BCSWomen
Specialist Group

Android Programming "Family Fun Day" using AppInventor

Table of Contents

This is the BCSWomen AppInventor workshop handout. It goes along with the workshop, and probably won't make sense on its own.

All of the workshop materials will be available online at http://www.hannahdee.eu/appinventor

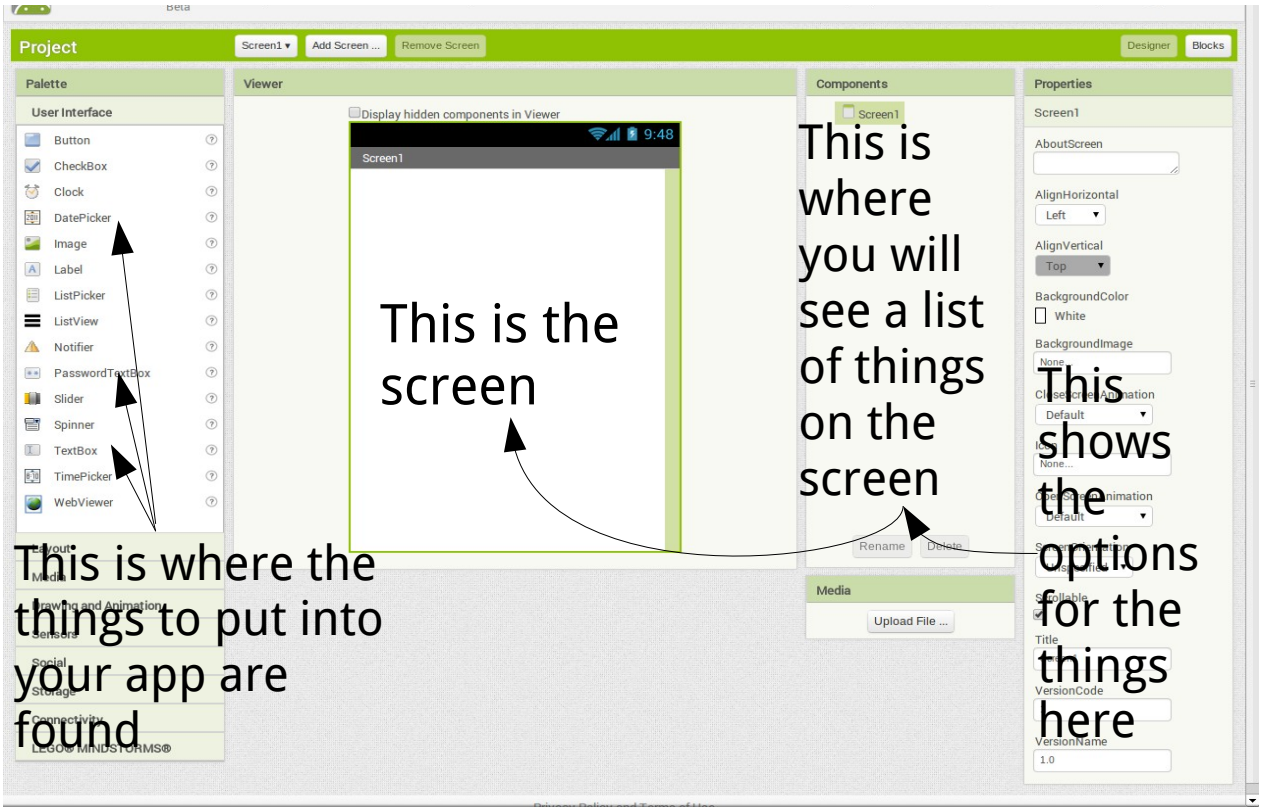
This includes the latest versions of all talk slides, this handout, speaker notes, equipment lists, and the pack of images and sounds used in the workshop.

If you'd like to run this workshop yourself, using exactly the same materials, go ahead, but please let us know.

This is version 2.0 of the handout, written by Hannah Dee and updated for AppInventor 2 by Tilly Horsley.

A step-by-step guide to making a simple app[1]

1. Go to http://appinventor.mit.edu/
2. Click **Create** on the top right corner of the page.
3. Sign into your Google Account.
4. Click on Create New Project in the upper left corner.
5. Enter the project name meow in the dialog box that appears, click OK.
6. You should get something that looks like this:



Basically the screen is in 4 columns.

On the left, things you might want to use

Next over, in column 2, a picture of the phone showing the things you've chosen to put on it. We're going to call this **The Screen.**

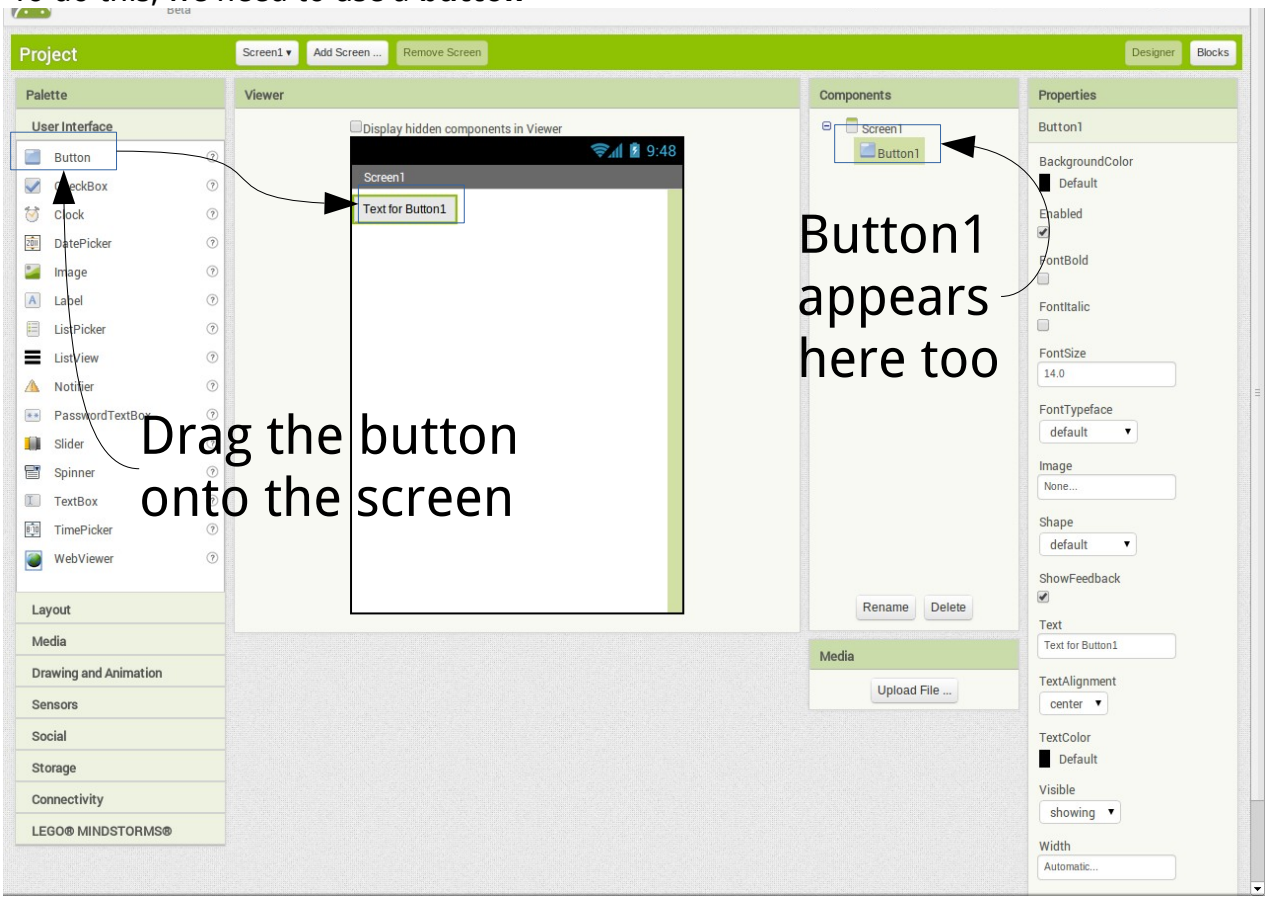
Column 3 will show what goes on behind what you have put on the screen.

Column 4 on the far right has options which are also to do with what goes on behind the screen.

These things all let you work out how the stuff you put on the screen will work and what it will do. Some of the things we're going to use are visible (like pictures and buttons) and some are not visible (like sounds).

1These instructions are based on the first AppInventor project guide, on the MIT pages. You can find the original here:
http://beta.appinventor.mit.edu/learn/setup/hellopurr/hellopurrphonepart1.html
It has fewer pictures, and a couple of extra instructions to do with renaming your objects.

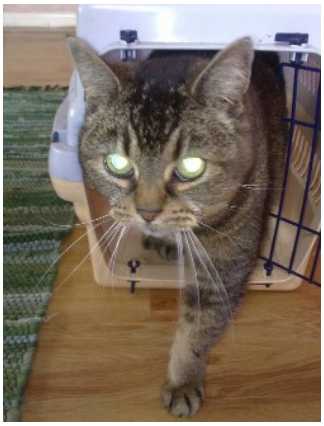
What we want our app to do is this:
Show a **picture** that we can **tap**, and it will make a **sound**
To do this, we need to use a **button**



Now we have a button on the screen we need to make it more interesting.
We're going to do this by adding a picture of a cat.
You have the choice of 2 cats: Kitty (the original, I have no idea whose cat that is) or KatyCat (my cat, who is lovely but stupid, and has laser eyes). Both of these pictures are in the folder you should have got from a usb stick earlier.

Kitty

<----

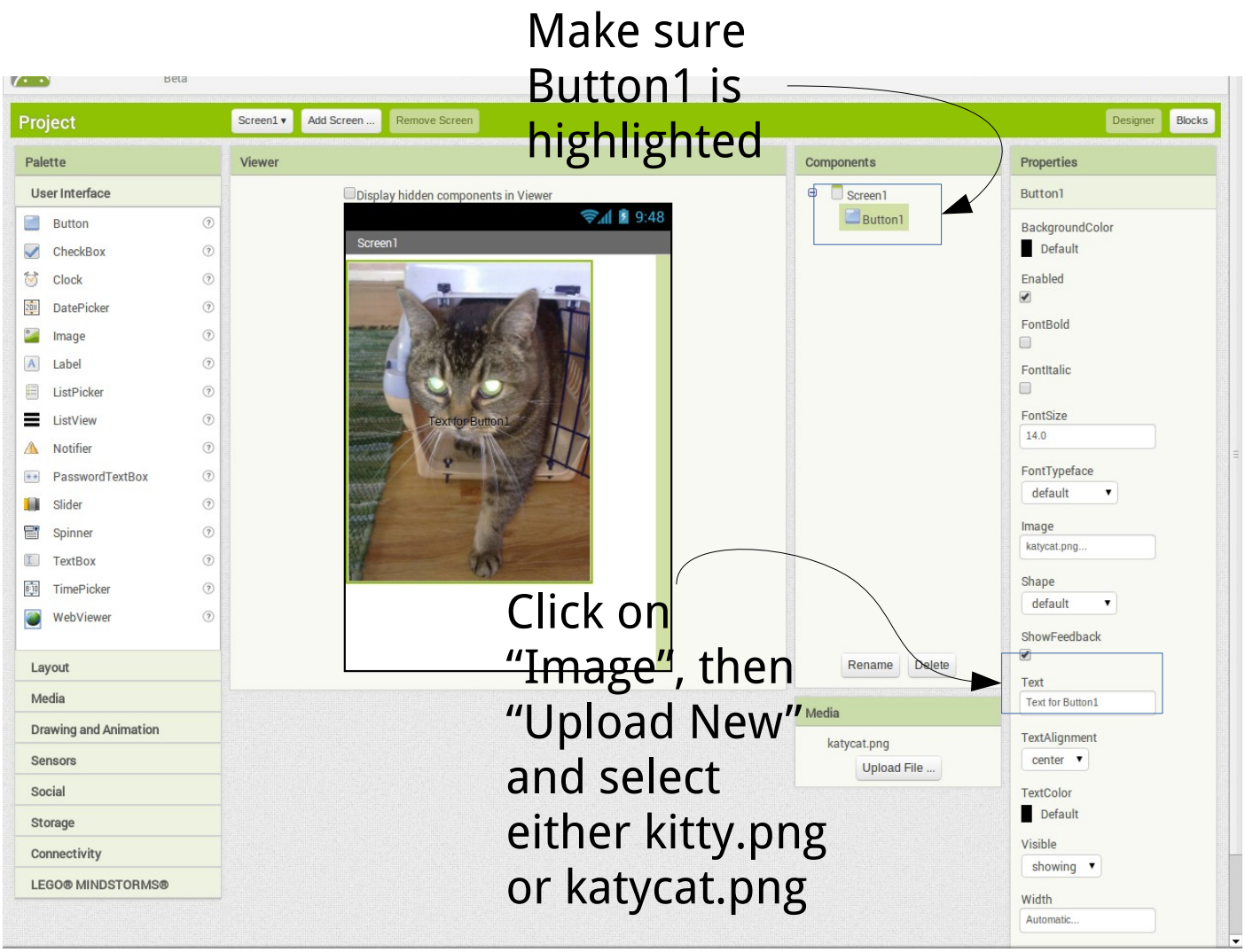


KatyCat

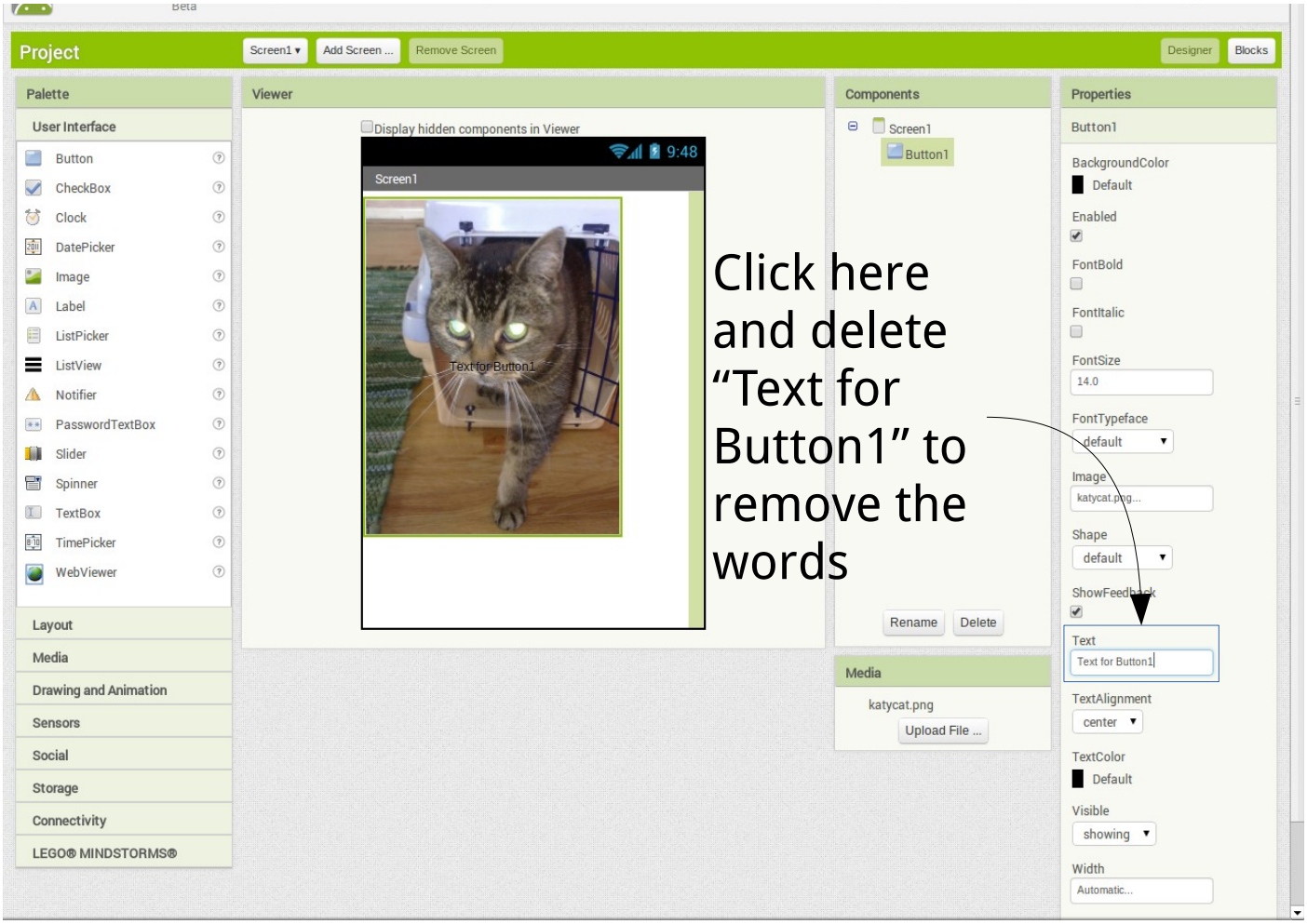
---->

To get the picture on the screen...
First make sure button1 is highlighted (under “Components”).
It should be highlighted already, but if not, don't worry, just click on it once.

Then under Properties on the far right, click on “Image”, then browse and choose the image file you want to use.



You will spot that there is some ugly text on the cat's face. We can fix that.



Now you should have a nice button with a picture of the cat on it.

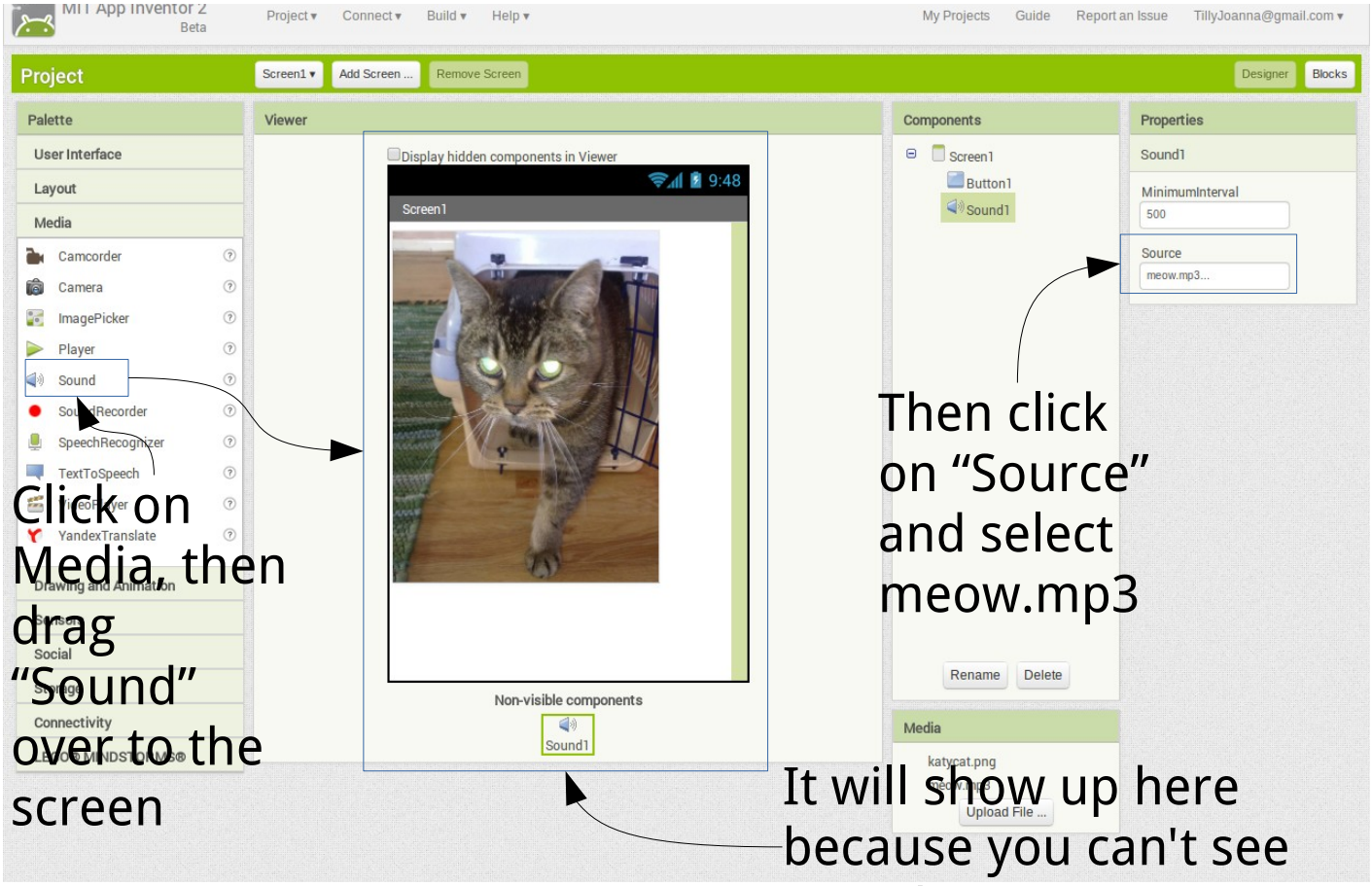
But what next? Well, we need to give the button something to do.

What we are going to do is add a meow sound.

Click on Media at the left, then Sound, and drag it over to the screen. You can just drag it and drop it on top of the cat picture.

It won't appear on top of the cat picture -you can't see sounds, so they show up below the screen as "non visible components".

Once you have the sound added, click on "Source" at the far right, and then "Upload New", and you will find meow.mp3 in the same place as the cat pictures.



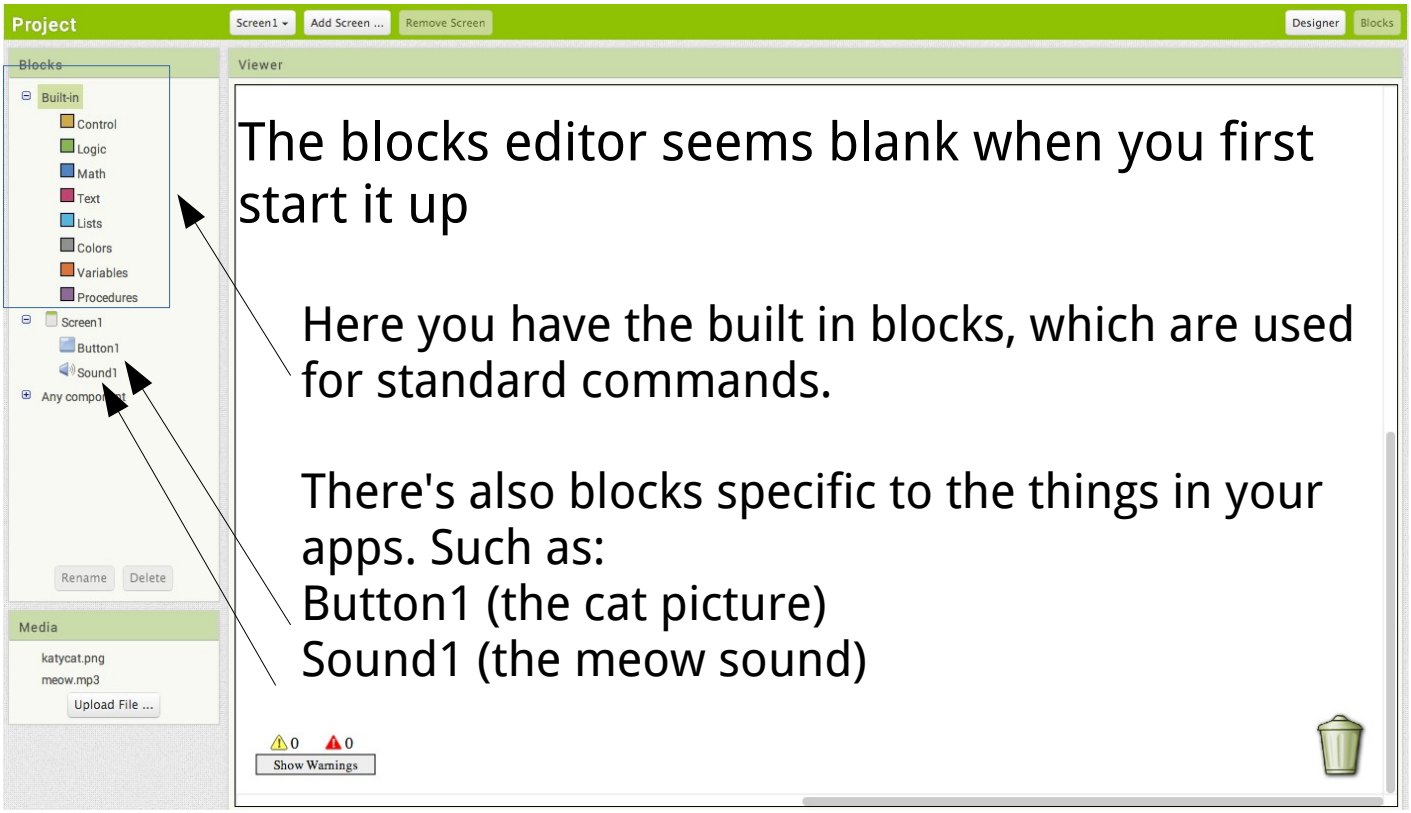
Now we have a picture of a cat

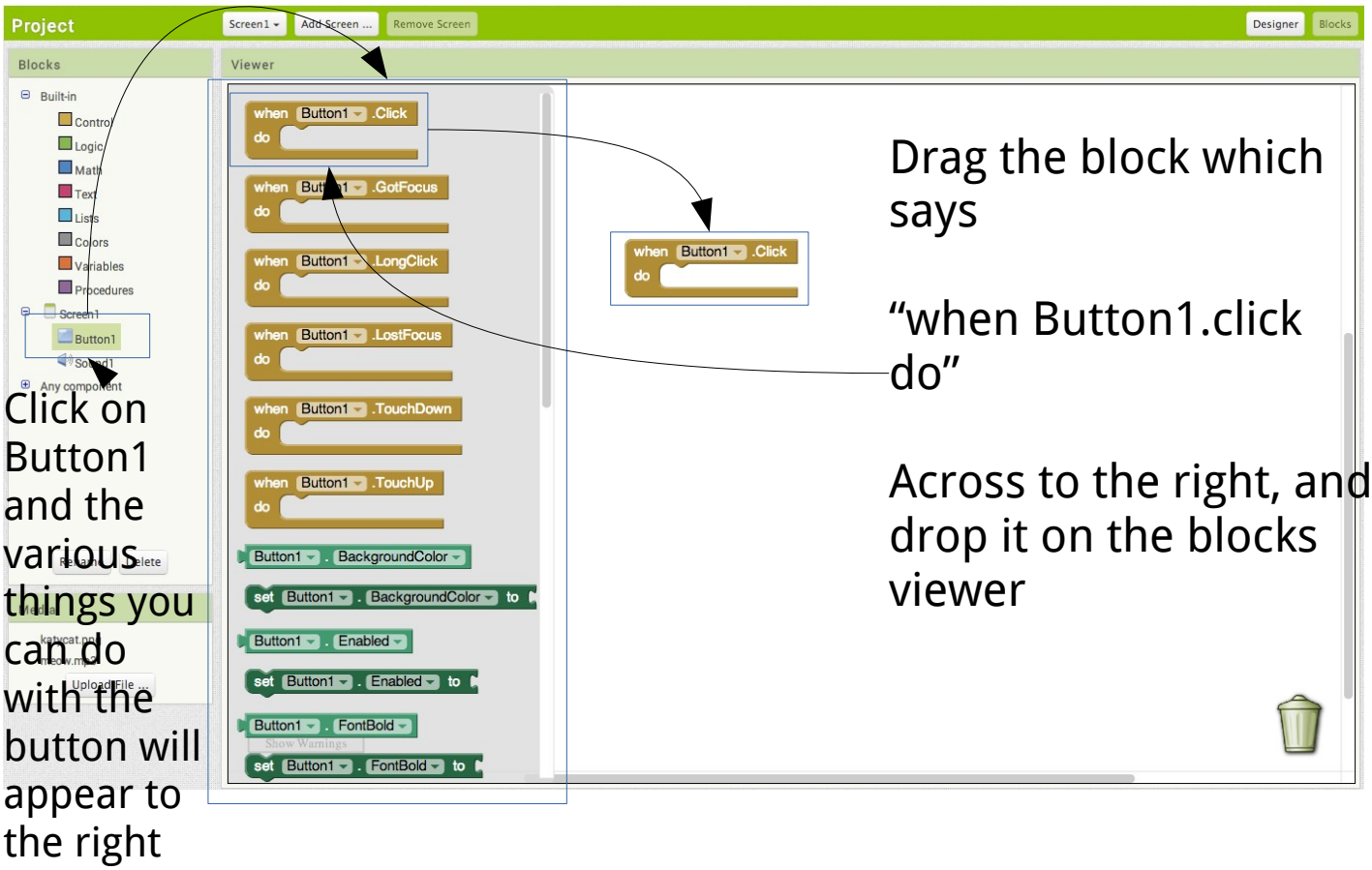
And we have a meow sound

What we need now is to put together **A PROGRAM** which makes the meow sound play when we tap the cat.

To write programs in AppInventor, we need to start the Blocks Editor.
Click on "Blocks", which is at the top of the window on the right.

You'll probably have to wait a little bit. Once it starts up it will look a bit like this:


Project
Screen1
Add Screen ...
Remove Screen
Designer
Blocks
Blocks
Built-in
Control
Logic
Math
Text
Lists
Colors
Variables
Procedures
Screen1
Button1
Sound1
Any component
Rename
Delete
Media
katycat.png
meow.mp3
Upload File ...
Viewer
The blocks editor seems blank when you first start it up
Here you have the built in blocks, which are used for standard commands.
There's also blocks specific to the things in your apps. Such as:
Button1 (the cat picture)
Sound1 (the meow sound)
0
0
Show Warnings



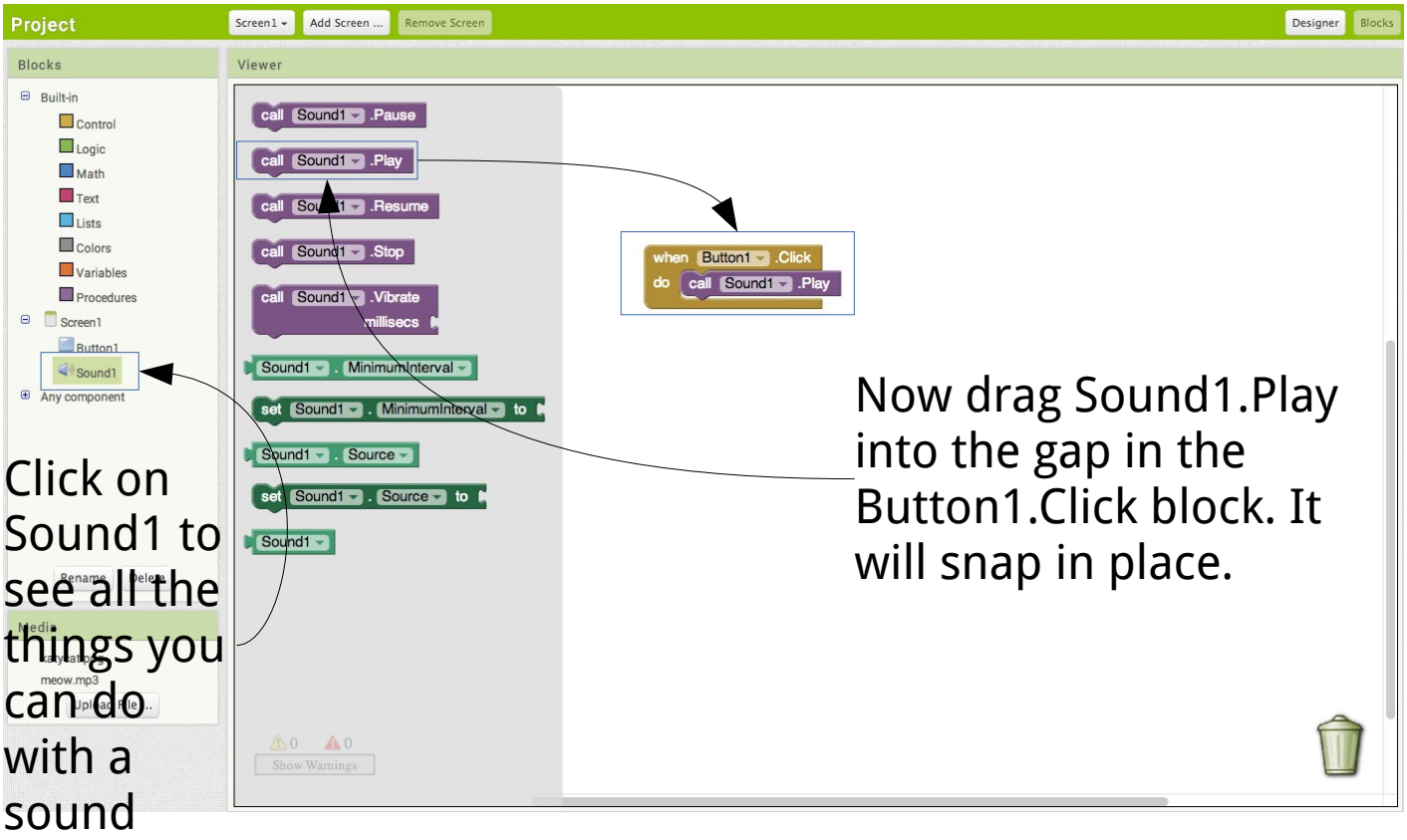
What this block does is that it detects when the button has been clicked (remember, the button looks like a picture of a cat in our system!).

But it does not say what to do when the button is clicked. That's the last stage...

Click on Sound1 on the left, which will show you all the things you can do with sounds.

Drag Sound1.Play across to the space in Button1.Click and drop it there.

You don't have to be super accurate here, the Blocks editor will work out what is going on and put it in the right place. And if it doesn't you can always drag it around a bit to fix it.



Now you have your app put together

- the display half (which decides where things go to on the screen)
- and the blocks half (which decides what happens to those things)

What's left now is getting that onto your android phone, or running it on the emulator.

Getting your app running on the emulator

The emulator is a way of running your app on an android “phone”, without actually having an android device (or without connecting your device) to the computer.

The emulator doesn't always work as well as you might hope, but when it does work it is really useful.

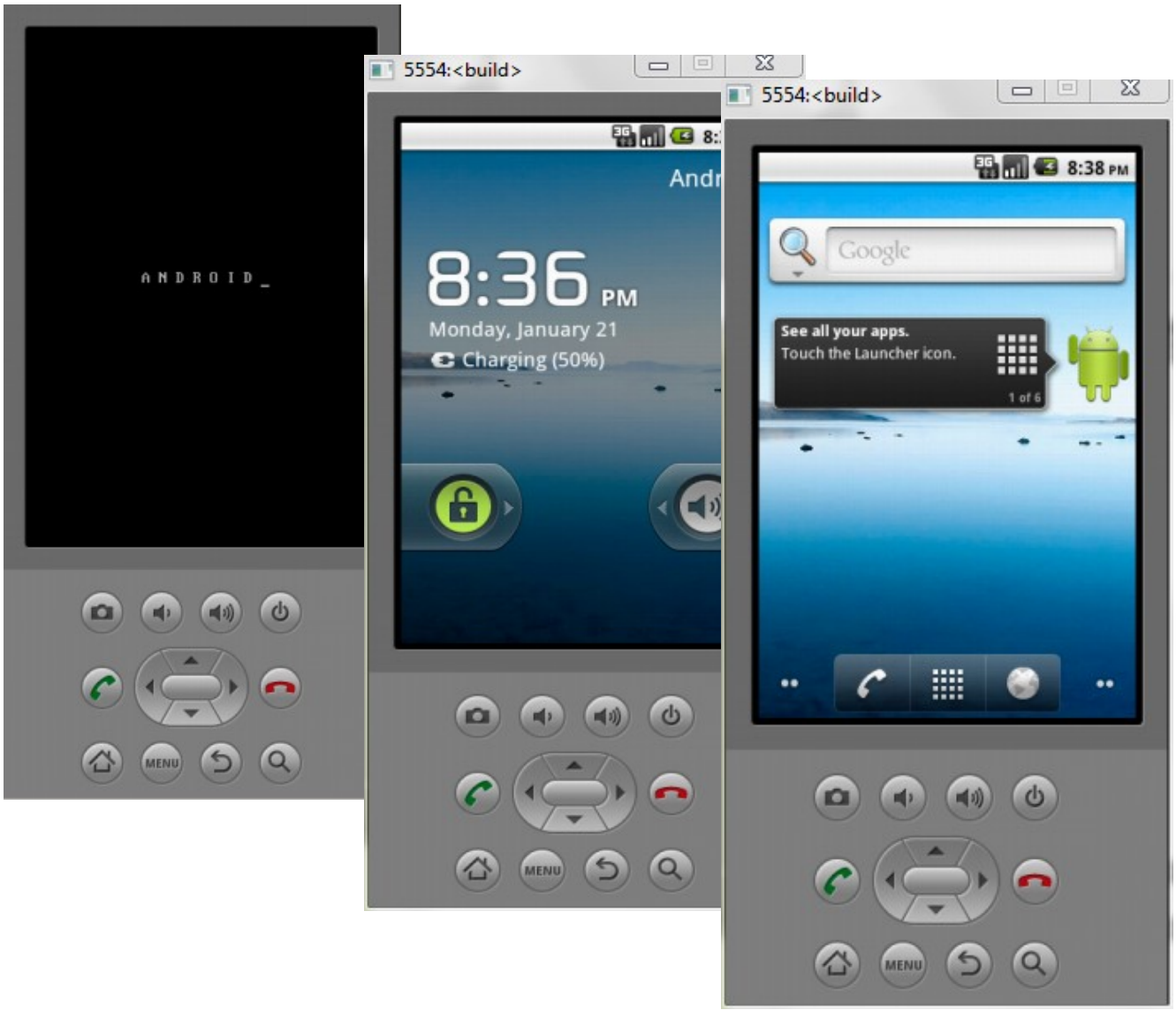
To give it a go, first download AIStarter which you can get from http://appinventor.mit.edu/explore/ai2/setup. There are instructions there for Windows, Mac and Linux

Once the emulator is ready, it will look like the middle picture below. You then need to slide the unlock icon to the right, using the mouse. At this point the emulator will look like the right-hand picture below.

Finally you need to connect to the emulator from the Blocks Editor – click on “Connect To Device” at the top right, and you should see your emulator there.

This is the step where the technology is most likely to go wrong, in our experience. If you can't get it to connect, try again (from the starting the emulator phase).

Sometimes it just doesn't work, though. Oh well, that's computers for you.



Getting your app onto your phone or tablet

If you have an Android phone or tablet, there are a few steps you will have to go through to get your app to run on it.

NOTE! The person who is logged into the phone has to be the person running AppInventor – both the computer and the phone need to be using the same Google account.

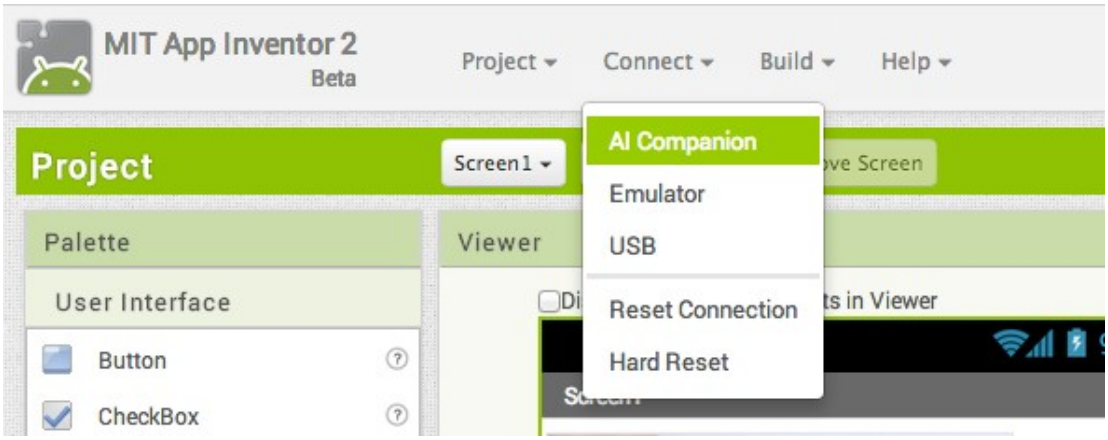
There are a couple of ways to get apps onto a phone, and at the end of this handout we cover the others. But ...

First – and easiest – is to use the AppInventor Helper App which is called MIT AICompanion. You can download this from Google Play, it's a small app which will let you see your apps on the phone using wifi.

This will open a barcode and a 6 letter code. You can either scan the barcode using the camera, or type the code into the box. For the barcode you'll need to click on “Connect using barcode” on your device, once you've scanned the barcode the app will open automatically. For the code you should type in the code first and then click “Connect using code”

It may take a while for the app to fully load, so don't panic when you don't see your complete app. The MIT AI Companion app can be a bit buggy too, it somethings not working quite right check your blocks (especially the errors and warnings in the bottom left corner of the blocks editor!) and if it's still not working completely close the app on your device, then click on “Reset Connection” on the website, and try again.

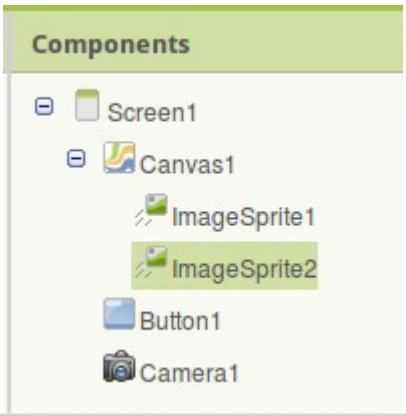
This wont save the app to your device, we'll discuss this at the end.



Some sample apps for you to investigate...

The aim of this section is to give you a quick look at the blocks behind some sample apps. You might be able to work out how to recreate these apps from this but if you can't that's not a problem, they are there for ideas and if you get stuck just ask one of the helpers.

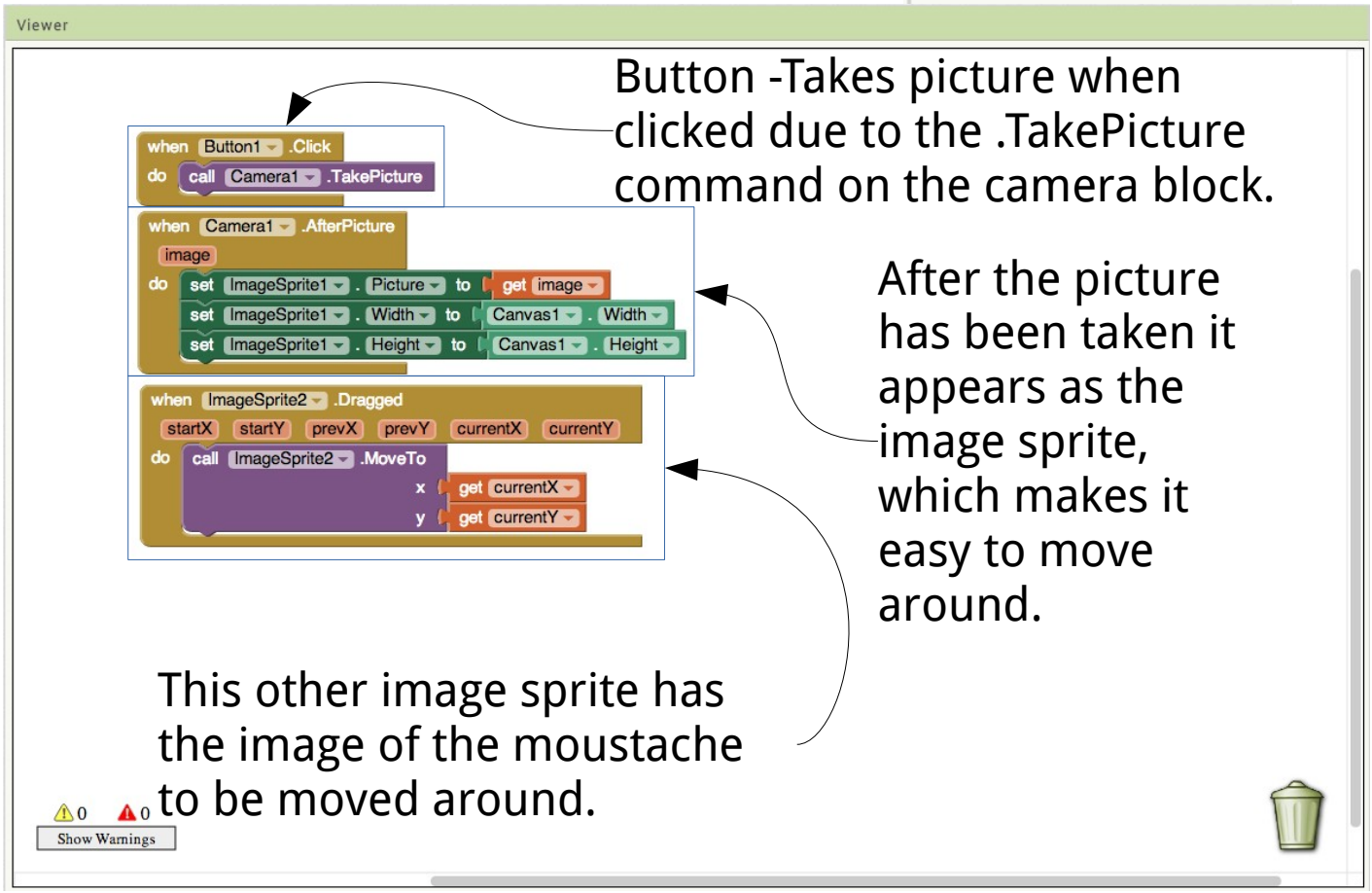
An app which uses the camera: moustache man



This app has a button, which when pressed makes the phone take a picture.

We do this by putting two ImageSprite components on a Canvas, and then when the camera has taken the picture we set the new picture to be on ImageSprite1. ImageSprite2 is a picture of a moustache, and we use another block to make it so that you can drag the moustache up and down the screen.

Here are the blocks for that...



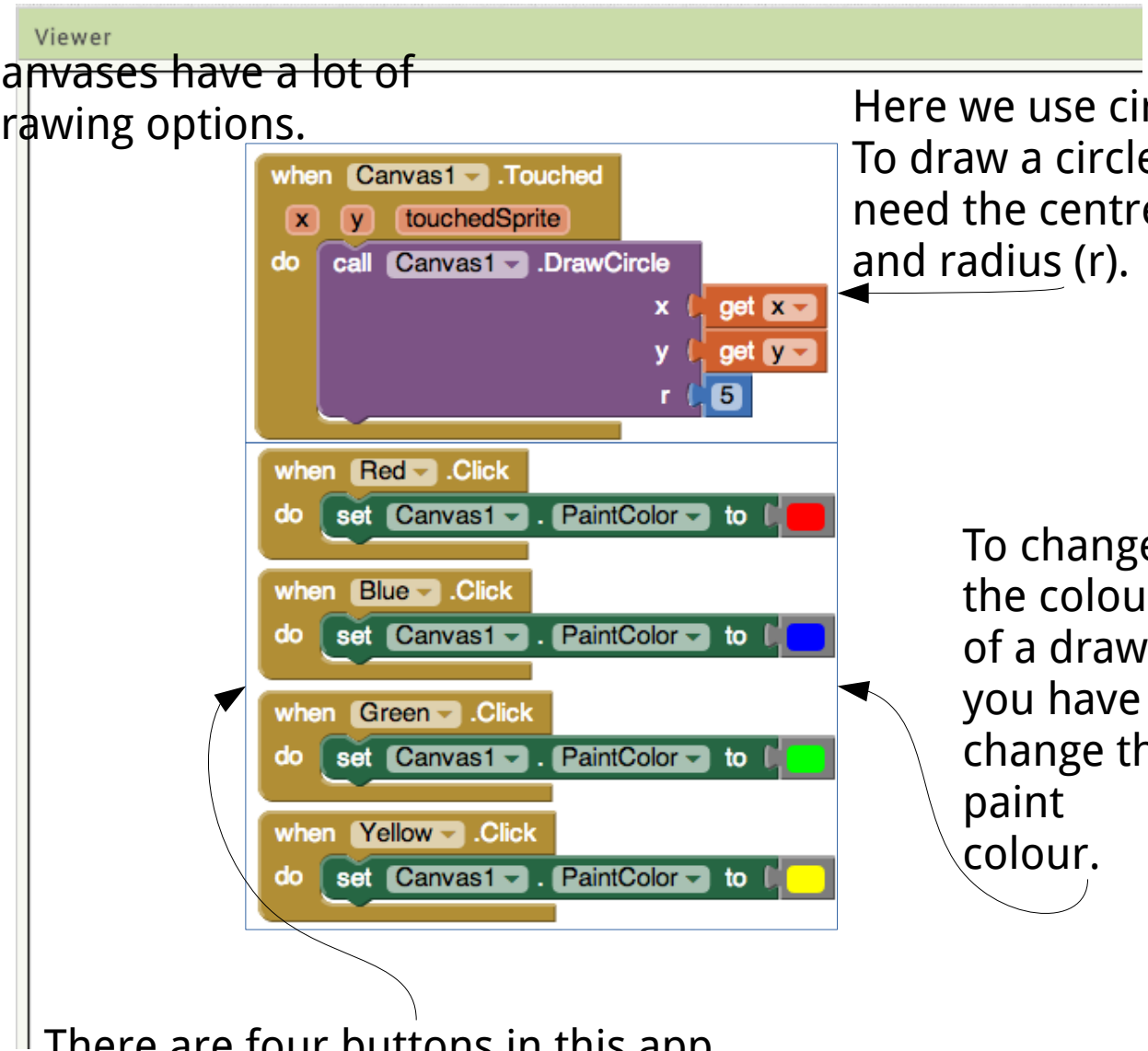
A simple drawing app that lets you put dots on a canvas

To do drawings using AppInventor you need to create a **canvas**

Canvases are like blank drawing surfaces, but they're not just blank drawing surfaces. Each canvas has a colour – they start off drawing in black but you can change the colour using the blocks editor.

To see the different colours available, look at the Colors option which you'll find at the bottom of Built-Ins on the Blocks Editor.

One other thing we have done with this app is to give the buttons sensible names. You can rename any of your components, using the main AppInventor window. In the components panel select the thing you want to rename (in this case I have renamed the buttons), and then click on Rename at the bottom.



An app which has a timer, and uses a variable: counting up

This app is actually really boring but it uses some key things.

The first really useful thing it has is a **clock** – in AppInventor, you can use clocks to control all sorts of things. A clock has a timer, which can be switched on or off, and then the timer will set things going every second, or minute, or any time interval you like.

In computer programming sometimes we need to save things so that we can use them again and to do this we use a **variable**. This is the second really useful thing. To set up a variable, you click on Built-In at the top left of the screen, and then you can set one up. Once you've set up a variable, you can get to it through My Blocks, and under My Definitions you will see a couple of ways to change variables. In this program we're calling the variable mycounter.

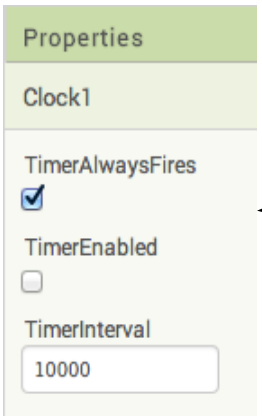
You could modify this program to make a phone Meow every 15 seconds, for example.

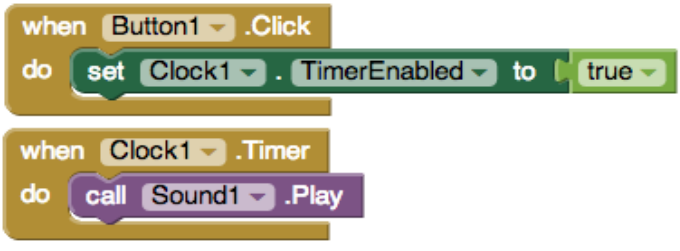
You wouldn't need the variable mycounter, you would just need to ...

a) Set the timer on the clock to 15 seconds (you can do this in the main AppInventor window), and

b) Change the thing inside the When Clock1.Timer block to be an instruction to “Go Meow!” rather than to print out a count and change a variable.

In properties for the clock on the design page, you should set time interval to 10000 (10 seconds) and untick the TimerEnabled box.

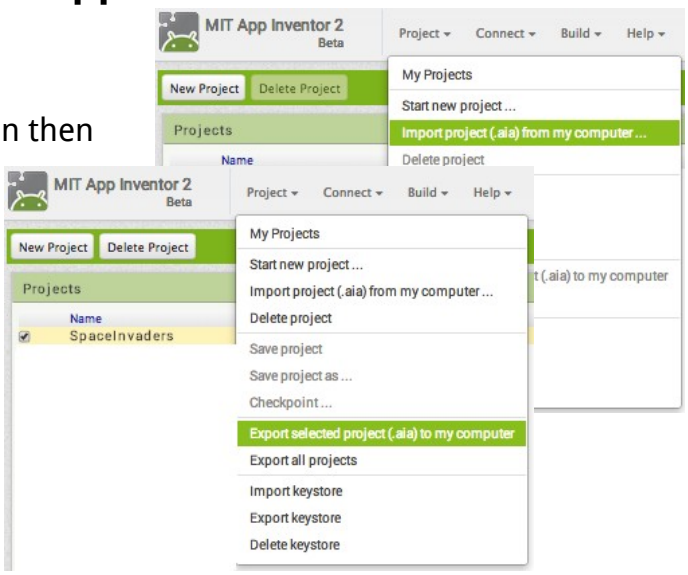




The blocks for this app are relatively simple, you should set them up like this.

How to download and share your app and code

Sharing code:

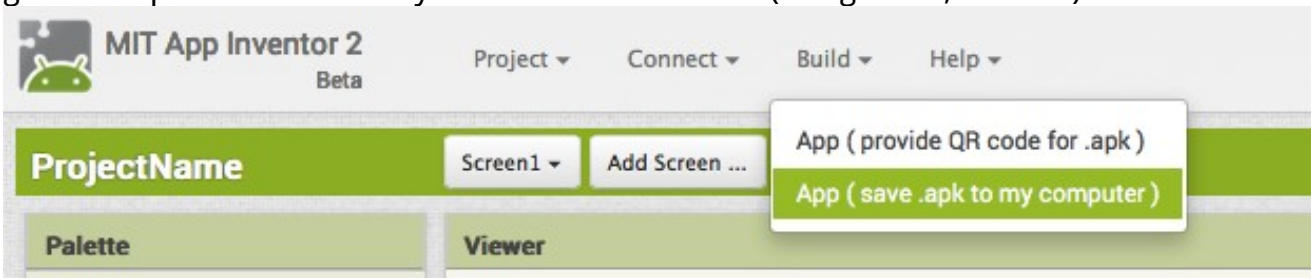


You can share your code with friends who can then alter or “remix” it. To do this, select the project you want to share in the list of your projects, then click “Project” and then “Export selected project (.aia) to my computer”.

You can then give this to your friends (share by email, or usb, or whatever). Your friends can then import the project to their AppInventor (Import Project, under the Project menu).

Sharing your App:

Want friends to download your app to play? Open the project you want to share with friends and click on “Build” here you can select “App (save .apk to my computer)” and then you can give the .apk file to whoever you want to download it (using email, usb etc.).



This can be downloaded by accessing the email and downloading it straight onto their phone or tablet, but first they should change some settings in their device. By default Android phones won't install stuff from anywhere, you have to enable this by turning off a security setting. To find this setting on older versions of Android (before 4.0) go to "Settings > Applications" and then check the box next to "Unknown Sources". For devices running newer versions of Android (4.0 or above) go to "Settings > Security" or "Settings > Security & Screen Lock" and then check the box next to "Unknown Sources" and confirm your choice.

Downloading your App:

If you want to download your own app onto your own device (and you're not interested in emailing and so on) there is a simpler way. You should change the settings to enable unknown sources (like we've just described above) and then get a QR code reader from the Play store if you don't already have one.

Open the QR reader, and then go on the app you want to download in appinventor, click on build, then “App (provide QR code for .apk)”. This will allow you to download it onto your phone without having to email it to yourself.

Uploading your app to the Play Store

If you want to upload your app to the Play store for everyone to download, it can be quite a long process, but it's certainly possible. There's a lot of reading involved and choices to make. You should get a parent's permission, as it will cost a bit of money to register, and you may also need some help!

First you need to register for a Google publisher account, this costs $25 (around £15), and you will need to put in all of your details (name, age, date of birth, and address). You may have to get one of your parents to register for you. You should then download the .apk file as discussed in sharing your app.

You can upload this app by going onto the Google Play developer console and clicking on “All Applications” on the left of the page. In the centre of the page theres a button saying “Add new application”, click this and follow the instructions to add the app carefully

You'll also have to set up a store description, including the name of your app, description and screen shots. Icons and art are also recommended so that the listing in Google Play makes sense. It's recommended that you have

- Screenshots for phones
- Screenshots for 7-inch tablets
- Screenshots for 10-inch tablets
- A high resolution icon (512x512 PNG file)
- A promo graphic (180x120 PNG file)

You should also have tested your app on as many different devices as you can.

It's a good idea to allow about an hour for the process of your first Google Play upload – they ask a lot of questions.

Here's a link to more information online:
http://beta.appinventor.mit.edu/learn/reference/other/appstoplay.html

Links, hints and tips for taking this further

If you have enjoyed this workshop and want to have a go at programming some more then that's great. Here are some links to get you started...

Scratch

Scratch is a programming language designed to help people learn – you can build games in it, and it's really similar to AppInventor. It doesn't work on mobile phones (yet!) but it's real fun and you can use it on pretty much any computer. I think Scratch is best for people who want to play with programming, and who are keen to give it a go, and it's useful for humans aged 6+

http://scratch.mit.edu/

Greenfoot

Greenfoot is a programming environment that's useful for learning Java. Android apps that are written by hand (rather than by AppInventor or similar) are written in Java, so learning it could be a useful thing.

Greenfoot is probably most useful for people aged 12+ - it involves much more typing than Scratch

http://www.greenfoot.org/door

If you want to build Android apps you can get all of the stuff for free. You'll need Java and some other tools (things that will let you connect to phones, if you have them, and things that will let you run simulators of those that you don't have). If you're interested in doing this, I'd recommend learning a bit of Java, getting the book “Hello Android”, and taking a good look at this website : http://developer.android.com/training/basics/firstapp/index.html

It's not easy, but it's not rocket science.

If you want to make iPhone apps there are a couple of tools, but pretty much everything you can use will require you actually get an iPhone, and possibly also a Mac. As far as I know there aren't any free systems for building iPhone apps. The language you'd need to learn to write iPhone apps is called C# (pronounced “See-Sharp”).

Copyright, licensing and all that stuff

The current version will be kept at http://www.hannahdee.eu/appinventor along with all other materials. If you've any suggestions for improvements, let me know on hmd@hannahdee.eu and I'll incorporate them in future versions

This is version 2.0 of the AppInventor Handout, written by Hannah Dee and updated for AppInventor 2 by Tilly Horsley