

Made by the people who make the **dummies** books!

BUILDING A *Mobile App*

Design
and
Program
Your Own App!

Sarah Guthals, Ph.D.
Social Entrepreneur and Engineer



INTRODUCTION

GET STARTED WITH MOBILE APPS

SO YOU WANT TO MAKE YOUR VERY OWN MOBILE APPS FOR ANDROID? That is an awesome idea!

Making mobile apps can be a lot of fun. You can make simple apps that provide people with information, fun apps that allow people to play with images, sounds, and drawings, and even make games! The world of mobile apps is huge, limited only by your imagination. This book guides you through the process of making mobile apps with software built by MIT called App Inventor; it's free, and people of all ages can use it! With just a few clicks and drags, you can build a mobile app — and learn how to program, all at the same time!

ABOUT APP INVENTOR

App Inventor is a website that makes building mobile apps easy and fun. There are two main parts to App Inventor, the design and the blocks. The design part is where you design what your app looks like; where the buttons go, what color it is, and add any pictures or things like drawing canvases and sounds. The blocks part is where you write real code to make your app do things; for example, you have to tell your app to play a dog barking sound when someone clicks on the dog button! Even though the blocks part is real coding, App Inventor makes it easy by putting the code into “code blocks” that you can click and drag to make your mobile app. This makes the entire process easy and fun!

ABOUT THIS BOOK

Building Mobile Apps: Design and Program Your Own App! has four projects that guide you through the process of building four different

mobile apps. First, you make a simple app that plays sound when you push a button. Then you move on to making a profile app all about you. In the third project, you add features to your profile app so that people can draw on your favorite pictures, making them even funnier. Finally, you make an actual mobile game called “Feed Winston.”

Also, if you’re reading this as an ebook, you can tap any web addresses that appear in the text, like www.dummies.com, to visit those websites.

ABOUT YOU

Everybody has to start somewhere, right? I had to start by assuming that you’re comfortable doing this stuff:

- » **Typing on a computer, using a mouse, and using a mobile device.** A Windows system or a Mac; either one will do. This book shows examples on a Mac and an Android tablet.
- » **Following Instructions.** Coding is very precise. It’s very important that you know how to follow instructions carefully, and that you compare pictures in the book to what you make on your own.

ABOUT THE ICONS

As you read through the projects in this book, you’ll see a few icons. The icons point out different things:



Watch out! This icon comes with important information that may save you from trouble the pro coders sometimes have.



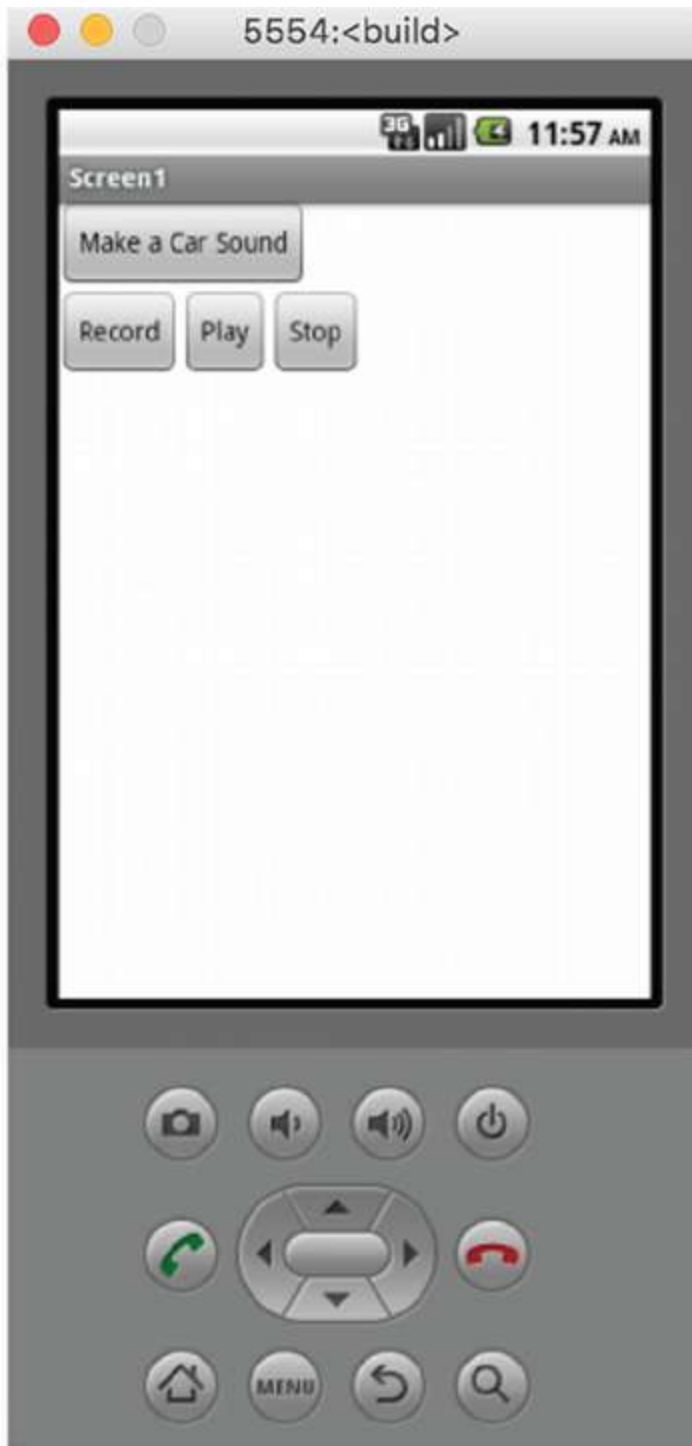
The Remember icon comes with ideas that you should keep in mind.



The Tip icon marks advice and shortcuts that can make coding easier.

PROJECT 1

BASIC MOBILE APP TOOLS



IN THIS PROJECT, YOU LEARN HOW TO SIGN UP FOR APP INVENTOR AND MAKE YOUR FIRST MOBILE APP. You also learn

all the basic tools to use when you are making apps: tools like buttons, text, and multiple screens.

THE SOFTWARE: APP INVENTOR

You can make mobile apps in a lot of ways. In this book, you make mobile apps for Android phones.



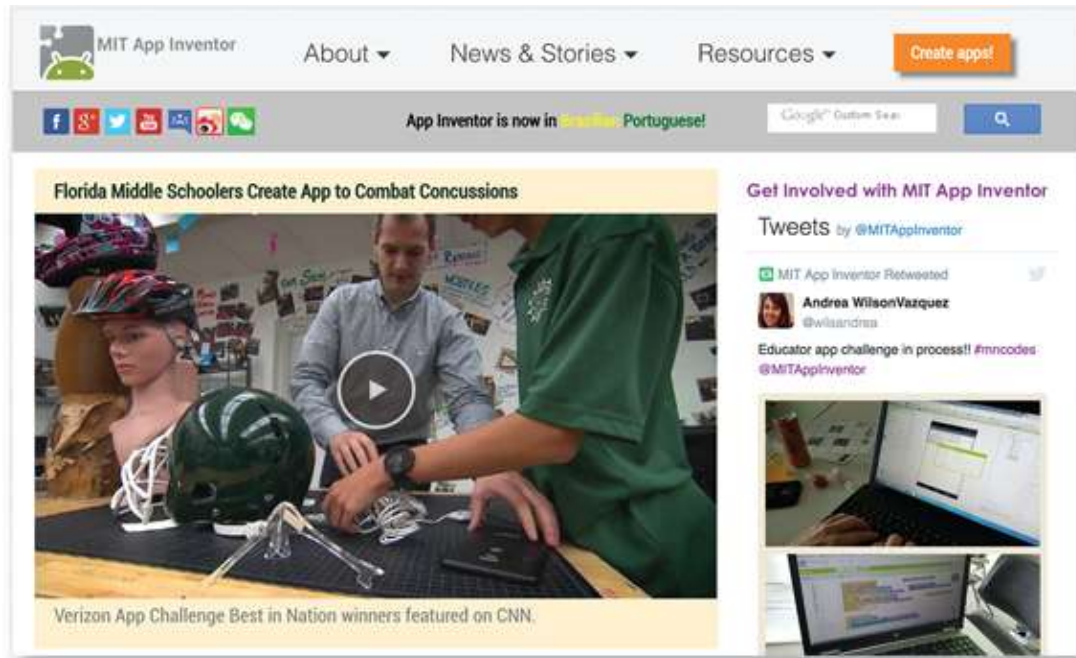
To make Android phones, Google wrote some code called an operating system. The neat thing is that this Android operating system runs on a lot of different kinds of phones, like Nexus, Samsung Galaxy, and Google Pixel. The apps you build in this book can all be installed on Android phones.

To make Android mobile apps, you can use a lot of different software and coding languages. You will use App Inventor, which was made by MIT (Massachusetts Institute of Technology). App Inventor is a block language, which makes it easier and faster to build Android mobile apps!

GET STARTED WITH APP INVENTOR

To get started with App Inventor, ask your parents to join you at the computer and follow these steps:

1. **Go to appinventor.mit.edu.**



2. **Click the Create Apps button.**
3. **Sign in with your Google account, or ask your parents to sign in with their Google account.**
4. **Choose Allow to let App Inventory use your/your parents' Google account.**
5. **Read through the terms of service with your parents and choose "I agree to the Terms of Service."**
6. **Decide whether you want to take a survey.**

You do not have to take the survey to build mobile apps.
7. **Choose an option to set up your Android device or Android emulator.**

This book's examples use the Android emulator.

This book's examples use the Android emulator.

Continue with setup - Choose the connection instructions you were following:



8. Follow the instructions for setting up the Android emulator.

You can find the instructions here:

<http://appinventor.mit.edu/explore/ai2/setup-emulator>.



Always ask your parents before installing any software or accessing any website.

9. After you have installed the App Inventor software, click the Setup Emulator button.

Then follow the instructions for setting up your emulator.



You might need help from your parent to install the emulator. You may need administrator access to your computer, meaning that your parent will probably have to enter her password.

10. Go back to App Inventor and click Continue.

You can choose to not show this popup again by checking the small box next to the Continue button.

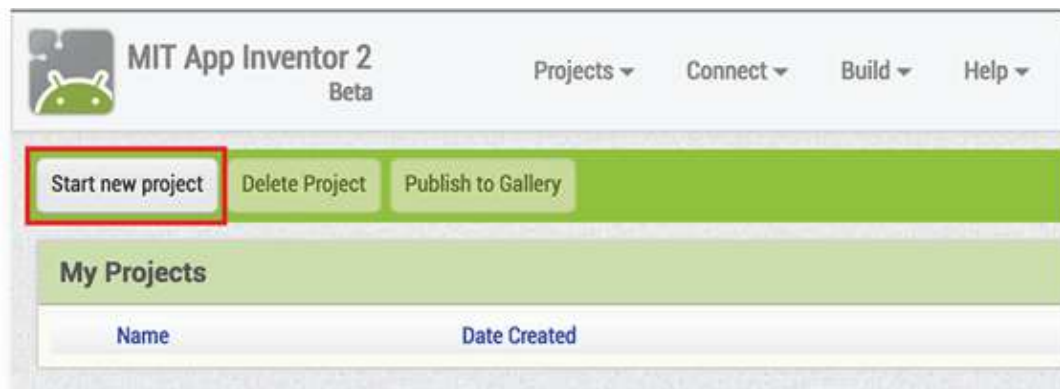
11. Congrats! You are now signed up for App Inventor!



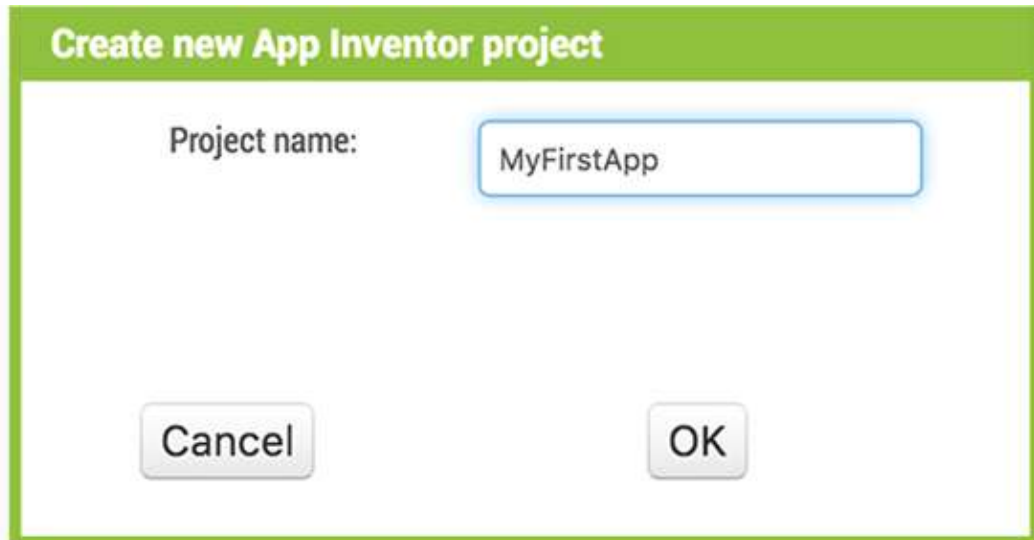
START YOUR FIRST ANDROID MOBILE APP

Now that you have the emulator installed, you can start making your first Android mobile app.

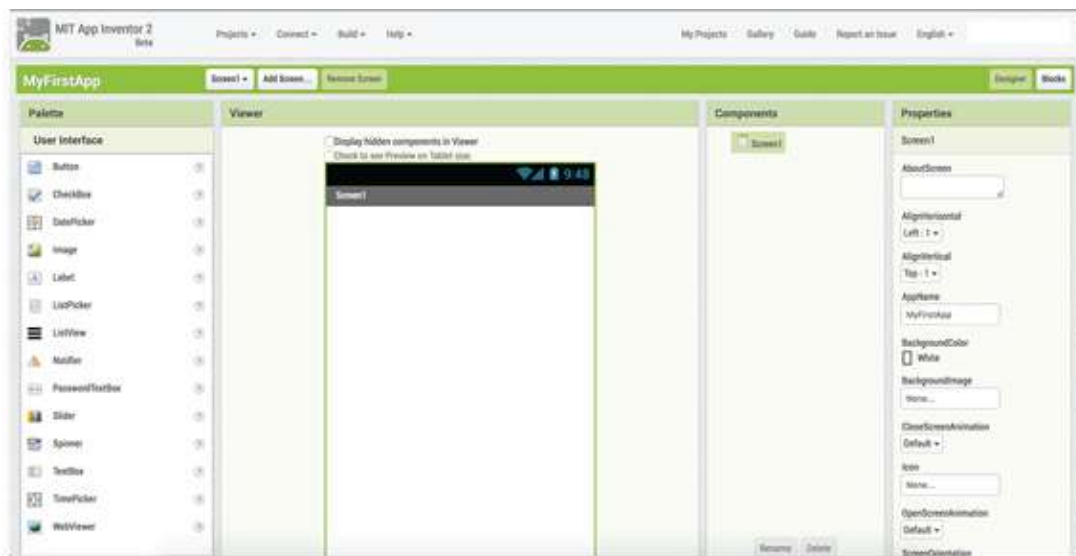
1. Go to ai2.appinventor.mit.edu.
2. Click the Start New Project button in the top left of your window.



3. Give your project a name, like "MyFirstApp," then click OK.



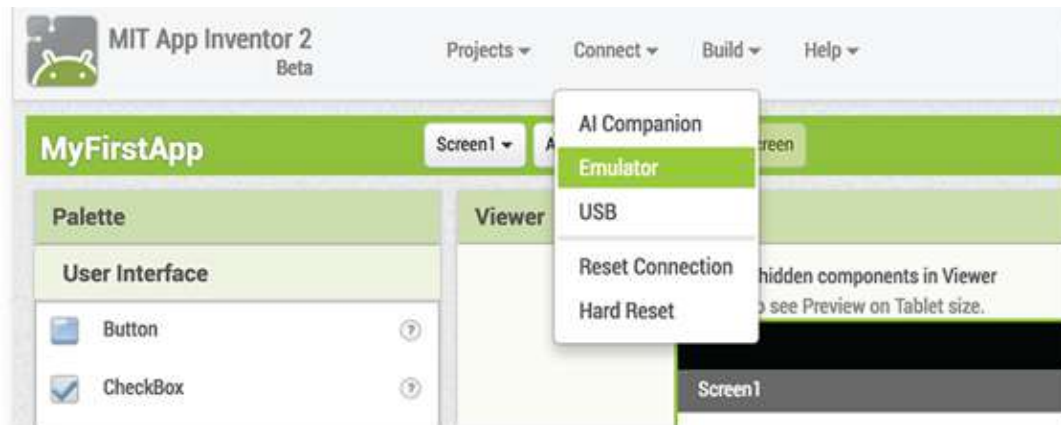
You should now see the app development screen.



CONNECT YOUR ANDROID EMULATOR

Before you start making your Android mobile app, make sure you can connect your emulator.

1. **Click open the Connect menu and choose Emulator.**



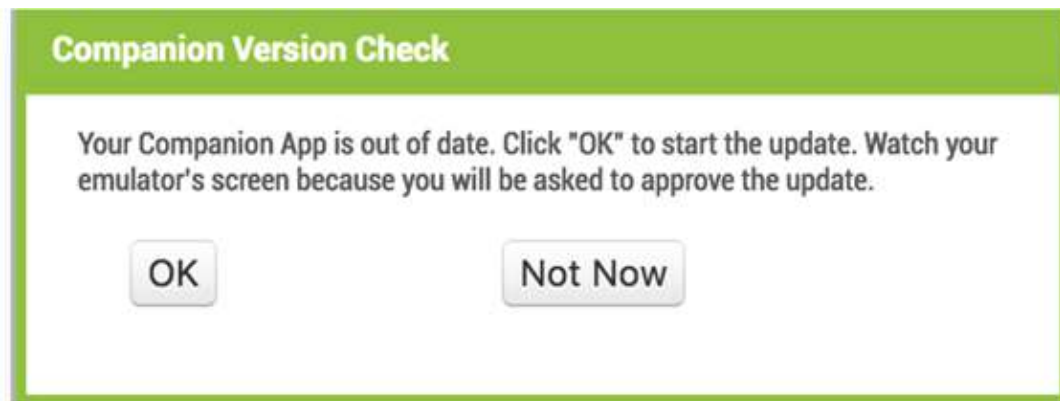


A popup lets you know that the emulator is starting, and a virtual phone starts loading on your screen.

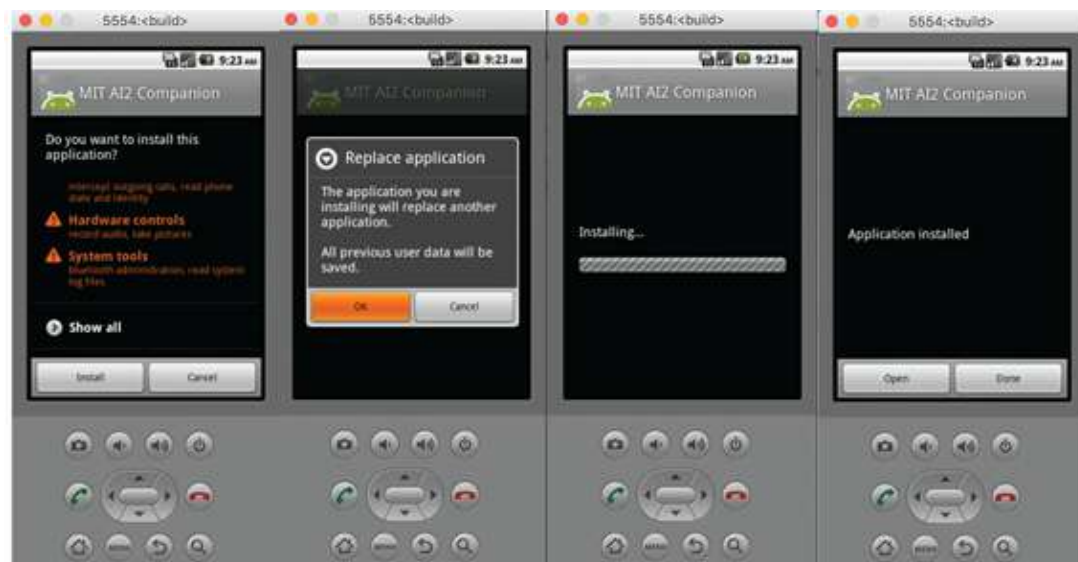


Do not push any buttons on your virtual phone until the home screen appears.

2. Back on your Internet browser, App Inventor asks if you want to update your application on your virtual phone. Click OK.



3. Authorize the app to install on your virtual phone.



Once the app is updated, you can open it by clicking on the MIT AI2 Companion app. It won't display anything yet, though, because you haven't built the app yet.

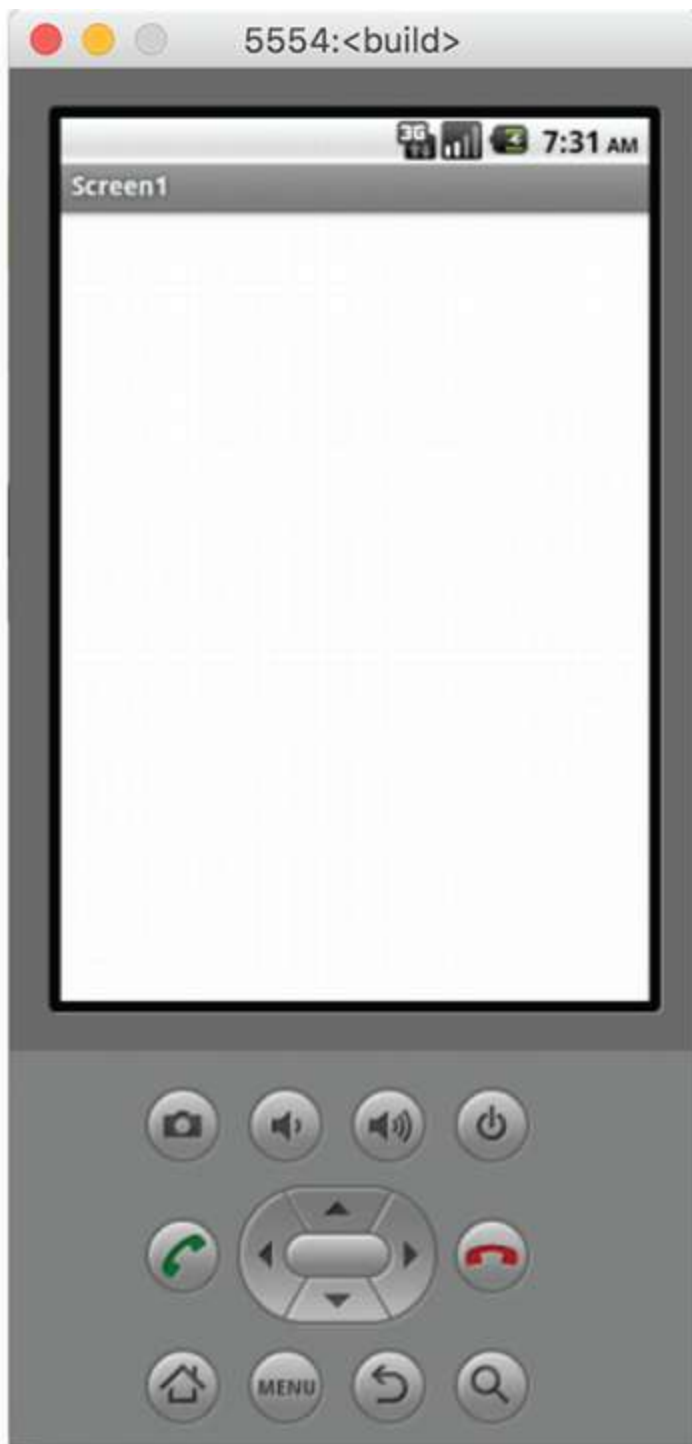


If you ever get lost on your virtual phone, click the home button, then click on the MIT AI2 Companion app, and your app will open.

Congratulations! You have all of your software set up! Now you can start coding your first Android mobile app!

CODE ON APP INVENTOR

If you haven't already, be sure to go back to the beginning of this project to set up your App Inventor coding environment and Android emulator (the virtual phone on your computer).

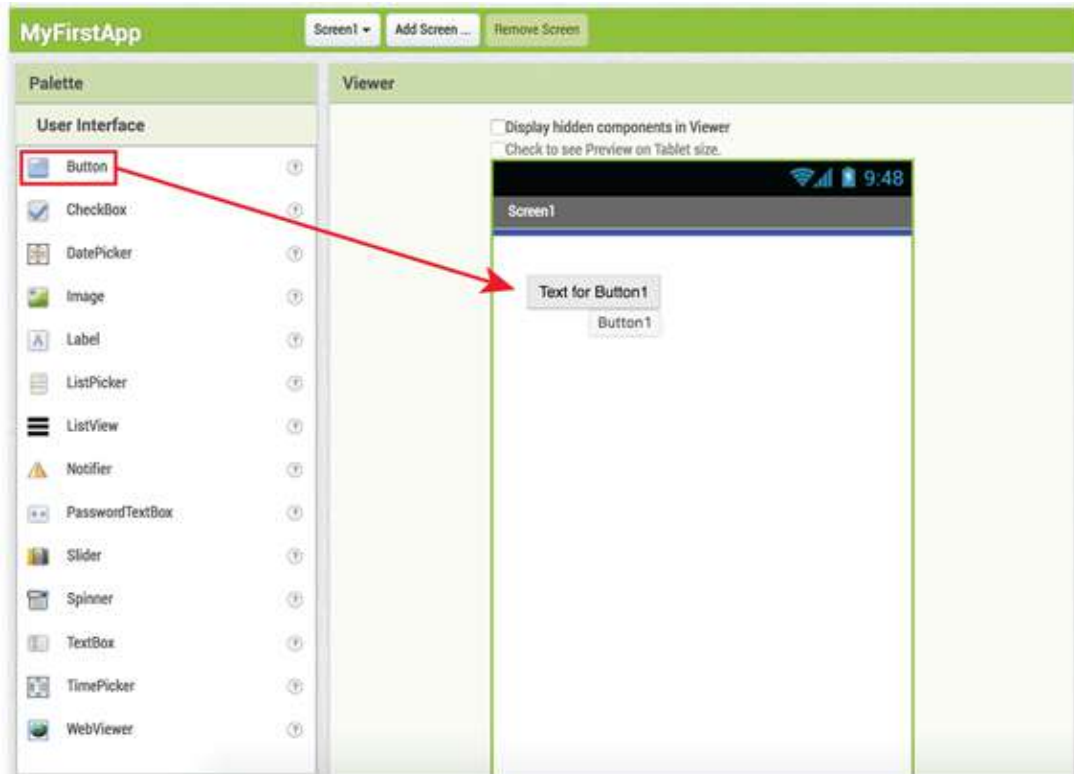


Now you have your environment set up. You should have an Android emulator open like this:

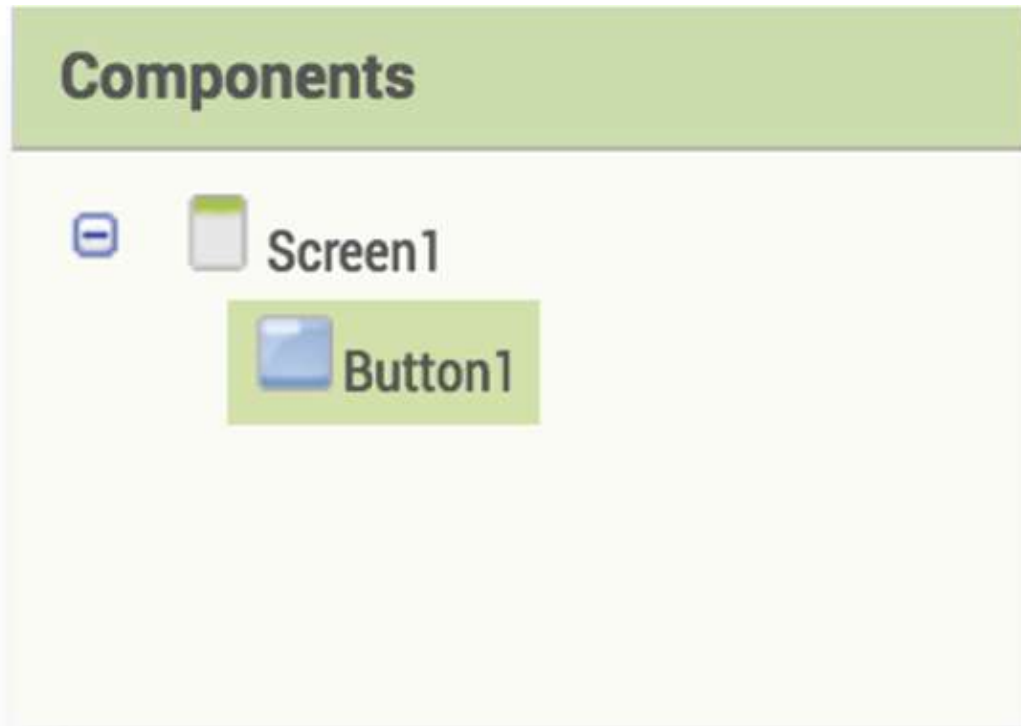
You are now going to add different components to your app.

ADD A BUTTON THAT MAKES A SOUND

1. From the User Interface section in the Palette column, click and drag Button to your app viewer.



2. In the Components column, make sure Button 1 is selected.



3. In the Properties column, change the text for Button 1 to “Make Sound.”

Properties

Button1

BackgroundColor
☐ Default

Enabled
☒

FontBold
☐

FontItalic
☐

FontSize
14.0

FontTypeface
default ▾

Height
Automatic...

Width
Automatic...

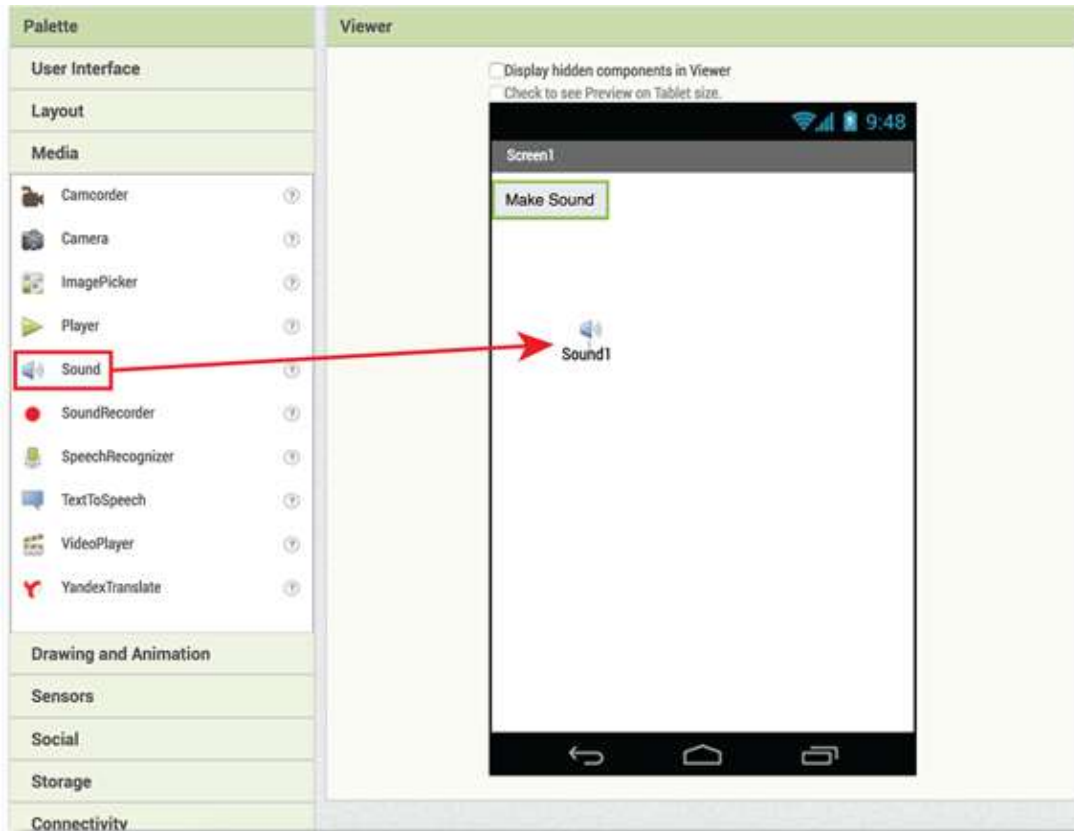
Image
None...

Shape
default ▾

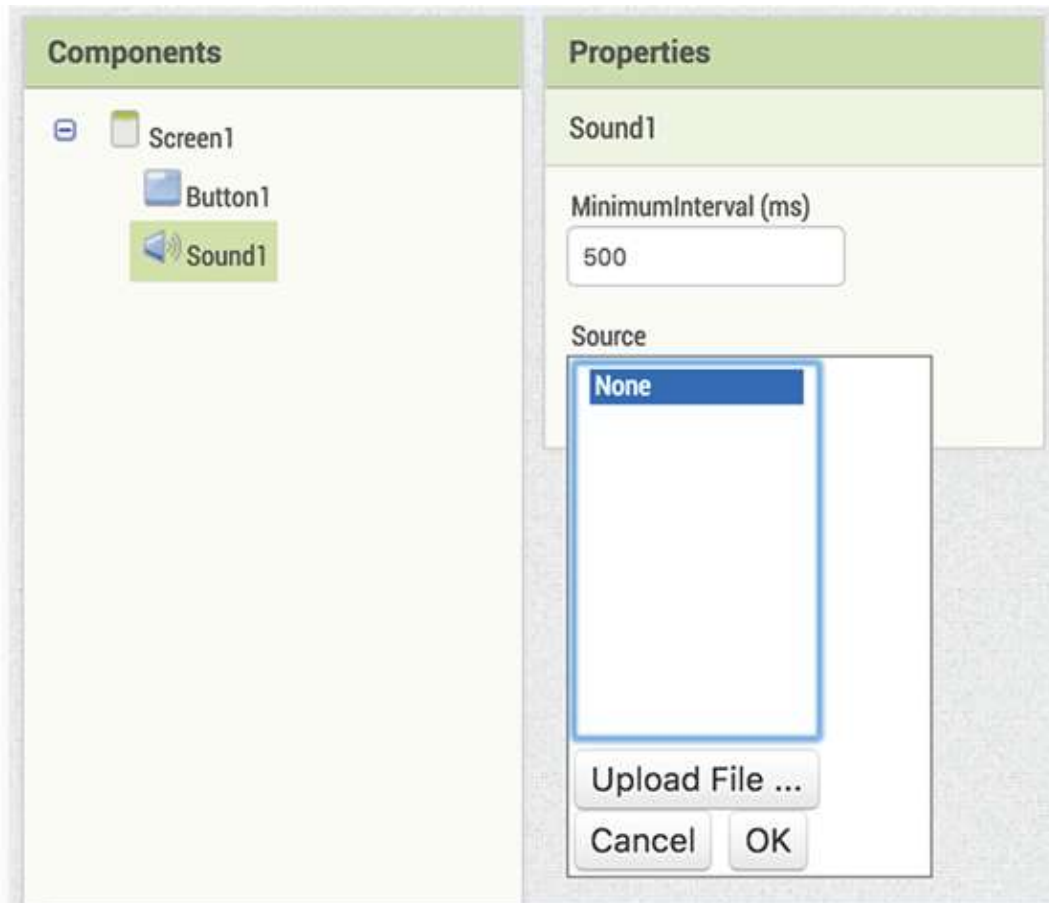
ShowFeedback
☒

Text
Make Sound

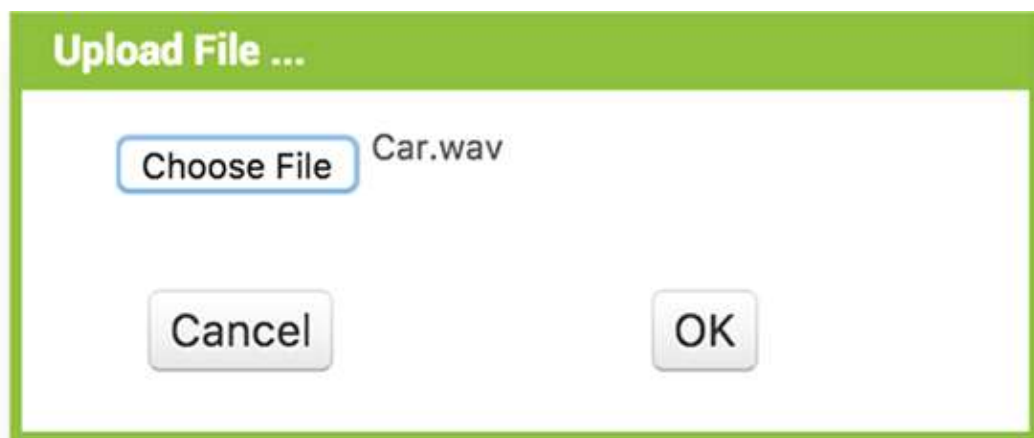
4. Under the Media section in the Palette column, click and drag Sound into your app viewer.



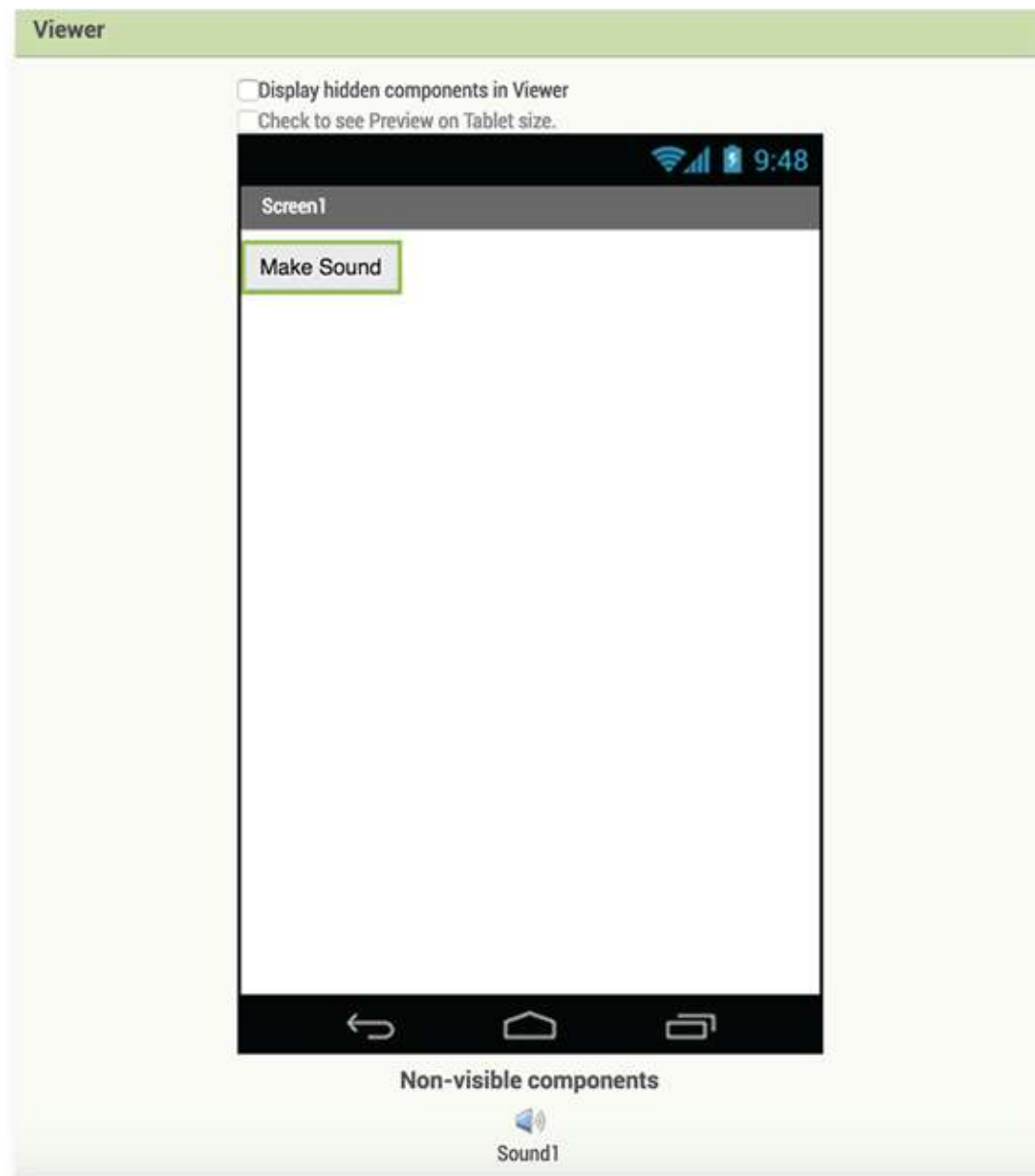
5. Download the sound file “Car.wav” from www.thewecan.zone/mobile-apps.
6. Under the Components column, make sure Sound is selected, and under the Properties column, find and click Source.



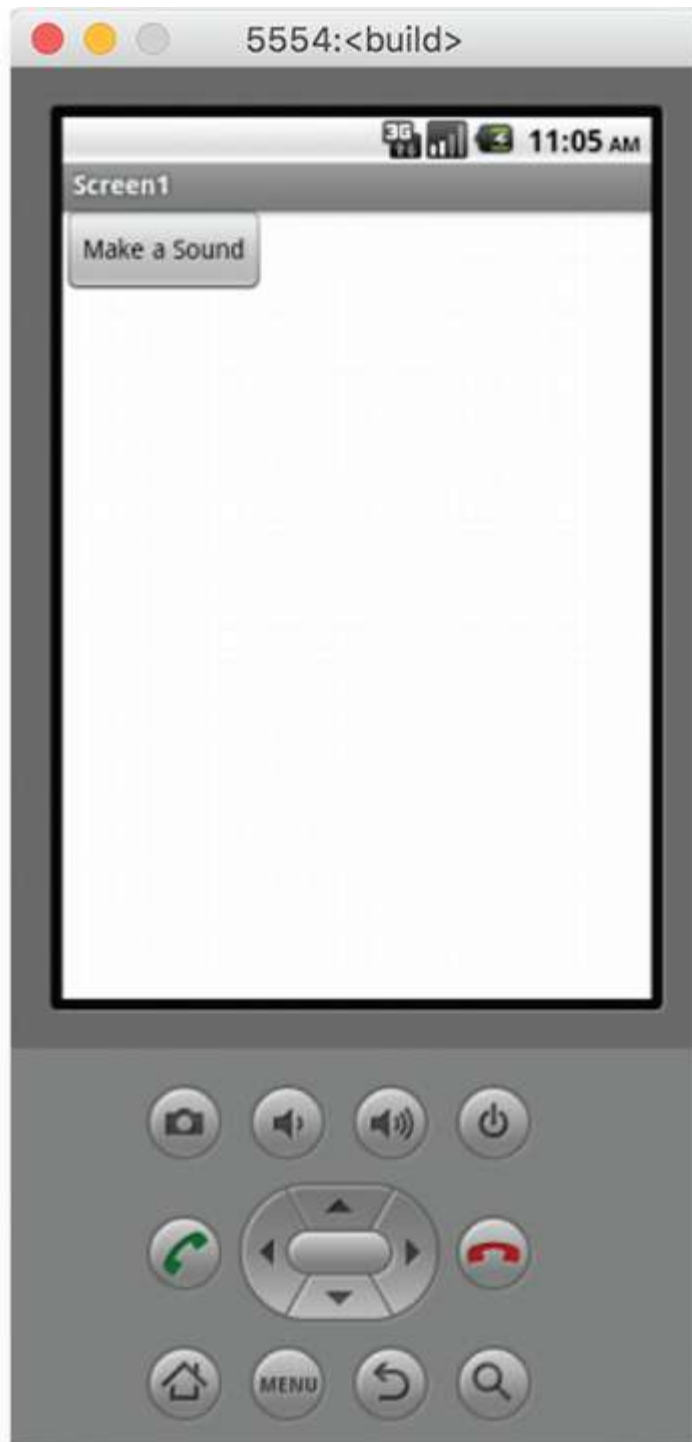
7. Click the Upload File button, and, using the popup, find the file that you downloaded in Step 5.



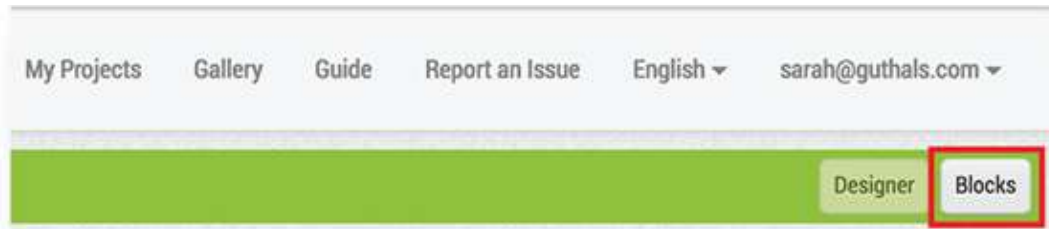
Your app viewer should look like this:



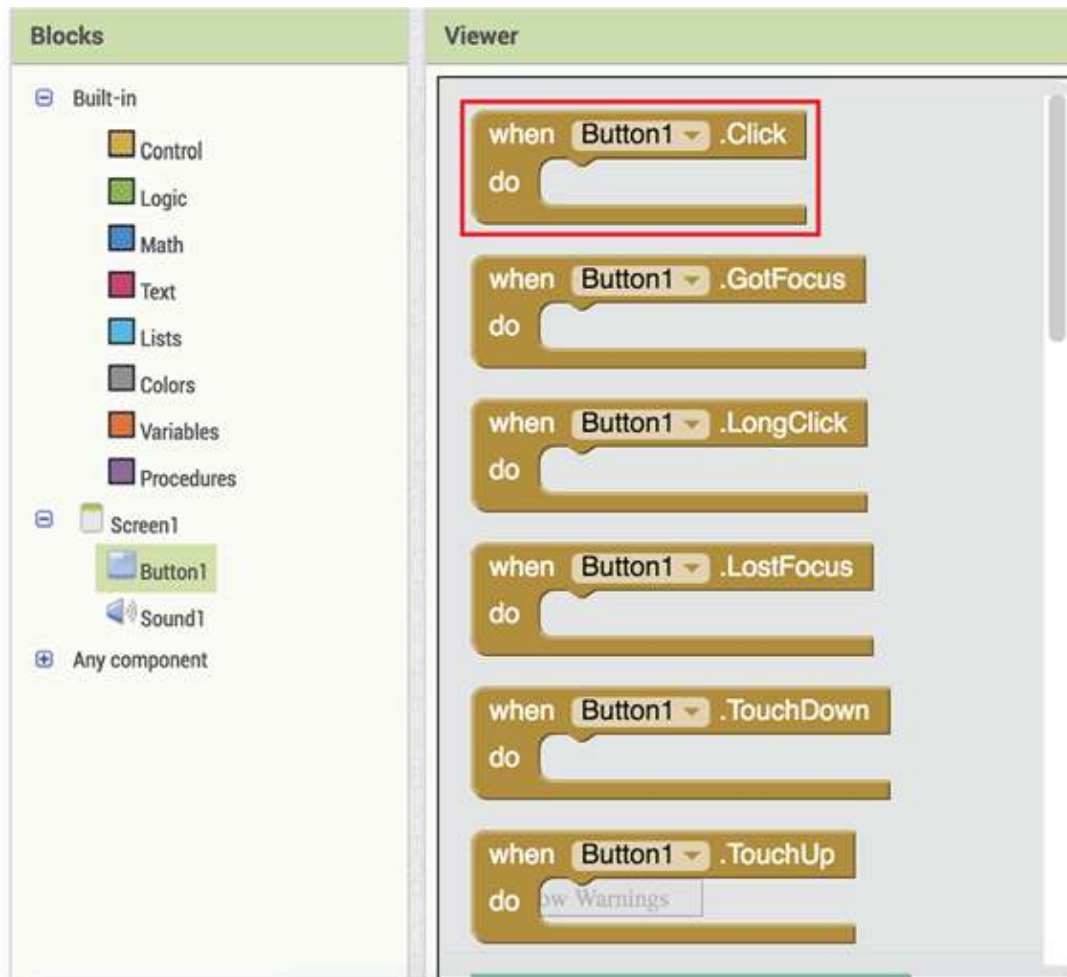
8. **Open your app emulator; you should now see the button you have added.**



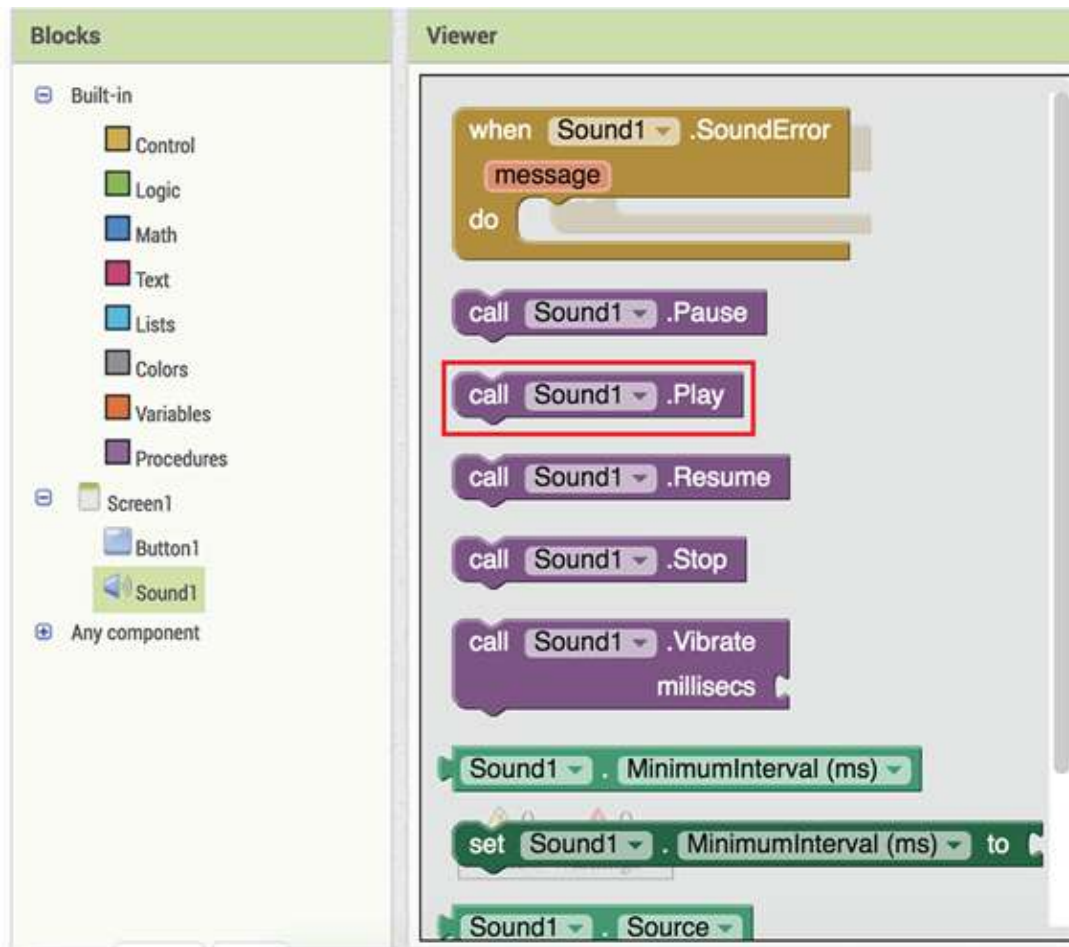
9. Go back to your Internet browser and click the Blocks button.



10. Click Button1 in the Blocks column and drag a When Button1.Click block into your viewer.



11. In the Blocks column, click Sound1 and drag a Call Sound1.Play block into your viewer. Place it *inside* the When Button1.Click block.



You just added code that tells the button you created to play the car sound when it's clicked!

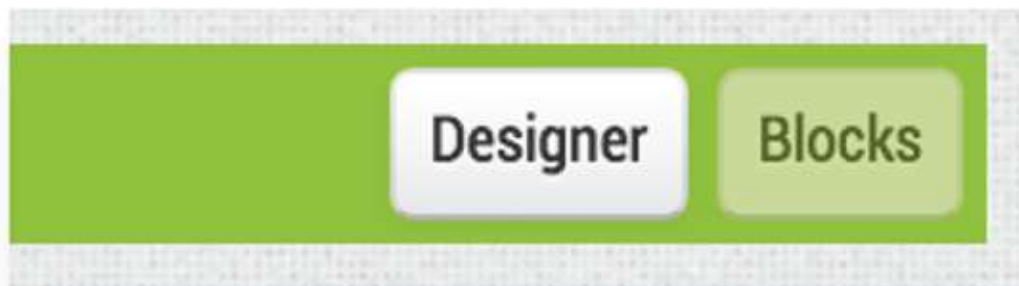


12. **Open your app emulator and click the button. You should hear a car sound!**

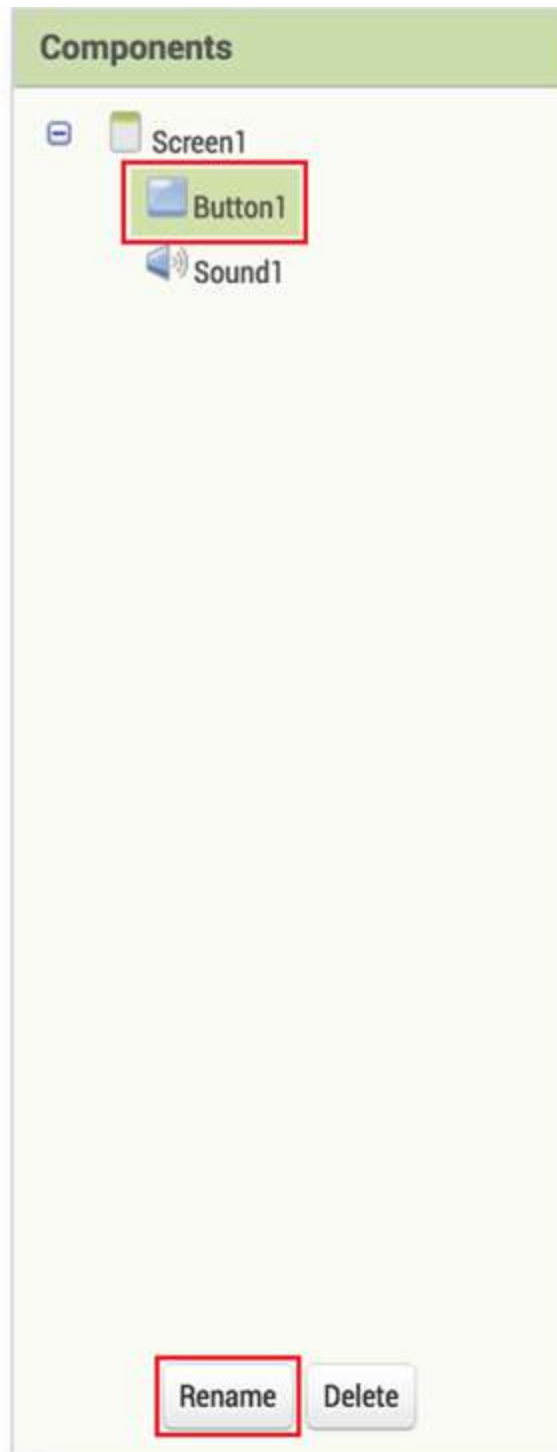
RENAME YOUR COMPONENTS

Before you add more to your app, you need to rename your components. “Button1” and “Sound1” work for now, because you only have one button and one sound. But in the next section, you will add three more buttons and one more sound, and you want to be able to know which button and which sound you are trying to write code for.

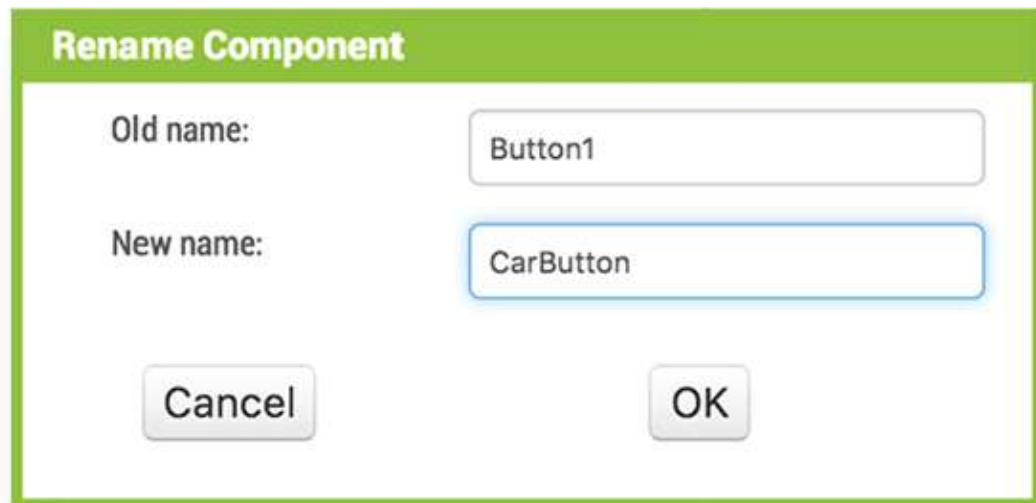
1. **In your Internet browser, click the Designer button.**



2. **In the Components column, click Button1, then the Rename button.**



3. Change the name of the button to “CarButton.”

A dialog box titled "Rename Component" with a green header bar. It contains two text input fields. The first field is labeled "Old name:" and contains the text "Button1". The second field is labeled "New name:" and contains the text "CarButton". At the bottom of the dialog, there are two buttons: "Cancel" on the left and "OK" on the right. The "New name:" field has a blue border, indicating it is the active field.

Rename Component

Old name:

New name:

4. In the Properties column, change the text on the button to “Make a Car Sound.”

Properties

CarButton

BackgroundColor
☐ Default

Enabled
☒

FontBold
☐

FontItalic
☐

FontSize
14.0

FontTypeface
default ▾

Height
Automatic...

Width
Automatic...

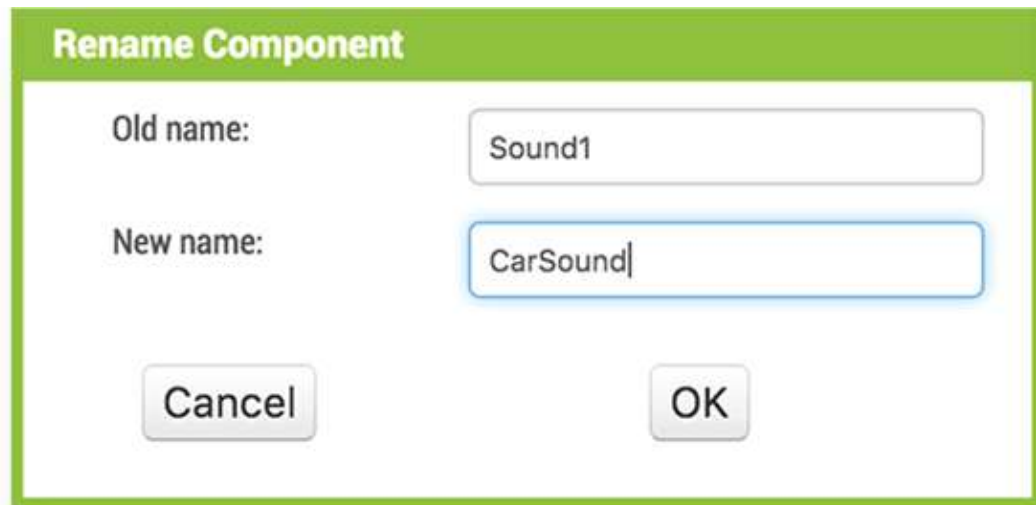
Image
None...

Shape
default ▾

ShowFeedback
☒

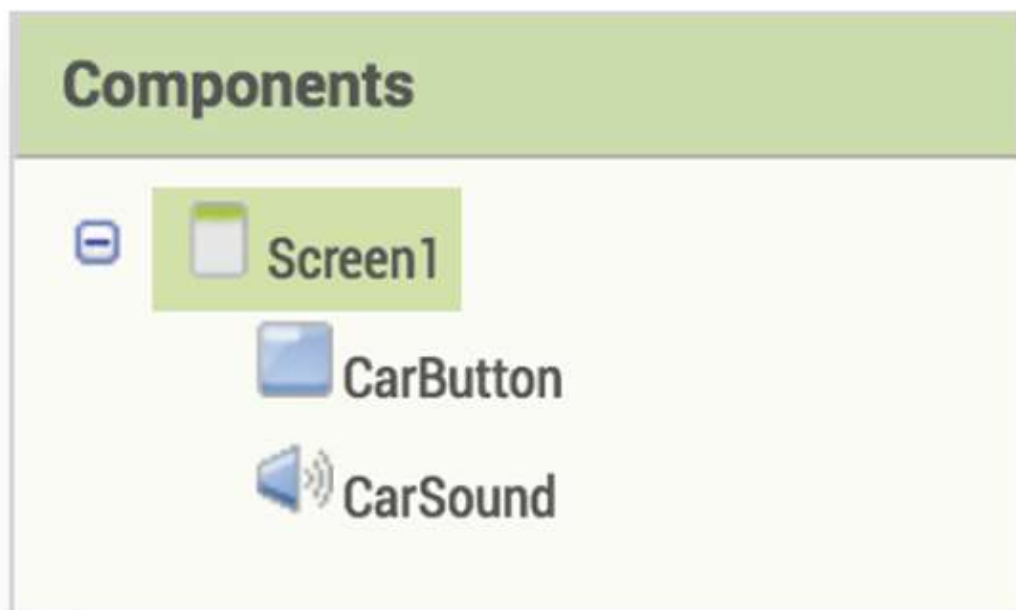
Text
Make a Car Sound

5. In the Components column, select Sound1, then click the Rename button and change the name of the sound to “CarSound.”

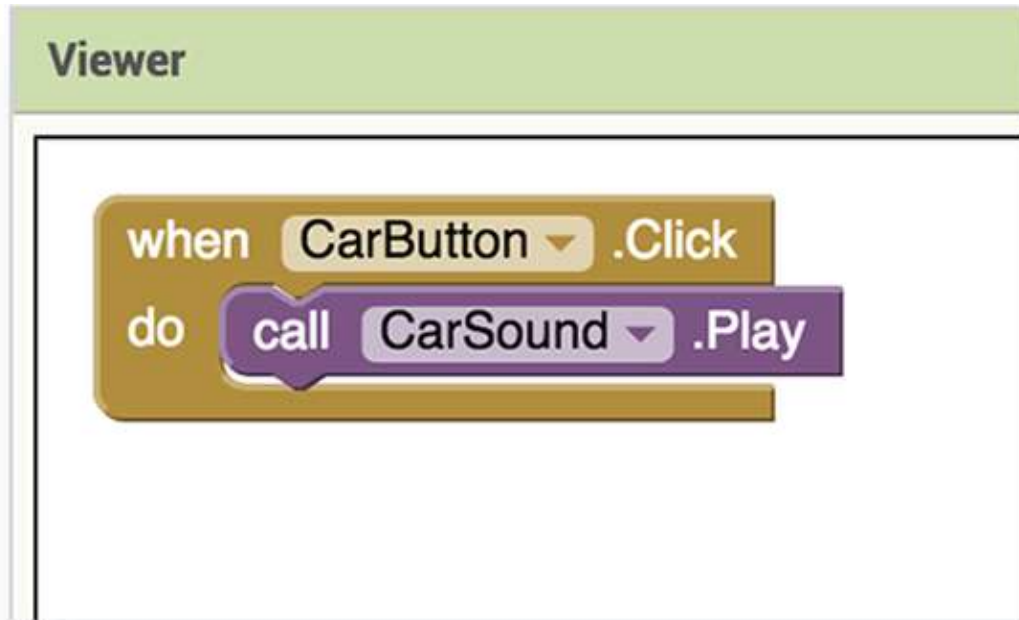


A dialog box titled "Rename Component" with a green header. It contains two text input fields. The first field is labeled "Old name:" and contains the text "Sound1". The second field is labeled "New name:" and contains the text "CarSound|". At the bottom of the dialog are two buttons: "Cancel" on the left and "OK" on the right.

6. Your Components section should now show the new names.



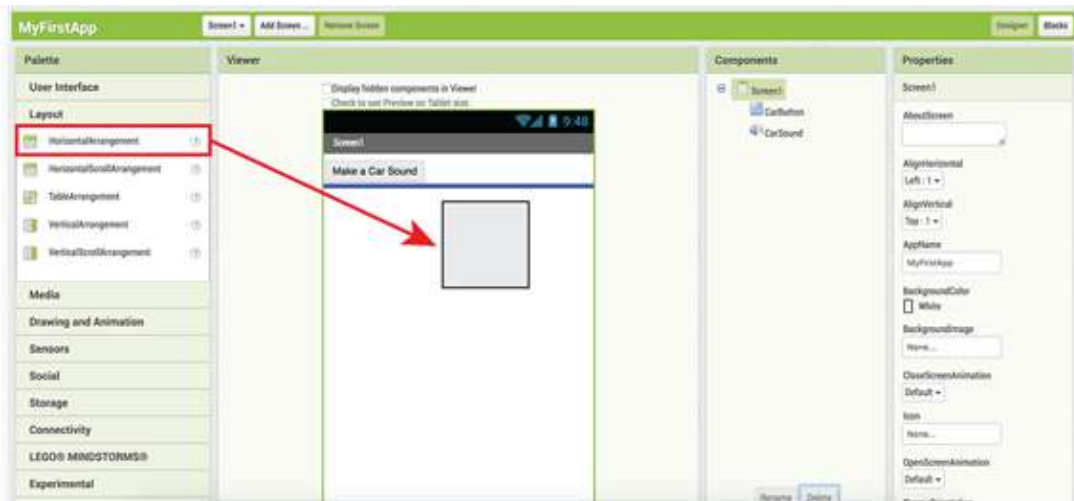
7. To make sure your code is telling the “CarButton” to play “CarSound” when it is clicked (it used to tell “Button1” to play “Sound1,” but those items don't exist anymore), click the Blocks button.



ADD A MUSIC PLAYER

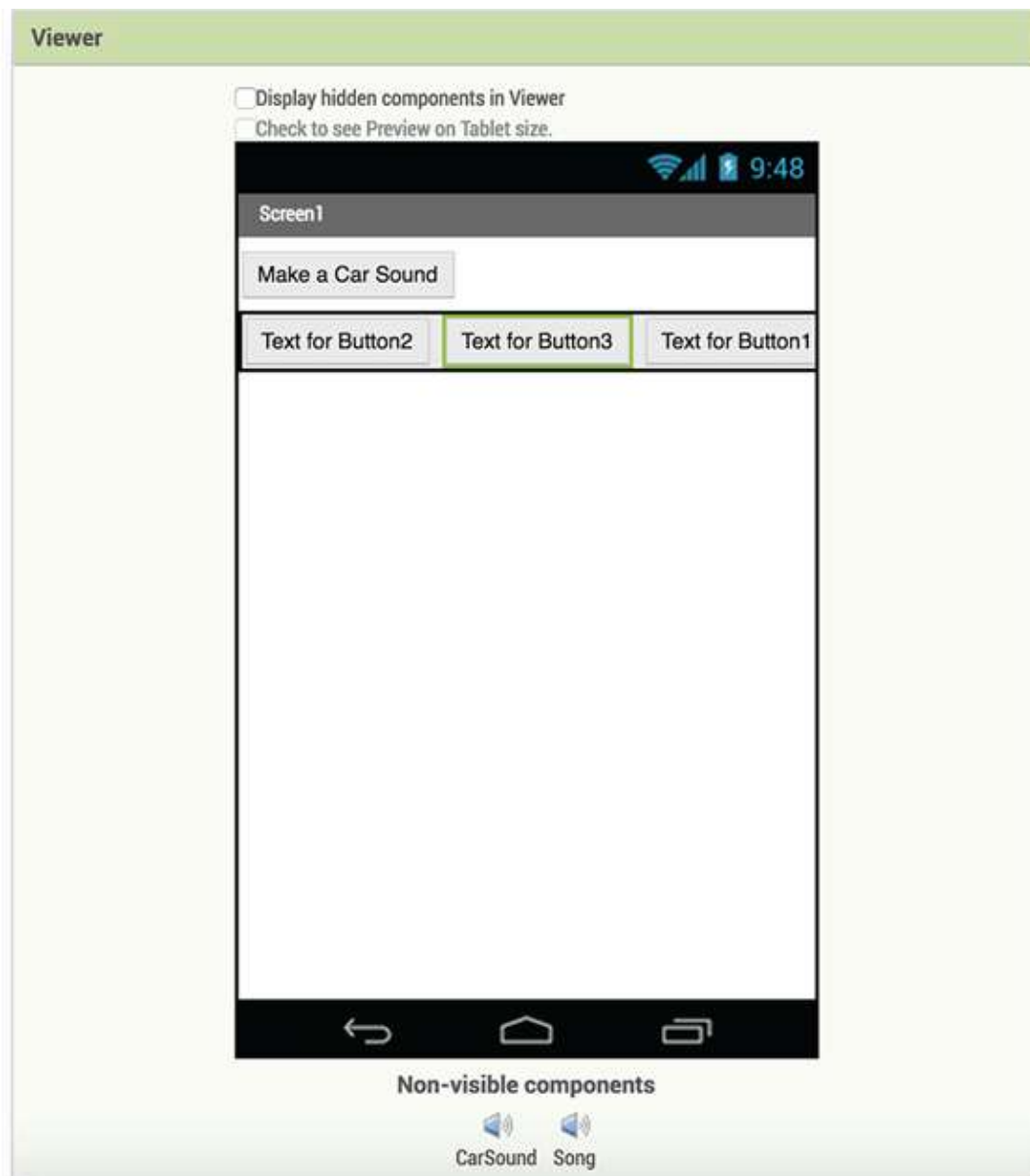
Now that you know how to add buttons that play sounds, you can make a simple music player!

1. **From the Palette column, drag a HorizontalArrangement to your viewer.**

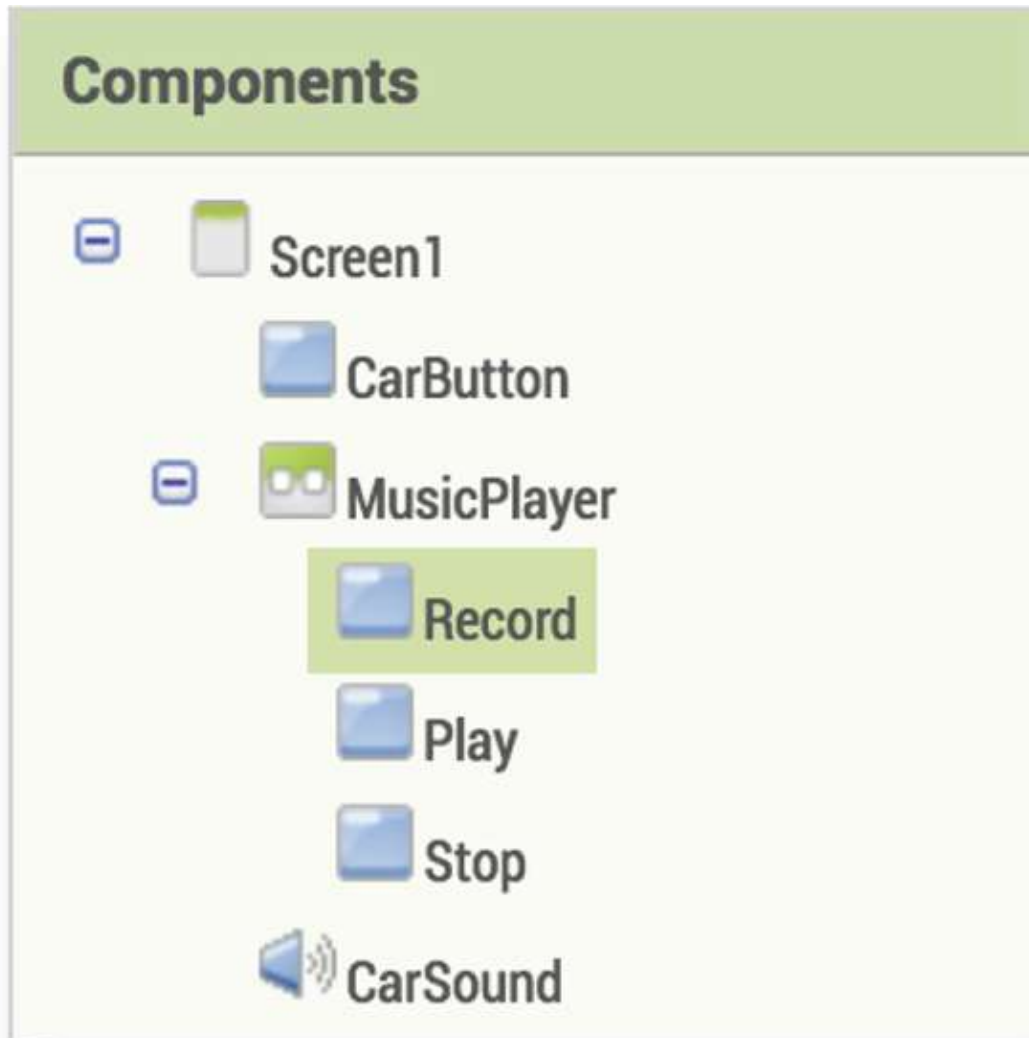


2. **Rename the HorizontalArrangement to "MusicPlayer."**

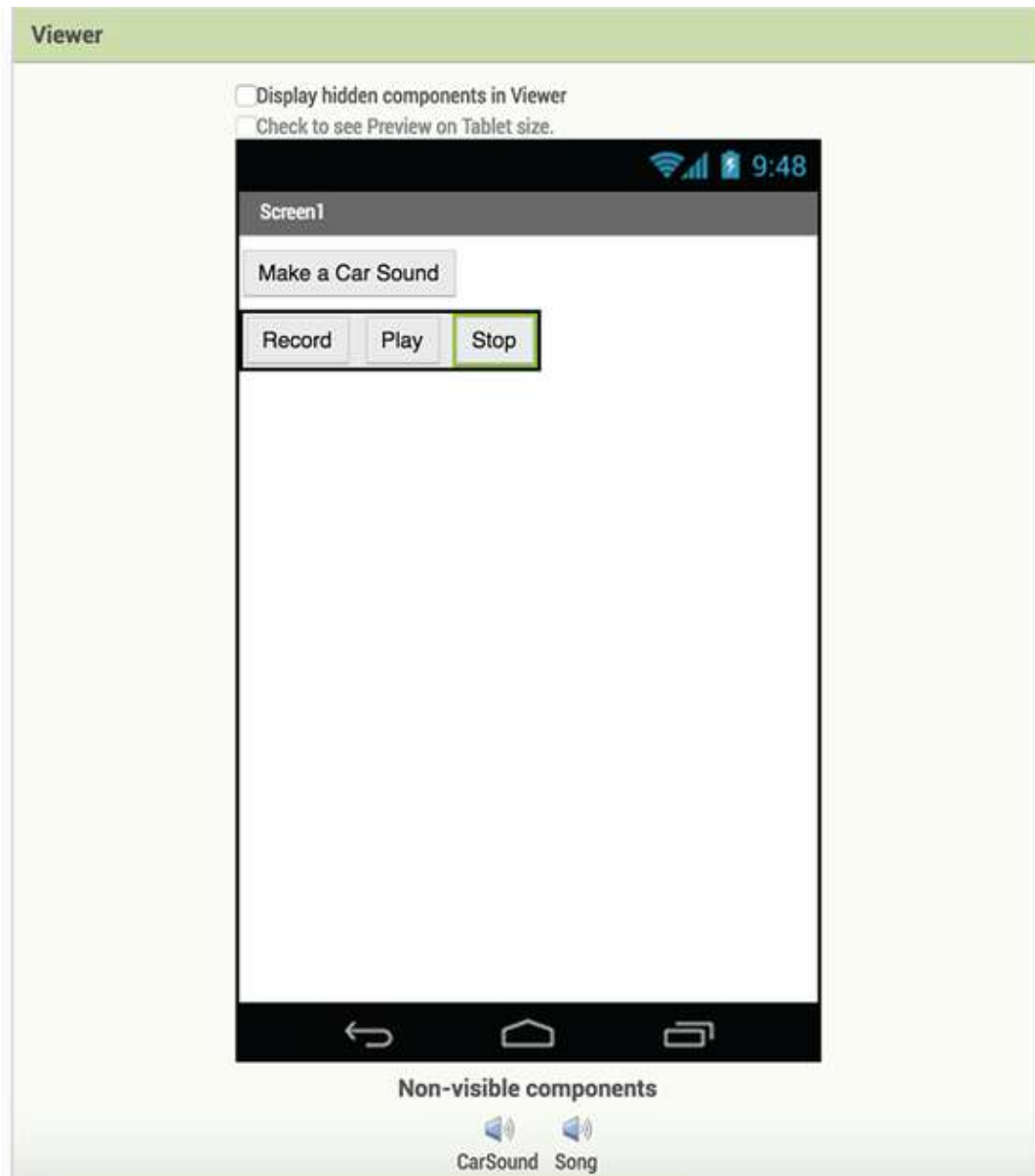
3. Add three buttons to your “MusicPlayer” layout.



4. Name the three buttons “Record,” “Play,” and “Stop.”

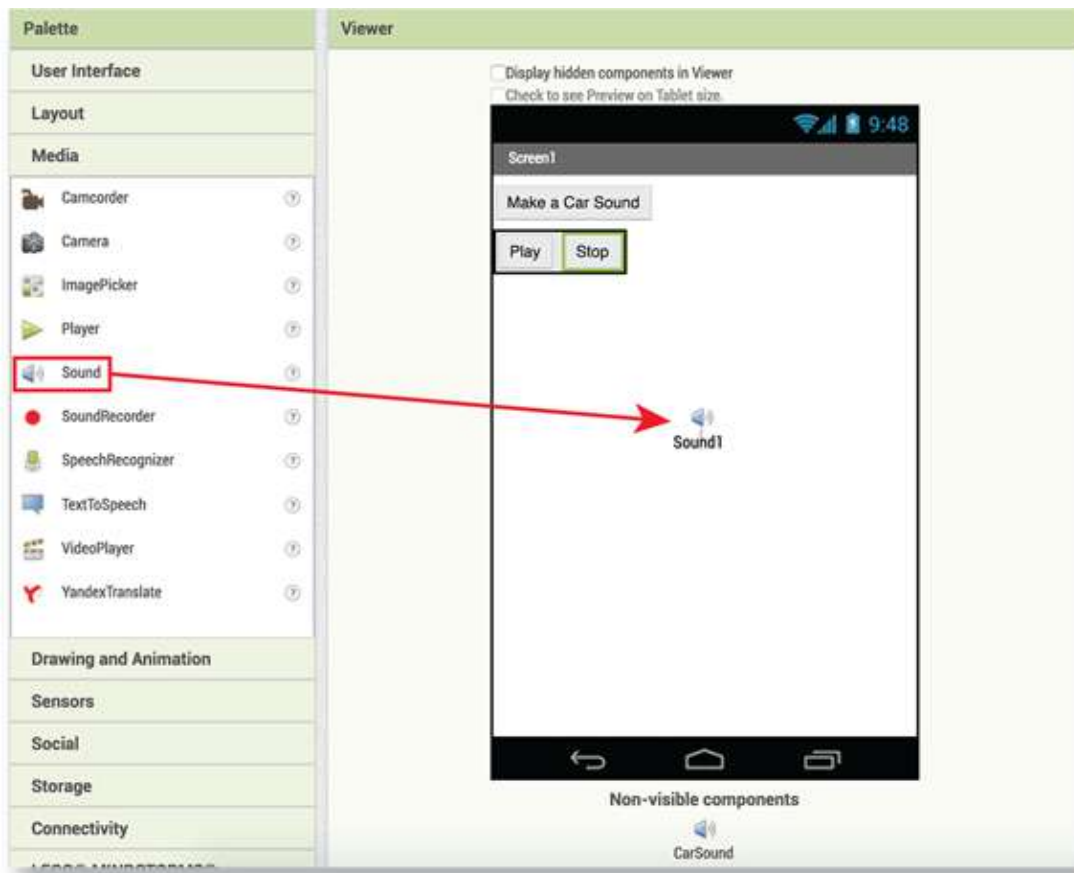


5. Change the text on the three buttons to “Record,” “Play,” and “Stop.”

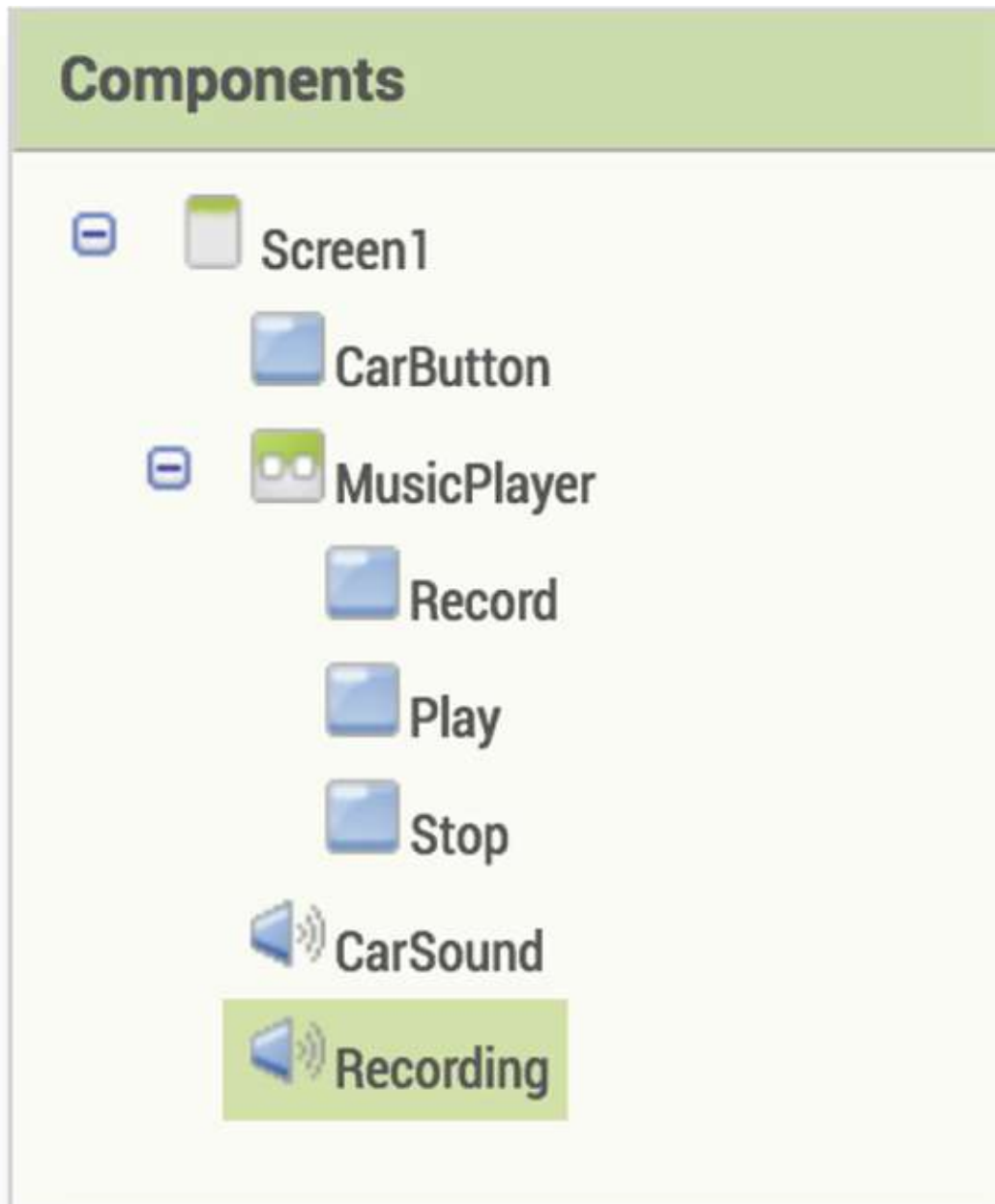


Make sure the text on the Play button is "Play." You don't want the text on the Stop button to be "Play," because that would be confusing!

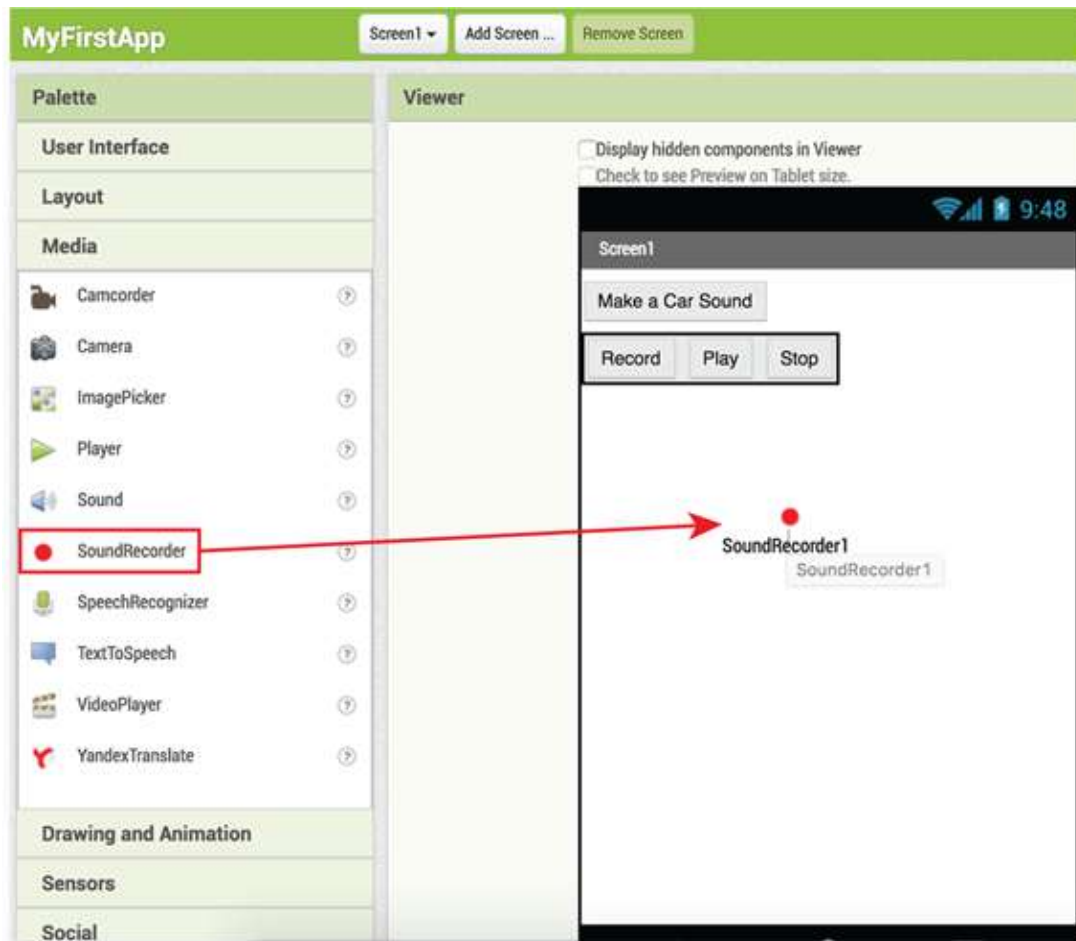
6. Add another sound to your viewer.



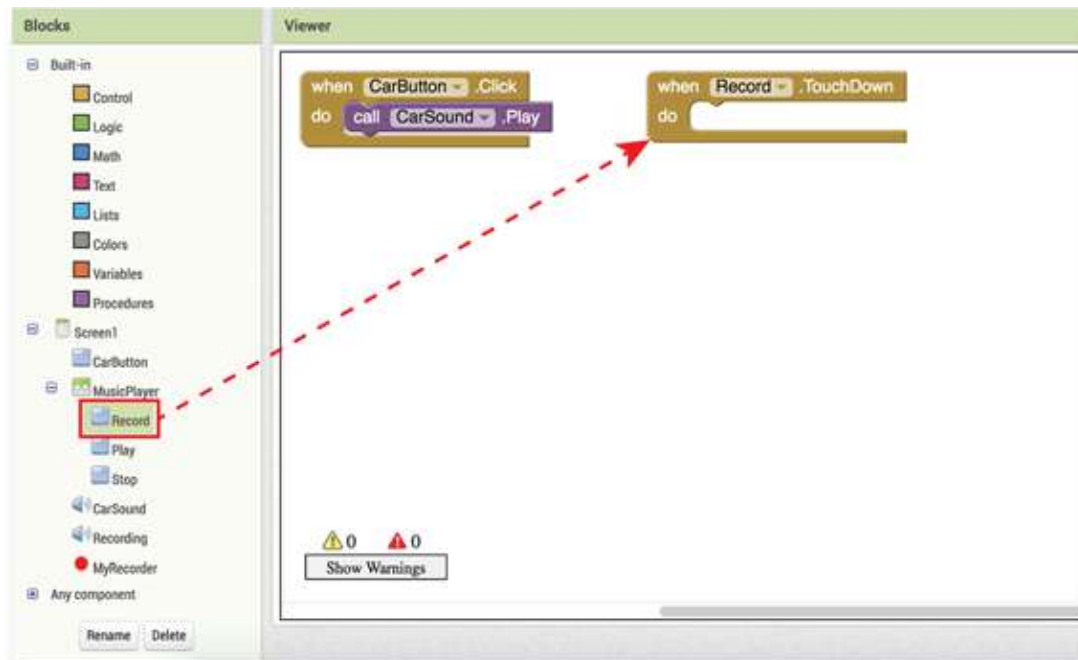
7. Rename Sound1 to “Recording.”



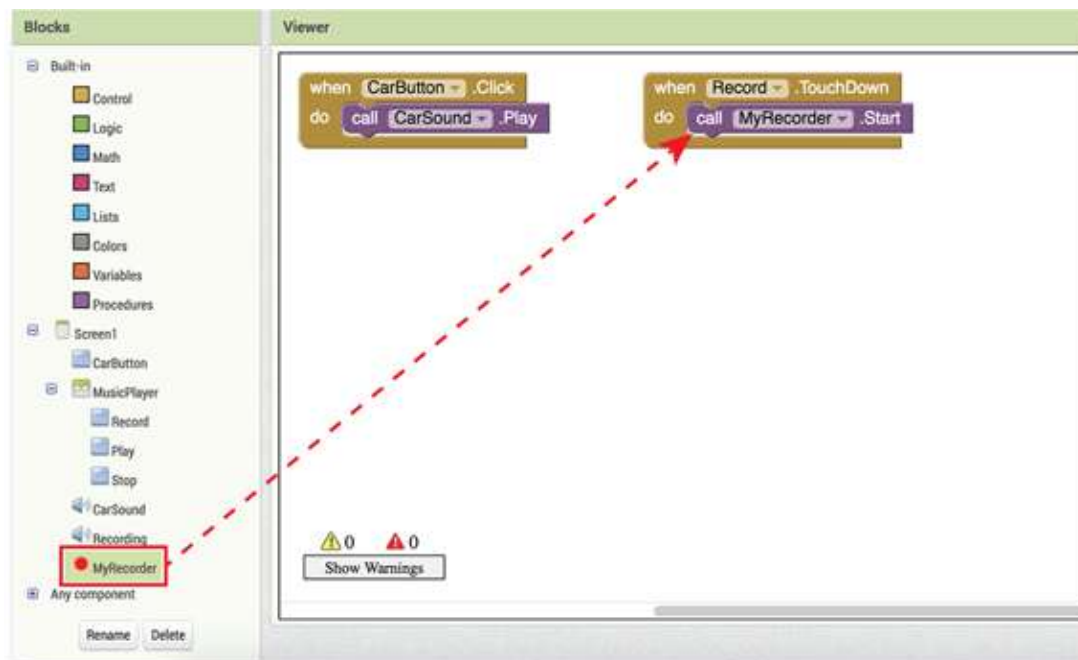
8. From the Media section of the Palette column, drag a SoundRecording into your viewer.



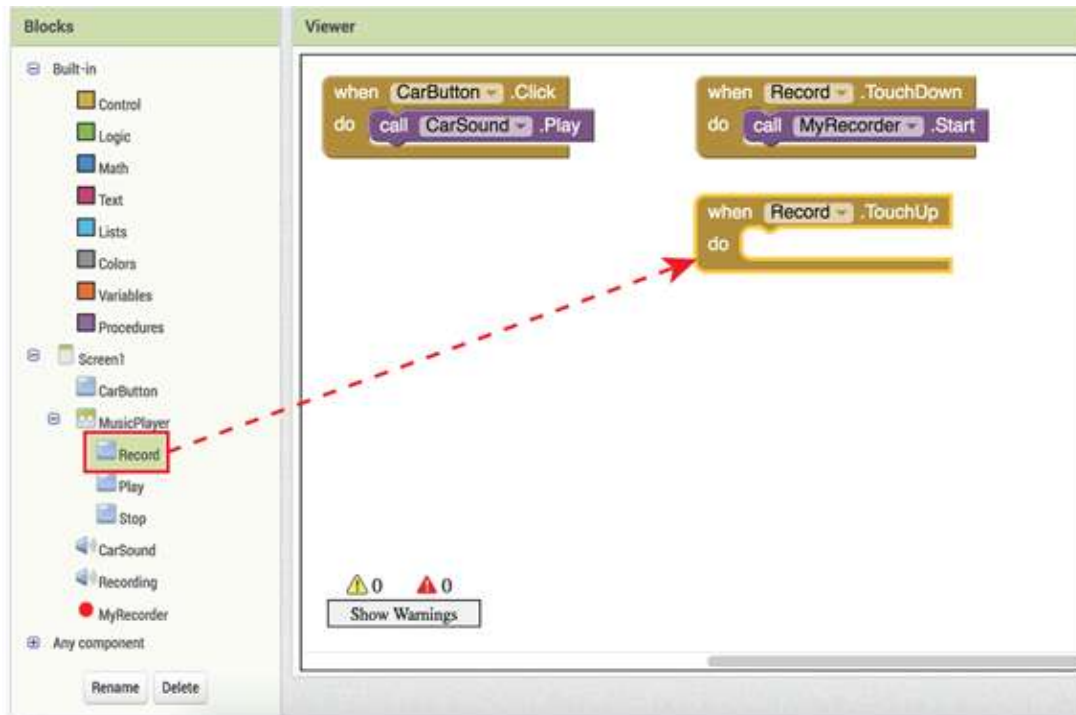
9. **Rename SoundRecorder1 to “MyRecorder.”**
10. **Click the Blocks button to get to the code viewer of your app.**
11. **In the Blocks column, select Record and drag When Record.TouchDown into your viewer.**



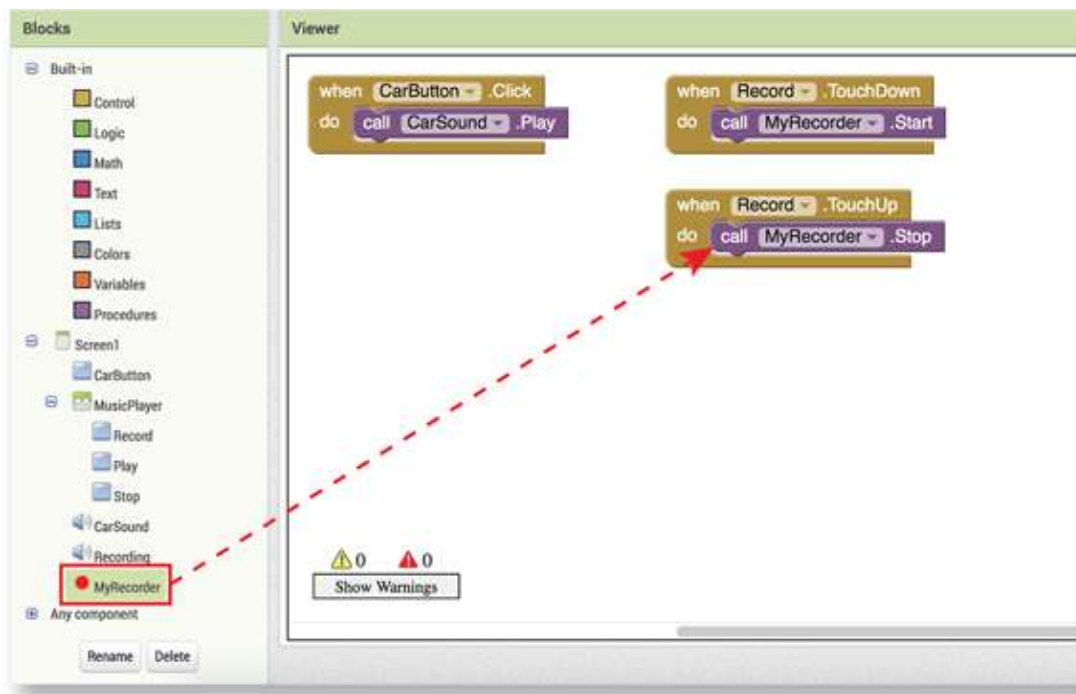
12. In the Blocks column, select MyRecorder and drag MyRecorder.Start into the When Record.TouchDown block.



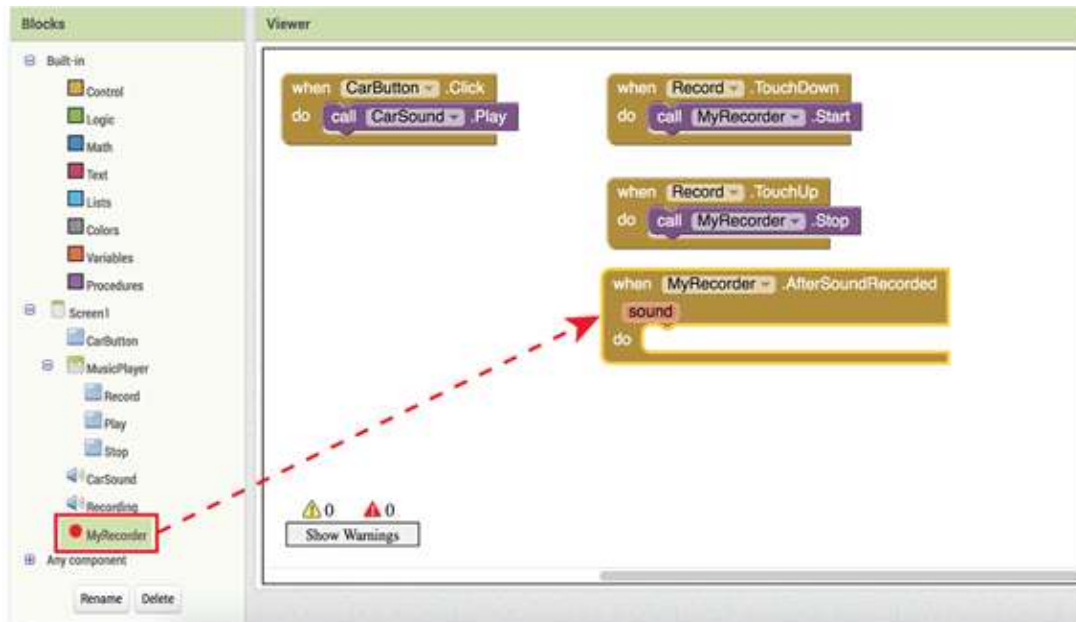
13. In the Blocks column, select Record and drag When Record.TouchUp into your viewer.



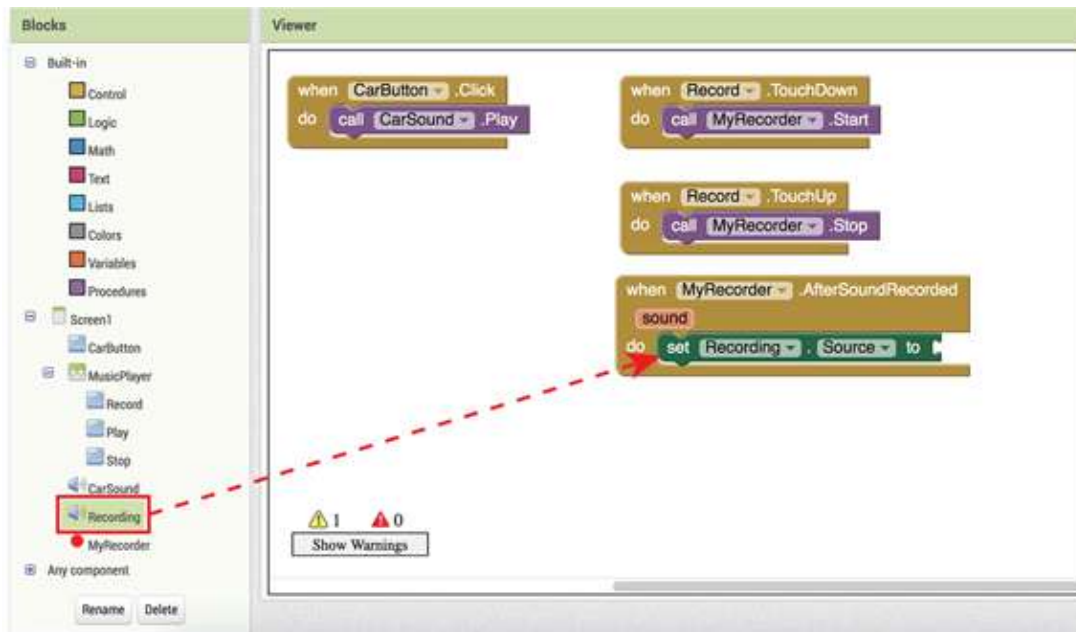
14. In the Blocks column, select MyRecorder and drag MyRecorder.Stop into the When Record.TouchUp block.



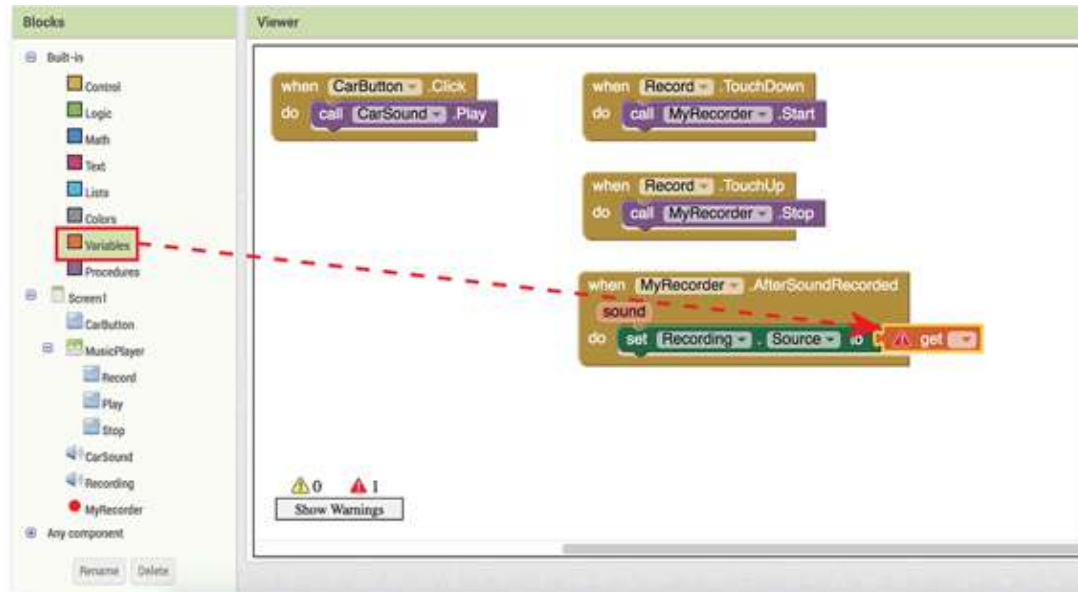
15. Select MyRecorder and drag When MyRecorder.AfterSoundRecorded into your viewer.



16. Select Recording and drag Set Recording.Source To into the When MyRecorder.AfterSoundRecorded block.



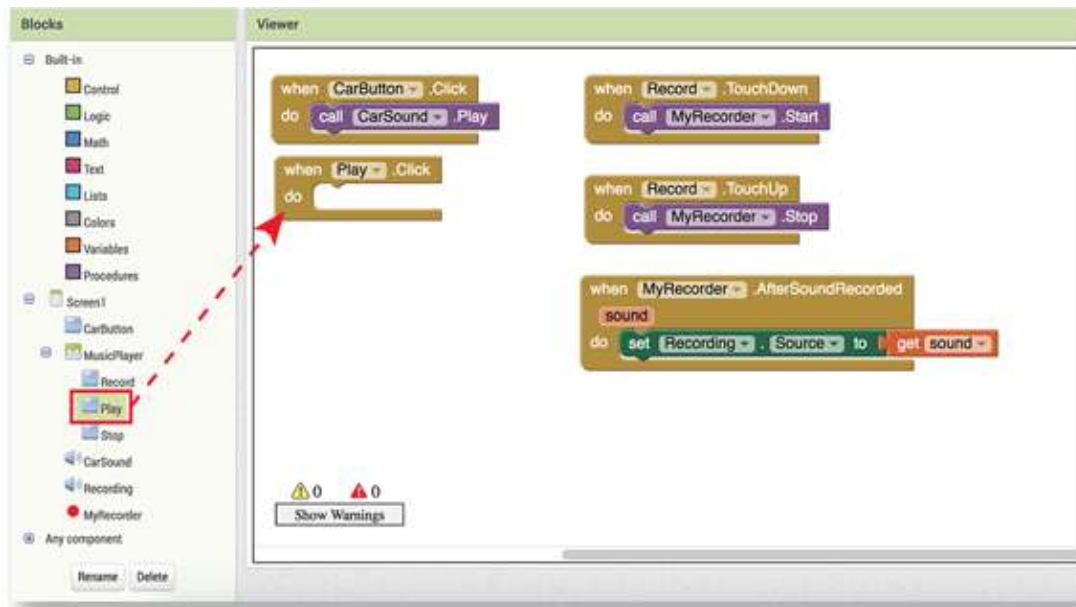
17. In the Built-In section of the Blocks column, select Variables and drag a Get block into your viewer, connecting it to your Set Recording.Source To block.



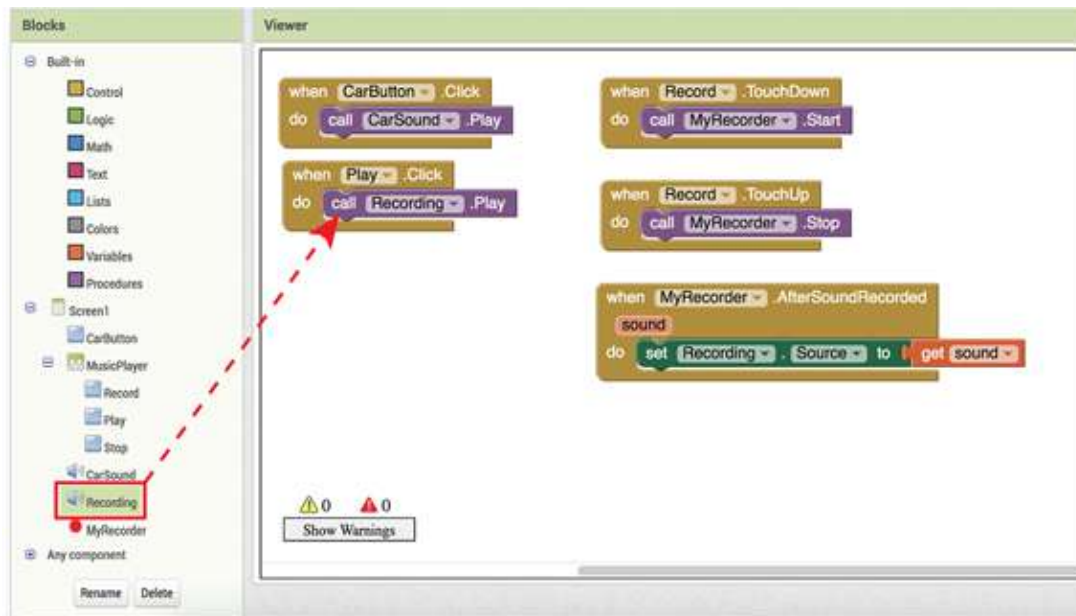
18. Click on the drop down on your Get block and change it to Sound.



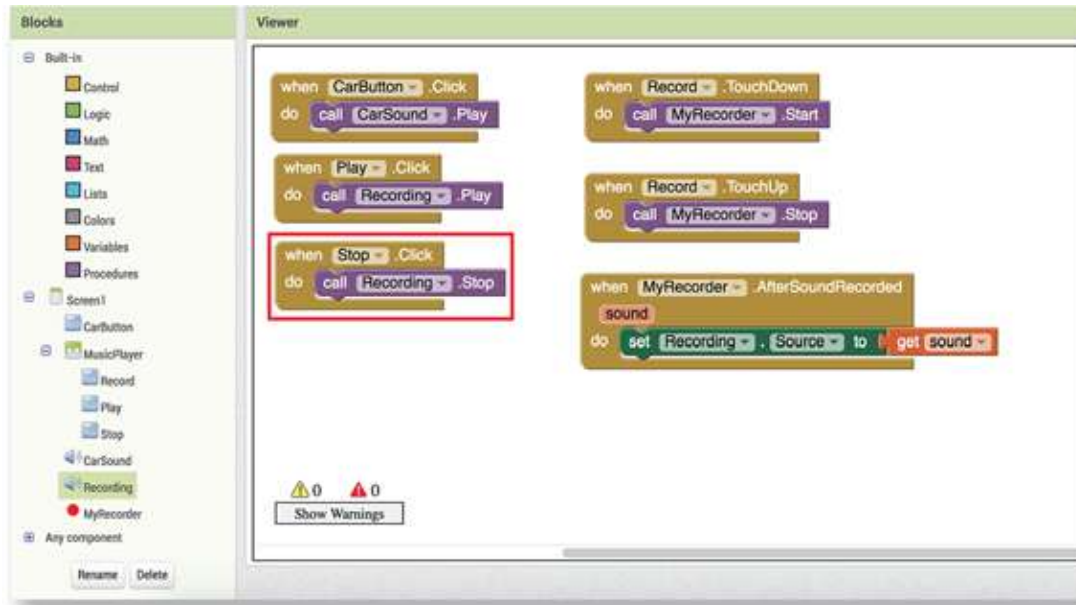
19. In the Blocks column, select Play and drag a When Play.Click block into your viewer.



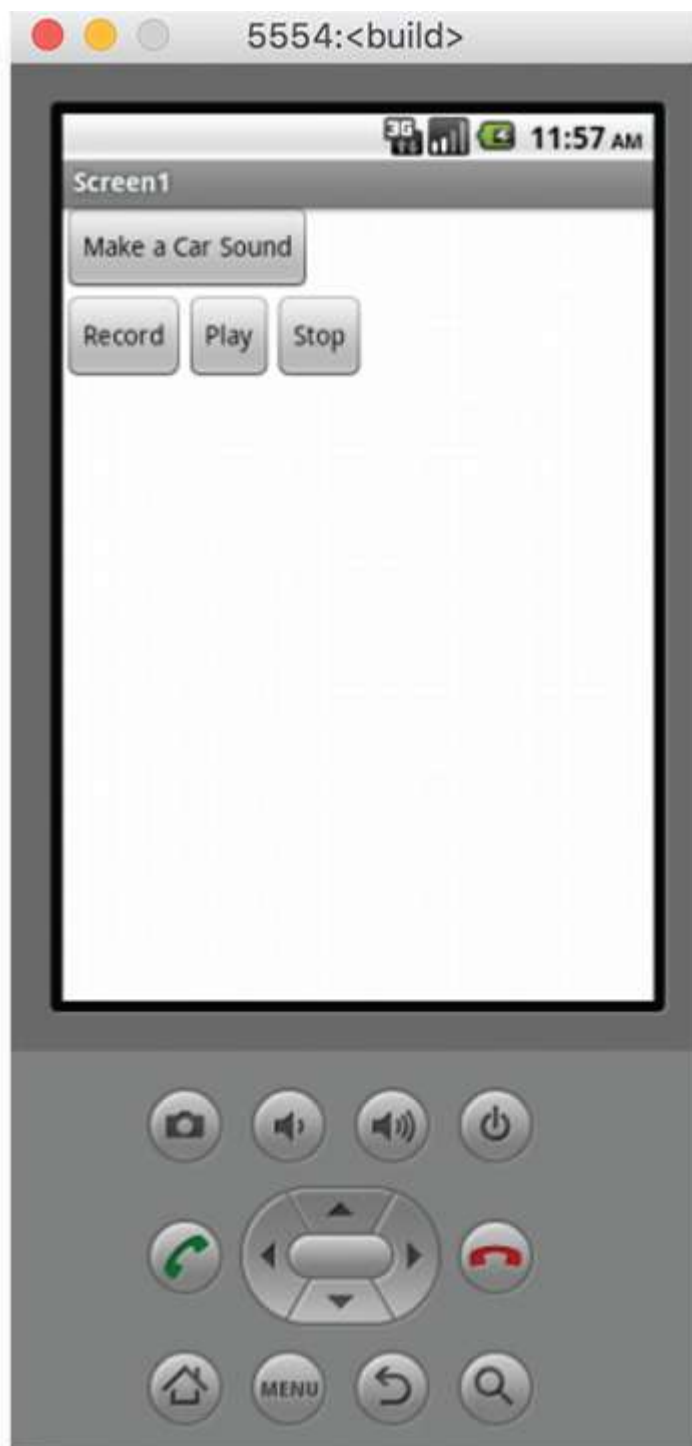
20. Select Recording and drag a Call Recording.Play block into your When Play.Click block.



21. In the Blocks column, select Stop and drag a When Stop.Click block into your viewer, then select Recording and drag a Call Recording.Stop block into your When Stop.Click block.

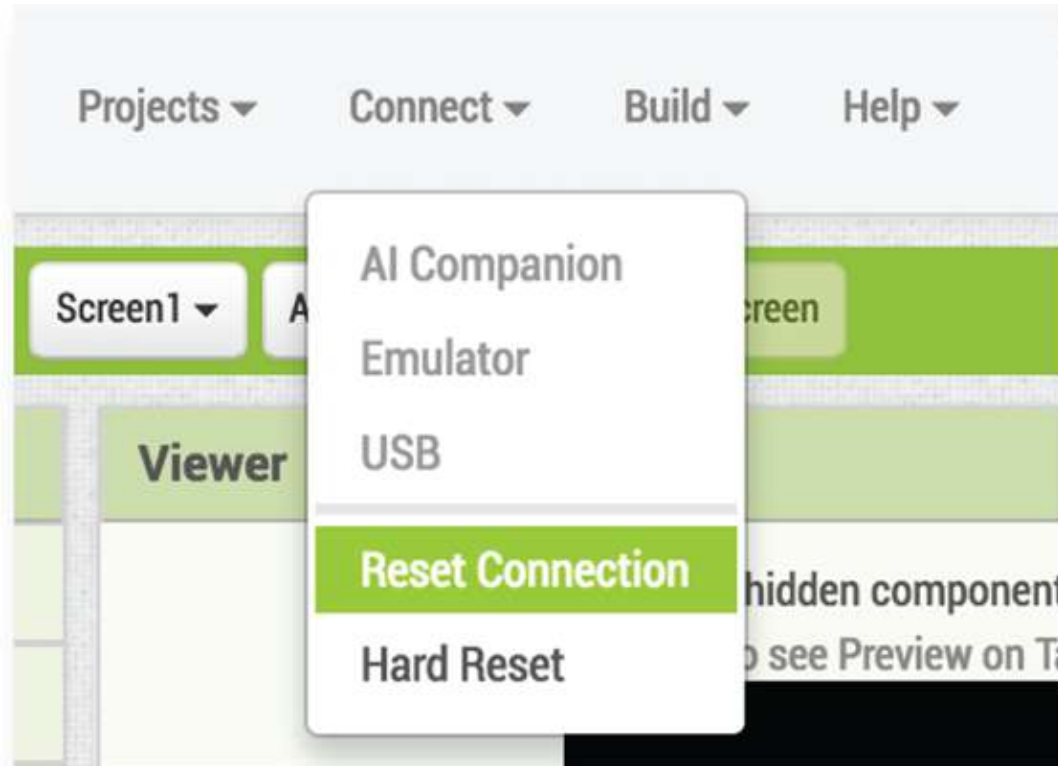


22. Open your app emulator. You now should see the renamed button and the three new buttons.





If your app emulator is not refreshing properly, you can go back to your Internet browser and choose Reset Connection in the Connect menu item at the top. Then, click open the Connect menu item again and choose Emulator to start the emulator again.



TEST YOUR MUSIC APP

Now that you have finished building your app, you can test it out!

First, click the Make a Car Sound button. You should hear the car sound.

Then, press and hold the Record button and sing something into your computer. Release the Record button and press the Play button. You should hear the song you just sang! If you press the Stop button while you are listening to the song that you recorded, it will stop!

Play around with the app you just made and maybe show it to one of your family members or friends!



Coding isn't about always following directions and doing only what you're told. It's also about being creative and trying something new! If you have an idea, go ahead and try it, even if you haven't learned how to do it yet!

PROJECT 2

MAKE AN APP ABOUT YOU



IN THIS PROJECT, YOU LEARN HOW TO MAKE AN APP WITH MULTIPLE SCREENS, AND YOU LEARN ABOUT A LOT OF THE FUN FEATURES OF APP INVENTOR. You first design your app on

paper, then build a skeleton app and fill in the content, learning the entire process of app making that the pro developers use!

DESIGN YOUR APP

Professional app developers always design their apps before they start building. Having a good design makes it easier to develop a good app, because you already know what you want to build.

Another name for a design of an app is a *paper prototype*. A *prototype* is an early version of something you are making. Making a prototype can help you show it off to people and get new ideas, without spending hours or days building it. Prototypes are used in fields other than computer science, too. For example, before building a bridge, an architect may build a prototype of the bridge to make sure it can hold the weight of all the cars, and to make sure it looks as nice as she thought it would.

Paper prototypes are early versions of the app you plan on making, but done only on paper. Other types of prototypes can be drawings made on the computer or even mini-versions of your completed project (like architects do with their bridges).

MAKE YOUR FIRST PAPER PROTOTYPE

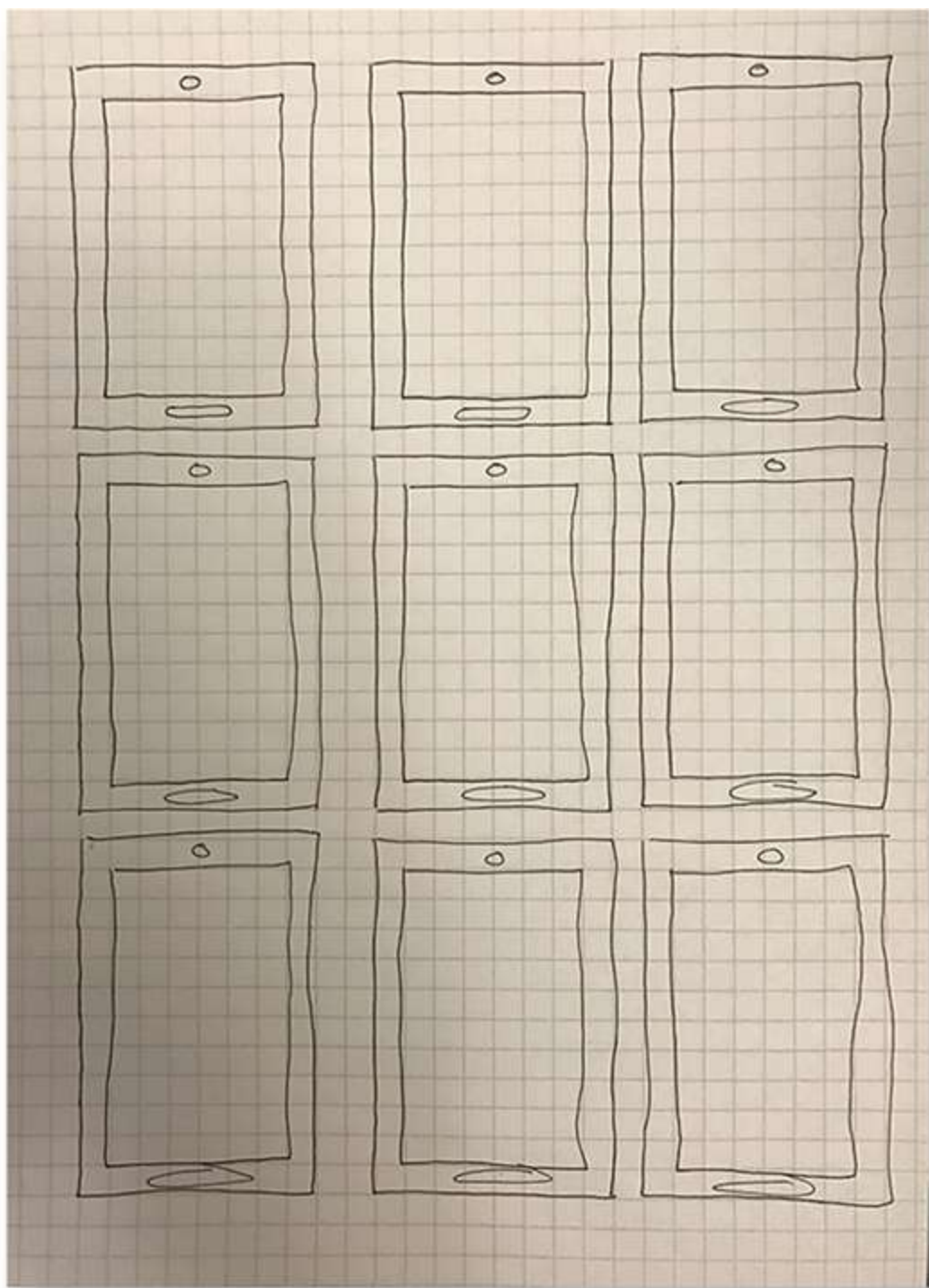
Now you should make your first paper prototype. To start, you need to get your paper ready for prototyping. Then, you can fill your paper phones in with what you want to show on your app. Finally, you can decide what the buttons will do on your app.

MAKE PAPER PHONES

Follow these steps to make paper phones:

- 1. Grab any piece of paper and a writing utensil. I'm using grid paper and a red, blue, and black pen.**
- 2. Draw rectangles for sketching out your app.**

These rectangles will be like your app. You can make them look like a phone, like I have, or you can just make simple rectangles.

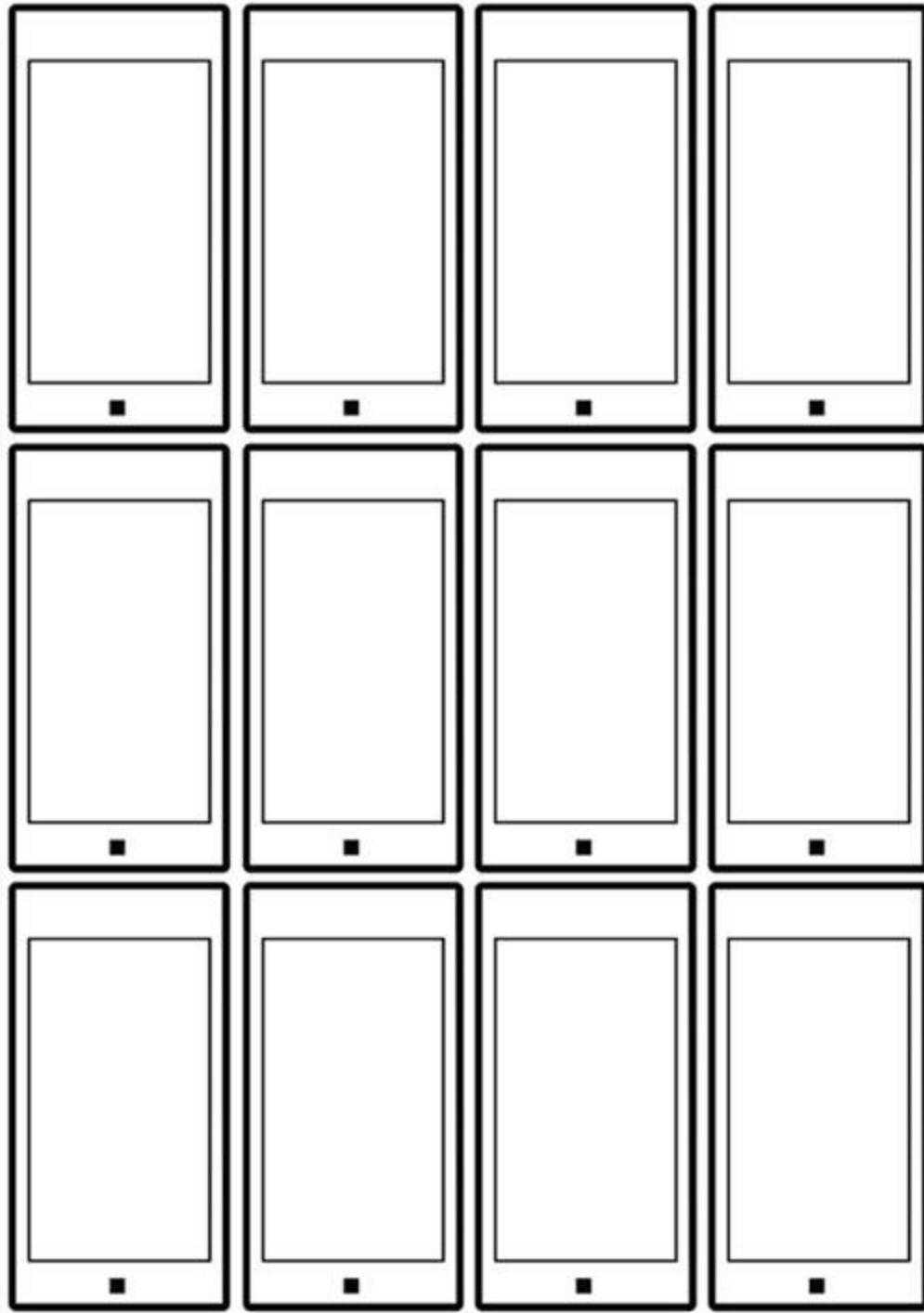




You might need more than can fit on one sheet. You can either make photocopies of the rectangles you drew or draw more!

3. Print paper smartphones if you want.

If you don't want to draw your rectangles, you can print them instead. Go to www.thewecan.zone/books/mobileapps and download PDF versions that you can print out and use.

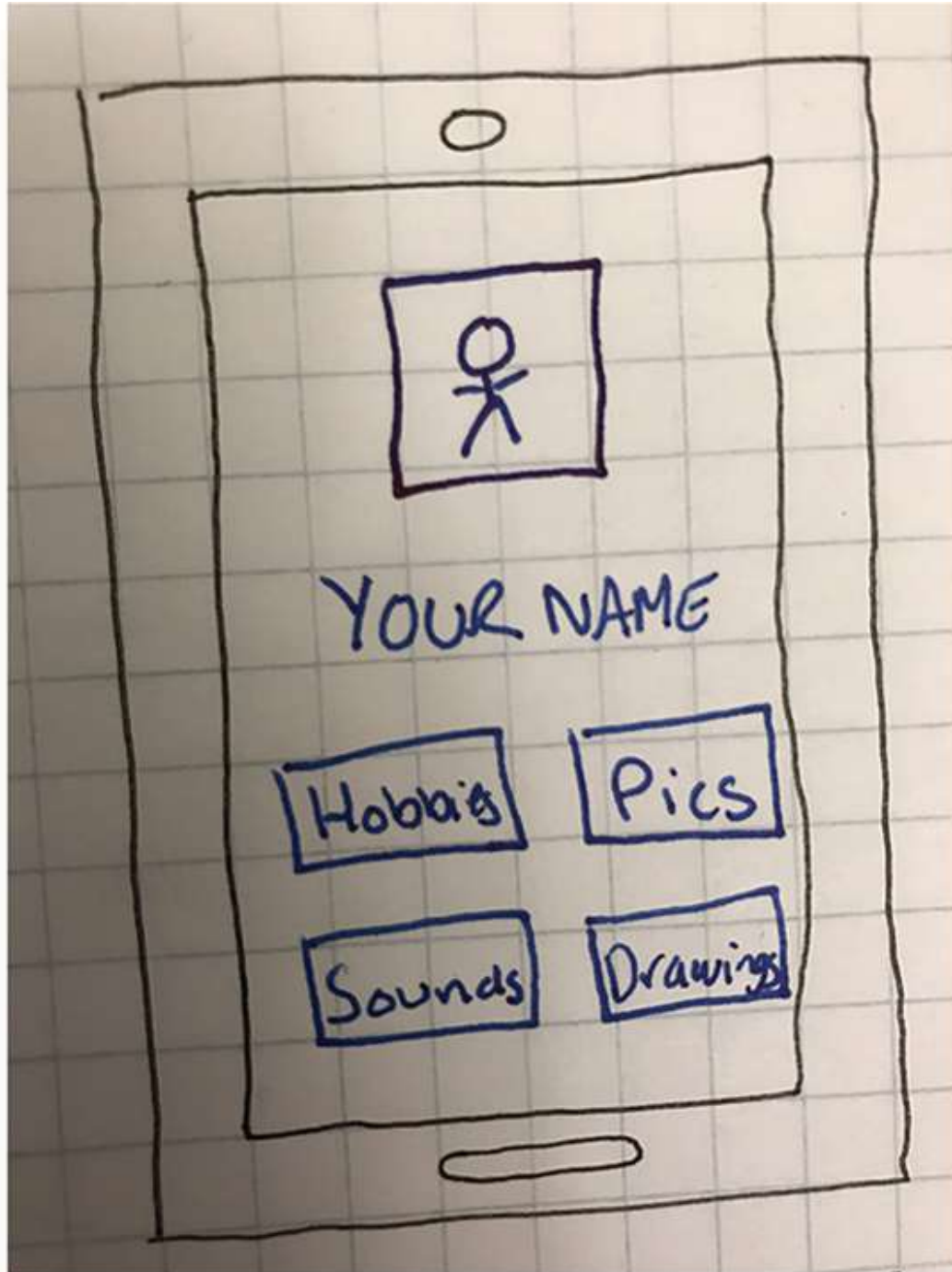


FILL IN YOUR PAPER PHONES

Follow these steps to decide what should be on each screen. Each paper phone should be considered a different screen.

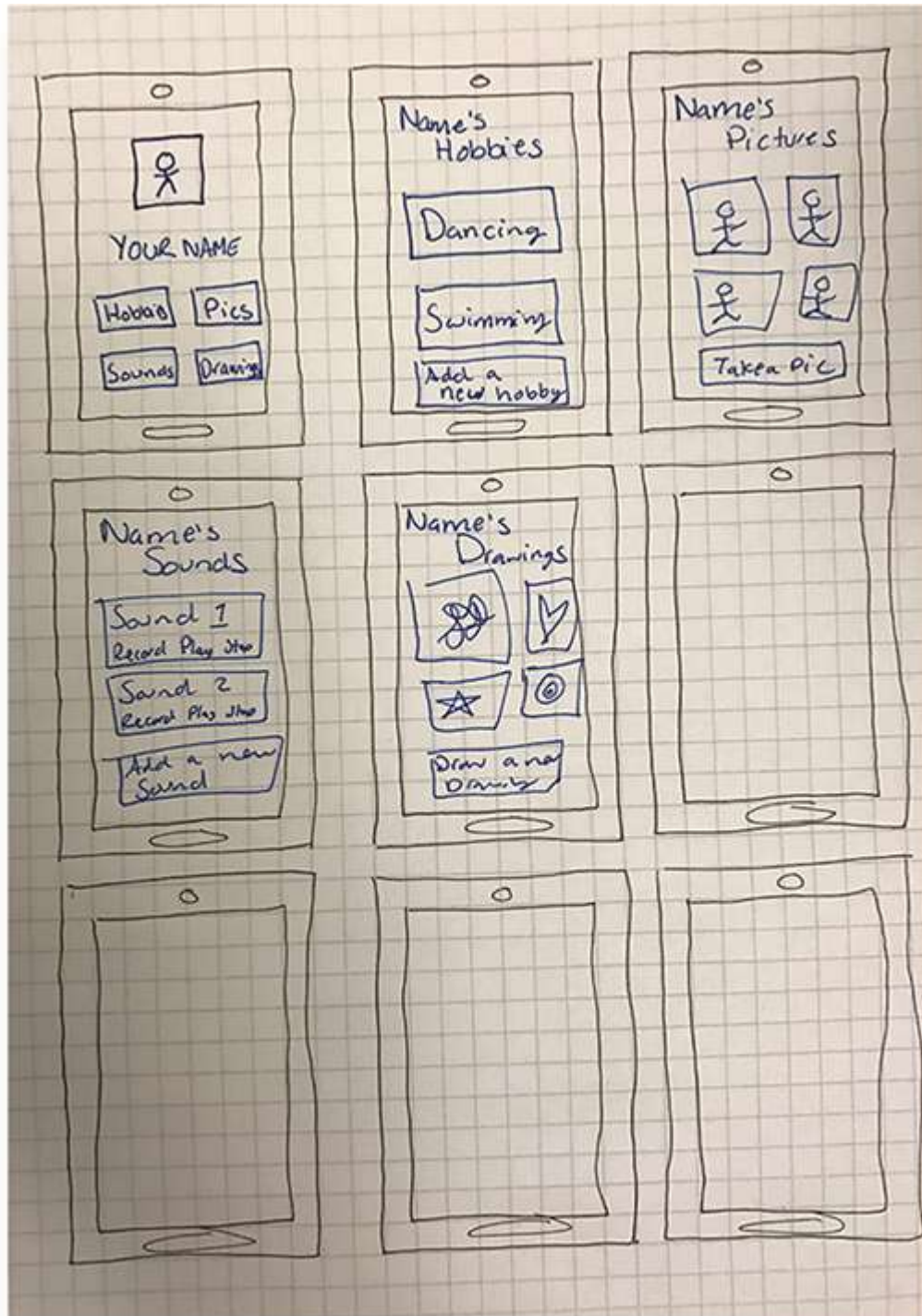
1. Design your homepage.

When your users open your app, the first screen that they see is called the *homepage*. Draw what they will see.



2. Plan your other pages.

After you decide on the categories you want in your app, start designing each of those screens.



MAKE THE SKELETON OF YOUR APP

In the previous section, you designed your app on paper. Now you build the design with App Inventor.

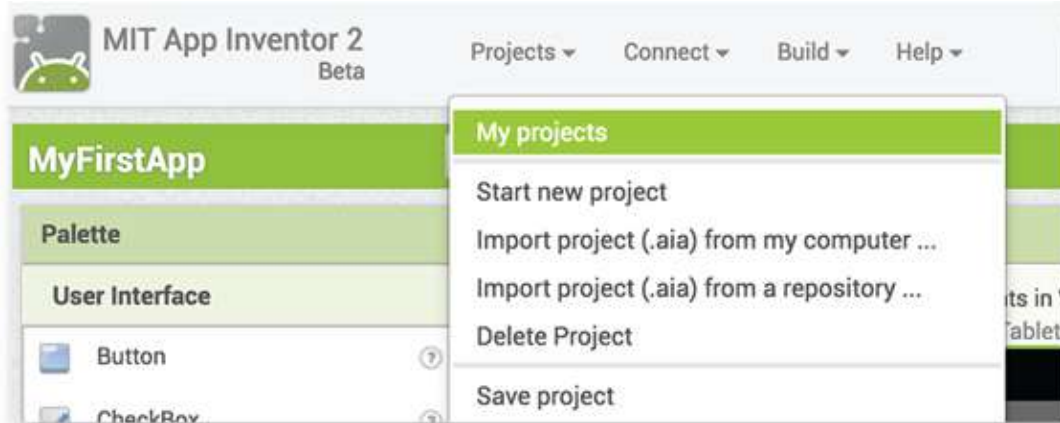
CREATE A NEW APP

First, create your new app. Follow these steps:

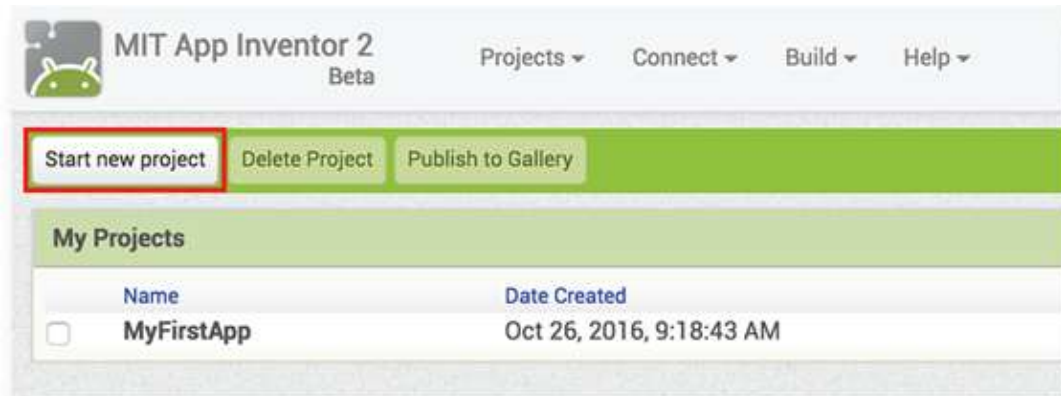
1. Go to <http://ai2.appinventor.mit.edu/>.



Sometimes, when you go back to App Inventor, you are taken to an app you have already started. If that happens, open the Projects menu and choose My Projects to get back to your homepage.



2. Click the Start New Project button.



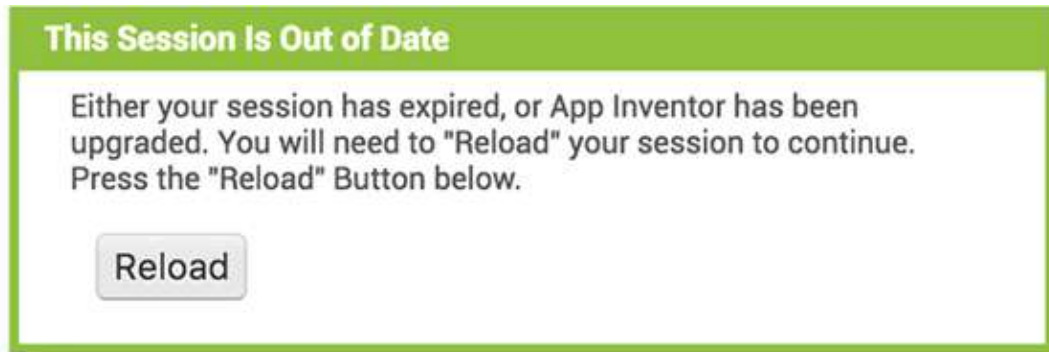
3. Name the project “AboutMe.”

The image shows a dialog box titled 'Create new App Inventor project'. It has a green header bar with the title. Inside the dialog, there is a label 'Project name:' followed by a text input field containing the text 'AboutMe'. At the bottom of the dialog, there are two buttons: 'Cancel' on the left and 'OK' on the right.

4. Congratulations! You are now ready to start designing your screens.



Sometimes App Inventor updates. If you get this warning, just click Reload.

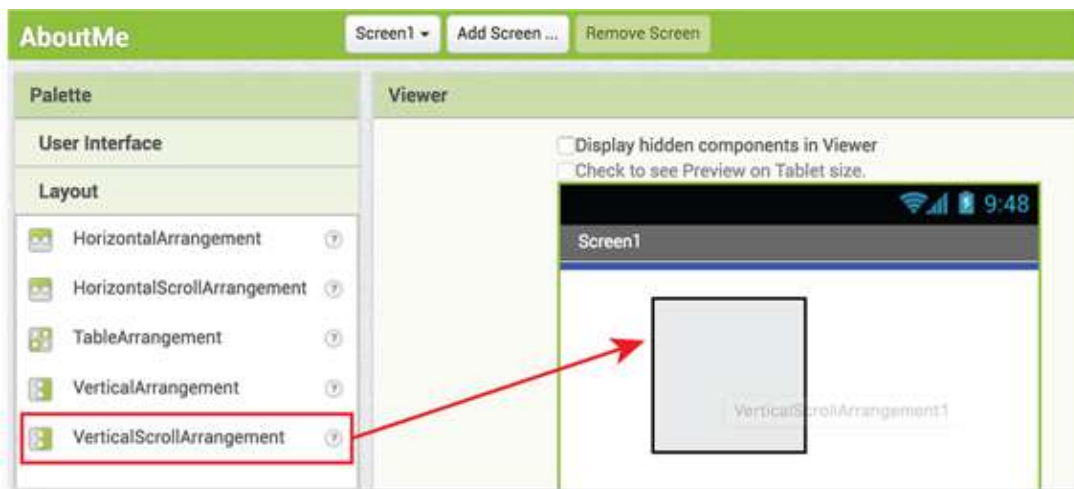


CREATE THE HOMEPAGE

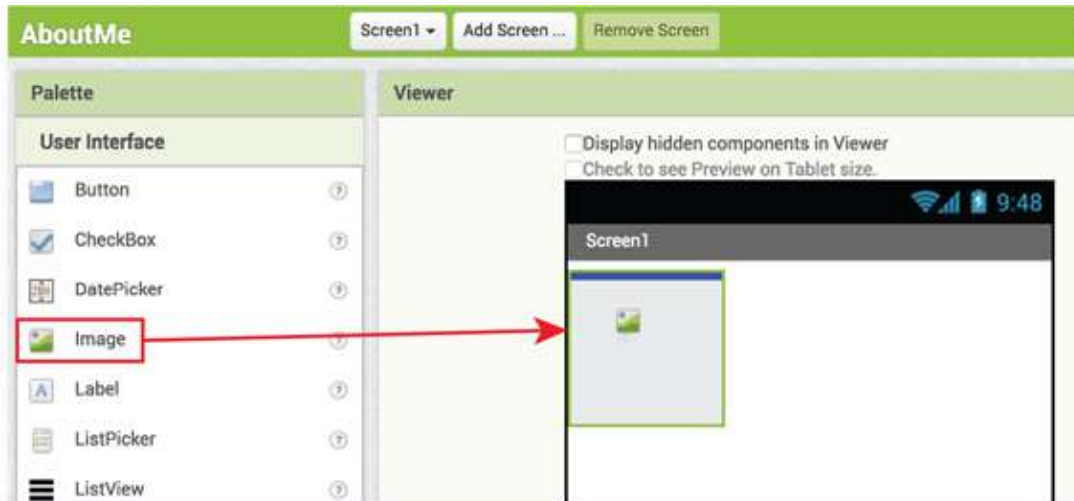
The first page you need to create is the homepage. Follow these steps:

1. **Drag a VerticalScrollArrangement onto your screen. You can find it in the Layout category in the Palette section on the left side of your screen.**

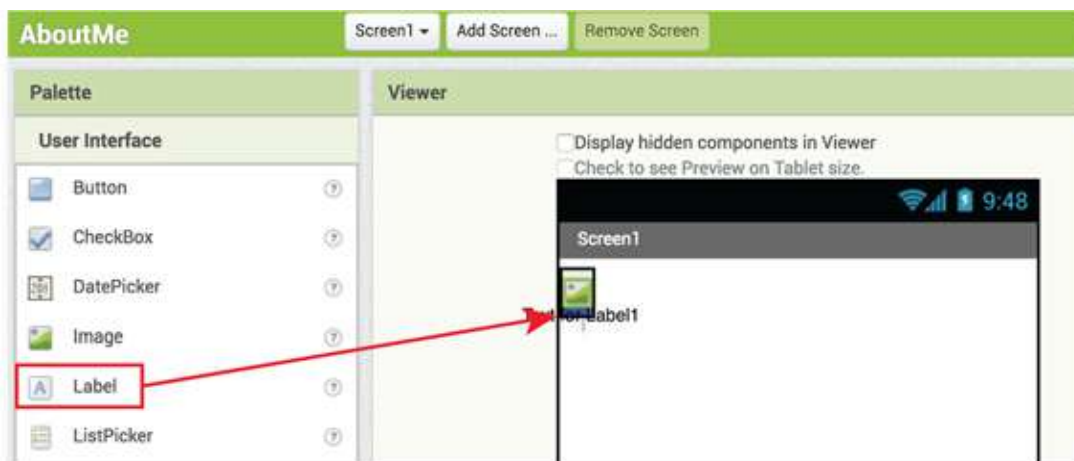
VerticalScrollArrangement means that if you add too many items to fit on the screen, the screen will scroll so that your users can see them all.



2. **In the User Interface section, find Image. Drag an image item into the VerticalScrollArrangement.**



3. In the User Interface section, find Label. Drag the label item into the VerticalScrollArrangement, underneath the image from Step 2.

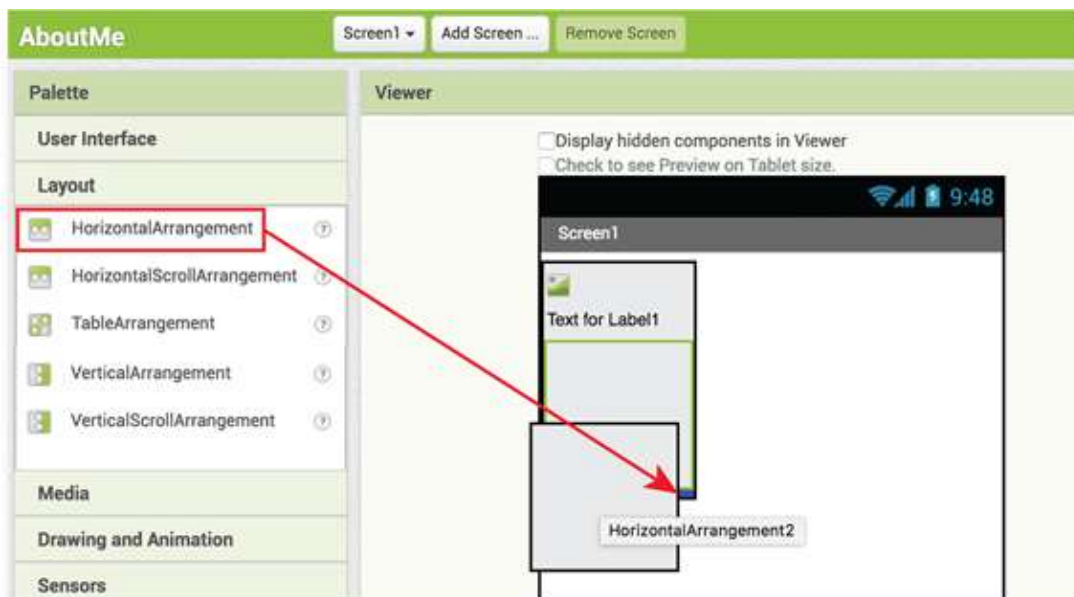


4. In the Layout section, find HorizontalArrangement. Drag a HorizontalArrangement into the VerticalScrollArrangement, underneath the label from Step 3.

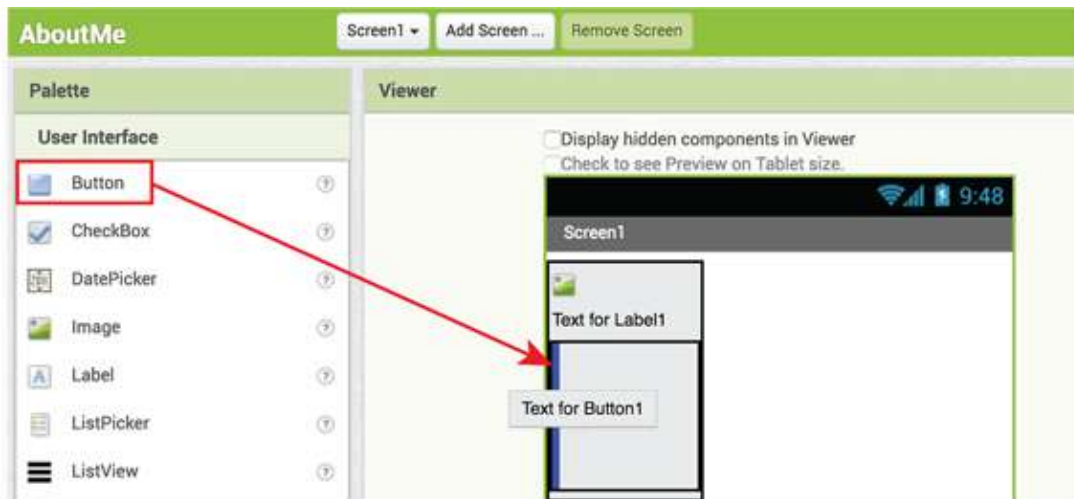


5. Drag a second HorizontalArrangement into the VerticalScrollArrangement.

This should be placed underneath the HorizontalArrangement from Step 4.

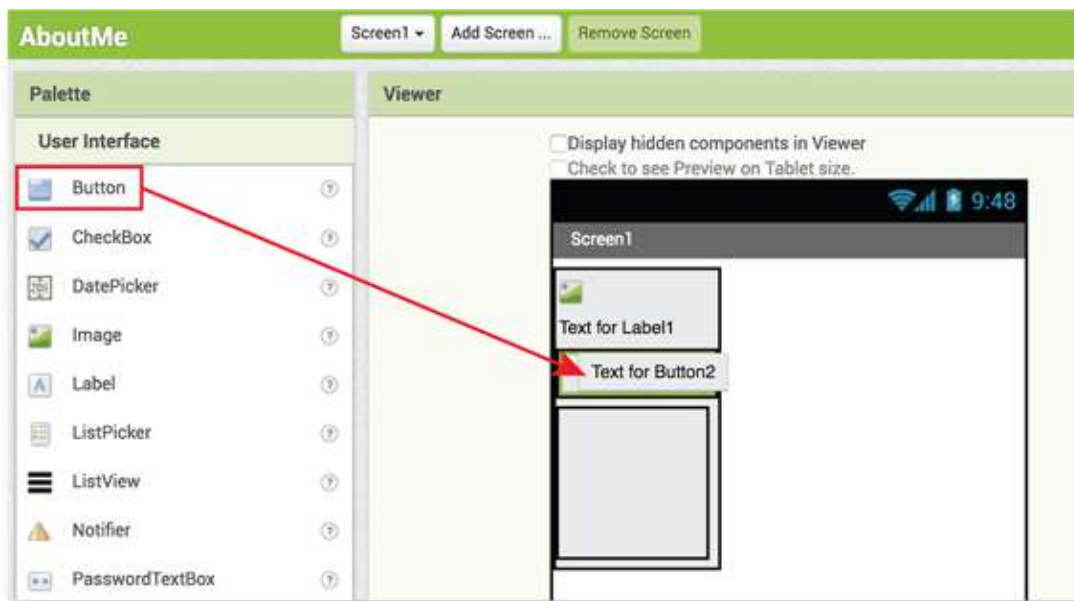


6. In the User Interface section, find Button. Drag a button item into the HorizontalArrangement from Step 4.

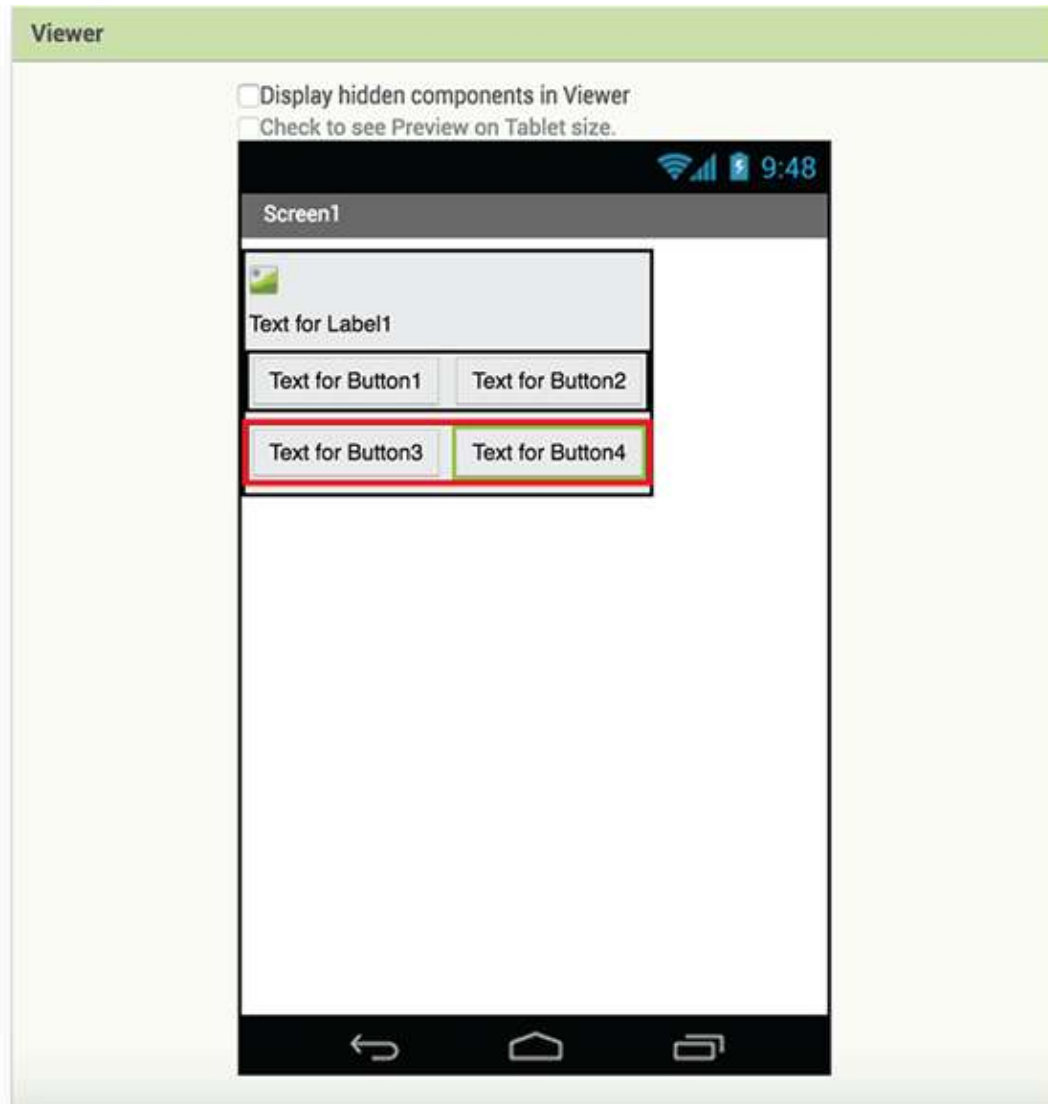


7. Drag a second button into the HorizontalArrangement from Step 4.

This should be placed to the right of the button from Step 6. Drop the button on top of the button from Step 6 to make it go to the right.



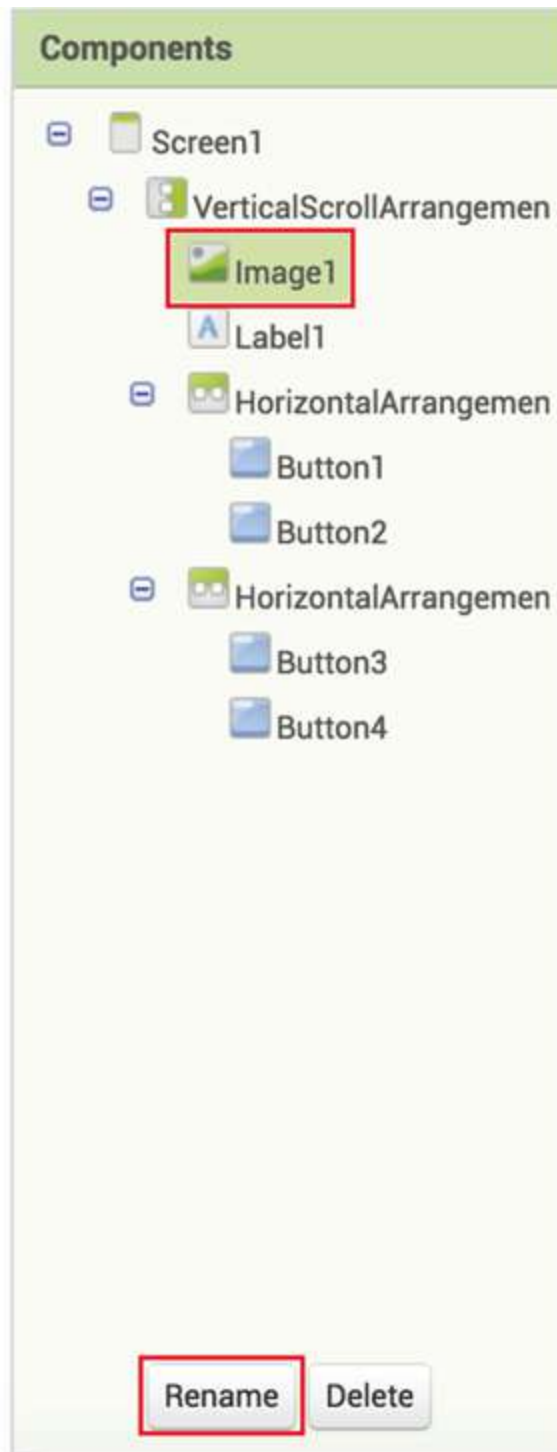
8. Add two more buttons into the HorizontalArrangement from Step 5.



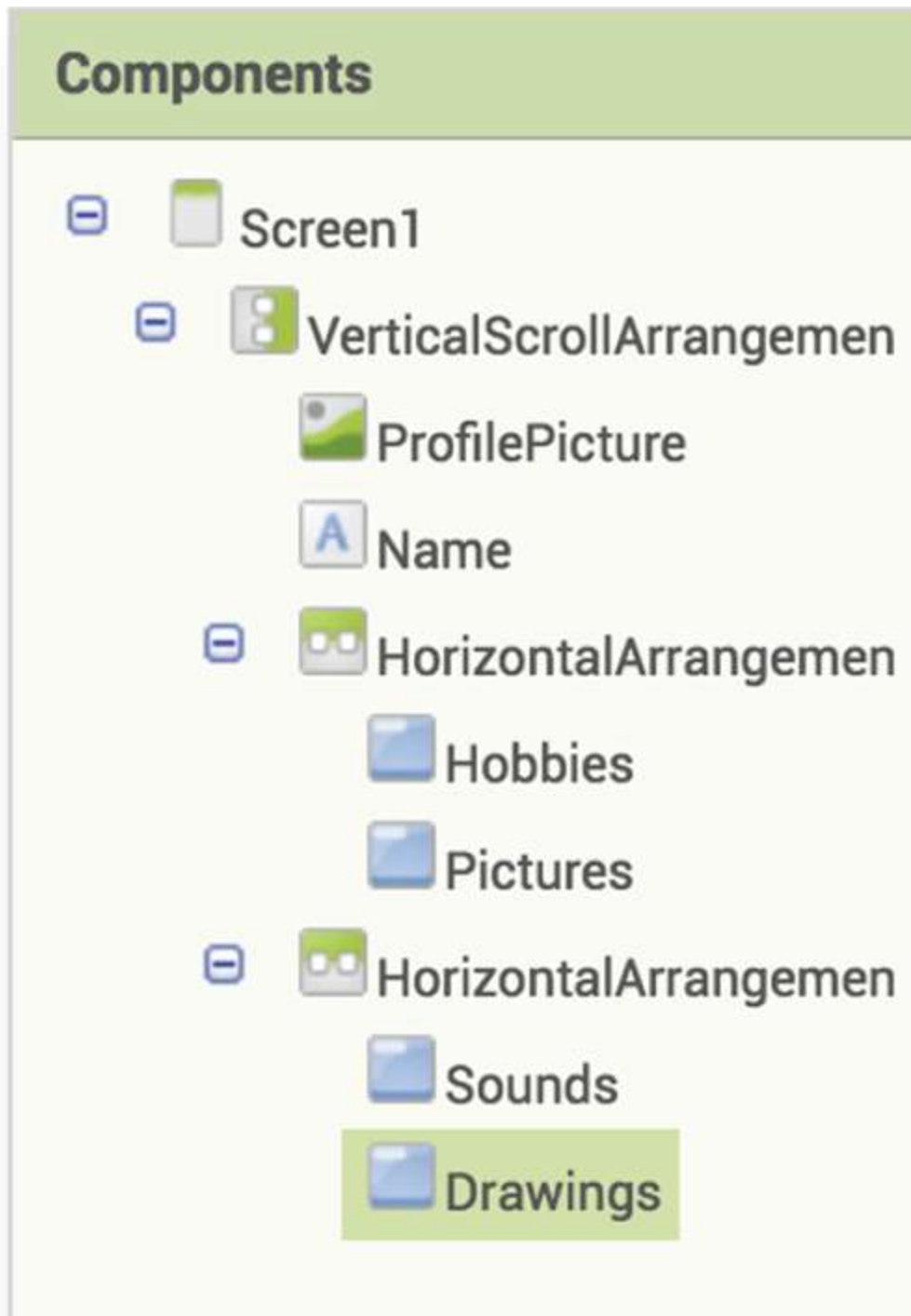
Congratulations! You completed the skeleton to your homepage!

Before moving on to another screen, it is important to name all of your items so that you know which is which. To do so, follow these steps:

1. **To rename each item, click on an item in the Components column and click the Rename button.**

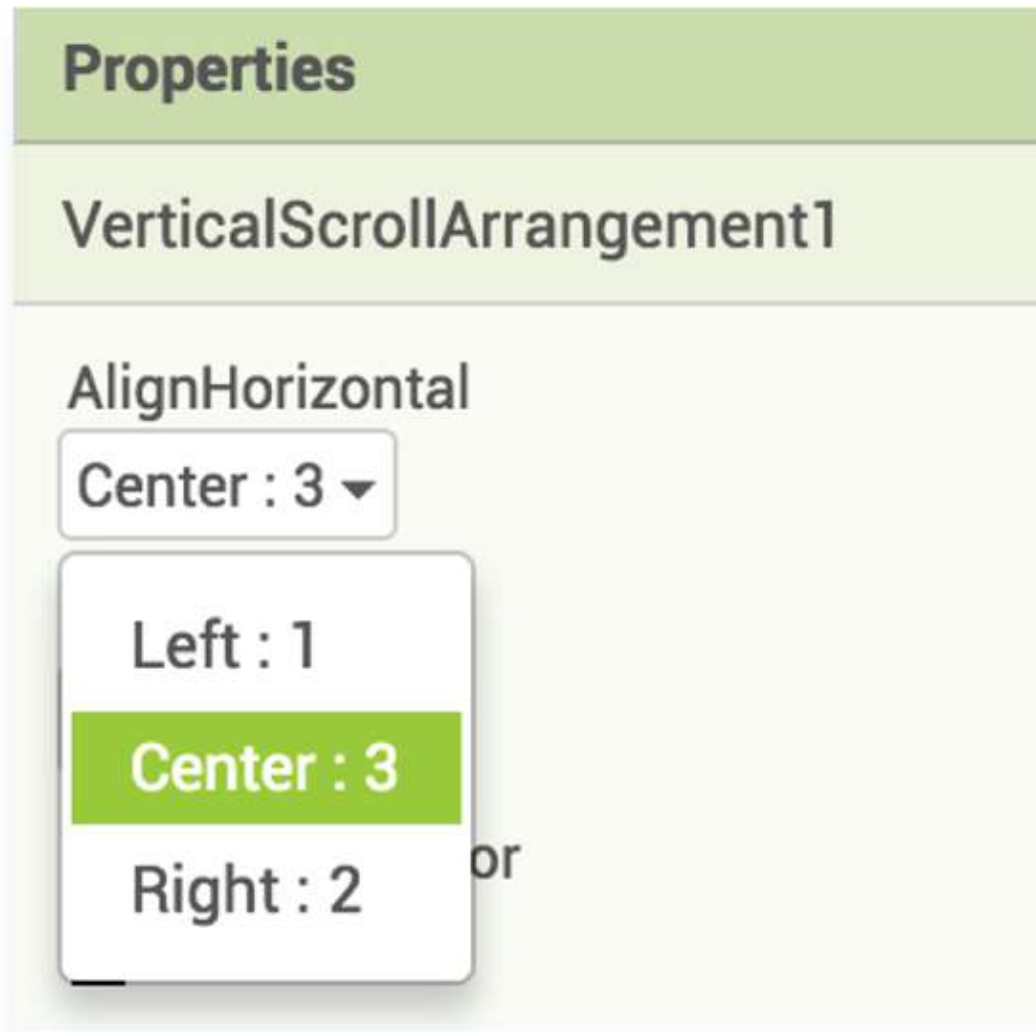


2. **Rename the item to whatever you want and click Save.**
3. **Do this for all items; your completed Component section should look like this:**



Finally, you need to change the properties for each item. Follow these steps:

1. First, fix the VerticalScrollArrangement. Select VerticalScrollArrangement in the Components column, and the Properties column displays the properties for that item. Change the AlignHorizontal property to “Center:3.”



2. Change the Width to “Fill Parent.”
That means that this item will fill the screen.

Width

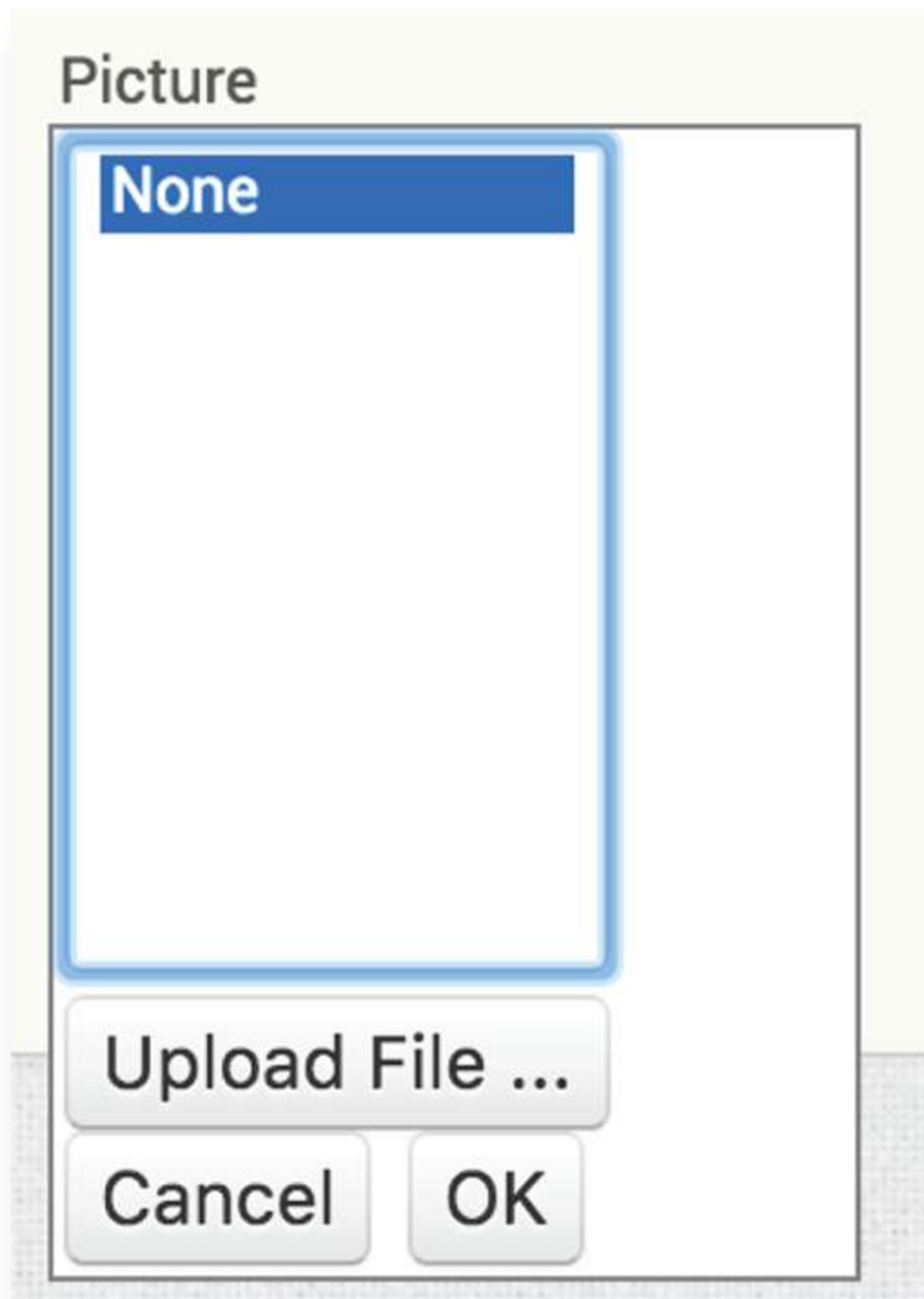
☐ Automatic

☒ Fill parent

☐ pixels

☐ percent

3. Select ProfilePicture and, in the Properties column, choose Picture and Upload File.



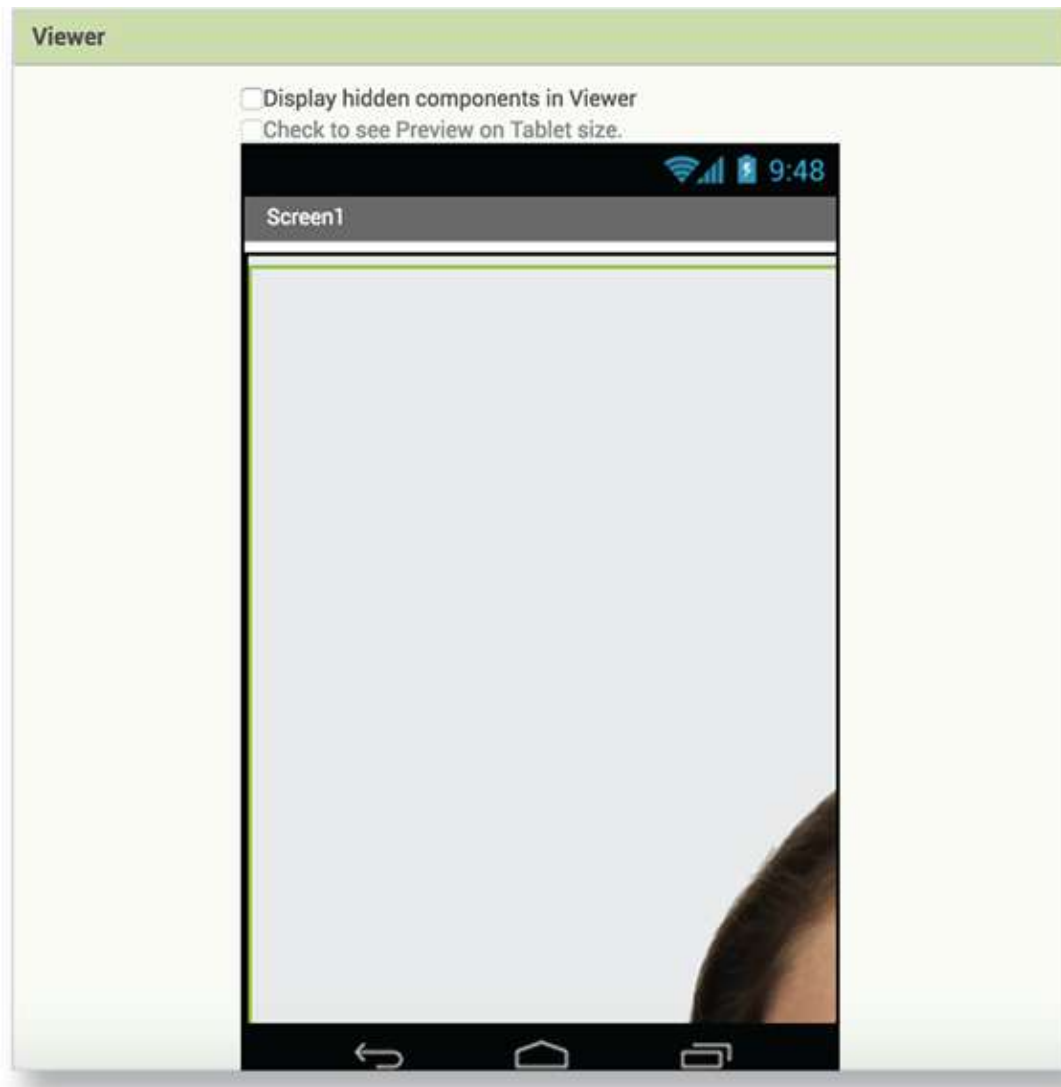
4. Click the Choose File button and choose a picture that you want to use as your profile picture.



5. Click OK.



The picture might be too big!



6. **Change the Height and Width properties to 50 pixels so that the image fits in the screen.**

Height

☐ Automatic

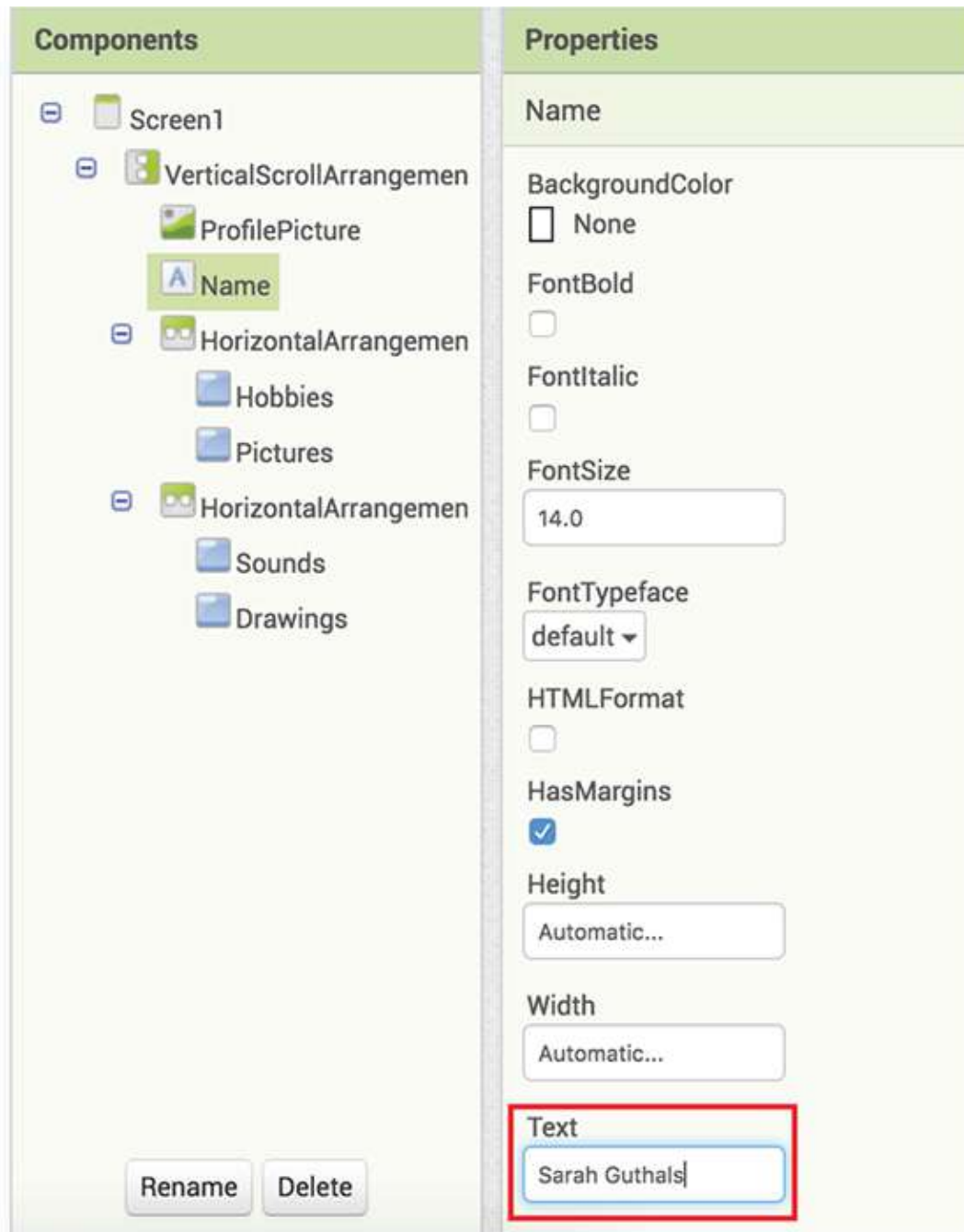
☐ Fill parent

☒ 50 pixels

☐ percent

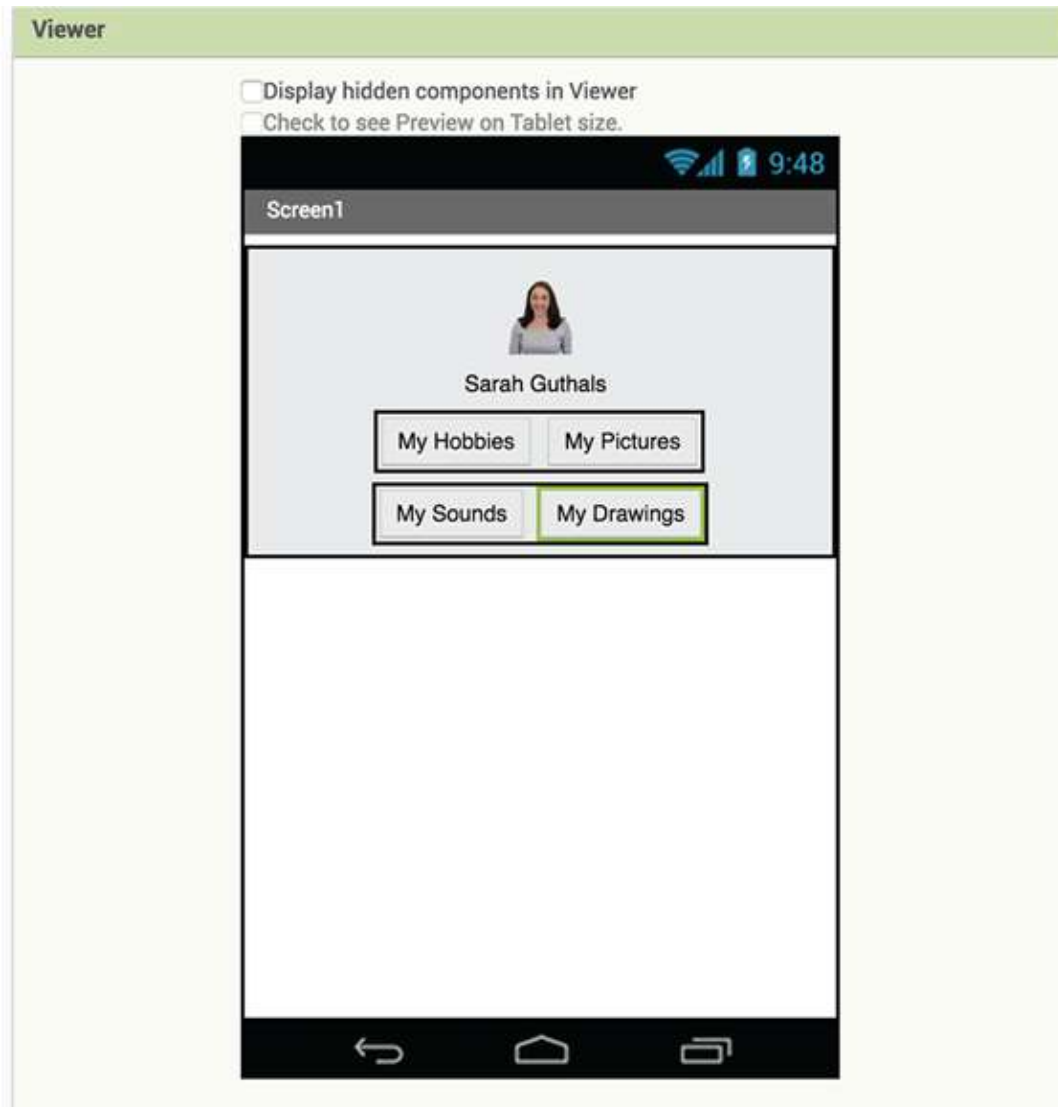
Cancel OK

7. Change the properties for Name by selecting Name in the Components column. Change the “Text” to your name.

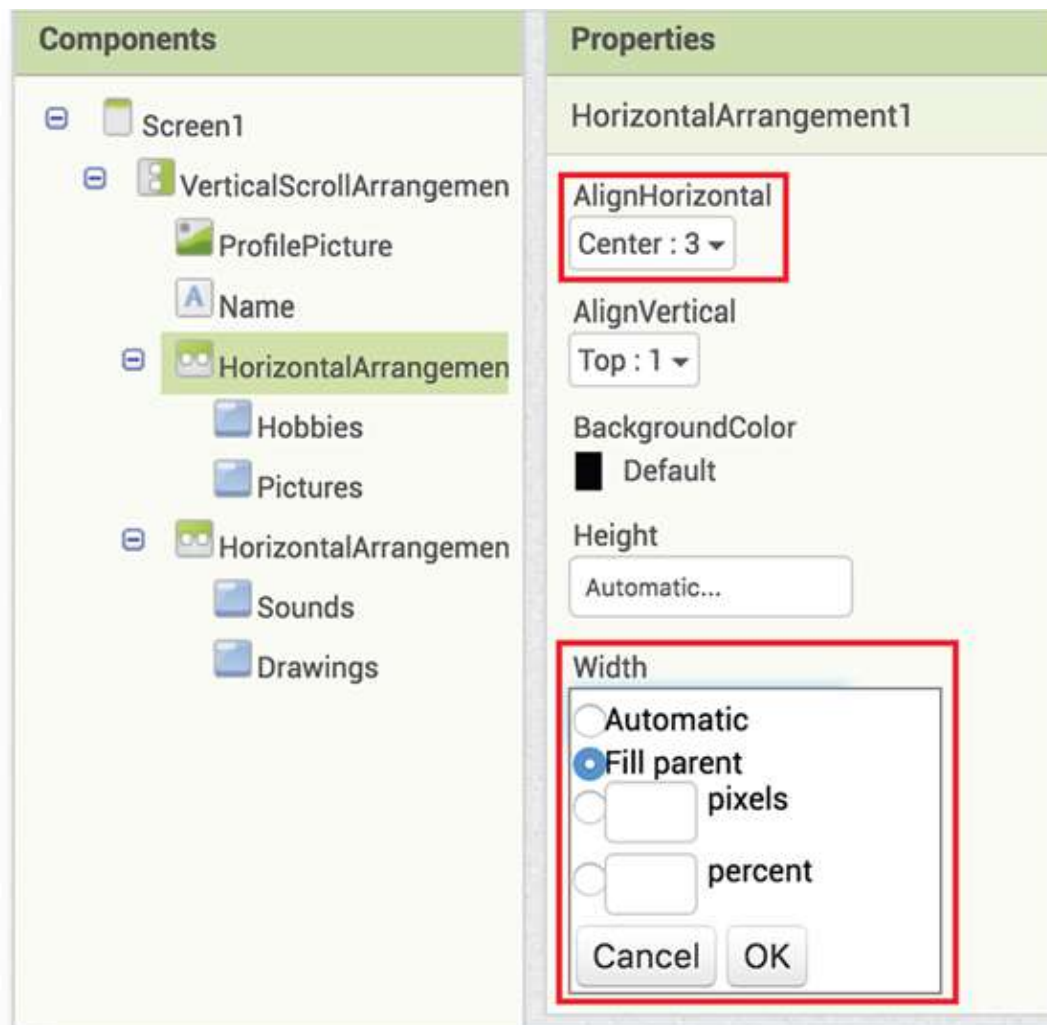


8. **Change the properties for Hobbies. Change the “Text” to “My Hobbies.” Do this by selecting Hobbies in the Components column, and then change the “Text” like you did in Step 7.**
9. **Change the properties for the other three buttons, Pictures, Sounds, and Drawings. Make the “Text” for each of them “My**

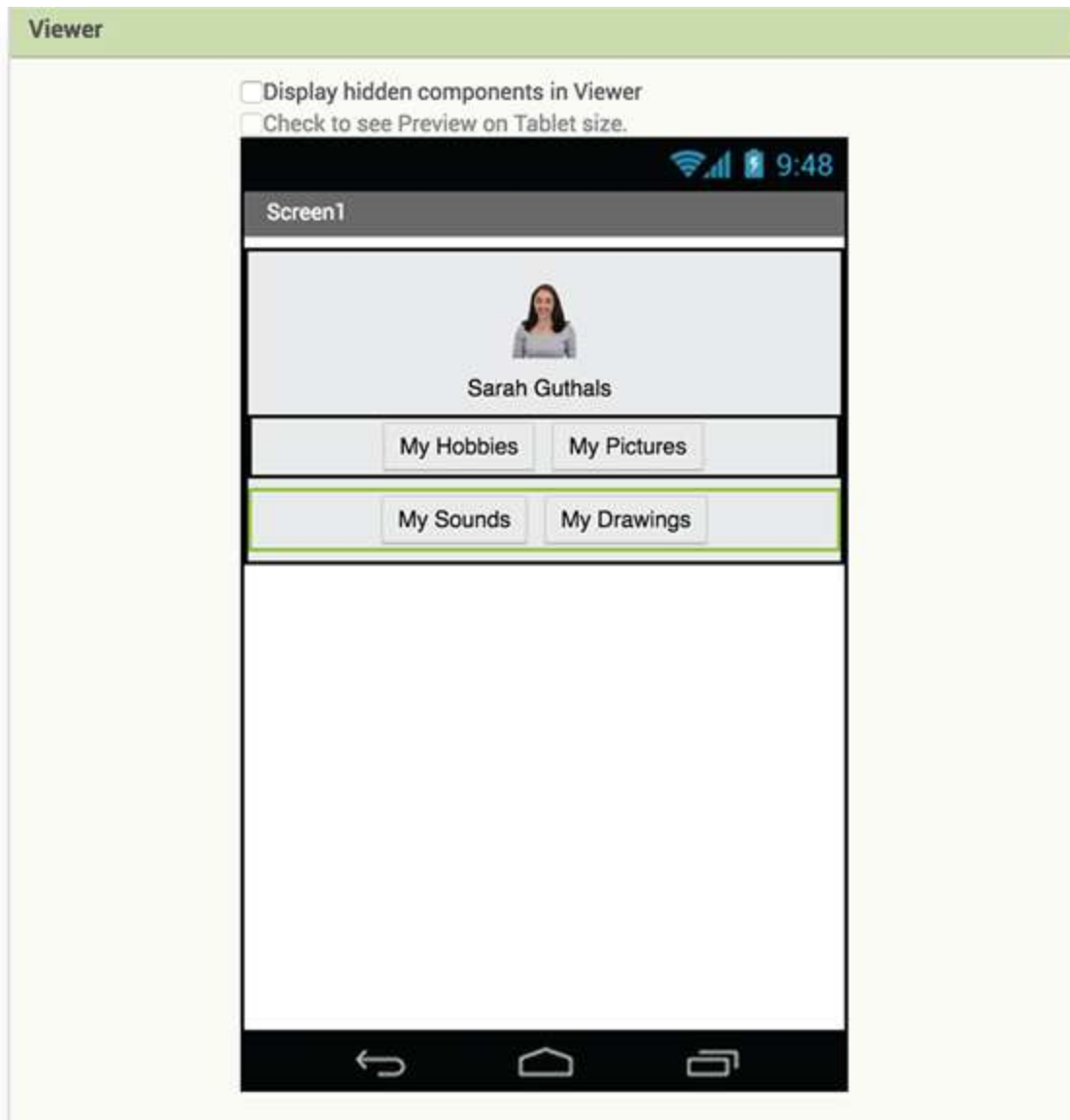
Sounds,” “My Pictures,” and “My Drawings.”



10. **Change the properties for both HorizontalArrangement items so that each has “Center:3” for AlignHorizontal and “Fill Parent” for Width.**



Congratulations! You have your app all planned out!



CREATE YOUR OTHER SCREENS

Now that you have your homepage completed, you can create the other eight screens in the same way! Here are the general rules for creating each screen. Review the previous section for how to add each component.

- 1. Click the Add Screen button.**



2. Name the screen something useful, like “Hobbies.”
3. Start building on it!

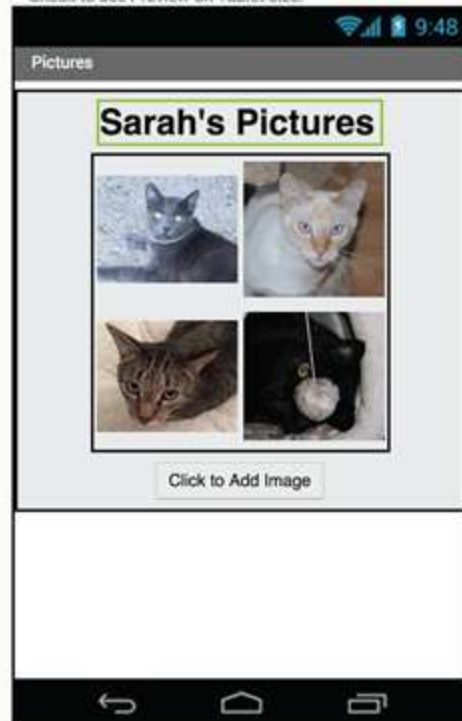


Sometimes as you build your actual app, you think of a better way to design it. That's OK! You can always change your design. For example, I decided that instead of including a separate screen for adding a new hobby, I would just make it so that I could add a hobby on my hobby screen!

Here are the final three screens that I designed for my About Me app:

Viewer

- ☐ Display hidden components in Viewer
☐ Check to see Preview on Tablet size.



Components

- Pictures
 - VerticalScrollArrangemen
 - MyPictures
 - TableArrangement1
 - Lewis
 - Clark
 - Luke
 - Princess
 - PicturePicker

Rename Delete

Media

Viewer

Display hidden components in Viewer

Check to see Preview on Tablet size.

Sounds

Sarah's Sounds

Play Bell Sound

Play Phone Sound

Non-visible components

Bell

Phone

Components

Sounds

VerticalScrollArrangemen

MySounds

PlayBellSound

PlayPhoneSound

Bell

Phone

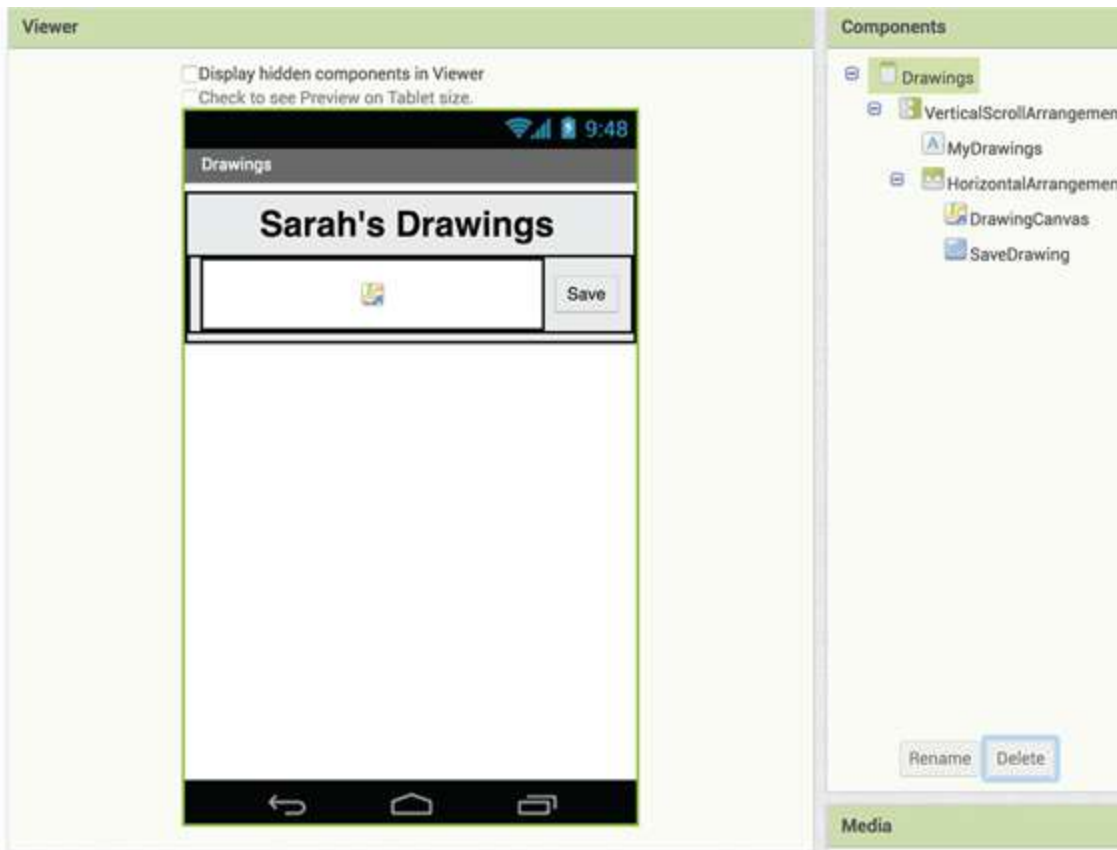
Rename

Delete

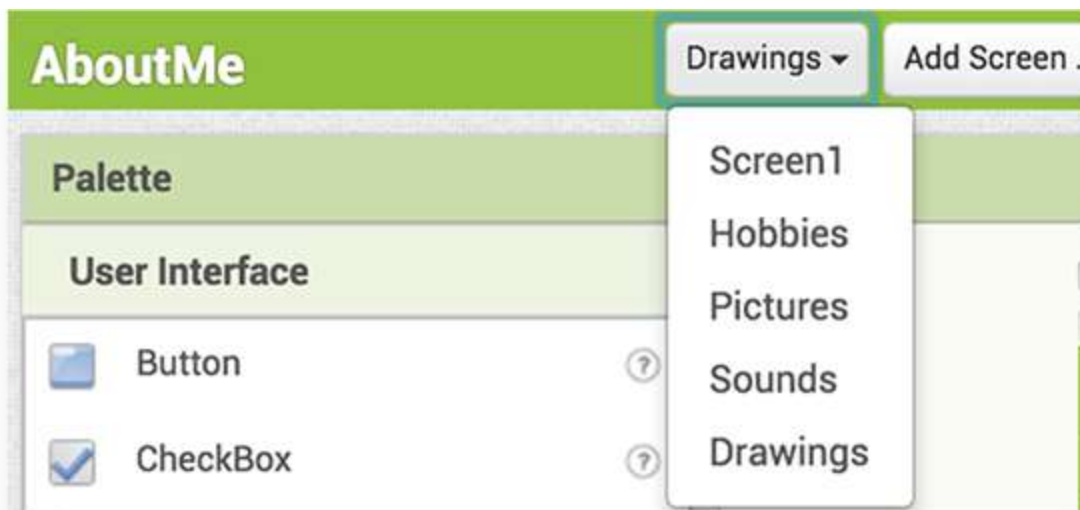
Media

Sarah_Guthals.png

Lewis.jpg



You should now have five total screens:



CODE YOUR APP

Now it's time to add code to your app! This is where it gets to be really fun! First, you code all the buttons to be able to see all of the screens. Then, in the next project, you code to add new things to each screen!

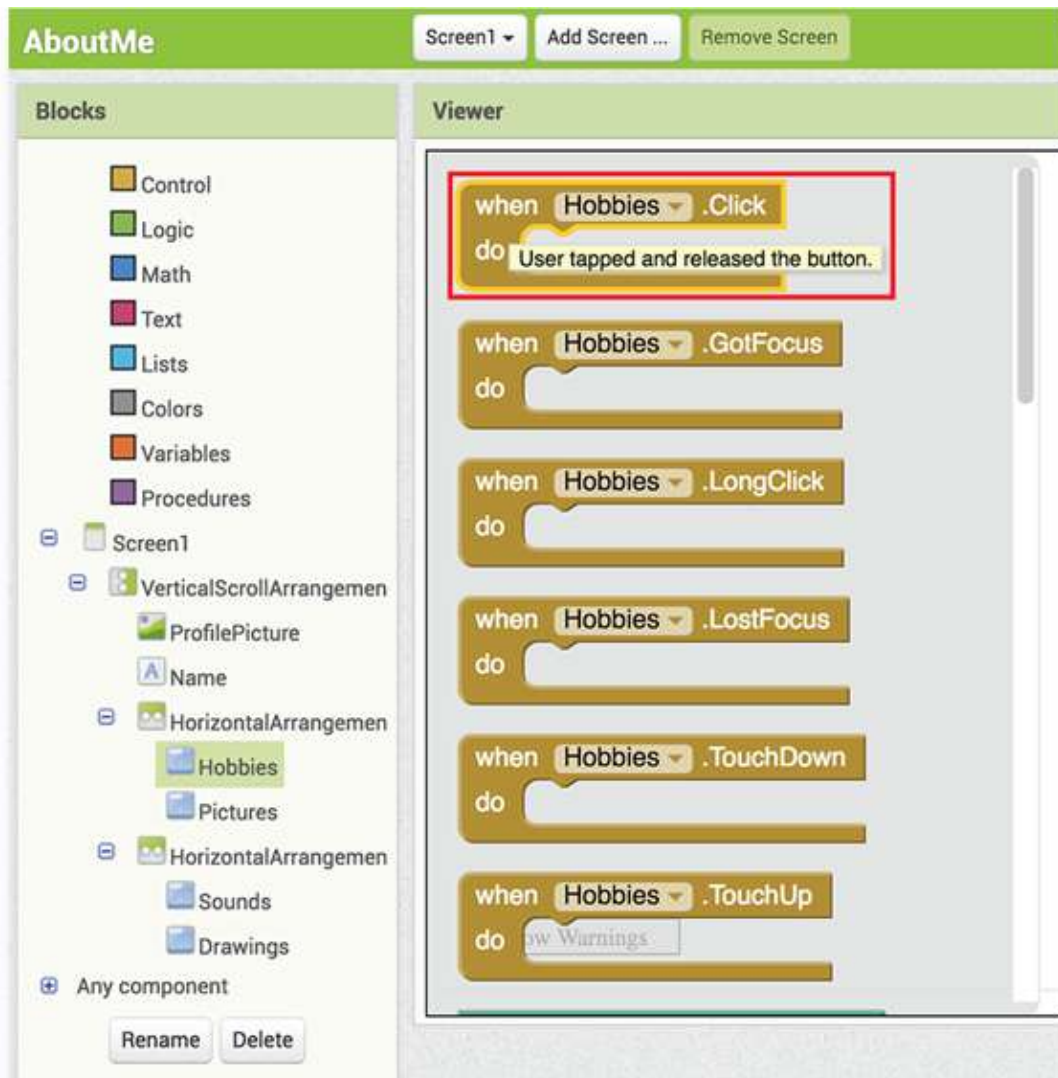
CODE UP THE BUTTONS

To get into the coding blocks, click the Blocks button. Then, follow these steps:

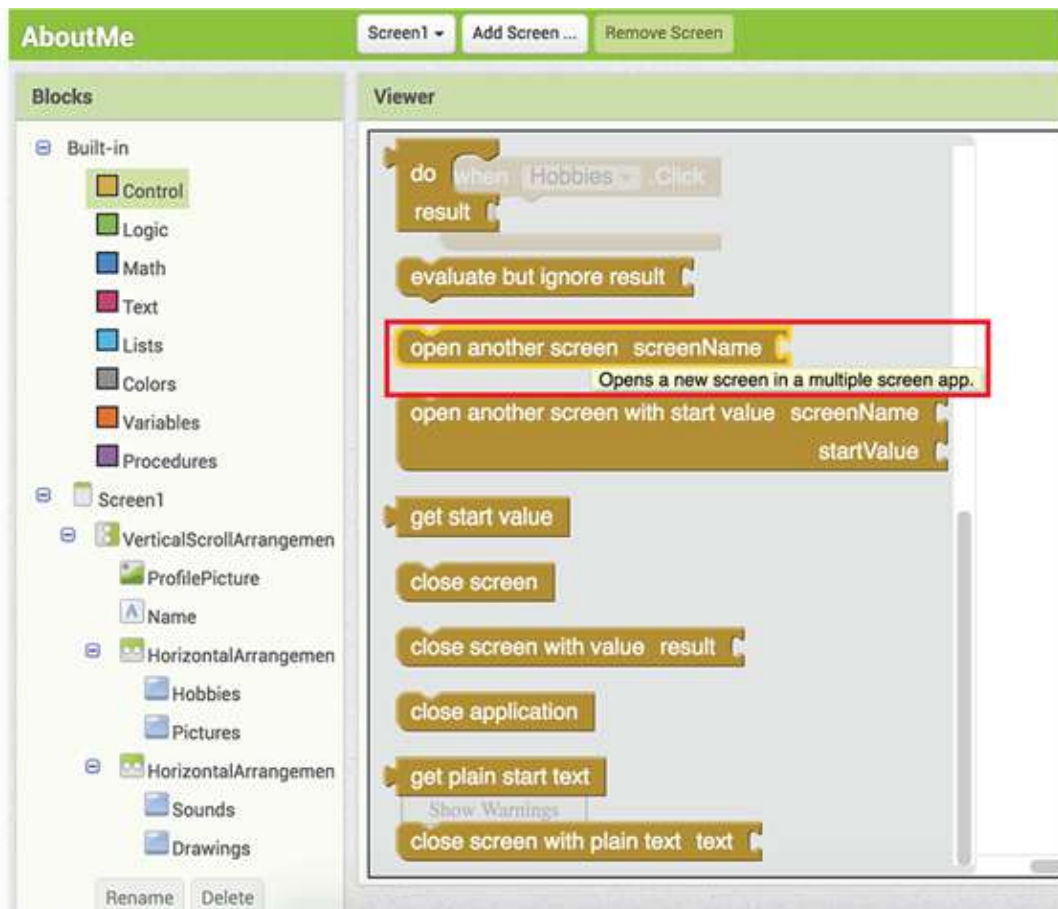


Make sure you are on “Screen1” so that you can code the homepage buttons first. That way you can get to the other screens when you test your app.

1. **Select Hobbies in the Blocks column and drag a When Hobbies.Click Do into the viewer.**



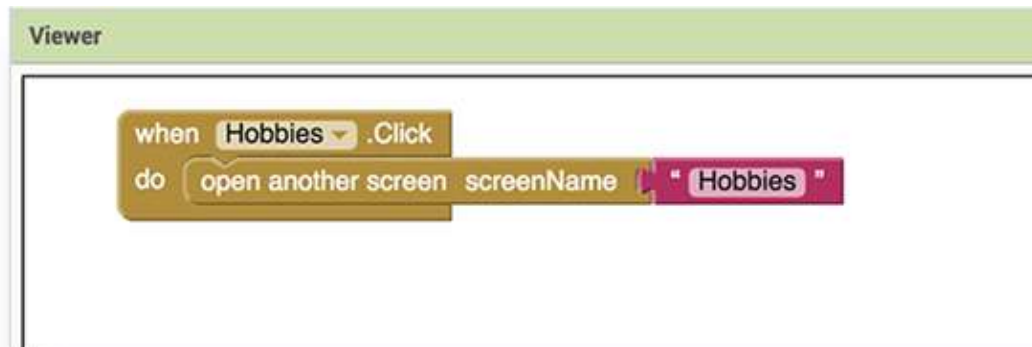
2. Select Control in the Blocks column and drag an Open Another Screen ScreenName into the block you added in Step 1.



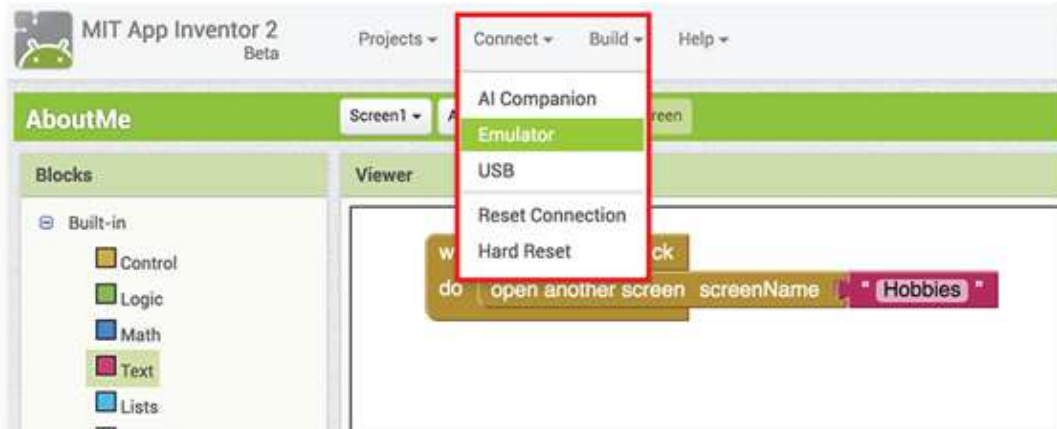
3. Select Text in the Blocks column and drag an empty text block onto the end of the block from Step 2.



Your code should look like this:



4. Click open the Connect menu and choose Emulator. Let the emulator start up on your computer.



Sometimes the emulator doesn't work the first time. If that happens, just close it and try again.

When your app loads up, you will see your homepage.



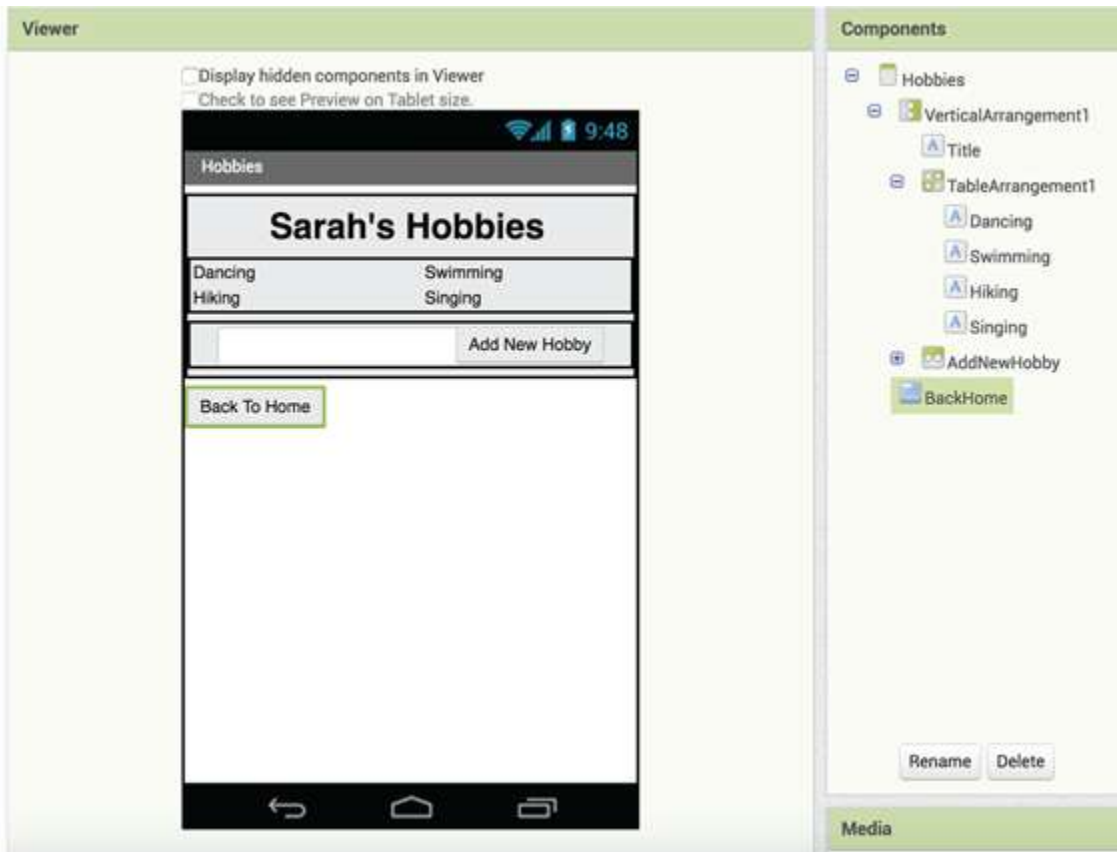
5. Click Hobbies; your hobby page should load up!



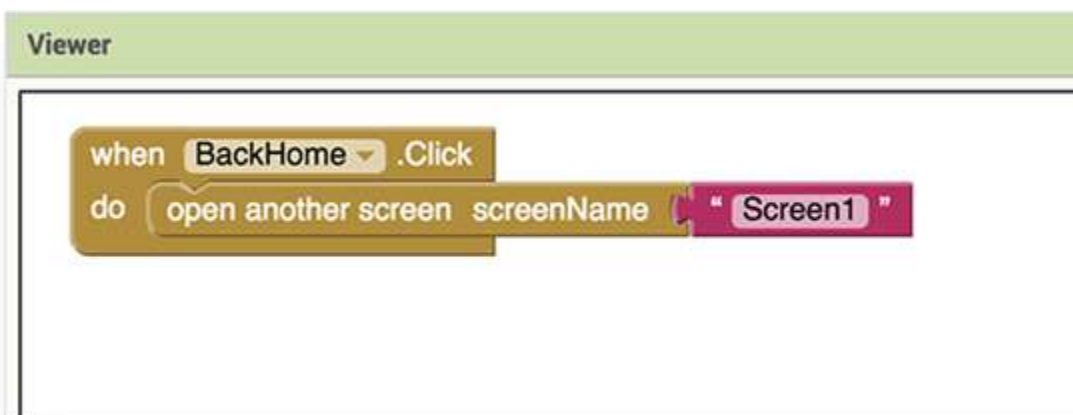
MAKE SIMPLE CHANGES

You might have noticed, when testing your hobby button, that you don't have a way to get back to your homepage. Don't worry! You can add a

“Go Back Home” button to each page.



And you can code them all the same.



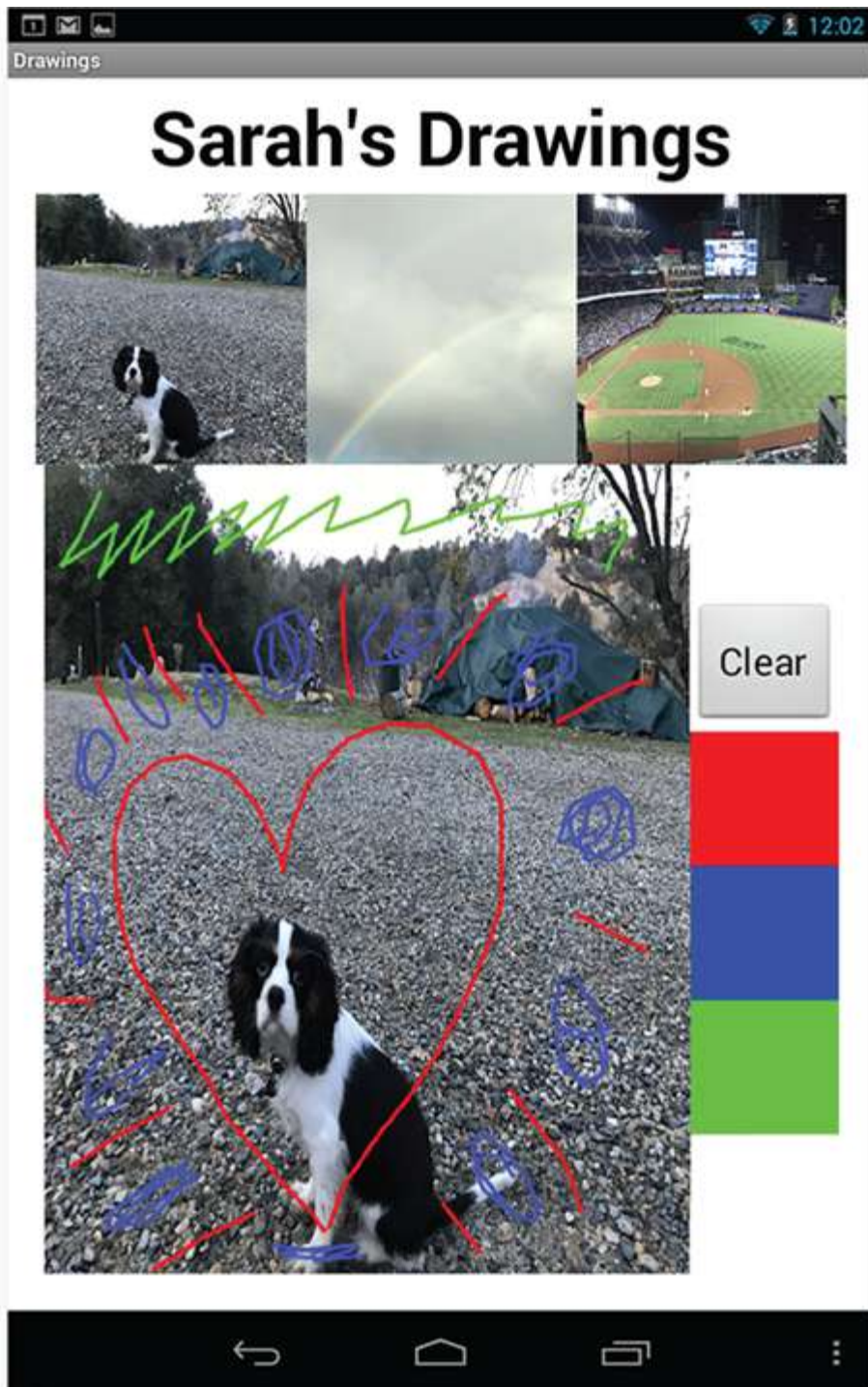
ADD CODE TO ALL YOUR BUTTONS

Now you can add code to all of your buttons. Don't forget to test every time you make a coding change.

In the next project, you learn how to add more features, like being able to do drawings on pictures!

PROJECT 3

MAKE A PHOTO-EDITING APP



IN THIS PROJECT, YOU LEARN HOW TO ADD PICTURES TO YOUR APP AND HOW TO EDIT THEM BY DRAWING OVER

THEM. You first learn how to add more pictures to your app, then how to draw on a canvas, and finally, how to draw on your pictures!

SET UP YOUR PHOTO-EDITING APP

For your photo-editing app you have two options — you can continue with the About Me app that you built in [Project 2](#), or you can start a new app.

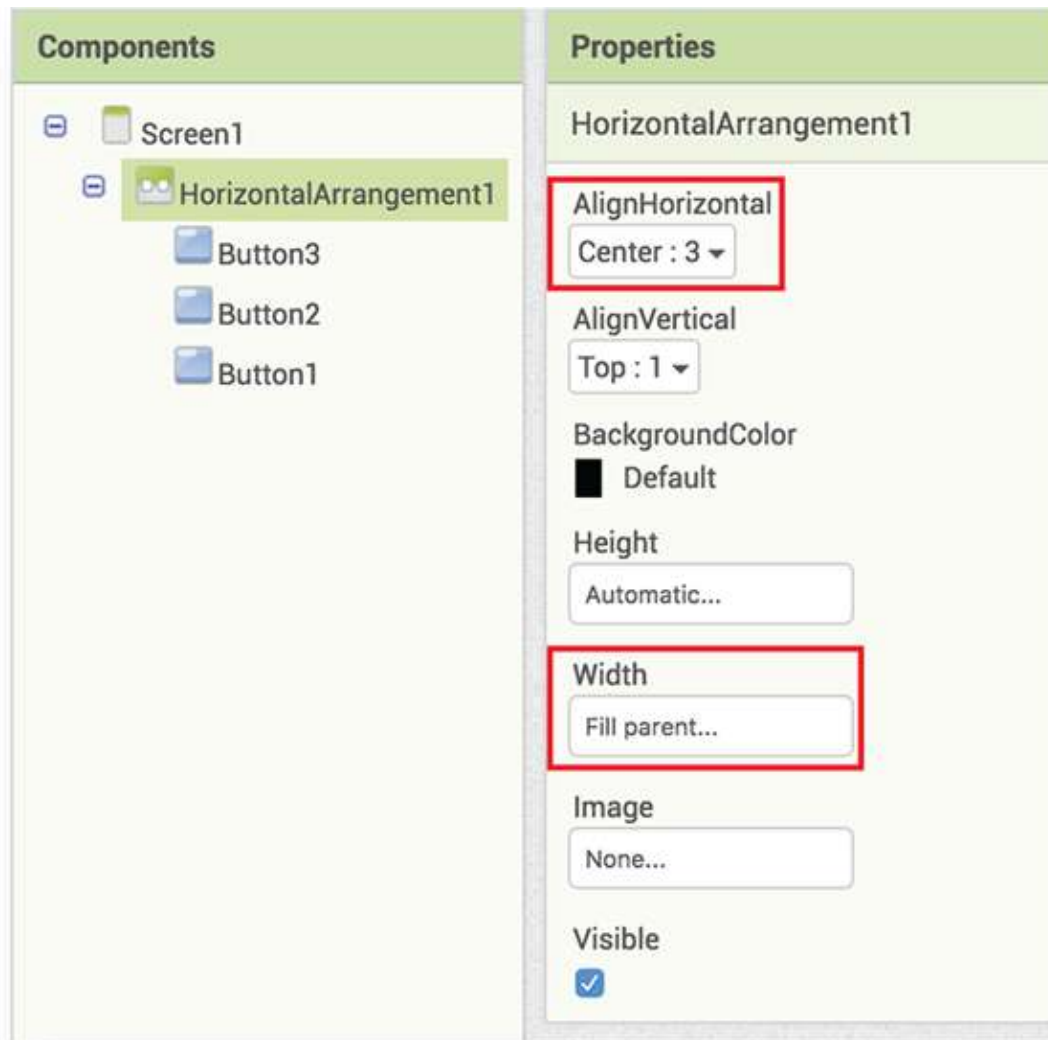
START WITH A NEW APP

You can make an app that is just a photo-editing app, instead of adding on to your About Me app from [Project 2](#). To do so, follow these steps:

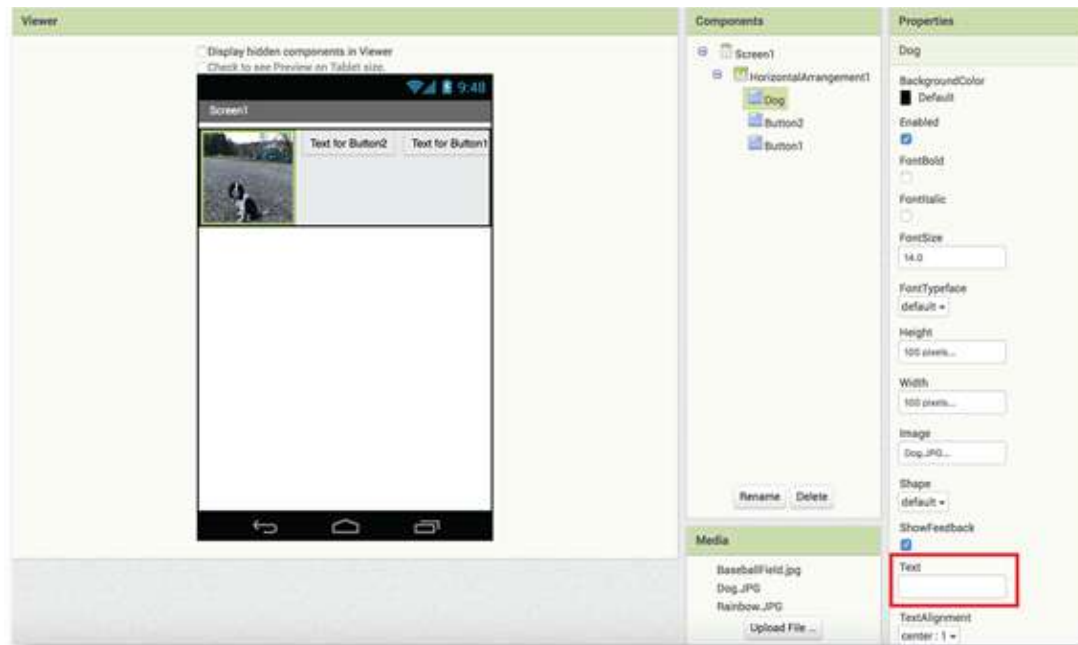
1. **Go to appinventor.mit.edu and log in. Then click My Projects ⇒ Start New Project and name it “Photo Editor.”**
2. **Add a HorizontalArrangement from the Layouts category, and put three buttons from the User Interface category inside the HorizontalArrangement.**



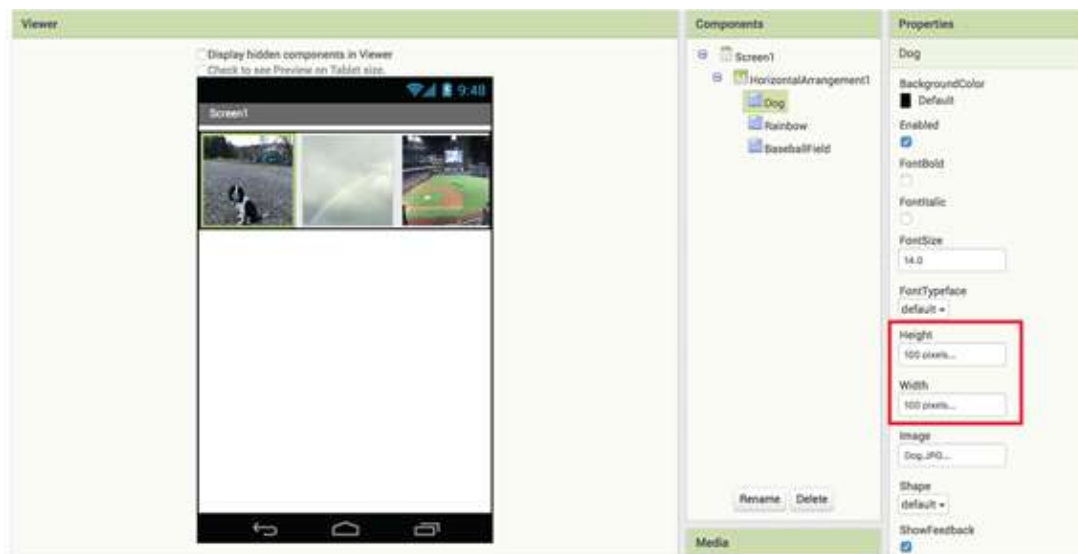
3. **Click HorizontalArrangement under Components and change the AlignHorizontal property to Center: 3 and the Width property to Fill Parent.**



4. Find three of your own pictures that you have saved onto your computer, or download the three pictures found on www.thewecan.zone/mobile-apps.
5. Set each of your Button components to have one image and rename them appropriately. Make sure you remove the text from the buttons, too, so that there isn't text over the image.



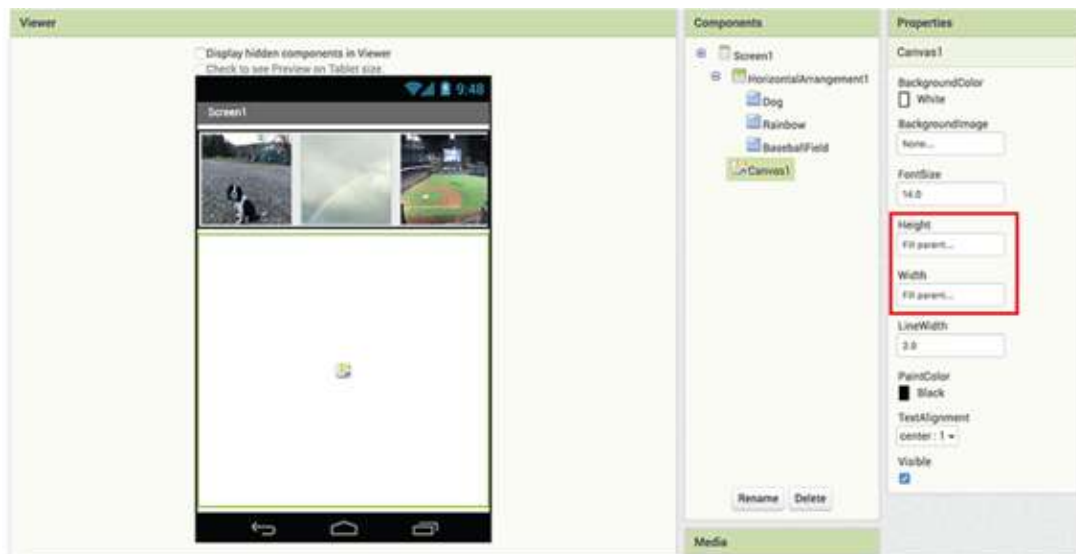
6. Change the Width and Height for each button to be 100 pixels each.



7. In the Palette column, find Drawing and Animation and, from there, drag a Canvas under your images.



8. Change the Width and Height of your Canvas to Fill Parent.



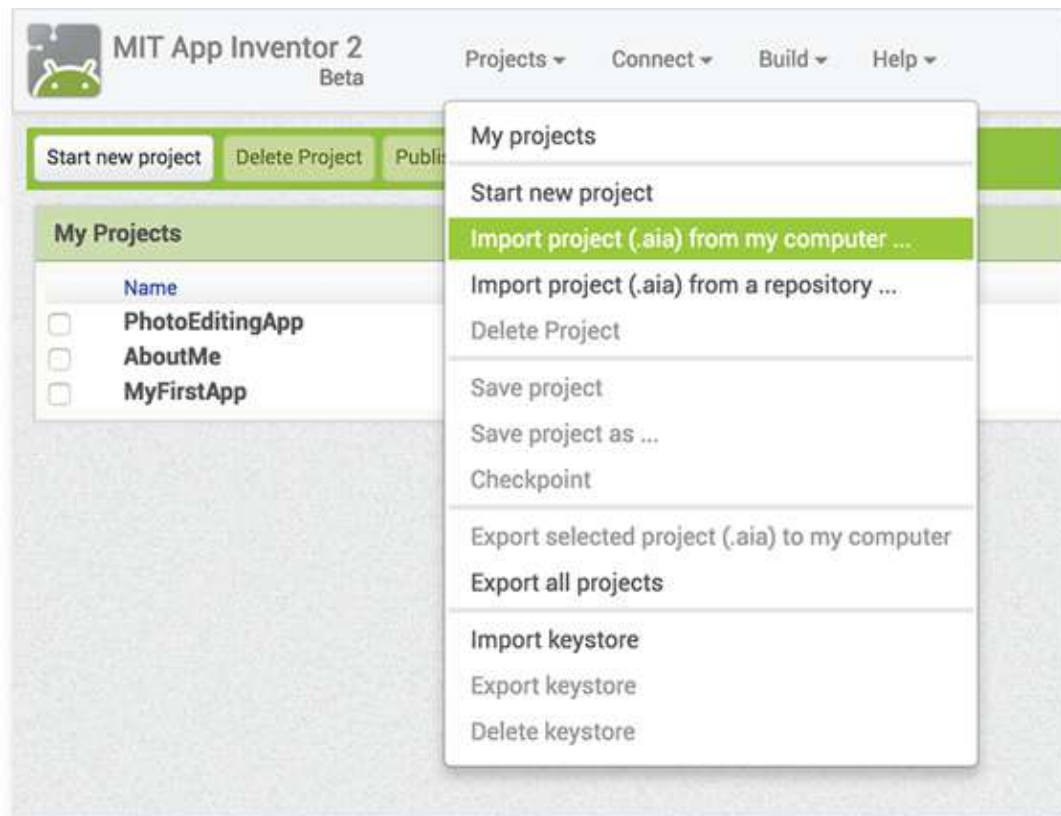
Congratulations! You are now ready to test and code your app. You can skip the next two sections, “[Start with an Already-Made About Me App](#)” and “[Start with Your About Me App](#)” and jump straight to “[Test Your App](#).”

START WITH AN ALREADY-MADE ABOUT ME APP

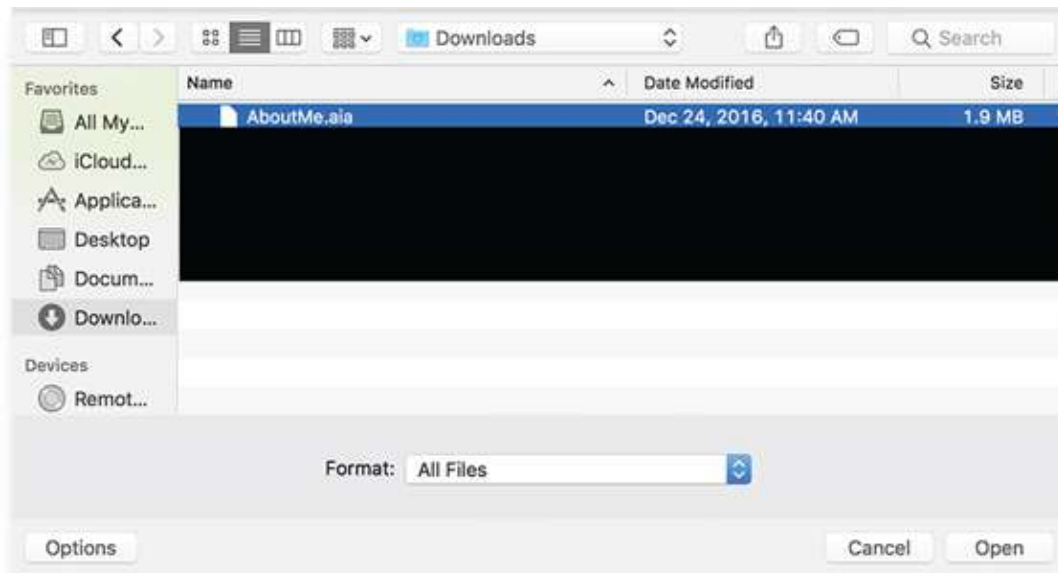
If you didn't make an About Me app in [Project 2](#), and you didn't want to start with a separate photo-editing app as described in the section above, “[Start with a New App](#),” then you have come to the right place!

You can use the About Me app that I made when I was writing [Project 2](#)!

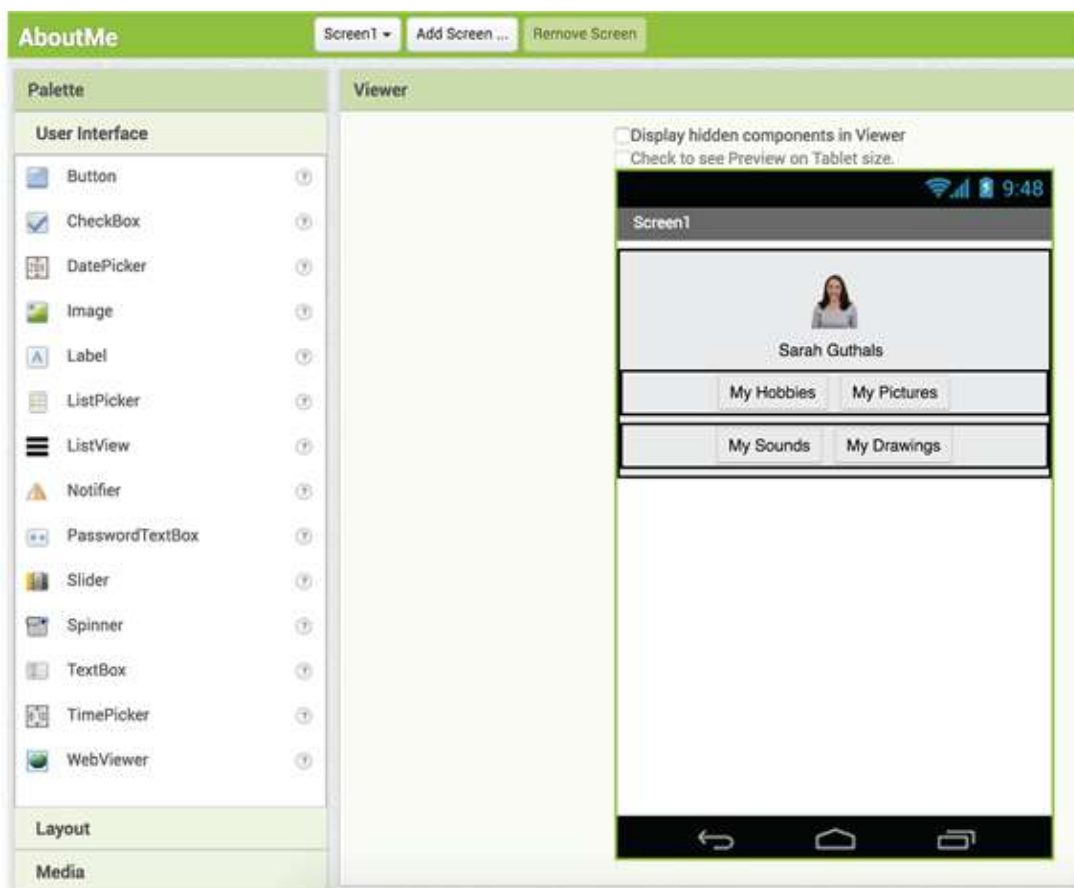
1. **Just go to www.thewecan.zone/mobile-apps and download the file called AboutMe.aia.**
2. **Then, go to appinventor.mit.edu and click the Projects button, then click Import Project (.aia) from My Computer.**



3. **Click Choose File, find the AboutMe.aia file that you downloaded in Step 1, and click OK.**



You should see my About Me app!





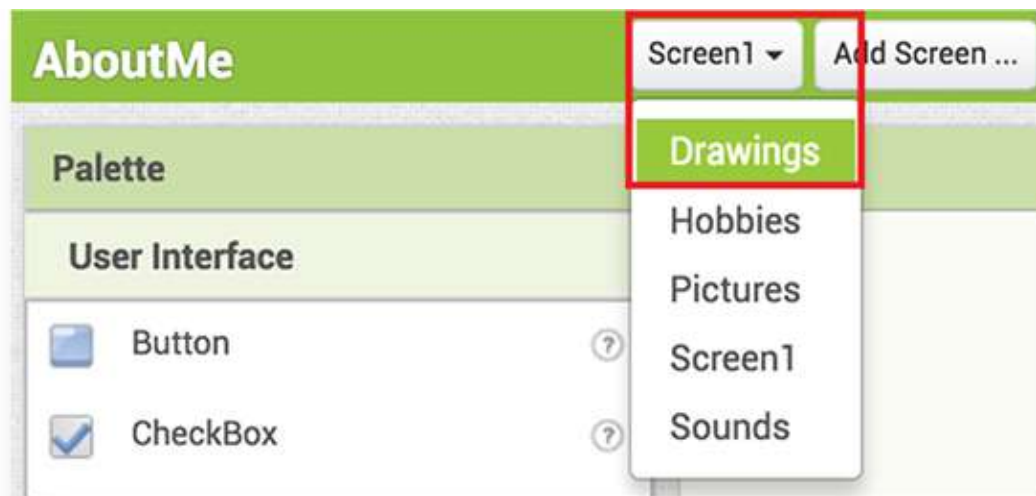
If you already created an app called “About Me,” you won’t be able to import my app because it has the same title. This is an easy fix though; just rename my About Me app to “SarahAboutMe,” or any other name you can think of.

Now that you have my About Me app, you can follow the instructions in the next section.

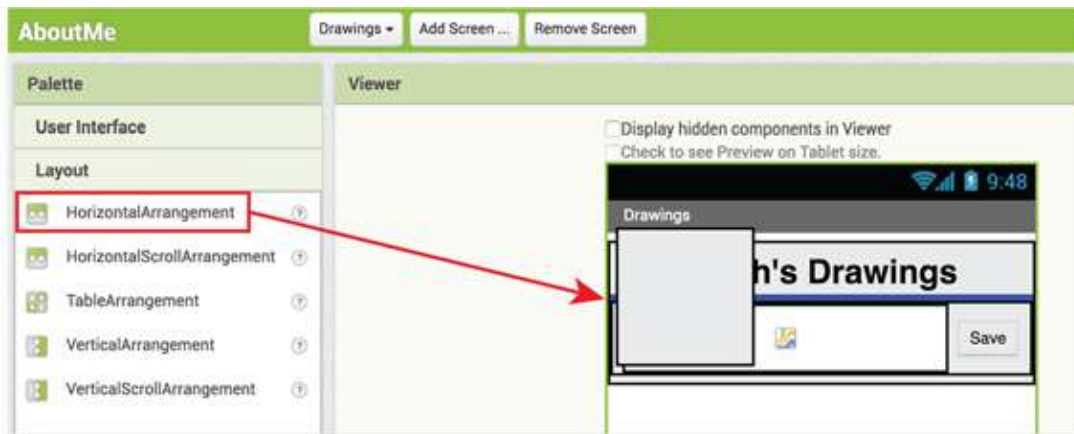
START WITH YOUR ABOUT ME APP

In this section, you learn how to get your About Me app ready to have a photo-editing feature!

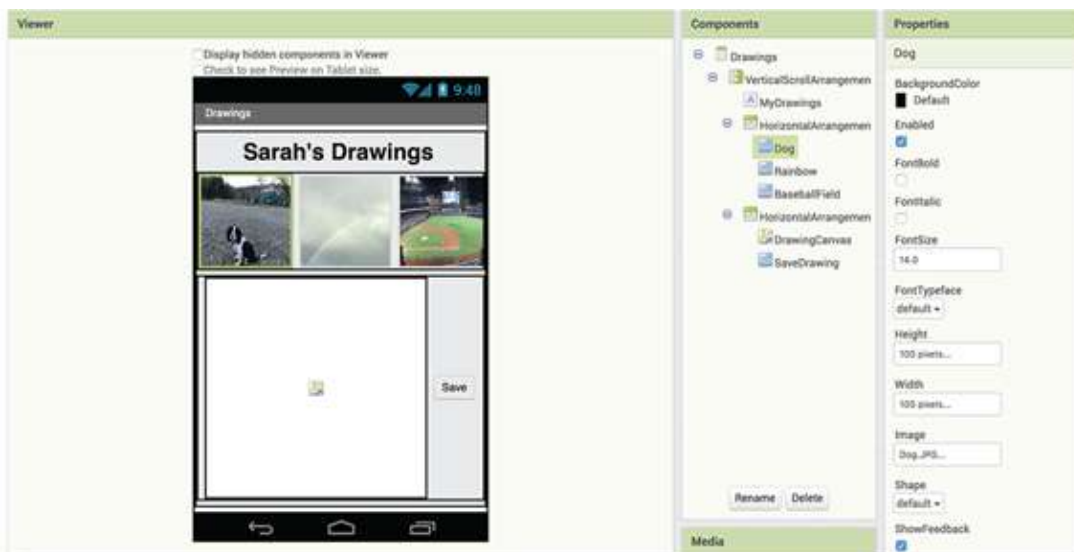
1. **Go to appinventor.mit.edu and open your About Me app, then go to your Drawings screen.**



2. **Under Components, change the height of VerticalArrangement, HorizontalArrangement, and Drawing Canvas to Fill Parent.**
3. **Add a second HorizontalArrangement above the HorizontalArrangement that has your Canvas and Save button in it.**



4. Go back to the section “[Start with a New App](#),” above, and follow Steps 3 through 7, and you should have three images to choose from above your Canvas.



Congratulations! You are ready to test and code your photo-editing feature.

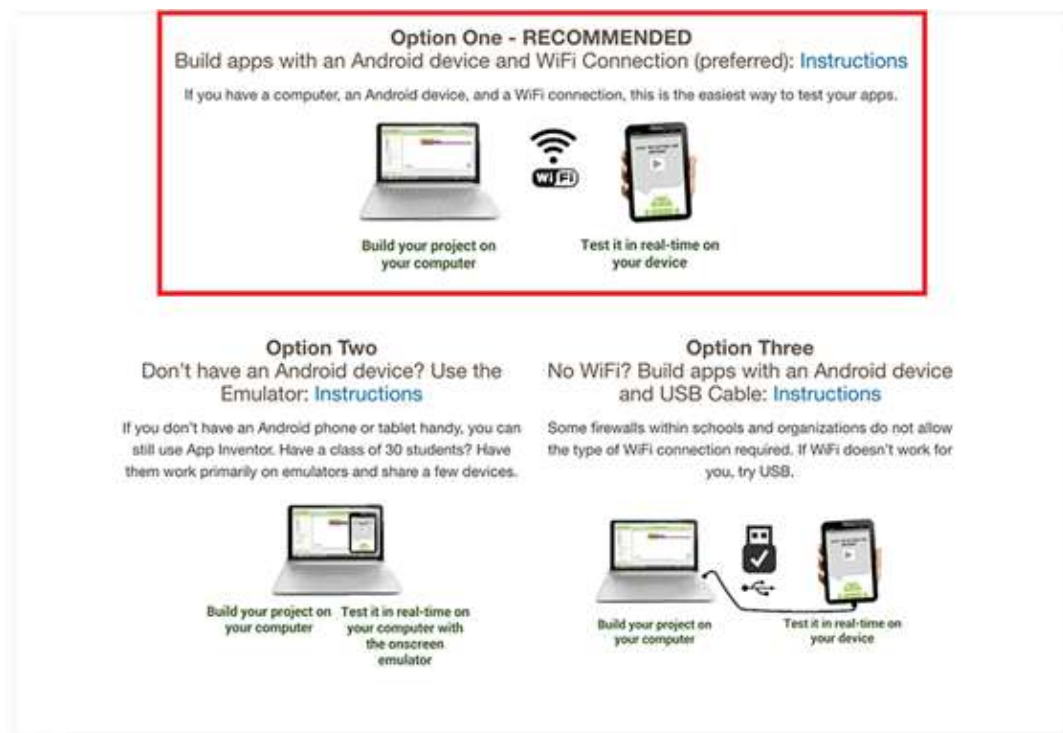
TEST YOUR APP

Before you start coding, make sure your app still works. In [Projects 1](#) and [2](#), the apps were tested using the emulator. This project tests the app using an Android device.

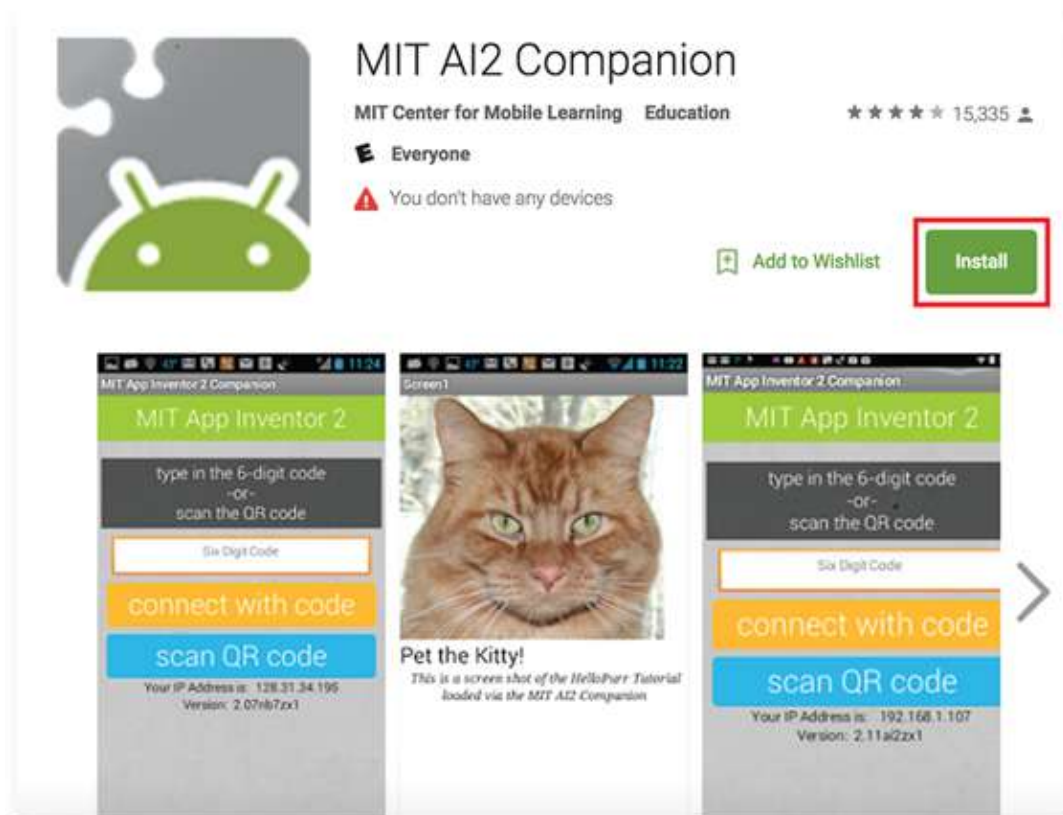


You can still use the emulator if you do not have an Android device. Refer to [Projects 1](#) and [2](#) for help.

1. Go to <http://appinventor.mit.edu/explore/ai2/setup.html> and you will see that three options for testing your app appear there. [Projects 1](#) and [2](#) followed Option 2 — using the emulator. This time, you use Option 1 — using an Android device and testing over Wi-Fi.

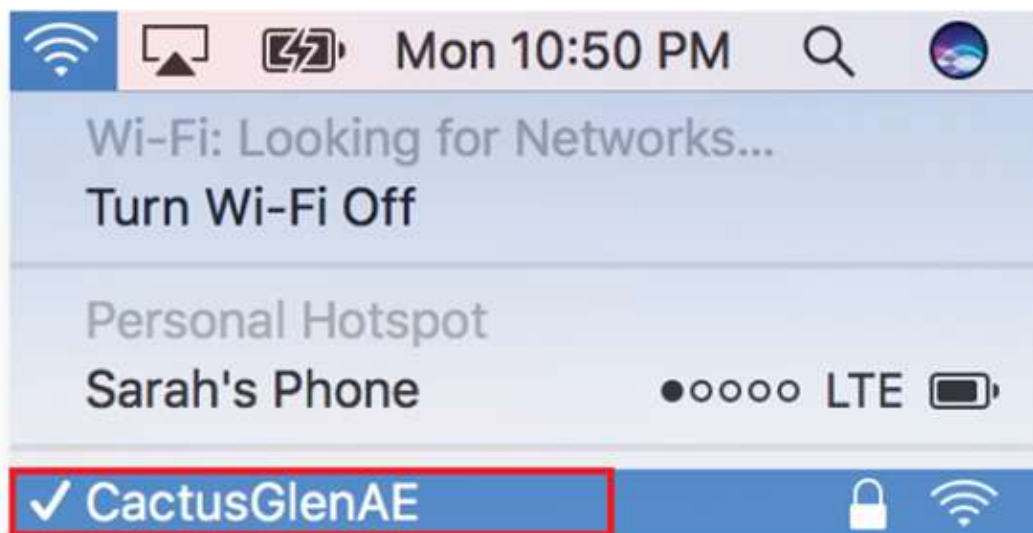


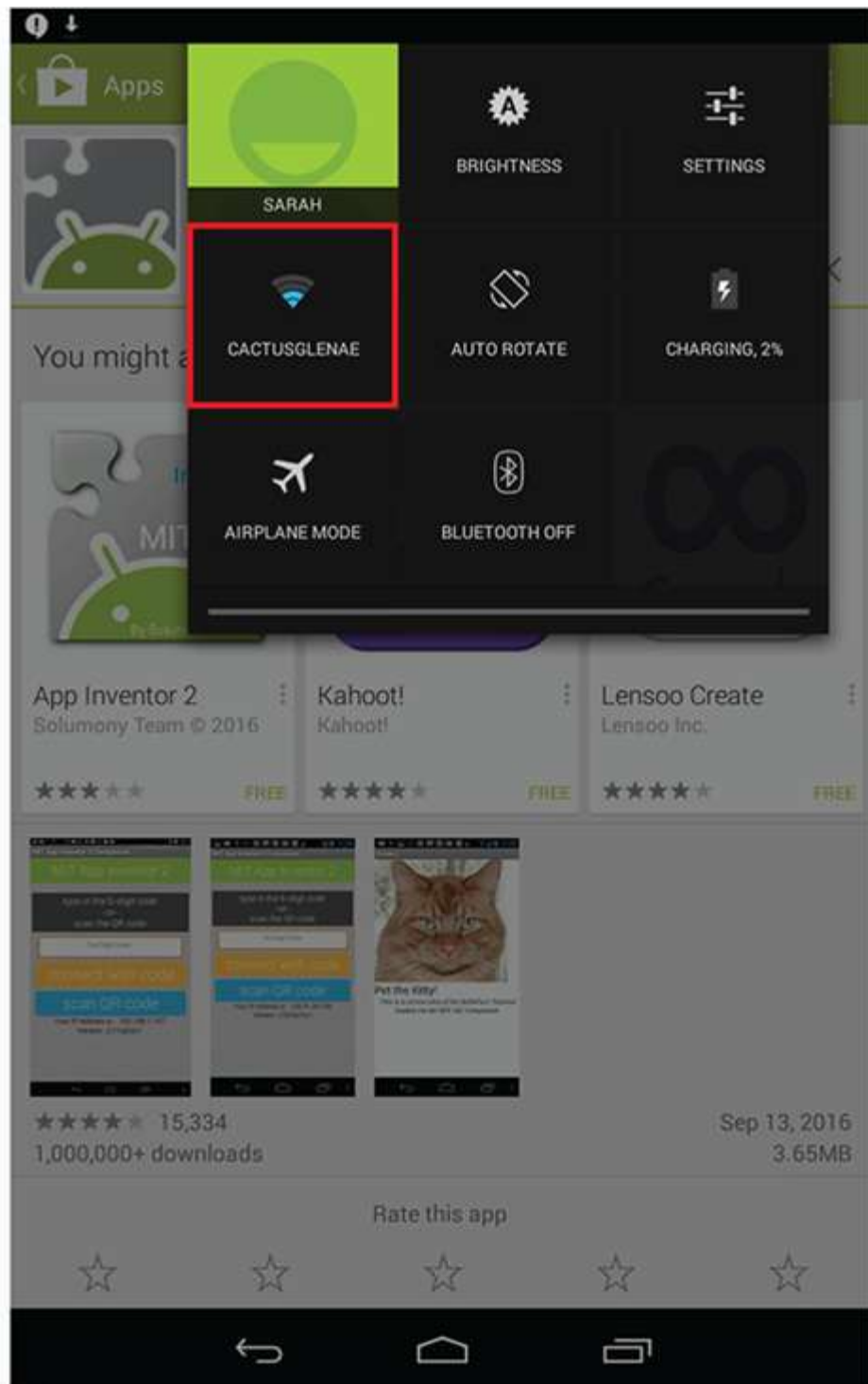
2. Follow the instructions on this page <http://appinventor.mit.edu/explore/ai2/setup-device-wifi.html> by first downloading the MIT AI2 Companion app on the Google Play Store.



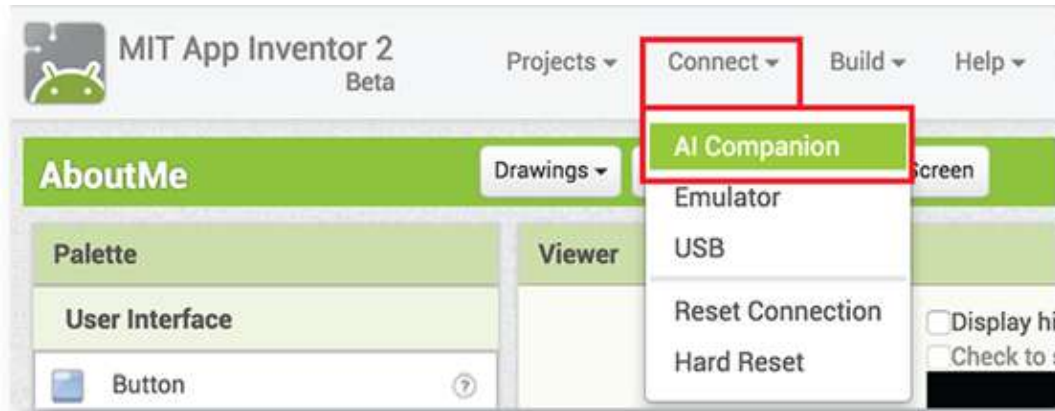
Always ask a parent for permission before downloading or installing any software.

- 3. Make sure your Android device and your computer are connected to the same Wi-Fi network.**





4. Go back to your app on <http://ai2.appinventor.mit.edu/> and click AI Companion.



5. **Open the AI2 Companion app on your Android device and type in the code that shows up on your screen.**

Connect to Companion

Launch the MIT AI2 Companion on your device and then scan the barcode or type in the code to connect for live testing of your app.

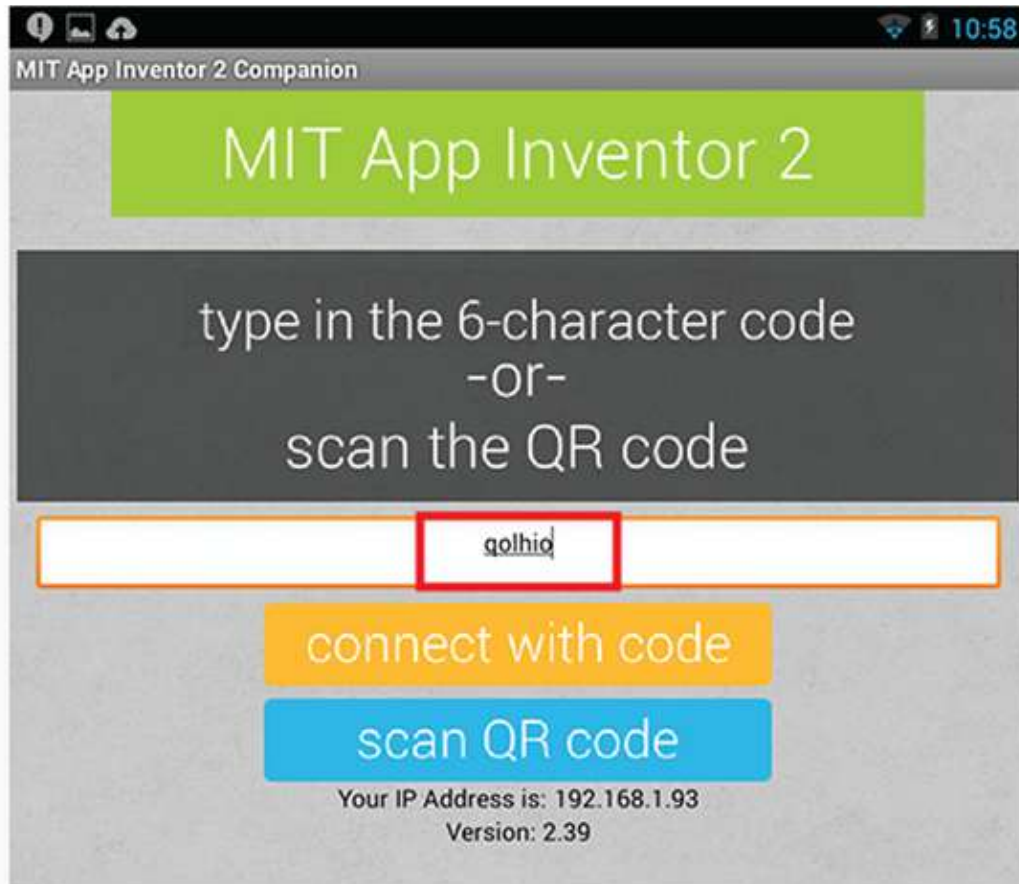
[Need help finding the Companion App?](#)



Your code is:

qolhio

Cancel



6. Click Connect with Code and your app should start!



Your code will be different from the one shown in this book, and might be different each time you test your app.



When you are testing your app, you can change the screen from your computer; just change to a different screen, and your app will change on your device, too!



CODE YOUR PHOTO-EDITING APP

The rest of this project continues with the photo-editing feature that was added to the About Me app. If you decided to create a new app just for photo editing, there might be some simple differences between what you have and what is in the book. Look out for “Warnings” that can help guide you.

SET YOUR BACKGROUND IMAGE

Before you start editing a photo, you have to have a photo! This section explains how to get a background image onto your Canvas.

1. **Click the Blocks button on the top-right of the screen.**

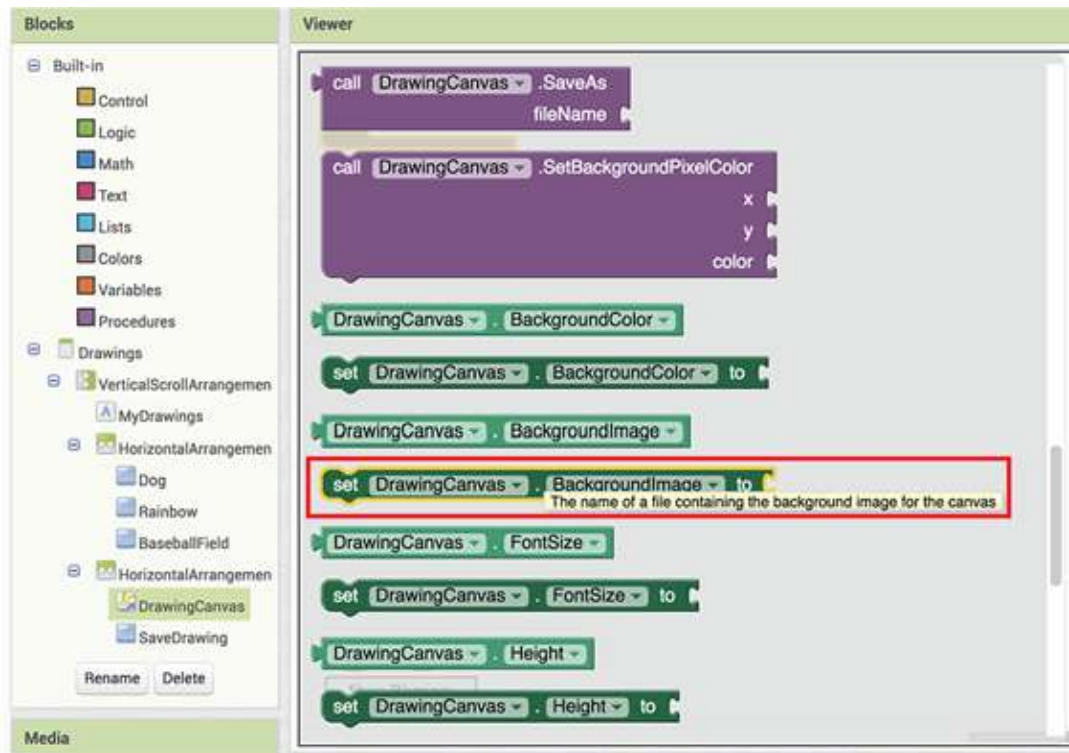


You should be on the Drawings screen.

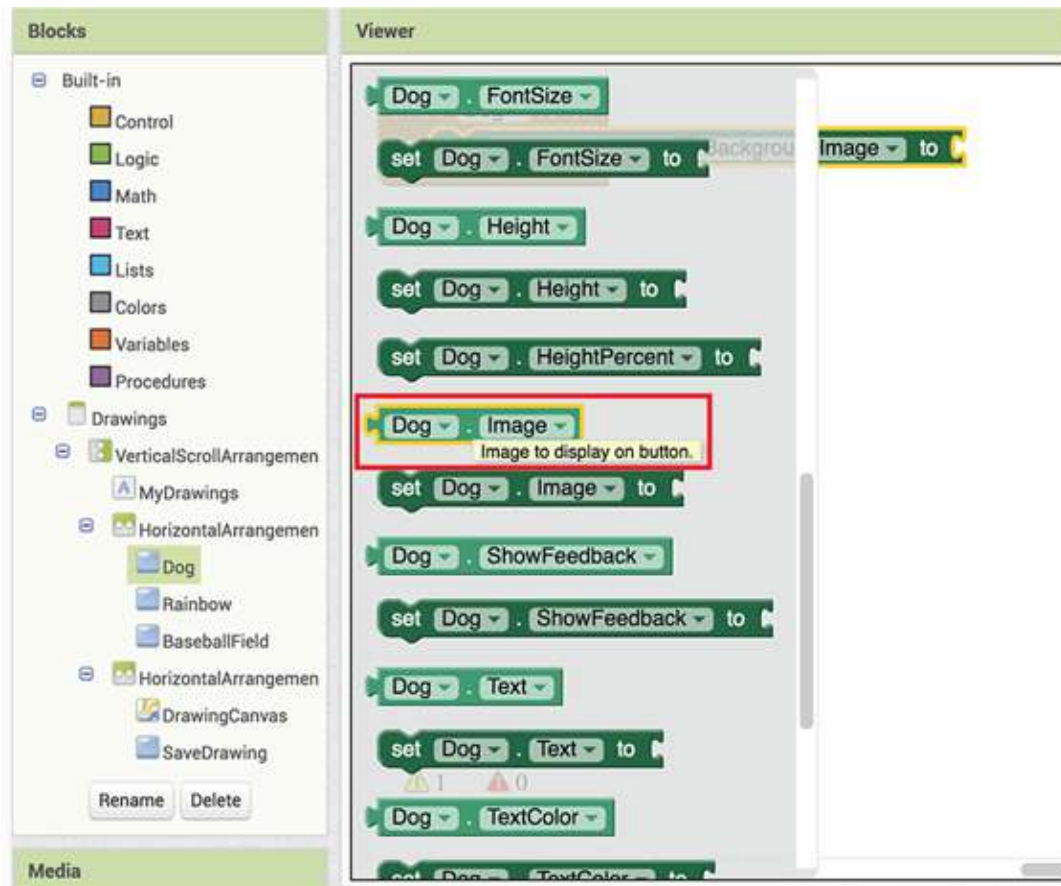
2. **From the Blocks column, click Dog and drag the block When Dog.Click into your coding area.**



3. From Blocks, click Drawing Canvas and drag a Set DrawingCanvas.BackgroundImage To block into the block from Step 2.



4. From Blocks, click Dog and connect a Dog.Image block to the block from Step 3.

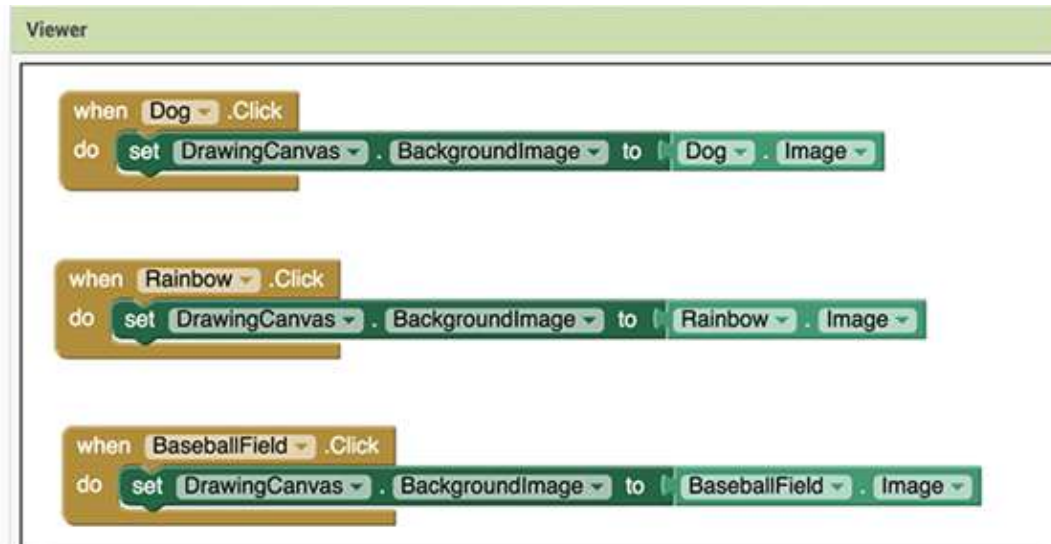


5. **Congratulations! Now when you click the Dog button, the Canvas background image changes to the dog!**



Make sure you test your app on your device!

6. **Now, add the same kind of code for the other two buttons, Rainbow and BaseballField.**



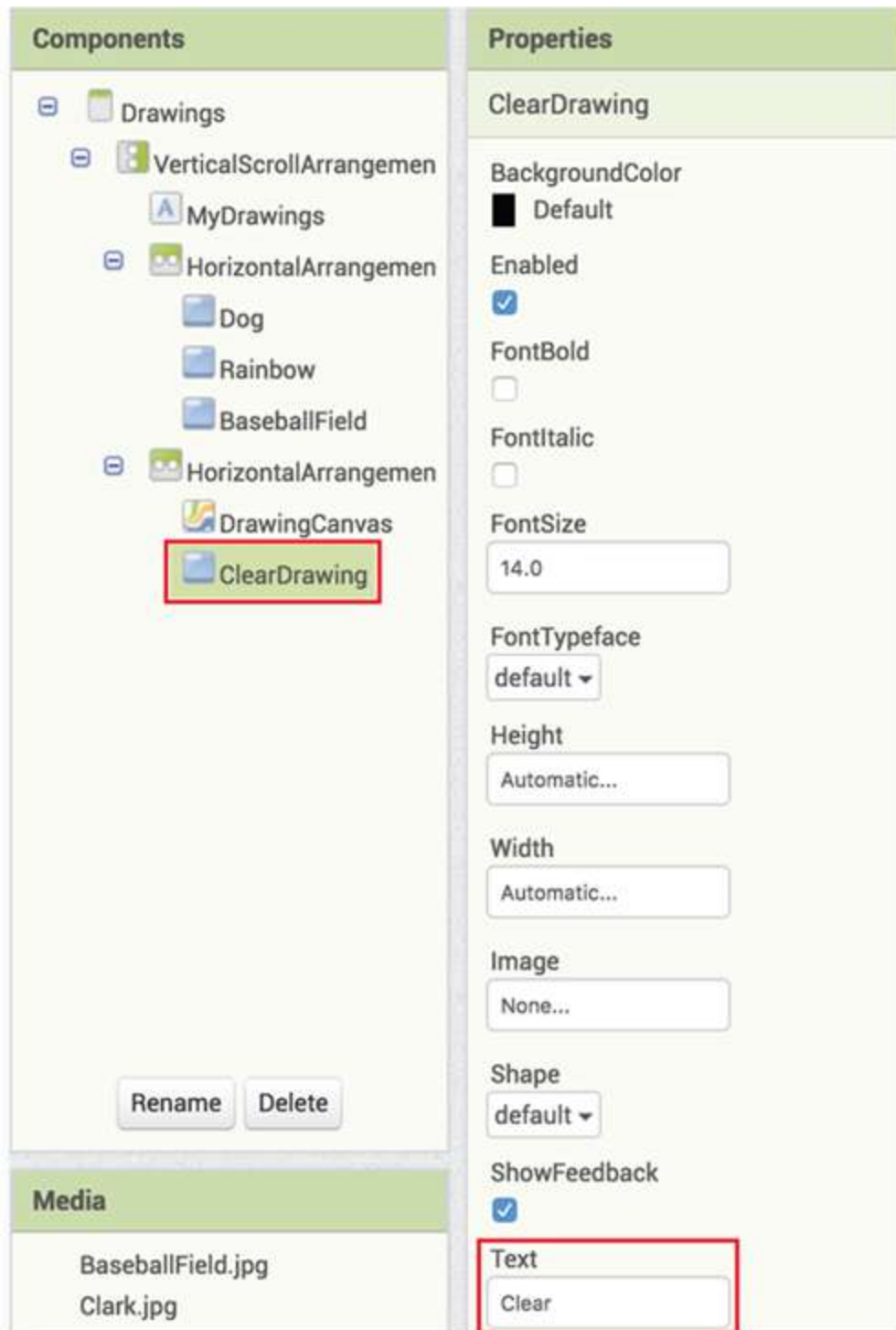
Make sure you test your app on your device!



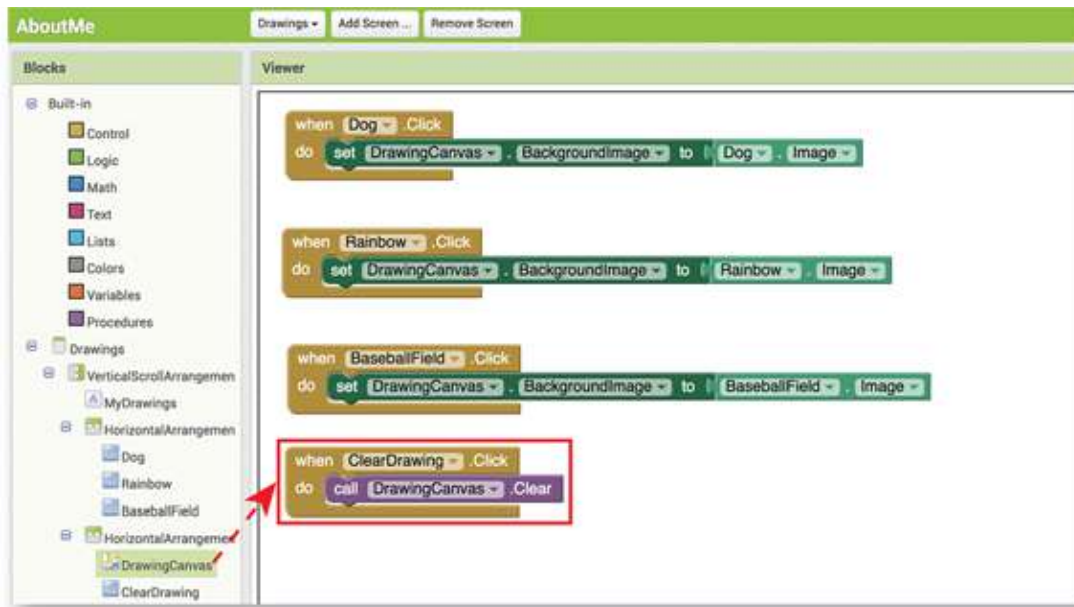
***EDIT YOUR DESIGN TO MATCH YOUR
FEATURE***

Sometimes when you start building your app you realize that what you thought you would make and what you actually made are two different things. This isn't a problem though; it happens all the time, even with professional coders. All you have to do is take the time to make sure that what you are building flows properly. For example, it would flow better if users could clear their drawing and start over. That way, they can draw a lot of different drawings, and if they want to save their drawing, they can just take a screenshot! These next steps show you how to change the Save button to a Clear button.

- 1. You can change your app to match your new feature by changing the Save button to a Clear button. First, change the name of the button.**

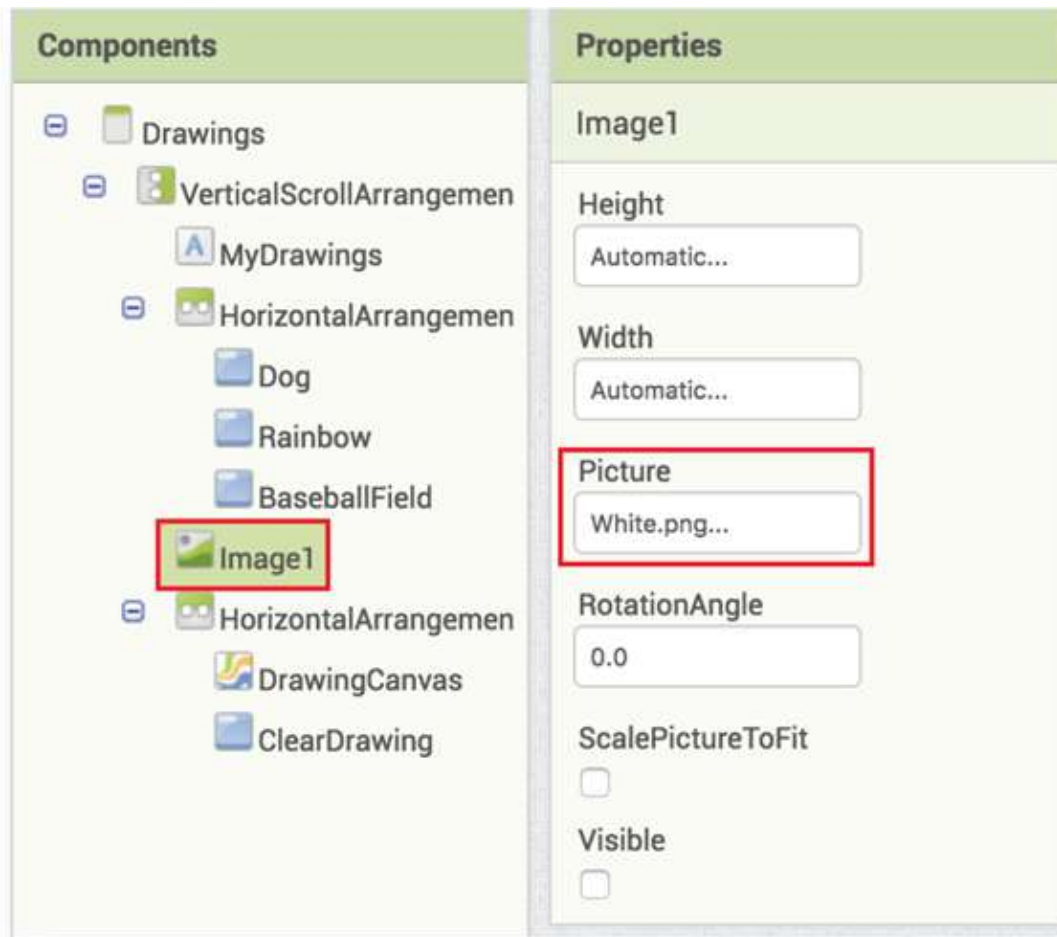


2. Then, in your code, add a **When ClearDrawing.Click** block and put a **Call DrawingCanvas.Clear** block in the block.



3. Next, add a new image, one that is just white, to your Drawings screen, and uncheck the Visible box.

You can get the white image from thewecan.zone/mobile-apps.



4. Finally, in your code, add a **Set DrawingCanvas.BackgroundImage To** block and attach a **White.Picture** block.



Test your app and make sure that when you click on a picture, the Drawing Canvas shows the picture, and when you click on the Clear button, the Drawing Canvas turns white.

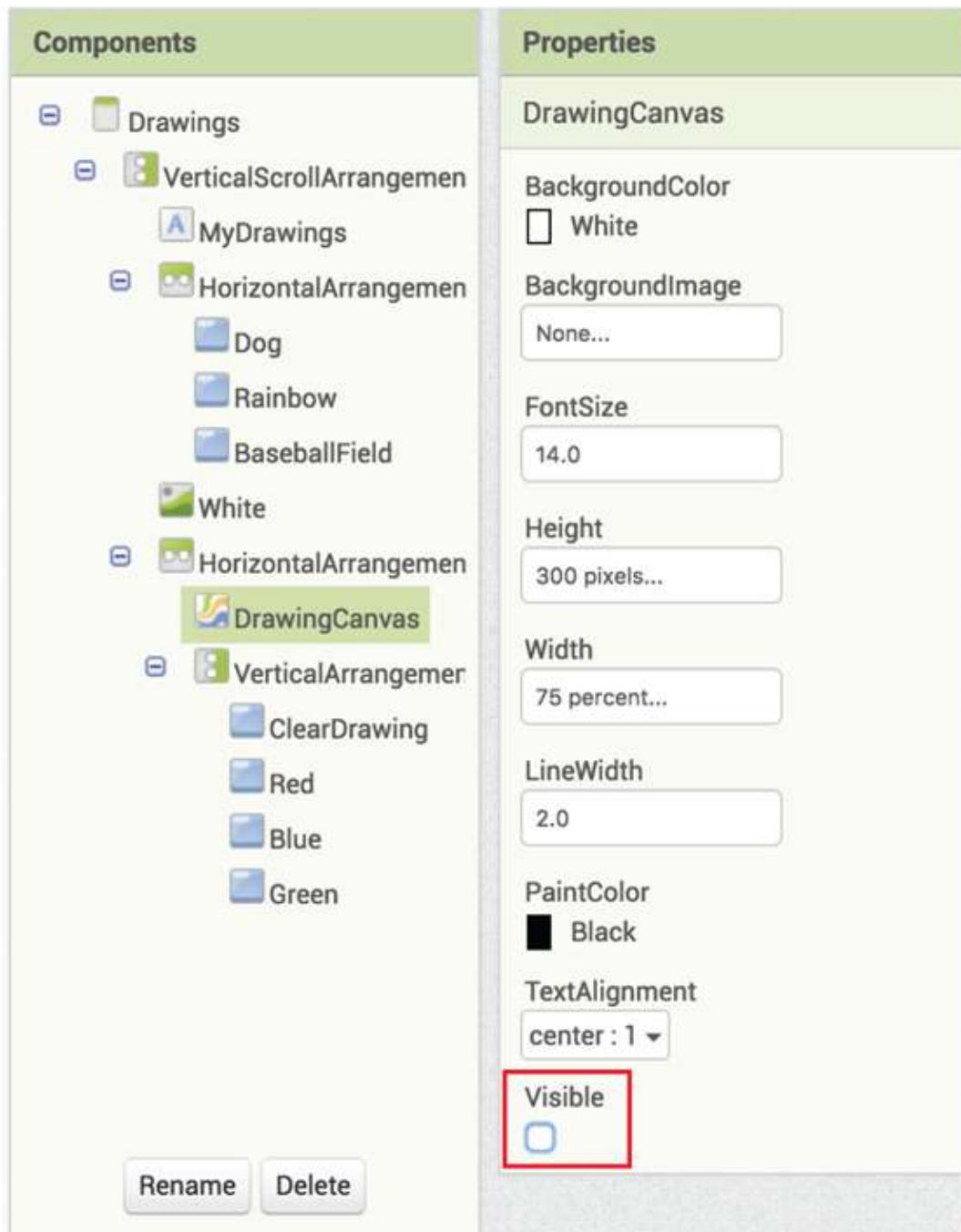


You might have to change the height of the Drawing Canvas to give yourself more room. Try changing it to 300 pixels by going back to “Designer” in the top right of your screen.

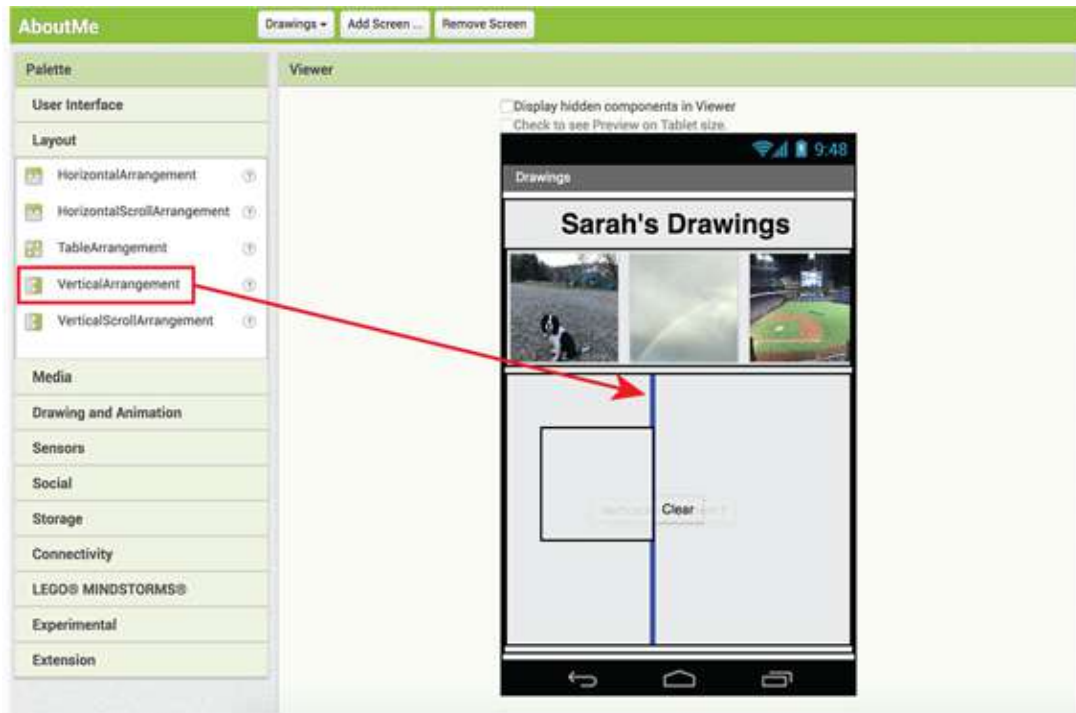
DRAW ON YOUR PHOTOS

It is finally time to make it so that you can edit your photos! Just follow these steps:

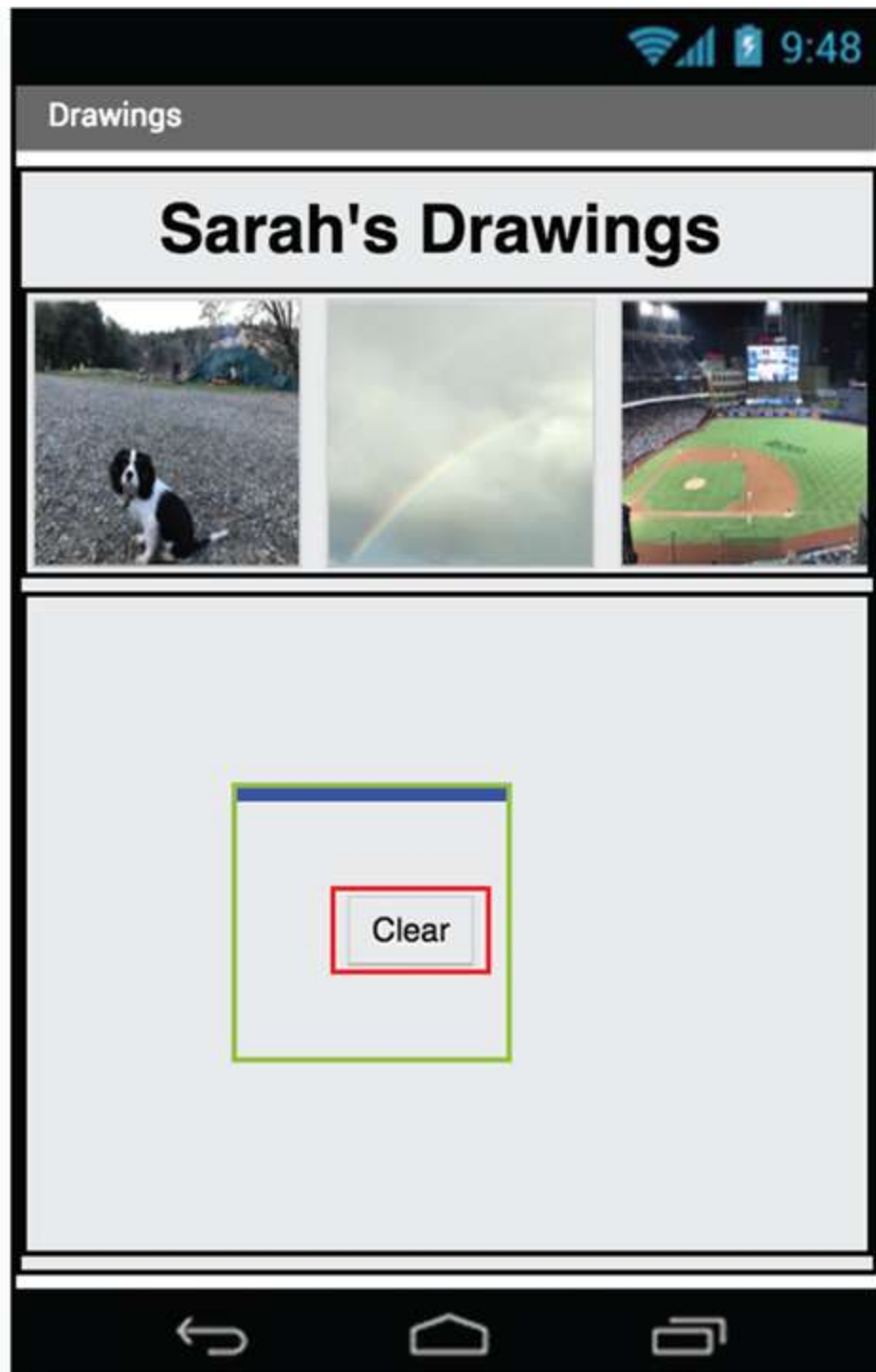
- 1. Uncheck the Visible box on your Drawing Canvas to give yourself more space while you design your screen.**



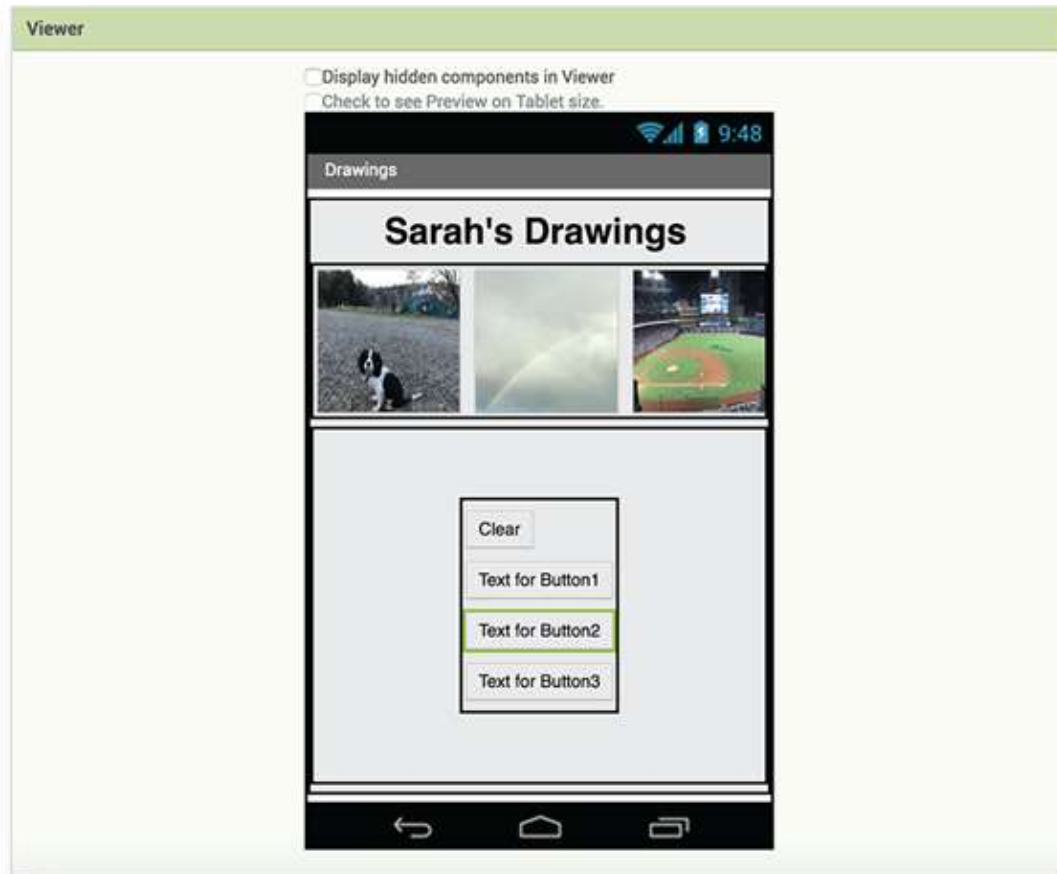
2. Add a VerticalArrangement next to the Clear button.



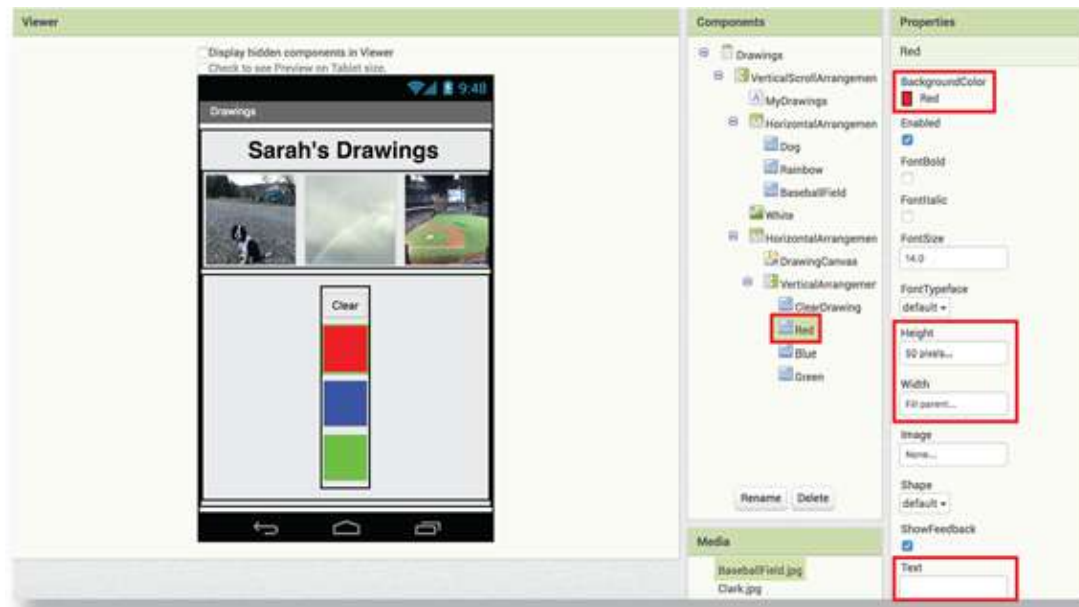
3. Put the Clear button inside the VerticalArrangement.



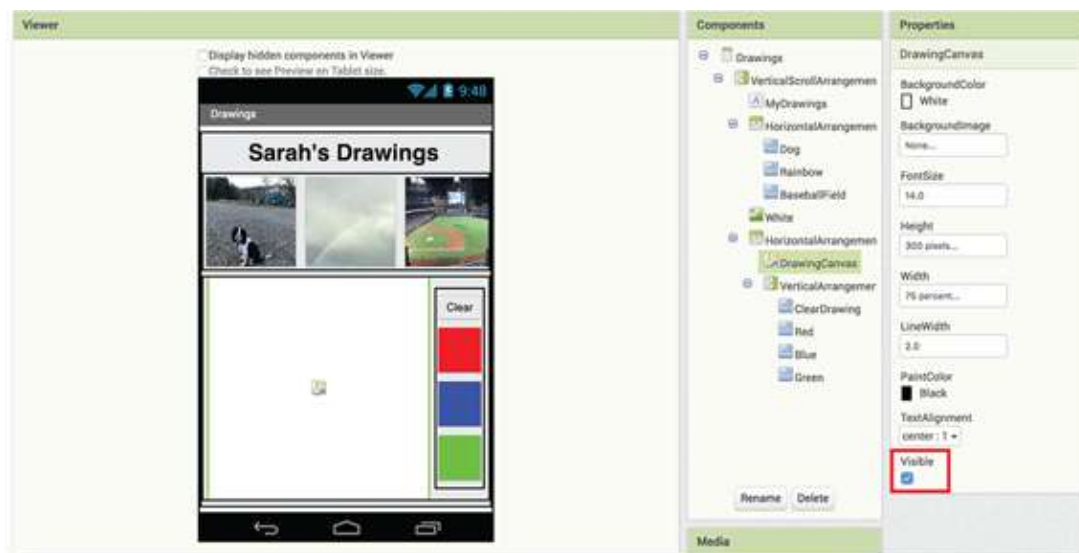
4. Add three more buttons underneath the Clear button.



5. **Rename the three buttons “Red,” “Blue,” and “Green.” For each button, remove the text, change the Height to 50 pixels and the Width to Fill Parent, and change the background color to match the name of the button.**



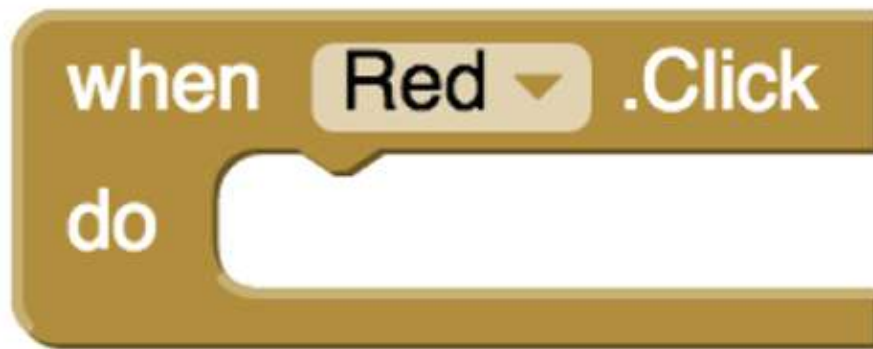
6. Make the Drawing Canvas visible again.



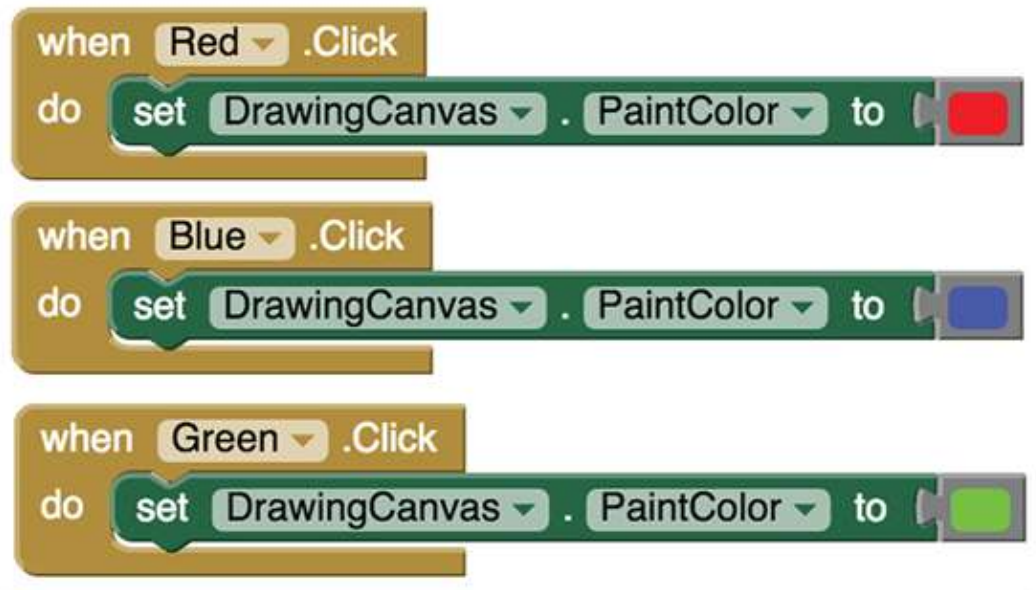
CODE THE ABILITY TO DRAW ON PHOTOS

Now that your colors are ready to be used, you have to code them to actually draw on your photos. Just follow these steps:

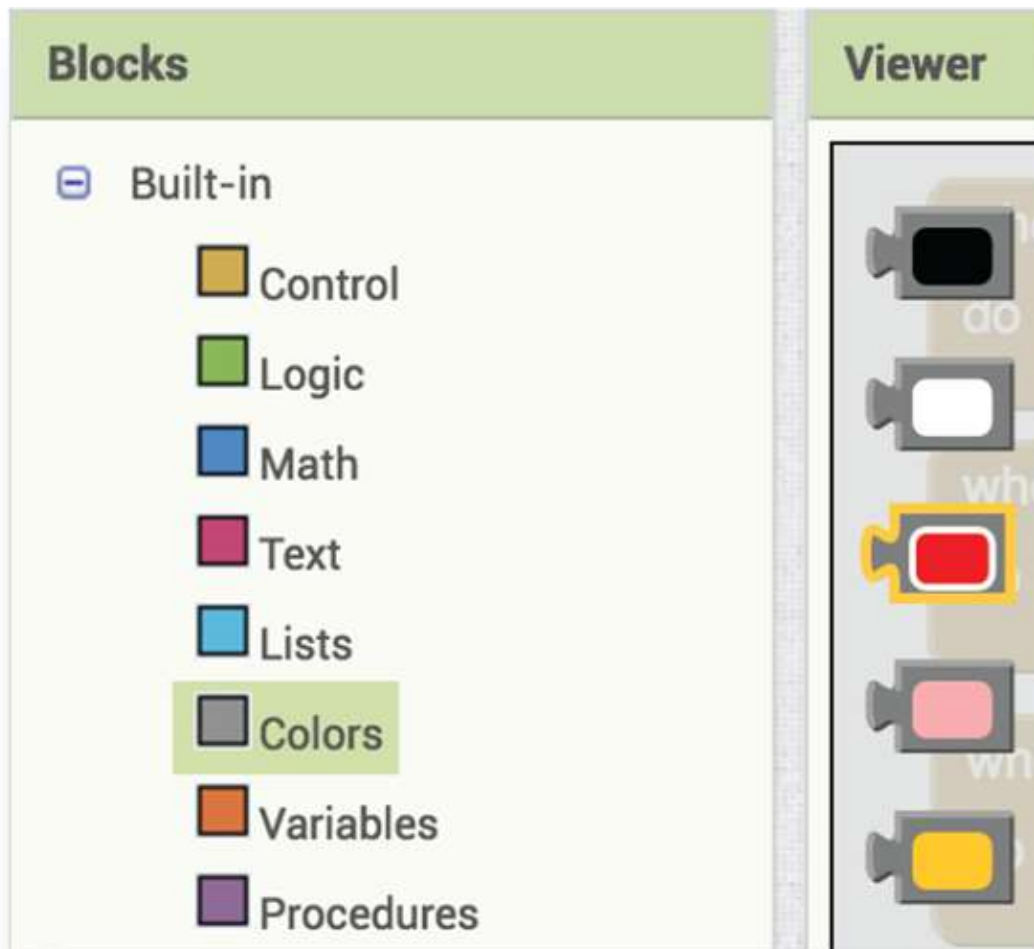
1. **Back on the Blocks screen (click the top-right Blocks button), add three code blocks: When Red.Click, When Blue.Click, and When Green.Click.**



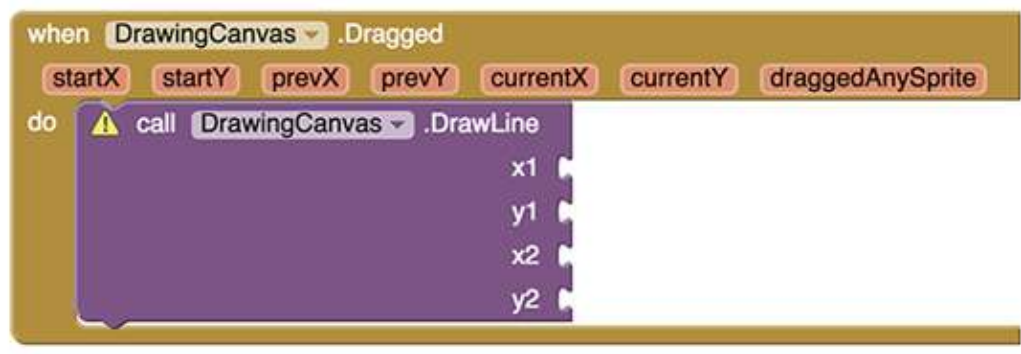
2. Add a Set DrawingCanvas.PaintColor To block to each of the code blocks from Step 1, and put the correct color into each one.



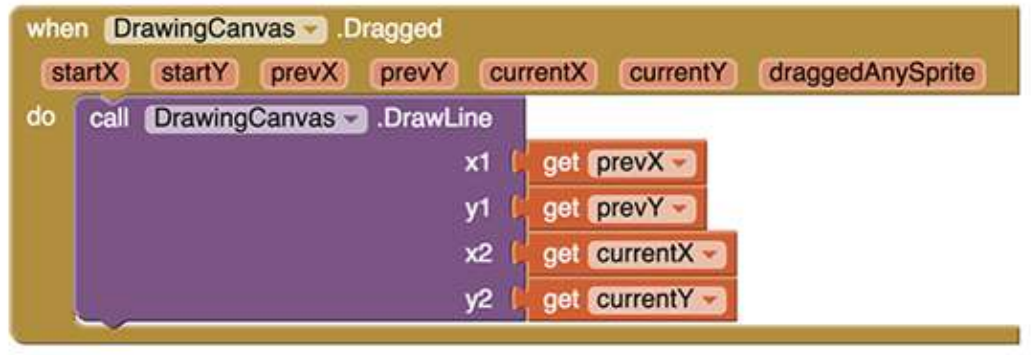
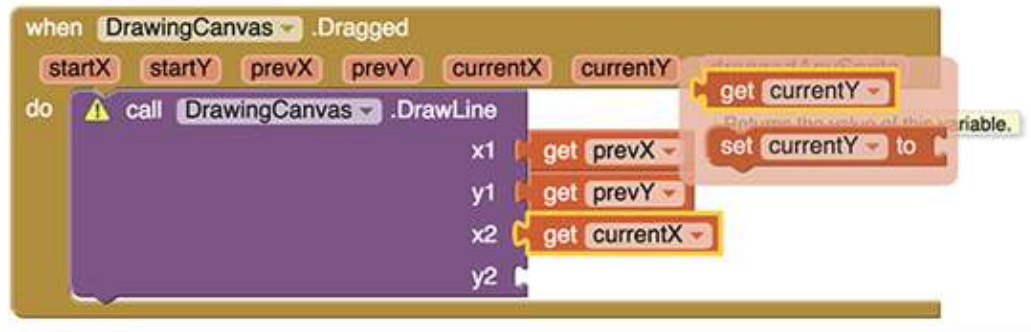
You can find the color blocks in the Built In category.



3. From the DrawingCanvas, drag When DrawingCanvas.Dragged and Call DrawingCanvas.DrawLine coding blocks into your coding area.



4. Hover over PrevX and drag a Get PrevX block into the X1 slot. Do this for PrevX, PrevY, CurrentX, and CurrentY.



5. Final step, similar to Steps 3 and 4: Bring in When DrawingCanvas.Touched and Call DrawingCanvas.DrawPoint blocks so that you can make dots on your Canvas, too.



6. Test your app, make sure you can draw on the pictures, and clear the screen!

Sarah's Drawings



Clear



PROJECT 4

MAKE A MOBILE GAME APP



IN THIS PROJECT, YOU LEARN HOW TO MAKE A GAME! You first learn how to make a picture move around your screen. Then, you learn how to make that picture “collect” points. Finally, you learn how to

add more game features like making things on the screen appear in random places!

DECIDE ON A GAME

Before you start building, you have to decide what kind of game to build. This book guides you through making a game in which you have to feed a puppy all the food in a short amount of time. It's kind of like Pac-Man eating the yellow dots, Flappy Bird collecting coins, or even Mario and Sonic. But this version is a bit simpler.

SET UP YOUR GAME APP

To get started on your game app, you have to make a new app in App Inventor. Then you can set up the basic version of the game.

CREATE A NEW PROJECT

If you have been following the entire book, then this will be the third project that you have created. So this should be something you are used to doing by now, but if you need additional support, check the previous projects for more details!

1. **Go to appinventor.mit.edu. Click the Create Apps button and log in.**
2. **Create a new project by clicking open the Projects drop-down menu and selecting Start New Project.**
3. **Name your project “FeedWinston.”**

Now you're ready to go! You can start designing your screen and coding your game!

SET UP YOUR SCREEN

Before you start coding, you have to create the screen. To make the first version of the game, this is pretty simple; just follow these steps to get started.

1. Drag a Canvas onto your screen.

You can find it in the Drawing and Animation section of the Palette column.



2. Make the Canvas fit your screen. Set the Height and Width to Fill Parent. Then, make the background color Cyan.

Properties

Canvas1

BackgroundColor

Cyan

BackgroundImage

None...

FontSize

14.0

Height

Fill parent...

Width

Fill parent...

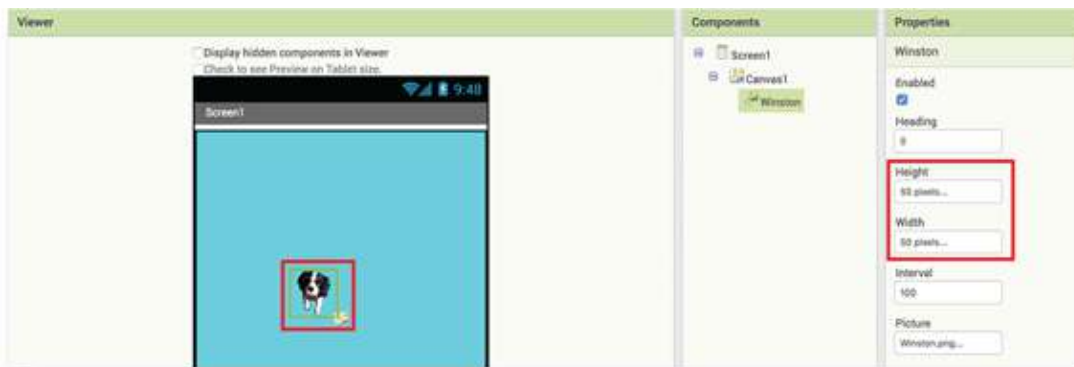
3. Drag an Image Sprite onto your Canvas.

You can find it in the Drawing and Animation section of the Palette column.



4. Turn the Image Sprite into Winston by performing the following steps:

- a. Download the Winston.png image from www.thewecan.zone/mobile-apps and upload it to App Inventor.
- b. Set the Image Sprite image to Winston.png and change the name to "Winston."
- c. Change the Height and Width to 50 pixels each.



Congrats! You can now move on to code Winston and make him move around the screen.

MAKE A SIMPLE GAME

Trying to come up with a new and exciting game idea can be a bit challenging, but when you're just getting started, it's a good idea to build off of things that you know! For this app, you need a game that fits on

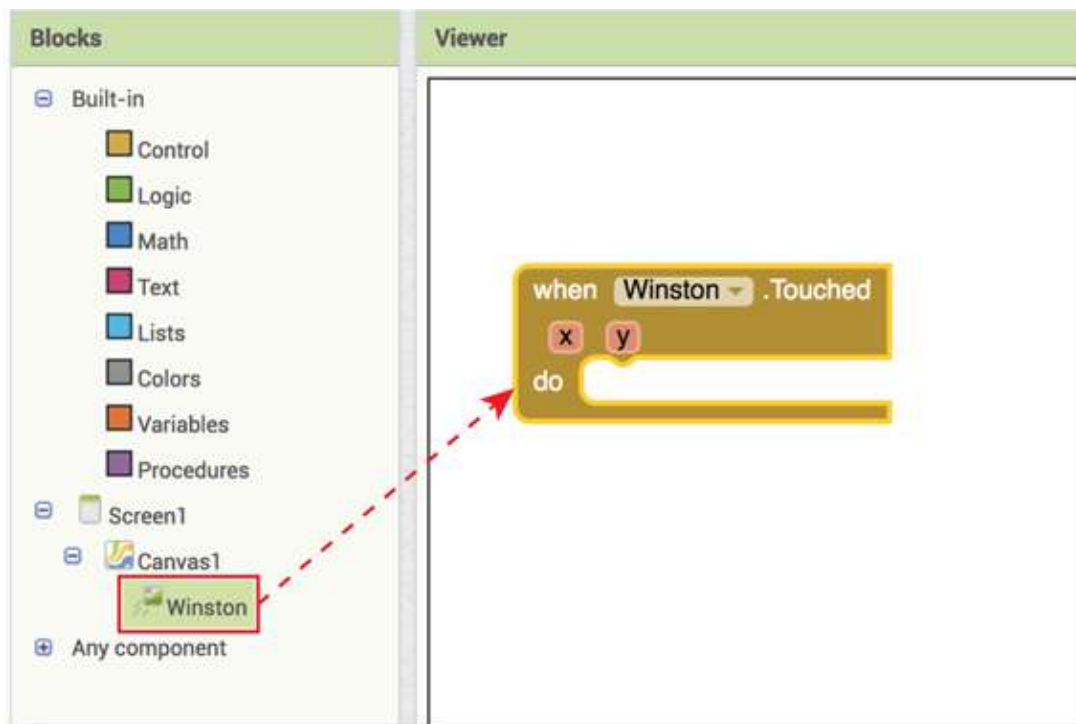
one screen, has a character you move around, and has something for the character to collect. Try thinking of other video games that you have played that are like this. (HINT: Ask your parents what games they liked to play when they were younger.) Follow the steps outlined in the next couple sections to make this game playable!

CODE WINSTON TO MOVE

Now that Winston is on the screen, you can code him to move around! Click the Blocks button in the top-right corner to go to your coding area.

Then follow these steps:

1. **Make Winston move when clicked: Click on Winston in the Blocks column and drag a When Winston.Touched into your viewer.**



2. **Set Winston's speed: Under Winston in the Blocks column, drag a Set Winston.Speed To block into the When Winston.Touched block you added in the last step. Then, click Math in the Blocks column to find a number block to connect to that.**



3. **Test your code:** Your code for a simple version is almost complete. Test your code in the emulator or on your Android device.

When you are testing your code, click on Winston and he should start to move.



If you cannot remember how to test your code, revisit the “[Test Your App](#)” section in [Project 3](#) for a reminder.

GET WINSTON UNSTUCK

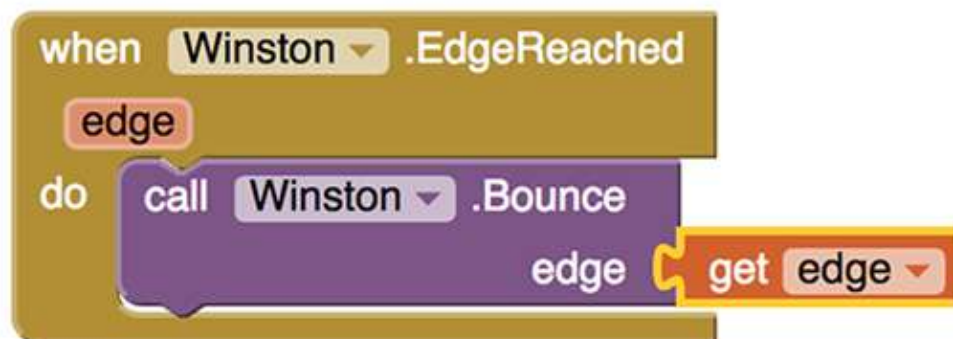
If you test your app and click on Winston, you will notice that he stops moving when he reaches the edge.

To fix this, we can add a bit of code:

1. From Winston in the Blocks column, drag a When Winston.EdgeReached block into your viewer. Then add a Call Winston.Bounce block into the Winston.EdgeReached block that you just dragged in.

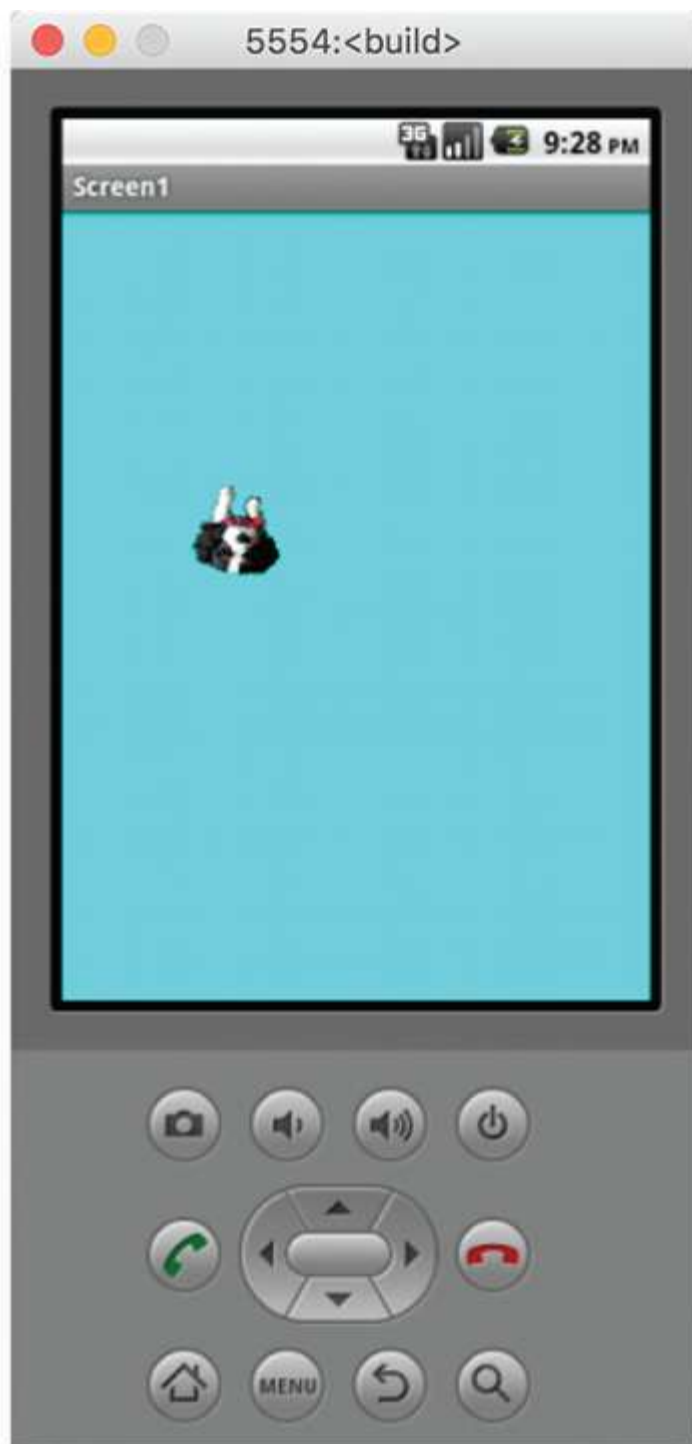


2. From Variables in the Blocks column, drag a Get block to connect to your Call Winston.Bounce block. Choose Edge from the drop-down menu.



3. Test your app again.

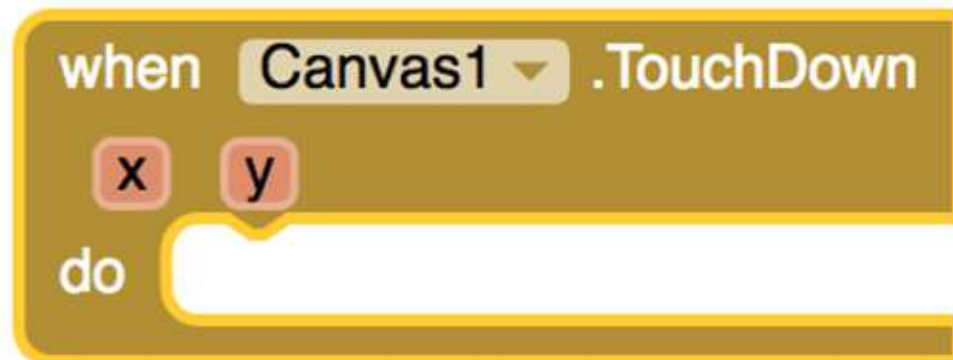
Go back to your app and you will notice that Winston moves back and forth across the screen, bouncing when he reaches an edge.



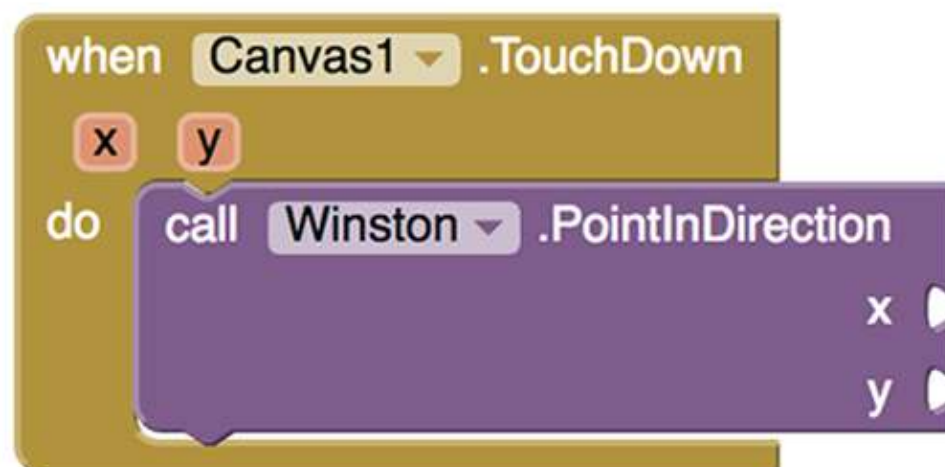
MAKE WINSTON LISTEN

Now that Winston can move, you should make it so that he will go where you want him to go.

1. From the Canvas in the Blocks column, drag a When Canvas1.TouchDown block into your viewer.



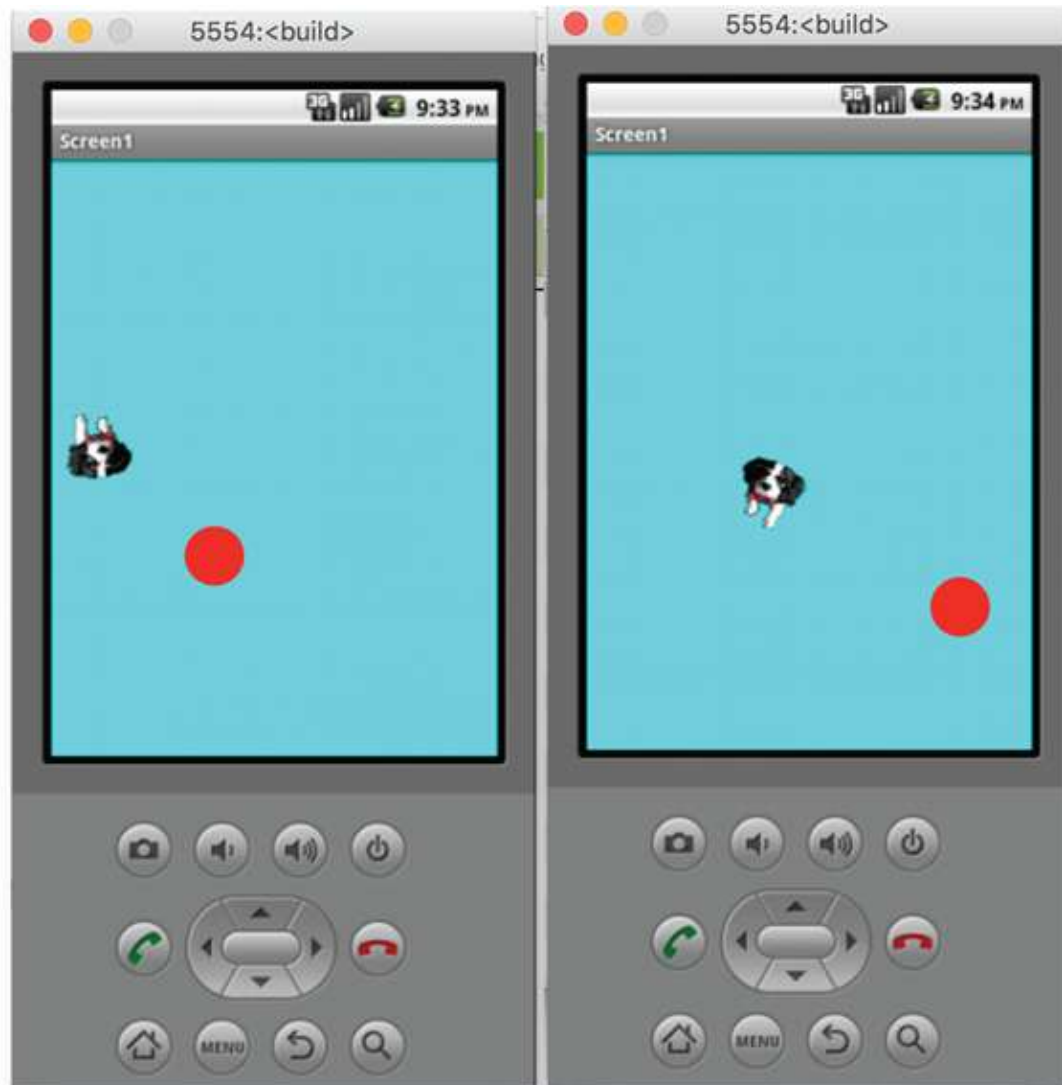
2. From Winston in the Blocks column, drag a Call Winston.PointInDirection block into your TouchDown block.



3. From Variables, drag two Get blocks to connect into your PointInDirection block, and then, in their drop-down menus, change one to X and one to Y.



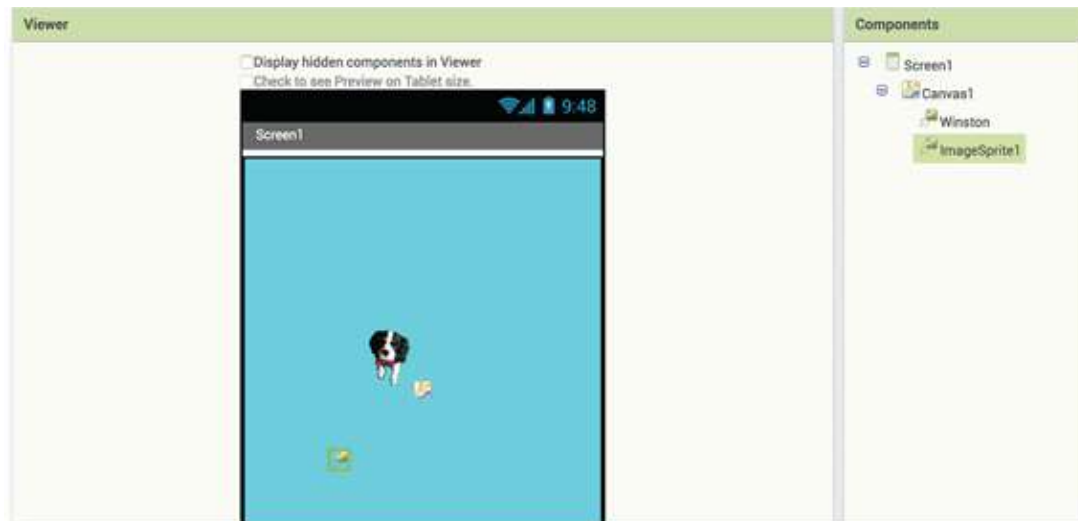
4. Test your app again and watch what happens when you tap around the screen.



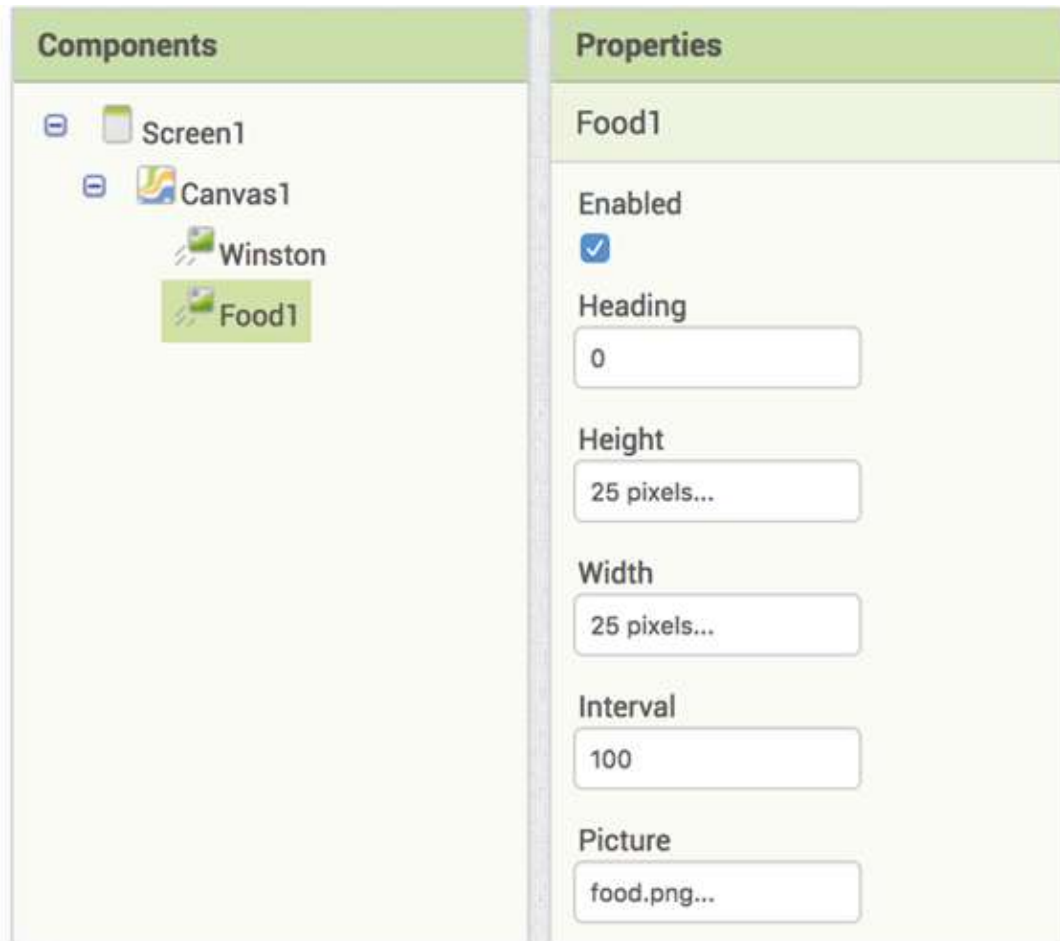
GIVE WINSTON SOME FOOD

Now that Winston can move around the screen, give him some food!

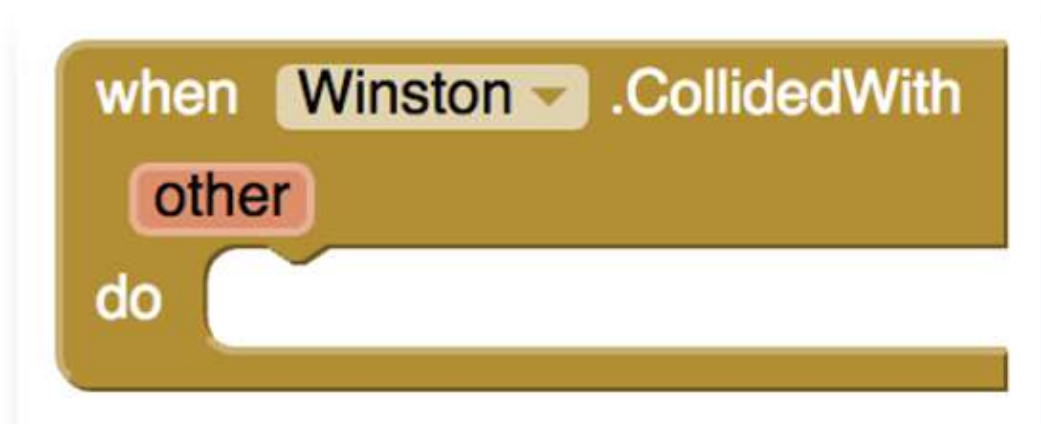
1. **Download some dogfood:** Go to www.thewecan.zone/mobile-apps and download the image file called food.png.
2. **Add food to your app:** Go back to the Designer screen and add another Image Sprite into your app.



3. **Set up your food sprite: Change the image of your new Image Sprite to be food.png and rename the sprite to “Food1.” Then, resize the Width and Height of Food1 to be 25 pixels each.**



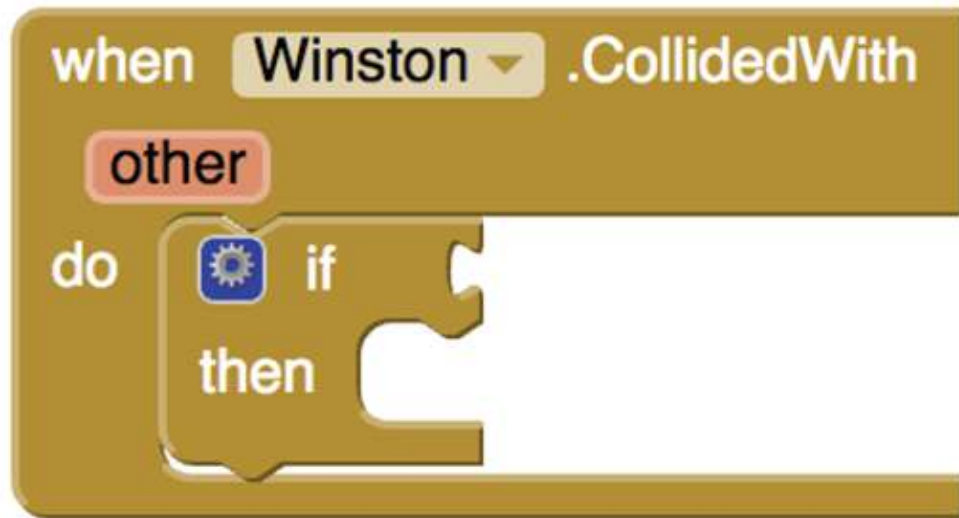
4. Check when Winston collides with the food: From Winston in the Blocks column, drag a When Winston.CollidedWith block into your viewer.



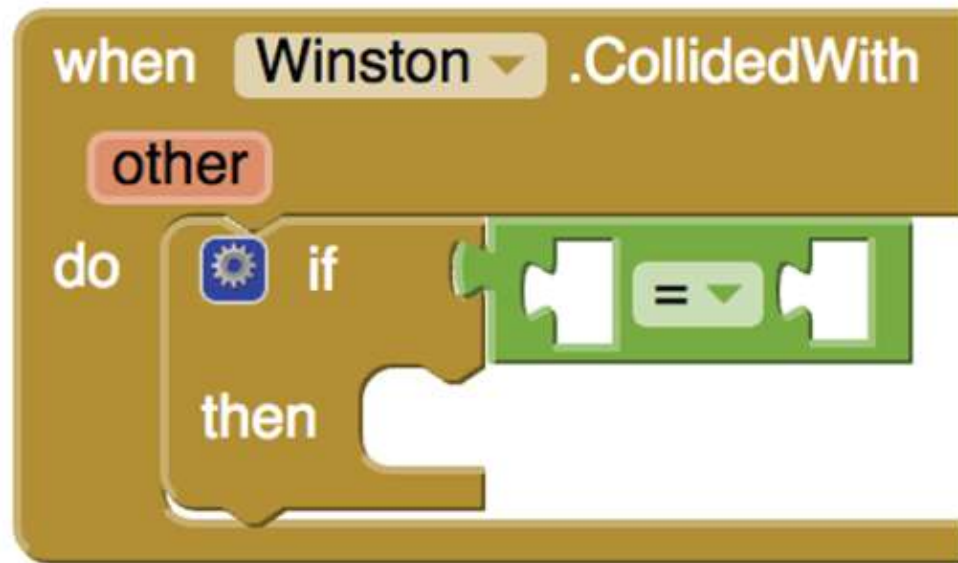


“Collide” means “to run into.” So, if you accidentally run into a pole, then you collided with that pole.

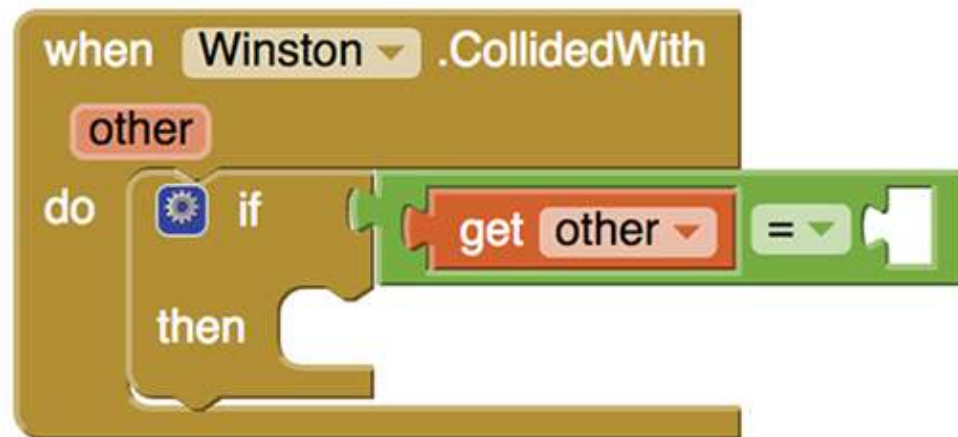
5. From Control, drag an If Then block into your CollidedWith block.



6. From Logic, drag an = block onto your If Then block.



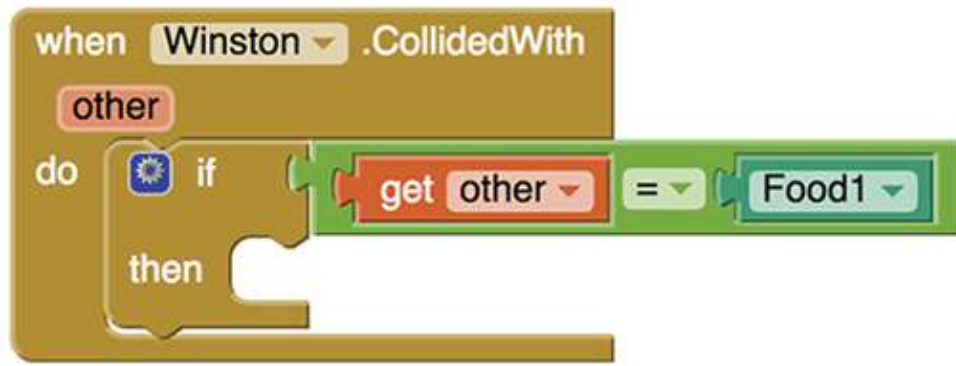
7. From Variables, drag a Get block into your = block, and in the drop-down menu, select Other.



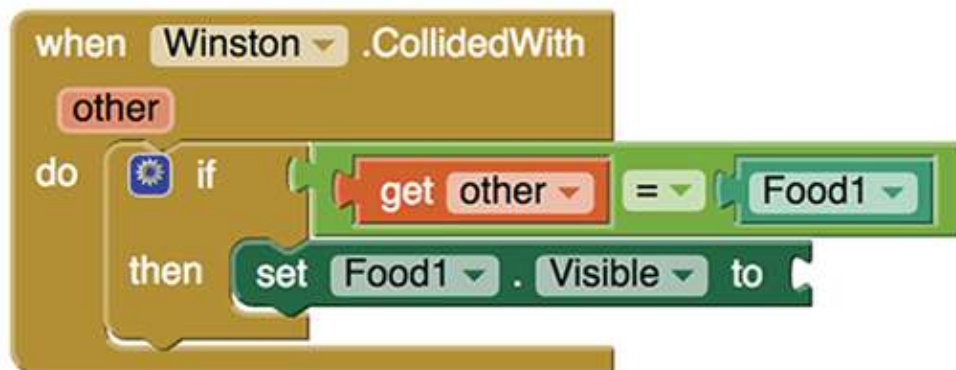
“Other” is just the name for whatever Winston hits. Later, when you add more food blocks, “other” will refer to whatever food

block he collides with, not just Food1.

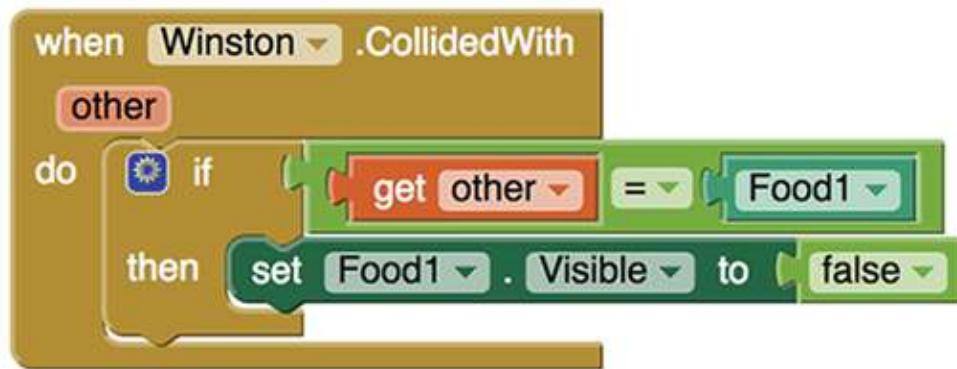
8. From Food1, at the very bottom of the list, drag a Food1 block into your = block.



9. From Food1, drag a SetFood1.Visible To block into your If Then block.



10. From Logic, drag a False block into your Visible block.



11. Test your app again. This time, when Winston collides with Food1, it should disappear!



Congrats!! You now have the simpler version of “Feed Winston”! The next few sections show you how to make it even more fun!

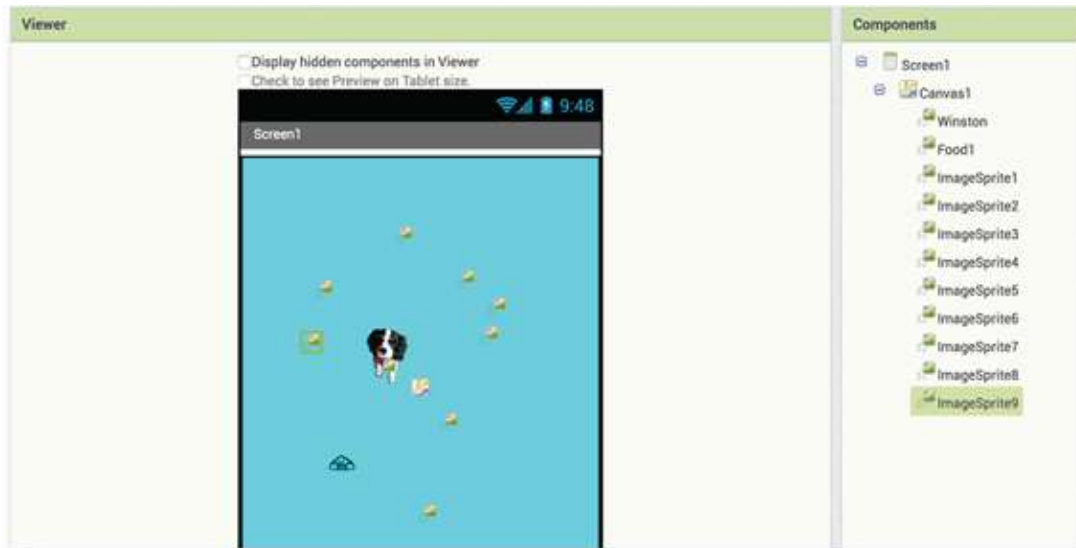
MAKE A RE-PLAYABLE GAME

As you may have noticed, after Winston eats his bowl of food, he just runs around the screen until you quit the app. To make this game a bit more fun, add more food and add the ability to restart the game.

GIVE WINSTON ALL HE CAN EAT

This section shows you how to give Winston even more food — ten whole bowls of it!

1. **Go to your Designer and add nine more Image Sprites into your Canvas.**



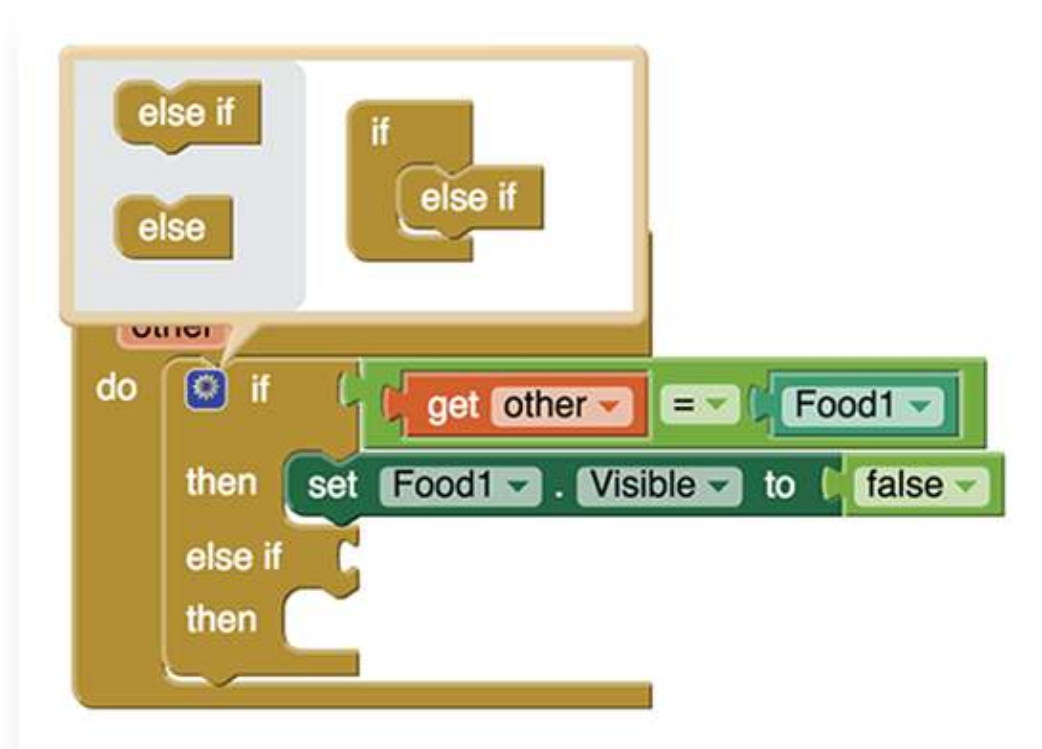
2. **Rename the sprites “Food2” through “Food10,” and change their images to food.png and their Width and Height properties to 25 each.**



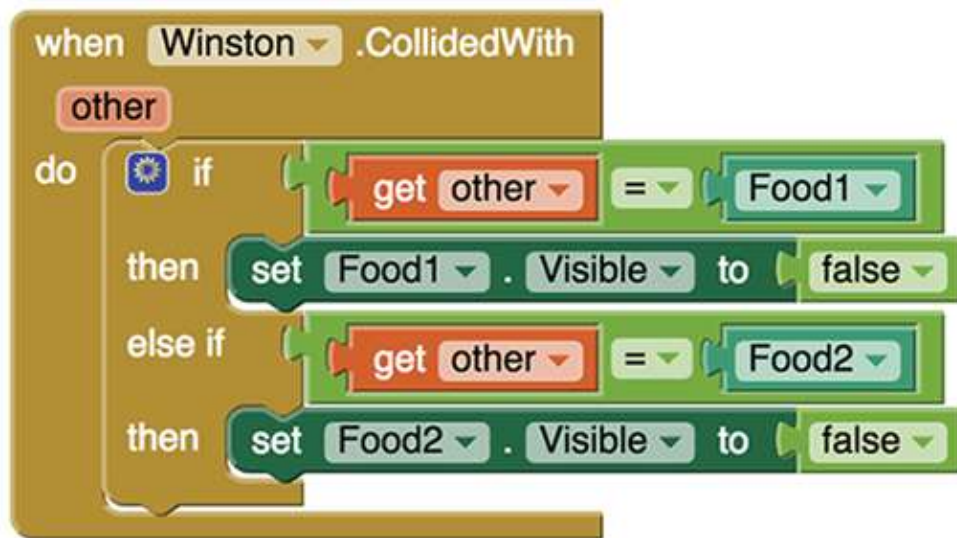
CODE WINSTON'S FOOD

Now that Winston has a lot of food on the screen, you have to make sure that when he runs into *any* of the food bowls, they disappear.

1. **Go back to Blocks (the coding area). Find the When Winston.CollidedWith block you already made, and click the blue button on the If Then block inside of it. Then, drag an Else If block into the If block that pops up.**



2. **Make changes to this new block so that it copies the code for Food1. However, the new block should say Food2 instead of Food1.**



WAIT! This is going to take forever! Let's use a shortcut: Instead of doing this for each food item, let's put all the food into one list and then iterate through that list!



"Iterate" means "to do again and again." In computer terms, iterating usually means to repeat an action using a sequence of numbers or values: The computer does the action for the first value, and then it does it again for the second value, and again for the third value, and so on. In this case, we have a list of food items and we "iterate" through that list to make each item invisible when Winston collides with it.

Use the following steps to iterate through all the food items:

1. **Create a list:** From Variables in the Blocks column drag an Initialize Global block into your viewer and rename it "theFood." Then, from Lists in the Blocks column, connect a Create Empty List to the Initialize block.



initialize global theFood to create empty list

2. **Add elements to the list:** From Lists drag an Add Items to List block into the When Winston.Touched block, since that is what starts the game. Click the blue button on the Add Items to List block and make sure there are ten items in the list.

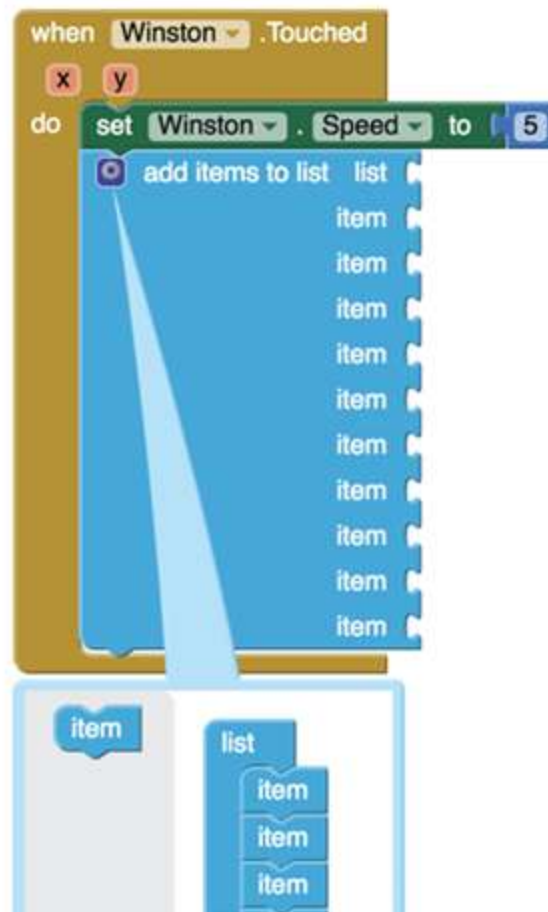
STEP 1:



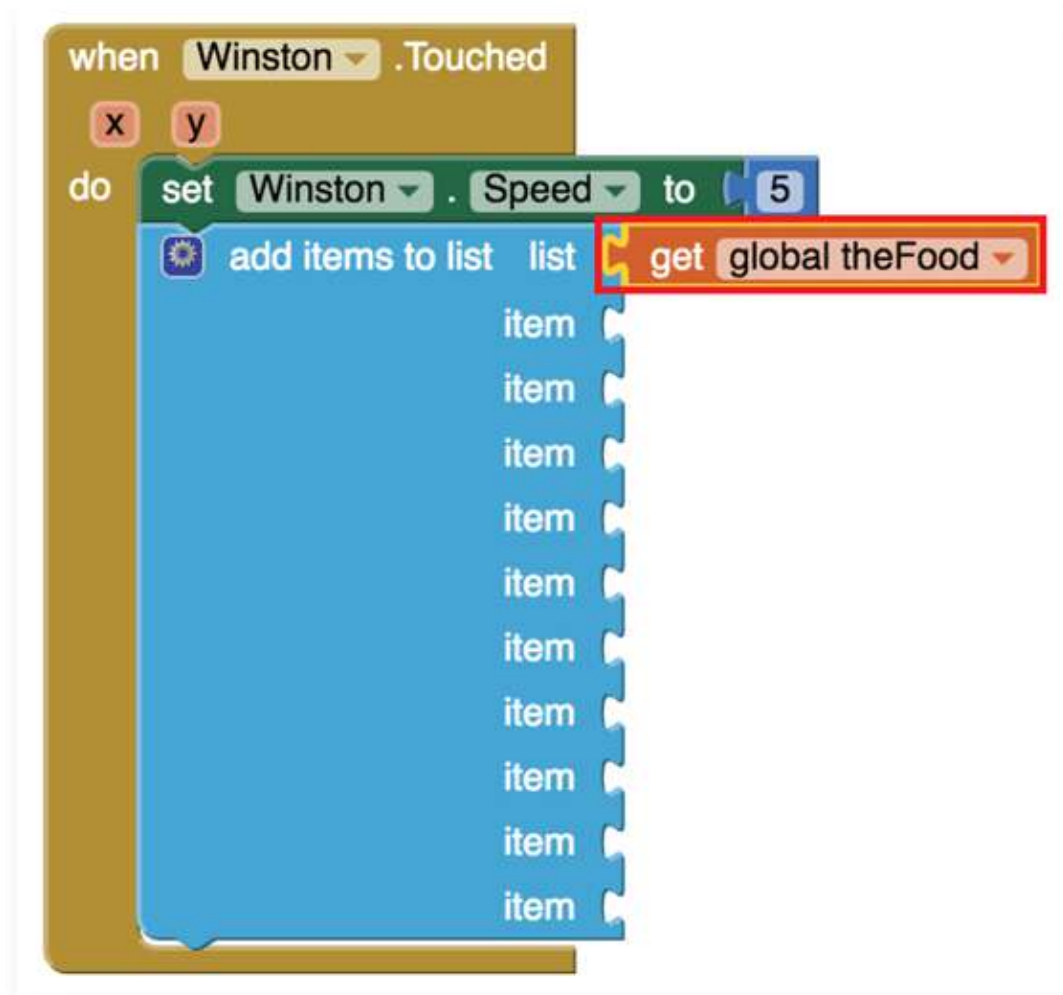
STEP 2:



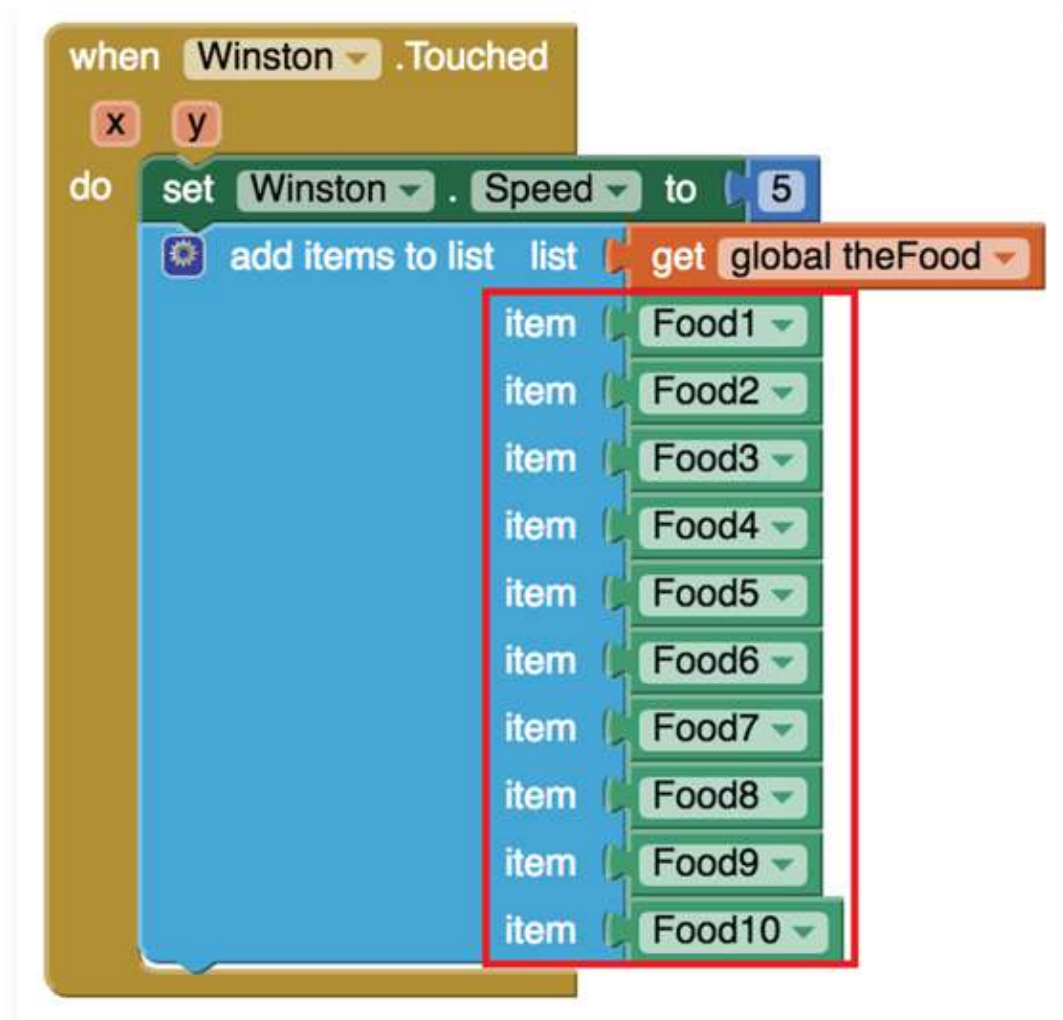
STEP 3:



3. Choose the list: From Variables drag a Get block and attach it to the Add Items block on the List opening. Make sure that the drop down is “global theFood.”



4. **Add the items:** Under each item in the Components section on the left you will find a green block with the name of the item, like “Food1” at the bottom of the list of blocks that you can use. Drag and attach those food blocks (Food 1 through Food 10) to the Items openings on our Add Items block.



5. **Change your collide block:** Remove the = blocks and Set Visible blocks from your If Then Else block. Then, remove the Else part by clicking the blue button on the If Then block and dragging the Else out. Then, from Lists, drag an Is in List? block and attach it to your If Then block.



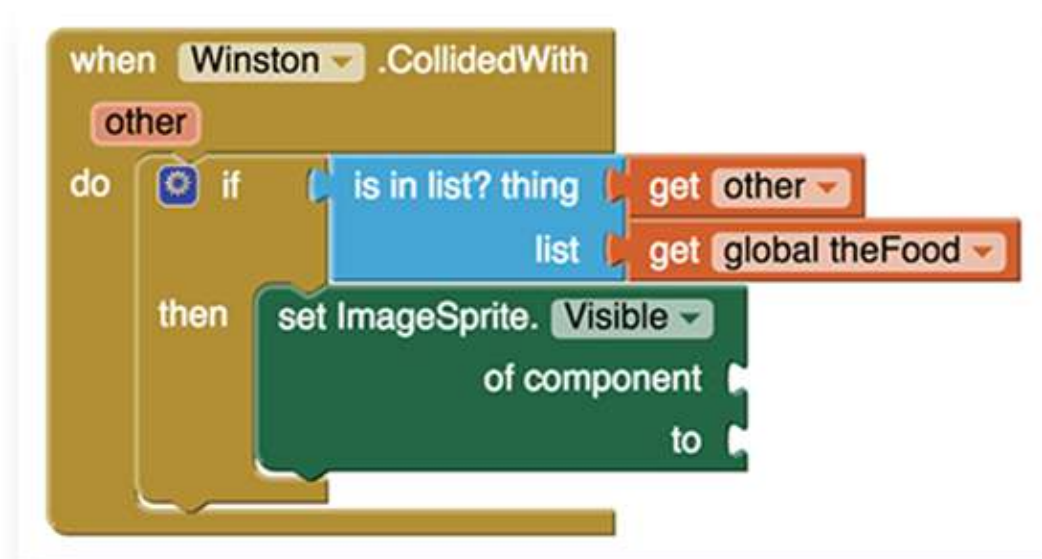
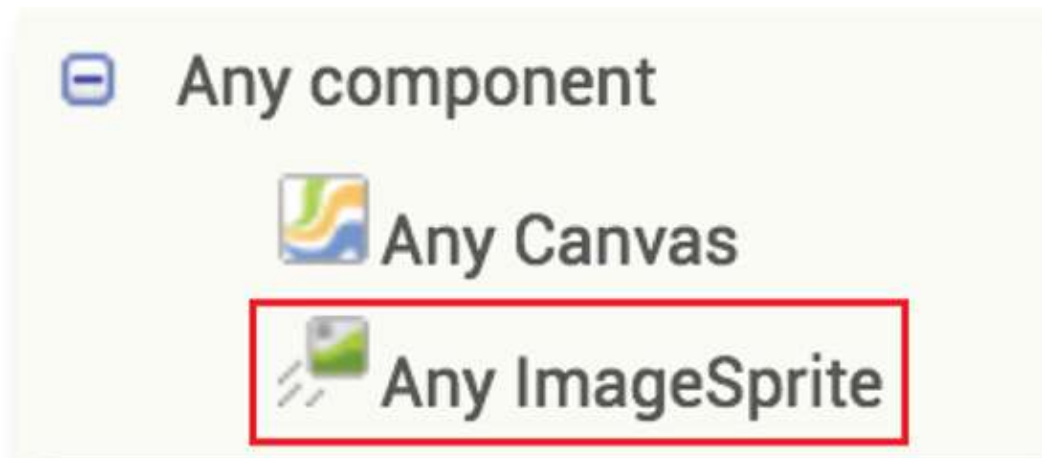
6. Add the list to iterate through: From Variables drag two Get blocks to your Is in List? block. Set one to be “Other” and one to be “Global theFood.”



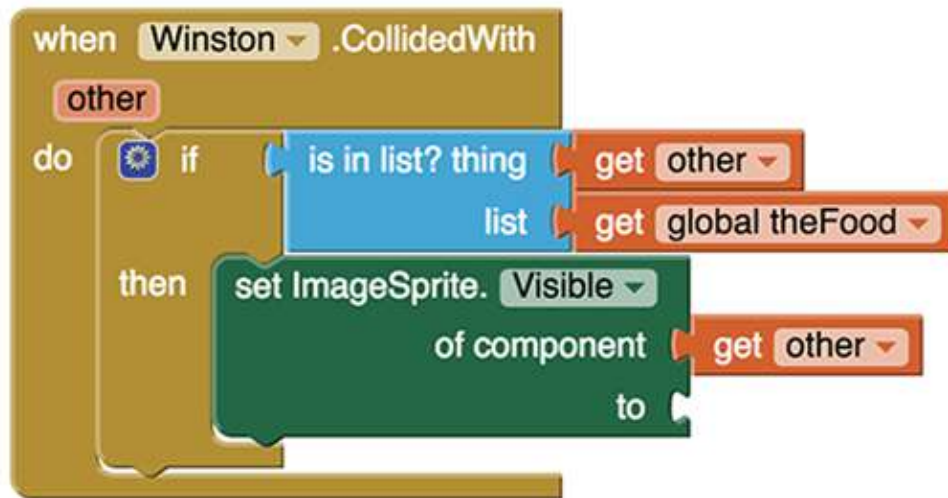
The Is in List block checks to see whether a certain thing — like Other, which refers to whatever Winston collides with — is in a

certain list. So this code checks to see whether Winston collided with a food item, since food is the only thing in the “theFood” list.

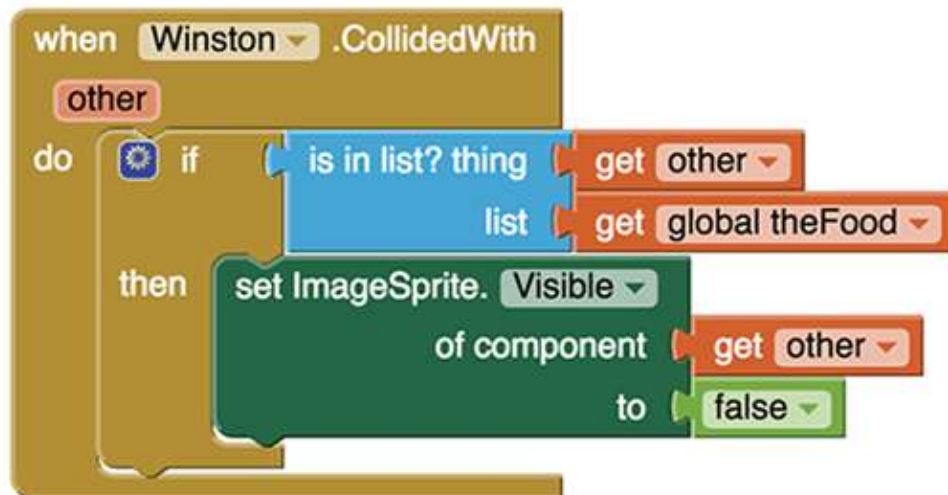
7. In your Blocks section, scroll to the bottom to find Any Component and click on Any ImageSprite. Add a Set ImageSprite. Visible block into your If Then block.



8. From Variables drag a Get block to the Component section of your Visible block.



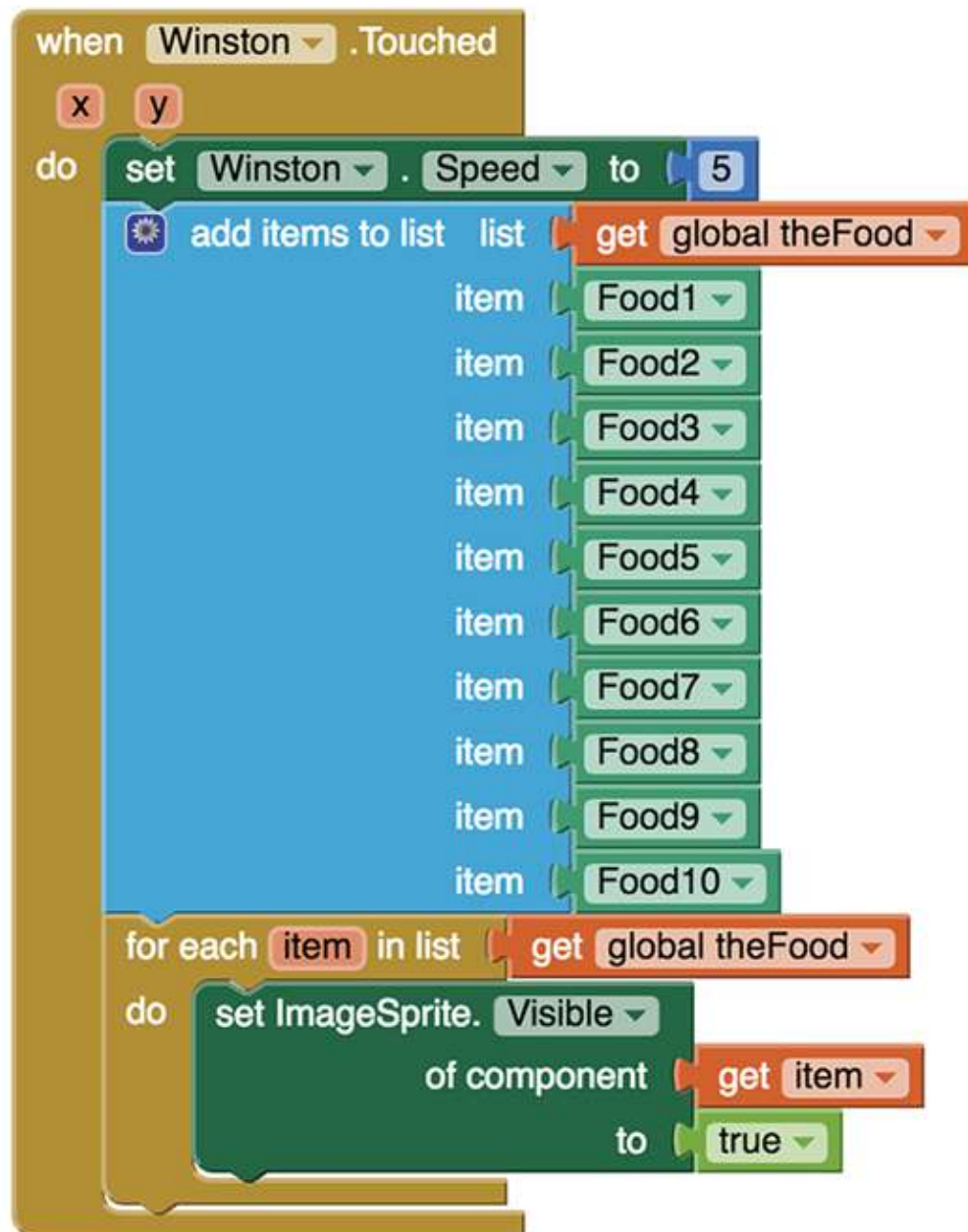
9. From Logic drag a False block to the To section of your Visible block.



10. From Control drag a For Each Item in List block into your When Winston.Touched block.



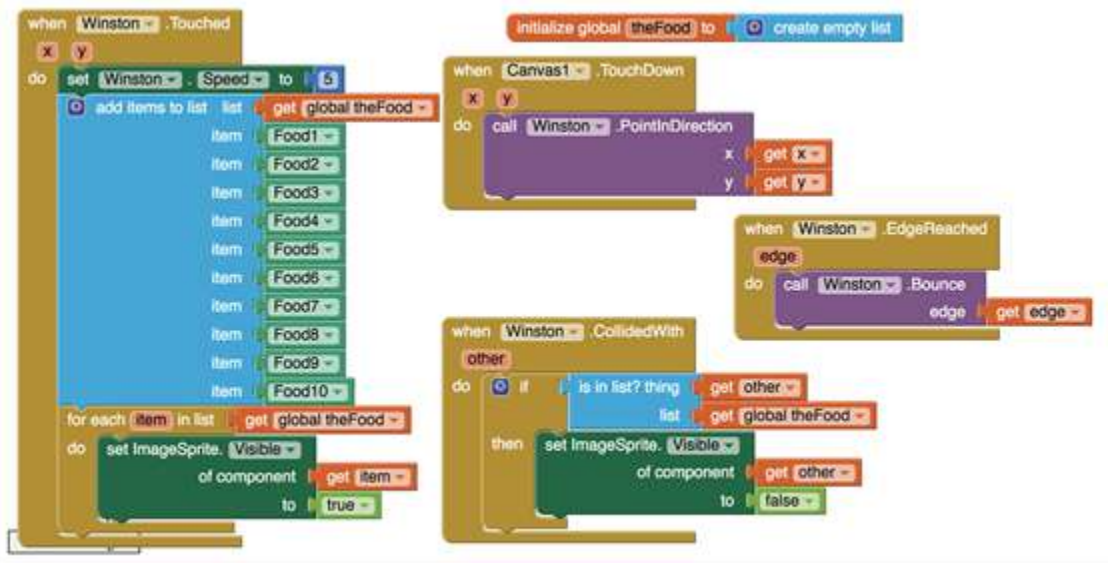
11. Set the list to “Global theFood” and change each item’s Visible property to True.



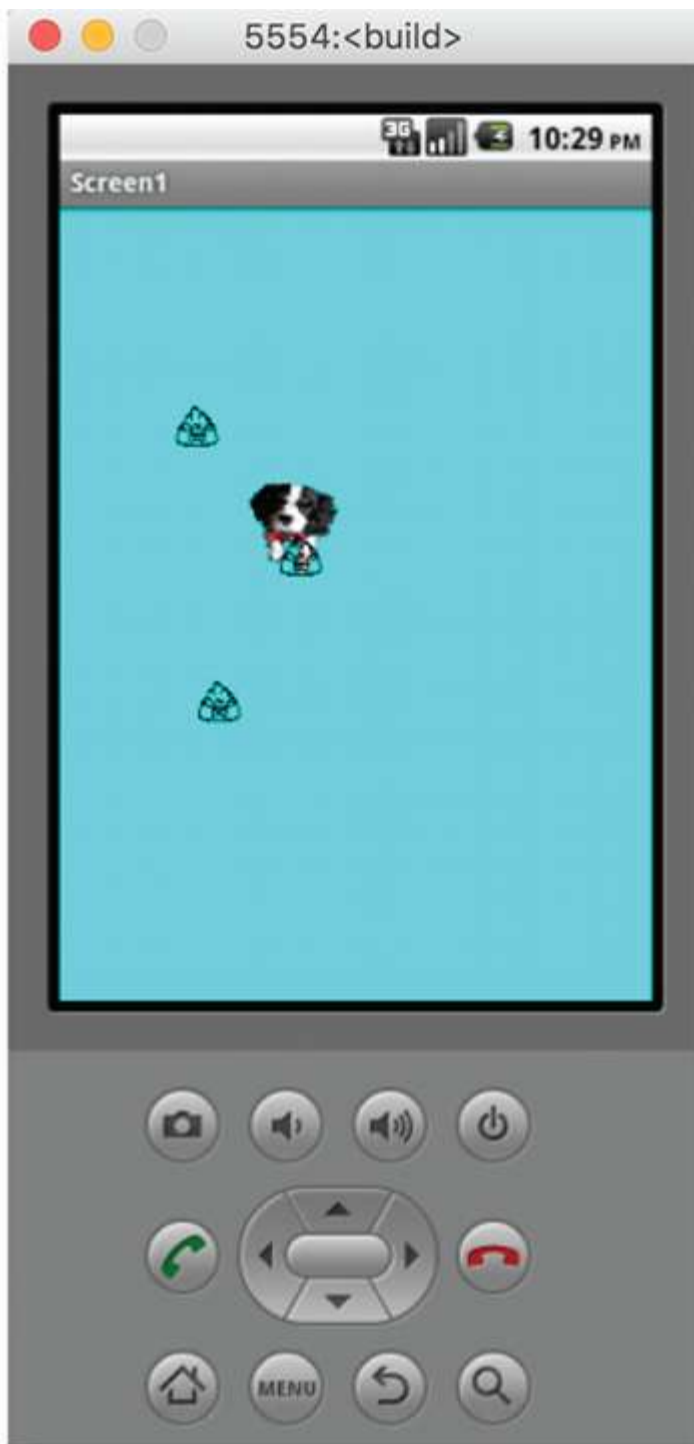
Whew! That was a lot of complicated steps! You should spend some time reviewing those steps carefully, so you can be sure you know everything that was going on there. And check back at thewecan.zone for updates and videos explaining more!

TEST YOUR NEW GAME

Now it's time to test your app. Your code should look like this:



When you run your app, you should be able to make Winston run around and collect all the food.



You have to click Winston to get him to start moving and to make all of the food visible again!



Because this app is getting bigger, it may take a little longer to load up in your emulator or on your Android device. Be patient!

MAKE YOUR GAME RANDOM

Finally, to make the game a bit more difficult, you can make the food appear in random places each time.

ADD A START AND RESET BUTTON

Before you add randomness, you need to add Start and Reset buttons to make sure that everything happens when it should. Just follow these steps:

1. **Go to Designer and add a HorizontalArrangement into the Layout section. Add it above your Canvas. Make sure the Width is Fill Parent and the Height is Automatic.**



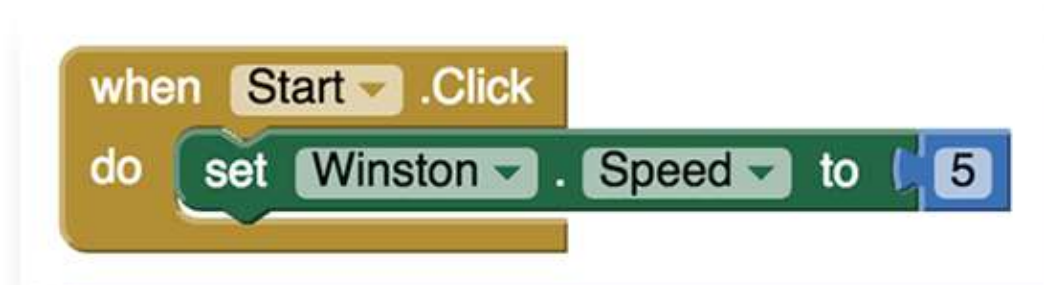
2. **Add two buttons to the HorizontalArrangement, one named “Start” with the word *Start* on it, and one named “Reset” with the word *Reset* on it.**



3. Go back to Blocks and code your Start button: From Start drag a When Start.Click block into your viewer.



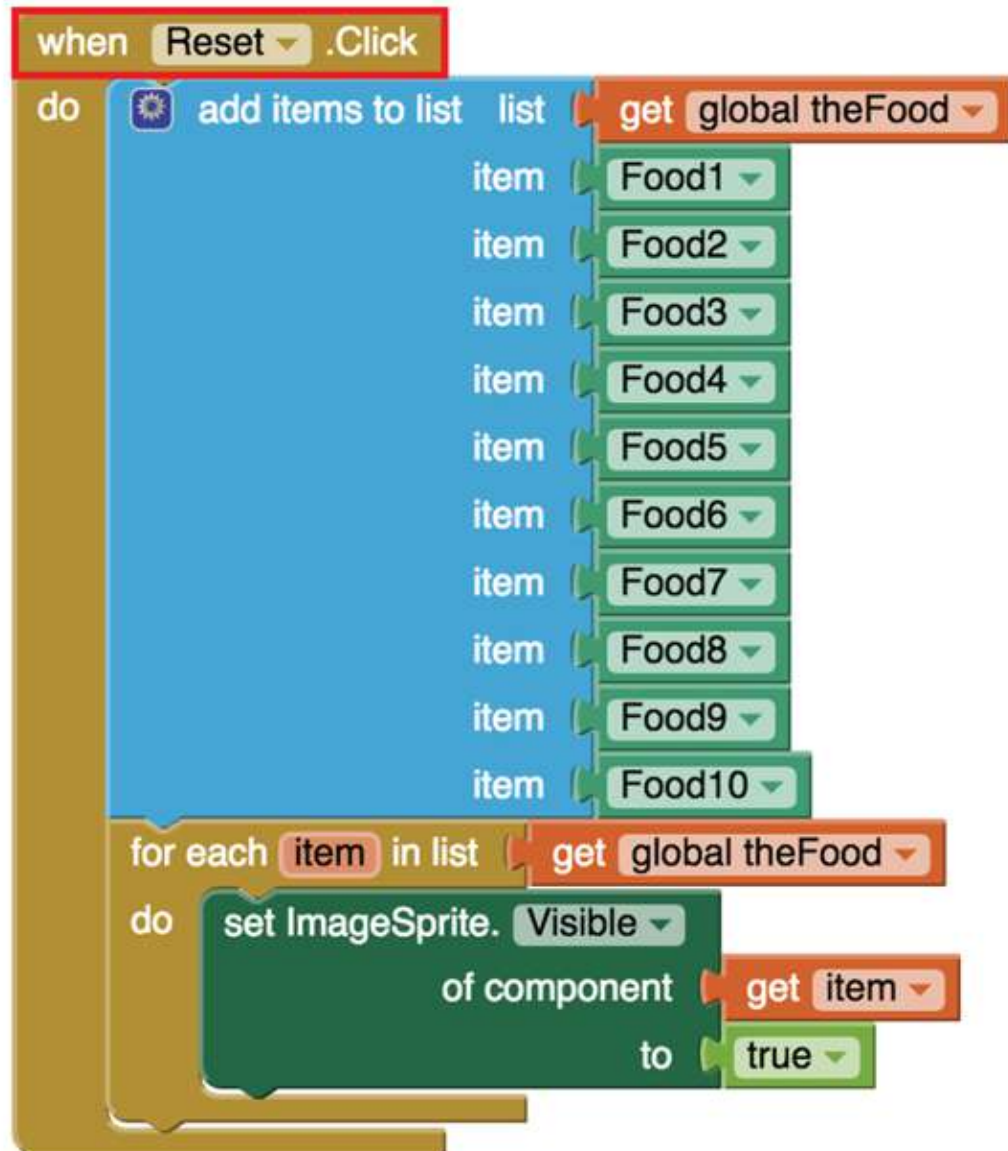
4. Move the Set Winston.Speed block from When Winston.Touched to When Start.Click.



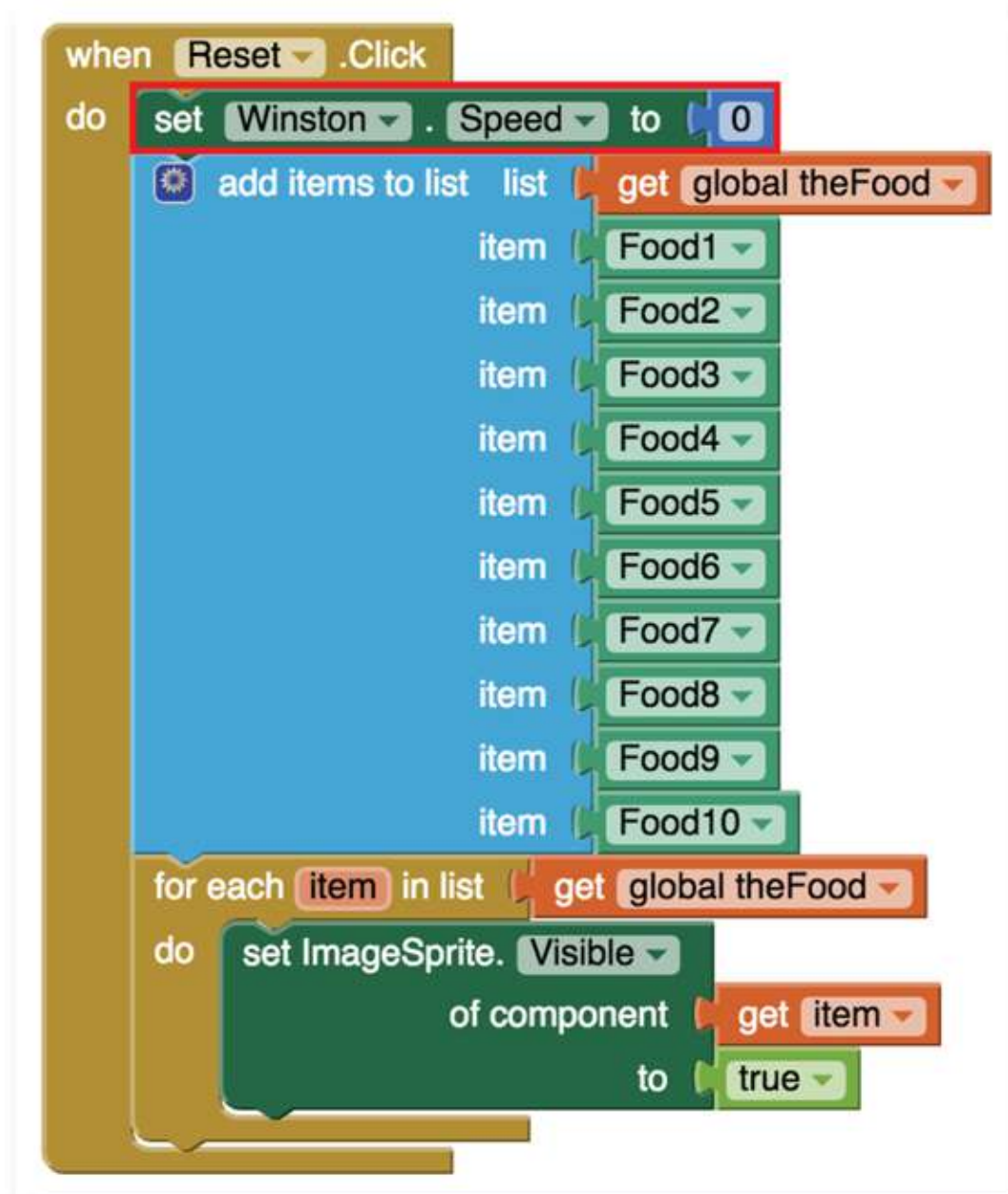
5. From Reset, drag a When Reset.Click block into your viewer.



6. Move the rest of the blocks inside of When Winston.Touched to When Reset.Click.



7. Your When Winston.Touched block should be empty now. Delete it.
8. Finally, from Winston drag a Set Winston.Speed block into the When Reset.Click block. Set Winston's Speed to 0.



TEST YOUR START AND RESET BUTTONS

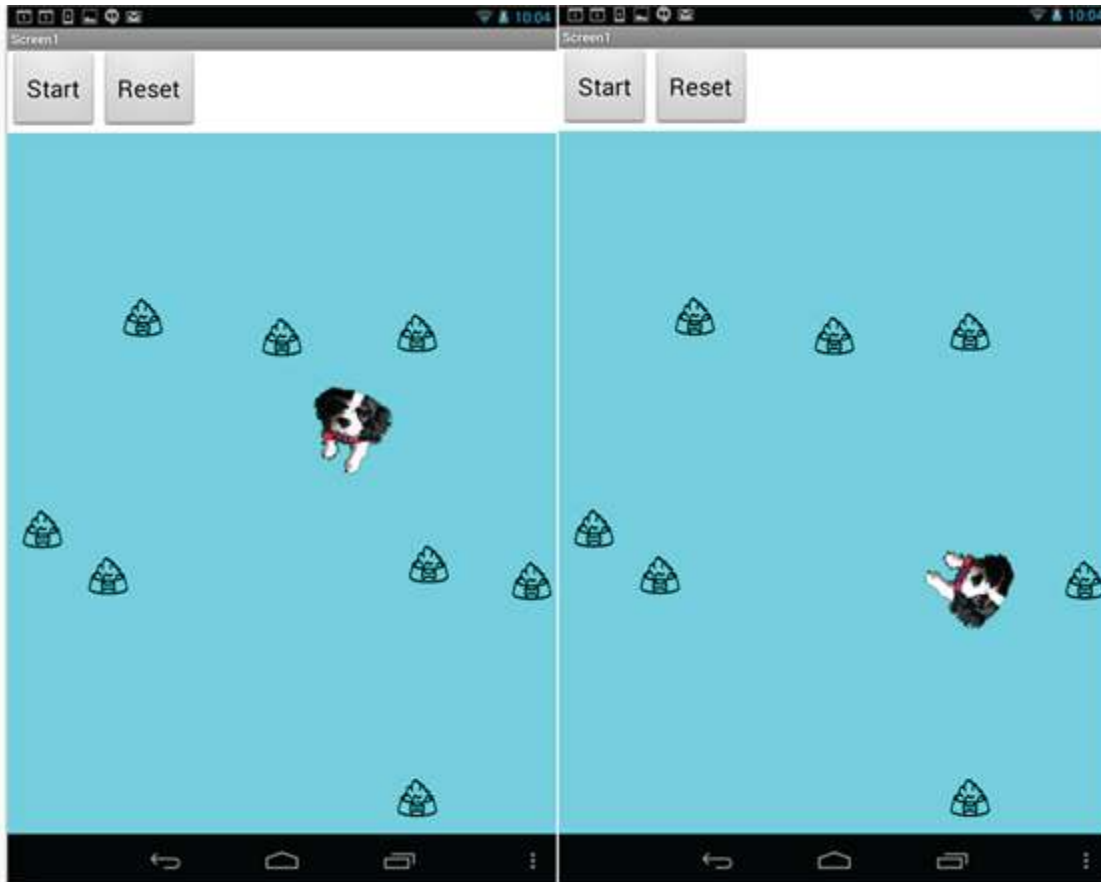
Now that you have your two buttons, test them out!



When you run your app on your emulator or Android device, you should click the Reset button first, and then click Start. Clicking

Reset first ensures that all of the food gets put into the list that you made.

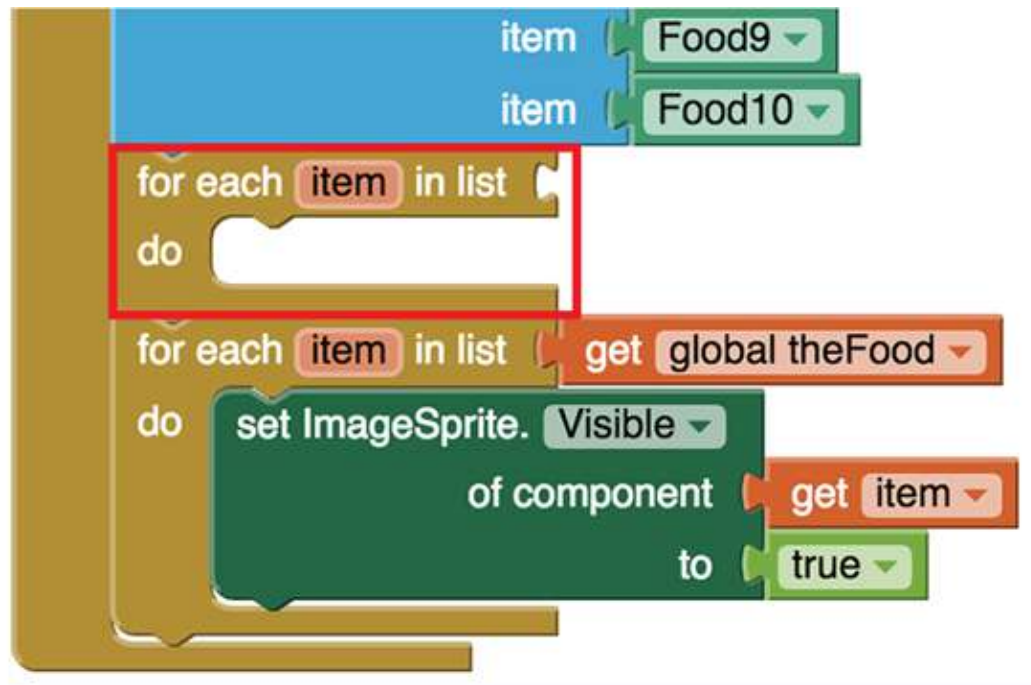
Make sure that when you click Reset, Winston stops and all the food reappears, and that when you click Start, Winston starts moving.



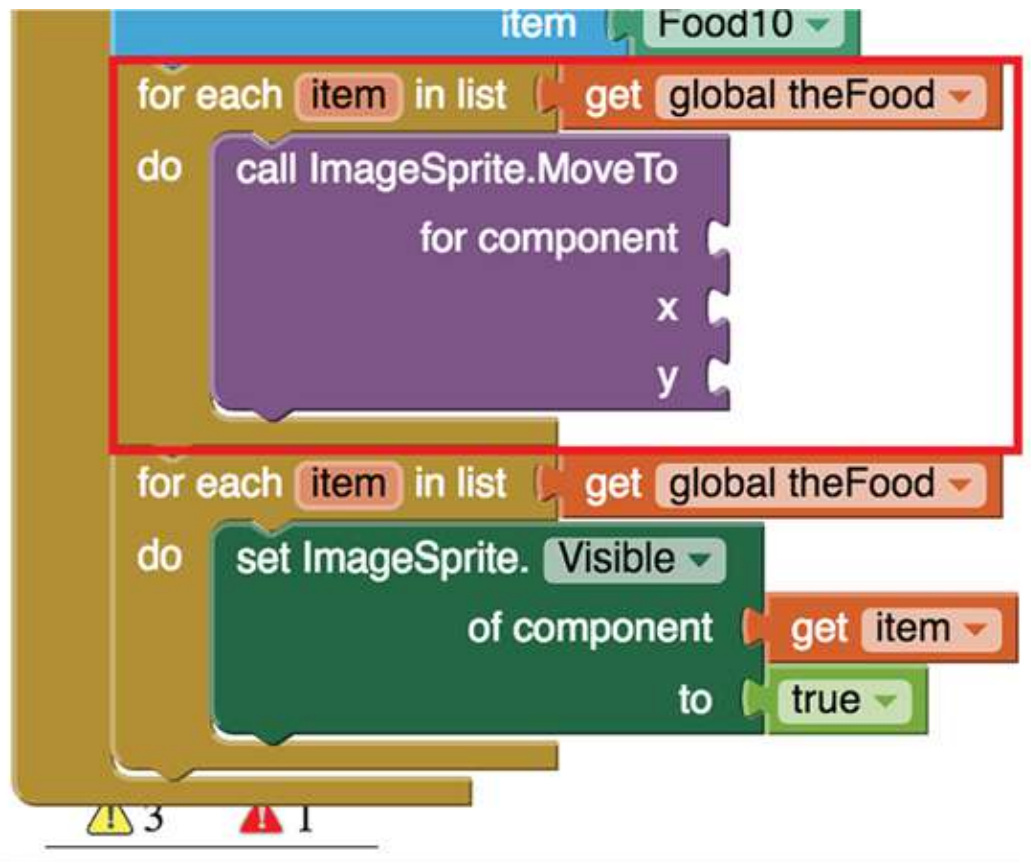
PUT YOUR FOOD IN RANDOM PLACES

Now that you are an expert at iterating through lists (doing something to everything in a list), this next section will be simple! Just follow these steps:

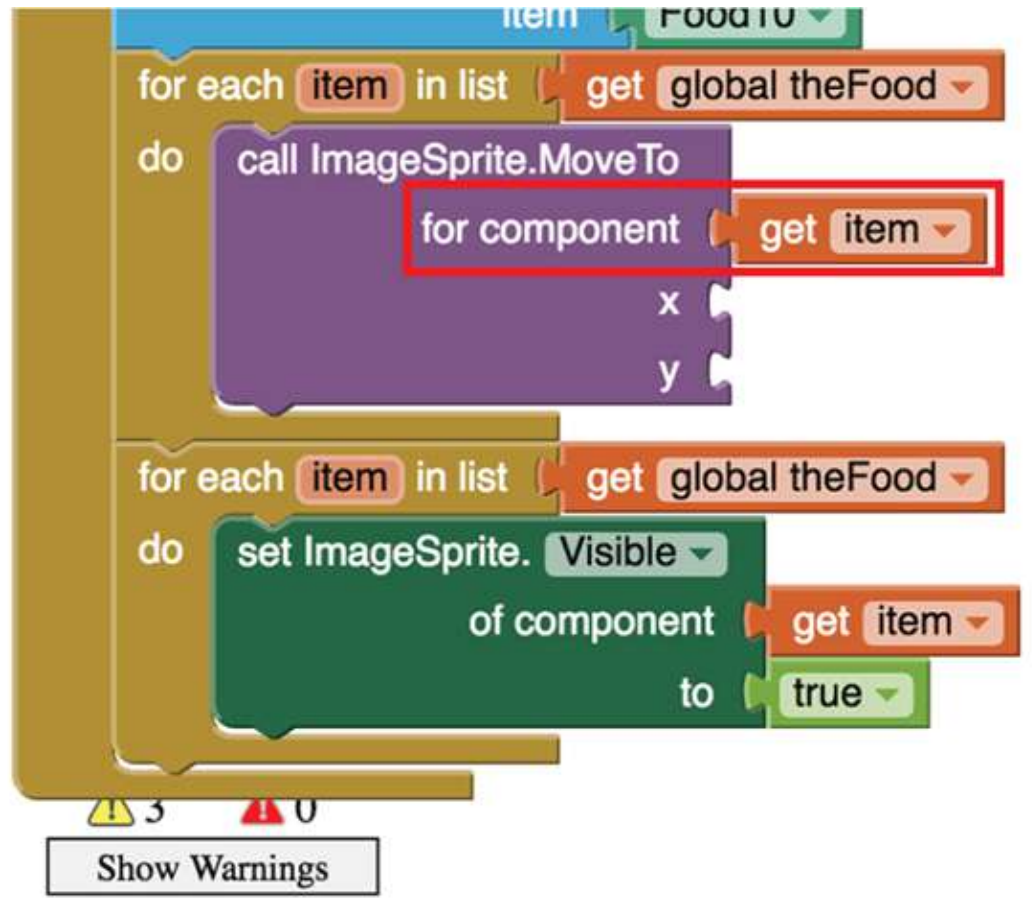
1. **From Control drag another For Each Item in List into your When Reset.Click block. This new block must be placed after you add the items to the list, but before you make them visible.**



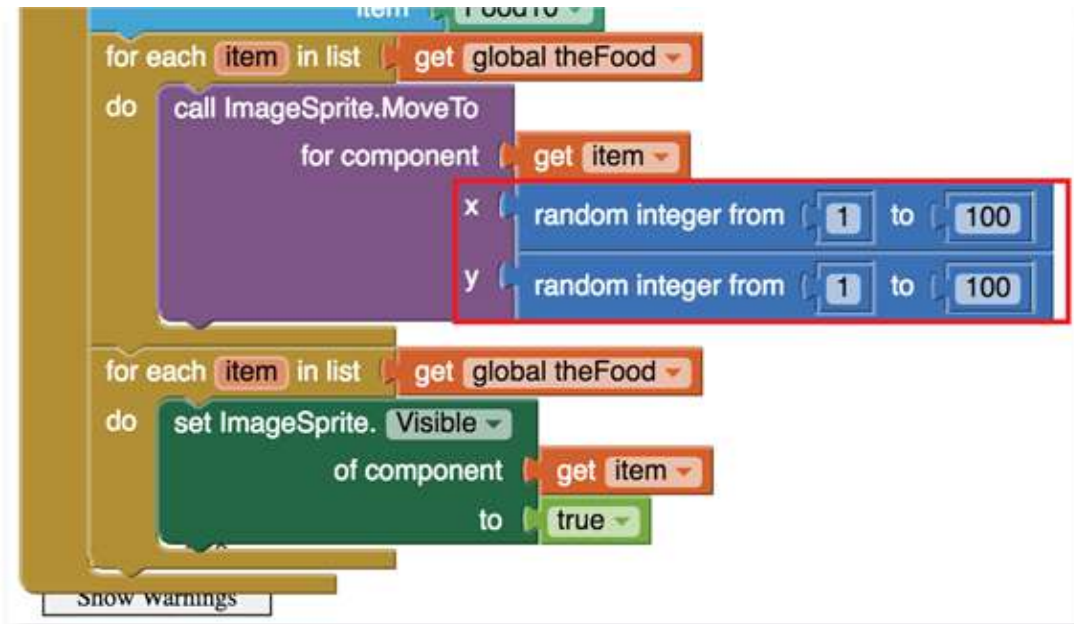
2. Add a Get Global theFood block to the For Each Item in List block. Under Any Component, click Any ImageSprite and drag a Call ImageSprite.MoveTo block into the For Each Item in List block.



3. Add a Get block from Variables to the component of your MoveTo block, and set it to Item.

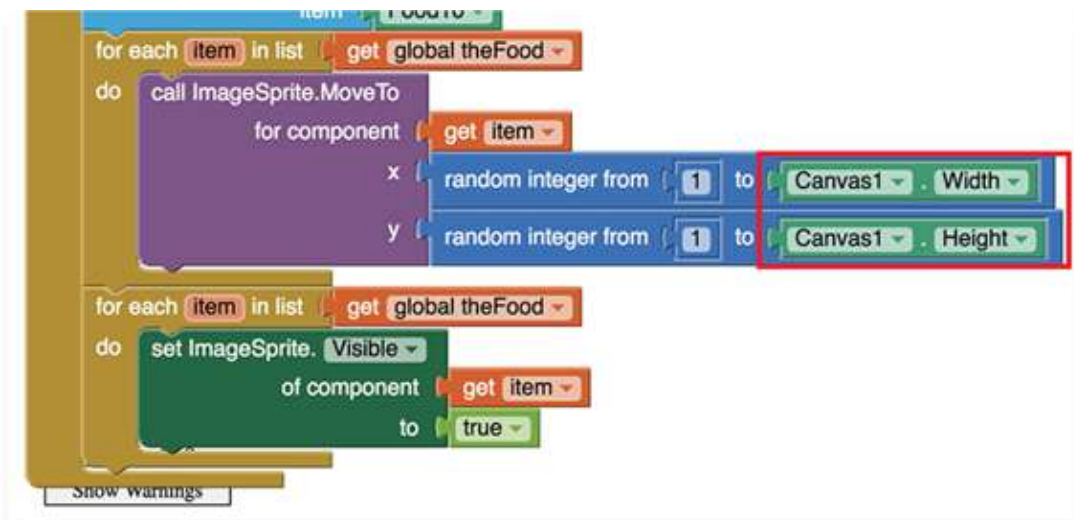


4. From Math, drag two Random Integer From blocks and put them in the X and Y openings of your MoveTo block.

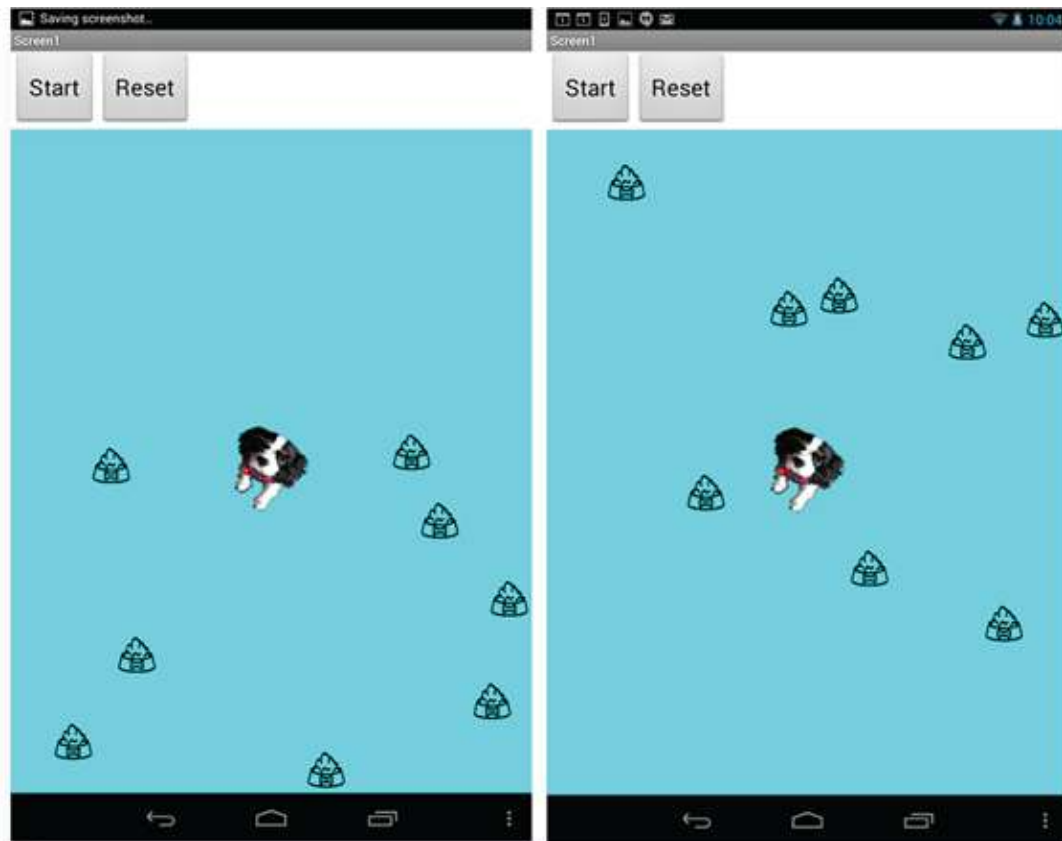


5. Delete the 100 blocks from both Random Integer From blocks and replace them with Canvas1.width and Canvas1.height blocks.

You can find Width and Height by selecting Canvas1.

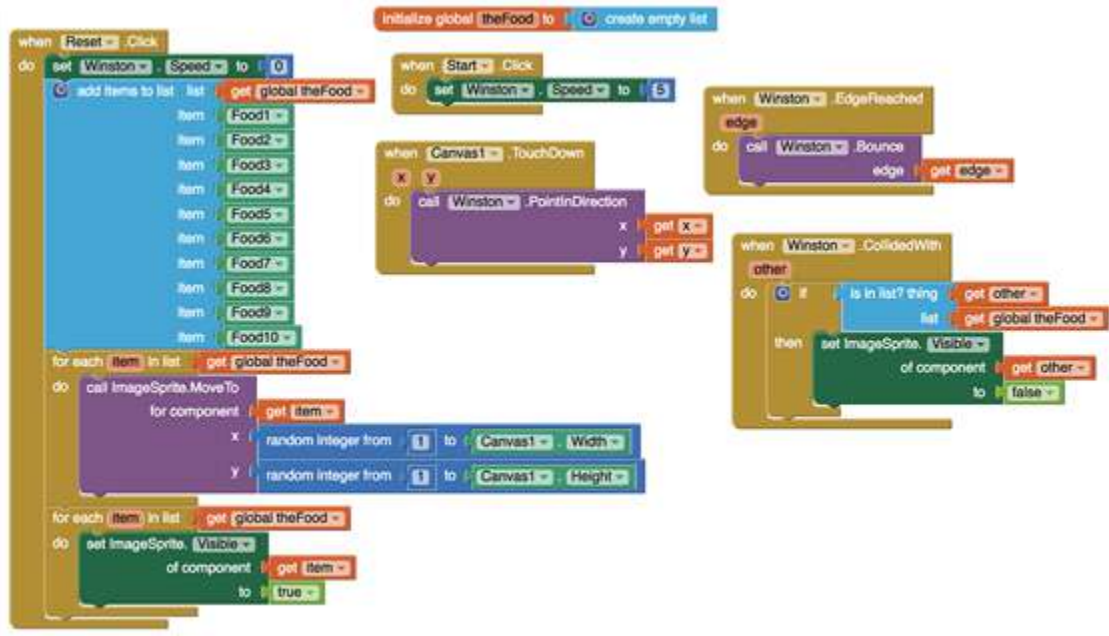


6. Now, each time you click Reset, your ten food items will appear in different places within the Canvas!



Wow! Congrats! You have made a pretty awesome mobile app! From here, you can even add more elements to your app, like a clock timer, or random blockades that make Winston move slower or faster. Try it out!

Here is the completed code for your Feed Winston game:



You can also play my version, which has different difficulty levels and a timer, by downloading the app from www.thewecan.zone/mobile-apps. Download it onto your Android device, open it, and install it. Then you should be able to play!

KEEP BUILDING

Now that you have built four different apps, it's time to put your skills to good use! Keep building and trying things out! Try using new components that were not introduced in this book, like TextToSpeech, VideoPlayers, or even sensors like the Pedometer (which tracks how many steps you take!).

You can keep getting ideas by visiting thewecan.zone/mobile-apps, because we will continue to make tutorials and YouTube videos to teach you even more about making mobile apps!



Coding can be difficult, and sometimes software doesn't always work. If you get stuck, just take a break, step away, and come back when you're able to concentrate a bit more. It's always good to ask

for help, or even just to ask someone to listen to you explain your problem. Sometimes saying your problem out loud helps you figure out the answer on your own!

Happy coding!

ABOUT THE AUTHOR

Sarah Guthals received her PhD from UCSD in computer science, specializing in CS education, in 2014. During graduate school, she built the beta version of CodeSpells, a 3D immersive video game designed to teach children to code through playing a wizard and writing “spells.” She went on to co-found ThoughtSTEM, a company that builds software (e.g., LearnToMod), curricula, and pedagogies for teaching children to code and empowering K–12 teachers to teach their students. She has written three books about Minecraft, launched a Coursera and edX course for teachers interested in teaching coding, was recently named in Forbes 30 under 30 in Science, and founded We Can — a company dedicated to encouraging *all* kids to do anything. Her passion is making coding accessible to everyone, with the goal of making it a basic literacy.

DEDICATION

I would like to dedicate this book to my close friends and family members who have supported me, not only in writing this book, but in becoming who I am today. I'd like to specifically dedicate this book to Adrian Guthals, who has always helped me to see that with passion and dedication I can really do anything I want.

AUTHOR'S ACKNOWLEDGMENTS

I would like to acknowledge all of the hard work that went into making Android and App Inventor — without these, making mobile apps would be so much harder. I would also like to acknowledge the teachers and parents around the world who have recognized the importance of coding in teaching so many valuable lessons to our next generation of makers.

PUBLISHER'S ACKNOWLEDGMENTS

Senior Acquisitions Editor: Amy Fandrei

Project Editor: Christopher Morris

Copy Editor: Christopher Morris

Editorial Assistant: Serena Novosel

Senior Editorial Assistant: Cherie Case

Technical Reviewer: Connor Morris

Production Editor: Selvakumaran Rajendiran

WILEY END USER LICENSE AGREEMENT

Go to www.wiley.com/go/eula to access Wiley's ebook EULA.

Made by the people who make the **for dummies** books!

BUILDING A *Mobile App*

Design
and
Program
Your Own App!

Sarah Guthals, Ph.D.
Social Entrepreneur and Engineer



BUILDING A
***Mobile App:
Design and
Program Your
Own App!***

by Sarah Guthals, Ph.D.



WILEY

BUILDING A MOBILE APP: DESIGN AND PROGRAM YOUR OWN APP!

Published by
John Wiley & Sons, Inc.
111 River Street
Hoboken, NJ 07030-5774

www.wiley.com

Copyright © 2017 by John Wiley & Sons, Inc., Hoboken, NJ

Published simultaneously in Canada

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States

Copyright Act, without the prior written permission of the Publisher.

Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at

<http://www.wiley.com/go/permissions>.

Trademarks: Wiley, For Dummies, the Dummies Kid logo, Dummies.com, and related trade dress are trademarks or registered trademarks of John Wiley & Sons, Inc. and may not be used without written permission. All other trademarks are the property of their respective owners. John Wiley & Sons, Inc. is not associated with any product or vendor mentioned in this book.

<p><u>LIMIT OF LIABILITY/DISCLAIMER OF WARRANTY:</u> THE PUBLISHER AND THE AUTHOR MAKE NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE ACCURACY OR COMPLETENESS OF THE CONTENTS OF THIS WORK AND SPECIFICALLY DISCLAIM ALL WARRANTIES, INCLUDING WITHOUT LIMITATION WARRANTIES OF FITNESS FOR A PARTICULAR PURPOSE. NO WARRANTY MAY BE CREATED OR EXTENDED BY SALES OR PROMOTIONAL MATERIALS.</p>

THE ADVICE AND STRATEGIES CONTAINED HEREIN MAY NOT BE SUITABLE FOR EVERY SITUATION. THIS WORK IS SOLD WITH THE UNDERSTANDING THAT THE PUBLISHER IS NOT ENGAGED IN RENDERING LEGAL, ACCOUNTING, OR OTHER PROFESSIONAL SERVICES. IF PROFESSIONAL ASSISTANCE IS REQUIRED, THE SERVICES OF A COMPETENT PROFESSIONAL PERSON SHOULD BE SOUGHT. NEITHER THE PUBLISHER NOR THE AUTHOR SHALL BE LIABLE FOR DAMAGES ARISING HEREFROM. THE FACT THAT AN ORGANIZATION OR WEBSITE IS REFERRED TO IN THIS WORK AS A CITATION AND/OR A POTENTIAL SOURCE OF FURTHER INFORMATION DOES NOT MEAN THAT THE AUTHOR OR THE PUBLISHER ENDORSES THE INFORMATION THE ORGANIZATION OR WEBSITE MAY PROVIDE OR RECOMMENDATIONS IT MAY MAKE. FURTHER, READERS SHOULD BE AWARE THAT INTERNET WEBSITES LISTED IN THIS WORK MAY HAVE CHANGED OR DISAPPEARED BETWEEN WHEN THIS WORK WAS WRITTEN AND WHEN IT IS READ.

For general information on our other products and services, please contact our Customer Care Department within the U.S. at 877-762-2974, outside the U.S. at 317-572-3993, or fax 317-572-4002. For technical support, please visit

<https://hub.wiley.com/community/support/dummies>.

Wiley publishes in a variety of print and electronic formats and by print-on-demand. Some material included with standard print versions of this book may not be included in e-books or in print-on-demand. If this book refers to media such as a CD or DVD that is not included in the version you purchased, you may download this material at

<http://booksupport.wiley.com>. For more information about Wiley products, visit www.wiley.com.

Library of Congress Control Number: 2017934005

ISBN: 978-1-119-37642-2 (pbk); 978-1-119-37639-2 (ebk); 978-1-119-37638-5 (ebk)

Building a Mobile App: Design and Program Your Own App!

Table of Contents

COVER

INTRODUCTION: GET STARTED WITH MOBILE APPS

[About App Inventor](#)

[About This Book](#)

[About You](#)

[About the Icons](#)

PROJECT 1: BASIC MOBILE APP TOOLS

[The Software: App Inventor](#)

[Code On App Inventor](#)

PROJECT 2: MAKE AN APP ABOUT YOU

[Design Your App](#)

[Make the Skeleton of Your App](#)

[Code Your App](#)

PROJECT 3: MAKE A PHOTO-EDITING APP

[Set Up Your Photo-Editing App](#)

[Test Your App](#)

[Code Your Photo-Editing App](#)

PROJECT 4: MAKE A MOBILE GAME APP

[Decide on A Game](#)

[Set Up Your Game App](#)

[Make a Simple Game](#)

[Make a Re-Playable Game](#)

[Make Your Game Random](#)

[Keep Building](#)

ABOUT THE AUTHOR
END USER LICENSE AGREEMENT