All right.

So now that our app is able to show the questions in our list of questions and also match up the questions with the answers to check to see if the user got the right answer.

But at the moment it's a bit clunky.

We've got two lists that are tracking things that should be associated with each other because this is the answer to this question and ideally it would be great if they were stored together somehow.

So how can we do this?

Well we can create a new class. Just as we have our widgets which have different properties such as a child property or a padding property or a flex property,we can also create a class that have a question property and a answer property, so that we can associate the questions and answers together in a single object. In order to do this,we're going to go into our lib folder where all of the action happens, and we're going to right click and go to new.

I'm going to select to create a new Dart file, and you're going to call this Dart file question, because that's exactly what it's going to represent.

And then it says, 'do you want to add this to git?' You can either click yes or no.

It doesn't really matter in this case because we're not really using git for this project.

So in order to create a new class, we have to use the keyword class.

And afterwards we give our class a name.

So in this case, we're going to call our class question. And it's by convention that when you create a

new class, it starts with a capital letter just as all our classes down here, say our Quiz page or our widgets all start with a capital letter, because they each belong to a different class.

So over here, we've created our question class and we're going to open a set of parentheses to define what is so special about this question class. The question class is only going to have two properties,and they're going to be properties that every question should have.

So one is going to be a string and it's going to be called the questionText, and the other is going to be a boolean and it's going to be questionAnswer.

Now inside this question, we also have to give these variables a value when we create a new question.

In order to do that,we have to create what's called a constructor. So that when we construct this question object, we give it some values for the question text and question answer. And we do this by writing, the same as our class name, question. And then we open a set of parentheses and then I'm going to add some curly braces and this is beginning to look a bit like our functions.

This is where our inputs would normally go. Well in this case these are the values that we're going to provide when we create a new question.

So we're going to have a string that's going to be called q, and we're going to have a bool that's going to be called a. And you can see where I'm going here.

Inside these parentheses that I've just opened up, we're going to set this question's questionText to equal the value of q, and then we're going to set the question answer to equal the value of a.

And this part of the code has a special name.

It's called a constructor.

So now that we have this structure for a question object, we can create a new question object in our main.dart

So inside here, let's create a new question object and we do that by specifying its type.

So this is a new question type. And at the moment, it doesn't know this class question. This is undefined class question, but it's right here right?

So how do we tell this file about this file?

Well we have to import it.

So let's write our inputs.

And here we write our question.dart file.

So now that we've imported that file, then the error goes away and it now knows about the question in class.

So we're creating a new variable which has a data type of question.

And this is gonna be called, let's call it q1. And we're going to set it to equal a new question and we're going to use that constructor that we built over here.

We're going to provide a q and we're going to provide an a.

So when I hit enter, you can see I can tap into the q property where I'm going to put in the first question right here. And this is expecting a string so that satisfies the criteria, and then I'm going to put in a a which is going to be the answer for the first question, which is just a false which is a boolean.

So now I've created a new question object from my question constructor which I built over here. And my new question q1, now has some question text which is set to whatever I put in here and it has a question answer which is set to equal what I put in here for the a.

So now what I can do is, I can say print q1. question Text or print q1. questionAnswer. And notice how these two pieces of data are all associated with that object that I created up here, and then no longer separate like how we had it up here. And we don't have to make sure that they each match in that order in the list. The data for the text and the answer are now grouped together in this object called q1. Now instead of having our list of questions and answers, here's a challenge for you.

Can you figure out how to create a list of questions instead, using the questions and answers that we have down here?

So pause the video and see if you can complete this challenge.

All right.

So let's go ahead and comment out all these lines of code. And right below,we're going to create a new list and the data type of the list are question objects.

So it's gonna be a list of questions. And you can call it anything you want but I'm gonna call it a question bank, and I'm gonna set it to equal a new list using that square bracket notation. Now inside the list, I have to create some question objects. And I'm going to create it exactly the same way as I did up here for q1.

I'm going to use the question constructor. I'm going to provide a value for q, which is the first question, and then I'm going to provide a value for a, which in this case is false.

So now let's go and create our next question and this one has a cue of this string.

And the a should be set to true. So let's put that here.

Now finally, we're on to our last question and we just have to provide our q and or a just as we did before. And this one is also going to be true.

All right.

So now we have all 3 question objects inside a list that contains question objects and it's called our questionBank.

So we have no need for any of this anymore.

And it's now very succinctly linked up our questions with our answers for every one of our questions in our question bank. So you can either delete all of this code or you can keep it around as a comment if you want to refer to it.

But I'm going to get rid of it.

So now we have some errors and we can tell that we have errors because Dart analysis tells us that.

But also you can see at a glance on the right hand side, when you have a little red button there it tells you, you've got errors. In order to find our errors, we can see there's one here there's one here and there's one here.

And if we scroll down, our first error is here. The text inside our text widget. Because previously, we had a list called questions and we pulled out the item at whichever one questionNumber is at.

So at the moment, it's the first item in that list.

But now questionNumber is gone,so how do we get it out of the question bank?

Well it's pretty much the same way. We tap into our list called questionBank.

We pull out the question at this index.

So it'll be a question bank at index 0 when we first start up our app.

So we now have access to this question object. But that doesn't work because our text widget needs a string.

So we need to display this part which in our question class, is known as the questionText. So we can simply write question bank at this index, .questionText. And now we'll be able to do exactly the same thing but using our new question bank instead.

So let's scroll down and fix the next one.

The next issue is here, and we used to have our answers list but now that's gone and is replaced by our question bank.

So we're going to get the correct answer by tapping into the question bank at a particular question number, and then we're going to write .questionAnswer.

And that of course comes from this property right here which is set to the value of a, when we create this question object. Now that we've addressed that one, we've got one more error to go and it's right here.

And it's exactly the same as before, question bank at the index of question number .questionAnswer to get the correct answer for the current question. So we fixed all our errors and if we save and go ahead and hot restart our app, then it should reset our question number back down to zero, and we should start on the first question.

If we click true for this question, you can see that down here, we get user got it wrong because the first question's answer is false. And you can progress through this of course until you run out of questions at which point it'll crash out again.

But now, we've organized our questions and answers together into what's called an object. And we created our objects by creating a class. So if the concept of classes and objects is unfamiliar to you and you're not quite sure what's actually going on here, then be sure to head over to the next lesson where we do a deep dive on classes and objects and how they work in Dart.

Now if you're very familiar with this and you can understand how the constructor works and how our classes are created and how our objects are constructed, then feel free to skip the next lesson and continue building out our app.