Make Your Own Virtual Pet in Scratch!

Introduction

Welcome to Make Your Own Virtual Pet in Scratch! We are so glad you joined us today, and are excited to show you how to make your own game.

What is We Can Code IT

We Can Code IT is a non-profit organization that teaches girls and women about technology and engineering in a fun, creative way.

What is Scratch

Scratch is a fun, easy to use, programming language. Basically, it's like scripting your own play. You make actors perform scripts on a stage. You can make games, art, animations, tell stories, and so much more, using Scratch.

What is our goal?

Our goal is to have you learn some programming basics and have fun while doing it! The end result today will be that you have created your own "virtual pet" game using Scratch. Feel free to make it your own! Change pictures, add your own scripts to the actors, play around and explore!

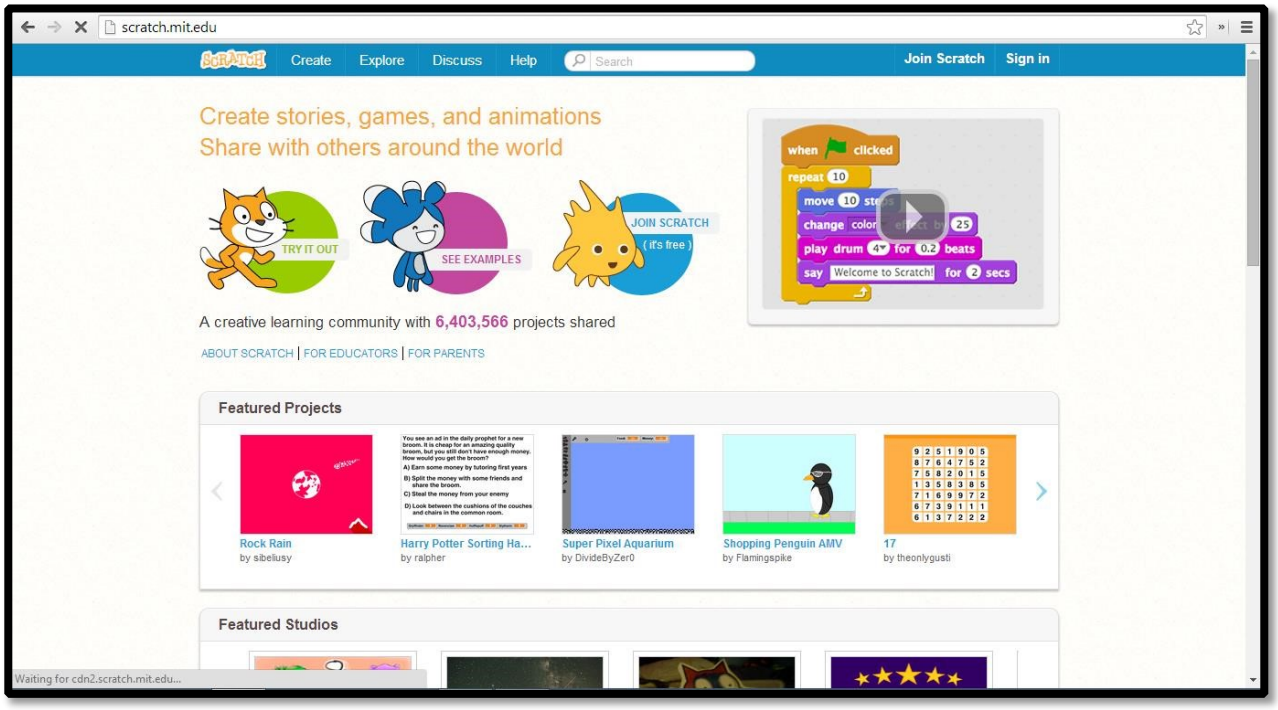
In programming there are often many ways to accomplish the same task. If you find a new way, great!

We also want to teach you about Computational Thinking, which means, you start thinking like a programmer. Don't worry about this, it will come naturally.

STEP 1: Open Scratch in your Browser.

Here is the URL (the web address): http://scratch.mit.edu

It will look like this



//Comment: Do you see this page?

If (yourAnswer = true) then

Say “HURRAY!”; Else

raiseYourHand();

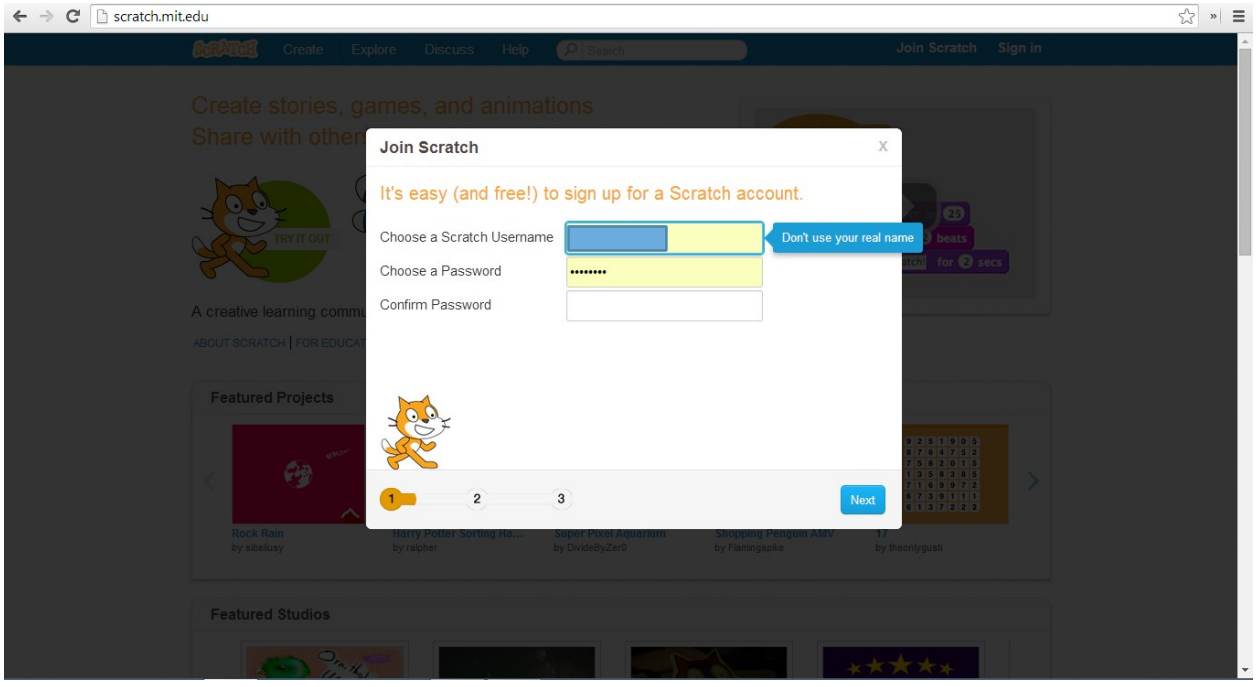
Step 2: Join Scratch so you can save your project!

Click on the Fish Picture to Join Scratch.

Fill out the username and password fields. Make these something memorable to you.

If you need help, then

```
raiseYourHand();
```



When you click "Next," you'll be asked for an **email address**, amongst other information like your gender, and birth month and year. Fill out those fields, and continue filling out the registration information until finished.

We're almost there! Let's pause for a moment and talk about the weird way I've been writing.

At this point, you've seen a few of these statements

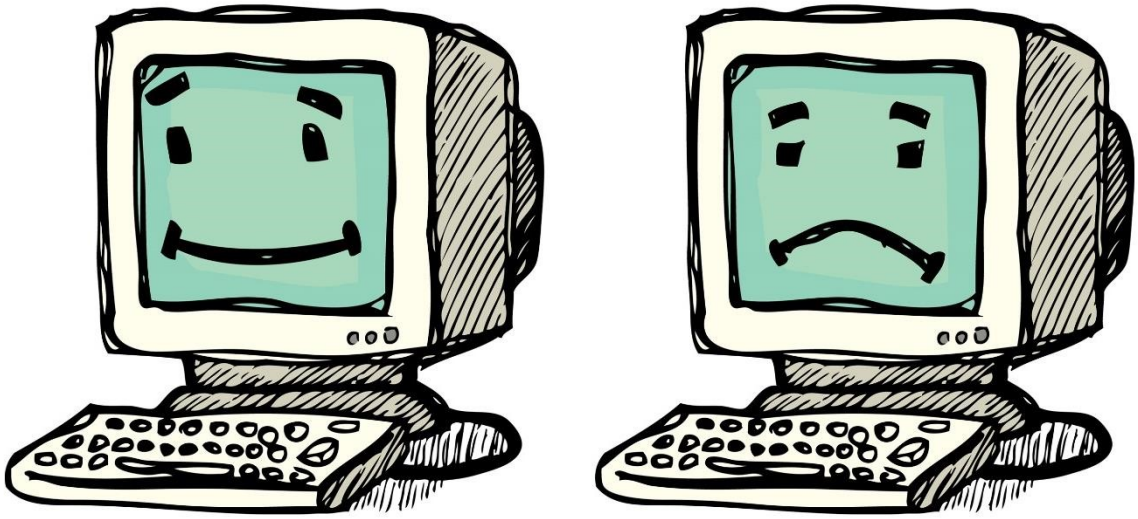
```
If(you'veNoticedTheseStatements =
true) then
        raiseYourHand();
Else
        keepYourHandDown();
```

What are some of the things you notice about these statements?

What does this mean to you? How would you say it in English?

Step 3: Writing “pseudocode”

Computers are not very bright, but people are. If you bought a computer from the store and it didn’t have an operating system, like Windows or Mac operating systems, then the computer would just sit there, looking at you with a blank stare.



A computer needs someone to tell it what to do. It needs very specific instructions!

That’s where you come in!

Let’s write down some instructions that a computer might understand. When you write this using English (or any other human language), it’s called ***pseudocode***.

Using pseudocode, let's tell the computer how a person raises her hand.

function raiseYourHand(){

//write your instructions here, between the curly braces.

}

Here is an example:

function raiseYourHand(){

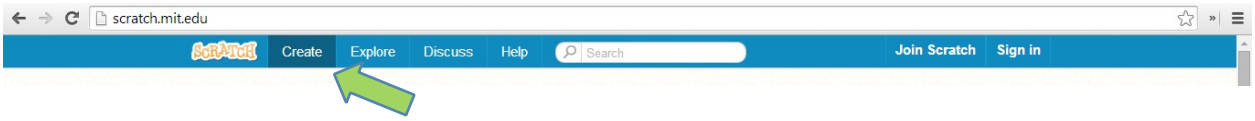
put my arm up, towards the sky; point fingers to the sky; if I am called on then put my arm down to a natural position;

}

Your version may differ. This is the fun of programming, you get to make your own recipes!

What did you notice about your pseudocode? How was it different than the example? Would you change anything?

Step 4: Create with Scratch

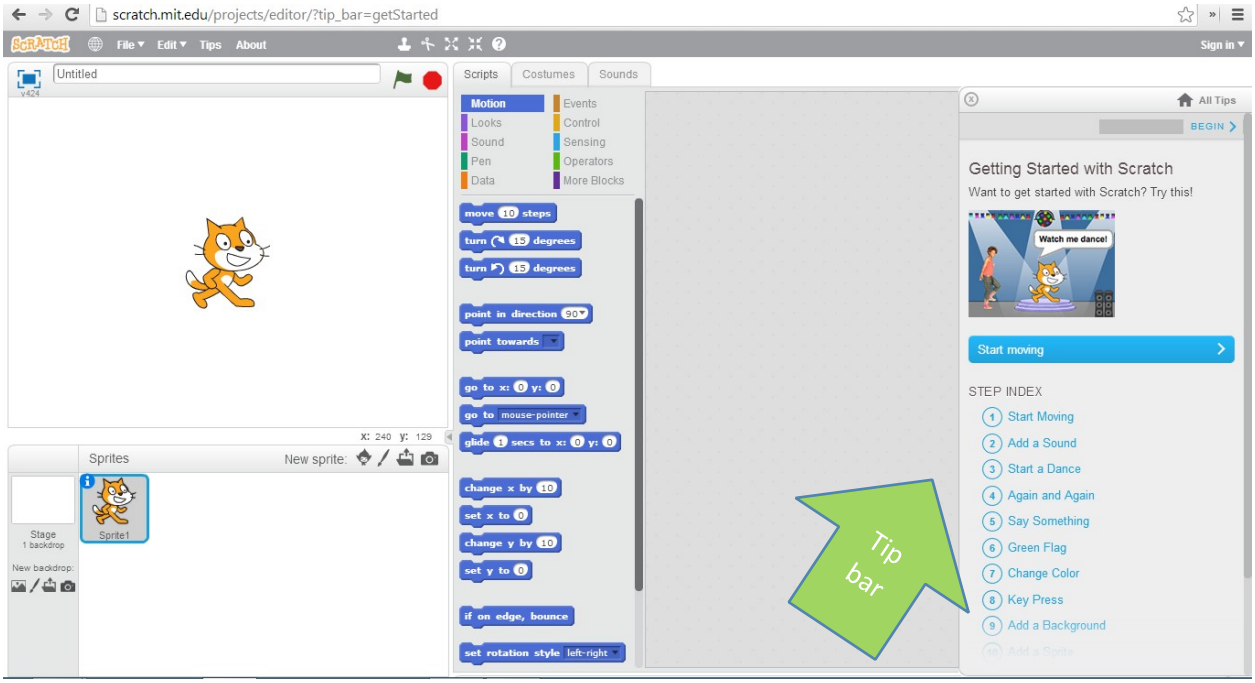


On the scratch web page, click “Create.”

Remember:

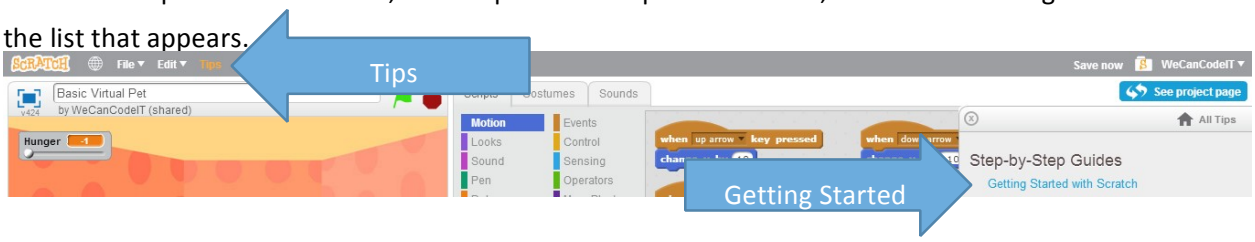
If (needHelp = true) then

raiseYourHand();



Notice the tip bar on the right. Go ahead and follow along for the next several minutes, and play with Scratch!

Note: If the tip bar doesn’t show, click “Tips” at the top of the screen, then click “Getting Started” from the list that appears.

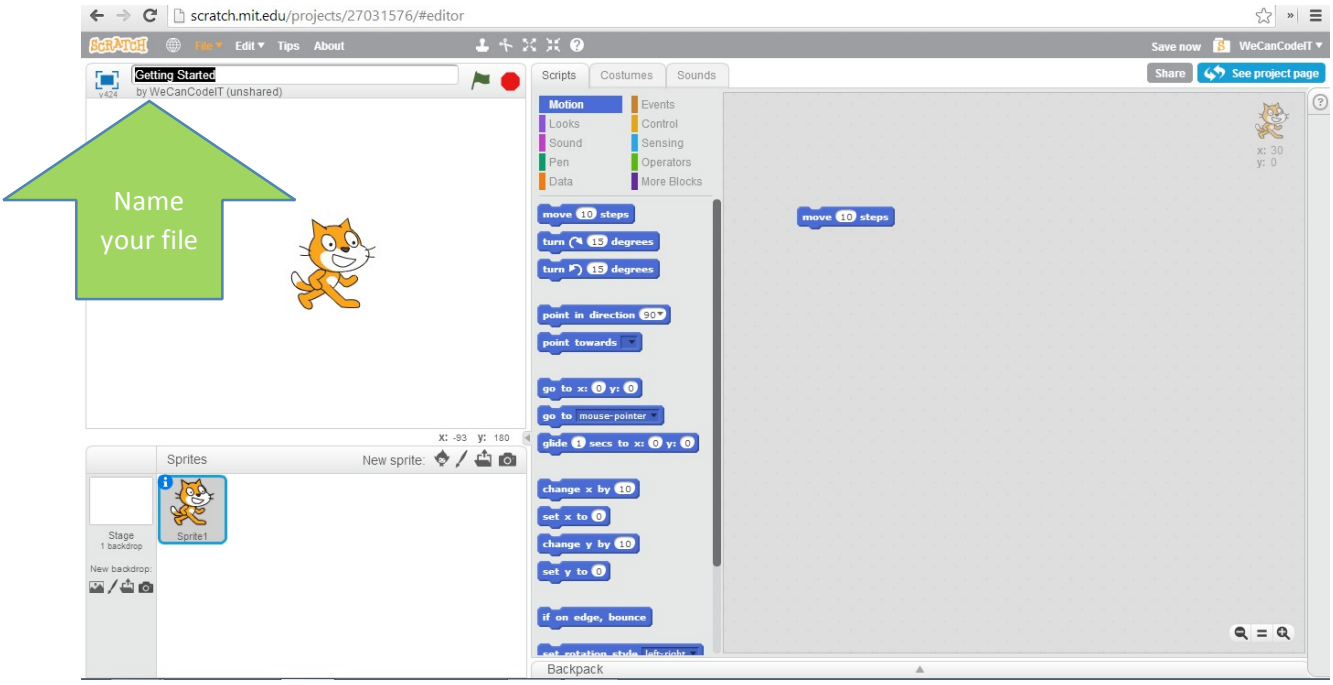


Step 5: What can you tell us about Scratch?

You've played around with Scratch, so what are some of the things you've noticed? Share with the class.

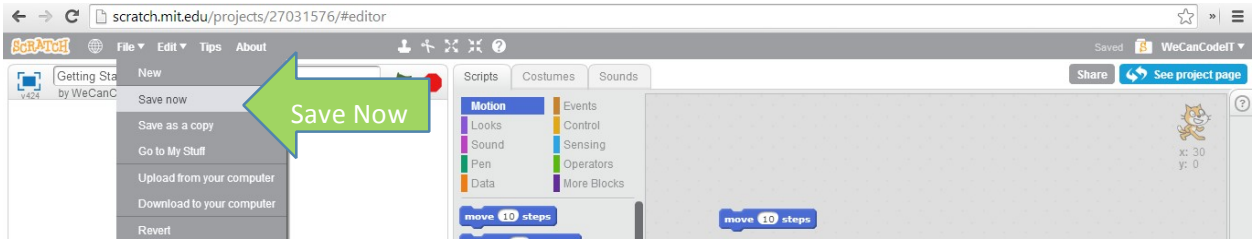
Did you notice colors, stages, actors, commands?

Step 6: Saving a Project:



Look at the arrow in the image, above.

Name your project something memorable, like "Getting Started."



Then Click File -> Save Now.

Here's some pseudocode explaining what we just did.

```
If (doneWithProject = true) then

        nameProject(); clickFile();
        clickSaveNow();
```

WeCan{Code}IT

Step 6: Create Your Virtual Pet – move cat around with arrows

Click File -> New

This starts a new project.

You'll see the cat,  called Sprite1 is selected, and we are on the Scripts tab.

Click Events under scripts.

Drag the  **block in to the scripts area.**

Change "space" to "up arrow" 

Step 6: Create Your Virtual Pet – move cat around with arrows (continued)

We are telling the computer to do something in this Scratch Program when the up arrow is pushed on your keyboard. This is our first step into the world of programming, and we are using Scratch as our programming language!

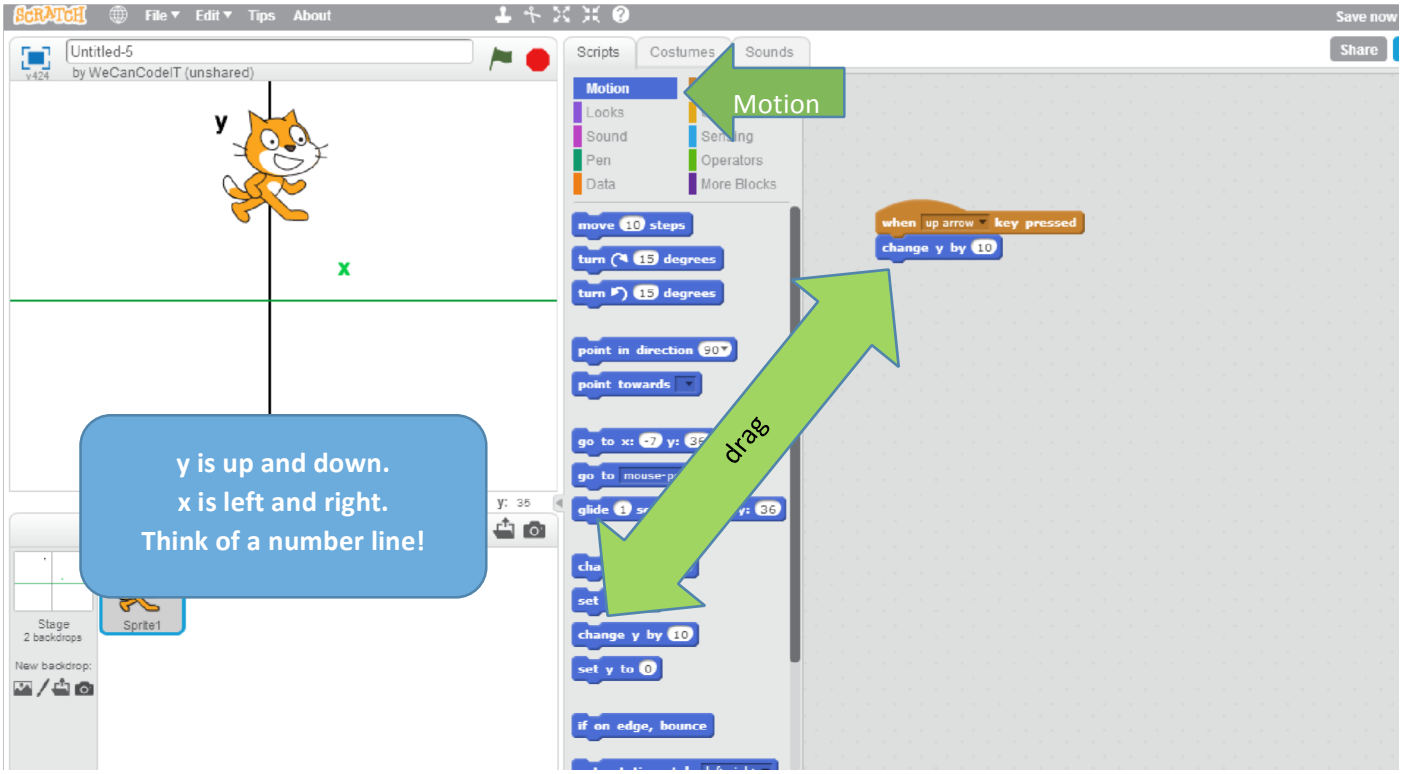
Let's continue!

Click Motion

Drag "change y by [10]" in to the scripts area,

Attach it underneath the "when [up arrow] key pressed" block.

Test! Click the up arrow on your keyboard and see what happens.



Step 7: Test and Debug!

The best programmers test their creations as they go along. It's a way to see if your instructions are correct, and catch any issues as they happen.

How would you test to see if your Scratch code is working?

Did you test it? Is it working?

If not working, raiseHand() and debug();

What is debugging

Errors in programming are called bugs. We want to catch the "bugs" in our code as soon as possible.

Debugging is sort of like getting rid of bugs in your house. Bugs mess things up. There's a funny story about how computer bugs were named!

As the story goes, pioneer computer programmer **Admiral Grace Murray Hopper** coined the phrase in the early 1940s, when her computer crashed. Searching for the cause of the problem, Admiral Hopper discovered the original computer bug, inside her computer, was a moth.

As she removed the moth, her coworker asked what she was doing.
"I'm debugging the machine," she answered. http://indianapublicmedia.org/amomentofscience/computer-bugs-and-amazing-grace/

How do you fix these bugs? You review your work to see if it's a typo or a simple mistake, like not joining related blocks together, in Scratch. You ask for help. You look things up in books or online. You take classes to learn more. You practice, practice, practice!

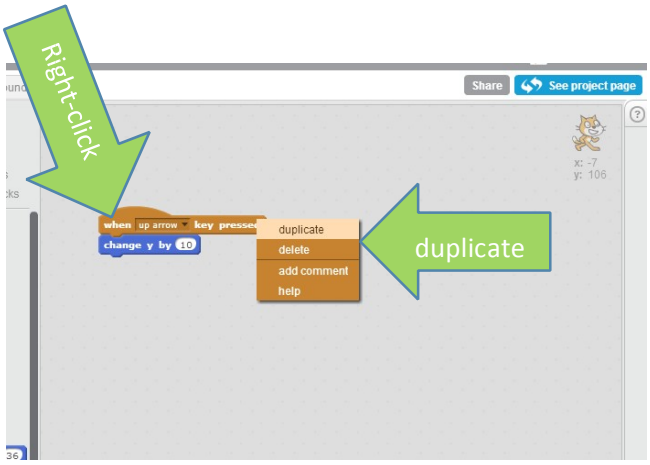
Step 8: Duplicating and Code Reuse in Scratch

Now that your up arrow is controlling the cat, moving her up (name that axis, X or Y?), let's reuse that set of blocks, that code, to create events for the right arrow, down arrow, and left arrow.

Right-click on the set of blocks that you want to copy.

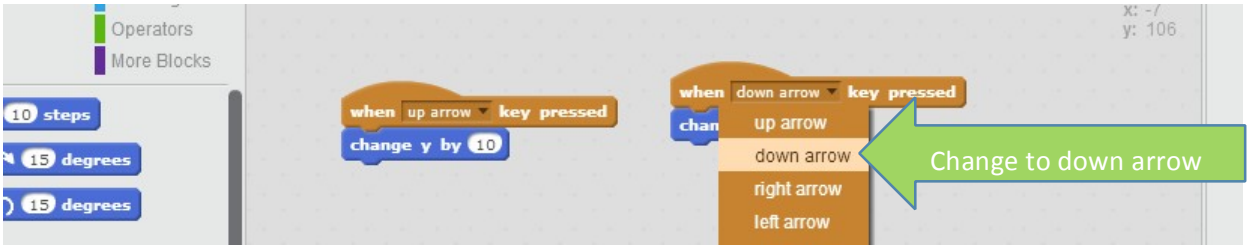
A menu will appear.

Select "duplicate" from the menu by left-clicking on "duplicate."



Drag the newly-formed set of blocks to an open space in the scripts window, then left-click to place it.

Now, click the drop-down on this duplicated set of blocks, to change when [up arrow] key pressed, to when [down arrow] key pressed.



Now Test and Debug!

Step 8: What is missing?

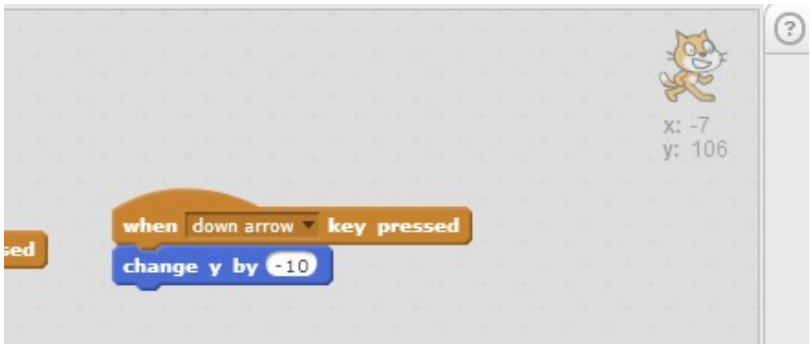
Did you test the down arrow?
What did you expect to happen? What *did* happen?

The cat still moved up when you pressed the down arrow, didn't she? Why?

How do we make her go down? *Hint: think of positive numbers as up and right, and negative numbers as down and left.*

Change the value of "change y by [10]" to the correct number [-10], under the duplicated block.

Test.



Step 9: Deleting Blocks and Considering X-axis

Duplicate the up arrow block again.

Drag it in to an available area of the scripts workspace.

Change "up arrow" to "right arrow"

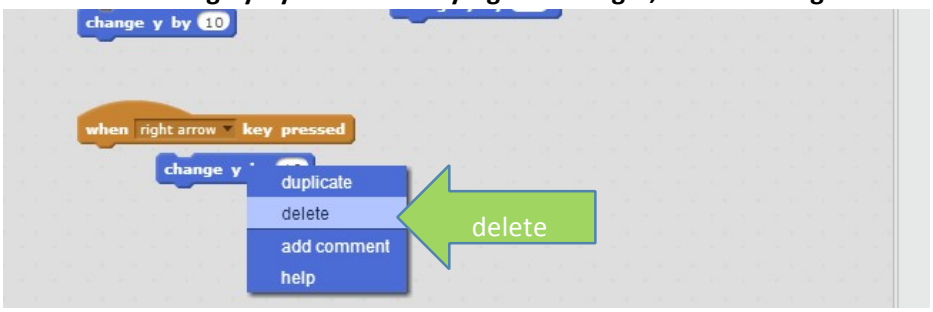
Test

Debug: what is the problem?

Y is up and down. We don't want to change the Y position when we click the right arrow, we want to change the X position.

Let's disconnect the "change y by . . . " motion block from the "when right arrow key pressed" event by dragging it away from that block.

Delete the "change y by . . ." block by right-clicking it, and choosing "delete."



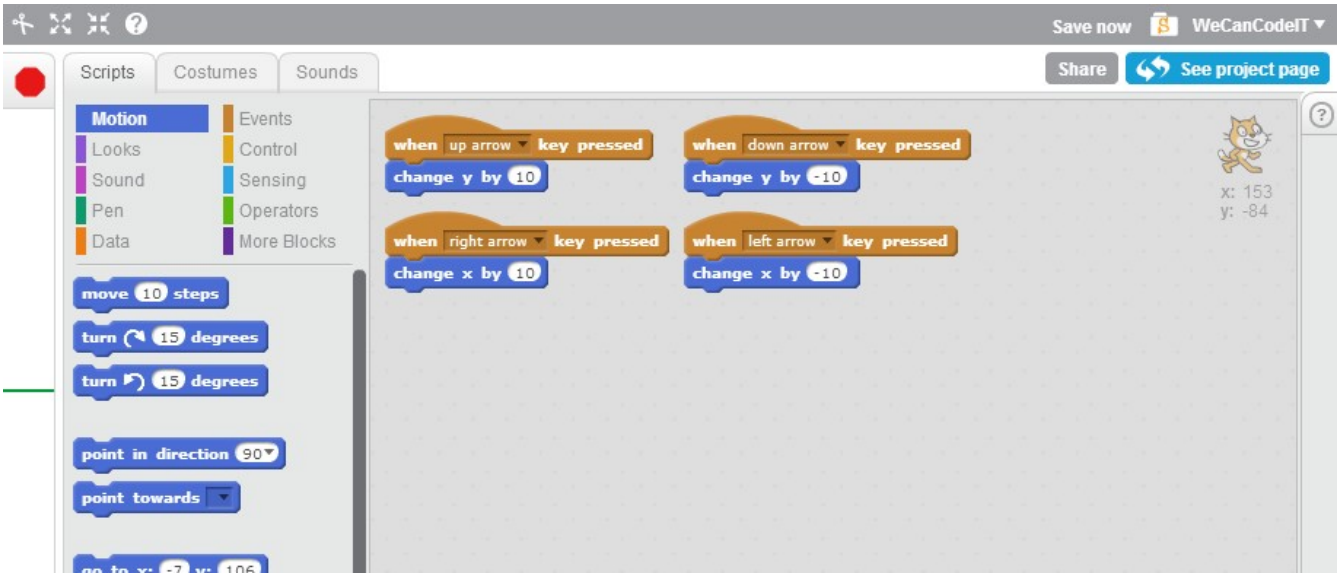
Now, attach the "change x by . . ." block under the "when right arrow key pressed" event, and leave the number at 10.

Test!

Step 10: On Your Own

Your goal is to create the when left arrow key pressed event programming on your own. Raise your hand if you are stuck.

Test!



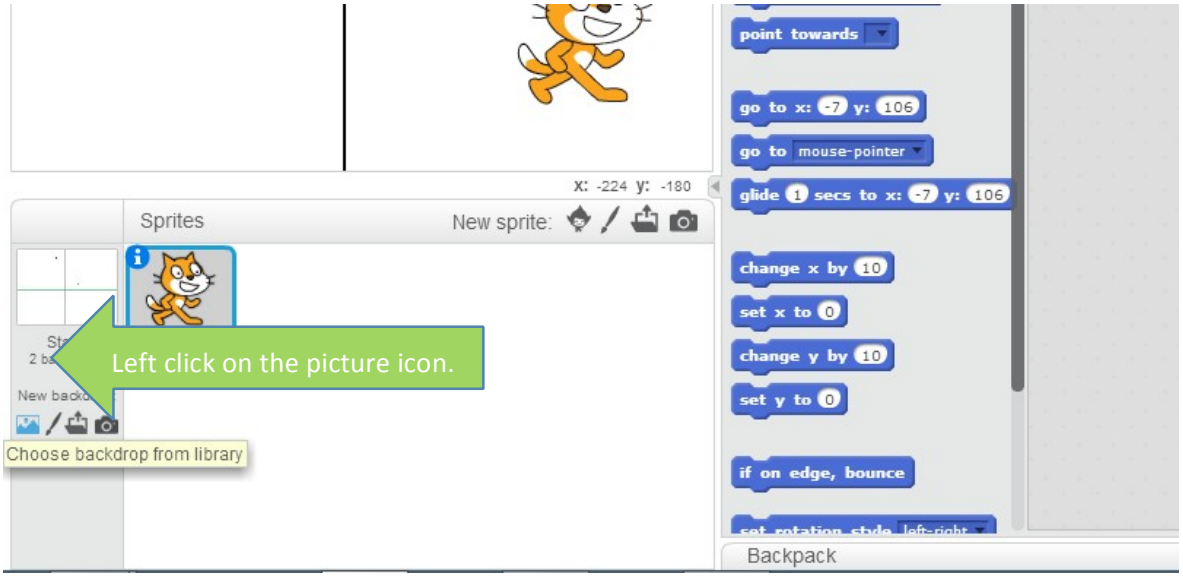
If you are done and waiting,

1) Help someone else debug. Helping them will help you learn too!
2) Right click in an open area of the scripts window. What does “clean up” do?
3) Click around and see the different types of blocks available: Motion, Looks, Sound, Pen, Data, Events, Control, Sensing, Operators, and More Blocks.

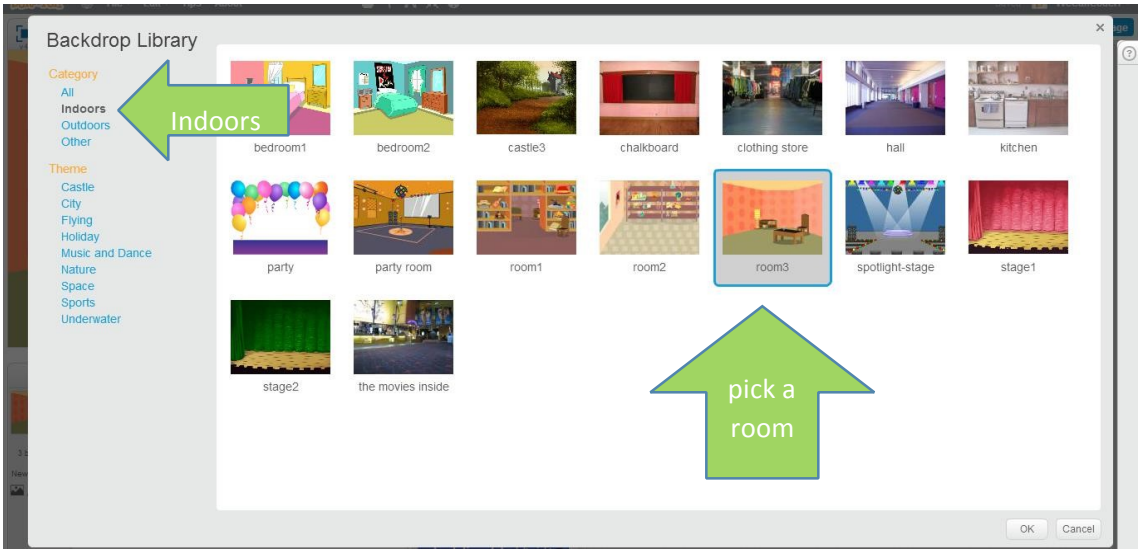
Step 11: Add a backdrop

Let's dress up our stage by adding a backdrop of a room to it. This is where your pet will live.

Left-click on the image icon at the bottom left of your screen.

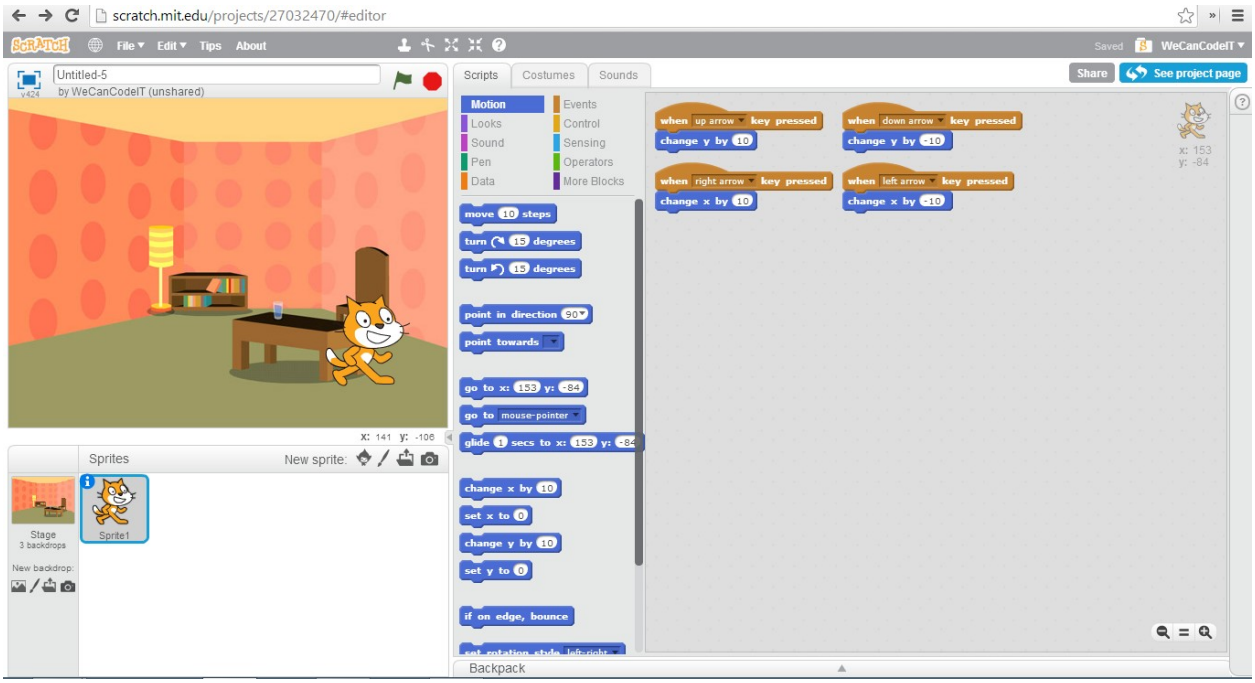


A new screen appears. **Click the "Indoors" category and double- click a picture of an inside of a room.**



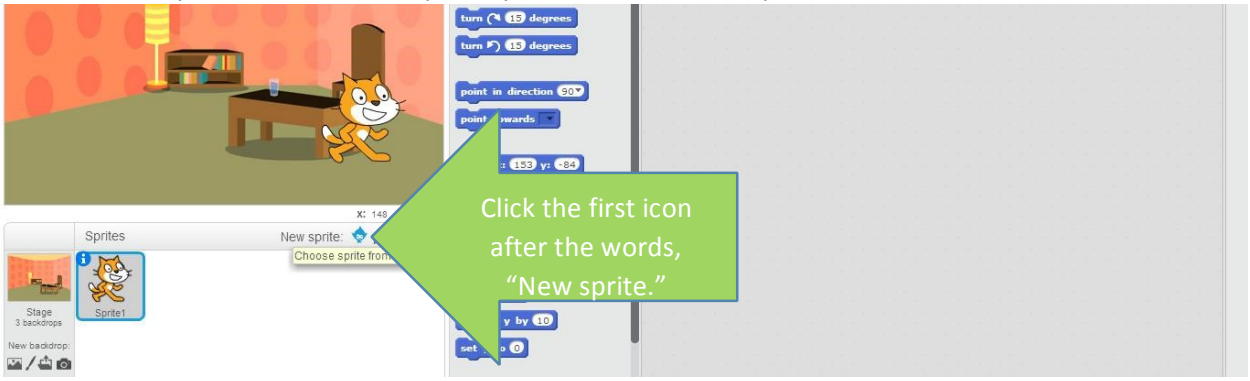
Step 11: Add a backdrop (continued)

The result? A new backdrop in your game!



Step 12: Feed your Pet, add the look

Let's add a new sprite (another actor or prop) to the scene. This will be food for your pet. Doing this is similar to how you added a backdrop, but you click on the Add Sprite icon.

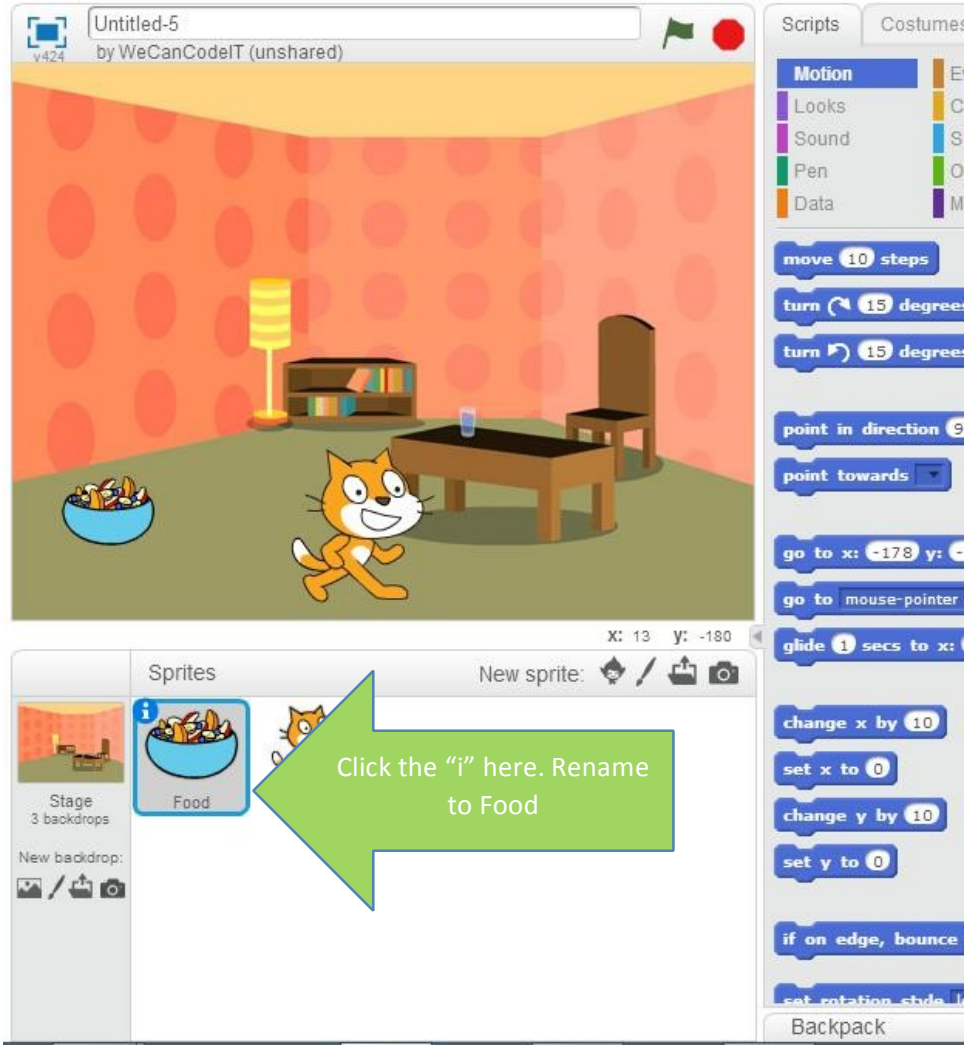


A new window will appear.

Step 12: Feed your Pet, add the look (continued)

You'll see your food item appear.

Click the "I" in the image of your food item, and rename your food to "Food"



Test.
Move around.
It's looking good, but nothing much is happening. Let's change that!

Step 13: Feed your Pet, make a variable to store the amount of food

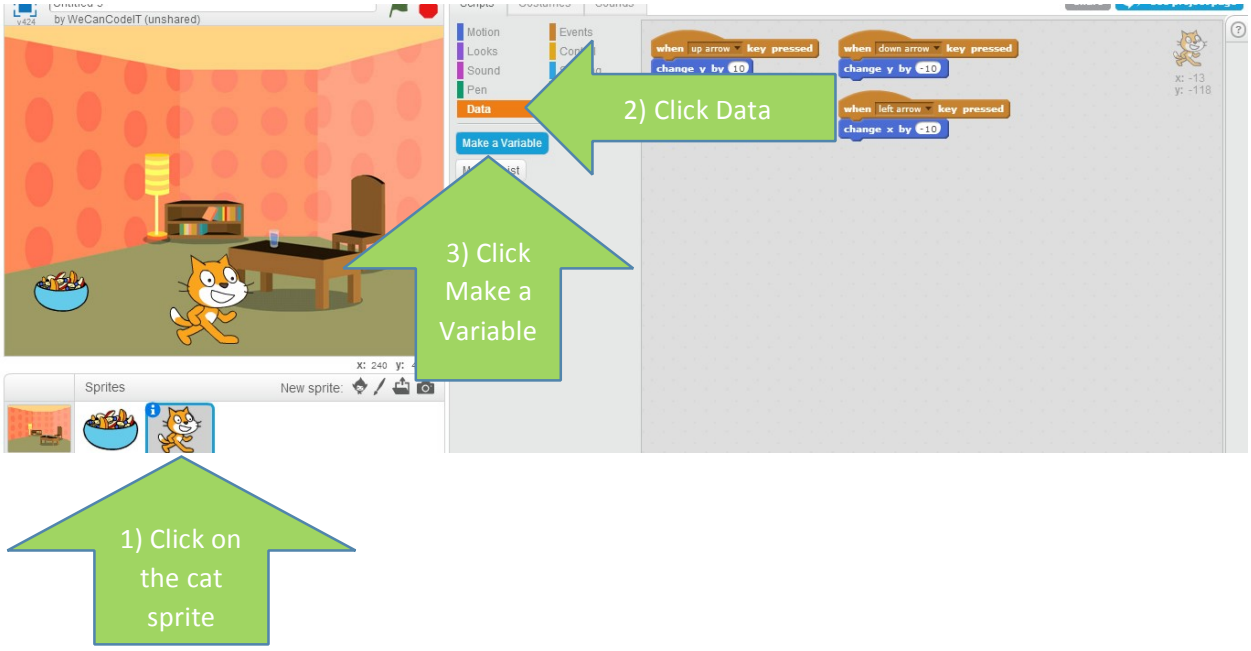
You have a stomach. You fill it up, and it empties. It's a place to store (and process) food. Well, we need to mimic storage of food in Scratch.

We store values in special blocks called "variables." Variables are a type of data. Their values can vary, hence the name "variable."

Click on the Cat Sprite

Click Data

Click Make a Variable

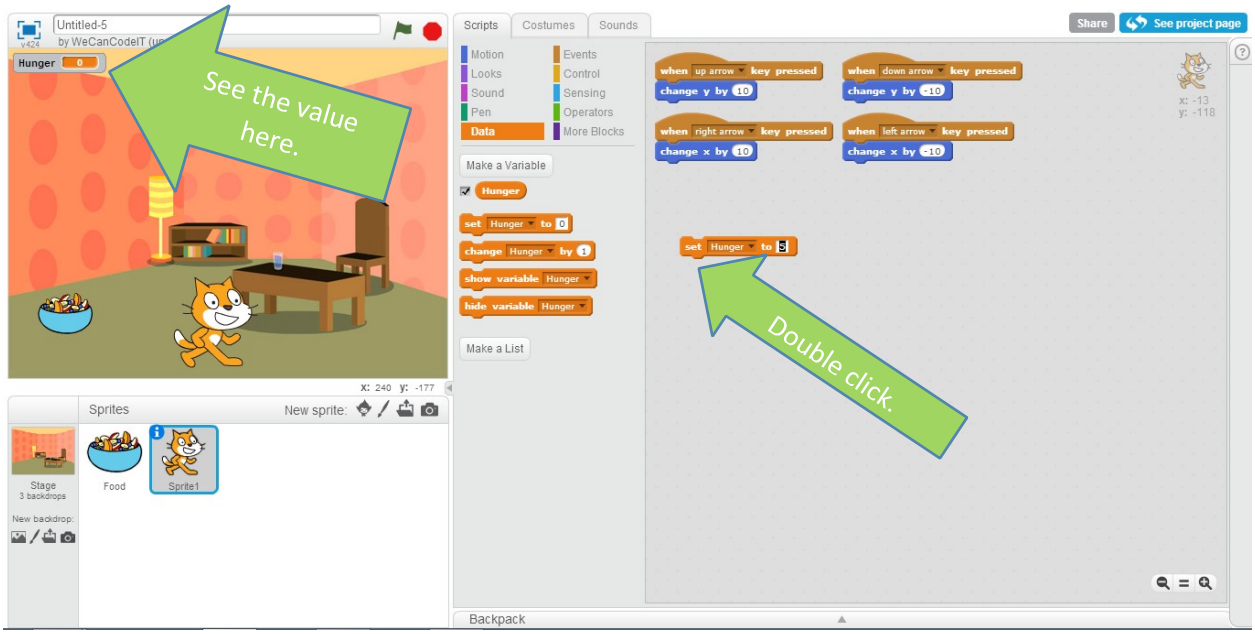


Name your variable Hunger, then click the Ok button.

Step 13: Feed your Pet, make a variable to store the amount of food (continued)

Drag the "set Hunger to 0" block in to the scripts window.

Change the value to 5 (we'll say that 0 is starving and 10 is full, so 5 is right in the middle) This is called "initializing" the variable.



Test: the "Set Hunger" block by double clicking on it. Did it change the Hunger variable to 5?

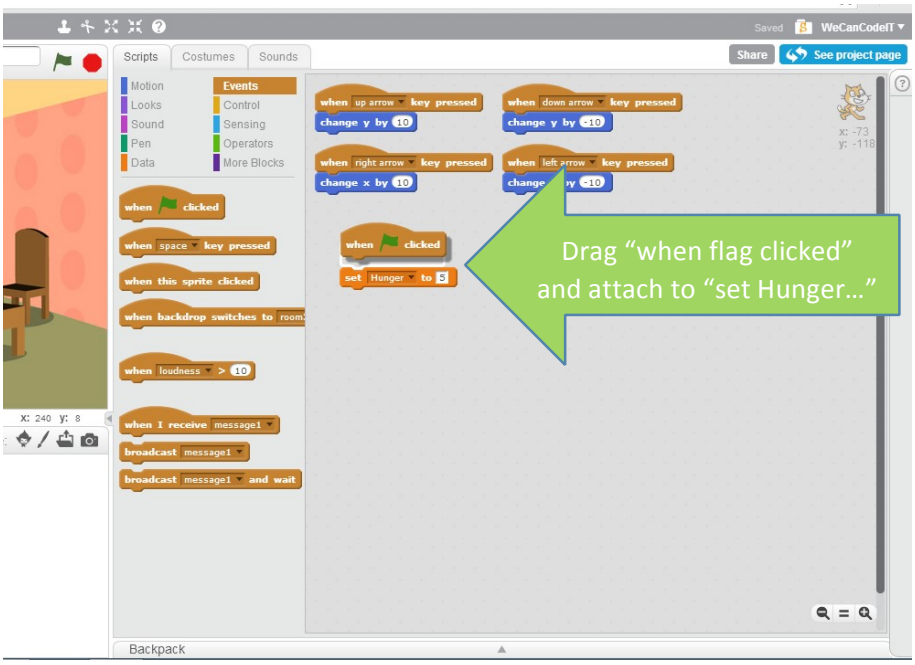
Step 14: Starting the program

So far, so good, but how do we start the program in order to set the Hunger variable to 5? A special event object comes in to play.

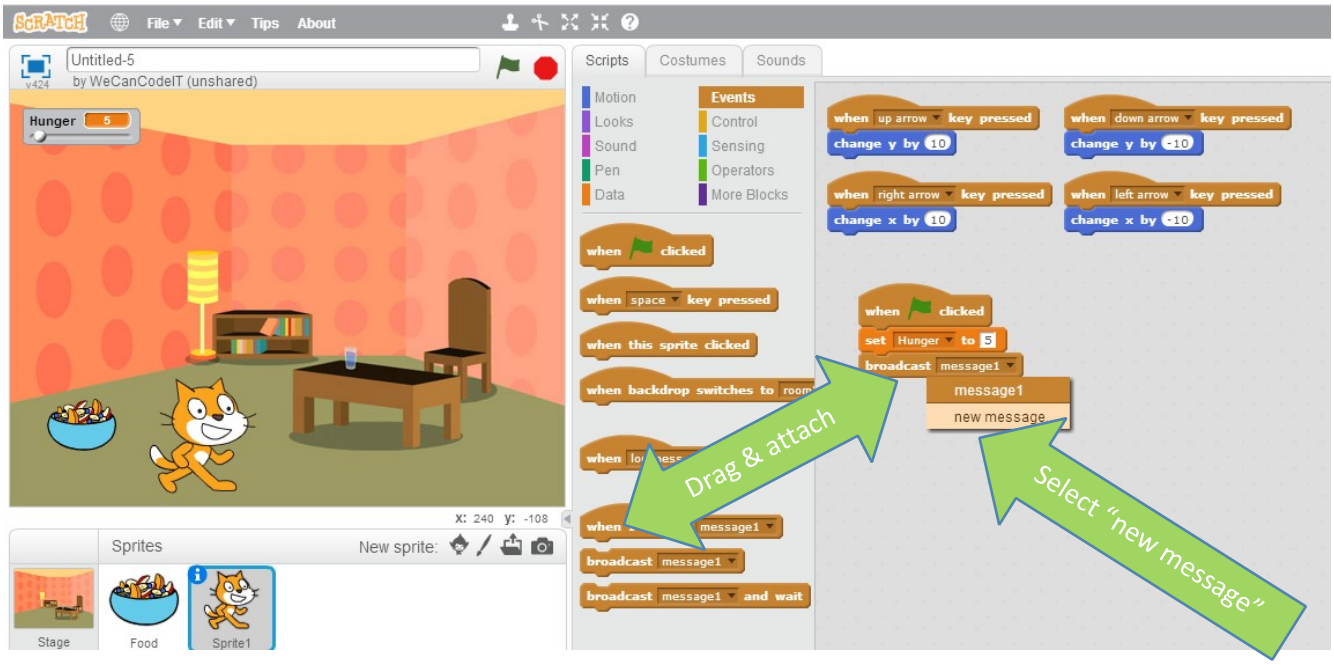
The green flag starts the program.

Click Events

Drag the "when flag clicked" event on to the script area, and attach "set Hunger to 5" on it.



Step 14: Broadcast a Message

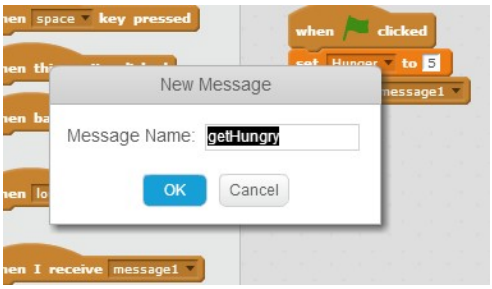


We're going to do something special now. We are going to "broadcast," or a shout out, to a new message. It's like phoning a friend. Let's broadcast a new message simply count down time, and subtract a number from Hunger every 10 seconds. Just like you get hungry when you don't eat over time, so will the cat.

Drag "broadcast [message1]" on to the "set Hunger . . . " block.

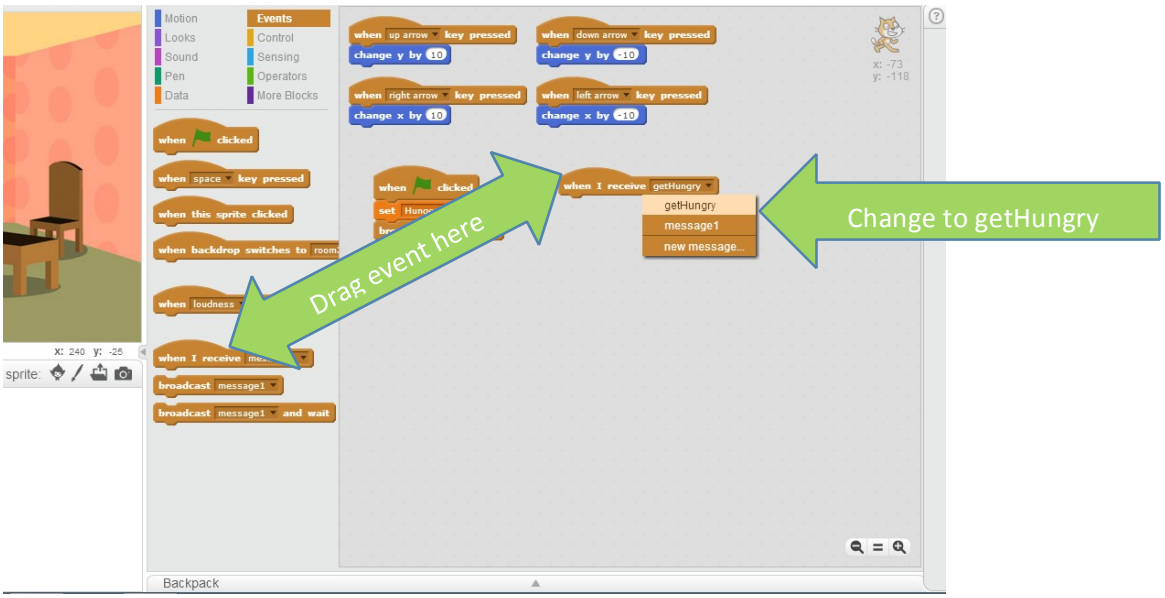
Click the dropdown by the word "message1" and left-click on "new message."

A new window will appear.



Name the message getHungry, and click the OK button.

Step 14: Add a Listener



Now, let's have an event pick up that message, like a friend who will always pick up a phone call.

Locate "when I receive [message1]" in the events area.

Drag that block to a blank area in the scripts window.

Change the dropdown to "getHungry" instead of "message1."

Pseudocode Time!

When feeling hungry, is your stomach less full or more full?
What variable should we change?
How often should we do this?
Do you get hungry once, or does it happen over time, and go on and on until you eat?

Share your thoughts with the class.

Step 15: Adding Code to the Listener

Did your pseudocode read something like this?

Find and attach the two control blocks, and 1 data block to the "when I receive getHungry" event listener.

Test! (Click the flag and wait 10 seconds. Did Hunger go down?) If

(needHelp = true) then

raiseHand();

---

*If you are waiting . . . find out what happens when you wait by 1 second. What happens if you change hunger by a different value? See what other items can **Control** the program flow?*

Step 16: Understand "Loops" or "Iteration"

In the real-world, many of the same things happen over and over again.

The sun rises every day. You eat several times a day. You sleep every night. You have a birthday every year.

When you do something over and over again, in a cycle, you might say it's a "loop." Another word for this is "iteration."

Programmers use loops/iteration to their advantage! Why write something many times when we can simply call the same commands over and over again inside the loop?

Go to Sleep Loop Pseudocode

Forever

Wait [24] Hours

Brush teeth

Wash face

Put on pajamas

Lie down in bed

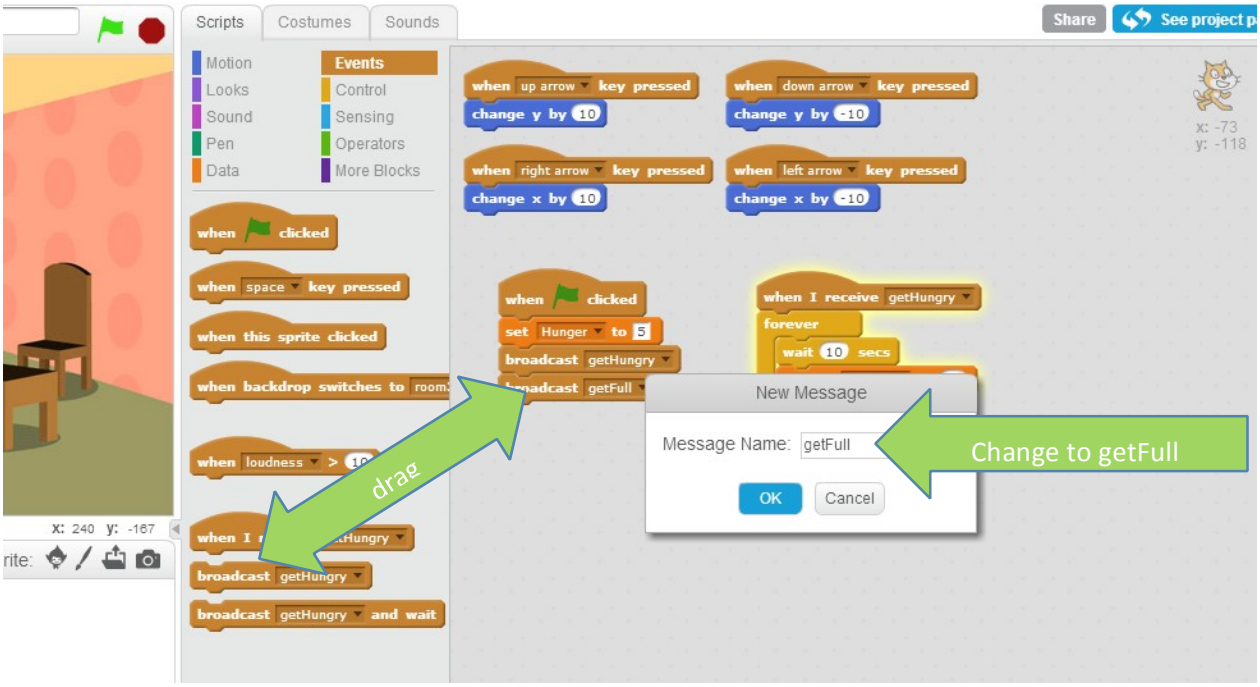
Close Eyes

Fall Asleep

Does this seem a little bit like our "forever" loop?

Step 17: Add a Feed (getFull) Event Broadcast

We need to add another event broadcast, like we did for getHunger, but this one will be called getFull. We will add code to simulate what happens when the cat eats.



In Events, drag "broadcast [getHungry]" and attach to the other "broadcast [getHungry]" block to call it once the program starts.

Click the dropdown of the newly added block, the last one in the flag group, and click "new message."

When the New Message window appears, add the message name getFull.

It's the same process as the getHungry block. There's nothing much new to test yet, so let's continue by adding an event listener for getFull.

Step 18: Add a Feed (getFull) Event Listener

Again, our listener hears your call, and responds by doing a set of things.

Let's flush this out together.

Pseudocode Time!

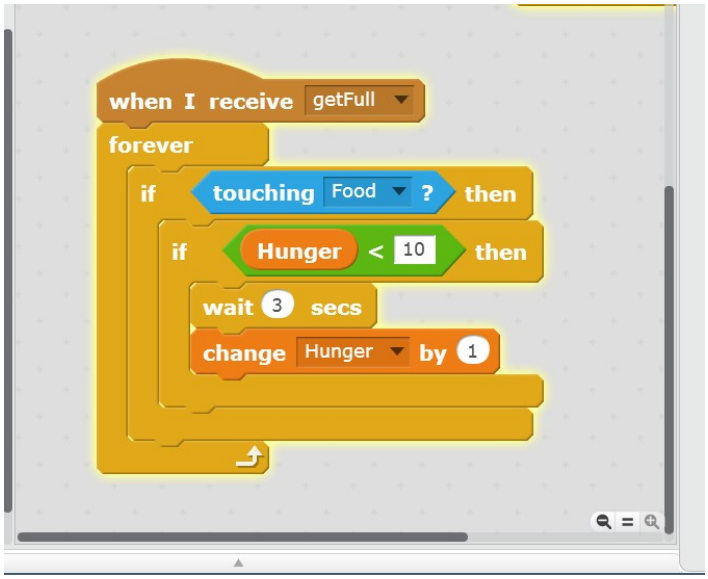
When should we check this?
How fast should the cat's stomach fill up?
Is there a maximum amount the cat can eat?
When you eat, is your stomach less or more full? What variable should we change?

Write your pseudocode down and share your thoughts with the class.

Pseudocode for event listener getFull

```
Forever
    If Cat is touching Food
        If Hunger < 10          //10 is our "full" state.
        Wait 3 seconds
        Add 1 to Hunger
```

Step 18: Add a Feed (getFull) Event Listener (continued)



Review the blocks above. Do they match our pseudocode?

Find the blocks above and add them to your script workspace.

I know this is more challenging than previous exercises, but **give it your best try**. Ask a friend. Help a friend. Feel free to raiseYourHand()!

Test by clicking the Green Flag to start. Wait 5 seconds to see if hunger still goes down. Move your cat to the food bowl. Wait 3 seconds. Does the Food variable go up? What happens when you reach a hunger value of 10?

When on food, after 3 seconds, if Hunger is less than 10 AND

If (Hunger = previousHungerValue + 1) then

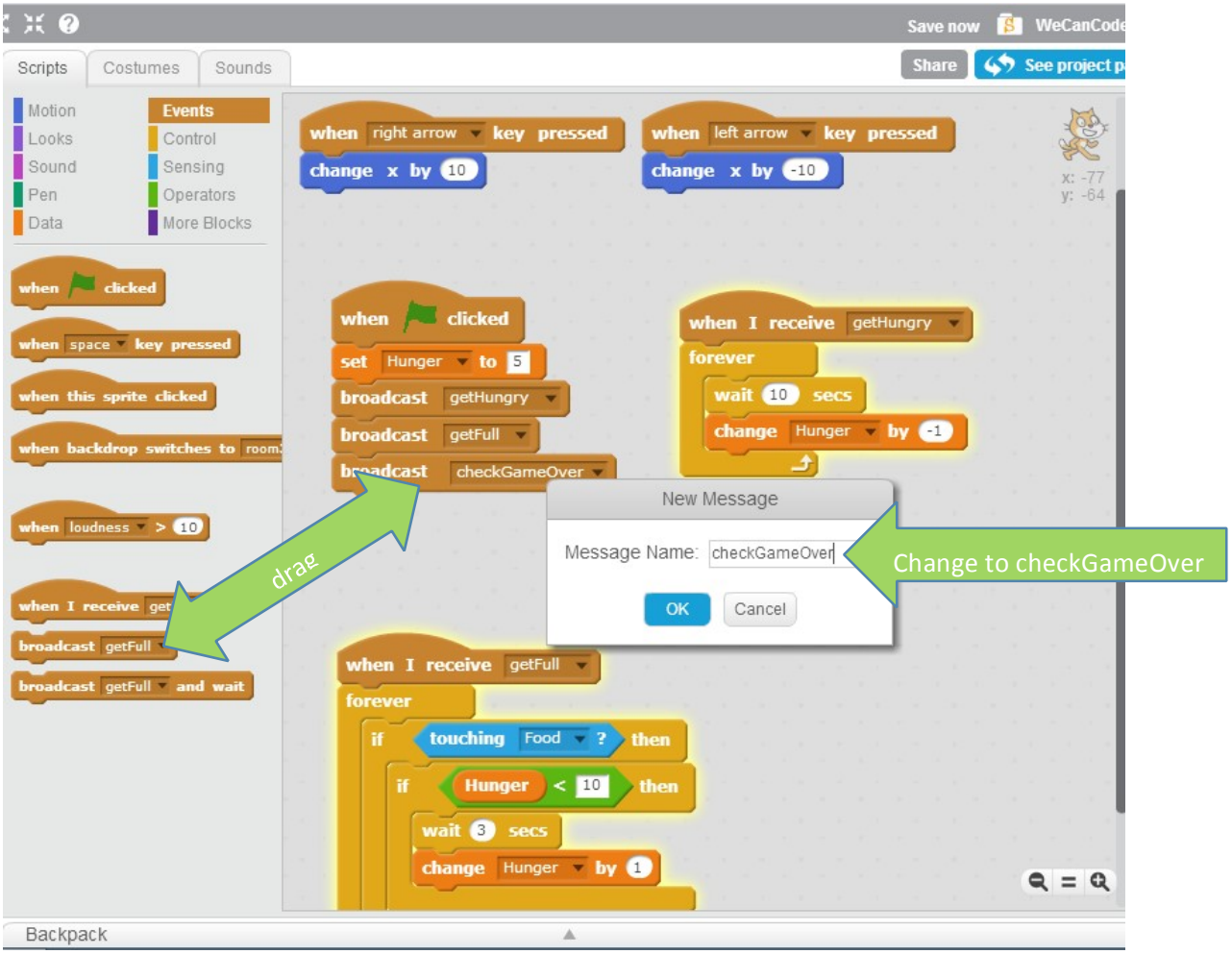
Celebrate!

Else

Debug.

Step 19: Add a checkGameOver Event Broadcast

We need to add at least one more event broadcast, like we did for getHunger and getFull, but this one will check if the game is over. The game is over if Hunger is below 0. Why? Because we're making it, and we call the shots! Call this event broadcast checkGameOver.



In Events, drag "broadcast [getFull]" block and attach it after the "broadcast [getFull]" block. This calls the new event broadcaster once the program starts.

Click the dropdown of the newly added block, the last one in the flag set, and click "new message."

When the New Message window appears, add the message name checkGameOver.

You can test to see if anything is broken, but we won't see anything new until we add an event listener.

Step 20: Add a Game Over (checkGameOver) Event Listener

Again, this listener hears your call to the checkGameOver Event Broadcast, and responds by doing a set of things.

Let's flush this out together.

Pseudocode Time!

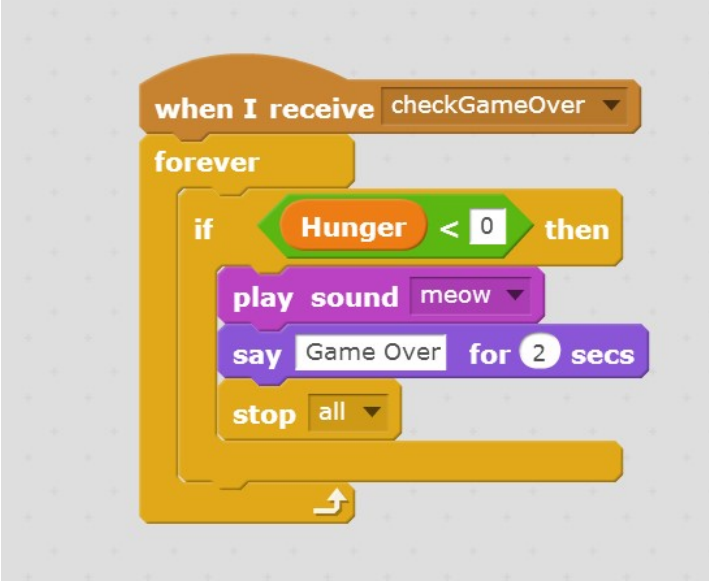
When should we check this?
What variable(s) determine if the game is over?
What should the variable value(s) be in order for the game to end? What should we do once the game ends?

Write your pseudocode down and share your thoughts with the class.

Our Pseudocode for event listener getFull

```
Forever
    If Hunger < 0                    //0 is our "game over" state.
        Have the cat Meow (sound) Have the
        cat Say "Game Over!"
        Stop the game.
```

Step 21: Add a Game Over (checkGameOver) Event Listener (continued)



Review the blocks above. Do they match our pseudocode?

Find the blocks above and add them to your script workspace.

I know this is challenging, but **give it your best try**. Ask a friend. Help a friend. Feel free to raiseYourHand()!

Test by clicking the Green Flag to start.

Wait until your hunger becomes less than 0. Does the cat meow (if you have sound on your computer)? Does the cat say "Game Over?"

Keep testing. Click the Flag again, and see if your full game works by moving your cat to the food bowl. Wait 3 seconds. Does the Food variable go up? What happens when you reach a hunger value of 10? Do the values still decrease when you have waited around 3 seconds and are away from the bowl?

If test is true then

    Congratulations!

Else

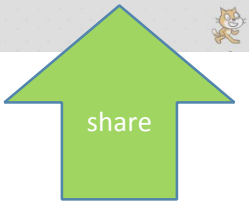
    Debug.

We Can {Code} IT

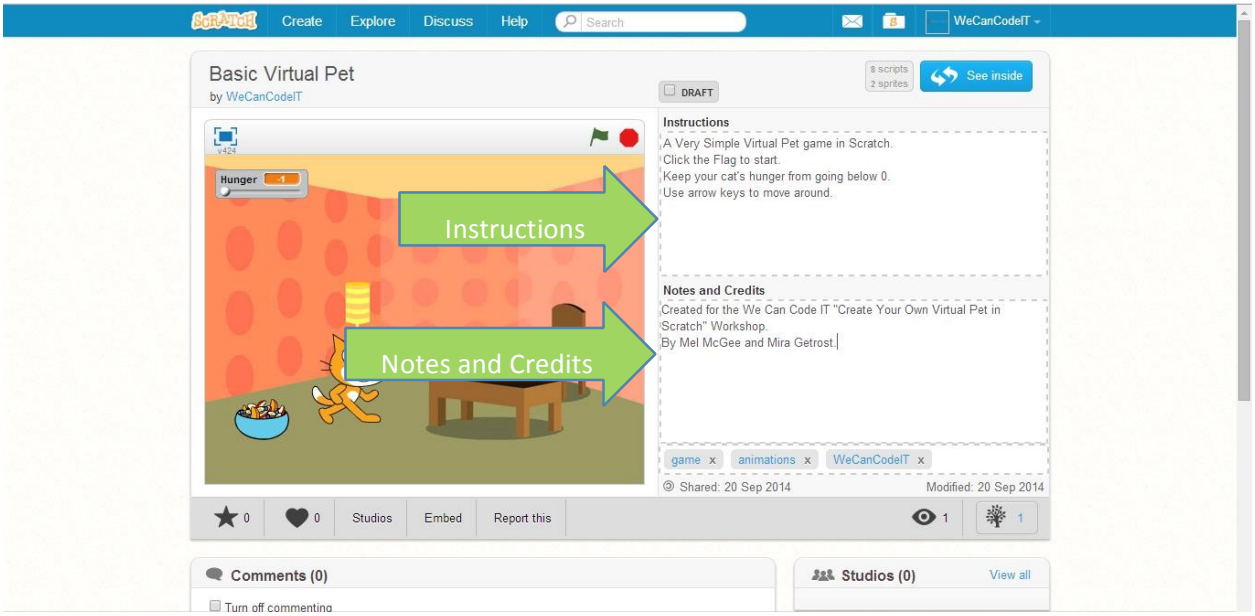
Share Your Project in Scratch

Save your project by going to File, then clicking Save Now.

Share your project with others by clicking the Share button.



The Project Page

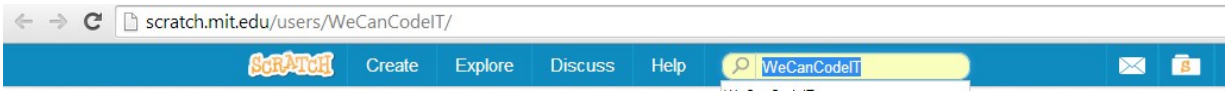


After you click "Share," you will end up on your project page. Fill out the instructions, as well as the Notes and Credits. If you are building off of someone else's project, it's important to give them credit. You would want the same, right?

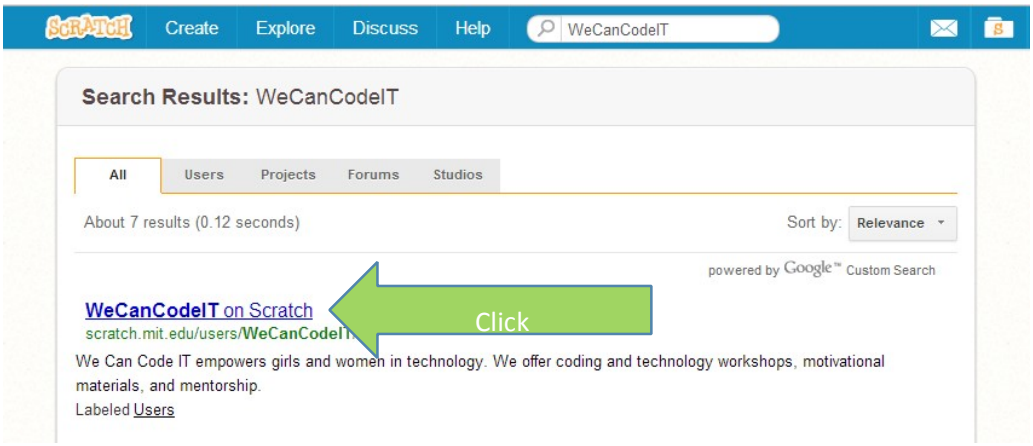
We Can {Code} IT

Follow We Can Code IT on Scratch!

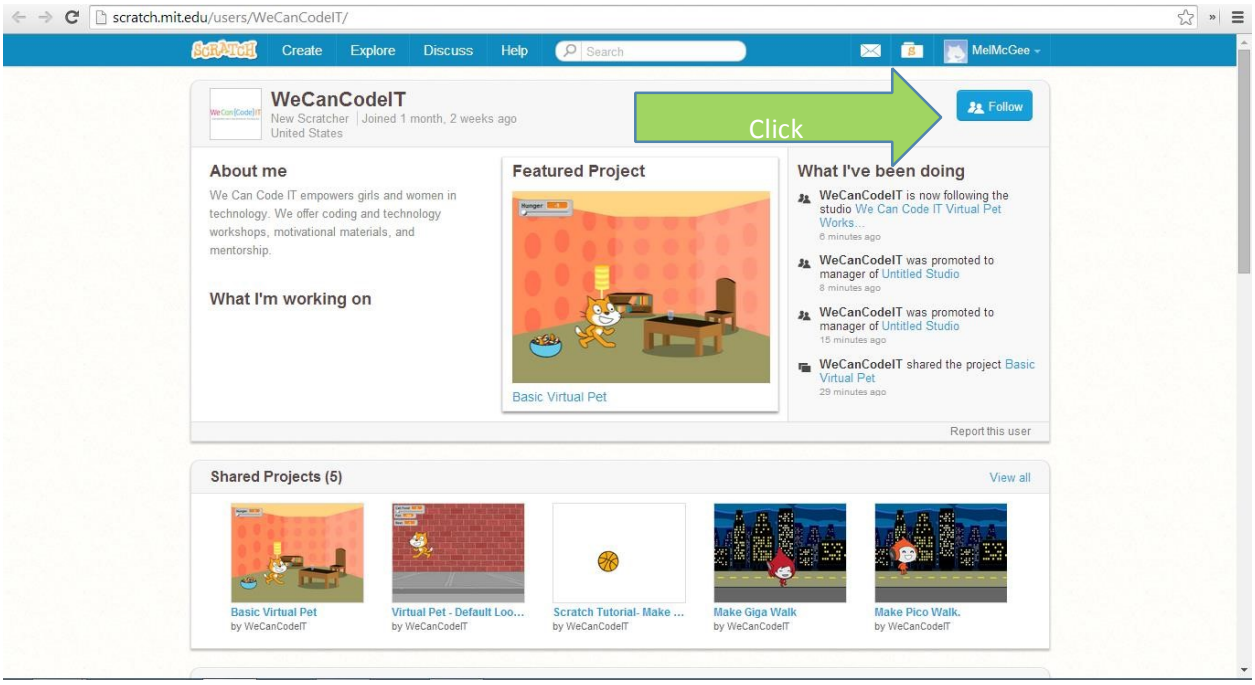
In the Search bar, type in WeCanCodeIT (no spaces), and click Enter on your keyboard.



Click the first link. That's us!

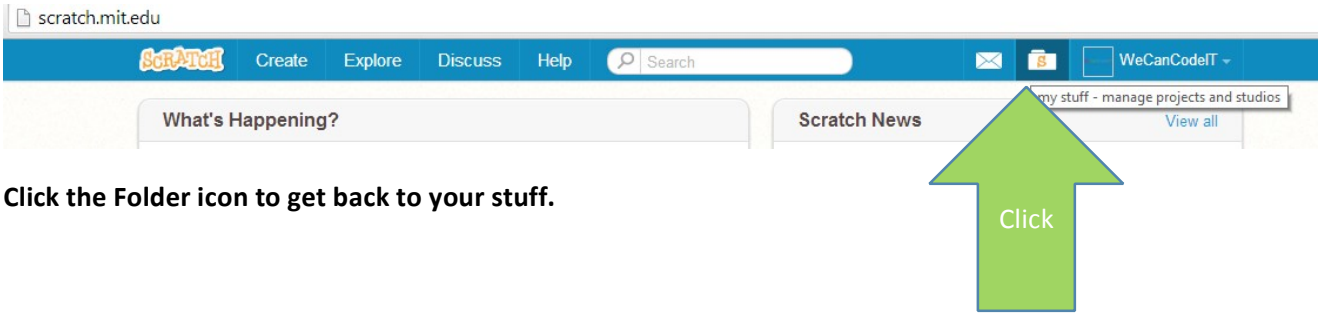


Click the Follow Button.

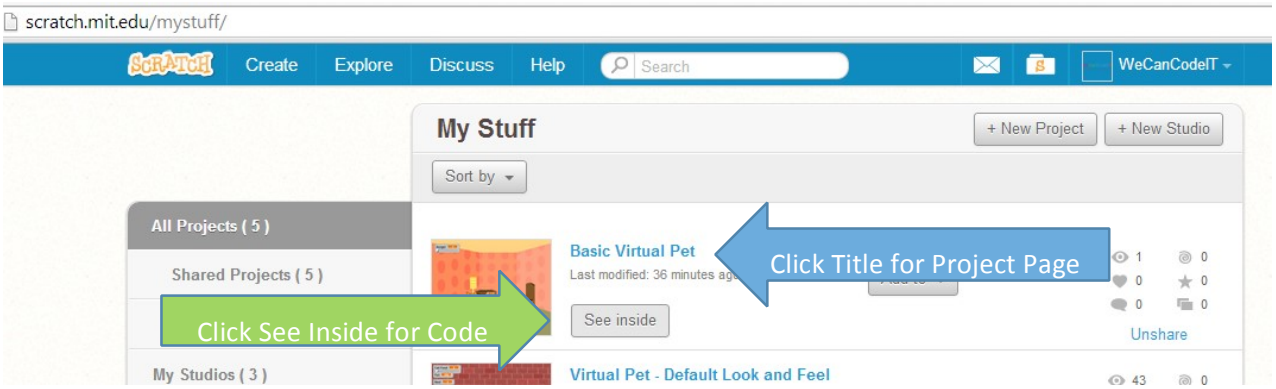


WeCan{Code}IT

Get Back to Your Stuff and Create Your Own Adventure!



Click the Folder icon to get back to your stuff.



Click "See Inside" to get back to your Scratch Code

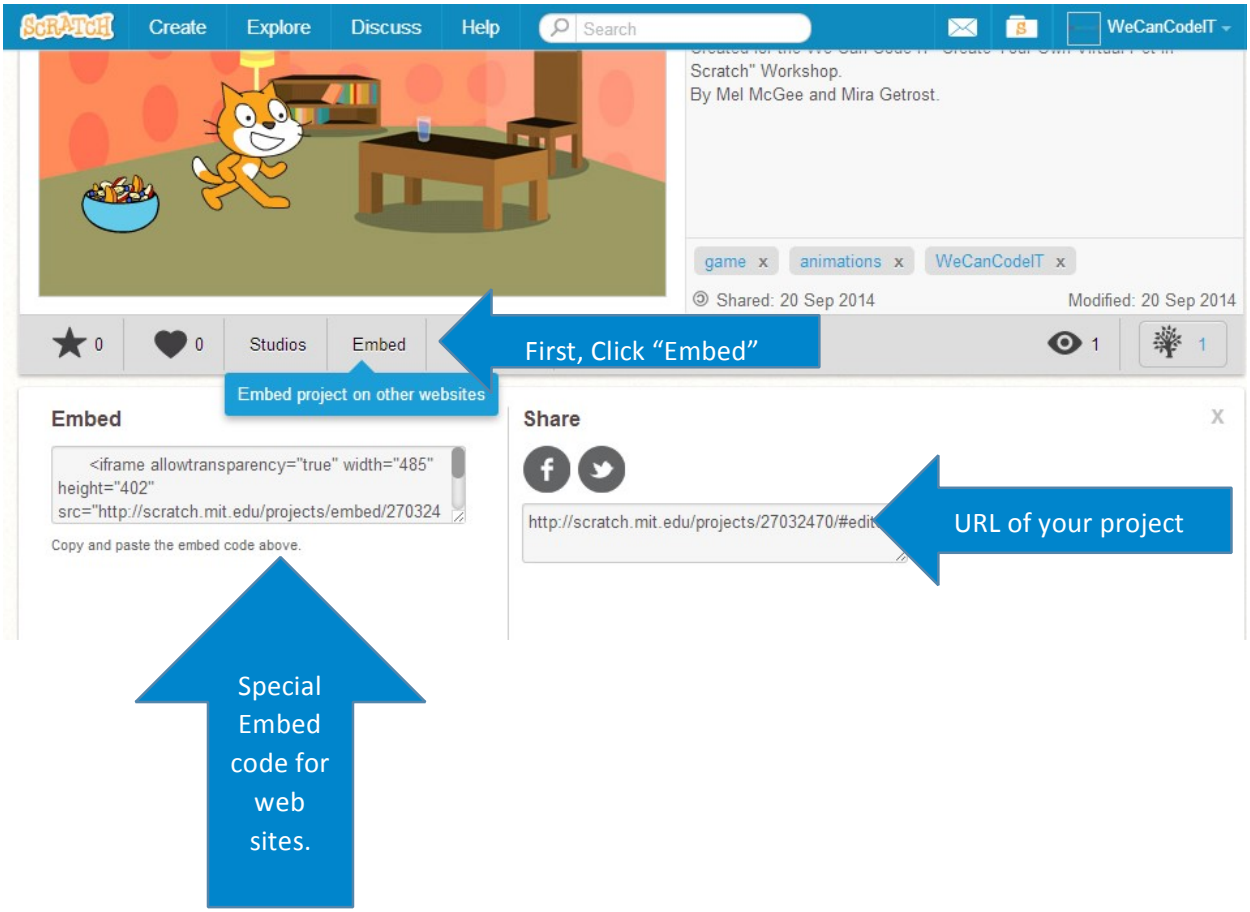
OR

Click the title of your project to see the Project Page.

We Can {Code} IT

Create Your Own Adventure: Project Page

You can Share your project outside of Scratch with your friends by clicking Embed, and giving them the URL of your project page. You can also add your project to Facebook, Twitter, or even code in a Web Page from the Embed area.



We Can {Code} IT

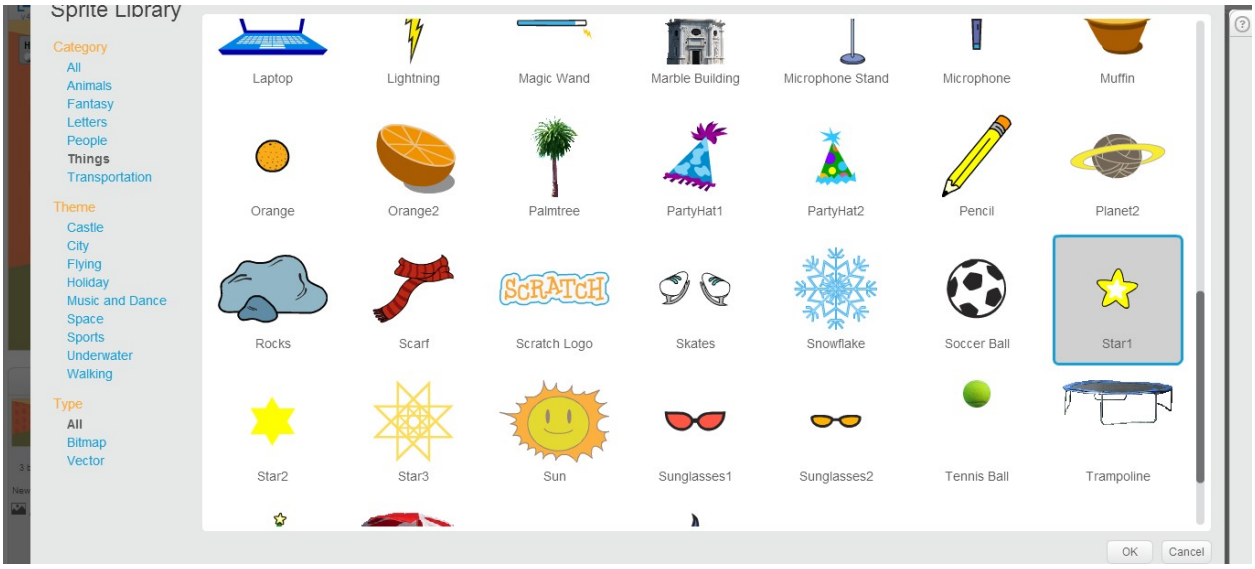
Create Your Own Adventure: Add Fun or Rest to your Virtual Pet

Try adding more code to your pet to handle Fun and / or Rest.

Let's try adding Fun into the equation!

Add a new sprite.

Click on the new sprite button. Choose a picture, I chose a star.



Place your star on the stage.

Click the cat again. (The last sprite on the stage that you click will go to the front, plus we will see the cat's code).

Add a new Data Variable called Fun

Initialize the Fun variable to 5 and place in the Flag block, after the Hunger's initialization.

Create a new broadcast message called checkFun, add it to the Flag block, after the checkFood broadcast event.

Duplicate the event listener code block for checkFood (right-click on the start of the block set, then select duplicate, then place the blocks), but change the name to checkFun, and decrease the Fun variable instead of the Hunger variable. Also consider changing the amount of time for the wait block.

Update the checkGameStatus event listener code block to add both the Hunger AND Fun variables and see if the result is less than 0 before ending the game.

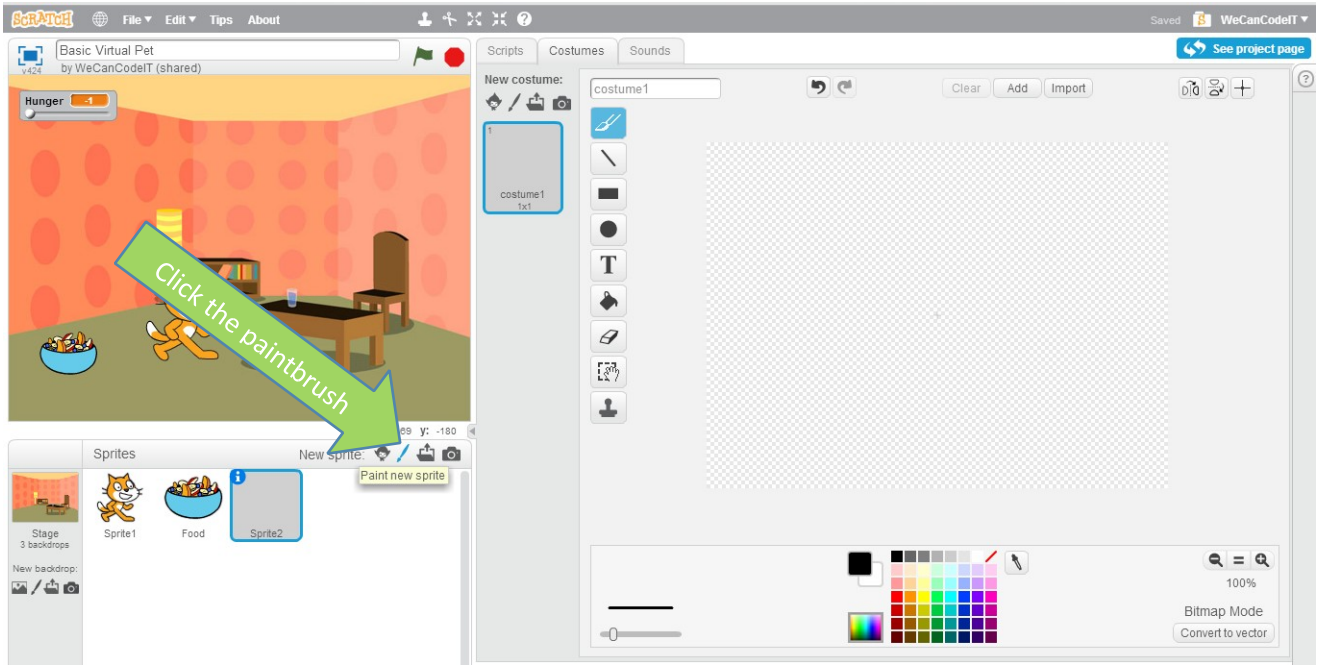
Follow the same process for Rest, or any other activity in which you want your virtual pet to engage.

WeCan{Code}IT

Create Your Own Adventure: Paint a New Sprite

Scratch allows you to select sprites from their library, but you may also paint your own sprites, upload your own pictures to use as a sprite, and more!

Let's start by painting a new sprite.



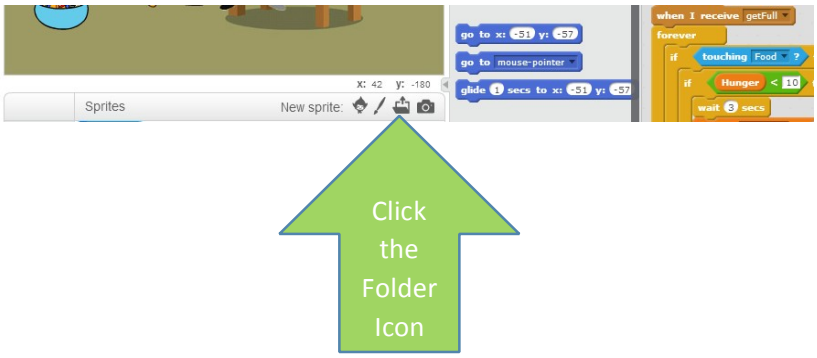
Click the paintbrush icon above the spite window region.

The rightmost window changes to an artist's canvas. Explore by dragging your mouse in this area. Click different colors, Different shapes. See what they do!

You can use these sprites as characters and props, in the same way we used the cat and the food!

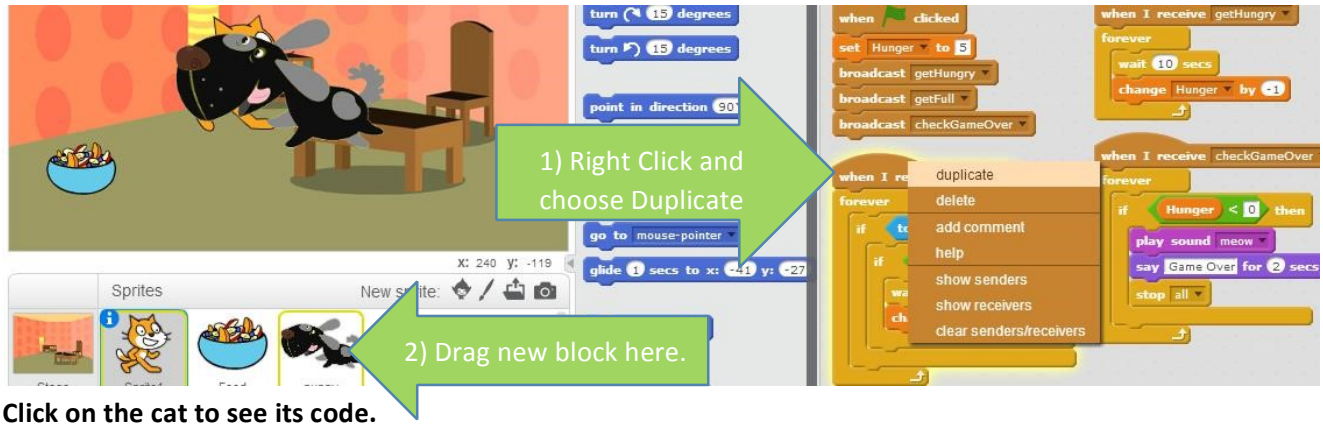
We Can {Code} IT

Create Your Own Adventure: Upload an Image File to Use as a New Sprite



Click the New sprite: folder icon.

Locate the puppy.png file in My Documents and double-click on it to start the upload.



Click on the cat to see its code.
For every block set, right-click and duplicate, then drag onto the dog thumbnail in the Sprites window. This will copy the code from one sprite to the other.

Test.

Notice all values may go up and down by 2x since the cat and dog are moving together. This will happen if they are on top of each other.
Save as a different file name to be safe, then delete the cat from the Sprites window.

We Can {Code} IT

Keep Coding!

Links to We Can Code IT

Please follow us and spread the word by sharing our posts!

Web: http://WeCanCodeIT.org

Facebook: http://facebook.com/WeCanCodeIT

Twitter: http://twitter.com/WeCanCodeIT

Follow us on Scratch: http://mit.scratch.edu/users/WeCanCodeIT

Email us at hello@WeCanCodeIT.org

We have monthly workshops. Sign up for our newsletter online to get all the info!