**Modify an XML widget, create a button**

In Android, user interface elements are called widgets. Widgets can do several different things, including interaction for the user, layout components, and display media, such as images and video. If we look at our project, we can look at the existing widgets that are in our main activity. Android studio has two main ways were you can work with your activity layout, called content\_main.xml. Currently, I'm looking at the text view. This shows the XML code that defines each of our layout components. You'll see we have a relative layout widget, that contains a single text view widget.

The text view widget has a number of XML attributes that are organised on multiple lines. Although it is on multiple lines, the XML tag for the text view widget, is a self closing tag. It is only the attributes that are listed on multiple lines. The surrounding relative layout widget has a distinct, opening tag at the top, and a closing tag at the bottom. We can adjust properties of our text view, by entering in XML code. For instance, we can change the text, that displays in the widget by updating the android:text attribute, in the text view tag.

Let's change it to the phrase "Let's Play!". The android portion at the beginning of the attribute, is what the SDK uses to differentiate android oriented attributes from others. Over time, you will work with other SDKS, that will have different prefixes for attribute names. When you change the contents of the tag, you'll the widget in the preview pane updates as well. There's an alternate way to work with your widgets, and that is design view. Open that by clicking the design tab, at the bottom of the window.

The XML code goes away, and you'll see a palette of available widgets. The widgets that are currently in your project, called the component tree, a thumb preview, and the properties panel. Select the TextView widget we modified earlier. You can do that in the component tree. Or simply click the widget in the preview area. When you select the widget, you'll see the properties display a number of options. These are a list of common properties, that are associated with the TextView widget. You'll see the property "text". It will display the phrase we added in XML. This is just a sub-set of all the properties.

If you click the link at the bottom, or the double arrow button at the top, you can flip between common properties, and all properties. Let's add a new widget to our app. Let's add a button. In the palette panel, scroll to the top, and you'll see the button widget. Click and drag it to the preview pane, below the TextView that we already have. As you move the widget around, before you release the button, you'll see some guidelines appear, to help you snap the sides, center, and spacing around other widgets.

Move the widget to the center of the display, and release the button. You'll see the properties panel display some information, about our new button. The first is the item ID. This is the name you'll use to connect Java code to the widget. It defaults to a generic name. But let's change that to rollButton. Notice I start with a lower case "R", and I have a capital "B" but button. Capitalization is important with Java and XML. They need to match when we connect our XML code, with our Java code that we'll create later.

Lower down on the properties panel, you'll see a text property. This will set the text of the button. Even though the phrase is in a mixed case, the button is using the material design style, that converts it to all caps. Let's update the field to read "Roll!", with an exclamation point. When you press enter, you'll see the preview update, with the new text for the button widget. All the changes you make in design view generate XML code in the background. If we switch back to text view, we can see the changes that were made. You'll at the bottom, that we have a new button tag.

Inside of this tag are a number of attributes, that define the properties we created for our widget. There is an error that will appear in the layout underscore below attribute. We will fix that soon, so you can ignore that for now. Android studios created each of these attributes, based on where we position the widget, and the fields we updated in the properties panel. Such as the text and ID fields. We can adjust properties entirely in code if we want, in fact, let's alter the amount of space between the button and the text field above it. Widgets have two sets of spacing properties: margins refer to the amount of space outside of the walls of the widget, padding refers to the amount of space inside the widget walls.

If we increase the padding, we're making the widget bigger. If we increase the margin, we're creating additional space around the widget. Let's add some more space between the Button, and TextView widgets. In XML code, look for the attribute, layout\_marginTop. This will set the amount of space above, or at the top, of the widget. When working with mobile devices, pixels aren't always the same. Some devices have low density pixels, whilst others have super high density pixels, to create nice, clean, crisp displays.

To help manage various pixel densities, or dpi, android uses a measurement called device pixels, or dp. You'll see our Layout\_marginTop property has a value set in device pixels. The app runs on various different screen types, it will use the dp setting, and proportionally adjust so that the layout looks consistent, even if the pixel sizes vary. Let's set the top margin to 25 device pixels. Enter two five, and make sure that you still keep dp. You'll see the widget move a bit in the preview, when you make the adjustment.

These changes are visible back in the design view as well. Click back to design view. Display all the properties using the toggle at the top, and we expand the field for layout margin, we'll see Layout\_marginTOP is set for 25 device pixels. Widget properties can be viewed and modified, in a number of different places, both directly in code using text view, and using the properties panel in design view.