」を返します。

または、20〜25の場合は、「ショートパンツとTシャツの時間です」と表示されます。必要に応じてこれらのメッセージをカスタマイズしたり、好みに応じて華氏に変更したりできます。

しかし、本質的に、これは天気アイコンを提供し、これはメッセージを提供し、どちらも文字列を返します。

ロケーション画面内で、weather.dartファイルを一番上にインポートして、そのファイルをタップしましょう。

インポートし、servicesフォルダーからパスを開始します。

weather.dartファイルを探します。ここでも、温度条件と都市名の変数を作成した場所のすぐ上で、WeatherModel型の新しい変数を作成し、単に単にweatherという名前を付けます。そして、新しいweatherModelオブジェクトと等しくなります。

updateUI内で使用できるようになり、この条件を渡し、weather.getWeatherIconメソッドを呼び出して、この条件オブジェクトを渡します。

したがって、300の条件を取得した場合、これらのifおよびelseステートメントのすべてに対してチェックし、その番号に応じて正しい絵文字を返します。このメソッドから返されるアイコンは、scaffold内、buildメソッド内で使用するものです。

そのため、条件変数を保存する代わりに、それを文字列型に変更し、weatherIconと呼びます。

したがって、updateUI内にのみ存在する変数に条件を変更し、weatherIconにこのメソッド呼び出しの結果を保存します。

そこで、weatherDataに基づいて、ここにのみ存在するオブジェクトを作成しています。

そのため、その気象条件を引き出し、その結果をgetWeatherに渡します。

条件としてのアイコン。

本当にコードを削減したい場合は、潜在的にそれを置き換えることができ、それでも動作します。

しかし、これはもう少し読みやすく、何が起こっているかを2行に分割するだけで理解しやすくなると思いますが、それはあなた次第です。これで、この変数に保存されている天気アイコンができたので、ここからテキストウィジェット内で使用できます。そして、太陽の絵文字をweatherIconに置き換えることができます。

現在、絵文字は文字列とまったく同じ方法で処理されます。

そのため、文字列があり、そのデータ型を使用して文字列を解析できます。

今、あなたのための挑戦です。 weatherModelオブジェクトを使用して、updateUI内で取得した温度でメッセージ解析を取得します。

つまり、weatherDataから既に温度を取得しており、既に整数に変換していることがわかります。

それでは、このメソッドを使用してメッセージを取得してみてください。そして、そのメッセージをここに記入します。そして、ここの2番目の部分は都市名になります。もちろん都市名は天気データから取得されます。

だから、ここからそれが来るつもりです。

そして最後に、特定の都市名のメッセージをまだ読むべきです。

そのため、ビデオを一時停止して、このチャレンジを完了することができるかどうかを確認してください。

大丈夫。

温度、天気アイコン、都市名のプロパティがあります。

そこで、気象メッセージと呼ばれる別の気象メッセージを追加する必要があります。気象データモデル、つまり気象オブジェクトを使用して気象メッセージを作成し、get messageを呼び出します。そして、整数を期待しているので、この温度になる温度を渡します。そのため、入力として渡します。

これで、このテキストの代わりに、天気メッセージをここで使用できます。

ここで、通常は単一引用符を使用する必要がありますが、このテキストには二重引用符が含まれていることに注意してください。

この理由は、文の途中にアポストロフィがあるためです。

したがって、Dartの規則では通常、すべてのテキストを単一引用符で囲む必要があるとされていますが、内部に別の単一引用符がある場合は、バックスラッシュを使用してエスケープできます。

そして、私たちはこれを何度も見てきました。今では、これを手短なトリックとして使用している人がいるかもしれません。一重引用符を含む文字列を二重引用符で囲むと、実際にはその一重引用符をエスケープする必要はありません。

しかし、変数を使用するようになったので、そこで何を使用するかは問題ではなくなりました。

それをDartのデフォルトの単一引用符に戻して、メッセージのこの部分を天気メッセージプロパティに置き換えて、ドル記号から始めましょう。そして、2番目の部分は市の名前に由来します。そのため、その部分を都市名プロパティに置き換えることができます。

そこで、別のドル記号を追加し、都市名を追加します。感嘆符を保持するか、あなたが感じる熱意に応じてそれを削除することができます。

ここで覚えておく必要があるのは、ウィジェットの状態を更新するためにプロパティを使用している場合、新しい天気データがあるとしましょう。そして、すべてのウィジェットはこれらのプロパティに依存していて、表示する必要があるもの。

そのため、この更新時に、ウィジェットに更新を反映させたい場合は、もちろんセット状態でラップする必要があります。

それでは、これらのプロパティの変更すべてに取り掛かりましょう。これで、保存してアプリを実行できます。

これで、画面上のすべてのデータがライブ気象データに基づいて更新されていることがわかります。

だから今クパチーノでこれを記録している時は、摂氏12度で晴れです。

そしてメッセージは、クパチーノに備えてジャケットを持参するように言っています。