あなたがデザインとユーザーインターフェースのスタイルに非常に敏感な人なら、ドロップダウンリストのようなものがないことに気づいたでしょう

iOS。設計者が本当に台無しにしない限り、これはほとんど見られません。

したがって、これは非常にAndroidスタイルのユーザーインターフェイス要素です。 iOSでは、さまざまなオプションから選択する方法はドロップダウンリストではなく、ピッカービューです。

したがって、スクロールするものであり、スクロール動作を使用して新しい選択を選択できます。

では、アプリにこれをどのように実装できますかFlutterのウィジェットを使用する場合、これまでスライダーやボタンなどのマテリアルコンポーネントに限定し、これまでに使用したものはiOSとAndroidの両方に存在します。そして、両方のプラットフォームで実行されるアプリで見栄えがよくなります。

しかし、ピッカービューなどの一部のものについては、非常にiOS固有のユーザーインターフェイス要素です。

したがって、このクパチーノまたはiOSスタイルウィジェットのセットを見てみると、Flutterチームはすでにこのことについて考えています。

iOS固有の特定のアイテム、たとえばCupertinoピッカーやCupertinoスイッチ、またはCupertinoスタイルのナビゲーションバーなどをiOSアプリに表示したい場合は、この特定のウィジェットライブラリであるCupertinoウィジェットを使用できます。

そこで、ドロップダウンボタンを使用する代わりにコードを変更して、Cupertinoスタイルのピッカーを使用しましょう。

そこで、ドロップダウンボタンを含むすべてのものを選択します。

これを切り取り、後で使用するために脇に置きます。

したがって、コードからもコメントアウトします。

そして今、ピッカーであるコンテナに子供を提供したいと思います。したがって、Cupertinoカタログのウィジェットを使用できるようにするには、Cupertinoパッケージをインポートする必要があります。

クパチーノを検索すると、パッケージが見つかります：flutter / cupertino.dart

これで、ここで見たクパチーノピッカーであるウィジェットを使用できます。

そして、それを作成するために、使用した他のすべてのウィジェットと同じように、新しいCupertinoピッカーウィジェットを作成するだけです。

それをここに入れましょう。

したがって、クパチーノピッカーには3つのプロパティを入力する必要があります。

ピッカー内の各アイテムの高さであるアイテムの範囲、選択されたアイテムが変更されたため、ユーザーがそのホイールをスクロールして選択を変更し、最後に内部に表示される子ピッカー。

それでは、これらのことを今から記入しましょう。

私の目にはかなり自然に見えると思うので、アイテムの範囲を32ピクセルに設定します。

必要に応じて、それを高くしたり短くしたりできます。変更された選択されたアイテムは、ドロップダウンボタンと同様のコールバックを取得します。そして、ユーザーがピッカーを変更すると、ホイールをスクロールし、それらのアイテムの1つに収まると、このコールバックをトリガーします。

そのため、ピッカーで選択したインデックスが提供されます。

したがって、ピッカーの最初の行に落ちた場合、それはゼロになります。

ピッカーの3行目にある場合、2などになります。このコールバックを使用して、ユーザーが選択したものにアクセスできるようになります。

それでは、選択したインデックスをコンソールに出力してみましょう。

そして最後に、childrenプロパティについて、そのピッカーに表示するものを指定する必要があります。このchildrenプロパティを見ると、実際に非常に単純な型が必要であることがわかります。

これは単なるウィジェットのリストです。

ピッカーウィジェットやドロップダウンメニュー項目ウィジェットとは異なります。

ウィジェットの単純なリストが必要です。したがって、ここにテキストウィジェットのリストを追加するだけです。

テキストウィジェットを使用して、これをUSDと呼びましょう。マテリアルドロップダウンボタンを開始したときと同じように、数回コピーします。

そこで、ユーロとポンドを追加します。そして今、私は3つの必須プロパティすべてを提供しました。

保存してみましょう。アプリ内を見ると、ピッカーが表示されています。スクロールして、必要なアイテムを選択できます。

そして、アイテムに決着すると、表示されているアイテムのインデックスを取得することがわかります。

少し上にスクロールすると、最初のアイテムが表示され、さらにスクロールすると0番目のアイテムが表示されます。これは、選択されたアイテムの変更がトリガーされ、ピッカーがユーザーが選択した値をコールバックに渡してコンソールに出力するためです。

そのため、配色に関しては、明るい青色である青色の背景色と一致すると見栄えが良くなると思います。したがって、Cupertinoピッカーの背景色プロパティを、colors.lightBlueと同じ色に変更できます。保存してみましょう。

そのため、他のすべてと一致し、1つのピースに溶け込みます。

ここで、FORループを使用して、通貨のリストからすべての通貨をピッカーに入力します。

そして、これをチャレンジとして完了してほしい。このリストを手動で作成する代わりに、ドロップダウンアイテムに対して行ったことと同様に、

ここまでで新しいメソッドを作成してほしい。

したがって、getPickerItemsと呼ばれ、ウィジェットのリストを返します。ウィジェットのリストは、ここにある子であるため、ピッカーに入れます。したがって、このハードコードされたリストを置き換えます。もちろん、以前と同じように通貨リストの通貨を使用します。

成功した場合、リスト内のすべての通貨をスクロールできるピッカーが必要です。

そのため、ビデオを一時停止して、このチャレンジを完了してみてください。

大丈夫。

したがって、前と同じように、FORループを使用して、通貨リストを単純化するためにループする必要がありました。

また、for inループを使用し、通貨と呼ばれる文字列を作成し、通貨リストの各通貨をループします。そしてループ内で、テキストウィジェットに通貨を含める新しいテキストウィジェットを作成します。

本当に特定したい場合は、テキストウィジェットの新しいリストを作成できます。このpickerItemsを呼び出して、空のリストに等しくなるように設定できます。

今では、nullとして初期化するだけではないことが本当に重要です。

nullリストがあり、リストに追加しようとした場合、pickerItems.addが機能しないと言うためです。

そして、nullリストに追加できないため、実際にクラッシュします。

ただし、空のリストに追加することはできますが、これは何も見えない状態で作成された新しいリストです。

したがって、これはリストとマップの両方で注意すべき非常に重要なポイントです。

注意を怠ると、いくつかのバグが発生する可能性があります。テキストウィジェットのリストである空のピッカーアイテムができたので、これに追加できます。次に、通貨が内部にあるテキストウィジェットを追加します。ループは、通貨リスト内のすべての通貨を調べ、それをテキストウィジェットに挿入し、そのテキストウィジェットをテキストウィジェットのリストに追加します。そして、そのピッカーアイテムリストをメソッドの出力として返します。

ここで、FORループ内に返さないようにすることが重要です。これらの中かっこ内には、最初のテキストウィジェットを途中で返してしまい、FORループをすべてのループに渡さないようにするためです。通貨リストの通貨。

したがって、これを変更して、出力をテキストウィジェットのリストにするというより具体的なものに変更することもできます。

これができる理由は、テキストウィジェットがウィジェットのサブクラスであるステートレスウィジェットのサブクラスだからです。また、Flutterのドキュメントの継承セクションに概要が示されています。

そして、ここでbuildメソッドでこのメソッドを呼び出す準備ができました。

このハードコードされた子のリストを削除し、代わりにget pickerアイテムを呼び出して、ここに挿入します。

それでは、ピッカーを保存してチェックしてみましょう。オーストラリアドルからカナダドル、リストにあるすべての通貨までスクロールできることがわかります。 ご覧のとおり、クパチーノパッケージのクパチーノウィジェットは、マテリアルウィジェットを使用するのと同じくらい簡単に使用できます。

ビルドメソッドに追加し、設定可能なさまざまなプロパティで初期化するだけです。 次のレッスンでは、両方を共存させる方法について説明しますか？ Cupertinoウィジェットを表示するにはどうすればよいですか

ユーザーがiOSでアプリを実行しているときに起動し、ユーザーがAndroidでアプリを実行しているときにマテリアルウィジェットを表示しますか？

それで、そのすべてについて、次のレッスンでお会いしましょう。