So in the last lesson we've already gotten pretty far with designing our app to make it look like the

mockup.

And now all we have to do is add our list view with our task items inside.

Let's go into our Android Studio and let's go ahead and expand everything out so that we see our structure.

So

currently we have a scaffold with a floating action button and a column.

That column contains a container which is the top part of the app

and also another container that's taking up the bottom part of the app.

So it's in this bottom container where we want to add our list view.

Let's select it so that we can get to the right place in our widget tree

and here's the challenge coming up.

You've already created list views in the last module,

you know what they look like,

you know what they require

and you've got the mockup for what we want to create.

So try and use some of the Flutter documentation or maybe have a look online to search around on Stack Overflow

and see if you can recreate the appearance of the mockup by creating your very own list view using what

you've learned so far.

And once you've had a go at it, then come back over here and I'll show you the solution.

So once you're ready, pause the video and give that a go.

All right.

So let's go through the solution.

Firstly you'll notice that the place where we want to create our list view is our container.

And this particular container doesn't yet have a child.

That's what we're going to add, the child property.

Now the child in this case is of course going to be a list view which is going to help us setup and

layout, our To Do list.

Now inside our list view, there are a whole bunch of properties that we could change

but the most important is it's children.

What's going to go inside that list?

Well in our case, we're actually going to use another handy widget,

the ListTile.

And this is going to make it super easy for us to simply just create a reusable tile which has properties

such as the title,

so what's going to go into the main part of the list.

Things like trailing, what's going to go in at the end or leading, what's going to go in at the front

and it has properties such as onTap and everything you would normally expect from a core widget.

So let's go ahead and add a title and it wants a text widget and this text widget is going to say, 'This

is a task.'

So now let's hit save.

You see our list item

show up right here.

So what about the little checkbox?

Well that's gonna go into the trailing part, so the end of this list tile. And easily enough

there's a checkbox widget and all we have to do is to give it a value.

So do you want it to be checked or unchecked?

Well let's start it out with unchecked.

So there's our little checkbox. And if you want more than one less tile then let's just go ahead and

copy another one in.

So we've got two tasks now in our list.

But notice how, once I've got my tiles in, it starts looking a little bit odd

right?

Because if we take a look at our mockup, you can see that everything is all neatly aligned and this is

really one of the designers best kept secrets. As long as everything aligns to a single point or a single

line,

then you can make anything look pretty good without a lot of effort.

So let's do that for our list view as well. So because we're in a container we of course have our padding

property.

So I'm simply going to add a padding that goes on the left and right.

So that's going to be edge insets symmetric.

So I only need to specify the horizontal,

that means the left and the right.

And it's going to be maybe a 20 pixel to match up with the previous one.

So now it looks a lot neater and a lot more designed

right?

So now that we've pretty much created our list view, it's kind of getting a bit messy in here in our

task screen. And we don't want everything to be all sitting in the same place.

So let's go ahead and extract some of our widgets. Now

firstly the one that we want to get rid of is probably going to be our list view.

So let's click on our list view in our Flutter outline and let's go ahead and extract our widget.

So I'm going to call this the TasksList and go ahead and hit refactor. And then down here inside our

task list,

we've still got our list tiles right?

So I'm going to take one of these and I'm gonna go ahead and extract it.

So this is going to be a TaskTile.

And now I've got it all refracted and we're ready to tidy things up.

As always, I'm going to get rid of the new keyword

and I'm also going to get rid of all the constructors for the automatically created or extracted widgets

because we don't really need them.

We're not actually passing anything over to construct.

And then we've got another new keyword here and that will clean up our code a little bit.

So now that we've extracted our task tile into a separate widget, we don't have to write out all the

code for the widgets anymore.

We can just simply create another task tile as in when we need it.

So let's create another task tile here and maybe let's add a third one down here.

So now we have three tasks all with that checkboxes and the titles.

Now we're at a checkpoint so let's hit save and let's go ahead and create a new folder to keep hold

of each of these little items, little extracted widgets. And I'm going to call this folder

the widgets folder because these are kind of all custom widgets that we've created

right?

And inside the widgets folder let's create some Dart files.

One of these is going to be that tasks\_list and another one is going to be the task\_

title.

So in both of these we're going to need to import our material package.

So we might as well go ahead and do that now. And then I'm simply going to copy over the relevant stateless

widgets.

So tasks list is going to go into here

and the task tile is gonna go over here. So now we've got a couple of errors in our Dart analysis.

The method task list isn't defined for this class so it doesn't know about it because we haven't imported

it.

So, naughty us.

So let's go ahead and add widgets/tasks\_list.dart

and now it knows exactly what we're talking about.

But over here, we've still got three errors because it doesn't know what task tiles are.

So let's go ahead and import our widgets folder and our task\_tile.dart.

So now, we've pretty much gotten rid of all of our errors, so the ones in red, and all we have now are

simply just some warnings,

so the ones in yellow. And inside our app, we've pretty much got our whole interface created where we've

got the top area, our column of title and icon and little bit of subtitle and then we've got our little

list here which is scrollable, we've got checkboxes and the list items and we've got our action

button.

Now don't worry if you you're trying this out and it doesn't work. The checkbox won't work yet because

we haven't yet added any code that tells it how to behave, which we're going to do very shortly. But for

now if we take a look over here in our Dart analysis, we've got some things that we need to sort out. Firstly

the parameter onPressed is required.

So whenever a widget has required properties or parameters that we haven't given it, it will highlight

it in this light yellow.

And when you hover over it, it'll tell you what's missing, in this case our floating action button has

all of this nice layout but it actually doesn't yet know what it should do when it's pressed.

So let's add our onPressed call back and inside here, what should happen.

Well when we click on this plus button here, we actually want it to slide up something that allows us

to input a new task. And if we look back at our mockup and we go onto the second page, you can see this is

what we're kind of trying to do right?

We're gonna get a new container that slides up and it's also got the nice round edges and here we'll

be able to type in a new task and then add it to our to do list.

So that's what we're going to be tackling in the next lesson.

So for all of that and more, I'll see there.

