さて、前回のレッスンでは、ストリームビルダーウィジェットを使用して、送信されるメッセージと送信される新しいメッセージでFlutterアプリを更新できるようにしました。このレッスンでは、チャットアプリのスタイルとユーザーエクスペリエンスの向上に焦点を当てます。

これには何が関係しますか？

まず、私たちが何を目指しているかをお見せしましょう。

このレッスンの終わりまでに、アプリの外観は次のとおりです。

ご覧のとおり、個々のチャットメッセージはこれらの美しい青い泡でき​​れいに分離されています。

しかし、それだけではありません。

実際にチャットメッセージをスクロールすることができます。また、新しいメッセージを作成して送信ボタンを押すと、テキストフィールドがクリアされます。

今、私たちが今どこにいるか見てみましょう。

たとえば、画面上のスペースに収まらないほどメッセージが多すぎる場合、どうなると思いますか？

メッセージが多すぎる場合に何が起こるかを最初に示したいのですが、さらに20個のメッセージを作成する代わりに、テキストウィジェットのスタイルを変更するだけです。

そこで、スタイルプロパティを追加し、このテキストスタイル内で、フォントサイズを現在のサイズ（おそらく約14ピクセル）から50に変更します。

それをもっと大きくして、何が起こるか見てみましょう。

ここでアプリを見ると、各テキストウィジェットが巨大になることがわかります。

そして、私たちにはたくさんのメッセージがあるので、それらは今や画面から流れ出ています。

キーボードをドロップしても、画面からはみ出しており、実際には非常に大きいため、送信ボタンとテキストフィールドも完全に画面から押し出されています。

5つのメッセージを受信することしかできないチャットアプリを作成できないため、これは私たちが望むものではありません。

これを変更しましょう。

代わりにスクロールして、チャット画面内に無限の、ほぼ無限の量のデータを保存し、一度に1つのパーツのみを表示する必要があります。

そのため、スクロールして古いメッセージを確認します。

Flutterでこれを行うために、ListViewと呼ばれるものがあります。また、Androidでリストビューを設定するのにどれだけの労力を費やすか、iOSでテーブルビューを設定する場合の代替案を検討する場合は特に、非常に簡単に使用できます。

必要なことをすべてお見せしましょう。現在、Stream Builderは、テキストウィジェットのリストである子を含む列を返しています。

テキストウィジェットはそのままにしておきますが、列ウィジェットを使用する代わりに、ListViewウィジェットに変更します。

また、送信ボタンとテキストフィールドを含む同じ列内にコンテナがあるため、リストビューを展開されたウィジェット内にラップします。この方法では、画面全体を占有しようとするのではなく、使用可能なスペースだけを占有します。

そのため、保存をクリックしてアプリを見ると、スクロール可能なリストビューにメッセージがあり、上下にスクロールできることがわかります。

また、ここにはテキストフィールドと送信ボタンもあります。

仮想キーボードを上げると、リストビューがバックグラウンドにまだあることがわかります。実際には変更されず、スクロールすることはできますが、テキストビューはサイズが上下します。キーボードの。

スクロール動作を実装したので、今度はチャット画面をスタイリングして、本物のチャットアプリのように見えるようにします。

これは、Flutterのホットリロードの動作と、ワークフローで実際にどのように役立つかを確認するもう1つの良い機会です。

ここで最初にしたいことは、リストビューにパディングを追加することです。

パディングを少し追加すると、edgeInsets.symmetricになります。

そして、水平に約10、垂直に20を追加します。

セーブを押してみましょう。上から少し押し下げられ、左右から少し押し込まれます。そして次に行うことは、メッセージがひどく見えるため、メッセージの表示方法を改善することです。

そのため、沼地の標準テキストウィジェットを使用する代わりに、カスタムを作成しましょう。

それで、一番下まで行き、新しいステートレスウィジェットを作成します。そして、私はそれをMessageBubbleと呼ぶつもりです。

そして、このMessageBubbleはコンテナを返す代わりに、テキストウィジェットや他の多くのものも含むようになります。

それをここからカットして、ここに貼り付けます。

ここにテキストウィジェットを移植したので、「メッセージテキストとメッセージ送信者」というこれらのものを提供できる方法が必要です。

そこで、送信者と呼ぶ最終的な文字列変数と、テキストと呼ぶ最終的な文字列変数を作成します。

そして、メッセージバブルが構築されると、これらの値、つまり送信者とテキストを初期化できるようになります。

したがって、メッセージテキストを使用する代わりに、単にテキストをコード化し、メッセージ送信者を使用する代わりに、送信者と呼ばれるようになりました。

したがって、ビルドメソッド内では、沼地の標準テキストウィジェットだけを返すのではなく、マテリアルウィジェット内にラップします。マテリアルウィジェットがある場合、背景色を指定できるからです。

それを多分水色のアクセントに変えましょう。それで、私がデザインしている間、それがどのように見えるかを見てみたいです。

それでは、実際に使用できるように、コードをここに修正してみましょう。

そのため、以前はテキストウィジェットと同等の最終的なメッセージウィジェットがありましたが、今度はメッセージバブルから作成されます。そして、送信者をmessageSenderとして渡し、テキストをmessageTextとして渡します。

そして、私たちがここにいて、物事の名前をメッセージバブルに変更している間に、先に進んでこれらの変数名も変更しましょう。

したがって、messageWidgetと呼ばれるすべての場所を変更します。

リファクタリングと名前変更を行い、単なる沼地の標準ウィジェットと呼ぶ代わりに、messageBubbleにします。それをリファクタリングしてから、メッセージウィジェットと呼ばれるのではなく、メッセージウィジェットがある場所で、リファクタリングしてメッセージバブルのリストを呼び出します。以前にそのメッセージを指定したため、ここでエラーが発生します。バブルはテキストウィジェットのリストのデータ型であり、テキストウィジェットではなくなりました。

実際、いくつかのmessageBubbleウィジェットです。

ここですべてのコードを修正したので、保存をクリックして、メッセージのバブルがどのように見えるかを確認する準備ができました。そして、現状では、まだ見栄えがよくありません。

背景は青色ですが、各メッセージは他のメッセージにぼやけています。

それでは、少しパディングを追加してそれらを分離しましょう。マテリアルウィジェットの周りに、パディングウィジェットを追加する予定です。初期化されたコンテンツをDart 2.0以降のオブジェクトとして削除できます。

すべてのedgeInsetsがあるので、各メッセージバブルの周りに10ピクセルのパディングを追加してみましょう。

また、少なくともそれらが互いに分離しているのを見ることができます。これは素晴らしいですが、それでも見た目は良くありません。

それで、最大の目障りは何ですか？

まあそれは私たちのテキストが50ポイント大きいという事実です。それを修正して、これをより通常の15ピクセルフォントに変更しましょう。

そして、私はここにいる間に色を変更しますcolors.white。

それでは、saveを押してみましょう。それでもかなり奇妙に見えます。今のところ、私たちの問題は、テキストの周りにある資料がテキストに近すぎるということです。

それでは、テキストを素材から少しパディングしてみましょう。

これもいくつかのパディングでラップしましょう。

そして、おそらく対称パディングを少し選択するように設定し、垂直を10ピクセルに、水平を20に設定します。

それで、垂直よりも水平の周りにもう少し多くのパディングがあり、メッセージのように見え始めていますか？以前より少し良くなり始めています。

隆起ボタンまたはフローティングアクションボタンで見た素材に追加できるものの1つは、標高です。

5ピクセルの高さを追加して、各メッセージに少し影を付けましょう。

そして、今では真のバブルのように見え始めますが、円形でない限り、それは円形ではありません。

それでは、この正方形を丸いオブジェクトにするにはどうすればよいでしょうか？

さて、境界線の半径プロパティを利用できます。

境界線の半径をすべて丸い円形の半径に設定できるため、境界線の半径を利用して、30ピクセル程度の円形の半径を指定できます。

そして今、それは実際の泡のように見えます。

これで、見栄えの良いバブルができましたが、メッセージのバブル内にこれらの送信者詳細を入れるのは意味がありませんか？

それで、それを別々にしてみましょう。

泡の上部にテキストウィジェットを追加できるように、マテリアルを列内にラップします。

そして、ここに送信者が表示される場所があります。その後、「メッセージバブル」という素材があります。

その部分を削除して、テキストウィジェット内にのみテキストを入れましょう。

保存ボタンをクリックすると、送信者の詳細が上部に表示され、そのすぐ下にテキストが表示されます。

しかし、現時点では、送信者の詳細が少しばかり注目を集めています。黒すぎて大きすぎます。

それでは、そのテキストウィジェットのスタイルを変更しましょう。

フォントサイズを12に下げ、色を黒54に変更できるように、少し軽くします。

そして今、あなたは主なポイントがこれらのメッセージであり、送信者名がその上にあるものであることがわかります。

この時点で、スタイリングの面で多くの作業を行いました。しかし、外観をさらに改善するにはどうすればよいでしょうか？

現時点では、バブルはすべて画面の中央にあります。それでは、先に進んで、一方に移動してみませんか。

そのため、この列内で、交差軸の配置をcrossAxisAlignment.endに変更してみましょう。これにより、一方の側に移動します。そして、実際のメッセージングアプリに非常によく似たポイントに到達し始めていますか？

私はピノキオのようなもののような本物のメッセージングアプリについて話し続けています。

まるで本物のメッセージングアプリのようです。

まだ実際にはそうではありませんが、まだいくつかのことが必要です。かなり多くのコードをアプリに追加してきたので、今度はコードのリファクタリングを検討します。

おそらく、チャットアプリケーションをもう少しシンプルにするために、コードを再編成することができます。

たとえば、チャット画面の状態が少し長くなり、扱いにくくなっていると思います。

では、なぜストリームビルダーを別のウィジェットに抽出しないのでしょうか。

メッセージバブルのすぐ上に、新しいステートレスウィジェットを作成し、MessagesStreamと呼びます。

そして、ここで、ストリームビルダー内にあるこのすべてのコードを1つの場所、つまりこの場所に移植できるようにしたいと考えています。

そのため、コンテナを返す代わりに、ストリームビルダーを返します。

そして、代わりにここでMessageStreamを作成します。それで今、私たちが持っている領域は、Firestoreがチャット画面の状態で現在ここでしか利用できないものであるという事実に関連しています。

では、なぜインポートステートメントの一番下に移動して、このファイル全体にアクセスできるようにしないのでしょうか。この問題を修正したので、メッセージストリームにエラーがなくなり、アプリ内で作成されたすべてのメッセージからメッセージバブルを作成できるようになりました。

したがって、「Hey there」という新しいメッセージを入力して送信をクリックすると、ここに表示されます。

今は注文についてあまり心配しないでください。

作成された順番にならない理由については、後で少し説明します。

しかし、スタイリングに関して修正したい最後のことは、送信ボタンを押した瞬間、すでに送信されたものからのテキストがあるのは意味がないので、テキストフィールドを本当にクリアする必要があるということです？

これは空で、次のメッセージの準備ができているはずです。

では、どうすればこれを行うことができますか？

テキスト編集コントローラーと呼ばれるものを作成できます。メッセージテキストフィールド用に作成して、messageTextControllerと呼びます。そして、新しいテキスト編集コントローラーと同等に設定します。

このコントローラーを作成し、タイプミスがないことを確認したら、テキストフィールド内で使用できるようになります。

したがって、テキストフィールドがある部分はここにあり、テキスト編集コントローラーを期待するコントローラーと呼ばれるプロパティがあります。

そして、何を推測しますか？

すでに作成しています。

メッセージテキストコントローラーです。

これで、このテキストフィールドを制御できるコントローラーができたので、ユーザーが「フラット」ボタンをクリックして送信ボタンを押すと、このテキストフィールドを制御していることがわかっている同じmessageTextControllerをタップして、 clearメソッドを使用して、現在内部にあるテキストをクリアします。

保存ボタンを押して、別の新しいメッセージを呼び出して送信ボタンを押すと、送信ボタンを押すとすぐにメッセージが消えます。

パーフェクト。

それがまさに私たちが望むものです。次のレッスンでは、別のことを取り上げます。メッセージが異なる送信者から届く瞬間にangela@email.comまたはangela@gmail.comまたは他のユーザーを言うと、すべて同じ色で表示されるという事実に対処します。同じフォーマット。

私が送信したメッセージが他の人からのメッセージであるかどうかを判断するのは本当に難しいです。

そのため、次のレッスンで修正します。

そのすべてについて、また会いましょう。