さて、以前のレッスンでは、状態を解除することと、その場合に状態を解除する必要がある理由について説明しました。これは、子ウィジェット、タスクタイルよりもツリーの下にあるタスクチェックボックスがあるためです。

そして、taskCheckboxで発生したユーザーインタラクションを使用して、親ウィジェットの状態を決定および設定できるようにしたいので、その状態を使用して、チェックボックスを更新するとともに、木。ウィジェットツリーの状態を上げることにより、より多くの子ウィジェットにダウンストリームを渡すことができます。

そのため、ツリーの上位に状態を実際に必要とする別の状況になりました。

問題を説明させてください。

そのため、「追加」ボタンを押して「タスクの追加」画面がポップアップ表示されたときに、ウィジェットツリーの外観を考えると、これは一種の簡易バージョンです。

TasksScreenが上部にあり、AddTaskScreen画面とTaskListが含まれ、AddTaskScreenにテキストフィールドがあります。

TaskListには、多数のタスクタイルといくつかのテキストウィジェットがあります。

タスクリストは、タスクと呼ばれる状態を使用して作成されます。

これがタスクオブジェクトのリストです。

また、リストビュービルダーを使用して、タスクタイトルの動的リストをレンダリングできます。

しかし、ユーザーがAddTaskScreenに新しいタスクを入力して[追加]ボタンを押すと、テキストフィールド内のデータをテキストウィジェットまでどのように取得するのでしょうか。

タスクリストのスコープを増やす必要があります。再び状態を持ち上げる必要があります。リストビュー内に追加された最新のアイテムをレンダリングできるように、タスクリストを追加して子ウィジェット全体で使用できるようにするために、タスクリストを上に移動できるようにする必要があります。

さあ、挑戦の時です。これまでに学んだすべてのことと同様に、状態を上げることとコールバックについて学んだことを使用して、テキストフィールドに入力されたデータをユーザーが押したときにTo Doリストアプリを組み込むことができますか？ [追加]ボタン-[最新のTo Do]項目がリストに追加され、[タスクタイトルを単純なステートレスウィジェットとして保持しながら、チェックオフできる機能を保持しますか？]

そのため、これを達成するには、状態の解除、コールバックについて学んだことを考え、この問題に取り組むためにあなたが知っているすべてを使用する必要があります。

そこで、このチャレンジを試して完了するために、ビデオを一時停止してください。ぜひ試してみてください。

難しい課題の1つであることはわかっていますが、それはできると思います。まだ試していない場合は、ビデオを一時停止し、続行する前に必ず確認してください。これは、解決策または少なくとも1つの解決策を公開するためです。同じこと。

したがって、最初にやりたいことは、現在、タスクの名前を含むタスクがあり、タスクがタスクリストステートフルウィジェット内で実行されている場合、ツリーのこのレベルにあることです。ただし、AddTaskScreen内とタスク画面内の他の場所の両方で使用できるように、このタスクリストの状態を解除する必要があります。たとえば、残っているタスクの数や、タスクの名前。

そこで、タスクリストに移動し、タスクをカットして、ツリーを上に移動してタスク画面に移動します。

そして、これは変更可能なプロパティであるため、すぐに警告を受け取ります。つまり、ステートフルウィジェットに切り替える必要があります。

また、ここでのタスクについても何も知らないので、モデルフォルダーとtasks.dartをインポートする必要があります。これで、タスク画面ウィジェット内にタスクのリストができました。

[タスクの追加]画面やタスクリストなどのダウンストリームにアクセスできるようになりました。

それでは、まずタスクリストに渡します。

そのため、ここでコンテナ内に新しいタスクリストを作成するとき、そのようなタスクを渡すことができるようにしたいのです。

したがって、これを行うには、タスクリストにtaskというプロパティが必要になります。これは、タスクオブジェクトのリストになります。

それをタスクとして作成しましょう。最後に「s」があることを確認してください。

そして、タスクリストを作成するときに、タスクが確実に渡され、このプロパティとして初期化されるようにします。

これで、タスクリストの状態内で、ウィジェットオブジェクトを使用してこのアイテムを利用できます。したがって、widget.tasksを使用して、現在赤い波線があるすべての場所に追加できます。

そのため、オブジェクトを渡して状態に移行します。

そのため、ウィジェットは、新しいタスクリストを作成するときにそのプロパティを取得するステートフルウィジェットを指します。このタスクリストは、このプロパティを使用して自分自身を構築する方法を知っているため、ここでエラーがなくなります。

これで、次の段階に進むと、ウィジェットツリー（タスク画面）のほとんど何も変わっておらず、タスクがより高いポイントから来ていることがわかります。

これで、AddTasksScreenを機能させる準備ができました。

したがって、ここに入ると、まずテキストフィールドがonPressedにデータを渡すことができることを確認できます。したがって、データが別の場所にあることを覚えておいてください。テキストフィールドはここにありますが、ユーザーが[追加]ボタンを押したときにのみそのデータを渡します。もちろん、テキストフィールドにはonChangedというプロパティがあり、変更されると、コールバックを取得して新しいテキストをコールバックに渡します。

そして、それが起こったら、その文字列を新しいタスクタイトル内に保存し、このコールバック内で設定します。

したがって、新しいタスクのタイトルは、ここのテキストボックスに追加される新しいテキストと同じになります。

そして、onPressedが押されたら、最初に新しいタスクタイトルの値を印刷してみましょう。

ここで、テキストフィールドが変更されるたびにonChangedがトリガーされることを思い出してください。

だから今、私が先に進んで新しいタスクを追加し、キーボードをiOSでポップアップさせたい場合は、それを行うのはCOMMAND + Kだけであることを覚えておいてください。

それでは、新しいタスクを追加しましょう。わかりませんが、

「ジムに行きます」を押して「追加」を押すと、この文字列がこのテキストに渡されるため、ここに印刷されている文字列が表示されます。新しいテキストは、ここにある新しいタスクのタイトルに割り当てられ、フラットボタンを押すと印刷されます。

もちろん、この新しいテキストは、テキストフィールドが変更されるたびに実際に更新されます。したがって、入力が完了した時点がわからないため、書き込みが完了した時点で更新するだけではありません。

たとえば、ここに新しいテキストを印刷すると、新しい文字を追加するたびに、その印刷が再度呼び出されることがわかります。

再びこの考えに戻って、ユーザーがいつ完了したかはわかりません。

「ジムに行って何かを食べる」と考えるために一時停止するだけの時間があるのか​​どうかはわかりません。

これが私がすることです。

うーん、冗談です。

そのため、いつ終了するのかわかりません。

そのため、Addボタンを押して実際に通知するまで待つ必要があります。

したがって、入力したすべてを追跡し、追加ボタンを押した瞬間にそれを最終的なタスクタイトルとして取得し、それを使って何かを行います。 「タスクの追加」画面に入力したこの新しいタスクを取得する方法がわかったので、実際にどのようにタスク画面に戻ってタスクのリストに追加するのでしょうか？

まあ前と同じですか？

コールバックが必要です。

それでは、ここでコールバックを作成しましょう。

タスクを追加する画面が実際にコールバックを取得するように作成して、コールバックが

括弧のセットと中括弧。

そして、ここで、ユーザーが入力したものを何でも印刷してみます。

それをnewTaskTitleとして、新しいタスクタイトルを試して印刷します。ただし、このトップレベルウィジェットで実際に管理していることを示すために、タスク画面を追加する代わりにタスク画面内に入れます。 。

それでは、タスクの追加画面を作成するとき、機能を使用できる必要がありますか？

ステートレスウィジェットはaddTaskCallbackと呼ばれる最終関数を取り、タスク追加画面を作成するときに追加されます。これでthis.addTaskCallbackになり、保存をクリックしてタスク追加画面に戻ります。エラーが消えるのがわかります。したがって、このコールバックがタスク追加画面に渡され、タスクコールバックと呼ばれるこのプロパティに入れられます。これをonPressed内で使用する必要があります。ここでは、新しいタスクタイトルの値を印刷する代わりに、タスクコールバックを追加するコールバックを呼び出して、新しいタスクタイトルをここに渡します。

これはタスク画面に戻され、ここで印刷できるかどうかを確認します。

アプリを再度実行して、[追加]ボタンをクリックし、[Go to the gym]を追加して[追加]を押すと、印刷が表示されます。

そのため、このコールバックを使用して、ユーザーがタスク画面内で入力したものにアクセスできるようになりました。

したがって、ここで印刷する代わりに、実際にタスクのリストに追加することができます。

tasks.addをタップすると、新しいタスクタイトルに基づいて名前を持つ新しいタスクアイテムを追加します。

そしてもちろん、isDoneプロパティはデフォルトでfalseに設定されます。

そのため、タスクリストに新しいアイテムがあり、それを設定状態でラップすると、[追加]ボタンを押すとすぐにタスクリストの更新が表示されます。

それで、先に進んで[保存]をクリックし、[追加]ボタンを押して、この下のシートを閉じると、最新のタスクがタスクリストに追加されたことがわかります。

しかし、毎回手動でボトムシートを閉じる必要はありません。

当然のこととして、[追加]ボタンを押すと、一番下のシートが消え、追加した最新のアイテムが表示されます。そのためには、ナビゲータに呼び出しを追加して、ボトムシートになる現在のコンテキストをポップする必要があります。

それでは、最初からアプリを最初から実行して、新しいアイテムを追加してみましょう（「ジムに行く」ので、追加ボタンを押すと、下のシートが消えます）、最新のタスクが追加されます。リスト。

だから私のソリューションはあなたのものとは異なるかもしれませんが、あなたが私たちのtasksオブジェクトの状態をタスク画面状態まで持ち上げて、それを取得するためにコールバック内でタスクタスク追加の両方の両方で使用できる限りテキストフィールド内で作成され、タスクリスト内でも使用され、更新時に最新のタスクを入力するタスクは、この課題でほぼ成功を収めました。

ここで追加できる最後の1つは、タスクリストを使用することです。これは、ここでハードコードされたもので、タスクリストにいくつのタスクがあるかを示します。

したがって、毎回1​​2を書き込む代わりに、文字列補間を追加してタスクリストにタップし、lengthプロパティをタップして、実際に表示されるタスクの数をここに表示できます。それでどうですか？状態を解除し、コールバックを使用して、チャレンジを完了できましたか？

理にかなっていますか？その後、コールバックのレッスンを確認し、それについて学んだことをこのアプリ内で実践してみてください。

次のレッスンでは、状態管理についてもう少し詳しく説明し、プロバイダーパッケージと呼ばれるものについて説明します。

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。