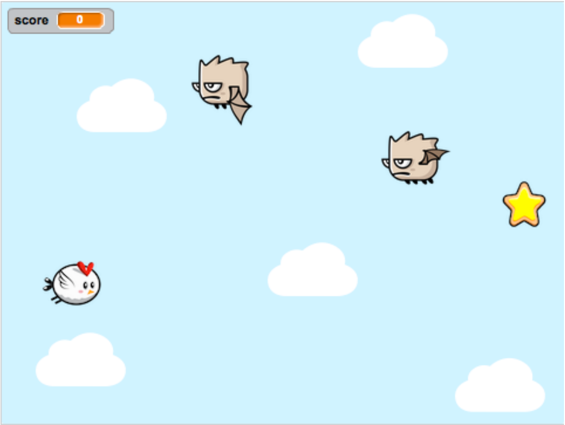
Making Cookie Bird

UoN CS4S Introductory – Visual Programming

Introduction

In this tutorial we will create a game in Scratch, titled *Cookie Bird*. The aim of the game is to get the flying chicken to the star-shaped cookie on the right hand side of the screen without running into any bats. Each time the chicken reaches the cookie you will get +1 to your score. The current score (0) is pictured in the top left of the screen in the image below. If the chicken runs into one of the bats the score will reset back to 0 and the chicken will move back to where it started from.

The finished game will look something like the image below:



Acknowledgements

This tutorial is an extended version of the activity *"Make a Simple Game"* available at: https://scratch.mit.edu/help/videos/

The chicken and bat graphics were created by Bevoulin (http://bevouliin.com/). The cookie graphic was sourced from Game Art Guppy (http://gameartguppy.com/), and the chicken sounds have been sourced from SoundBible (http://soundbible.com/).

Resources

The resources for this activity can be downloaded from the following link: https://goo.gl/to1rX7

Please download the .zip file from the above link. The download button is shown in the image below:

Extract the zip folder (right click on the file, and select *Extract here*) in the *Downloads* folder or *My Documents* folder on the lab computer, once it has finished downloading. Note that if you have a Scratch account you can save your project, and you will be able to access your work from home. The completed project is also available as a file (*Cookie Bird.sb2*) you can download and upload later in the activity's resources, available here: https://goo.gl/Cy5OPi

Getting Started

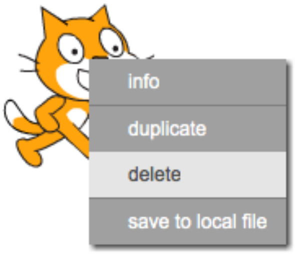
This tutorial assumes you have completed the *Introduction to Scratch* activity. If you would like to keep a copy of the program you created in the *Introduction to Scratch* activity, you will be able to download it and upload it later. Or, if you have registered a Scratch account, you can save the program online so you can access it through your account later. For further instructions on how to do this please ask one of the workshop assistants.

If you are ready to begin making *Cookie Bird* click **File > New** in the Scratch Editor. When you click this option a popup may display with the message *"Discard contents of the current project?"* If you have saved your program, or don't want to keep a copy of the Introduction to Scratch program, click OK. If you haven't saved your program and want to keep a copy of the program, please ask one of the workshop assistants for assistance.

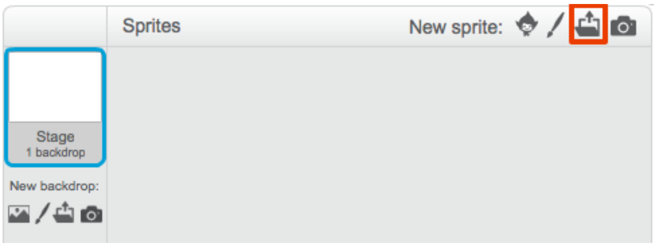
1. Moving the Chicken

Our first step will be to add our chicken sprite to the canvas. We'll then add scripts to allow us to use the keyboard's up, down, left and right keys to move the chicken around the canvas.

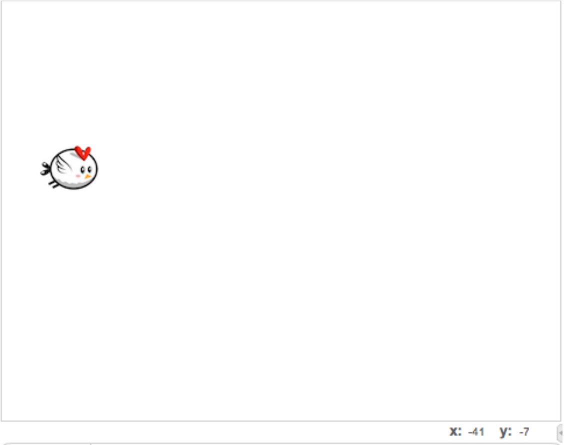
Start by deleting the cat sprite, which can be done by right clicking on the cat sprite and clicking on the *delete* option, as shown below:



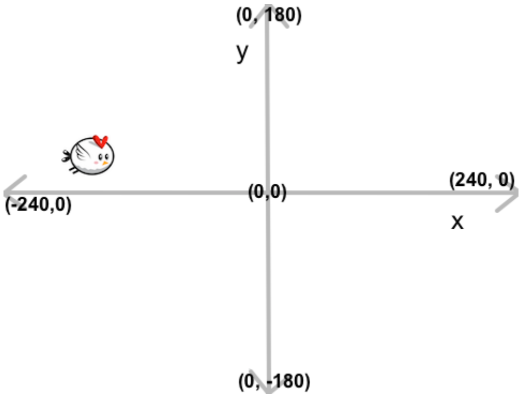
We'll now add the chicken sprite to the canvas. To do this first click on the *"upload icon from file"* button, highlighted in red below.



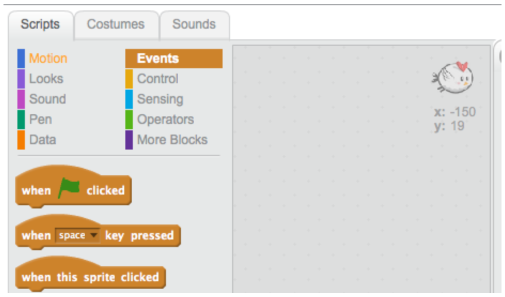
In the Open File Dialog that opens, navigate to where you downloaded the activity resources, select the file named **“chicken_01.png”** and click *Open*. The chicken sprite should now appear on the canvas. Drag and drop the chicken to the left hand side of the screen. The canvas should look something like this now:



Before we add any scripts, let’s have a quick look at how the positioning in Scratch works. Scratch’s canvas is 480 pixels wide, and 360 pixels high. If we were to draw 2 axes (x and y) on the canvas, as shown in the image below, then the points range from (-240, -180) [the bottom left hand corner] to (240, 180) [the top right hand corner]. The point (0, 0) is at the centre of the canvas as shown in the image below.

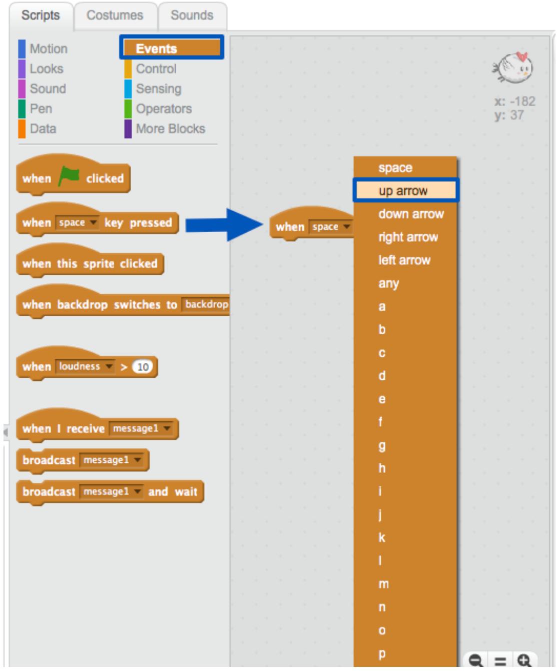


The position of the chicken in the example above is around (-150, 19). You can see the position of the currently selected sprite in the top right hand corner of the Scripts editor, as shown below:



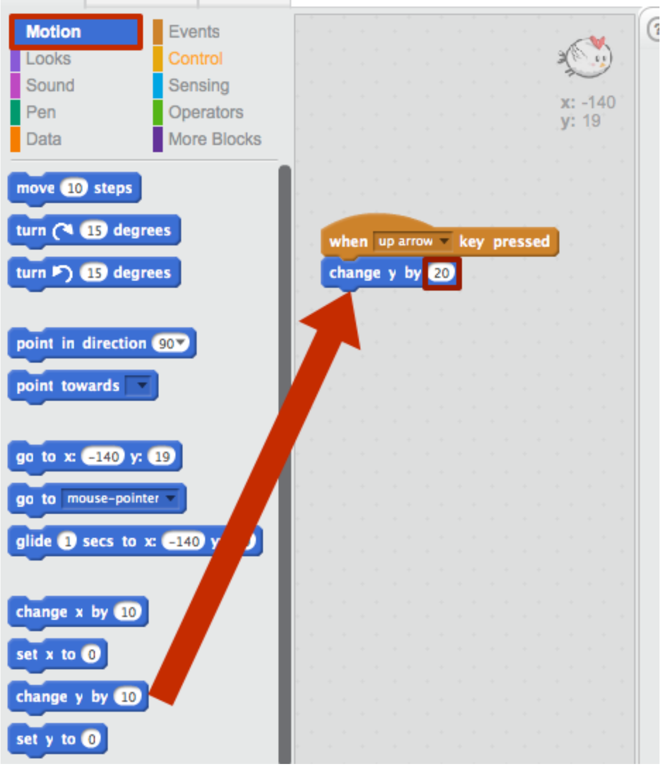
Now we'll add scripts to move the chicken around the canvas.

Select the **Events** section in the *Scripts* tab, then drag the *"when space key pressed"* block over to the scripts area, as shown below.



In the dropdown (where it says *"space")* click and select the *"up arrow"* option instead of *"space"*, as highlighted in blue in the image above.

Next, select the **Motion** section, and drag a *"change y by 10"* block and snap it in place under the *"when up arrow key pressed"* block you placed in the last step as shown below. Next, click the *"10"* in the *"change y by 10"* block and type in *"20"* instead.

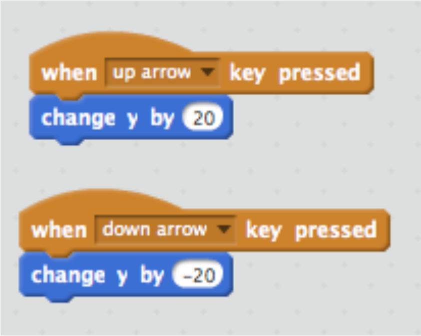


This means that when you tap the up arrow key on your keyboard the chicken sprite's y position will change by 20 – so it should move towards the top of the canvas. Note that we can't move the chicken downwards, so if it reaches the top of the canvas it won't be able to move at all anymore.

To add downwards movement, we'll repeat the adding of the two blocks in the previous step, but this time we will:

- Change the *"space"* in the *"when space key pressed"* block to the *"down arrow"* option
- Change the *"10"* in the *"change y by 10"* to *"-20"* instead

After doing this, you should have the following blocks in your scripts area:



Now, tapping the up and down arrow keys will move the chicken up and down.

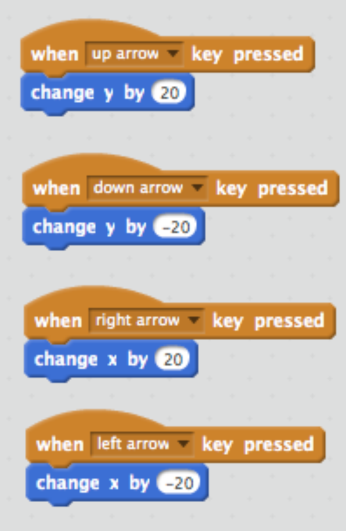
Next, we will add scripts to move the chicken right and left across the screen.

This can be done by using the *“change x by 10”* blocks, along with the *“when key pressed blocks”.*

To move the chicken right we need to add a *“when right arrow key pressed”* block, and place a *“change x by 10”* block underneath it in the scripts area. Then the *“10”* in the *“change x by 10”* block should be changed to *“20”.*

To move the chicken left we need to add a *“when left arrow key pressed”* block, and place a *“change x by 10”* block underneath it in the scripts area. Then the *“10”* in the *“change x by 10”* block should be changed to *“-20”.*

Your scripts area should look similar to the image below:



```
when up arrow key pressed
change y by 20

when down arrow key pressed
change y by -20

when right arrow key pressed
change x by 20

when left arrow key pressed
change x by -20
```

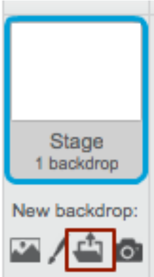
You should be able to move the chicken up, down, left and right with the arrow keys now.

2. Adding a Backdrop and Prize

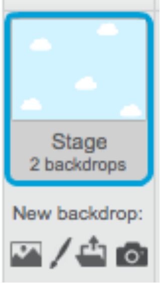
2.1 Adding a Backdrop

Now that we have the chicken moving around the canvas we'll add a sky backdrop for the chicken to fly around in, and a prize for the chicken to collect.

To add a backdrop in Scratch you have to upload an image, draw your own or select an image from the gallery. For this tutorial we'll use a background image that is included in the activity's resources you downloaded earlier. To upload this, click the *"upload background from file"* button highlighted in red in the image below. In the Open File dialog navigate to where you download the activity's resources and select the file **"background.png"**.



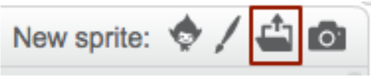
If the image has uploaded successfully you should see the image in the backdrops section, as shown below, and it will be visible on the canvas.



2.2 Adding a Prize

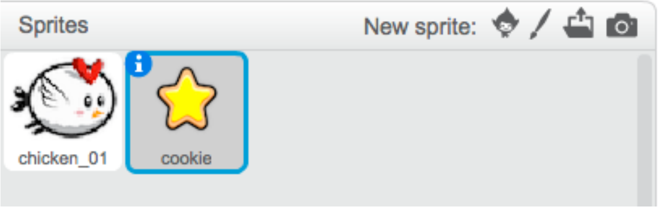
Next we will add a prize (a cookie) that we want the chicken to fly into. We will add a scoring system later in the tutorial. But, for now when the chicken flies into the cookie we will move the chicken back to its original position, without adding to the player's score.

To do this, we'll upload a new sprite, as we did with the chicken sprite earlier in the tutorial.



To upload the prize graphic, first click the *"upload sprite from file"* button (highlighted in red in the image above). Then navigate to where you downloaded the resources, and go to the directory **"Day 1/Visual Programming/Resources"** and select the **"cookie.png"** file.

After uploading the image, the cookie should be visible on the canvas. The cookie will also be shown in the Sprites area, as can be seen in the image below:



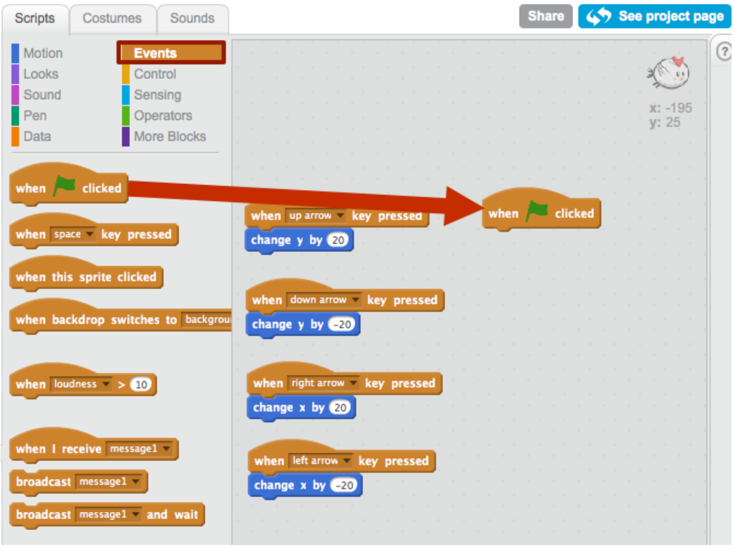
The blue border around the sprite indicates the sprite that is currently selected. The scripts shown on the right hand of the screen will only apply to the selected sprite. For example, when the cookie is selected and you place a *"move 10 steps"* block in the Scripts area and click it, this will make the cookie move 10 steps. However, it will not make the chicken move.

Drag the cookie sprite to the right hand side of the canvas, so that the chicken has to fly across the screen to reach it.

Next, we will add what is often referred to as "collision logic" – scripts that are run when two sprites are colliding with (or touching) each other. When the chicken reaches the cookie we want it to move back to the left hand side, so that the game starts over.

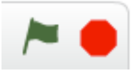
First, make sure that the chicken sprite is selected. To do this, click *chicken_01* in the Sprites area. When selected, a sprite will have a blue border (as the cookie sprite does in the above image).

Next, select the *Events* section in the Scripts tab, and drag a *"when Green Flag clicked"* block into the Scripts area, as demonstrated below:



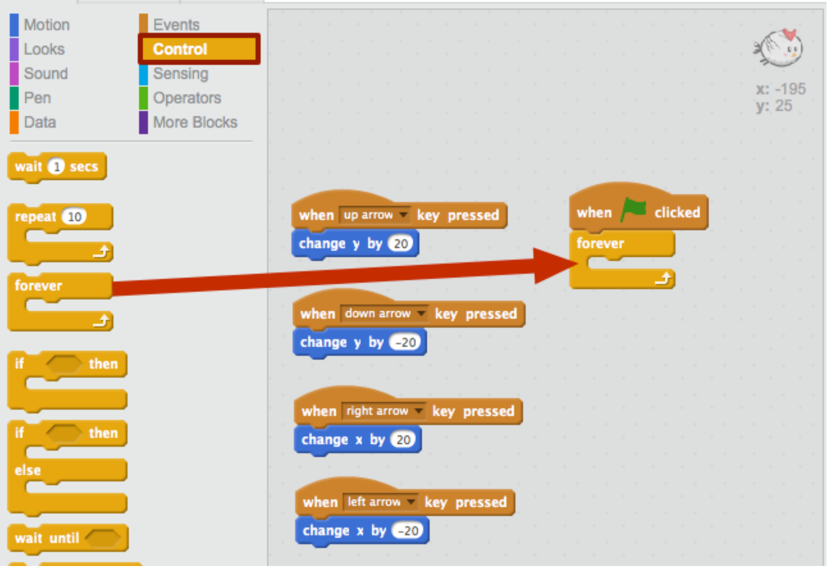
```
when up arrow key pressed
change y by 20

when down arrow key pressed
change y by -20

when right arrow key pressed
change x by 20

when left arrow key pressed
change x by -20

when clicked
```

All blocks that are attached to the "when Green Flag clicked" block will run when the Green Flag button (which is above the canvas to the right, next to the stop button) is clicked.

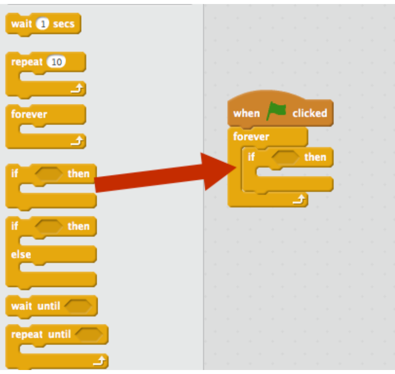


In our game these buttons will act as Play (Green Flag) and Pause (Red stop sign) buttons. So, when the Green Flag is clicked the game will resume from where it left off, and when the Red Stop Sign is clicked the game will pause.

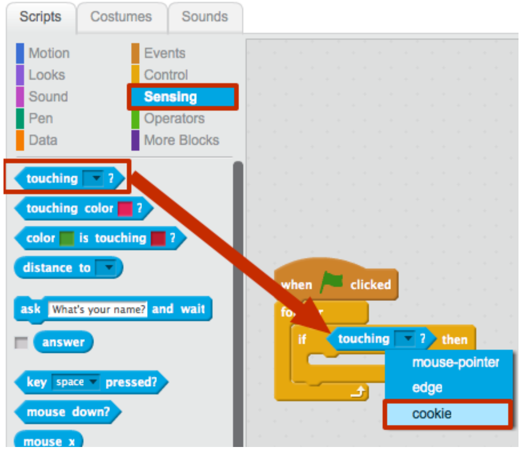
Next, we will add scripts to check if the chicken is touching the cookie. Select the *Control* section in the Scripts tab, drag a *"forever"* block under the *"When Green Flag clicked"* block and click it in place, as demonstrated in the below image:



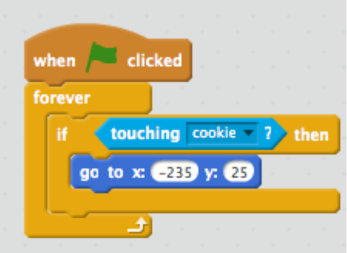
Next, drag an *"if then"* block into the "forever" block, so that the *"forever"* block is wrapped around it, as shown below:



Now we will add blocks to check when the chicken and cookie are *"colliding"*. Start by selecting the *Sensing* section in the Scripts tab. Next, drag a *"touching ?"* block into the first slot of the *"if then"* block, as pictured below. Finally, select *"cookie"* from the dropdown in the *"touching ?"* block.



Next, select the *Motion* section of the Scripts tab, and drag one of the *"go to x: y:"* blocks so that it is wrapped around the *"if then"* block. The result should look similar to that shown below:



```
when [flag] clicked
forever
    if touching cookie ? then
        go to x: -235 y: 25
```

These blocks will continuously check whether the cookie and cookie are touching as long as the game is running. You can tell when the game is running by the colour of the Green Flag. If the flag is bright green then the game is running, if it is dark green it is not running.

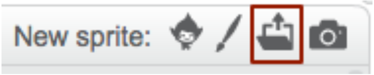
To make sure the chicken moves to the left hand side of the screen after it collides with the cookie, make the following changes to the *"go to"* block:

- Change the *x* value *"-235"*
- Change the *y* value to *"25"*

Now, run the game by clicking the green flag. Move the chicken so that it reaches the cookie – it should move back to the left hand side after the chicken runs into the cookie.

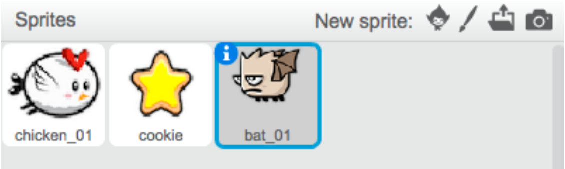
3. Adding an Enemy

Next, we will add an enemy that the chicken will have to pass to get to the cookie at the right hand side of the screen. We'll begin by uploading the bat sprite, by clicking the *"Upload sprite from file"* button in the Sprites area, as highlighted in red below:



In the Open File dialog, navigate to where you downloaded the activity's resources, select and upload the **"bat_01.png"** image.

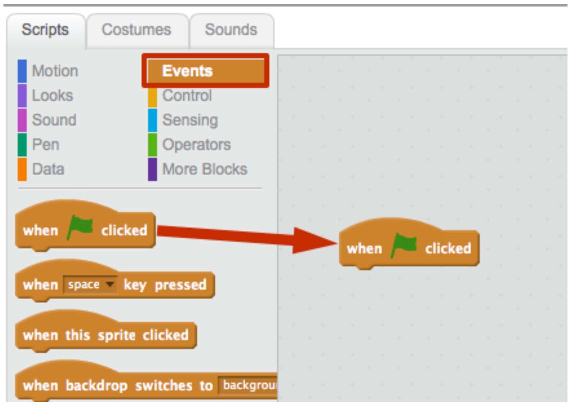
Once it has uploaded, it should be visible in your Sprites area, similar to the below image:



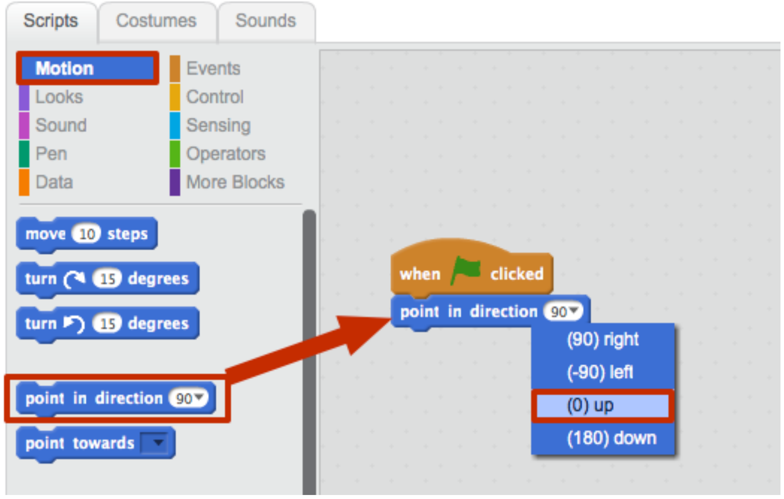
Next, we will add movement to the bat so that it continuously moves up and down the screen, blocking the chicken's path to the cookie.

First, make sure that the *bat_01* sprite is selected. It should have a blue border, like that in the above image. If it's not selected, click the *bat_01* sprite and the blue border should appear.

Select the *Events* section in the Scripts tab, and drag a *"when Green Flag clicked"* block onto the Scripts editor, as shown below.

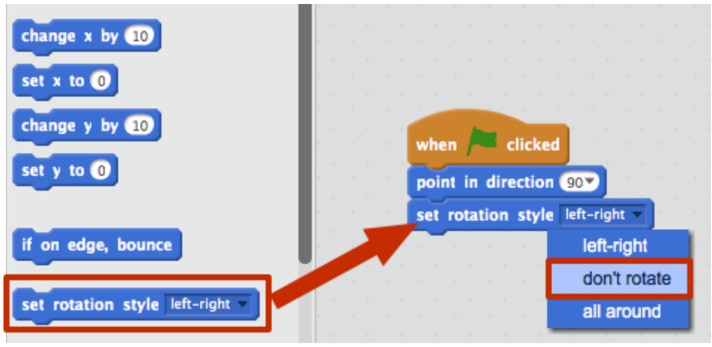


Now we have to add some blocks that will help us move the bat up and down the screen. Firstly, select the *Motion* section in the Scripts Tab. Drag a *"point in direction 90"* block and click it in place under the *"when Green Flag clicked"* block. Click the dropdown and select the *"(0) up"* option. This process is illustrated below:



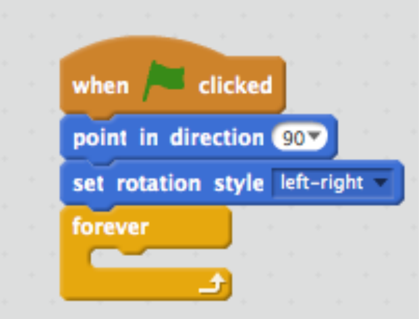
```
when clicked
point in direction 90
```

Next, drag a *"set rotation style left-right"* block from the same (*Motion*) section and place it under the *"point in direction 0"* block. Choose the *"don't rotate"* option from the dropdown. These steps are shown in the image below:



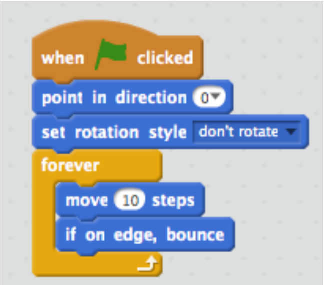
```
when clicked
point in direction 90
set rotation style left-right
```

Next, select the *Control* section in the Scripts tab, drag out a *"forever"* block and click it in place under the *"set rotation style don't rotate"* block. The result of this step is shown below:



```
when clicked
point in direction 90
set rotation style left-right
forever
```

Now, select the *Motion* section in the Scripts tab again, and drag a *“move 10 steps”* block into the *“forever”* block. Next, drag a *“if on edge, bounce”* block and place it under the *“move 10 steps”* block. The *“forever”* block should be wrapped around both the *“move 10 steps”* block and the *“if on edge, bounce”* block, as shown in the below image:



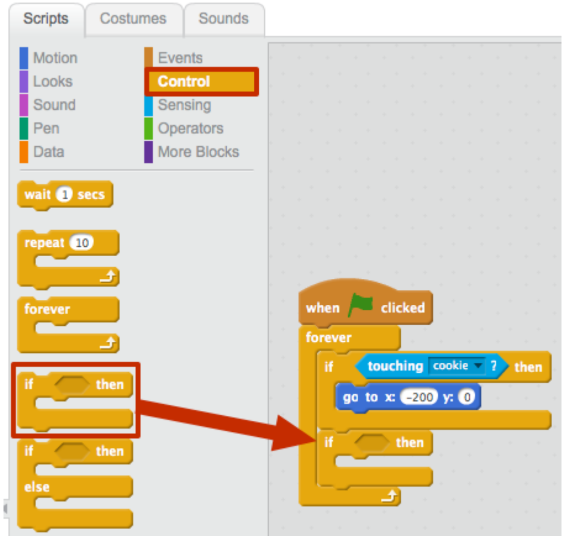
```
when [flag] clicked
point in direction 0
set rotation style don't rotate
forever
    move 10 steps
    if on edge, bounce
```

Now, run the game by clicking the Green flag and the bat should start moving up and down the screen.

Test to see what happens when the chicken runs into the bat. Nothing? This is because we haven’t set up any *collision logic* for the chicken and bat. We will do that next by adding a check to see if they are touching, similar to that we added for checking if the chicken and cookie are touching.

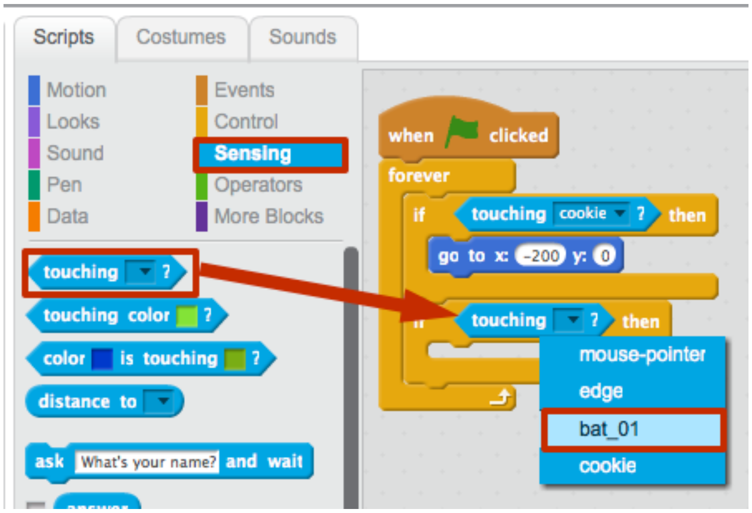
Our first step in adding the check for the chicken and bat are touching, will be to select the *chicken_01* sprite in the Sprites area. If selected, the *chicken_01* sprite will have a blue border.

In the Scripts tab, select the *Control* section and drag a *“if then”* block into the *“forever”* block, underneath the *“if touching cookie then”* block, as pictured below:

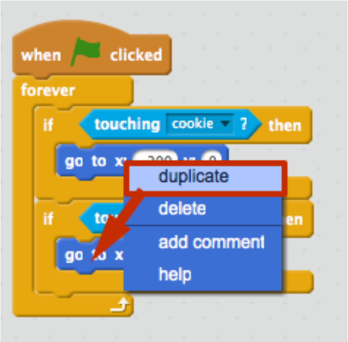


```
when [flag] clicked
forever
    if touching cookie ? then
        go to x: -200 y: 0
    if then
```

Next, select the *Sensing* section in the Scripts tab, and drag a *"touching ?"* block into the slot in the *"if then"* block. In the dropdown select "bat_01", as shown below.



We will now add a block to move the chicken back to its original position when the chicken collides with the bat sprite. We can duplicate the *"go to x: -200 y: 0"* block and use this to move the chicken back to its original position. To do this, right click on the *"go to x: -200 y:0"* block and select *duplicate*. This will create a new block. Move the new block under the *"if touching bat_01 then"* block. This process is illustrated below:



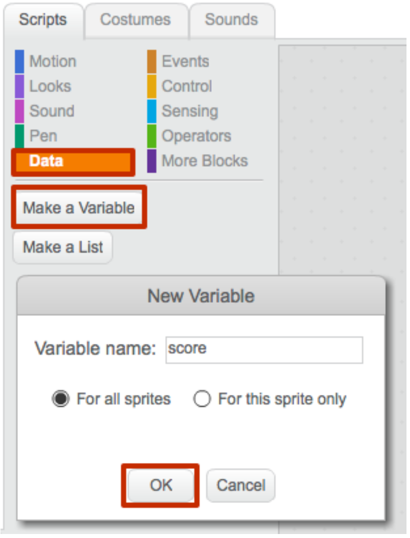
Now, run the game and try and reach the cookie. What happens when you run into the bat? If the above scripts are working as expected, the chicken should return to the left hand side after running into the bat or running into the cookie.

4. Adding a Score

Now we will a scoring system to our game. The scoring will work in the following way:

- Every time you start the game (by clicking the Green Flag) your score will reset to 0
- Every time you reach the cookie you will get +1 point
- Every time you run into the bat your score will reset to 0

To keep track of the current score, we will use what is called a variable. To create our *"score"* variable, select the *Data* section of the Scripts tab, and click *"Make a Variable".* In the *"New Variable"* dialog type in *"score",* keep the *"For all sprites"* option selected and click *"OK".* This process is illustrated below.



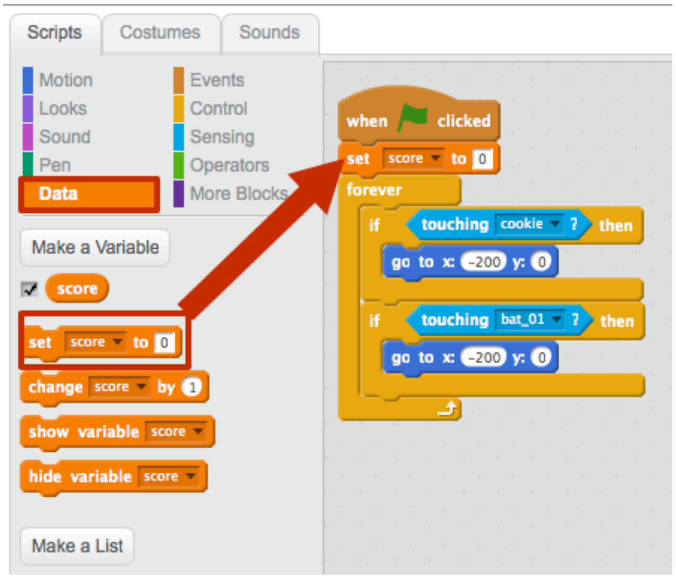
New Variable

Variable name: score

● For all sprites ○ For this sprite only

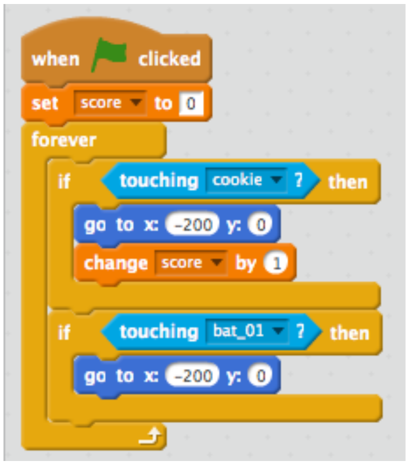
Now, make sure the *chicken_01* sprite is selected in the Sprites area. If it's selected it will have a blue border around it.

To make the score reset whenever the game is restarted, we'll use a *"set score to 0"* block. Drag a *"set score to 0"* block underneath the *"when Green Flag clicked"* block, as shown in the below image.



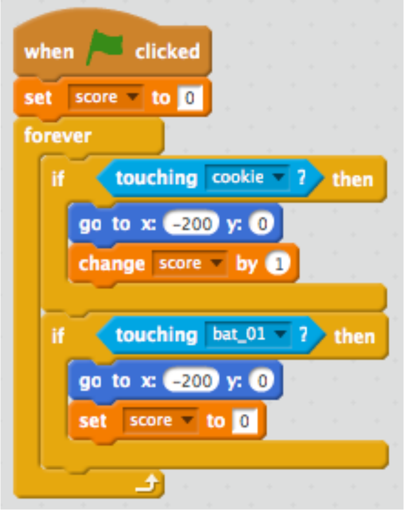
```
when clicked
set score to 0
forever
    if touching cookie ? then
        go to x: -200 y: 0
    if touching bat_01 ? then
        go to x: -200 y: 0
```

Next, we will add a block to increase the score every time the chicken touches the cookie. With the *Data* section of the Scripts tab still selected, drag a *"change score by 1"* block into the *"if touching cookie ? then"* block, placing it underneath the *"go to x: -200 y: 0"* block. The final result should look similar to the image below.



```
when ⚑ clicked
set score ▾ to 0
forever
    if touching cookie ▾ ? then
        go to x: -200 y: 0
        change score ▾ by 1
    if touching bat_01 ▾ ? then
        go to x: -200 y: 0
```

Now we will add a block to reset the score when the chicken collides with the bat. To do this, drag a *"set score to 0"* block underneath the *"go to x: -200 y: 0"* in the *"if touching bat_01 ? then"* block. The resulting blocks should look something like this:



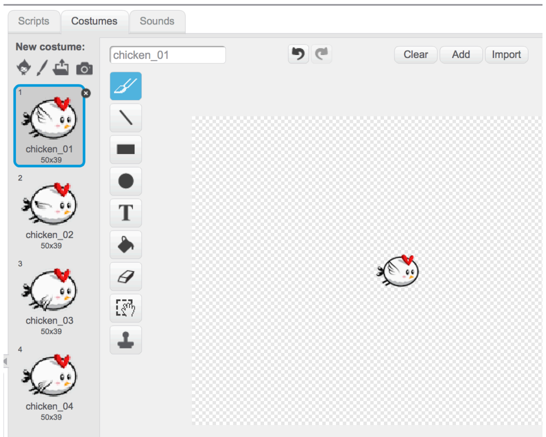
```
when ⚑ clicked
set score ▾ to 0
forever
    if touching cookie ▾ ? then
        go to x: -200 y: 0
        change score ▾ by 1
    if touching bat_01 ▾ ? then
        go to x: -200 y: 0
        set score ▾ to 0
```

Now, every time the chicken collides with the bat the score will be reset.

We now have a fully working game, good work! The next couple of steps will involve adding flying animations to the chicken and bat. Then, we will add more enemies to make the game more challenging.

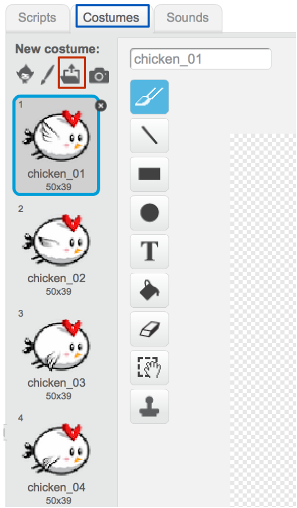
5. Animating the Sprites

Next we'll animate the chicken and bat sprites so they look like they're flying. Each sprite can have a number of costumes, but will only display one at a time. We can use blocks to cycle through a sprite's different costumes to show an animation. An example is shown below – the *chicken_01* sprite's current costume is the **chicken_01.png** file. But the *chicken_01* sprite has 3 other costumes (**chicken_02.png**, **chicken_03.png**, **chicken_04.png**). If we set up blocks to cycle through the different costumes it will make the chicken sprite look like it is flying.



Our first step for getting the chicken sprite to animate will be to upload the different costumes shown above.

First, make sure that the *chicken_01* sprite is selected in the Sprites area. If it's selected the border around it will be blue. Once the *chicken_01* sprite is selected, click the **Costumes** tab (near the **Scripts** and **Sounds** tab). Next, we can upload the different costumes by first clicking the *"upload costume from file"* button, highlighted in red in the below image. In the Open File dialog navigate to where you downloaded the activity's resources and select all of the chicken images (**excluding chicken_01.png**). Upload all of these, either all at once or one-by-one.

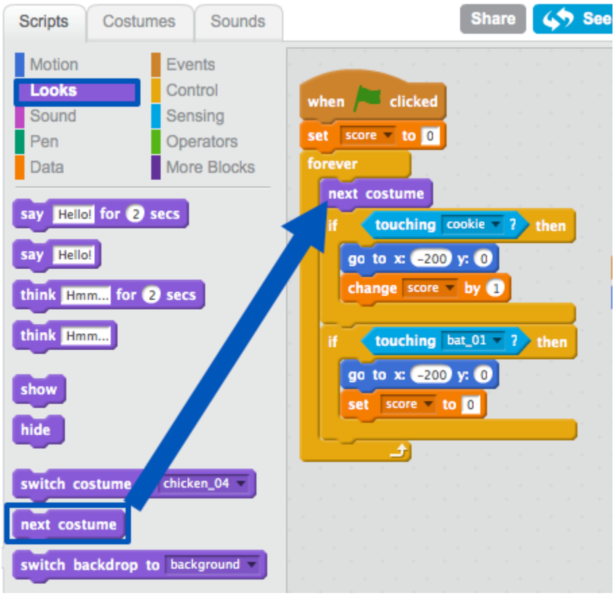


The order of the costumes should match that above. If they're out of order the costumes can be dragged and dropped to be rearranged in to the correct order.

Next, we will add blocks to the Scripts editor to switch between the costumes.

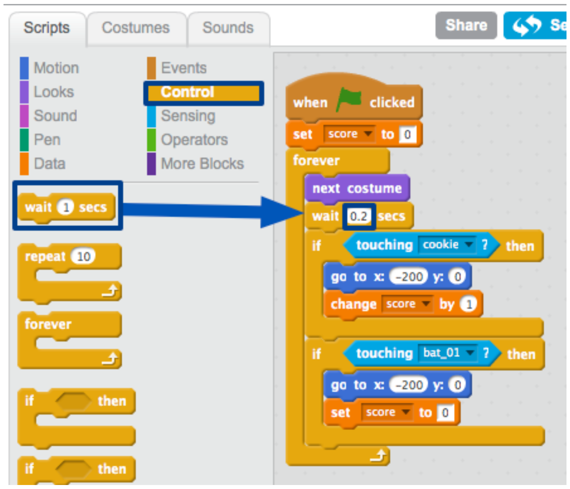
To do this, first make sure that the *chicken_01* sprite is selected in the Sprites area. If selected, the *chicken_01* sprite should have a blue border.

Next, select the *Looks* section in the Scripts tab, and drag a *"next costume"* block above the *"if touching cookie ? then"* block, so that it is wrapped in the *"forever"* block. This is illustrated in the image below:



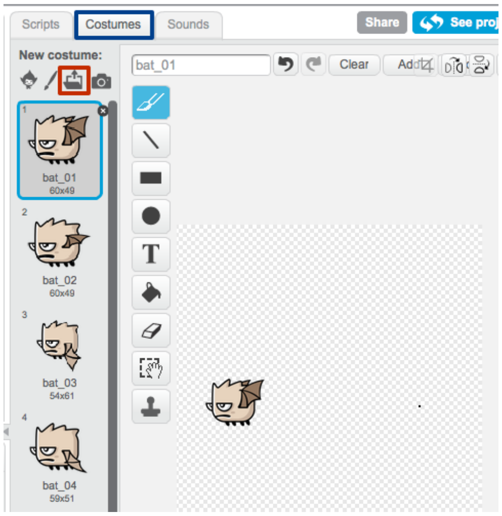
Run the game (by clicking the Green Flag). The chicken should animate and flap its wings. However, it's flapping its wings a bit fast, don't you think? We need to add a script which will add a delay to the switching of the costume, so that the costume doesn't change so quickly.

We will add a *"wait"* block, which will add the delay to the costume switching. To do this, select the *Control* section in the Scripts tab. Then, drag the *"wait 1 secs"* block and place it under the *"next costume"* block. Waiting 1 second to change costume seems too long though, so click in the *"wait 1 secs"* block and change the *"1"* to *"0.2"*. These steps are shown below:



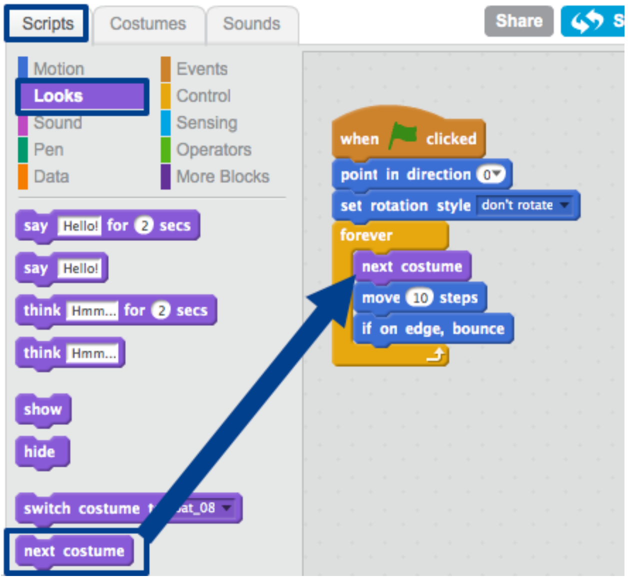
When the game is running the chicken will now flap its wings at a better pace – not too fast and not too slow.

Now that the chicken is animating, we will add a flying animation to the bat. The steps to this are similar, but the bat sprite will have 8 different costumes instead of 4. We will add the costumes to the *bat_01* sprite in a similar way to how we added the chicken costumes. First, make sure that the *bat_01* sprite is selected in the Sprites area. If selected, the *bat_01* sprite will have a blue border around it. Next, click the **Costumes** tab. Click the *"upload costume from file"* button, highlighted in red in the below image, and upload the other bat costumes (**bat_02.png** – **bat_08.png**). These can be uploaded one-by-one or all at once. You may have to re-order the costumes by dragging and dropping them so that they are in the correct order.



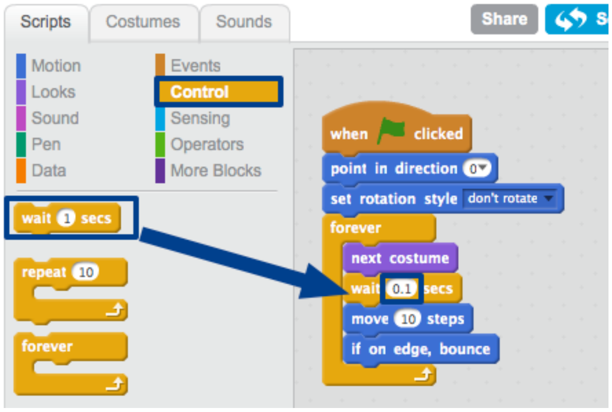
Note that the above image only shows 4 of the *bat_01* sprite's costumes, but there should be 8 (the extra 4 have been cut off in the above image).

Now we will add the flying animation to the *bat_01* sprite. To do this, switch back to the Scripts editor by clicking the **Scripts** tab. Next, select the *Looks* section in the Scripts tab, and drag a *"next costume"* block above the *"move 10 steps"* block, so that it is wrapped in the *"forever"* block. These steps are shown in the image below:



If you run the game you will see the bat flap its wings – but like the chicken we will have to add a delay so the bat doesn't flap its wings so fast. To do this we will add a *"wait"* block.

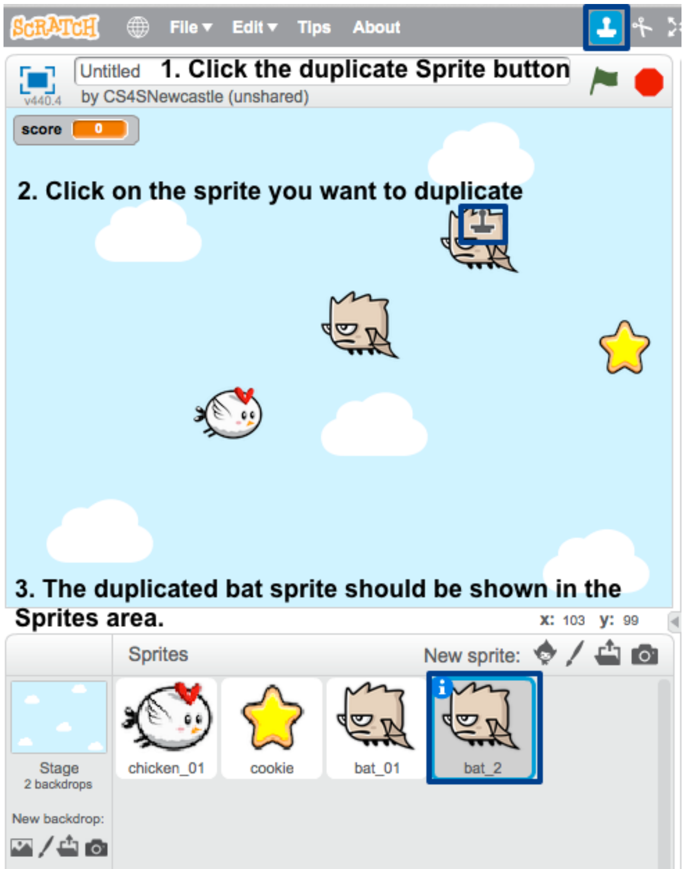
This time we'll add a shorter wait - 0.1 seconds instead of 0.2. To add the *"wait"* block, select the *Control* section in the Scripts tab. Next, drag a *"wait 1 secs"* block so that it sits under the "*next costume"* block in the *bat_01* sprite's Scripts editor, wrapped in the *"forever"* block. Change the *"1"* in the *"wait 1 secs"* block to *"0.1"* instead so the animation is not too slow. The process of adding the *"wait"* block for the bat sprite is illustrated below.



Now, run the game and the bat should be flapping its wings as it moves up and down the screen.

6. Adding More Enemies

We can make the game more challenging by adding more enemies. Scratch makes it very easy to duplicate sprites. When you duplicate a sprite, you will also duplicate all of the sprite's scripts. To add another bat enemy for the chicken to avoid, use the *"duplicate sprite"* button. The duplicate sprite button is highlighted in blue in the top right corner of the image below. When you click this button your cursor will turn into a stamp. When it does this, click on the *bat_01* sprite and it will create a new sprite called *bat_2.* This is illustrated in the image below. As you can see in the image below, duplicating the sprite will add a new sprite (with the blue border) in the Sprites area.



1. Click the duplicate Sprite button

2. Click on the sprite you want to duplicate

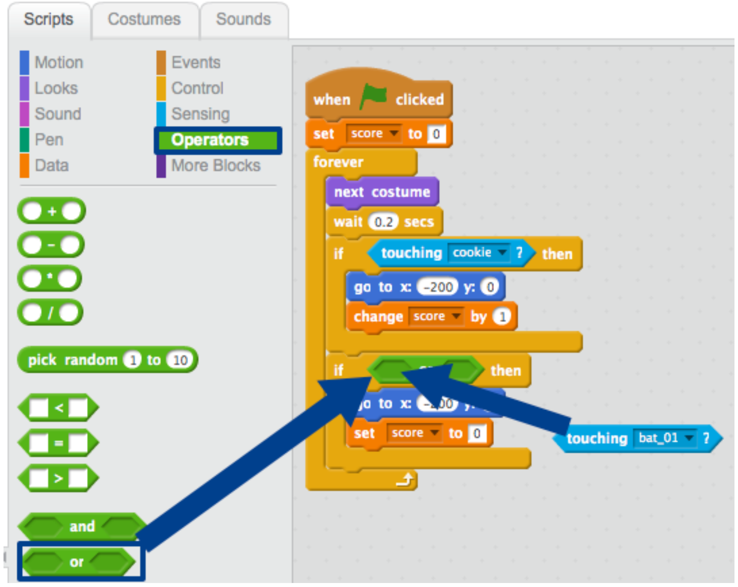
3. The duplicated bat sprite should be shown in the Sprites area.

The duplicated bat sprite will appear at the same position as the original bat, so you will want to move it closer to the left hand side of the screen, and at a different height to the original bat.

There's one more step we need to do to complete the tutorial. If you run the game, and run the chicken into the newly created bat – you may notice that nothing happens. However, the chicken should move back to the start when it runs into the newly created bat and the score should reset. This is because there's no *collision logic* set up for the chicken and new bat. To solve this, we will have to add a block to the chicken's scripts.

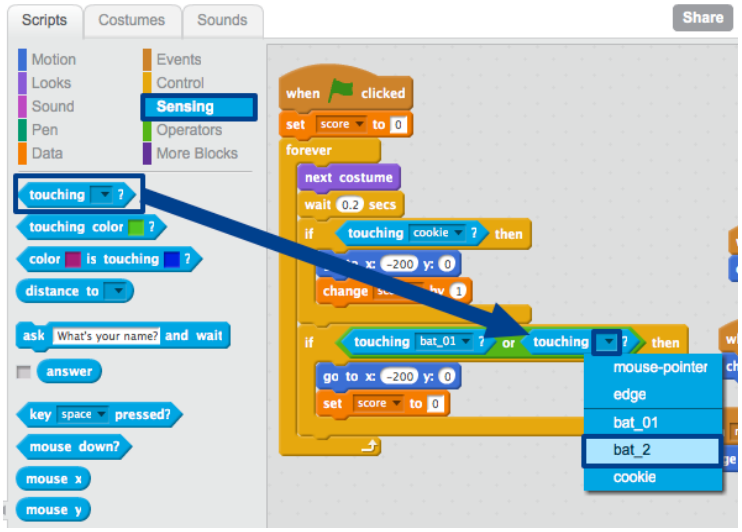
Make sure that the chicken sprite is selected in the Sprites area. If it is selected, there will be a blue border around it in the Sprites area.

Next, select the *Operators* section in the Scripts tab. Drag the *"or"* block to where the blue *"touching bat_01 ?"* is. The *"touching bat_01 ?"* block should pop out of place and move away from the other blocks. The *"or"* block will replace the *"touching bat_01 ?"* in the *"if then"* block. The *"or"* block has 2 slots. Drag the *"touching bat_01 ?"* block into the first slot of the *"or"* block. This process is illustrated in the image below:



```
when [flag] clicked
set score to 0
forever
    next costume
    wait 0.2 secs
    if touching cookie ? then
        go to x: -200 y: 0
        change score by 1
    if < > or < > then
        go to x: -200 y: 0
        set score to 0
```

Next, select the Sensing section of the Scripts tab and drag a *"touching ?"* block into the second slot of the *"or"* block. Select the *"bat_2"* option from the dropdown. This means that the scripts wrapped in the *"if touching bat_01 ? or touching bat_2 ? then"* block will run when the chicken collides with either of the bats. These steps are shown in the image below:



```
when [flag] clicked
set score to 0
forever
    next costume
    wait 0.2 secs
    if touching cookie ? then
        go to x: -200 y: 0
        change score by 1
    if touching bat_01 ? or touching ? then
        go to x: -200 y: 0
        set score to 0
```

Now, see what happens when you run the chicken into the duplicated bat – your score should be reset and the chicken should move back to its original position.

Congratulations, you have finished the *Making Cookie Bird* tutorial. Good Work! The next section contains some suggestions for extra features you can add to the game if you still have time left over at the end of the workshop session.

7. Extras

If you have completed the rest of the tutorial, here's a few extra tasks you can complete to improve the *Cookie Bird* game. If you need any help getting started on these please ask one of the workshop assistants.

7.1 Playing Sounds

There are two sound files included in the activity's resources. These are **"chicken_lose.mp3"** and **"chicken_score.mp3"**. Can you make the **"chicken_lose.mp3"** file play when the chicken runs into a bat? Can you also make the **"chicken_score.mp3"** file play when the chicken flies into the cookie? The blocks for playing Sound are available in the *Sound* section of the Scripts tab.

7.2 Adding Lives

At the moment the player has their score reset after running into the bat once. How could you make it so that you have a number of "lives" that you can lose before the player's score is reset? For example, the player may start off with 3 lives, and lose 1 each time they run into a bat. Once they have 0 lives and run into a bat their score will be reset.

7.3 Adding a +1 Animation to the Cookie

There is an image in the activity's resources titled **"plus_one.png".** How could you make it that the cookie sprite temporarily changes to the **"plus_one.png"** graphic, to indicate when the player has increased their score? This can be done in a couple of ways, for example using a *"broadcast message"* block (in the *Events* section) from the *chicken_01* sprite, or by checking for collisions on the *cookie* sprite.