前回のレッスンでは、ドロップダウンボタンと多くのドロップダウンアイテムを組み込む方法を説明しました。

そして、これはもちろんAndroidの外観、つまりマテリアルデザインウィジェットです。最後のレッスンでは、iOSピッカーのiOSの外観であるCupertinoピッカーを使用する方法について見てきました。

次に、Cupertinoピッカーを作成するか、ドロップダウンボタンを作成できる2つのメソッドを作成します。

そのため、ドロップダウンボタン用にこのコードをカットし、別のメソッドに組み込みます。

そのため、このドロップダウンボタンを取得します。そして、このメソッドの内部では、コピーしたものを貼り付けるので、ドロップダウンボタンのコードはすべてです。そして、コメントを外し、これを出力として返します。

そのため、最後にセミコロンを追加し、ここで戻り値の型を文字列項目を含むドロップダウンボタンに変更します。

それでは、Dartを保存して、コードを再フォーマットし、コメントを追加して、その再フォーマットを少し見栄え良くしましょう。

これで、ドロップダウンアイテムの取得を使用してそのアイテムを埋めるすべてのドロップダウンボタンを取得できました。

ドロップダウンボタンを取得するためだけに2つのメソッドを用意するのは少し余分に思えます。

それで、これをリファクタリングして、1つに結合してみましょう。

そのため、getドロップダウンアイテム内のこのコードをすべて削除します。

ドロップダウンボタンの一番上に貼り付けて、ドロップダウンアイテムを返す代わりに、ここに単純に配置します。

そのため、これらすべてを1つのメソッドに結合し、アイテムを取得してドロップダウンボタンに配置し、それを出力として返します。

だから今、私たちのクパチーノピッカーのために同じことをする時です。

したがって、ビルドメソッドのコードからそれを切り取り、Cupertinoピッカーを出力する新しいメソッドを作成し、この単純なiOSピッカーを呼び出します。そして、このメソッド内で、コピーしたものを貼り付けて出力として使用し、Cupertinoピッカーを返します。そして、同じ方法でこれをすべて取得し、iOSピッカーに貼り付けます。ピッカーアイテムを返す代わりに、ピッカーアイテムをCupertinoの子として使用します。ピッカー。

そのため、このget picker itemsメソッドも削除できます。

そのため、iOSピッカーメソッドを使用してiOSピッカーを取得するか、AndroidピッカーまたはAndroidドロップダウンを取得することができます。これをAndroidのドロップダウンに変更する必要があるかもしれません。そのメソッドはドロップダウンアイテムを取得するため削除したため、ここでビルドメソッドからも削除して、右側に表示されるエラーを取り除く必要があります。

コンテナ内でAndroidドロップダウンを表示するように選択できるため、保存をクリックすると、ドロップダウンリストがここでレンダリングされているか、iOSピッカーに変更するかを選択できます。

それでは、保存をクリックして、iOSスタイルピッカーを確認します。

ここでメソッドを作成するだけでいいのではないかと思います。ウィジェットを返すだけのget pickerを呼び出して、get pickerがiOSでアプリを実行しているかどうかを確認するだけです。

それでは、iOSピッカーを返します。しかし、Androidで実行している場合は、Androidドロップダウンを返します。

それはクールではないでしょうか？

Flutterを使えば、まさにそれを行うことができます。

実際にコードをチェックして、アプリが現在実行されているプラ​​ットフォームを確認できます。

したがって、コードをiOSデバイスで実行している場合は、iOSピッカーを返すことができることを事前に確認できます。

ただし、Androidデバイスでコードを実行している場合は、Androidドロップダウンを返すことができます。コードが実行されているプラ​​ットフォームを確認するには、dart：ioというDartライブラリが必要です。これをpubspec.yamlに追加する必要はありません。必要なことは、Dartの数学またはDart変換ライブラリをインポートした方法でインポートするだけです。

これで、実際には、dart：ioパッケージ内にあるすべてのものをprice screen.dart内で使用できるようにしたくありません。 Dart SDKを見て、実際にIOパッケージに移動すると、多くのDartファイルが正しいことがわかります。

ファイルからの読み取りの処理方法、ディレクトリの処理方法、その他のさまざまなio機能の実行方法などがあります。私が欲しいのは、実際にこのplatform.dartファイルです。これは、Android、Windows、またはMac OSのiOSでアプリを実行しているかどうかを教えてくれるこれらのブール型のプロパティを持っているからです。そして、これは私が必要なものです。

したがって、ioライブラリ全体を組み込む代わりに、プラットフォームクラスのみを表示します。もちろん、そのプラットフォームクラスはplatform.dartファイル内にあり、このプラットフォームクラスです。

それが、その資本Pの由来です。

ただし、ProcessInfoクラスなどの別のクラスだけが必要な場合は、プロセス情報クラスのみを表示すると言うこともできます。このようにして、ライブラリを絞り込み、ファイルにプラットフォームクラスである関連部分のみを組み込むことができます。

さて、hideと呼ばれる非常に役立つ別のキーワードがあります。

この行が言っていることは、dart：io内にあるすべてのものをインポートするので、ここにあるすべてのもの、このファイルで使用できるすべてのクラス、ただしplatformと呼ばれるものを非表示にします。このプラットフォームクラスを見たくありません。

そのため、hide以外のshowを使用することになります。たとえば、httpパッケージに対して行った場合、このパッケージ全体の名前を変更して名前を付けるだけです。そのため、そのキーワードの後に​​指定できるhttpパッケージの名前があります。

だから、ショーがあり、非表示があります。

したがって、showを使用して、dart：ioからプラットフォームクラスを公開するだけです。そして、コードのここで、IFおよびELSE IFステートメントを文字通り使用して、platform.isiOSであるかどうかを確認できます。それでは、iOSピッカーを返しますが、プラットフォームがAndroidであれば、Androidドロップダウンを返します。

ここで、iOSピッカーまたはAndroidピッカーを選択する代わりに、ここで両方を選択できます。

ピッカーを取得すると言うことができます。今、保存を押すと、iOSでこれを実行しているので、デバイスをここに表示します。

したがって、AndroidとiOSの両方で同じアプリ、同じコードを実行している場合、これがAndroidエミュレーターとiOSシミュレーターにあることを検出し、物理デバイスでもまったく同じことが起こることがわかります。

オペレーティングシステムがAndroidまたはiOSである限り、iOSピッカーがここに表示されている間にAndroidドロップダウンがここに表示されます。

クールじゃないですか？これで、このget pickerメソッドを必要とせずに、コードをさらに簡単にすることができます。単に子プロパティが値を期待する場所に入り、その三項演算子を使用する場合、platform.isiOSである場合、それが正しい場合、iOSピッカーをレンダリングします。ただし、それが当てはまらない場合は、AndroidまたはWindows Phone、またはアプリが実行される可能性のある場所で、代わりにAndroidドロップダウンをレンダリングします。

そのため、1行のコードでプラットフォームを確認しましたが、実際にiOSスタイルピッカーを必要とするのはiOSプラットフォームだけです。それ以外の場合は、代わりにドロップダウンリストをこのコンテナに入れるだけです。

それで、コードが実行されているプラ​​ットフォームを確認し、そのチェックに応答する方法を確認できます

そのプラットフォームに応じて異なるユーザーインターフェイス要素を提供します。