GSOC STEM – “Scratch” Programming Workshop – The Tile Game

Overview:
This two-hour workshop will introduce girls to basic programming and online game design fundamentals. The workshop can be presented in two one hour sessions or a single two hour session. This guide is designed for leaders or mentors that would like to lead a workshop or to get a better idea of the workshop’s objectives in order to acquire a workshop leader.

The objective of this workshop is to create a virtual sliding tile puzzle game using the Scratch programming language.

Requirements:
Each girl needs a laptop computer with a web browser supporting adobe flash (Firefox web browser is preferred) and an internet connection. *** IMPORTANT * each girl needs to know their parent or guardian’s email address.**

Scratch is a “cloud” application meaning that each girl will create a Scratch account that they can then use after this workshop from home.

Alternatively, Scratch can be installed as a desktop application eliminating the need for an internet connection, however, girls will not be able to access their Scratch work after the workshop.

The leader will need the same laptop configuration as the girls with the additional ability to project their screen or otherwise conveniently share their screen with the girls. Ideally the leader should obtain a physical example of the sliding tile game and a have a whiteboard to facilitate “brainstorming” game design which we will discuss within these instructions.

*** OPTIONAL FUN STEP *** If the leader has basic familiarity with the Python programming language, girls can optionally use their own photo for the tile puzzle face. Alternatively, they can use the ready made clock face image tile images found within this GitHub repository’s image folder.

GSOC STEM – "Scratch" Programming Workshop – The Tile Game

<u>OPTIONAL FUN STEP:</u> How to use your own picture as the face of the tile puzzle (20 minutes)

1) Use your phone (tested with iPhone) to take a picture of your face. Try to fill the frame as completely as possible as per the example image on this page. *** IMPORTANT *** take the picture in landscape mode which has a 4:3 aspect ratio.

2) A python script "SplitImageIntoTiles.py" is included within this workshop's GitHub repository . This script will appropriately resize the photo image for Scratch and split it into 9 tiles. A "blank" or empty tile (TILE-BLANK.jpg) of the appropriate size is included within the images folder of the GitHub repository. Note the instructions with GitHub repository's readme file regarding the installation of the Python pillow module which the SplitImageIntoTiles.py script requires.

3) Edit the python script and read the comments at the top of the script. Replace the reference of variable "infile" within the script with the location of the picture file from your phone.

4) Run the script and verify that 9 tile files (e.g. TILE-1.jpg – TILE-9.jpg) were created. Record the folder containing these files (or alternatively move them to a convenient folder) as you will be browsing these files into the Scratch project.

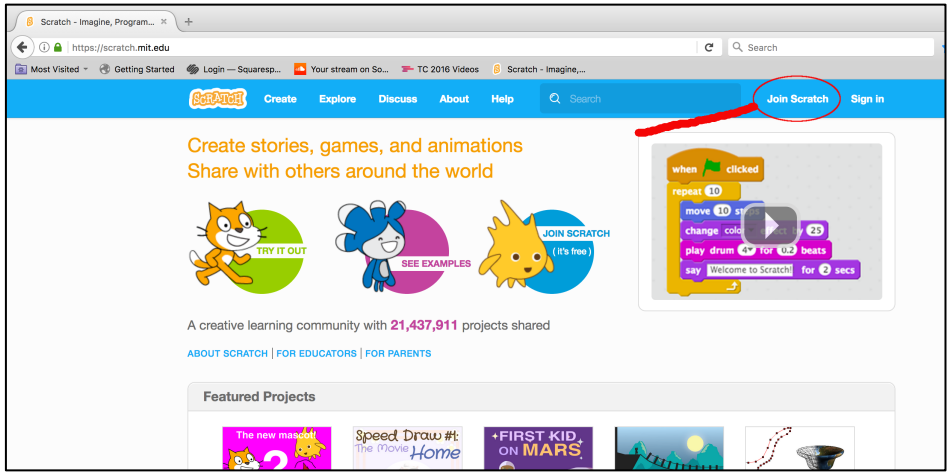
GSOC STEM – “Scratch” Programming Workshop – The Tile Game

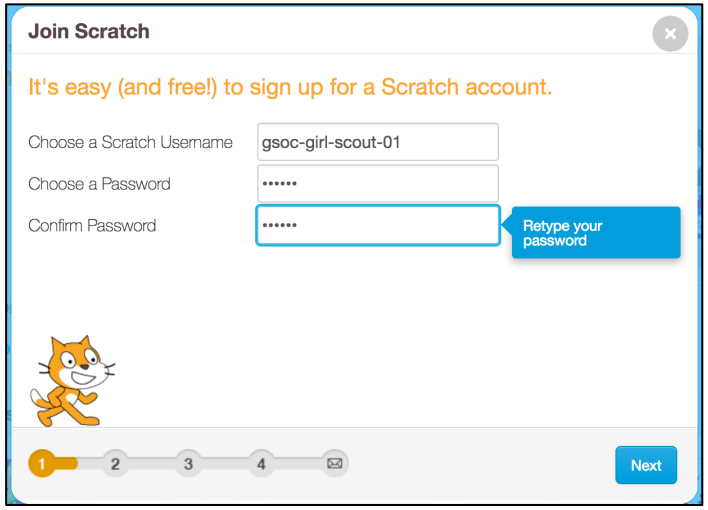
Using Scratch – CREATING A SCRATCH ACCOUNT (10 minutes)

Click here for a Scratch introductory video. Each girl will need to create a Scratch account as per the steps below.

1) **Navigate your browser to https://scratch.mit.edu**

2) Click on the Join Scratch link as show on the image to the right.



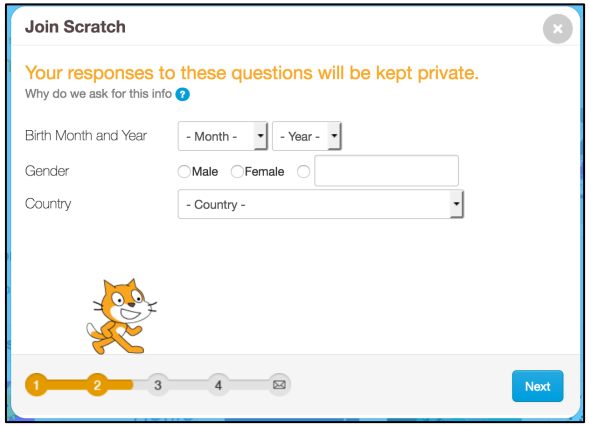


3) Follow the provided instructions, starting with selection of a Scratch username and password. As per the instructions, don’t use your name as your Scratch username. Don’t share user accounts as your work will be overwritten.

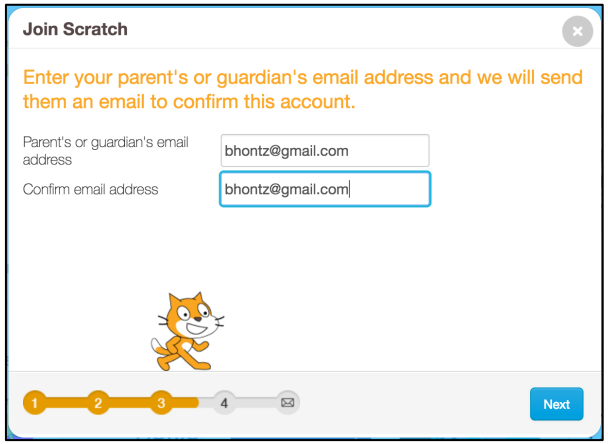
GSOC STEM – “Scratch” Programming Workshop – The Tile Game

CREATING A SCRATCH ACCOUNT (continued)

4) Enter the required age, gender and country information.



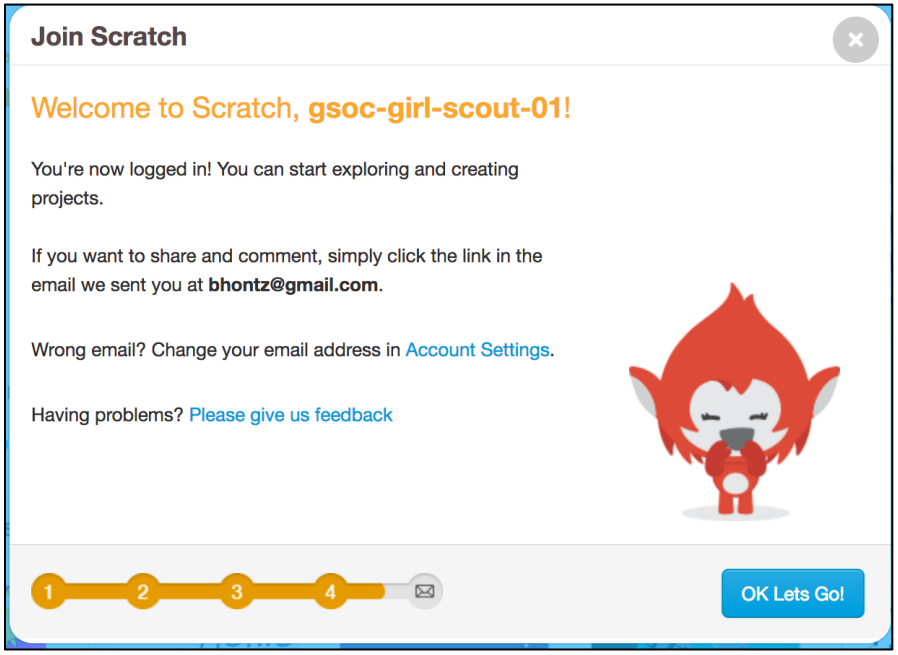
5) Each girl will need to enter their parent’s email address at this point. *** IMPORTANT * this step is required only if the girls wish to SHARE THEIR WORK**, if girls do not have a parent’s email address to use, then can use an email address that the leader provides and continue with this workshop without the leader’s email confirmation. **Confirmation of the email generated by this step simply enables the girls to share their work with others, and does not preclude the girls from using Scratch to complete this workshop.**



GSOC STEM – “Scratch” Programming Workshop – The Tile Game

CREATING A SCRATCH ACCOUNT (continued)

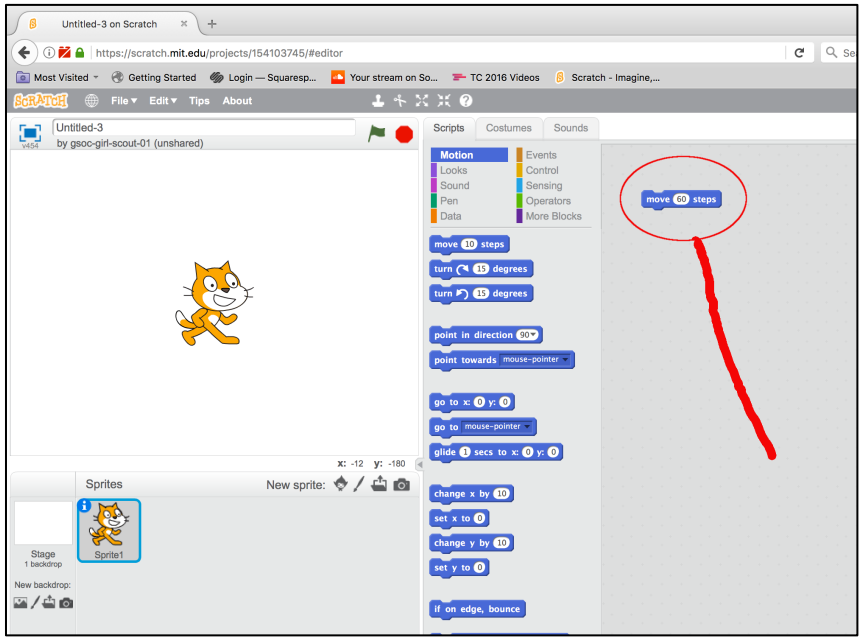
6) You’ve successfully created a Scratch account when the confirmation dialog appears as shown in the image to the right.



GSOC STEM – “Scratch” Programming Workshop – The Tile Game

Getting Started with Scratch (20 minutes)

What’s a “Sprite”?. The cat shown in the upper left corner of the image below is a Sprite. Sprite’s move about on the Stage, the white box that the cat Sprite sits on. The Stage is 240 pixels wide and 180 pixels high. A pixel is the smallest sized Sprite you can create on the Stage.



Each Sprite on the stage can has a script which controls the behavior of the Sprite’s movements on the Stage and how the Sprite interacts with the user and other Sprites. This interaction is the basis for creating an unlimited number of games!

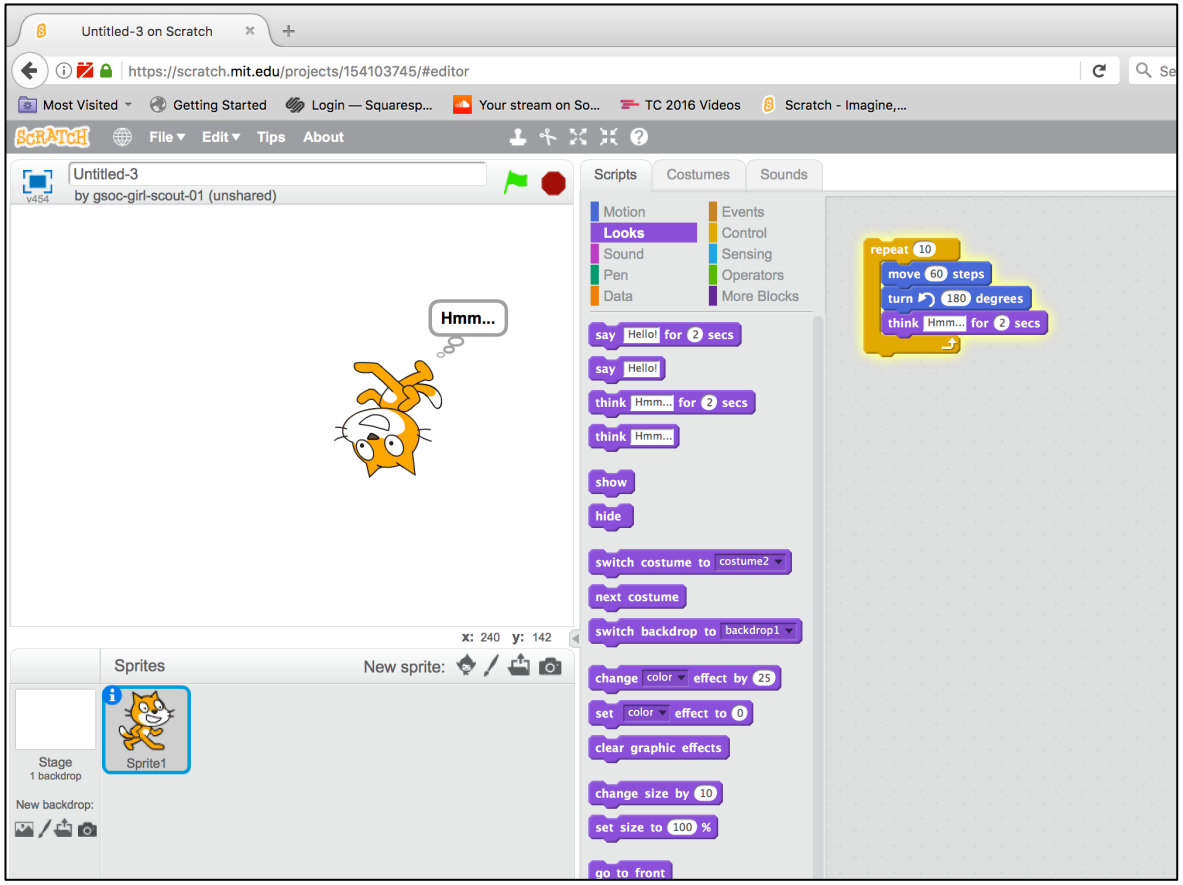
To get started, let’s move the cat forward one-quarter of the way across the screen, or 60 pixels (240 divided by 4). Drag the “move steps” block from the Motion block category to the Scripting area and change the optional value from 10 to 60 (as show to the left).

Now click on this move steps block and watch the cat move forward!

GSOC STEM – “Scratch” Programming Workshop – The Tile Game

Getting Started with Scratch (continued)

Let’s expand on our cat Sprite’s scratch as shown below. From the Motion script block category, drag the “turn to the left by degrees” block beneath our “move steps” block, changing the value to 180 degrees. Next, click on the Looks script block category and select the “think for secs” block, dragging



It beneath our newly added “turn to the left by degrees” block. Finally, let’s wrap these three instructions within a loop or “repeat block”, allowing the instructions to repeat.

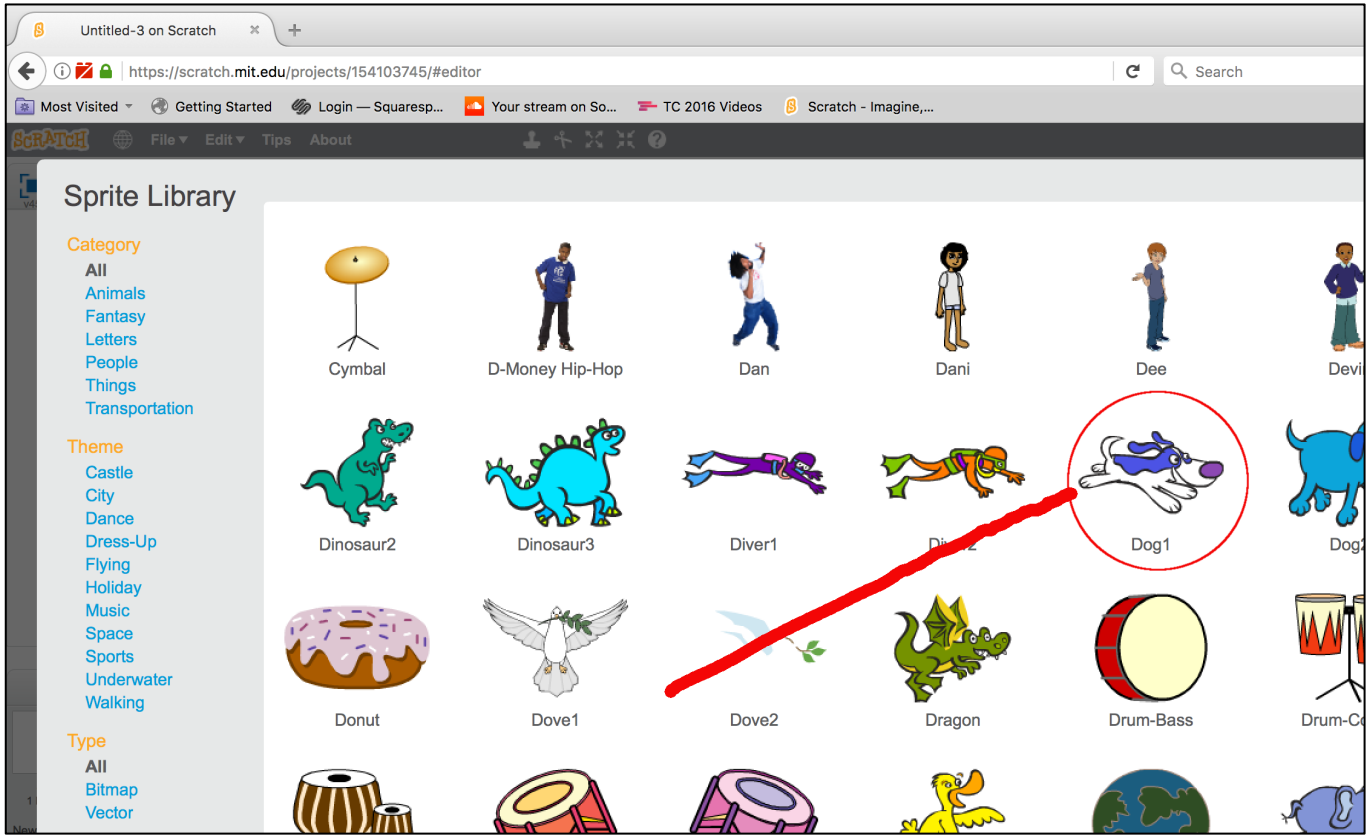
Click on the Control script category and select the repeat block, dragging it as shown so that our three previously added blocks rest inside of the repeat block.

Click on the top part of repeat block to see our cat Sprite spin and move about the Stage. Note that this behavior repeats 10 times, consistent with our repeat block’s setting.

GSOC STEM – “Scratch” Programming Workshop – The Tile Game

Getting Started with Scratch (continued)

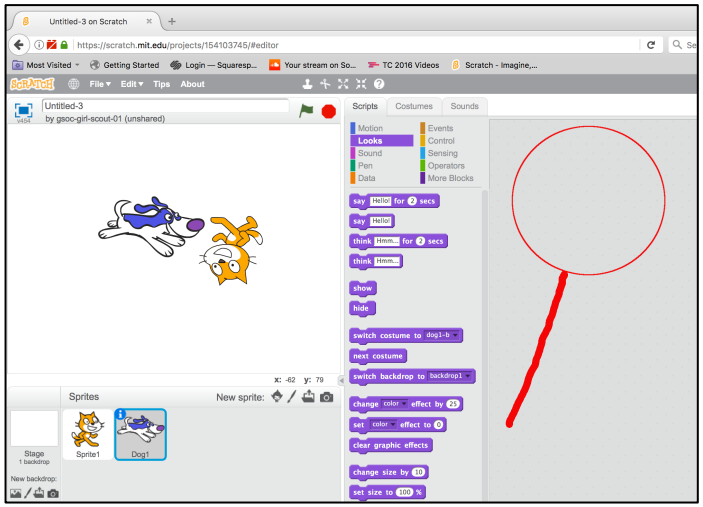
Let’s add another Sprite to learn how Sprites can interact with one another. Click on the first icon (Choose Sprite from Library) to the right of the “New sprite” reference found immediately beneath the Stage. Select the dog sprite from the Sprite Library as show in the screenshot below.



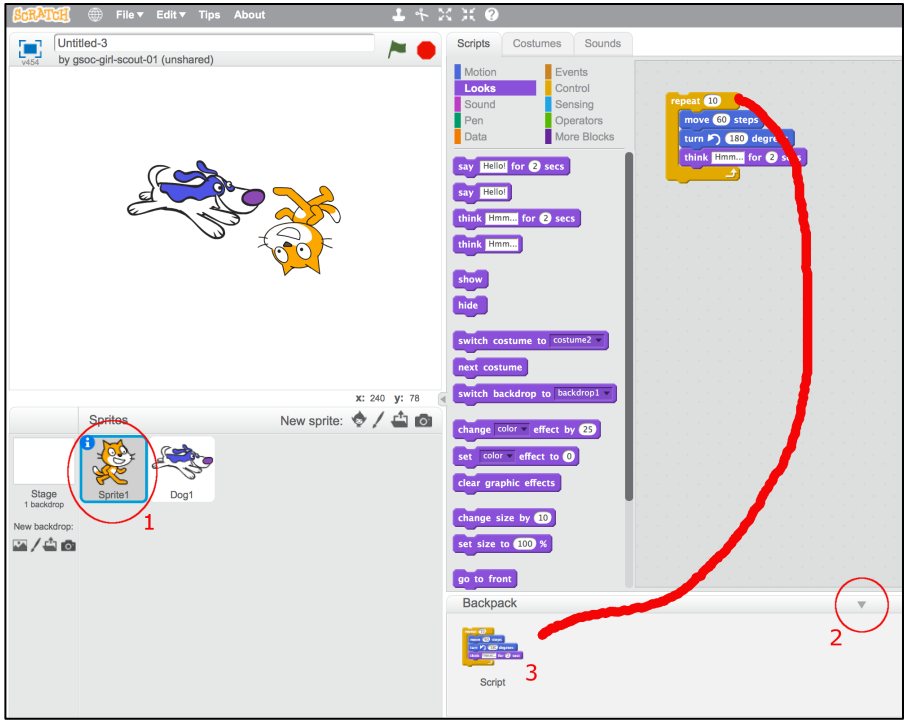
GSOC STEM – “Scratch” Programming Workshop – The Tile Game

Getting Started with Scratch (continued)

Our dog Sprite is now added to the Stage and the sprite’s icon is selected beneath the Stage. Note (as show to the right) that the dog Sprite does not have any scripting.



We can easily copy our cat Sprite’s scripting to the dog Sprite by using Scratch’s scripting “Backpack” **(1)**. The Backpack is a clickable window found at the bottom of the scripting window.



Follow the numbered steps on the image to the left. Step 1 - click on the cat Sprite to select this Sprite and expose its script. Next click on the Scratch “Backpack”, expanding this window, as shown in step 2. Drag all of the blocks of the cat Sprite’s script into the backpack as shown in step 3.

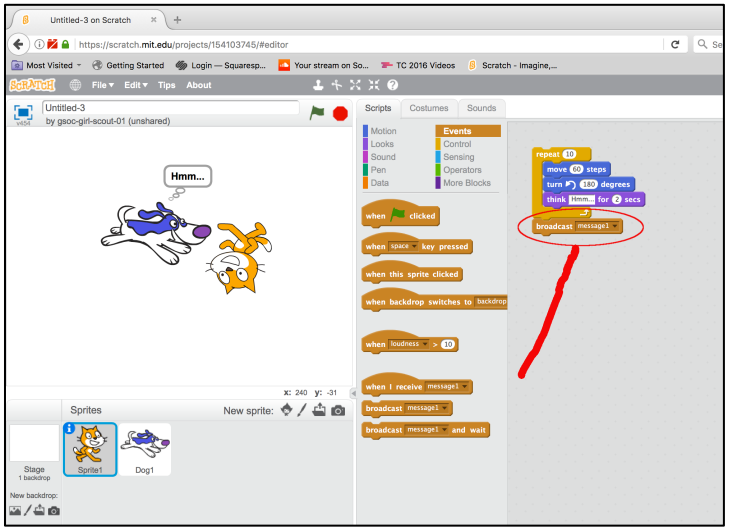
Now click on the dog Sprite’s icon (next to the #1 number cat icon in the image to the left) and drag the script within the backpack to the dog Sprite’s script area.

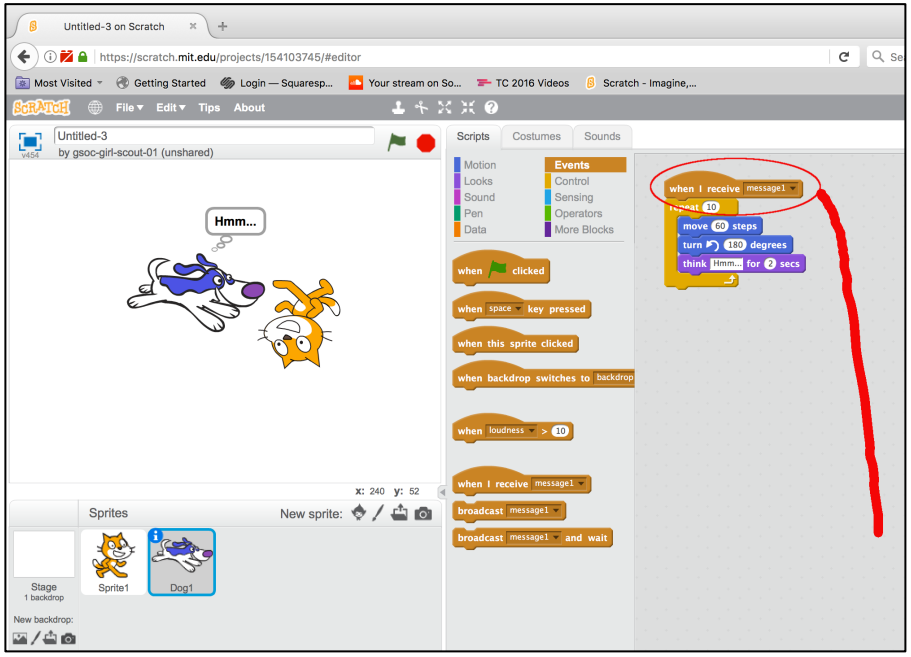
Click on the dog’s repeat script block and verify that the dog Sprite now moves in the same way as our cat!

GSOC STEM – “Scratch” Programming Workshop – The Tile Game

Getting Started with Scratch (continued)

Now let’s have our cat Sprite “talk” to our dog Sprite. First click on our cat Sprite beneath the Stage to select it. Next, from the Events script category, drag the events category “broadcast message” block to the cat’s script beneath the repeat script block as shown to the right. When we run the cat’s script, we’ll additionally send “message1” to any other Sprite that is listening for this particular message.





Click on the dog Sprite beneath the Stage to select it as show in the image to the left. Next, from the Events script category, drag the “when I receive message” block to the top of the dog Sprite’s scripts as shown to the left.

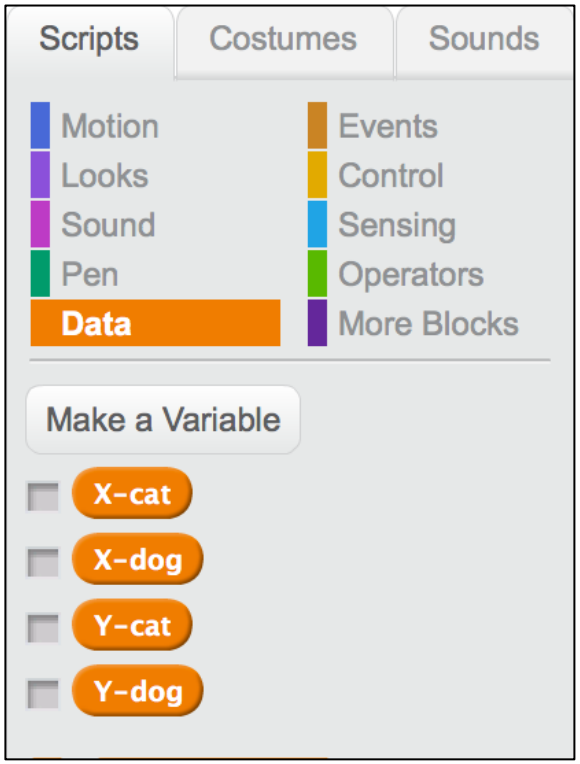
Now the cat and the dog Sprite’s are talking to each other! When the cat Sprite’s script finishes, the cat sends “message1” which the dog Sprite is always listening for. When the dog Sprite “hears” this message, the dog Sprite starts its own script.

This is called “event based programming” and it is the fundamental basis of programming user interfaces like MS-Windows as well as games.

GSOC STEM – “Scratch” Programming Workshop – The Tile Game

Getting Started with Scratch (continued)

Now that we understand messaging between Sprites, we need to learn one last fundamental skill before tackling our tile game; switching the positions of Sprites. To perform this task, we'll need to: a) understand the initial position of our Sprites in a manner both Sprites can understand, b) move Sprite A to Sprite B's position, and lastly c) send a message from Sprite A to tell Sprite B to move to Sprite A's former position. We'll additionally need to understand “global variables”; variables that all Sprites share and understand, in order to complete this skill.



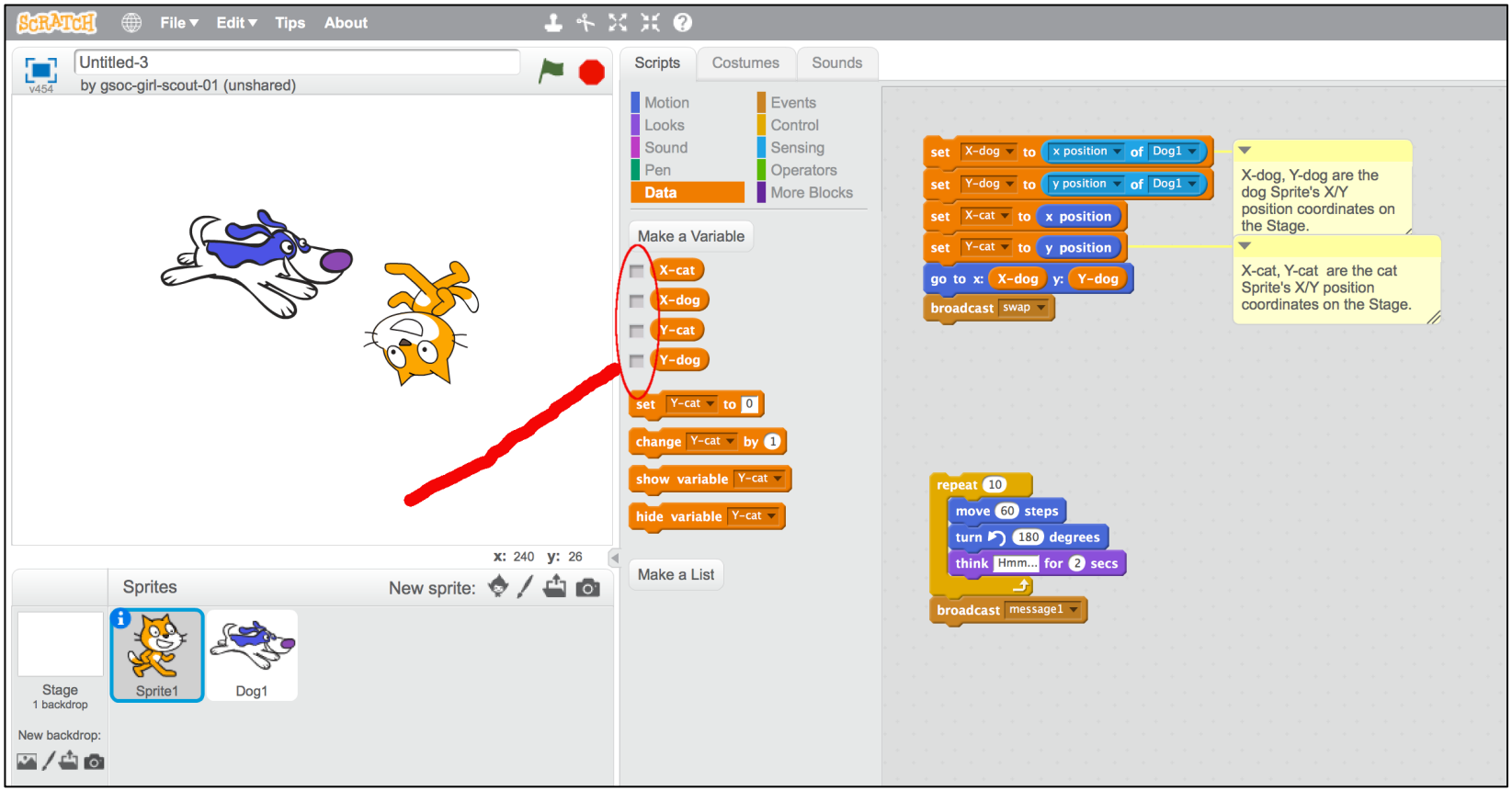
Start by selecting the Data script category and clicking on the **Make a Variable** button. Add the following four variables as GLOBAL variables: X-cat, Y-cat, X-dog, and Y-dog. Note that a checkbox precedes each variable as you create it and that the checkbox is initially checked. If you remove the check, then the variable's value is no longer displayed on the Stage. Go ahead and uncheck all four of these variable's checkboxes as we do not need to display their values on the Stage at this point.

These four variables will contain the X (horizontal) and Y (vertical) Stage coordinates of our cat and our dog. Now we need to assign values to these variables.

GSOC STEM – “Scratch” Programming Workshop – The Tile Game

Getting Started with Scratch (continued)

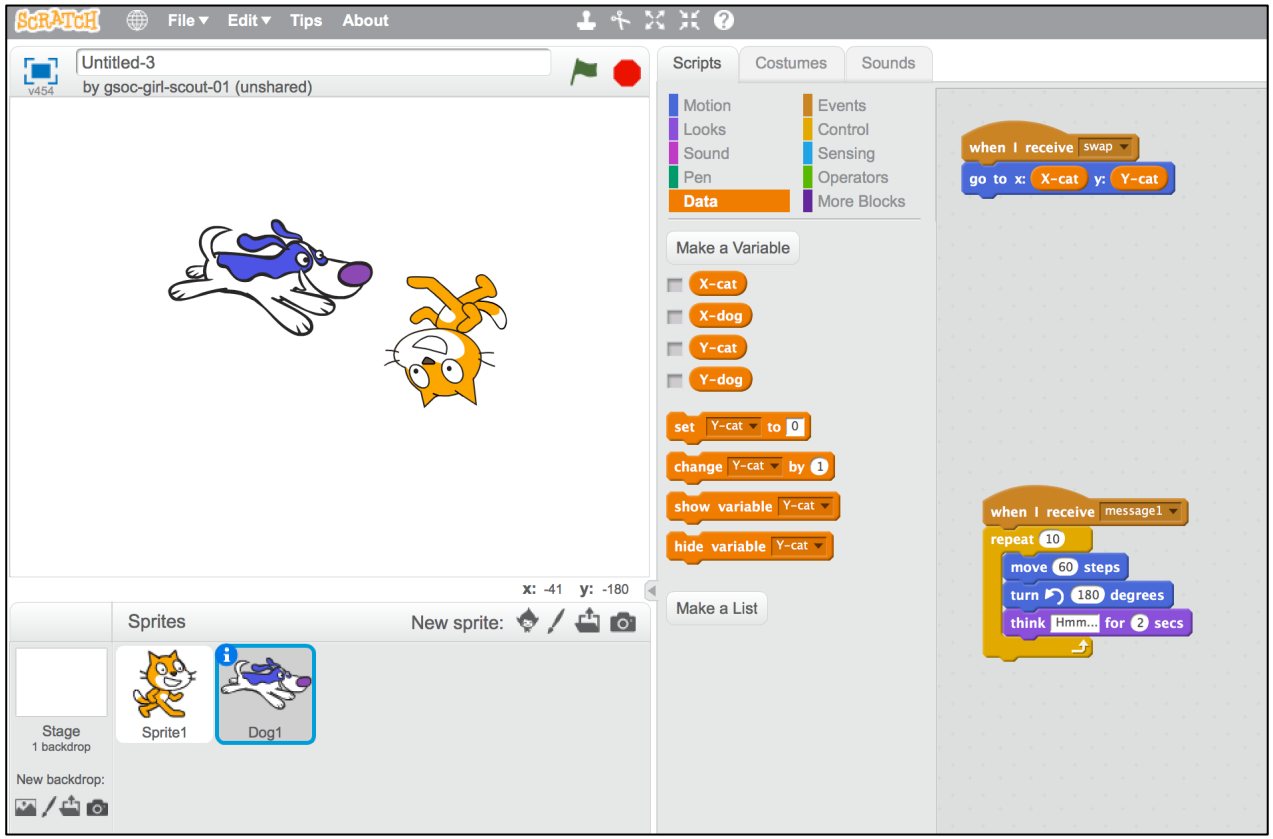
Here we see our four new variables (unchecked). Now we need to select the cat Sprite to set the values of these four new variables. Within the cat Sprite’s script, we drag four of the Data script category “set to” blocks to the cat Sprite’s script, and assign each one to each of our four new variables as shown below. We then use the Motion script category block “go to x/y” to move our cat Sprite to the dog Sprite’s x/y coordinates, thereafter broadcasting a new message we’ve named “swap”.



GSOC STEM – “Scratch” Programming Workshop – The Tile Game

Getting Started with Scratch (continued)

Now click on the dog Sprite beneath the stage in order to select it as shown, then drag the “when I receive” Events script category block to the dog Sprite’s script. Add the Motion script category block “go to x/y” beneath the “when I receive” and use the variables X-cat and Y-cat. This will move the dog Sprite to cat Sprite’s *former* position, in other words, the position prior to cat’s movement to the dog’s position. Select the cat Sprite’s and then click on the cat Sprite script’s first “set” block to swap the cat and dog Sprites on the Stage. ***Talk through this process step by step so that it is well understood.***



GSOC STEM – “Scratch” Programming Workshop – The Tile Game

Brainstorming the Tile Game (30 minutes) (Preferably use a physical tile game and whiteboard)

Discuss the physical attributes or “rules” of the tile game:

- Each tile has a unique “face” that does not change.
- Each tile has a changeable position within the game board’s “grid”.
- One of the tile spaces within the game board is empty (i.e. the “empty space”).
- Any tile adjacent to this empty space can move into the empty space, but any tile that is not adjacent to the empty space can not be moved at all.
- Once a tile moves into the empty space, the space that tile formerly occupied becomes the new empty space.
- The game is “won” by moving all the tiles back into a state in which the collective face of the tiles is restored to their original state.

How can we emulate the physical game’s attributes within Scratch?:

- Each tile could be a Sprite.
- The empty space could be a Sprite.
- Clicking on a Sprite adjacent to the empty tile’s Sprite swaps their positions.
- Clicking on a Sprite that is NOT adjacent to the empty tile’s Sprite does nothing.
- For any grid position within our game board, we can list the adjacent positions. Using this list, an understanding of the empty space’s Sprite position, and an understanding of our selected Sprite’s position, we can decide if the empty position Sprite and our selected Sprite can swap positions. (see next page for more on this point).
- We will need a way to keep track of all Sprite’s positions as they move about the game board.

GSOC STEM – "Scratch" Programming Workshop – The Tile Game

Brainstorming the Tile Game (continued)
Discuss the 3 row by 3 column game board grid below:

1	2	3
4	5	6
7	8	9

Assume the 9th game board grid position represents the empty space Sprite. Sprites in the game board grid positions 6 and 8 are the only Sprites that can be moved. Clicking on either of these two Sprites would swap their position with the empty space Sprite's position. Further, if we clicked on the Sprite in position 6, we know that the position 6 Sprite will move only if the empty space Sprite happens to be in game board grid positions 3, 5 or 9. We can therefore create a list of the "legal adjacent" game board grid positions of the empty space for each of our selected game board grid positions:

Clicked Position:	Adjacent Positions:	Clicked Position:	Adjacent Positions:
1	2,4	5	2,4,6,8
2	1,3,5	6	3,5,9
3	2,6	7	4,8
4	1,5,7	8	5,7,9
		9	8,6

We will need to capture the information in the table above within a Scratch "variable"!

GSOC STEM – “Scratch” Programming Workshop – The Tile Game

Brainstorming the Tile Game (continued)

Scratch variables that we will need to create:

- A list of adjacent positions that we discussed in the prior slide. Scratch supports Lists variables.
- A list of current game board positions of each of the Sprites (including the empty space Sprite).
- In order to swap Sprites (as per our cat and dog work) we will need:
  - The X and Y Stage positions of the empty space Sprite.
  - The X and Y Stage positions of the selected (i.e. the “clicked on”) Sprite.

Sprites that we need to create:

- One Sprite per game board grid position (i.e. 9 sprites). Scratch allows imported pictures to form Sprites, we will import 8 parts of an image and one “blank space” image as the empty space.
- One of the 9 Sprites will be represented by the empty space (in other words we will only have 8 “face” Sprites and one “empty space” Sprite.)

Logic that the Sprites will require:

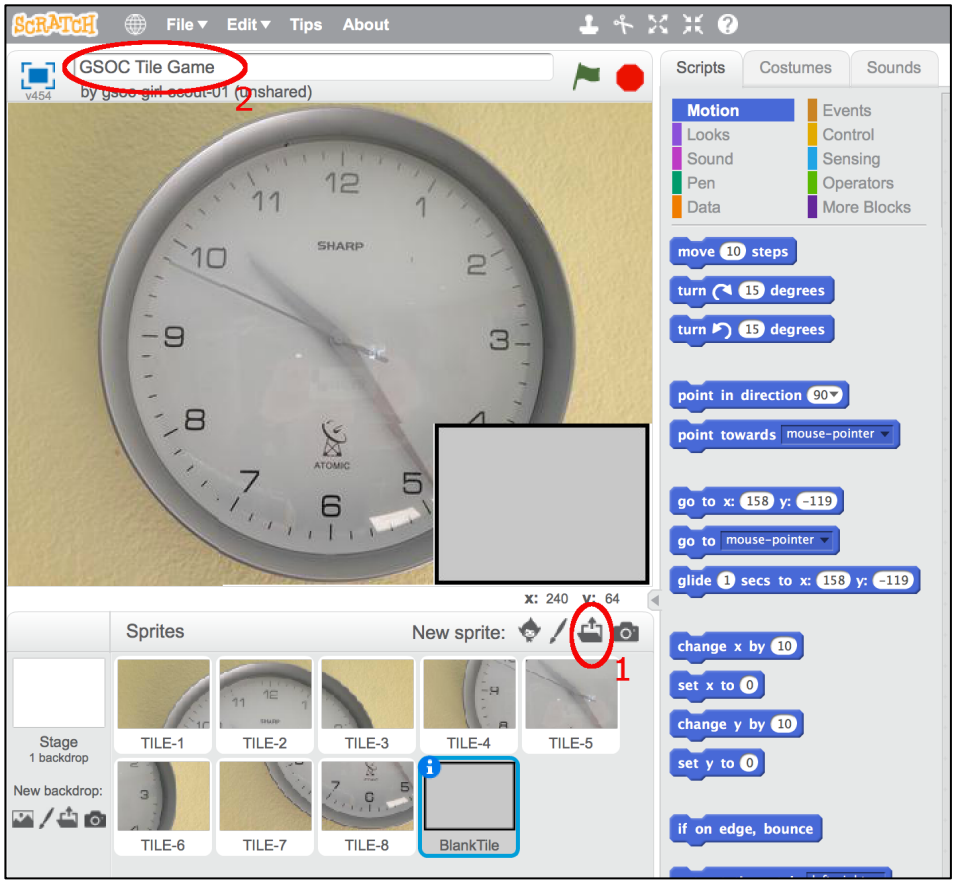
- When clicked, determine if the selected Sprite is adjacent to the empty space Sprite. We will use our “adjacent positions” list to determine this.
- If the empty space Sprite is adjacent (i.e. is within the adjacent positions list for the selected Sprite’s game board grid position), then swap the positions of the selected Sprite and the empty space Sprite.
- Update the list that contains the game board grid position of the two Sprites that moved..

Now let’s start coding our game!

GSOC STEM – “Scratch” Programming Workshop – The Tile Game

Coding the Tile Game in Scratch (60 minutes)

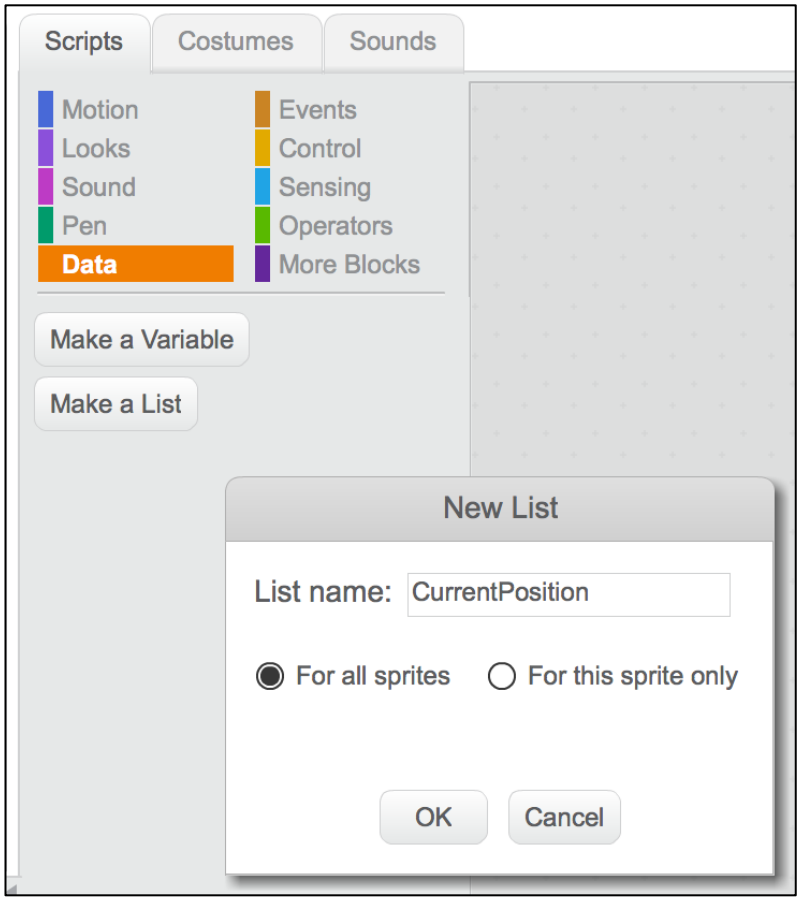
Scratch menu items File New creates a new Scratch project.
Right click on the Cat Sprite and select delete from the right click menu (we don’t need our cat!)
Click on the file folder tool to the right of New Sprite (see callout #1 below)
Browse in TILE-1 through TILE-8 and BlankTile files from the images folder of the GitHub repository.
Give your project a name as per callout #2 below. Drag the tiles on the Stage as shown.



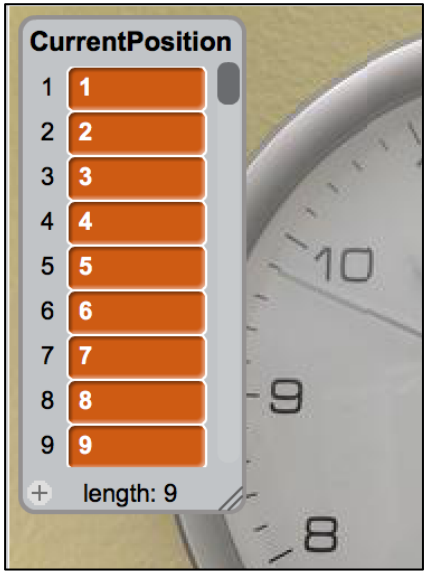
GSOC STEM – "Scratch" Programming Workshop – The Tile Game

Coding the Tile Game in Scratch (continued)

Let's create the variables or "data" we'll need for this project. From our previous work we know we will need to create two lists; one containing a list of the legal adjacent grid positions for the empty spaces, the other a listing of the current grid positions of our tiles. We know we also need to create four variables, the X-position and Y-position of the empty space and the X-position and Y-position of our selected tile. Click on the Data category within scripts to begin, the click on the **Make a List** button.



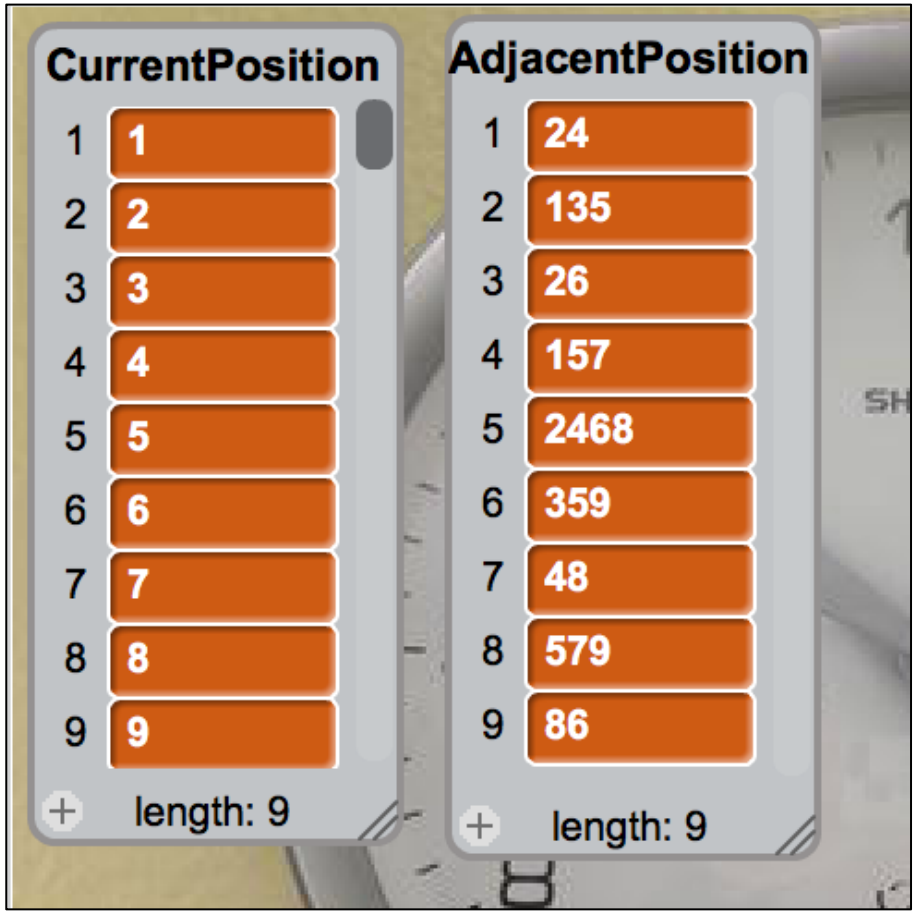
Let's call the first of our new list variables "CurrentPosition". Each item within the list represents a tile face; item #1 corresponds to TILE-1, item #2 to TILE-2, etc. We will reserve item #9 for the BlankTile. Click on the plus sign at the bottom of the list variable created on the stage to enter the values. Note that you can stretch the variable to display all nine values by dragging the bottom right of a list variable.



Since our tiles are currently arranged "in order" the values within CurrentPosition are in order as well.

Coding the Tile Game in Scratch (continued)

In the same fashion, create an additional list variable AdjacentPosition and enter the “legal adjacent position” values we discussed previously.



Interpretation of the values in AdjacentPosition:

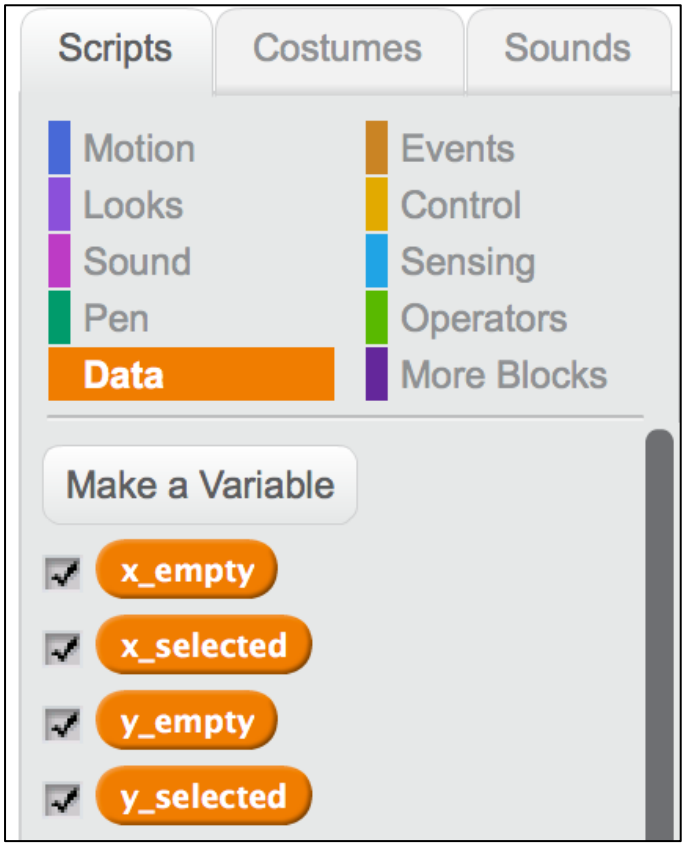
Item #1’s values of 2 and 4 corresponds to the legal adjacent positions of the tile that is in position 1 of the game board grid. Since we only have 9 game board grid positions, each of the legal adjacent positions is conveniently represented using a single digit.

Our script logic will “loop over” the individual digits (e.g. 2 and then 4) within an item of the AdjacentPosition list to determine if the empty space current resides within one of the game board grid adjacent positions. (this will be come more apparent as we build and review our script.)

You can “uncheck” the boxes for the CurrentPosition and AdjacentPosition variables once you verify they appear as show here.

Coding the Tile Game in Scratch (continued)

Create the other four variables we noted earlier using the Make Variable button. You can uncheck the checkmarks ahead of these variables so they do not display on the Stage.



Now let’s get started on our script! Be sure that TILE-1 is selected beneath the stage, click on TILE-1 so that it appears highlighted. You’ll note that the scripting block for TILE-1 is empty.

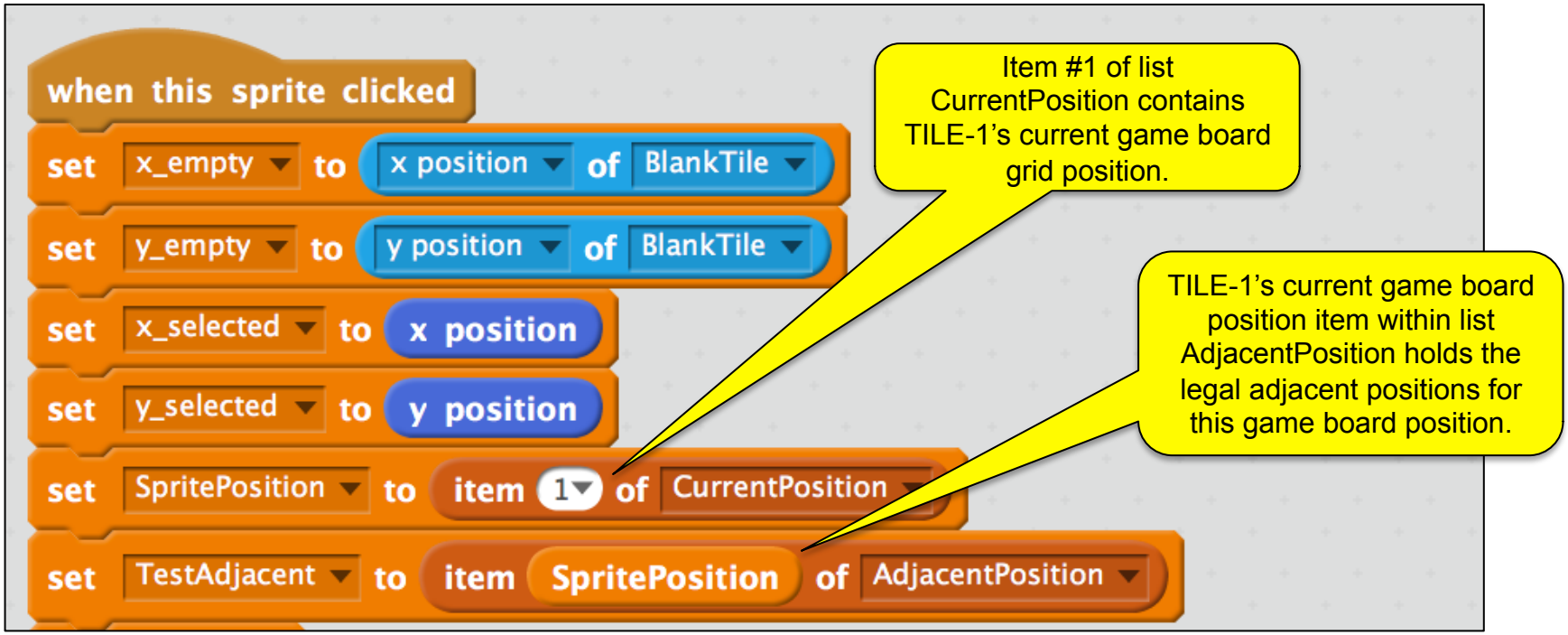
Let’s talk this through! When TILE-1 is selected (clicked) we want to store the current X and Y position of the tile within the x_selected and y_selected variables. We also want to store the current empty space tile’s position within the x_empty and y_empty variables. These variables will facilitate our ability to swap the positions of TILE-1 and the empty space tile.

After swapping, we will then need to update the CurrentPosition list variable with the new game board grid positions of TILE-1 and the empty space tile. TILE-1’s current game board grid position will be always be stored in item 1 of CurrentPosition, and similarly, the empty space (i.e. BlankTile) game board grid position will always be stored within item 9 of the CurrentPosition list variable.

Coding the Tile Game in Scratch (continued)

Our TILE-1 script starts with the Event category block “when this sprite is clicked”, followed by setting the variables we just discussed, using the Data category “set variable to” block, the Motion category x position and y position blocks, and the Sensing category’s “position of sprite” block to keep track of our empty space Sprite BlankTile.

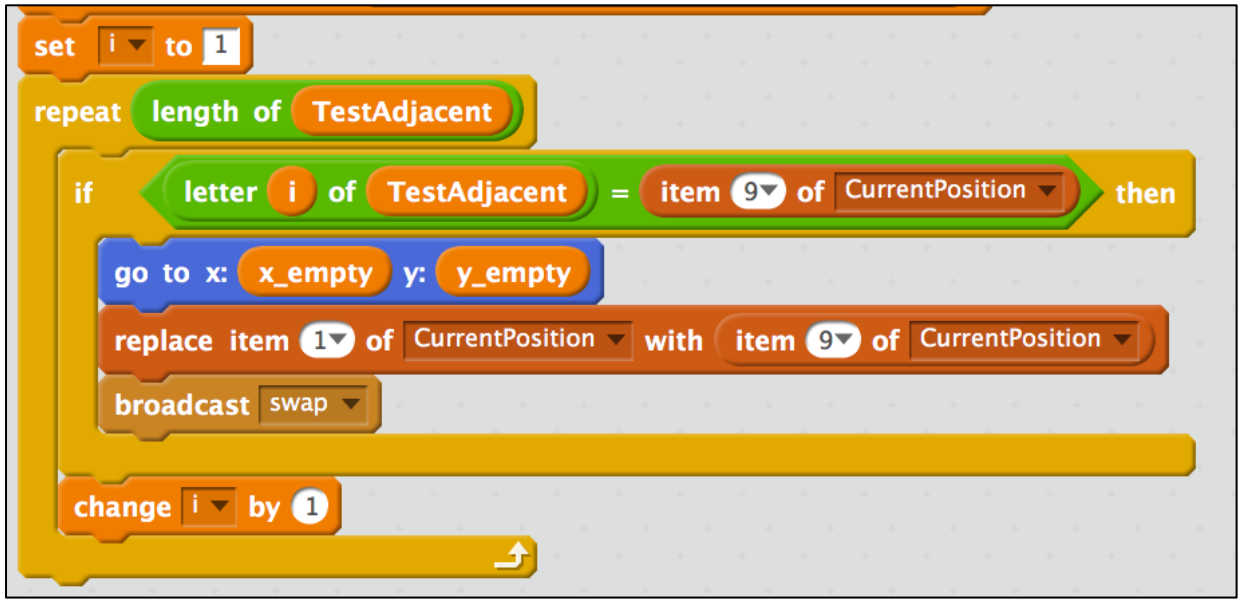
We obtain the legal adjacent positions for TILE-1’s current position from the Adjacent position block. For convenience, let’s create an additional variable TestAdjacent to hold the digits associated with legal adjacent positions of the game board grid position TILE-1 is currently occupying. Let’s also create another variable named SpritePosition to represent the current game board grid position of TILE-1.



Coding the Tile Game in Scratch (continued)

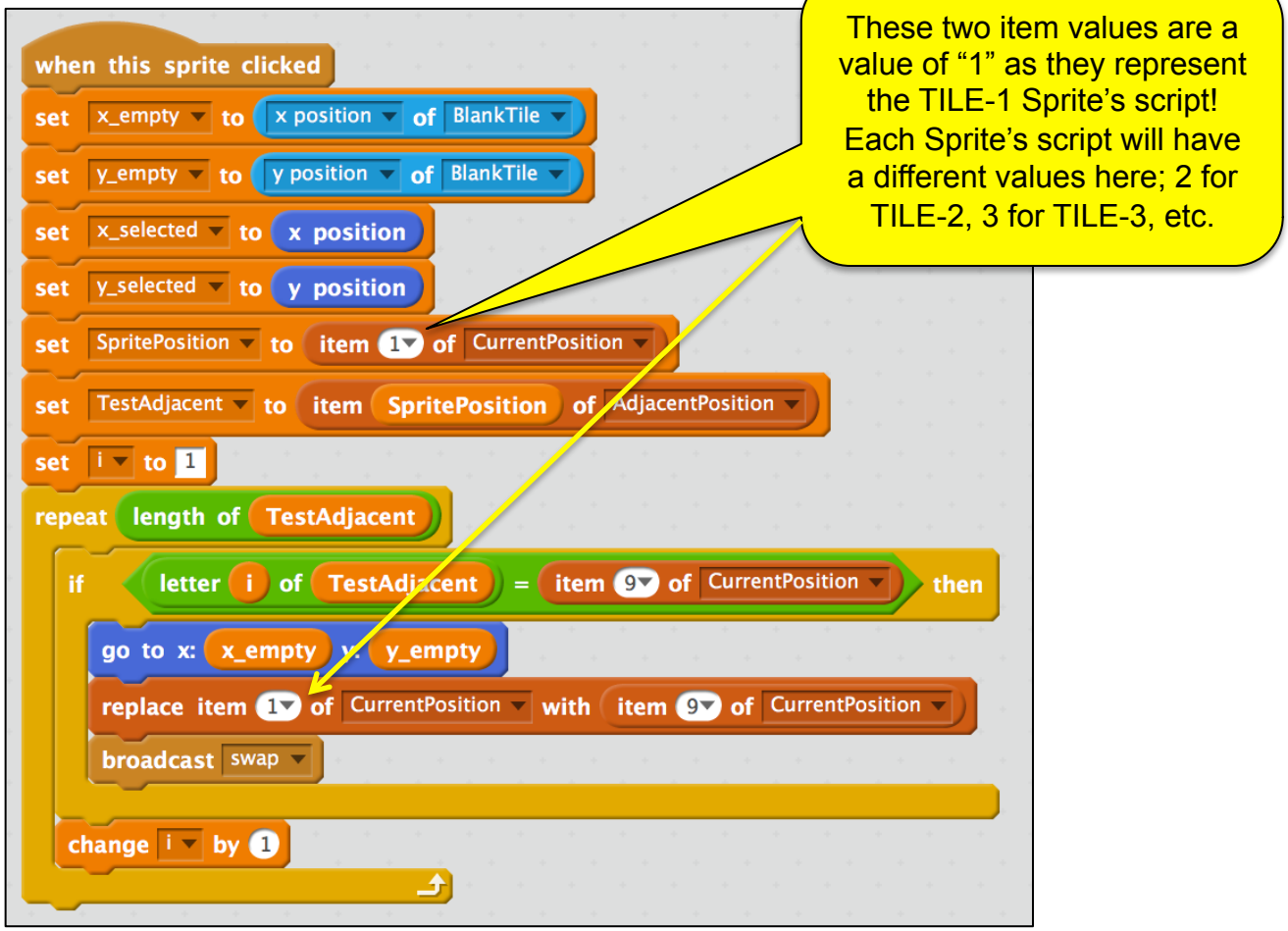
Now let’s implement the “logic” for swapping our selected tile (here TILE-1) Sprite and the empty space Sprite. We create the additional new variable “i” to represent the individual digits within the TestAdjacent variable. Using the Control category repeat block, we “loop over” each of the individual digits within the TestAdjacent variable, testing each individual digit, represented by variable “i”, by way of the Control category “if” block.

If the currently selected (i.e. the “i-th”) TestAdjacent digit corresponds to the current game grid position of the empty space (i.e. item #9 of the CurrentPosition list variable) then we know that our selected tile is currently adjacent to the empty space and we can perform the swap. We use the Motion category “go to” block to move the selected tile to the current empty space position, and broadcast the swap message to move the empty tile to the selected tile’s current position. We update the CurrentPosition list to keep track of the selected tile’s new game board grid position.



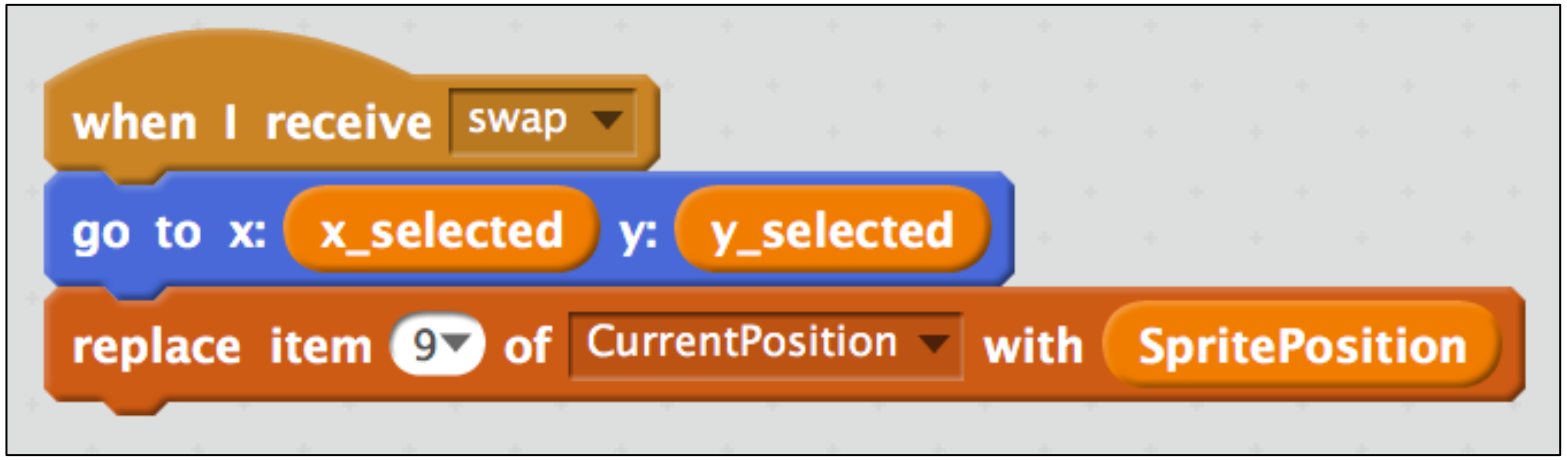
Coding the Tile Game in Scratch (continued)

Here's the entire script for Sprite TILE-1, combining the components of the prior two slides. We can't forget to increment our "i" variable which tracks each of the individual digits of the AdjacentPosition list values! Note that this incrementing needs to happen OUTSIDE of the Control "if" block, but INSIDE of our repeat loop.



Coding the Tile Game in Scratch (continued)

Click on the BlankTile Sprite beneath the stage to select it and edit this Sprite's script. Our BlankTile Sprite needs to listen for the swap message; using the Event script block "when I receive …". When the swap message is received, we move the BlankTile Sprite (i.e. the empty space) to the prior position of the selected (in our prior example, TILE-1) Sprite. We then keep track of the new game board grid position of the BlankTile Sprite using the Data script block's "replace item ... of list" block to update item 9 of the CurrentPosition list (which always holds the current game board position of the empty space) with the prior position of the selected Sprite, which we had saved to the SpritePosition variable.



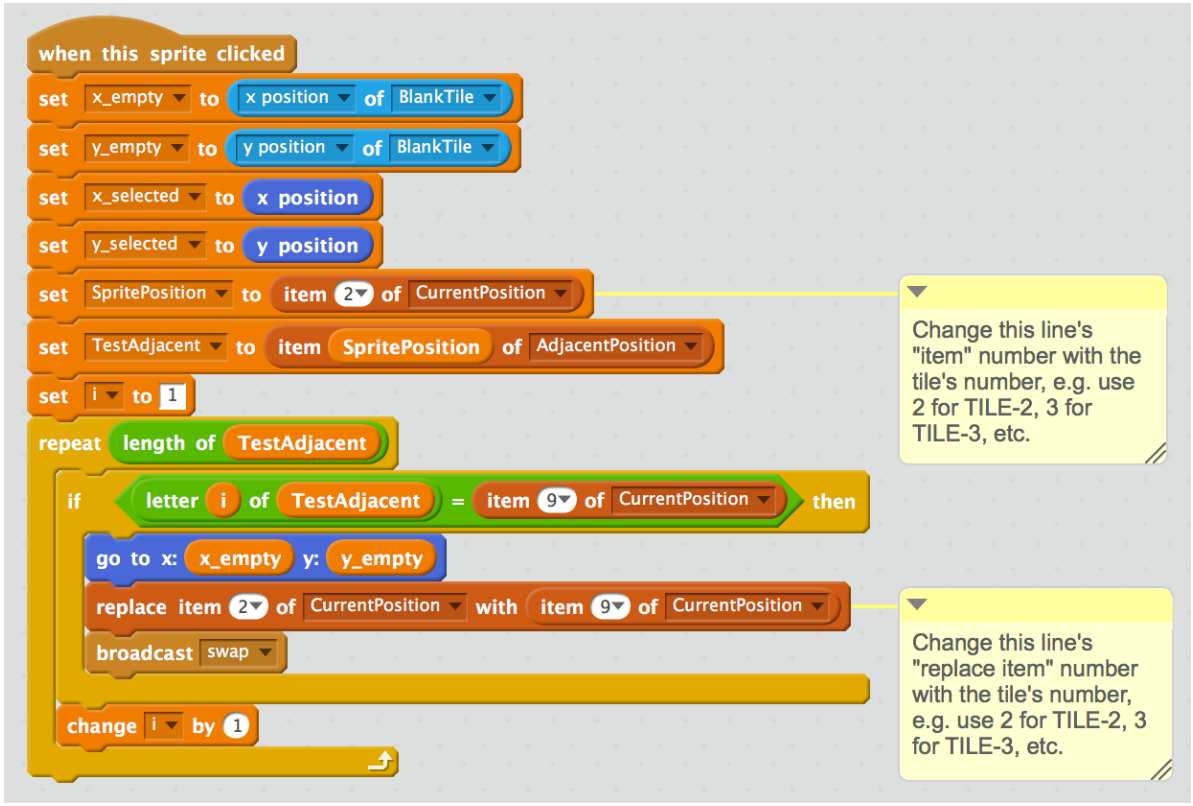
Now we only need to replicate TILE-1's script to TILE-2 through TILE-8, using Scratch's script "backpack" feature. Do you recall when we copied the cat Sprite's script to the dog Sprite? Same idea!

GSOC STEM – “Scratch” Programming Workshop – The Tile Game

Coding the Tile Game in Scratch (continued)

Click on the TILE-1 Sprite beneath the stage to select it. As per our cat and dog Sprite example, expand the Backpack beneath TILE-1’s script window and drag the entire TILE-1 script block into the Backpack. Now select the TILE-2 Sprite beneath the stage and drag the script block from the Backpack into TILE-2’s script window.

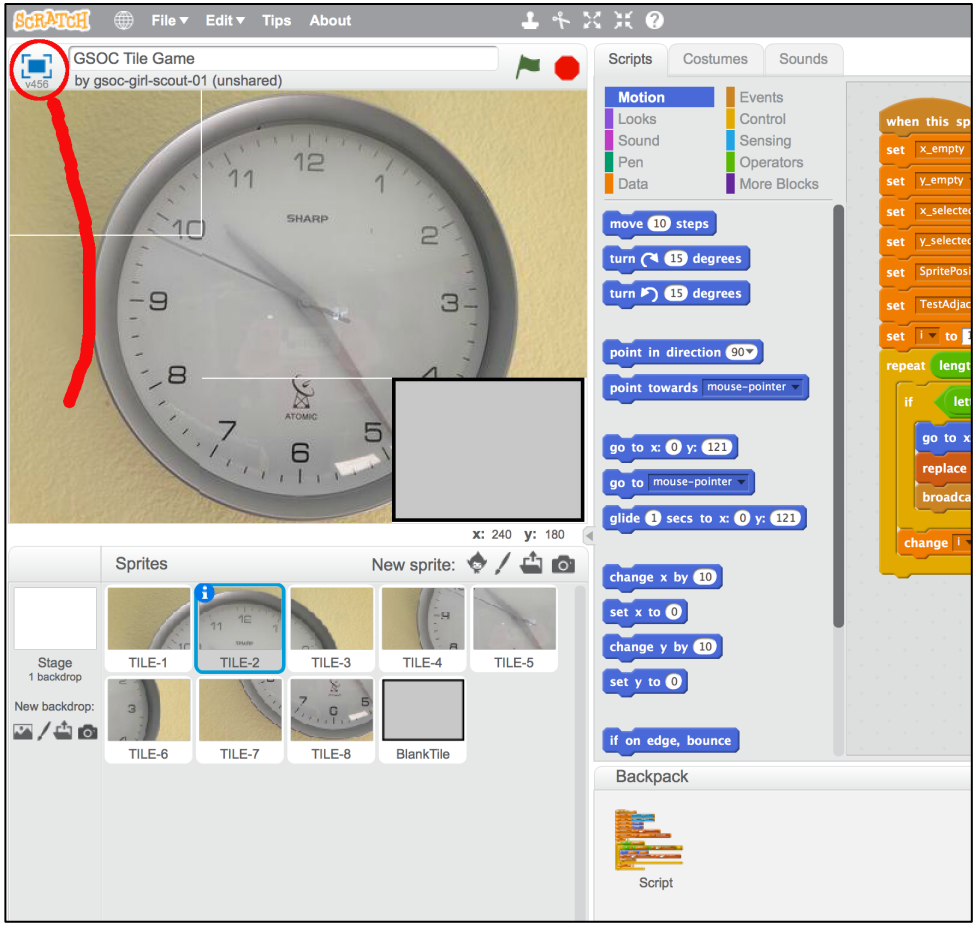
You then need to modify the two “item” positions called out previously on each individual tile’s script. The script below was modified appropriately for TILE-2, changing the item values from 1 to 2 (for TILE-2).



GSOC STEM – “Scratch” Programming Workshop – The Tile Game

Coding the Tile Game in Scratch (continued)

Repeat this process, dragging from the Backpack and modifying the script in two places, for Sprites TILE-3 through TILE-8. Now we’re ready to play! Click on the “Full Screen” tool show on the top left side of the stage area. This will change the view from the “development mode” to “playing mode”. Click on one of the tiles adjacent to the blank space and see if your scripting works as you’d expected!



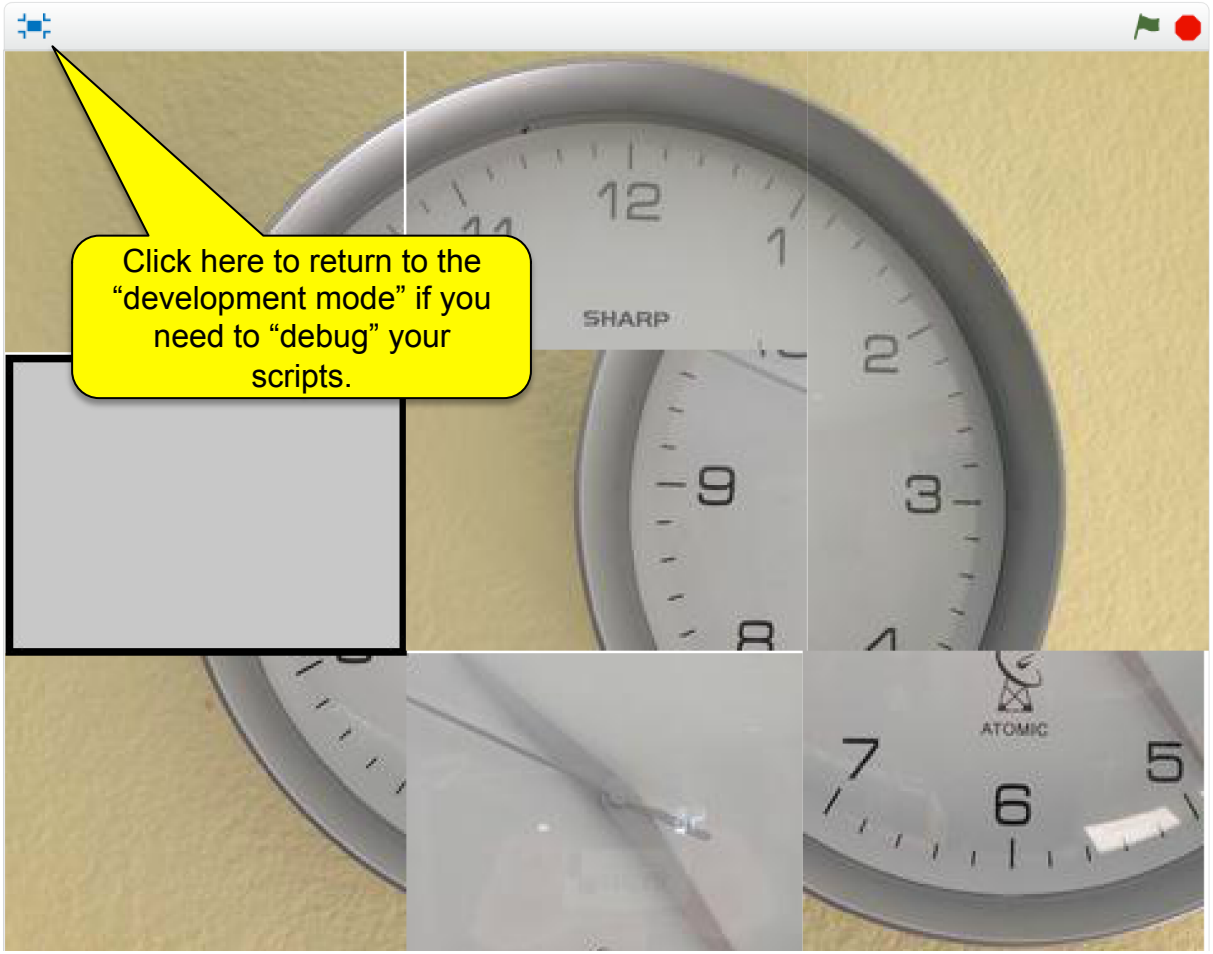
Don’t be surprised or discouraged if it doesn’t work the first time !
Programming inherently involves “debugging”, even when your code is just a few lines long.

Real programmers love the challenge of trying to figure out why their code doesn’t work as they’d expected! Switch back to the “development mode” by clicking this same button from the “playing mode” and try to use this document and trial and error to figure out what went wrong.

You may find it useful to turn the Data script block checkboxes back on so that you can see the values of the variables while your script is running. You can run the script (i.e. click on a tile) from within the development mode.

Coding the Tile Game in Scratch (continued)

Once you get the game working as expected, click on the tiles adjacent to the empty space until they are thoroughly mixed up. Take a break for a few minutes, then come back to your game and see if you can click on the tiles to recreate the original image. It’s not as easy as it looks!



GSOC STEM – “Scratch” Programming Workshop – The Tile Game

Coding the Tile Game in Scratch: WRAP UP AND NEXT STEPS

1) Programmers enjoy finding ways to improve their code or scripts. Generally this means making their scripts run faster or contain fewer “blocks”. How could we improve on our script?

2) Game designers enjoy adding new features and abilities to their game designs. What additional features could we add to this game?

3) Now that we know more about what Scratch can do, what other games or applications could you envision developing from Scratch?

4) Time permitting, have the leader show the girls how to search for inspirational projects that others have created.

(1) NOTE ABOUT Scratch’s BACKPACK feature: Backpack as illustrated within this workshop presentation is available only on the online version of Scratch. The desktop version does not expose the backpack menu beneath the scripting window, HOWEVER, you can “drag and drop” a script from the script editor on to any Sprite beneath the stage to perform the equivalent operation (e.g. copying the script).