


KNOW YOUR ANIMALS LAB

In this lab, you will create a picture dictionary that helps users identify various animals. Fill it with animals you think are interesting, including the exotic and fictional!

PART 1: OPENING THE PROJECT

Open the provided project template and run the app (clicking on the “Run app” icon  or “Debug app” icon  located in the menu near the top right hand side of the Android Studio window).

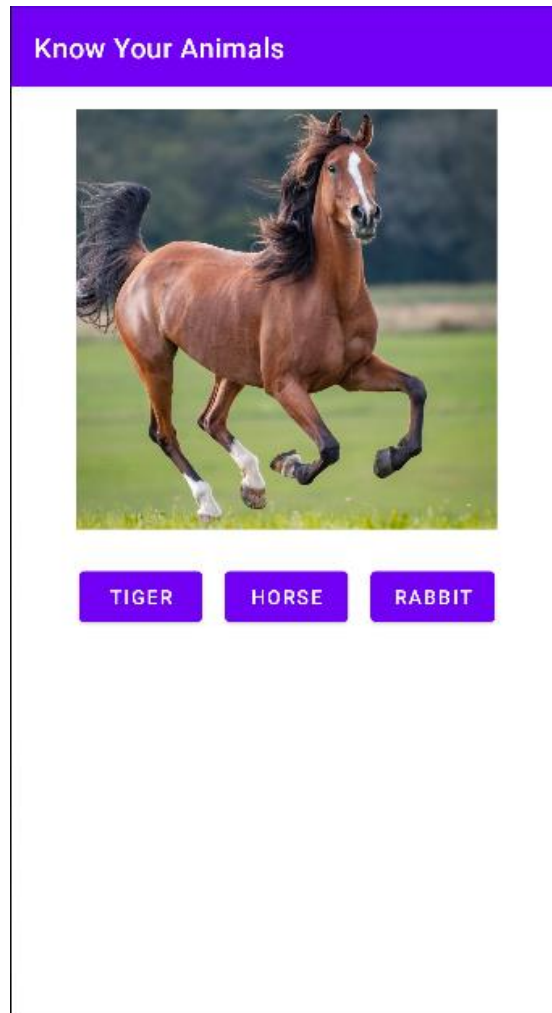
You will see that the app launches on the emulator looking like this:



There is an image of a horse right next to a button labeled “TIGER”. Clicking on the button does not affect anything in the app.

This seems like a weird way to lay out this screen. What’s causing it to look this way?

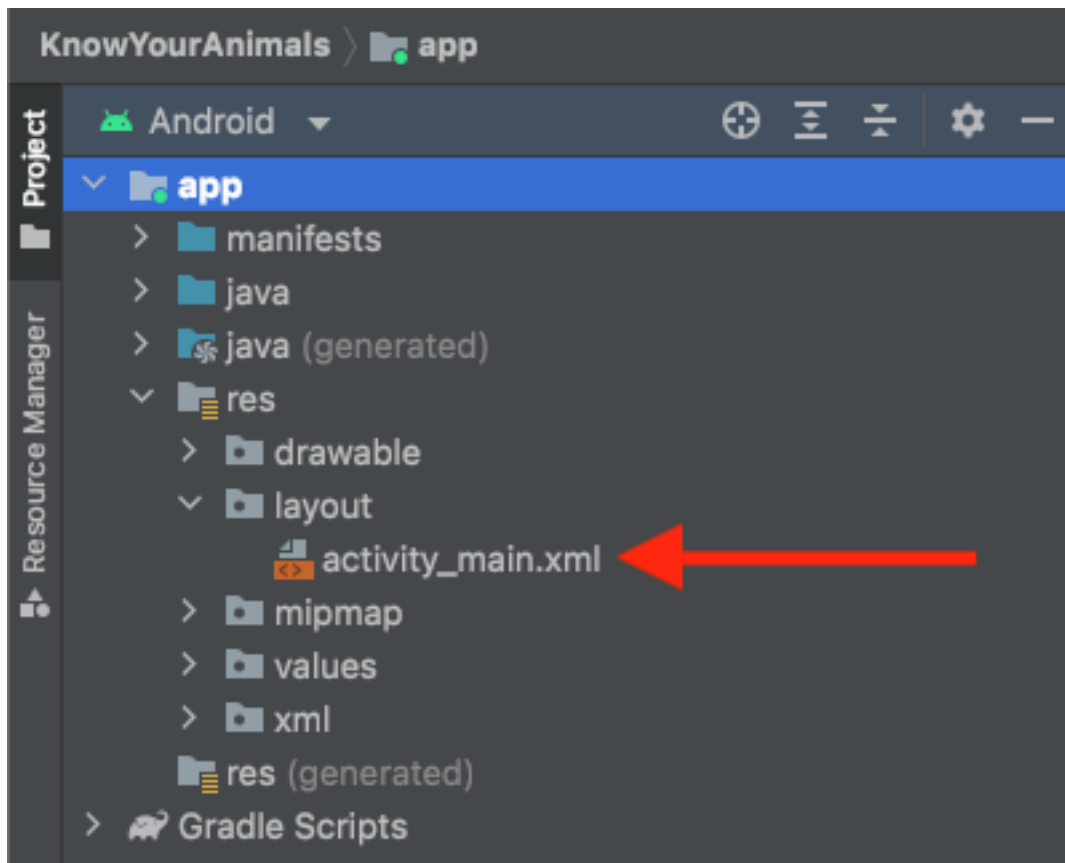
HINT -- the screen was meant to look like this:



PART 2: FIXING THE BUG IN THE LAYOUT

Since the issue looks like it’s in the layout, let’s take a look at the associated layout file “activity_main.xml”.

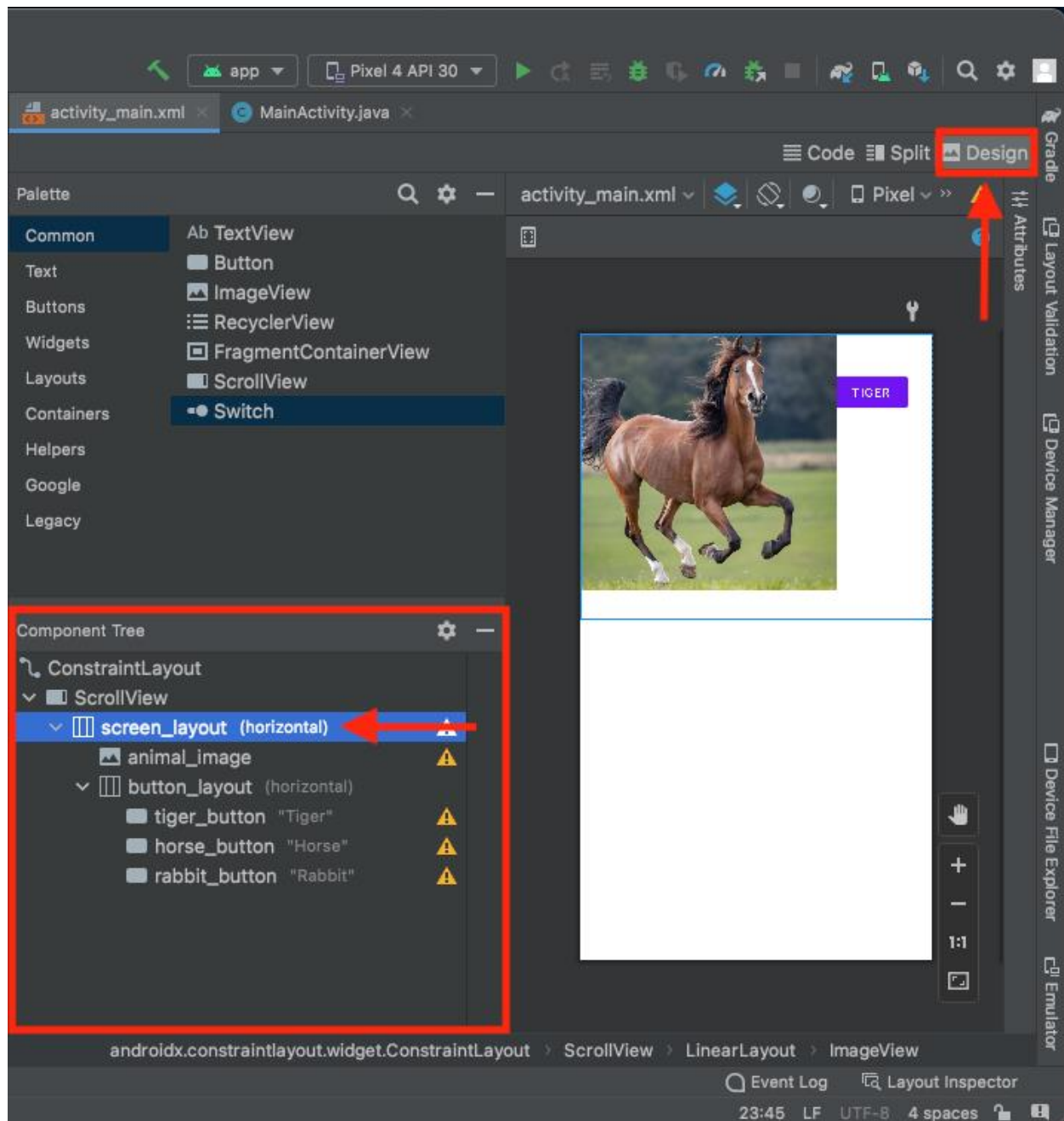
Click on the “Project” tab on the left side of the Android Studio window to open up the Android project structure, then open the layout file “activity_main.xml” under app/res/layout folder.



Here, you'll see that the screen contains an `ImageView` (ID: "animal_image") and a `LinearLayout` containing 3 buttons (ID: "button_layout"). They are both arranged inside a `LinearLayout` (ID: "screen_layout").

```
13      <!-- Learn Your Animals lab: What is the bug in this layout? -->
14      <LinearLayout
15          android:id="@+id/screen_layout"
16          android:layout_width="match_parent"
17          android:layout_height="match_parent">
18
19          <ImageView
20              android:id="@+id/animal_image"
21              android:layout_width="300dp"
22              android:layout_height="300dp"
23              android:layout_gravity="center_horizontal"
24              android:layout_marginTop="16dp"
25              android:layout_marginBottom="16dp"
26              android:scaleType="centerCrop"
27              app:srcCompat="@drawable/horse"/>
28
29          <LinearLayout
30              android:id="@+id/button_layout"
31              android:layout_width="match_parent"
32              android:layout_height="wrap_content"
33              android:layout_marginTop="8dp"
34              android:layout_marginBottom="8dp"
35              android:gravity="center">
36
37              <Button...>
38
39              <Button...>
40
41              <Button...>
42
43          </LinearLayout>
44
45      </LinearLayout>
```

According to the hint above, the ImageView should be placed above the Buttons. But currently on the screen, the ImageView is placed to the left of a Button, and not all the Buttons are displayed. So it seems that inside the LinearLayout “screen_layout”, “animal_image” and “button_layout” are arranged horizontally instead of vertically. You can confirm this inside the Component Tree panel, which can be seen when viewing the layout file in “Design” mode:



To arrange “animal_image” and “button_layout” vertically, set the “orientation” attribute of the LinearLayout “screen_layout” to “vertical”. If not specified, the “orientation” attribute of any LinearLayout is set to “horizontal” by default.

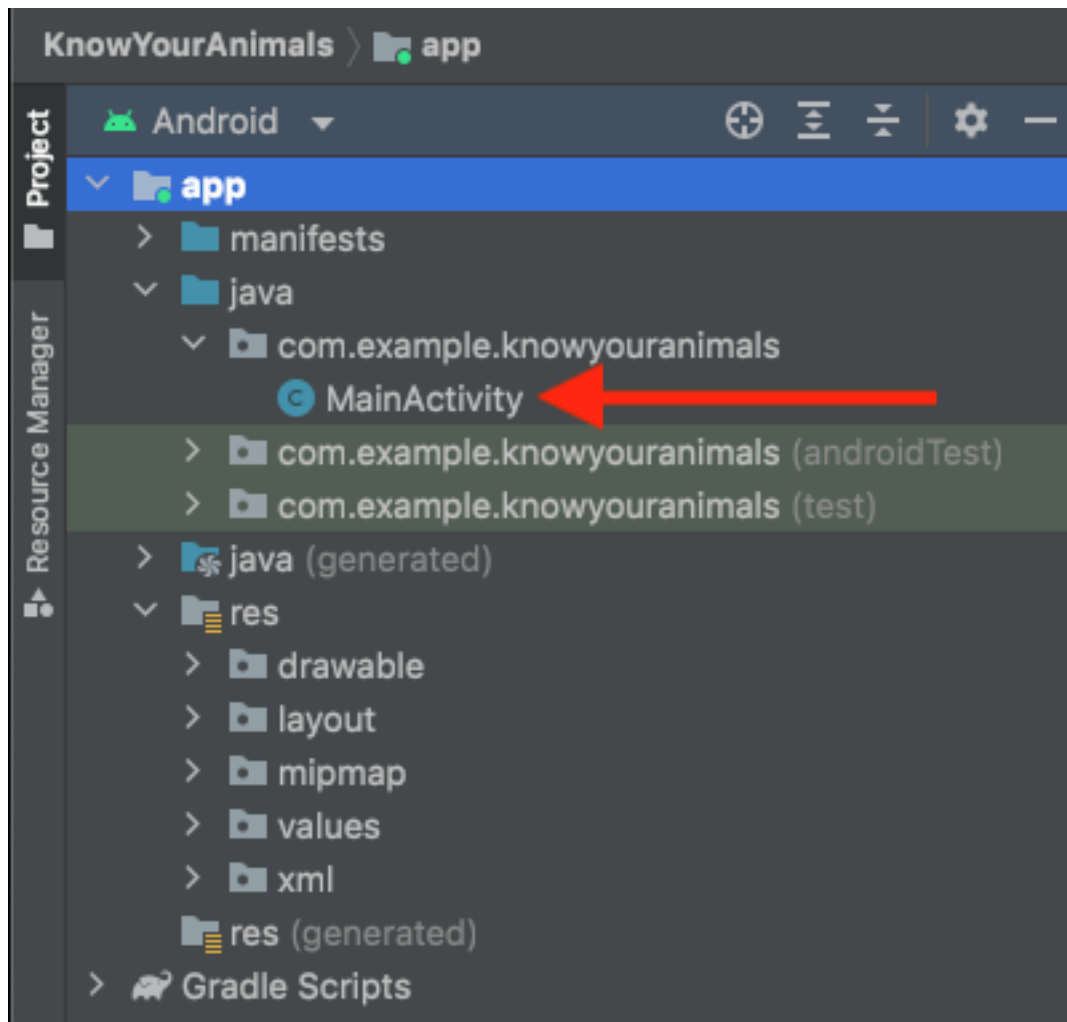
```
<LinearLayout
    android:id="@+id/screen_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
```

When running the app again, you should now see that an ImageView is arranged to be above 3 Buttons.

PART 3: HANDLING BUTTON CLICKS

Inside “button_layout”, there are 3 Buttons (IDs: “tiger_button”, “horse_button”, “rabbit_button”). Nothing seems to happen when any of these Buttons are clicked. To help the user understand what a tiger, a horse, and a rabbit look like, let’s define each Button’s behavior to show the image of the animal the Button represents when the user presses on it.

Click on the “Project” tab on the left side of the Android Studio window to open up the Android project structure, then open the file “MainActivity.java” under app/java/com/example/knownyouranimals folder.



Have each Button change the image inside the ImageView “animal_image” to display the animal the Button is referring to when clicked.

An example of how this works in code is provided for the Button “horse_button”.

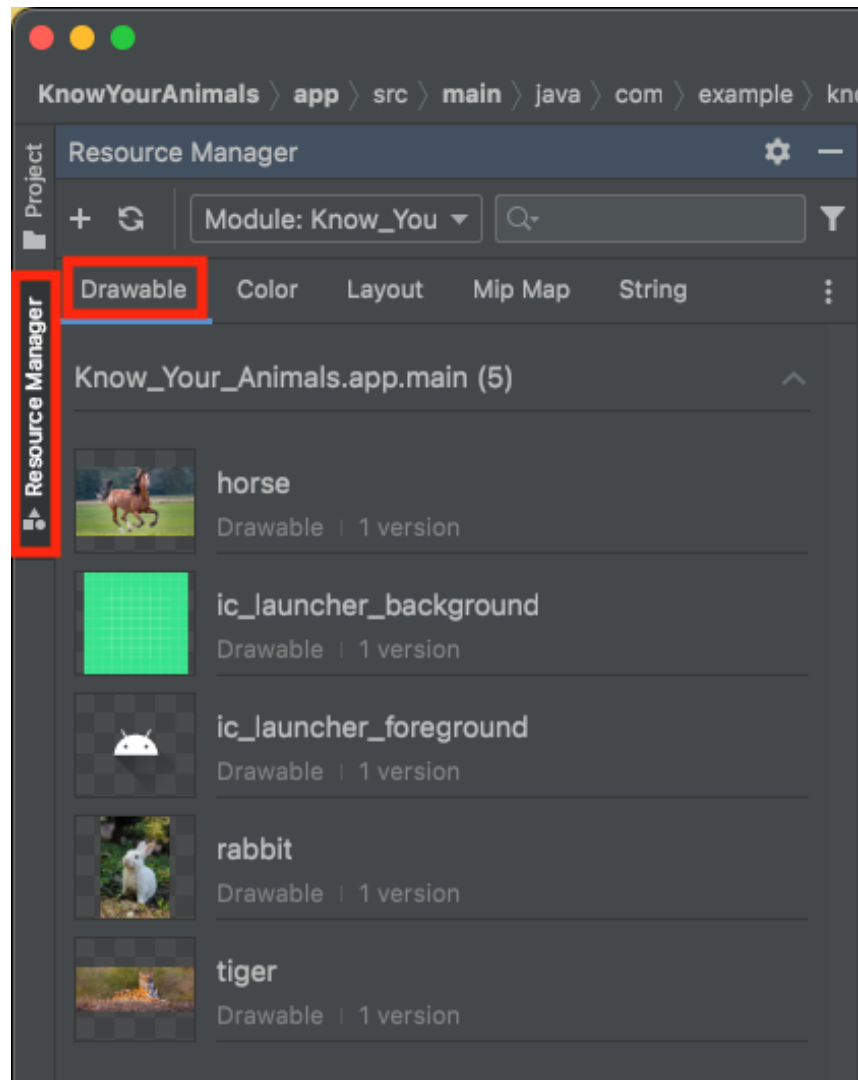
```
ImageView animalImage = findViewById(R.id.animal_image);

Button horseButton = findViewById(R.id.horse_button);
horseButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        animalImage.setImageDrawable(getDrawable(R.drawable.horse));
    }
});
```

Breaking this logic down:

1. Get a reference to the Button object using the ID of the Button defined in “activity_main.xml”.
2. Set up a listener to detect clicks and handle what should happen when this Button is clicked.
3. When the Button is clicked:
 - a. Retrieve the relevant drawable resource
 - b. Have the ImageView “animal_image” display this drawable resource

You can view the available drawable resources you have at your disposal by clicking on the “Resource Manager” tab on the left side of the Android Studio window (below the “Project” tab). All drawable resources can be viewed under the “Drawable” section. In the Java code, use “R.drawable.[drawable_resource_name]” to identify the drawable resource you want to use.

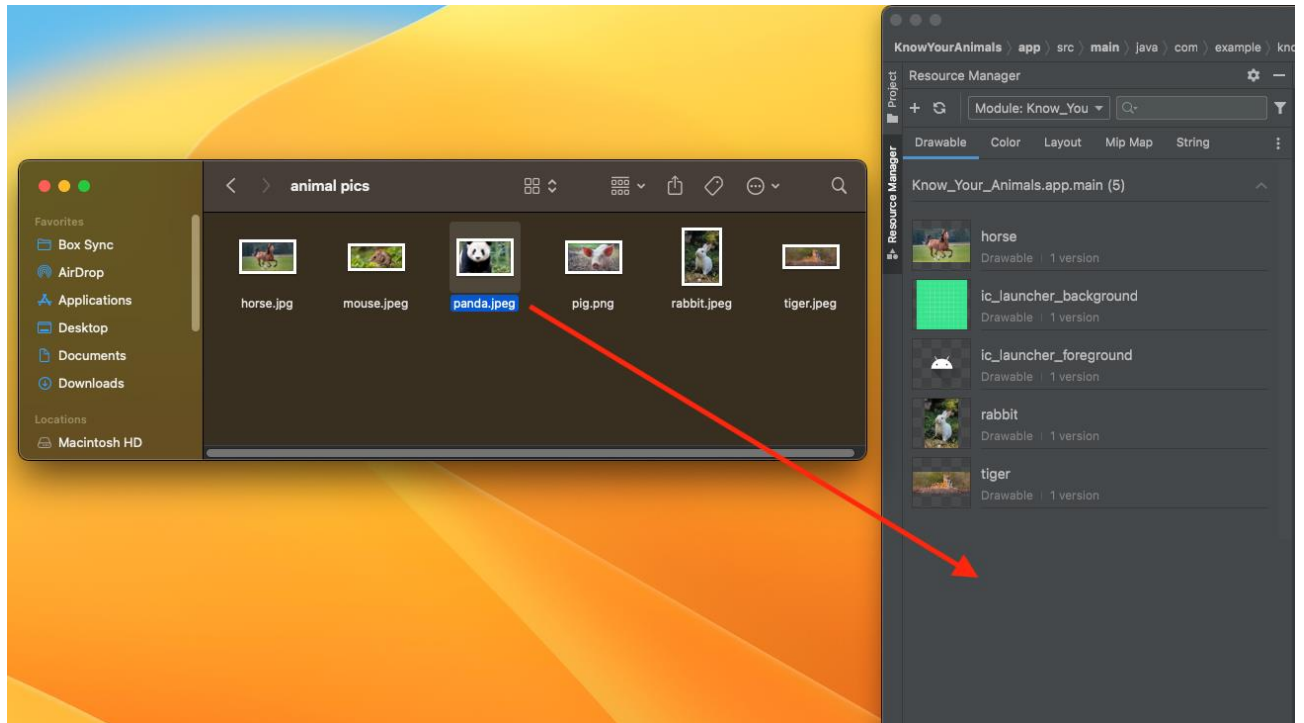


PART 4: ADDING MORE ANIMALS

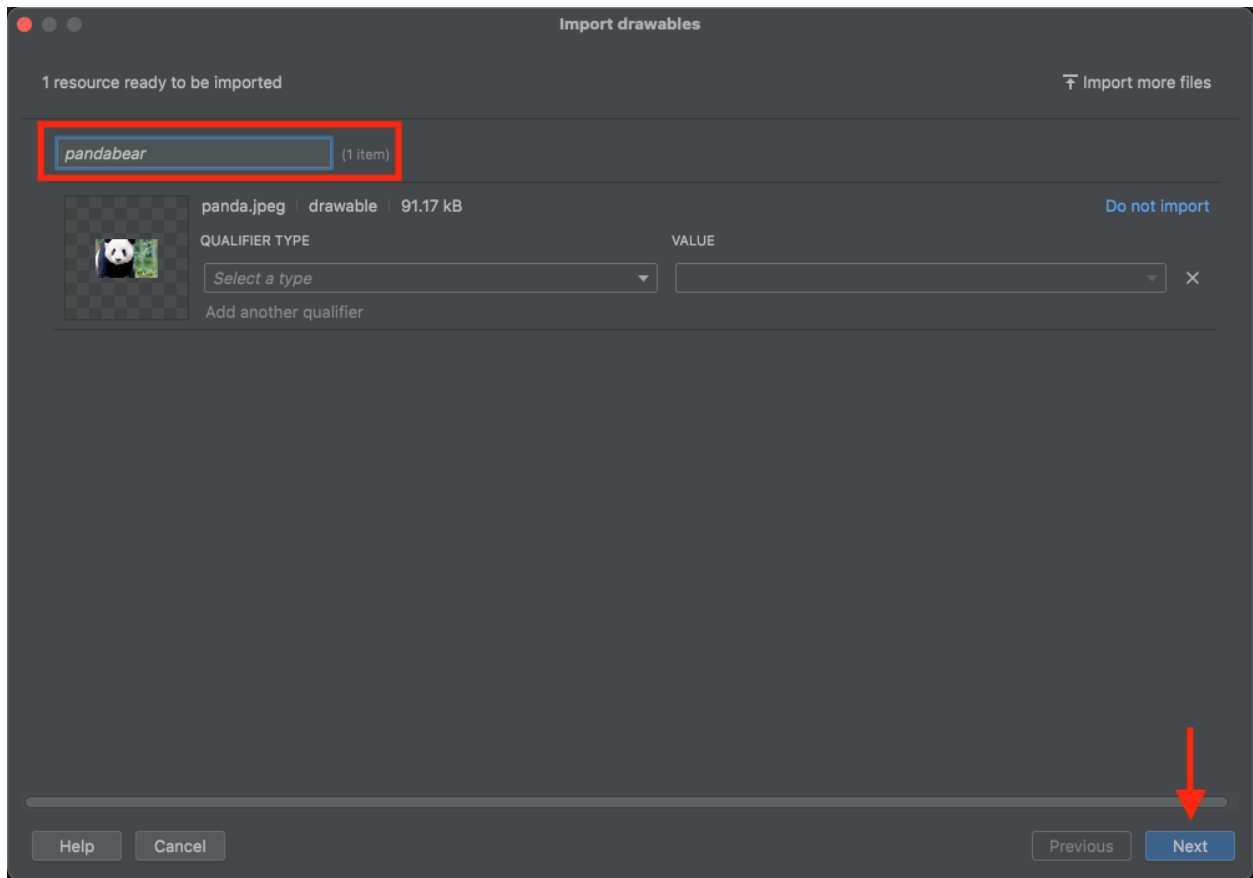
Your users most likely already know what a tiger, horse, and rabbit look like. Feel free to add more animals!

To do so, you’ll need to add an image of the animal you want to add as a drawable resource in Android Studio.

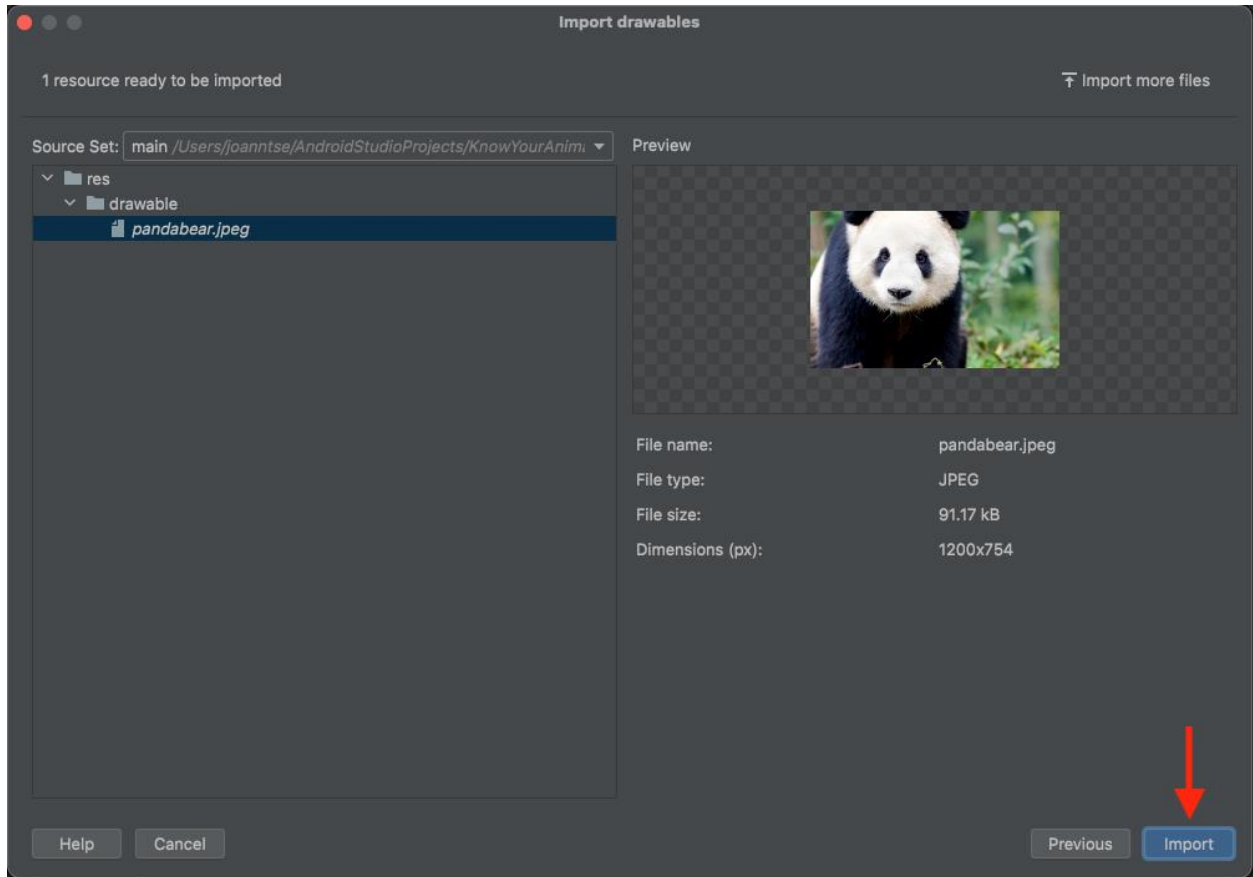
1. Drag an image file of the animal you want to add into the Resource Manager panel under the “Drawable” section.



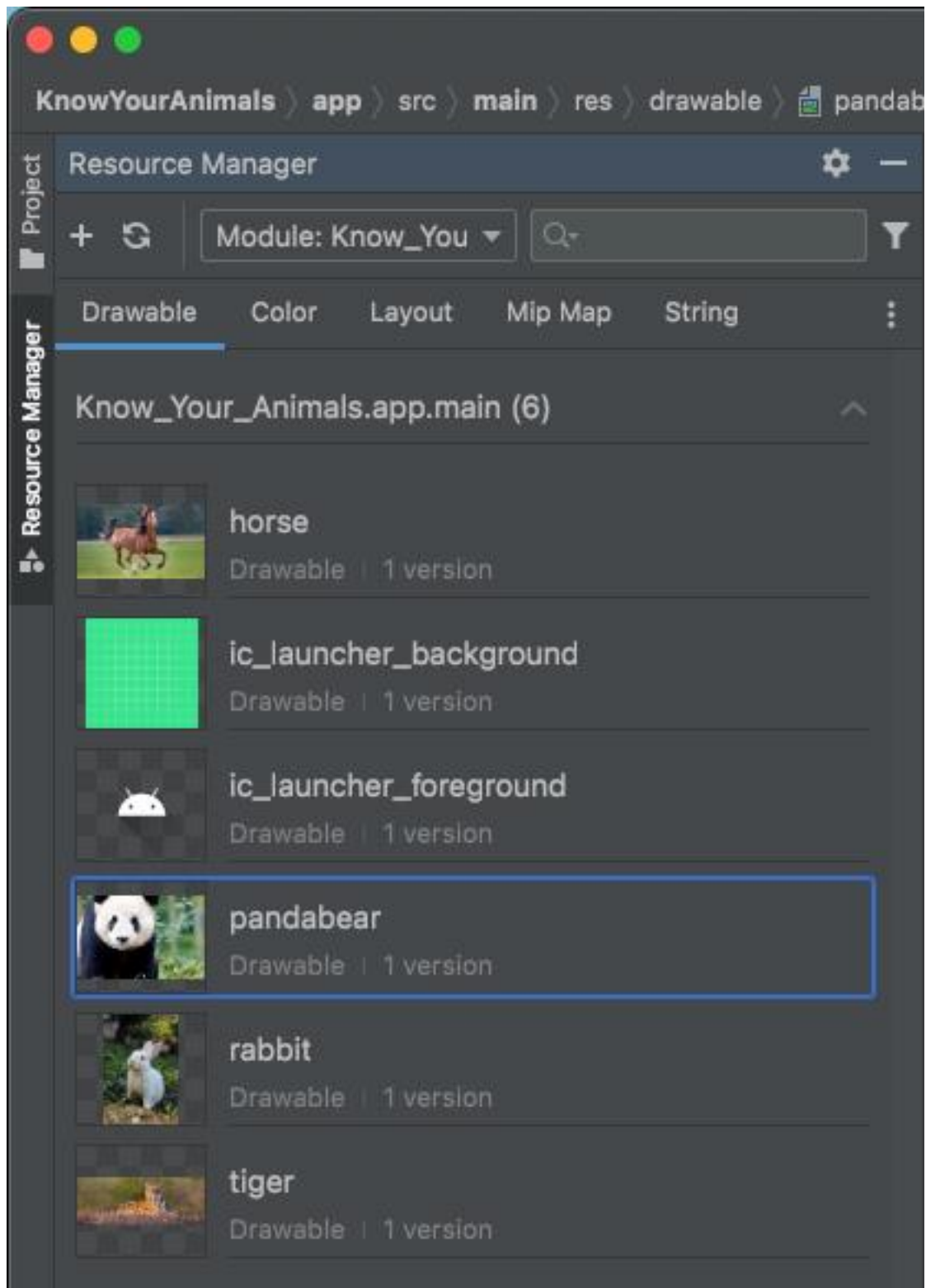
2. A window will pop up showing you the image file you are about to import as a drawable resource. By default, the drawable resource will be given the same name as your image file, but you can also choose to rename the resource. Click “Next”.



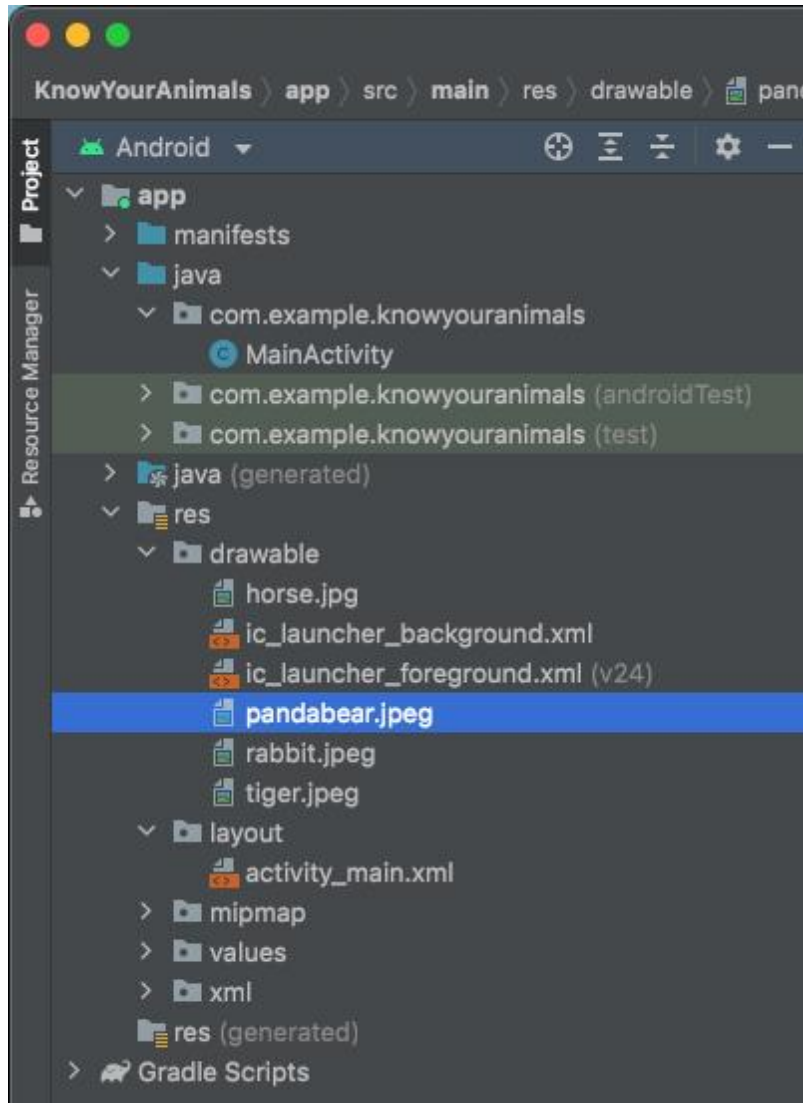
- Here, you are confirming that this drawable resource will be available under the app/res/drawable folder. Click “Import”.



In this example, you can now see the new resource “pandabear” in the available list of drawable resources.



Going back to the project tab, you'll also see that the drawable resource is available under `app/res/drawable` folder as `"pandabear.jpeg"`.



Now that you've added the drawable resource, you can:

- Add a button for your new animal in the screen layout defined in the layout file `"activity_main.xml"`
- Define the behavior of this button to display your new drawable resource when it is clicked in `"MainActivity.java"`

PART 5: TESTING THE APP

When using the app, clicking on any of the buttons should reveal an image of the animal the button was labeled with.

