**HelloPurrWoof**

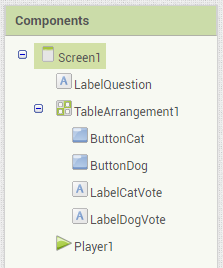
Start a new project in [App Inventor](http://ai2.appinventor.mit.edu/) and name it *HelloPurrWoof*.

Now we need to add the different ways users will interface with our app

* From the User Interface drawer, drag in a Label.
  + Rename the label **LabelQuestion**
* Under this label, drag in a *TableArrangement*,
  + From the User Interface drawer, drag in two Buttons
    - Rename the first button **ButtonCat**, rename the second **ButtonDog**
  + Below the buttons, drag in two Labels.
    - Rename the first label **LabelCatVote**, rename the second **LabelDogVote**

We also want to be able to play some sounds. To do this, drag in a Player component from the Media drawer.

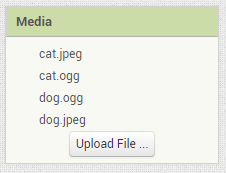
Your Components window should look like this:



Now we need to change the text of the labels and buttons.

* In the Properties tab of **LabelQuestion,** change the text to “Dogs or Cats?”
* Change the text of **LabelCatVote and LabelDogVote** as well to “Votes for Cat: 0” and “Votes for Dog: 0”.
* Finally, remove the text from the buttons.

We need sounds and pictures for our app. Find a picture of a cat, a picture of a dog, a meow sound, and a woof sound. Upload them to the Media window.



Under the Properties window of **ButtonCat** and **ButtonDog**, set the *Image* property to your uploaded images. You may need to adjust the size of the buttons or *TableArrangement* based on the sizes of these images.

It’s time to code our app! Switch to the Blocks window (top right corner).

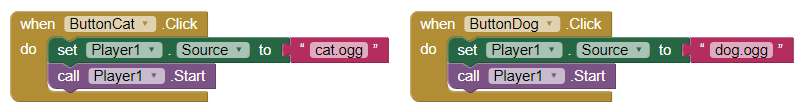


We want it so that clicking on the Cat button makes a cat sound, and adds 1 to the Cat Vote counter. Clicking the Dog button will make a dog sound, and add 1 to the Dog Vote counter. To do this, we need two variables for our vote counts. Pull out two *initialize* blocks from the Variables drawer, and pull out two 0’s from the Math drawer.



Then, we need to program the Player to play the sounds (in this case, cat.ogg and dog.ogg).

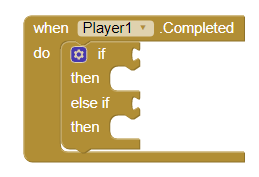
* Pull out **when ButtonCat.Click** and **when ButtonDog.Click** from their respective drawers.
* **Set Player1.Source** and **call Player1.Start** can both be found in **Player1**’s drawer.
* Finally, the empty text block can be dragged in from the Text drawer.



Now, we need to increment the vote counts. We could implement these within both Button Click events. But that’s repetitive code. We could also increment the counters in one place, with an “if else” block. Pull out a **when Player1.Completed** block.



Pull out an *if* block, and add an “*else if*” in.



From here, how can we tell if we had selected the dog or cat button? The player source! So if the **Player1.Source** is the cat sound, then the user had selected Cat; likewise for Dog. Here is the code for checking the source for the cat sound; writing it for the dog is up to you.



Now, if the cat button had been pressed, we need to “increment” *CatVote*; that is, *CatVote* = *CatVote* + 1, and likewise for *DogVote*. This code is up to you to write!

Finally, we need to update the **LabelCatVote** and **LabelDogVote**. You can do this with a “join” block. DogVoteLabel’s update looks like this:



Test your app! Does it work?

**Further Exploration Questions**

Why does the LabelDogVote and LabelCatVote update happen at the end? What happens if they are done before?

What are some other ways we could increment CatVote and DogVote?

Could we use two players instead of one player and changing the source? What else in the code would have to change if we used two players?

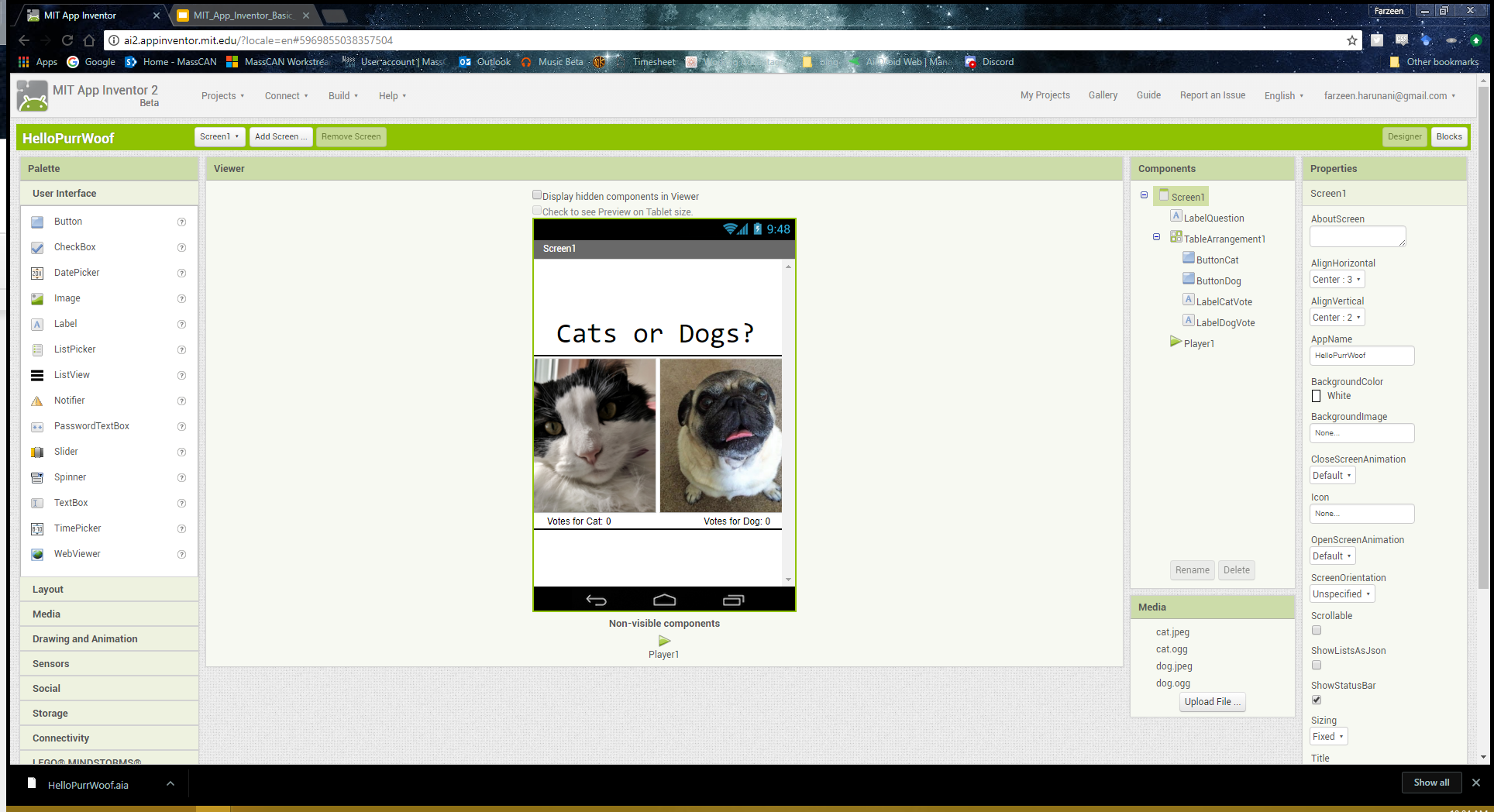
Is there anything to stop someone from voting 20 times?

What would we have to add so that we can click a button and “reset” the votes back to zero?

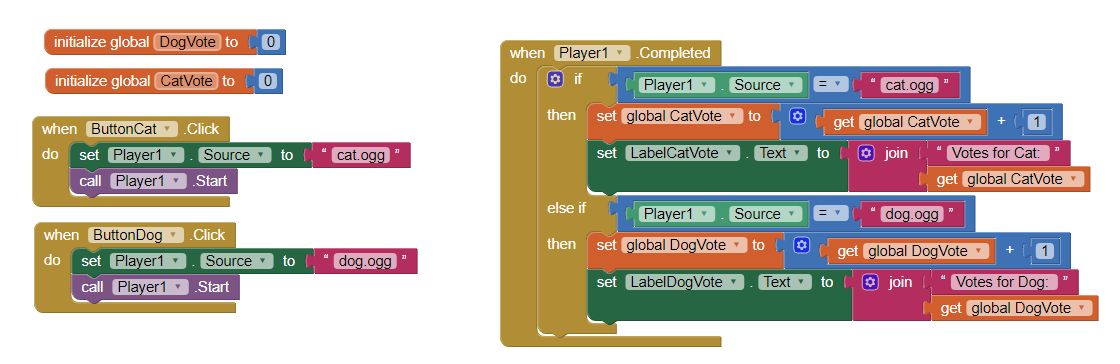
Change the app to be about something more interesting than voting if people like cats or dogs. What else could an app like this be used for?

**Completed App**

*Designer*

****

*Blocks*



*Companion Screenshot*

