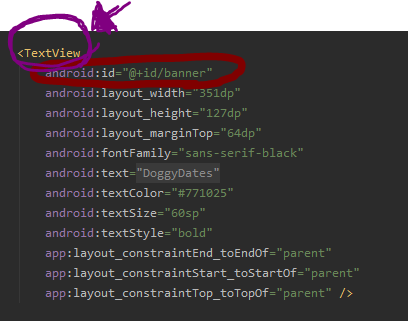
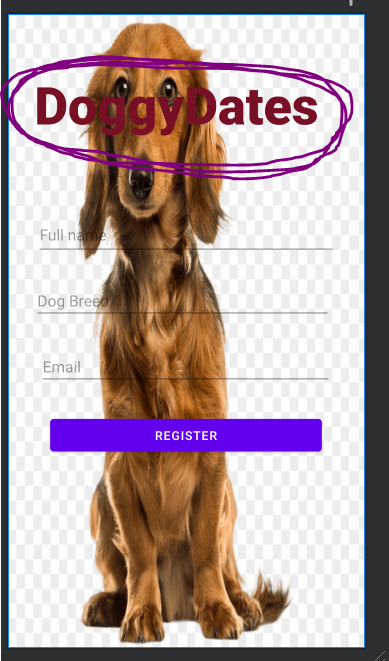
***XML***

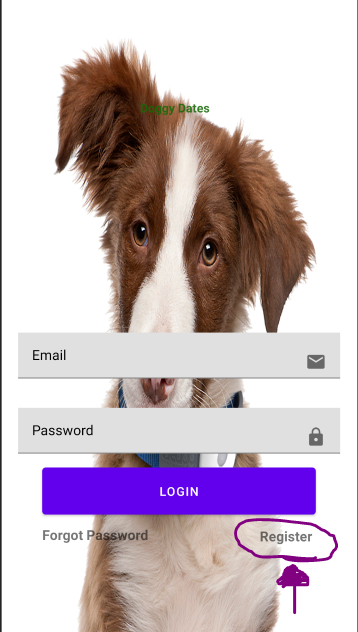


As you can see in the above image, XML calls the thing that you would like to include into the screen GUI. Here we see that I add a “TextView” with the red circling that it has an id equaling “banner”. It is important to note the “@+id/” portion is in order to call it by id name in as either java or Kotlin code. The rest is stuff to include into the added GUI subject. For example, we can see that a line exists starting with “android:text” which is set to “DoggyDates”. Everything you see in the above picture is to create the title for the App in the Registration window as seen in the picture below circled in purple.

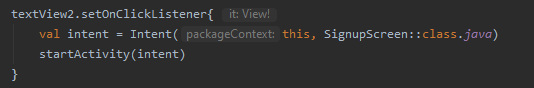


***Text View as a Button***

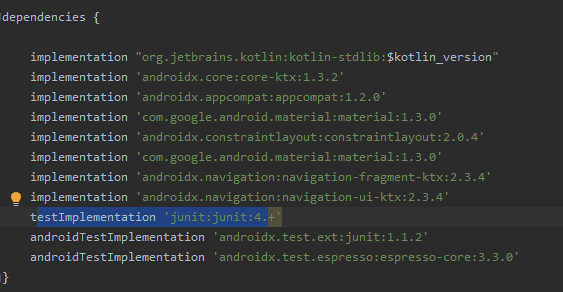
So in the below picture we can see that there is a Text View in the bottom right which says “Register” circled in purple. We can see that this is not a traditional button, yet this Text View acts as if it is a button. The reason this was done is to make the screen look cleaner and not so full of what a lot of buttons can do to the screen making the GUI for the user a better experience by not overloading the visual senses.



Now to do this you put the usual XML for including objects onto the screen. For this Text View I decided to just call it “textView2” simply because only three would ever be needed on this screen. To make the text view act as a button simply go to the kotlin coding area for the current screen and add the following text with the name of your Text View instead of what mine is called.

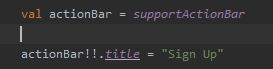


The dependencies in the gradle.build file are shown below to get this to work.



***Action Bar***

To include stuff into the top of the action bar. Below you can see that I called the included android library and assigned it a value to change the name of the window while on the screen. To do this simply go to the desired screens connected kotlin section and include the code shown below; same dependencies as above. This will only be viewable on an emulator or an actual phone. Android Studio will not show this in the Design window.



Below picture shows the code being used in the action bar circled with purple.



***ProgressBar***

XML is shown below and note the line for visibility. Later simply make visible, invisible as needed with kotlin or java page for the desired page. This creates the circle loading animation that we often see when loading a page and such.

