So now that we've learned all about Dart lists, you'll start seeing them all over the place. And to start with,we already said that the columns, when we're trying to set its children property,it always expects a list of widgets. And that's why we create them inside these square brackets,and also so do these rows.

They also want to have children that contains a number of widgets.

So now that we've seen our list in action and we've set up our scorekeeper, we want to actually go ahead and remove all of these icons and start off with a empty list.

And as the user scores points, we'll add checkmarks or cross marks to our scorekeeper so that we actually keep track of their score.

Now we can either go through our code and find all the places where we need to remove some of these things such as for example when we added it to here on the onPressed of our true button, or when we added our icons into our scorekeeper list.

But there's another way that you can do this with Android Studio.

So very helpfully, Andrid Studio is actually tracking everything that you're doing behind the scenes.

Now it sounds a bit creepy, but it's actually super helpful.

So first they go into the VCS menu up here, and this stands for version control system.

So a part of that has on local history.

If we go in there and we click on show history, then you'll see all of the code changes that we made.

For example 30 minutes ago, I added my check icon, that's green, to my scorekeeper. And even further along,maybe 34 minutes ago, I added this scorekeeper list to my Quiz page state. So you can see that this is current and this is 42 minutes ago, what it used to look like.

So what you can do is you can simply, if you wanted to rewind the clock and go back in time, then you can see that these are the additions that you made and these are the things that you deleted.

So if you wanted to revert back to this date 35 minutes ago, you can simply select it and click on this button which will revert to the previous changes.

So now if we go back, you can see it's like we time traveled back to 45 minutes ago. And now our icons are inside our row and we don't actually have our list.

Now what if you want to go forwards in time and you want to go to the future?

Well you can also do that.

You can go to the 32 minutes ago version for example and let's click on this, and now we have our scorekeeper list again.

Now it gets a little bit confusing if you time travel a lot.

And as with all good time traveling movies they always tell you that you can't do it too much. This is quite helpful when you've made some grave error and just using COMMAND + Z to go back wasn't enough.

Then using that history browser to go back and forwards will be really really helpful. Now just check to make sure that in your onPressed for your true button, there's nothing in there other than the comment that says well this is what should get triggered when they use it picked true.

And we're also going to delete all of our icons from our list so that we end up with a empty list to start with for our scorekeeper.

Now if you scroll all the way to the top, you might see you have this green squiggly line underneath the word Quizzler.

And this is because Android Studio thinks that Quizzler is a typo.

Now we know better. We know that Quizzler is actually a super awesome name for a app,so this is definitely not a typo.

And if you want to get rid of it, simply right click and go to spelling and click save this word Quizzler to the project level dictionary.

So now inside this project, it won't tell you that this is a typo anymore which is perfect.

So now let's focus on displaying some questions to the user because at the moment, our app just says this is where the question text will go and it doesn't actually show anything.

So we have to update this part. Now we want to create a list of questions and every time we press the true or the false button for it to check our answers and then to go to the next question in the list.

So in order to do this, we've included three questions to start you off.

And they each have their correct answers. So you can lead a cow downstairs but not upstairs, is in fact false.

You can apparently lead a cow up and downstairs because they're super awesome animals.

So we're going to start off with the first question. And then when you click the true button or the false button, it should go to the next question until the end of course. If you want to follow along with me,then we're going to get started tackling this bit by bit.

All right.

So first things first, let's review these three questions.

Now one thing that might be a little bit unusual here is this little backslash here.

Why is it saying a slug backslash's blood is green?

Well I actually want to say a slug's blood is green. But because in Dart, we use these single quotes as a way of telling the computer that this is a String,this is a piece of text.

Then if we have any extra single quotes in there, it gets confused whether if the string should end here or whether if the string should end there. And that's why we have this backslash. It helps to show the computer that the thing that comes after it is actually something that is a part of the English language and not the coding language.

So let me show you this in more detail.

So if we're going to display these questions, we're going to use a list.

So up here below where we have our scorekeeper,we're going to create a list of strings. And we're going to call it questions, and it's going to contain a list of all three questions.

So if we go ahead and copy our questions one by one including the single quotes, so that they go in as a string, then we can build up our list of questions and have three in our list.

So here's the second one and here's the final one now that we've got all three strings in our list, we can close off our list of questions.

And now you see that backslash in action.

Check out what happens when I delete it.

When I delete the backslash it thinks that this is the beginning of the string,and this is the end of the string.

And it interprets the rest of the text as code, which it obviously isn't.

So what we do is right in front of the single quote inside our string, we're going to add a backslash.

And now it knows to ignore this single quote and to use the next one as the end of that string.

Now that we've created our list of questions,the next thing to do is we need to update the question text that gets displayed in here to the first item in our list.

So we want this to say you can lead a crowd downstairs but not upstairs.

And I'm going to leave this to you as a challenge.

So identify the right place in your widget tree to update and try to use this list called questions to get it to show up inside this area here. Pause the video and try to complete this challenge. All right.

So let's first identify where this comes from.

Now we can either do it in a fancy way by clicking on this locate button, and then we can click on here and it'll locate to us the widget that's responsible for displaying this, which happens to be a text widget.

And this is where that text comes from.

And this is the original sort of hardcoded text that gets displayed.

Now instead of that, we're going to use our question list.

So in order to get the first item in the list, we have to write questions, to tap in to that list, and then we can either specify the index where the index is 0.

And this refers to the first item in this list of questions which is of course this one. Or we can use that trick that we saw earlier on which is questions.first

Both work exactly the same way. But in this case I'm actually going to stick to using a number and this will become clear very very quickly.

So now that we've managed to get our first text to show up inside our text widget, the next thing is we need to create some sort of tracker to track where we are in this list of questions.

At the moment we're on zero. But the next thing that we want to happen is when we click on the true or the false button, we want to progress to the next question.

So in order to track which question we're on, we want to create a new variable called questionNumber,and it's going to keep track of what is the current question that we have displayed in our text widget.

Try to complete this as a challenge and create your variable right here. So pause a video and complete the challenge. So we're going to create a new variable called questionNumber. And because it's going to keep track of where we are in our list of questions,it's pretty much going to be an integer. And it's going to start off with the value 0 to correspond to the first item in our question list.

Remember lists start numbering from 0.

And let's give it a day to type.

It's going to be an integer.

It's always gonna be a whole number because it's simply going to go from 0 to 1 and to 2. Starting with question number 0 we can now replace this part where we get questions at index 0 to questions at index questionNumber.

And if you hit save,nothing will change because questionNumber starts off being equal to 0.

So 0 goes in here and we tap into our questions list looking for the zeroth item,so no changes there. But the next thing to do is we want to increase this questionNumber by 1 whenever the user clicks on the true button or the false button.

Keep in mind what we learn in Dicee and Magic April about updating the state of the user interface. And using that hint,I want you to pause the video and see if you can figure out how to increase our questionNumber when we click on the true or false button and update our text widget.

All right.

So the place where we want to do this is of course in our onPressed.

So in this case this is where they use it picked true.

And we also want to do the same thing when the user picks false.

We're going to go to the next question no matter which choice they picked.

Right here below the comment, I'm going to try and increase the questionNumber by one.

Now you can either say questionNumber now equals the previous value of questionNumber + 1.

So if it was 0, it becomes one etc. or a slightly shorter way of doing this that you'll find in many programming languages is to simply write questionNumber++.

And this is where the programming language C++ comes from.

It's the C programming language increased by 1.

So it's kind of like the sequel to the C programming language.

This now will increase our questionNumber by 1 and we're going to do the same down here when the user presses false.

But right now if you hit save, even though we can print our questionNumber and we can assure ourselves that it's actually in fact going up by 1.

And let me just put that down here as well in both places and hit save.

Let's pull up our run and now, if I click on either the true button or the false button, you can see my questionNumber is going up. But my text widget isn't changing.

Why is that?

Well remember my hint about state because our text widget up here depends on this question number.

So when our questionNumber updates from 0 to 1, we should now be pulling in the first item from our questions list which is this question right here.

So in order for it to update, we have to get this widget tree rebuilt. And in order to mark the places where we've changed something, we have to use that set state method.

So we had the right set state and inside the curly braces is where we need to increase our question Number.

So I'm going to copy that down into the false button as well.

And now what will happen is when the button is pressed, it's going to mark anything that uses the question Number as dirty.

And the thing that uses the questionNumber is of course on text widget and it's going to trigger a rebuild of our widget tree.

And it's going to refresh this text widget with the latest question from our list of questions.

So let's check this out. Now,it's important that you keep an eye on the run area because at the moment, if I click on true, you can see that my app crashes.

And the reason is because right now, it's printing out a question number of 3 and we get an error in here. It says range error not in range 0..2.

So what's going on here?

Well notice that our question number is currently 3. But if we go into our list of questions and we count from the beginning 0, 1, 2, well there is no 3. There is no fourth question.

right?

So this is why it's telling you that it's outside of the range of this list of questions.

So what's the cause of this?

We just started our app.

Well it's because you might remember when we talked about hot reload and hot refresh. Hot reload actually keeps the state of your app.

What does that mean?

Well it means that when we have these variables such as questionNumber, and we know that every time we click on a button it grows by 1,that doesn't change. If we wanted to reset it back to zero then we have to click on hot restart instead.

So let's click on hot restart.

And now notice that the first question is displaying in the text widget.

And when we click on the true button, it will increase our question number by one.

So it will take it up to 1 and the question at index 1 is of course this one.

And now if we click on one of the buttons again, it will go to 2, which is a slug's blood is green.

Now we keep pressing then it's going to crash because we don't have any more questions.

That's all that we've got.

So our code now works until we run out of questions. If some parts of this was confusing such as how a set state works or what is the difference between hot reload and hot refresh, then be sure to revisit those modules where we covered those topics in more detail.

But once you've reviewed the code and once you're happy with it, then we're going ahead of it to the next lesson and we're going to set up the answers for the questions.