Now we can see that in the final version of our app when we click on either of these icon cards, they actually change color to show that they've been interacted with and they've been selected. But currently our cards don't really do anything when we click on it. And we can see that our cards or what it's based on be at the iconContent or the reusableCard,none of them actually can take any user input. They don't have an onPressed or an onTap or anything of the sort.

So let's fix this in this lesson and let's make our icon cards interactive. Now one way of doing this is we could wrap our reusableCard in a flat button. So we could hold down OPTION or ALT and hit ENTER.

Then we could wrap with the new widget and the widget would be a flat button.

And this of course will give us access to that crucial onPressed which allows us to tap into the moment when a user actually presses on either of these reusable cards, and we'll be able to change the color that way.

Now the problem with flat button is if I hit save right now, you'll be at to see that it actually affects the styling of our card.

It's trying to change the margin and various other styles of its children.

So it's quite an opinionated widget.

And the use case for a flat button is you want to be a to implement the style of a material flat button,and that includes various colors various themes and its appearance.

But in our case all that we actually need is just the ability to detect a touch on our already pretty much well styled reusable card.

So instead of a flat button, we can use something called a gesture detector. A gesture detector does pretty much what it says on the tin,right?

It detects gestures and the gestures it can detect are things like onTap, which is the same as onPressed,but it can also detect a whole lot more.

So for example it can detect a long press or drag or force press and a whole lot more.

All that this widget is concerned with is detecting gestures.

It doesn't try to force any sort of styling or appearance or animations onto the children widgets.

So this is a very pure way of detecting when a user interacts with a widget of ours.

So by having our reusable card as a child of our gesture detector, we can now start tapping into the gestures that it can detect.

So for example, onTap is your classic single tap on the screen.

So it's the same as the onPressed of a flat button. Or we can detect a vertical drag, horizontal drag,double tap, force press,whole lot more.

But all we need right now is the onTap gesture detector.

So I'm going to add our classic anonymous callback and inside here, I'm just going to print that the male card was pressed. And now if we hit save and go into our app making sure that our run tab is showing,and if I click on that male button, you can see that we get 'Male card was pressed' every single time I press on it.

So it works just like a flat button and it's really easy to implement. Now that we've got our gesture detector, we're able to detect when a user actually clicks on our cards, be it male or female.

And what we want to happen is when we click on the male card, we want it to change color to show that it has been selected.

And in order to do this we have to, of course, use our set state.

And we're going to update the color property of our reusable card.

This is why we have a stateful widget here so that we can actually change the state of our cards. The way that we're going to do this is our cards will start out with a slightly darker background color.

And when it's tapped, it will change to have the same color as the rest of the card.

So the active color.

So firstly up here in our constants, we're going to have a activeCardColor and we're also going to create a new constant for the inactiveCardColor. And the inactiveCardColor is also going to be a custom color and it's going to have a hex value of 0xFF111328 .

And you can see over here in the gutter that it's just a little bit darker from the activeCardColor, but it's not exactly the same as the main background color.

It's just a little bit lighter.

And if I go ahead and simply change the color for the male card for example to the inactive card color and hit save, then you can see that it's just a little bit darker than the active color but it's not quite the same as the background color so it doesn't just blend in.

So you can still see that there is a card here but it's obviously not selected and not activated, whereas this card looks the same as the other cards.

So it looks like it's the activated or the selected card.

So this is the look that we're going for.

And both cards should start out having the inactive card color but when one of them is tapped on, then we should change that color to the active card color.

So to do this we're going to go into our state widget which is mutable so its properties can change and mutate, so the properties for this class don't have to be final.

They can be variables that can change.

So the first one we're going to create is going to be of type color.

And it's going to be called maleCardColor and it's going to be set to equal the inactiveCardColor.

So this is going to be the starting value for both the male card color and also the femaleCardColor.

And we're going to be using this variables, femaleCardColor and maleCardColor inside our reusable cards.

So this way, we're able to selectively change the color for these cards depending on which one was selected.

Now in order to make our cards change onTap, we're going to create a method called updateColor.

And this is going to take an input which is going to somehow tell us which gender was clicked on.

So it's going to be called gender and it's simply just going to be an integer right now.

And we could say that maybe for example, if we pass in one then that equals male and if we pass and two then that equals female. Inside this method, we're going to use an IF statement to check to see if the gender is equal to 1.

So if we pass in the value one, then this means that it was the male card that was pressed.

Then in this case we're going to check to see if the maleCardColor is currently equal to an inactive CardColor.

Well in that case, we're going to change it to the activeCardColor.

So we're going to toggle the color of the card.

But if it was already in the activeCardColor, then we're going to change it to the opposite which is going to be the inactiveCardColor.

So now if we go into our onTap and instead of printing the 'Male card was pressed', we use a set state to update the state of our maleCardColor property which is used to determine the color.

Then we can call that method that we created just now updateColor.

And we said that the number 1 was going to represent the male gender.

So now if we hit save and you can see that my card both start out being the inactive color.

So they're not quite the same as this one but as soon as I click on it changes to the active color.

And that's because I'm passing in 1 into this method and while maleCardColor started out being inactive,as soon as it detects the touch, then it'll update it to the active color.

But right now it's the active color so if I click on it again, it'll go back to the inactive color.

So this way we can toggle the background color of the card to be able to represent when something is selected or not.

So let's go ahead and do the same for the female card. All we need to do is to add another ELSE, if the gender is not equal to one or if you want to have more than one gender in your app, then you might simply just add an IF statement that checks if gender equals 2.

So we'll say that 2 is equal to female card pressed.

And in this case, we're pretty much going to do the opposite.

So we're going to check to see if the femaleCardColor starts out being inactive.

If so, we're going to change it to the active color but otherwise we're going to toggle it to inactive.

So now inside our female card, we're going to also wrap our reusable card inside a new gesture detector.

And this is also going to have a onTap which is an anonymous function that gets triggered when the tap gets detected.

And when that happens, we're going to call set state to update the color and we're going to pass in the gender as number two, to say that the female card got pressed. So let's hit save and let's check out our app.

So now when I click the male card, the male card lights up and I can deactivate it by clicking on it again.

And if I click on the female card, the female card gets activated.

So now we're able to select a gender in our BMI calculator.

Now at the moment though when I click on the male card when the female card is selected, the female card stays selected right?

But let's say that I made a mistake and I actually meant to select this one, then when I click on it should turn off this one because they probably shouldn't both be selected at the same time.

So how can we do this?

Well it's pretty easy although it's going to add to our already super wordy method.

Don't worry, we're gonna make it a lot easier on the eye a little bit later on.

But we're first mapping out the functionality so we understand what's going on.

So if updateColor is called and it was the male card that was pressed, so gender is one, then we're going to not only switch on the maleCardColor, but we're also going to switch off the femaleCardColor. So we're gonna change this to inactiveCardColor. And the opposite is also true when the female card gets pressed,then we're going to switch off the maleCardColor.

So we're going to change that to inactive.

And now let's hit save and you can see that if I click on the male card, the female card switches off.

If I click on the female card, the male cuts which is off.

And this behavior makes a lot more sense. Now in the coming lessons,we're going to simplify all of this logic to something that's a lot shorter and more succinct and easier on the eyes.

Because at the moment, it's super wordy and very confusing what's actually going on.

But in order to achieve this, we need to learn a little bit more about Dart.

So in the next lesson we're going to talk about Dart enums. For all of that and more,

I see you there.