MIT Hackday App Inventor 2 Tutorial Notes (Long Version)

Eni Mustafaraj & Lyn Turbak
emustafa@wellesley.edu & fturbak@wellesley.edu

Saturday, January 25, 2014

1. Log in to App Inventor

Step 1.1: In a web browser, go to http://appinventor.mit.edu. Use Chrome, Firefox, or Safari; do **not** use Internet Explorer (which is not yet supported).

Step 1.2: There's lots of tutorials and documentation to explore later on this website. But for now, click on the 

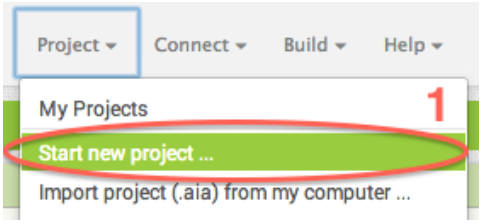
button, which brings you to http://ai2.appinventor.mit.edu/

Step 1.3: Log into App Inventor with your gmail or Google account name and password.

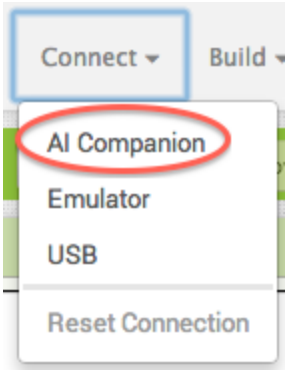
Step 1.4: In the *Welcome to App Inventor* popup, click **Continue.**

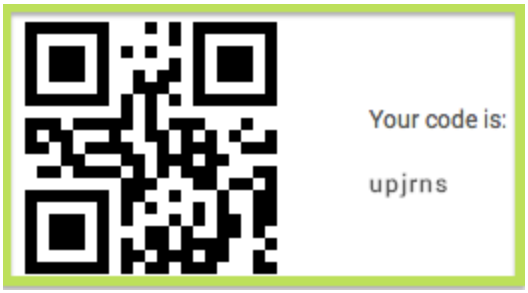
2. The AnimalsType Project

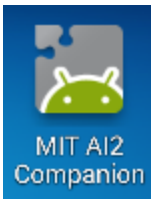
Step 2.1: Start a new project named `AnimalsType`



Step 2.2: Next "connect" the Blocks editor to your Android device by WiFi. From the `Connect` menu select `AI Companion`, which will display a two-dimensional visual QR code for a 6-character code like the one shown below:





On the your Android device, launch the `MIT AI2 Companion` app 

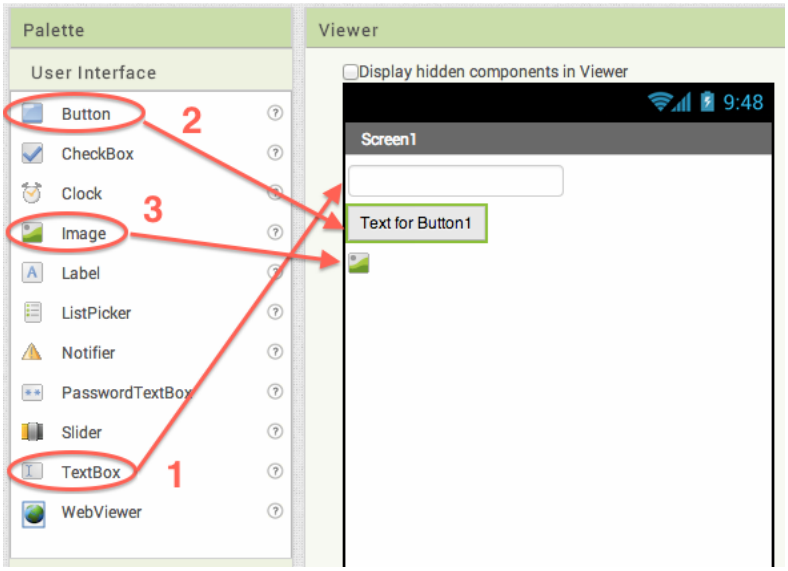
, press 

, and scan the QR code displayed in the browser. (Alternatively, you can type in the 6-character code and press 

). This initiates a **live development** connection between the browser and the Android device. You should now see the app's components on the device, and all changes you make in the Designer and Blocks windows should be reflected on the device.

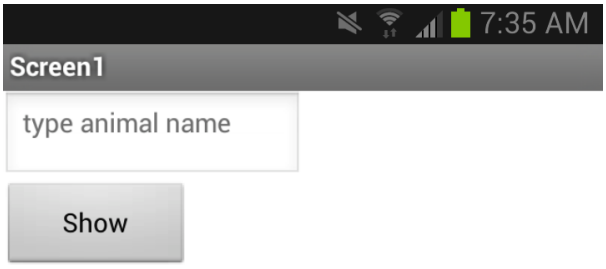
Step 2.3: You're in the 

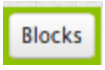
window, where you add components to your app.
Drag a Button, TextBox, and Image from the **Palette** into the **Viewer** to add them to your app.



Step 2.4: In the **Components** pane, select `TextBox1` and in the **Properties** pane, change its Hint property to `type animal name`. Similarly, change the Text property of `Button1` to `Show`.

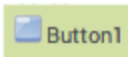
You should now see something like the following on your device:



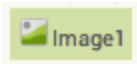
Step 2.5: Click the 

button in the upper right corner to switch from the Designer to the Blocks Editor, where you will specify the behavior for your app.

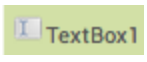
Step 2.6: Populate the Blocks Editor with blocks from the **Blocks** pane

- from the 

drawer, select the event handler block 

- from the 

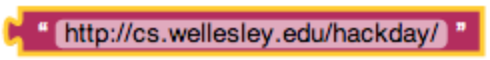
drawer, select the 

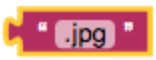
block.
- from the 

drawer, select the 

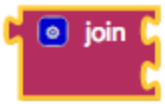
block.
- from the 

drawer, select two copies of the 

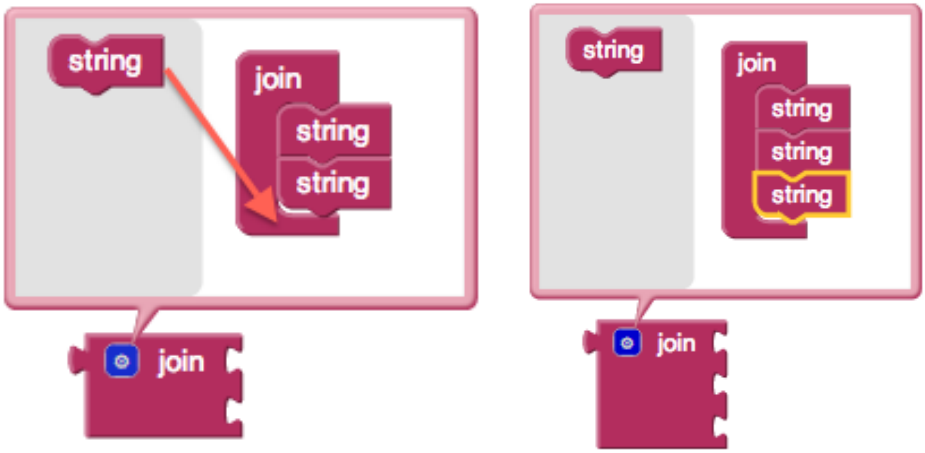
block, and edit one to say 

and the other to say 

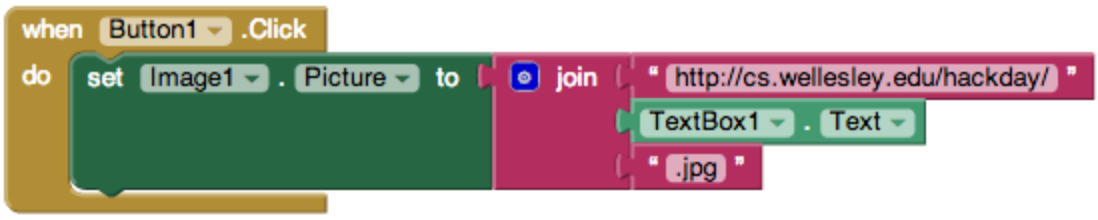
.
- from the 

drawer, also select a 

block. Click on the blue gear *mutator* symbol in its upper left corner, and in the mini-blocks-editor, drag a `string` block into the bottom of the `join` stack in order to add a third socket to the `join` block.



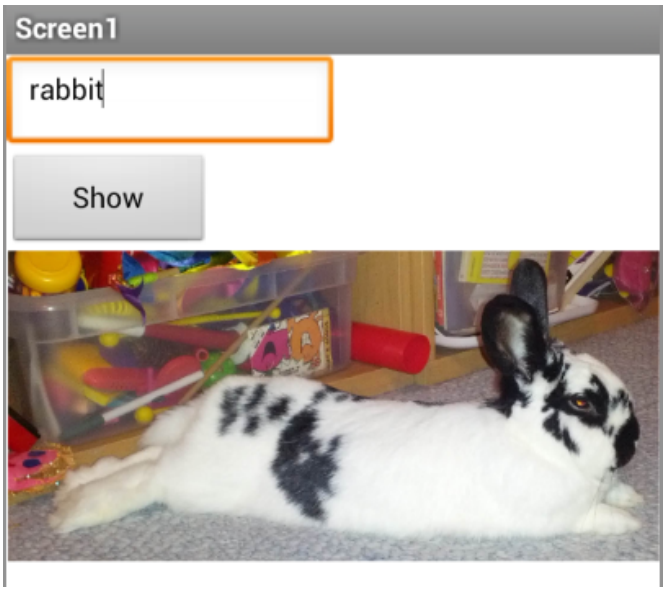
Step 2.7: Connect the blocks from the previous step to form the following assembly:



This is an App Inventor blocks program that says “When `Button1` is clicked, change the image displayed by the `Image1` component on the phone to be the image from the URL

`http://cs.wellesley.edu/hackday/animal.jpg`, where *`animal`* is the string typed into the `TextBox1` component on the phone.

Step 2.8: Test your program by typing one of the animal names `cat`, `dog`, or `rabbit` into the text box and clicking the **Show** button. This should display a picture from the associated URL.



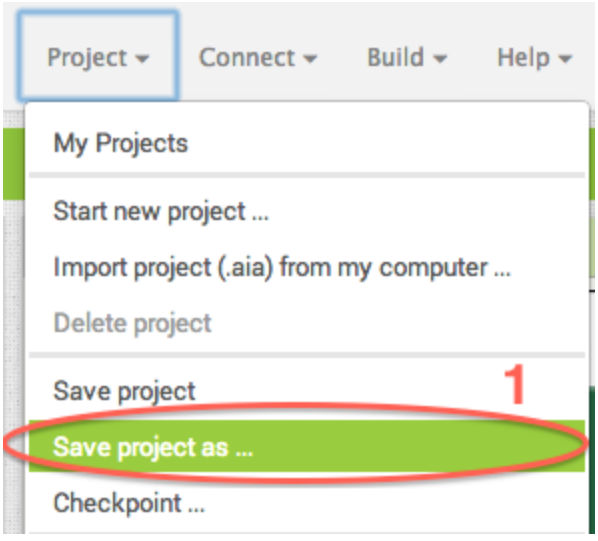
What happens if you misspell the name or type something else. Why?

You have now completed your first App Inventor program!

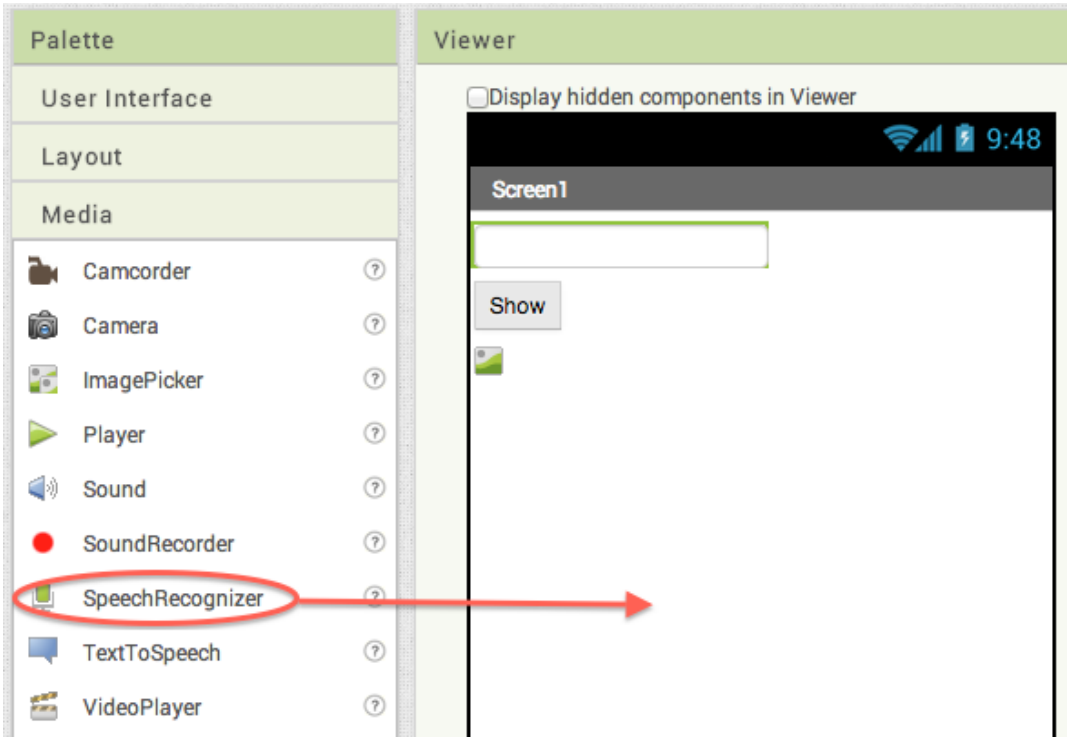
3. The AnimalsSpeak Project

It's tedious to type the name of the animal. We can use App Inventor's speech recognition capabilities to speak the animal name rather than typing it.

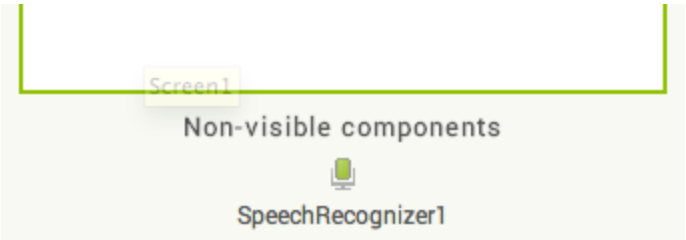
Step 3.1: Use the `Project>Save project as …` option to create a copy of `AnimalsType` named `AnimalsSpeak`:



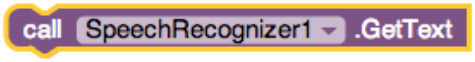
Step 3.2: Click the Designer button to go back to the Designer, and from the Media drawer of the **Palette**, drag a `SpeechRecognizer` component into the **Viewer**.



The `SpeechRecognizer` is a so-called non-visible component that will appear at the bottom of the Viewer:

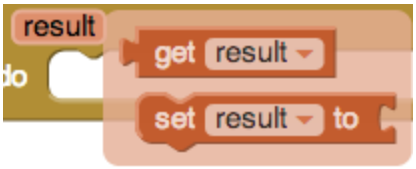


Step 3.3: Click Blocks to go back to the Blocks Editor, and from the SpeechRecognizer1 drawer select the 

and 

blocks.

In the first of these blocks, hover the mouse over result to get a mini-flyout menu

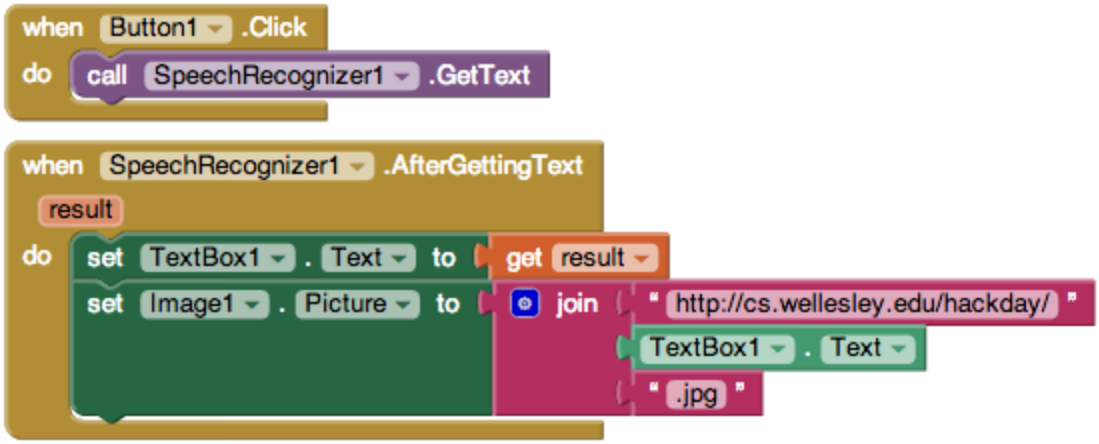


, and select the `get result` block from this menu.

Additionally, from the TextBox1 drawer, select the 

block.

Step 3.4: Reassemble the blocks to create the following program:

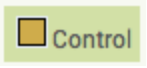


Step 3.5: The new program should still be connected to the Android device. (If not, redo Step 2.2) Test your program by pressing the **Show** button and speaking the name of one of the animals `cat`, `dog`, or `rabbit`. Isn't that easier than typing?

4. The AnimalsSiri Project

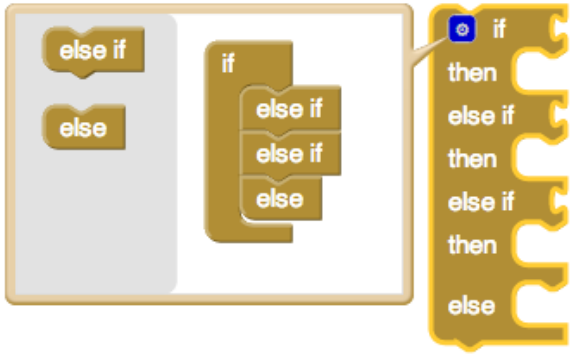
We conclude by making a very simple Siri-like program that can correctly obey commands like "Show me a picture of a dog, please" or answer questions like "What kind of animal goes meow?"

Step 4.1: As in Step 3.1, use the `Project>Save project as …` option, this time to create a copy of `AnimalsSpeak` named `AnimalsSiri`. This project will not have any new components, so you can just stay in the Blocks Editor.

Step 4.2: From the 

drawer, select the 

block, and use the mutator in the upper left corner to add two `else if` clauses and an `else` clause:

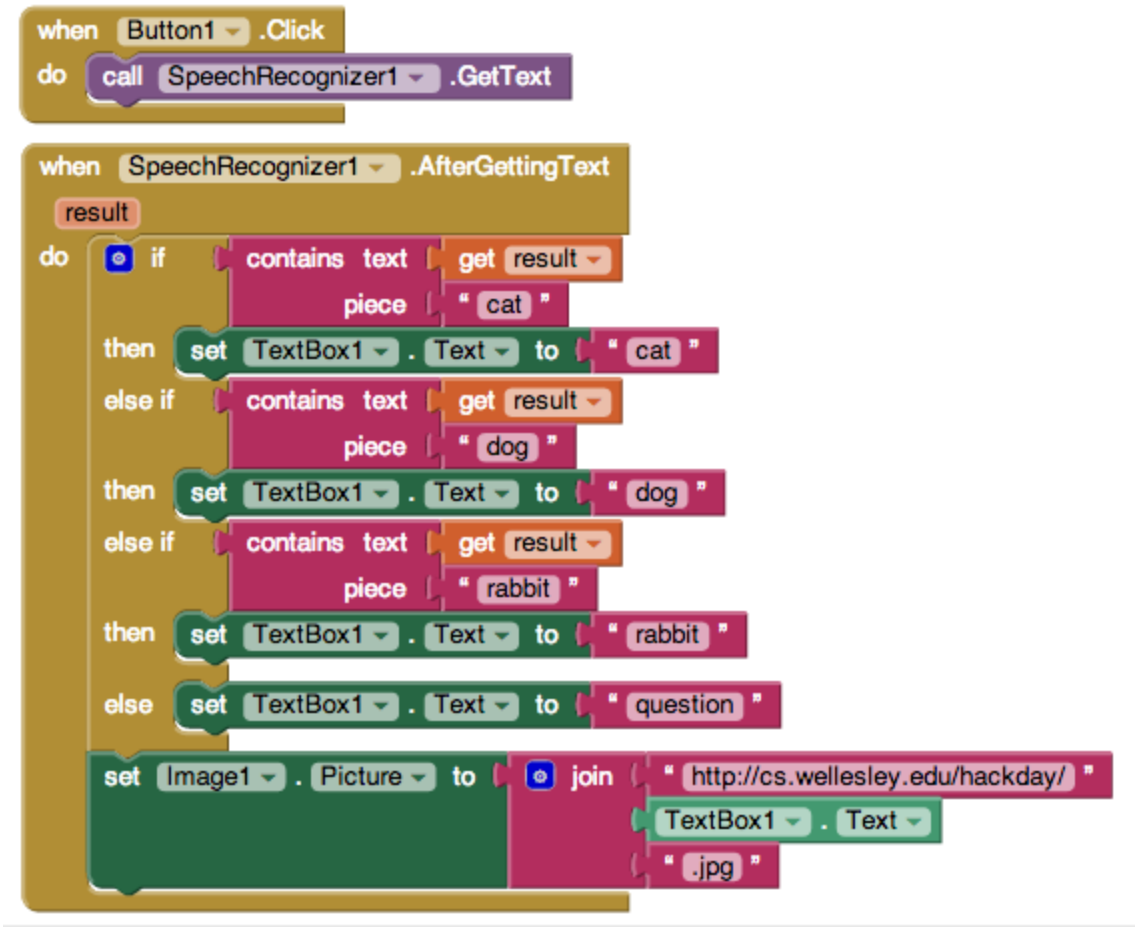


Step 4.3: From the 

drawer, select the 

block.

Step 4.4: Make enough copies of blocks (using Command-C/Command-V on a Mac or Ctrl-C/Ctrl-V on a PC) and reassemble them to create the following program:



Step 4.5: Test the program on the device and explain its behavior. To understand better why it behaves the way it does, it's a good idea to use the Designer to add an extra TextBox or Label to display the `result` of the Speech Recognizer.

Step 4.6: From the 

drawer, select an 

block, and replace the test for `cat` by the following.

How does this change the behavior of the program? Can you use more `or` blocks to modify the program to seem to be more "intelligent"?