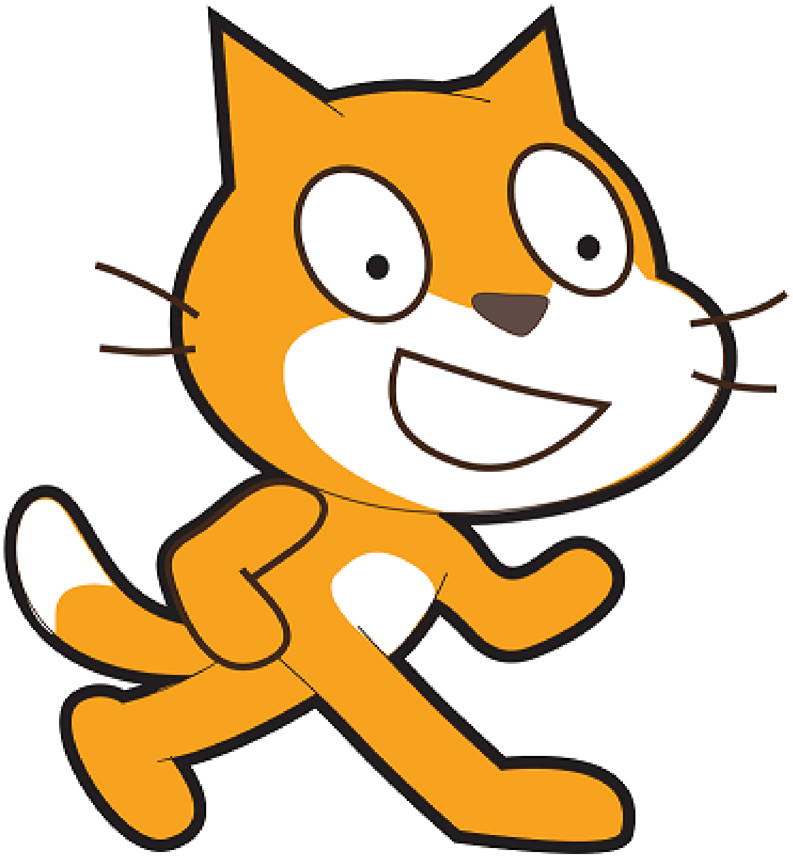
Scratch User Manual

version 1.0



Completed in partial requirements for:
TWR2004 Technical Writing II
By: **Theresa (Terri) Webster**
April 12, 2016

This manual is dedicated to my muse, a nine year old Scratch programmer and Maker of tomorrow.

Scratch programming environment information:

Version: Scratch v2.0

Scratch is a programming language and online community where you can create your own interactive stories, games, and animations—and share your creations with others around the world. In the process of designing and programming Scratch projects, young people learn to think creatively, reason systematically, and work collaboratively.

Scratch is a project of the Lifelong Kindergarten group at the MIT Media Lab. It is available for free at http://scratch.mit.edu

Table of Contents

About This Manual

The Scratch User Manual is an introductory guide to help children learn how to program with Scratch, a free online visual programming language. This manual is for children aged 8 to 16 with basic computer skills, who may or may not be familiar with programming. It is also for the parents and teachers who support these children in learning how to program. While parents and teachers need basic computer skills to use this manual, they do not need to be familiar with programming or Scratch.

Programming is an important life skill for the 21st century. Computers are a big part of our daily life. Instead of just using computers, we can create things and solve problems with them. Programming strengthens problem-solving and thinking skills. These skills are helpful in many areas of life.

Children (and adults) could experience a range of emotions around the idea of creating software programs. They could feel excited and impatient or frustrated and upset. For effective learning, frustration needs to be reduced. The Scratch User Manual reduces frustration by introducing programming in a visual way.

Getting started with Scratch

Scratch is a visual programming language. To create software programs in Scratch, you connect coloured blocks of code together (like puzzle pieces) to make **scripts** that tell the computer what to do. This makes Scratch programming quicker to learn than older programming languages because you don’t need to type long sets of instructions to tell a computer what to do.

In Scratch, you use **scripts** to make **sprites** (i.e. characters and objects) do things like move and speak. While you’re having fun creating and exploring with Scratch, you’ll be learning key programming ideas. By understanding these key ideas, you’ll have what you need to learn other programming languages too.

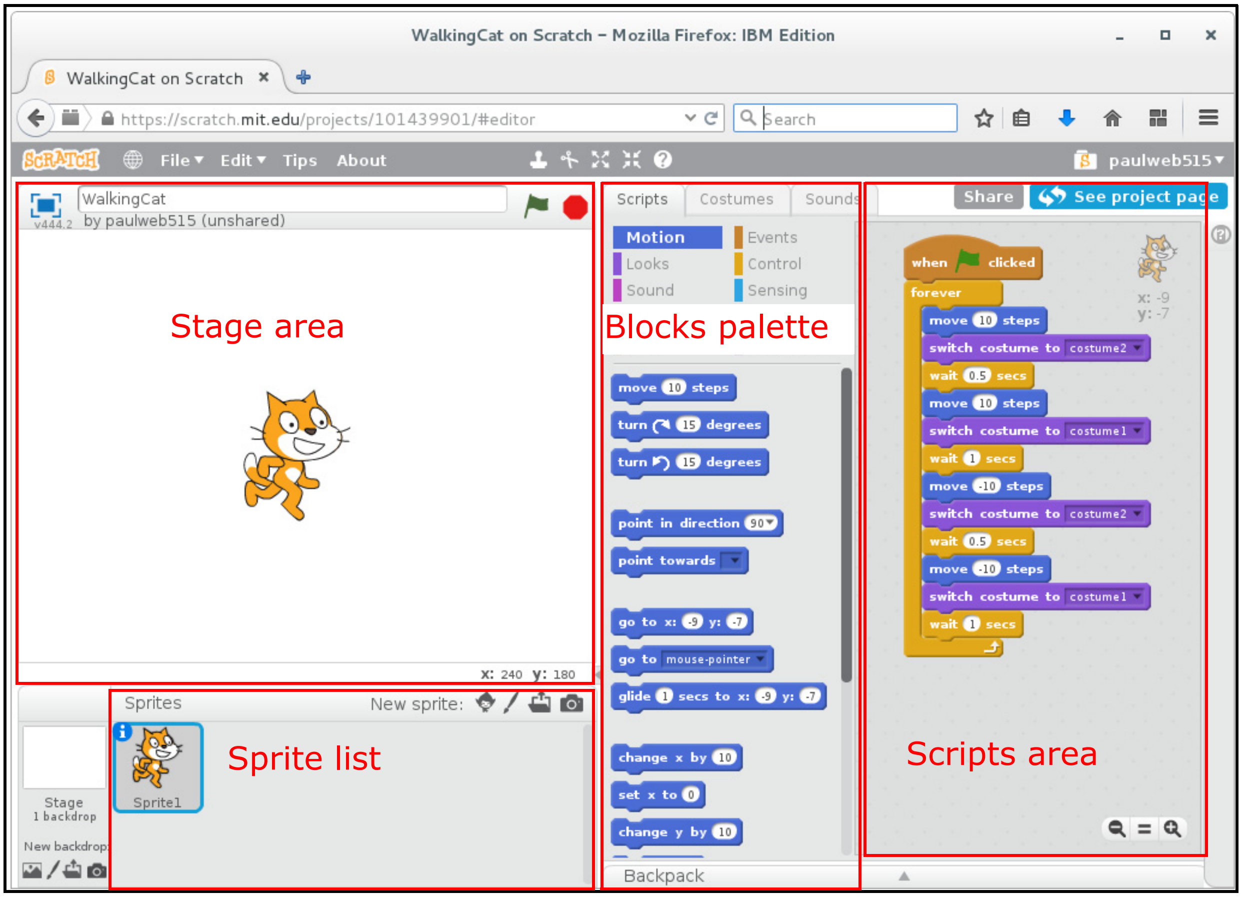
To run Scratch on your computer, you will need the following equipment:

- Computer (laptop with i3 or faster processor, or a desktop computer with at least x86)
- Operating system (Windows 7 or higher, or a version of Mac OSX)
- Browser (recent version of Safari, Internet Explorer, or Firefox with the Adobe Flash plug-in installed)

To access Scratch, go to https://scratch.mit.edu. You will need to create an account as described on the website in order to register and log in.

Looking at the Scratch user interface

The Scratch user interface is shown in **Figure 1**. The different areas of the user interface that you will be working with are labeled to help you get to know their names and locations.



```
when [flag] clicked
forever
    move 10 steps
    switch costume to costume2
    wait 0.5 secs
    move 10 steps
    switch costume to costume1
    wait 1 secs
    move -10 steps
    switch costume to costume2
    wait 0.5 secs
    move -10 steps
    switch costume to costume1
    wait 1 secs
```

Figure 1: Scratch User Interface

Finding information in the Scratch User Manual

The Scratch User Manual is organized into five main sections as summarized in **Table 1** on page 3. If you are new to programming, you'll want to read through the manual and complete the activities as you go. If you have some experience with programming or Scratch, you can start with any chapter that interests you.

At any time, if you‘re working on an activity that builds upon a previous procedure in the manual, you’ll find a link back to the relevant procedure with the information that you need to continue with your chosen activity.

Table 1: How the Scratch User Manual is Organized

To learn about	Turn to
Creating, renaming, editing, and changing the appearance of sprites	**Designing Sprites** on page 4
Making sprites move and repeat moves	**Making Moves** on page 12
Making sprites respond to mouse and keyboard input	**Making Things Happen** on page 20
Making sprites make sounds	**Making Sounds** on page 27
Definitions for frequently used technical terms	**Appendix A: Glossary** on page 33
Answers to frequently asked questions about Scratch programming	**Appendix B: FAQ** on page 34
Identifying the cause of and finding solutions for common Scratch programming problems	**Appendix C: Troubleshooting Guide** on page 36
Cursor and Graphics toolbar button descriptions	**Appendix D: Toolbar Reference** on page 38

Using the symbols to learn more

The symbols shown in **Table 2** appear in this manual to alert you to useful information in the form of notes and tips.

Table 2: Legend of Symbols

Symbol	Meaning
🗒	A note provides additional information about a procedure step.
💡	A tip provides an alternative method for performing a procedure step or a hint to learn more about a procedure step.

Designing Sprites

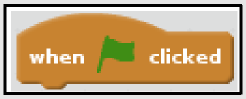
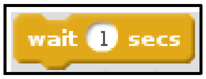
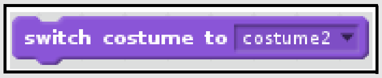
Scratch is a visual programming language that lets you create all kinds of programs. Each program is made up of **sprites** and **scripts**. **Sprites** are the characters and objects on the Scratch stage, such as animals and trees. **Scripts** are connected coloured blocks of code that you put together like puzzle pieces. You use the **scripts** to make your sprites do things, like run and dance.

This section shows you how to design sprites with Scratch. You will learn about:

- **Creating sprites**
- **Renaming sprites**
- **Editing sprites**
- **Changing the appearance of sprites**

The code blocks that you will be using are summarized in **Table 3**.

Table 3: Code Blocks Used in Designing Sprites

Code Block	Name	Type
	when green flag clicked	Events
	wait 1 secs	Control
	switch costume to costume2	Looks

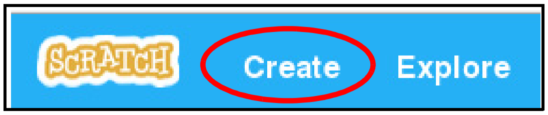
Creating sprites

Sprites are the actors on the Scratch stage. **Sprites** can be characters like people and animals, or objects like cars and trees. Scratch comes with a selection of suggested **sprites** called the **Sprite Library.**

You can have several sprites on stage together, each with their own scripts. When you create a new Scratch project, the cat sprite is already on the stage. You can add a sprite to the stage by creating a sprite.

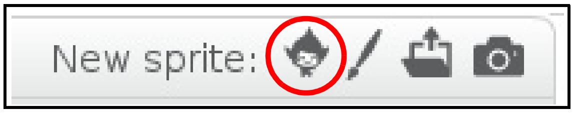
To create a sprite

1. Click **Create** to create a new Scratch project.



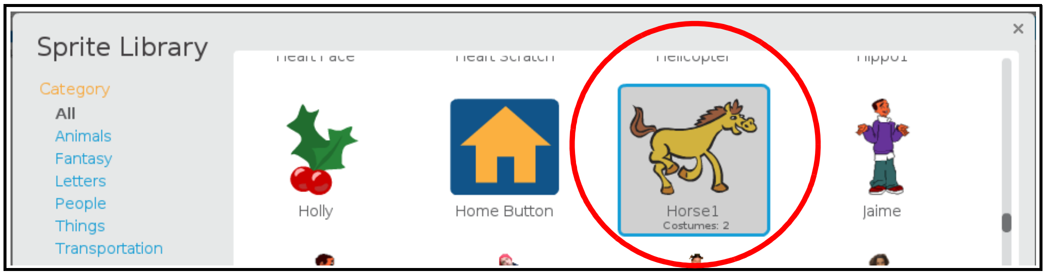
The Scratch interface opens a new project with one cat sprite in it.

2. From the **New sprite** toolbar, click **Choose sprite from library**.

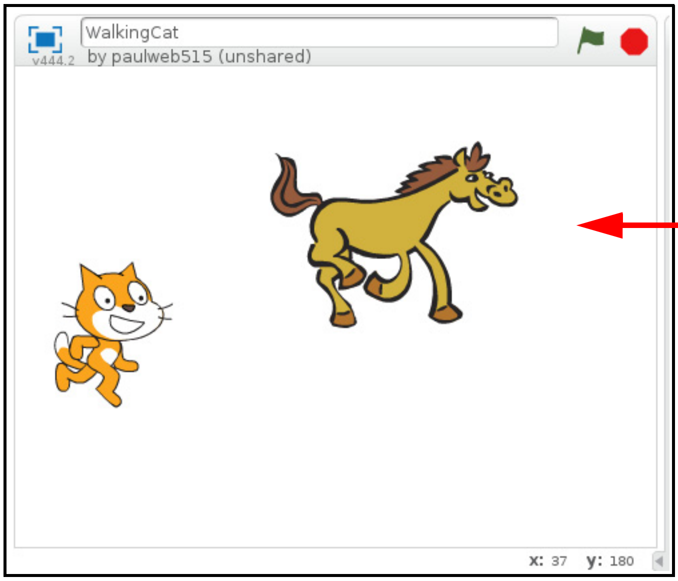


*The **Sprite Library** dialog appears.*

3. Select a sprite from the library and click **OK**.



Your selected sprite appears on the stage.



4. Click **File > Save Now** to save your project.

Your project is saved.

Now that you know how to create a sprite, you probably want to name it. See **Renaming sprites** to learn how to name your sprite.

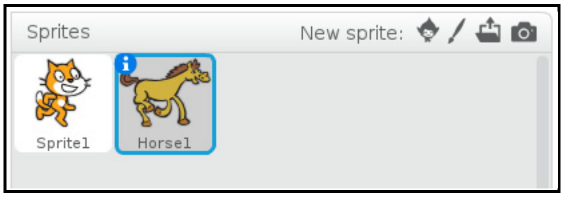
Renaming sprites

Sprites in the **Sprite Library** come with basic names. You can be creative and give your sprites more exciting names. This will make it easier to tell your sprites apart if you have more than one of the same kind of sprite (e.g. four cats) on the stage. You can change the name of a sprite in an existing project by renaming the sprite.

To rename a sprite

1. Open an existing Scratch project with at least one sprite in it.
2. From the **Sprite list**, select the sprite that you want to rename.

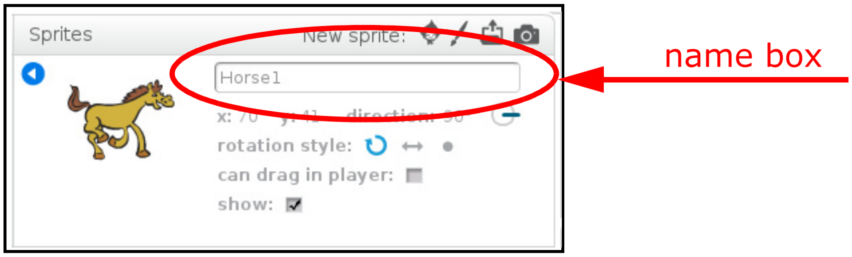
   A blue box appears around the selected sprite with a blue 'i' button in the upper left corner.



3. Click the blue 'i' button.

   *The **Sprite Information** panel opens.*

4. Type a new name for your sprite in the name box.



Horse1

x: 70 y: 41 direction: 90

rotation style:

can drag in player: □

show: ☑

name box

5. To close the **Sprite Information** panel, click the **blue arrow** ◀.

   The Sprite Information panel closes and the new sprite name appears below the selected sprite.

6. Click **File > Save Now** to save your project.

   Your project is saved.

Now that you know how to rename a sprite, you can come up with creative names for sprites in all of your Scratch projects. Another way to make your sprites unique is to make changes to their **costumes**. See **Editing sprites** to learn how to edit your sprite's costume.

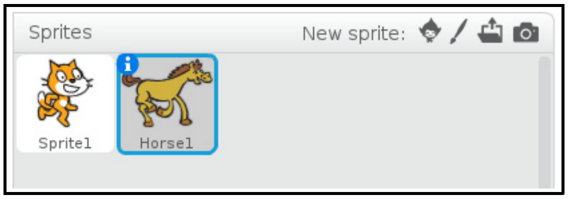
Editing sprites

To make your sprites unique, you can edit them by making changes to their costumes. You can be creative and give your sprites wings (or hats or skateboards). You can edit a sprite in an existing project by duplicating and making changes to its costume.

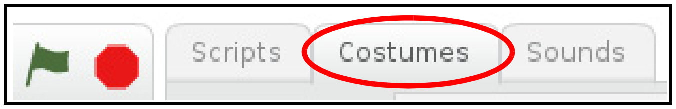
To edit a sprite

1. Open an existing Scratch project with at least one sprite in it.
2. From the **Sprite list**, select the sprite that you want to edit.

   A blue box appears around the selected sprite.

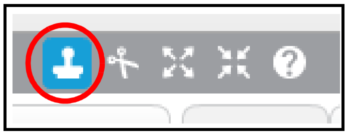


3. To see the available costumes for the selected sprite, click the **Costumes** tab in the **Blocks palette**.



   *The **Costume list** appears.*

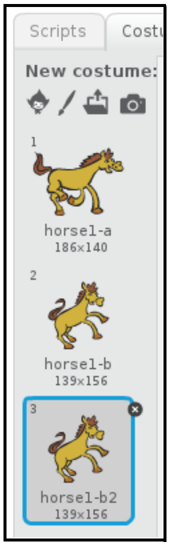
4. From the **Cursor** toolbar, click the **Duplicate** icon.



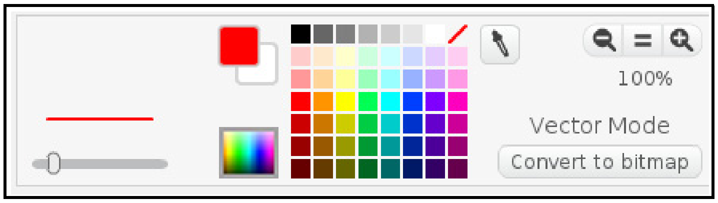
   *The mouse pointer changes to the **Duplicate** icon.*

5. Click on the costume you want to duplicate in the **Costume list**.

   *A copy of the selected costume appears in the **Costume list** and the mouse pointer returns to its original shape.*

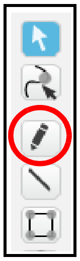


6. Choose a colour from the **color picker**.



The foreground box changes to the colour that you have chosen.

7. From the **Graphics** toolbar, click the **pencil** icon to use the pencil tool.



*The **pencil** icon is highlighted.*

8. Hold down the mouse button and draw wings on the costume in the **Costume editing** area.

   *Wings appear on your sprite (on the stage) in the colour that you chose from the **color picker**.*

9. Click **File > Save Now** to save your project.

   Your project is saved.

Now that you know how to edit a sprite by duplicating and making changes to its costume, you can come up with creative costumes for sprites in all of your Scratch projects. Next, you'll learn how to change the appearance of your sprite on the stage by creating a script to switch between your sprite's costumes. See **Changing the appearance of sprites** to learn how to change the appearance of your sprite.

Changing the appearance of sprites

It's fun to make sprites do things in Scratch. One of the ways you can make it look like your sprite is doing something is to change the sprite's appearance by switching between the sprite's costumes (e.g. a rocket sprite could have one costume where its landing gear is in place and a second costume with fire shooting out the back). You can change the appearance of a sprite in an existing project by creating a script to control the sprite's costume changes.

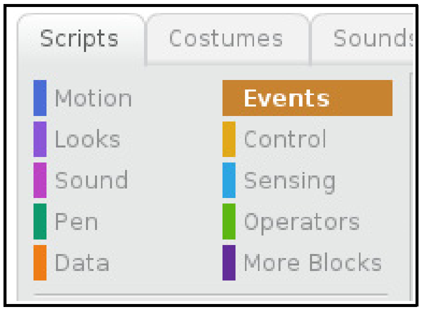
To change the appearance of a sprite

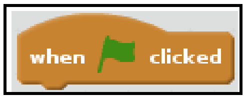
1. Create a new Scratch project with at least one sprite in it.
2. From the **Sprite list**, select the sprite whose costume you want to change.

   A blue box appears around the selected sprite.

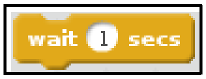
3. On the **Scripts** tab of the **Blocks palette**, click **Events**.

   *The **Events** code blocks appear.*



4. Drag a **when green flag clicked** block  to the **Scripts** area.

   *The **when green flag clicked** block appears in the **Scripts** area.*

5. On the **Blocks palette**, click **Control** and drag a **wait 1 secs** block  wait 1 secs to the **Scripts** area to connect to the bottom of the **when green flag clicked** block.

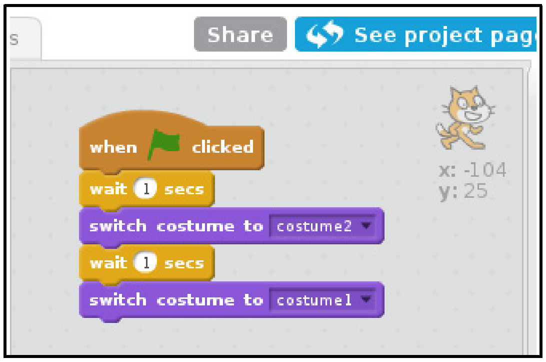
   *The **Control** code blocks appear and the **wait 1 secs** block is connected to the **when green flag clicked** block in the **Scripts** area.*

6. On the **Blocks palette**, click **Looks** and drag a **switch costume to costume2** block switch costume to costume2 to the **Scripts** area to connect to the bottom of the **wait 1 secs** block.

   *The **Looks** code blocks appear and the **switch costume to costume2** block is connected to the **wait 1 secs** block in the **Scripts** area.*

7. Add another **wait 1 secs** block and **switch costume to costume2** block.
8. On the **switch costume to costume2** block that you just added, choose **costume 1** from the drop-down combo box.

   *The last block is now a **switch costume to costume1** block. Your **Scripts** area should look similar to the one in the image below.*



```
when [flag] clicked
wait 1 secs
switch costume to costume2
wait 1 secs
switch costume to costume1
```

9. In the **Stage area**, click the **green flag** icon 🏁 to run your script.

   After a one second pause, your sprite's appearance changes by switching to its second costume. After another one second pause, your sprite's appearance changes again by switching back to its original costume.

10. Click **File > Save Now** to save your project.

    Your project is saved.

Now that you know how to change the appearance of a sprite by creating a script to control the sprite's costume changes, you can make the sprites in all of your Scratch projects look like they're doing things (e.g. a flying unicorn might have one costume with wings pointing up and a second costume with wings pointing down). So far your sprites are still in the same place on the stage. You can make your sprite's actions more realistic by moving your sprite around the stage. See **Making Moves** to learn how to move your sprite around the stage.

Making Moves

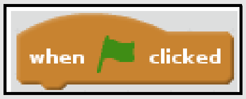
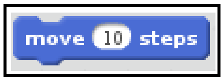
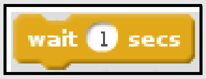
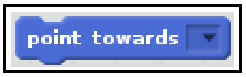
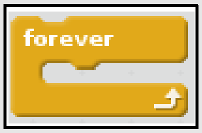
In the previous section, you learned how to design a sprite. By changing the appearance of a sprite, you were able to make it look like the sprite was doing something even though the sprite was still in the same place on the stage. To make a sprite's actions more realistic, you can move a sprite around the stage by creating scripts using **Motion** and **Control** code blocks.

This section shows you how to make a sprite move around the stage. You will learn about:

- **Moving sprites to the right**
- **Making sprites follow the mouse pointer**
- **Repeating sprite moves**

The code blocks that you will be using are summarized in **Table 4**.

Table 4: Code Blocks Used in Making Moves

Code Block	Name	Type
	when green flag clicked	Events
	move 10 steps	Motion
	wait 1 secs	Control
	point towards	Motion
	forever	Control

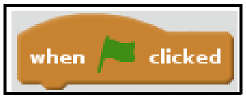
Moving sprites to the right

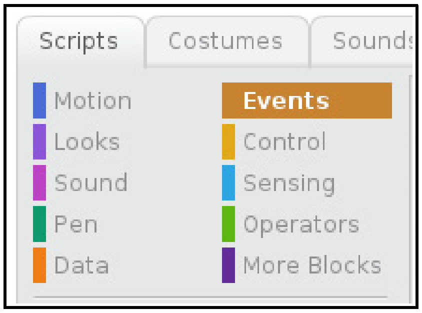
One of the ways to move a sprite around the stage is to take steps to the right (or left). You can make a sprite take steps to the right (or left) in an existing project by creating a script to control the sprite's steps on the stage.

To move a sprite to the right

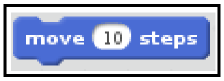
1. Create a new Scratch project with at least one sprite in it.
2. From the **Sprite list**, select the sprite that you want to move.

   A blue box appears around the selected sprite.

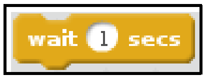
3. On the **Scripts tab** of the **Blocks palette**, click **Events** and drag a **when green flag clicked** block  to the **Scripts** area.

   *The **Events** code blocks appear and the **when green flag clicked** block appears in the **Scripts** area.*

4. On the **Blocks palette**, click **Motion** and drag a **move 10 steps** block  to the **Scripts** area to connect to the bottom of the **when green flag clicked** block.

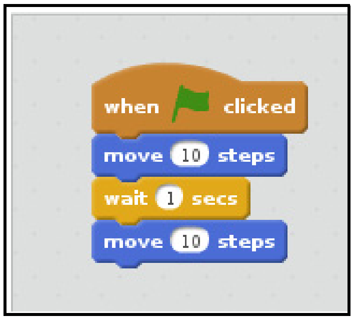
   *The **Motion** code blocks appear and the **move 10 steps** block is connected to the **when green flag clicked** block in the **Scripts** area.*

5. On the **Blocks palette**, click **Control** and drag a **wait 1 secs** block  to the **Scripts** area and connect it to the bottom of the **move 10 steps** block.

   *The **Control** code blocks appear and the **wait 1 secs** block is connected to the **move 10 steps** block in the **Scripts** area.*

6. On the **Blocks palette**, click **Motion** and drag a **move 10 steps** block to the **Scripts** area to connect to the bottom of the **wait 1 secs** block.

   *The **Motion** code blocks appear and the **move 10 steps** block is connected to the **wait 1 secs** block in the **Scripts** area.*



```
when ⚑ clicked
move 10 steps
wait 1 secs
move 10 steps
```

7. In the **Stage area**, click the **green flag** icon to run your script.

   Your sprite moves 10 steps to the right, pauses for a second, and moves another 10 steps to the right.

   If you type -10 in the top **move 10 steps** block of your script and run your script again, you will see that your sprite first moves 10 steps to the left, pauses for a second, and then moves 10 steps back to its original position on the right. What do you think your sprite would do if you typed -10 in the bottom **move 10 steps** block as well? Try typing different numbers in the **move 10 steps** blocks and rerunning your script to see what happens.

8. Click **File > Save Now** to save your project.

   Your project is saved.

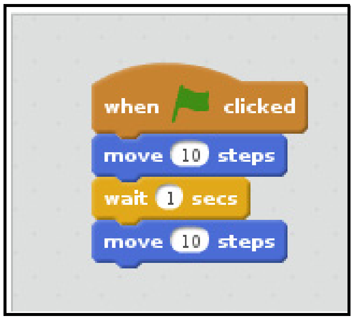
Now that you know how to make your sprite move by stepping to the right (or left) on the stage, you can make the sprites in all of your Scratch projects step to the right (or left). See **Making sprites follow the mouse pointer** to learn how to make a sprite follow the mouse pointer around the stage.

Making sprites follow the mouse pointer

Now that you know how to make your sprite move by taking steps to the right (or left), you're probably curious about how to make a sprite move around the rest of the stage. A fun way to do this is to build a script to make a sprite follow the mouse pointer around the stage.

To make sprites follow the mouse pointer

1. Open an existing Scratch project with a completed script for moving your sprite by using **move 10 steps** blocks as shown in the following image.

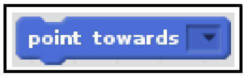


```
when [flag] clicked
move 10 steps
wait 1 secs
move 10 steps
```

To build a script like the one in the image above, see **Moving sprites to the right**.

2. From the **Sprite list**, select the sprite that you want to move.

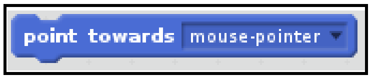
   A blue box appears around the selected sprite.

3. On the **Scripts** tab of the **Blocks palette**, click **Motion** and drag a **point towards** block  point towards to the **Scripts** area to connect to the bottom of the second **move 10 steps** block.

   *The **point towards** block is connected to the second **move 10 steps** block in the **Scripts** area.*

4. On the **point towards** block that you just added, choose **mouse-pointer** from the drop-down combo box.

   *The last block is now a **point towards mouse-pointer** block*

   point towards mouse-pointer

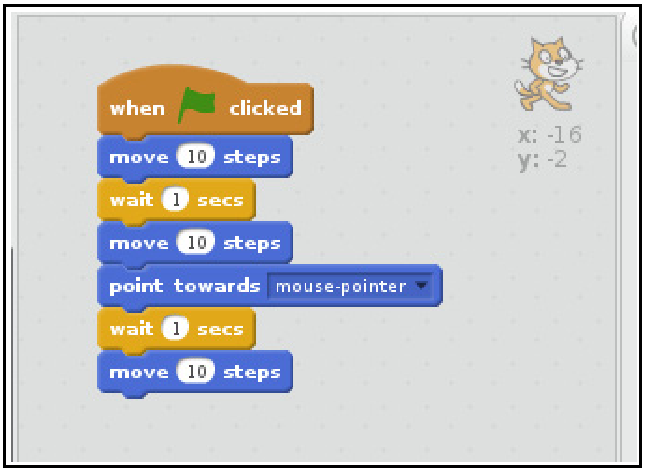
   .

5. On the **Blocks palette**, click **Control** and drag a **wait 1 secs** block to the **Scripts** area to connect to the bottom of the **point towards mouse-pointer** block.

   *The **wait 1 secs** block is connected to the **point towards mouse-pointer** block in the **Scripts** area.*

6. On the **Blocks palette**, click **Motion** and drag a **move 10 steps** block to the **Scripts** area to connect to the bottom of the **wait 1 secs** block.

   *The **move 10 steps** block is connected to the **wait 1 secs** block in the **Scripts** area.*



```
when ⚑ clicked
move 10 steps
wait 1 secs
move 10 steps
point towards mouse-pointer
wait 1 secs
move 10 steps
```

7. In the **Stage area**, click the **green flag** icon ⚑ to run your script.
8. While your script is still running, quickly place your mouse pointer on the stage below your sprite.

   Your sprite moves 10 steps to the right, pauses for a second, and then moves another 10 steps to the right while simultaneously turning toward your pointer, pauses for a second, and moves 10 steps toward your pointer.

   If you run the script again and place the mouse pointer on a different part of the stage, you will see that your sprite moves 20 steps, turns toward your pointer, and then moves another 10 steps toward your pointer. Can you make your sprite move in a circle around the stage by running the script more than once?

9. Click **File > Save Now** to save your project.

   Your project is saved.

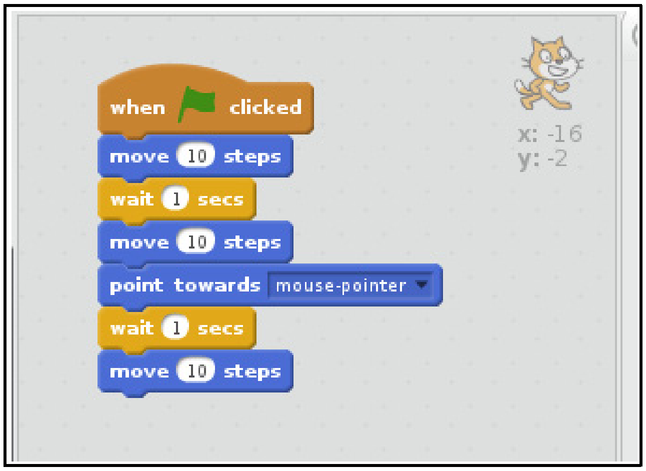
Now that you know how to make your sprite follow the mouse pointer, you can make the sprites in all of your Scratch projects follow the mouse pointer. You're probably wondering if there's a way to move your sprite around the stage without having to run your script over and over again. See **Repeating sprite moves** to learn how to make a sprite repeat moves.

Repeating sprite moves

Now that you know how to make your sprite follow the mouse pointer, you're ready to learn how to make a sprite move around the stage without having to run the sprite's script over and over again. You can build a script to make a sprite repeat a move or series of moves over and over again by using a **Control** block.

To repeat a sprite move

1. Open an existing Scratch project with a completed script for making your sprite follow the mouse pointer as shown in the image below.

   ```
   when [green flag] clicked
   move 10 steps
   wait 1 secs
   move 10 steps
   point towards mouse-pointer
   wait 1 secs
   move 10 steps
   ```

   To build a script like the one in the image above, see **Making sprites follow the mouse pointer** on page 14.

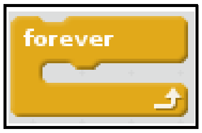
2. From the **Sprite list**, select the sprite that you want to move.

   A blue box appears around the selected sprite.

3. With the **Scripts** tab selected, in the **Scripts** area drag the top **move 10 steps** block away from the when **green flag clicked** block.

   *The **when green flag clicked** block is no longer connected to any code blocks and the **move 10 steps** block remains connected to the blocks below it in a stack.*

4. On the **Blocks palette**, click **Control** and drag a **forever** block to the **Scripts** area to connect to the **when green flag clicked** block.

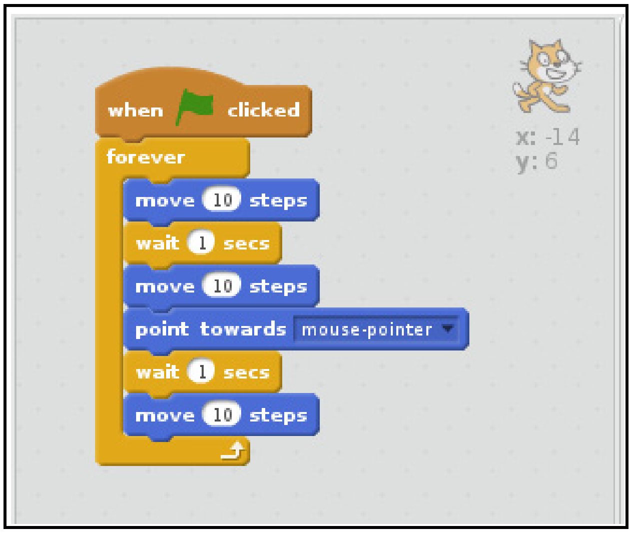
   

   forever

   *The **forever** block is connected to the **when green flag clicked** block in the **Scripts** area.*

5. Using the block shapes as your guide like puzzle pieces, drag the top **move 10 steps** block to connect the stack to the middle of the **forever** block.

   *The **move 10 steps** block stack is wrapped in the **forever** block in the **Scripts** area.*



```
when ⚑ clicked
forever
    move 10 steps
    wait 1 secs
    move 10 steps
    point towards mouse-pointer ▾
    wait 1 secs
    move 10 steps
```

6. In the **Stage area**, click the **green flag** icon ⚑ to run your script and place your mouse pointer on the stage.

   Your sprite moves a 10 steps to the right, pauses, moves another 10 steps to the right while simultaneously turning toward your pointer, pauses, and then moves another 10 steps toward your pointer.

7. Continue leading your sprite around the stage by placing your mouse pointer somewhere else on the stage every time your sprite moves toward it.

   Every time you move your mouse pointer to a new location, your sprite moves 10 steps, pauses, moves another 10 steps in the same direction while simultaneously turning toward your pointer's new location, pauses, and then moves 10 more steps toward your pointer.

In order to make the sprite follow the pointer in a new direction on the stage, you need to move the pointer as soon as the sprite moves toward it. You can keep this up as long as you like to make the sprite follow a path of your choosing on the stage (e.g. circle, figure eight, or back and forth). If you keep the mouse pointer in the same place long enough, the sprite will stop moving when it reaches the mouse pointer.

8. In the **Stage area**, click the **red stop sign** icon 🛑 to stop your script.

   The script stops running and your sprite stops moving.

9. Click **File > Save Now** to save your project.

   Your project is saved.

Now that you know how to make your sprite repeat moves over and over again, you can make the sprites in all of your Scratch projects repeat moves. While it is exciting to make your sprites repeat moves, you might be wondering if there's a way to make a sprite do different kinds of moves when you click with the mouse or type on the keyboard. See **Making Things Happen** to learn how to interact with your sprite by using the mouse and keyboard (i.e. to provide **mouse input** and **keyboard input**).

Making Things Happen

In the previous section, you learned how to make a sprite move around the stage by taking steps, following the mouse pointer, and repeating moves. To interact with your sprite, you can make a sprite do something in response to **mouse** or **keyboard input** by creating scripts using **Control**, **Sensing**, and **Motion** code blocks.

This section shows you how to make a sprite do something in response to mouse or keyboard input. You will learn about:

- **Making sprites respond to mouse input**
- **Making sprites respond to keyboard input**

The code blocks that you will be using are summarized in **Table 5**.

Table 5: Code Blocks Used in Making Things Happen

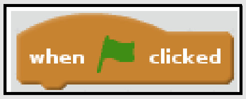
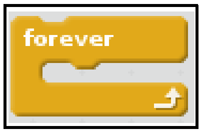
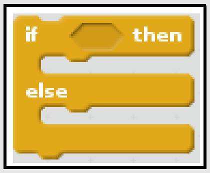
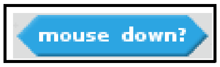
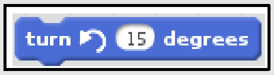
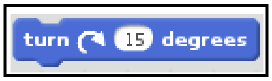
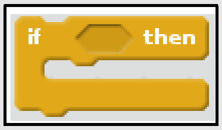
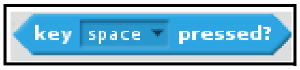
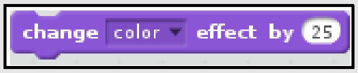
Code Block	Name	Type
	when green flag clicked	Events
	forever	Control
	if then else	Control
	mouse down?	Sensing
	turn counter-clockwise 15 degrees	Motion
	turn clockwise 15 degrees	Motion
	if then	Control

Table 5: Code Blocks Used in Making Things Happen

Code Block	Name	Type
 key space pressed?	key space pressed?	Sensing
 change color effect by 25	change color effect by 25	Looks

Making sprites respond to mouse input

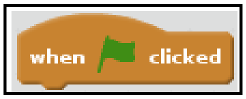
One of the ways to interact with a sprite is to make the sprite do something in response to mouse input. You can make a sprite do something in response to mouse input in an existing project by creating a script with **Control**, **Sensing**, and **Motion** code blocks.

To make a sprite respond to mouse input

1. Create a new Scratch project with at least one sprite in it.
2. From the **Sprite list**, select the sprite that you want to control with mouse input.

   A blue box appears around the selected sprite.

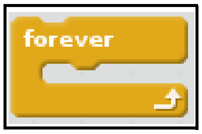
3. On the **Scripts** tab of the **Blocks palette**, click **Events** and drag a **when green flag clicked** block to the **Scripts** area.

   when clicked

   *The **Events** code blocks appear and the **when green flag clicked** block appears in the **Scripts** area.*

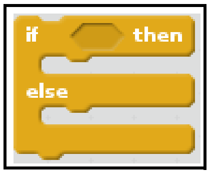
4. On the **Blocks palette**, click **Control** and drag a **forever** block to the **Scripts** area to connect to the **when green flag clicked** block.

   forever

   *The **forever** block is connected to the **when green flag clicked** block in the **Scripts** area.*

5. On the **Blocks palette**, drag an **if then else** block to the **Scripts** area and connect it to the middle of the **forever** block.



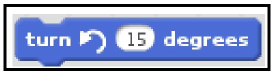
   *The **if then else** block is wrapped in the **forever** block in the **Scripts** area.*

   The **if then else** block tests whether a test statement is true or false. If the test statement is true, then the code blocks wrapped in the top of the **if then else** block run. If the test statement is false, then the code blocks wrapped in the bottom of the **if then else** block run.

6. On the **Blocks palette**, click **Sensing** and drag a **mouse down?** block to the **Scripts** area to place in the window of the **if then else** block.

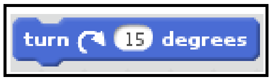
   *The **if then else** block becomes an **if mouse down? then else** block in the **Scripts** area.*

7. On the **Blocks palette**, click **Motion** and drag a **turn counter-clockwise 15 degrees** block to the **Scripts** area to connect in the top opening of the **if mouse down? then else** block.



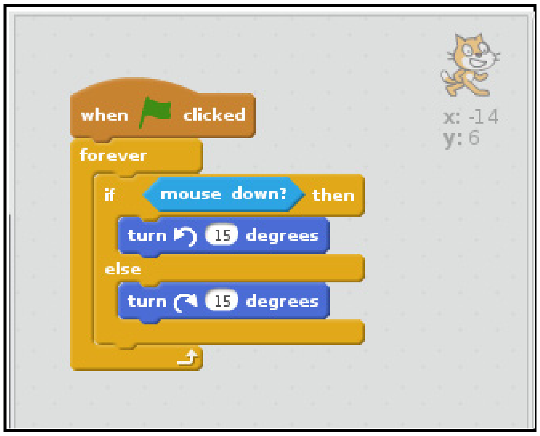
   *The **turn counter-clockwise 15 degrees** block is wrapped in the top of the **if mouse down? then else** block in the **Scripts** area. It becomes an **if mouse down? then turn counter-clockwise 15 degrees else** block.*

8. On the **Blocks palette**, drag a **turn clockwise 15 degrees** block to the **Scripts** area to connect in the bottom opening of the ***if mouse down? then turn counter-clockwise 15 degrees else** block*.



   *The **turn clockwise 15 degrees** block is wrapped in the bottom of the **if mouse down? then turn counter-clockwise 15 degrees else** block in the **Scripts** area. It becomes a complete **if mouse down? then turn counter-clockwise 15 degrees else turn clockwise 15***

***degrees** block. Your **Scripts** area should look similar to the one in the image below.*



```
when ⚑ clicked
forever
    if <mouse down?> then
        turn ↺ (15) degrees
    else
        turn ↻ (15) degrees
```

The test statement for your completed block stack is "The left mouse button is being held down."

9. In the **Stage area**, click **green flag** icon ⚑ to run your script and hold down the left mouse button.

   Your sprite spins clockwise before you hold down the left mouse button and then your sprite spins counter-clockwise when you hold down the left mouse button.

10. In the **Stage area**, click the **red stop sign** icon ● to stop your script and click **File > Save Now** to save your project.

    The script stops running and your project is saved.

Now that you know how to interact with your sprite by making your sprite do something in response to mouse input, you can make the sprites in all of your Scratch projects do something in response to mouse input. See **Making sprites respond to keyboard input** to learn how to interact with a sprite by making the sprite do something in response to keyboard input.

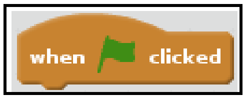
Making sprites respond to keyboard input

Another way to interact with a sprite is to make the sprite do something in response to keyboard input. You can make a sprite do something in response to keyboard input in an existing project by creating a script with **Control**, **Sensing**, and **Motion** code blocks.

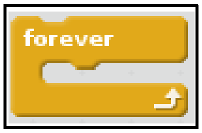
To make a sprite respond to keyboard input

1. Create a new Scratch project with at least one sprite in it.
2. From the **Sprite list**, select the sprite that you want to control with keyboard input.

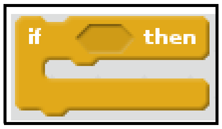
   A blue box appears around the selected sprite.

3. On the **Scripts** tab of the **Blocks palette**, click **Events** and drag a **when green flag clicked** block  to the **Scripts** area.

   *The **Events** code blocks appear and the **when green flag clicked** block appears in the **Scripts** area.*

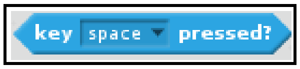
4. On the **Blocks palette**, click **Control** and drag a **forever** block  to the **Scripts** area to connect to the **when green flag clicked** block.

   *The **forever** block is connected to the **when green flag clicked** block in the **Scripts** area.*

5. On the **Blocks palette**, drag an **if then** block  to the **Scripts** area and connect it to the middle of the **forever** block.

   *The **if then** block is wrapped in the **forever** block in the **Scripts** area.*

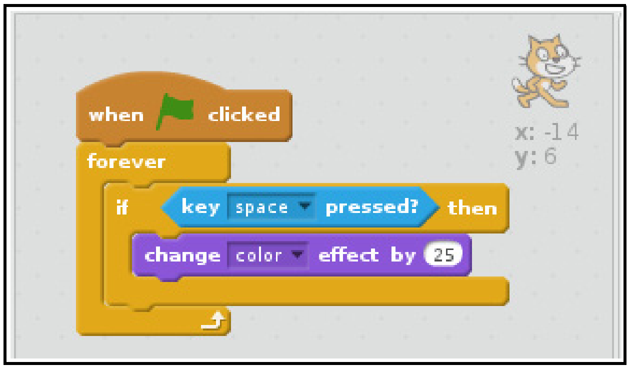
   The **if then** block tests whether a test statement is true or false. If the test statement is true, then the code blocks wrapped in the **if then** block run. If the test statement is false, then the code blocks wrapped in the **if then** block do not run.

6. On the **Blocks palette**, click **Sensing** and drag a **key space pressed?** block  to the **Scripts** area to place in the window of the **if then** block.

   *The **if then** block becomes an **if key space pressed? then** block in the **Scripts** area.*

7. On the **Blocks palette**, click **Looks** and drag a **change color effect by 25** block to the **Scripts** area to connect to the middle of the **if key space pressed? then** block.

   The ***change color effect by 25*** *block is wrapped in the* ***if key space pressed? then*** *block in the* ***Scripts*** *area and it becomes an* ***if key space pressed? then change color effect by 25*** *block. Your* ***Scripts*** *area should look similar to the one in the image below.*



```
when [flag] clicked
forever
    if key space pressed? then
        change color effect by 25
```

The test statement for your completed block stack is "The spacebar is pressed."

8. In the **Stage area**, click the **green flag** icon to run your script and press spacebar.

   Your sprite changes colour.

The sprite colour will change again each time you press the spacebar.

To use the 'a' on your keyboard to control your sprite, choose 'a' from the drop-down combo box of the **key space pressed?** block so that it becomes a **key a pressed?** block.

9. In the **Stage area**, click the **red stop sign** icon to stop your script and click **File > Save Now** to save your project.

   The script stops running and your project is saved.

Now that you know how to interact with your sprite by making your sprite do something in response to keyboard input, you can make the sprites in all of your Scratch projects do something in response to keyboard input. So far, your sprites have been quietly doing things when you run your scripts. You might be wondering how to make your sprite make a sound. See **Making Sounds** to learn how to make a sprite make a sound and record a sound for a sprite to make.

Making Sounds

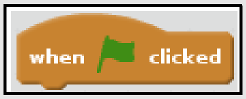
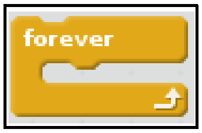
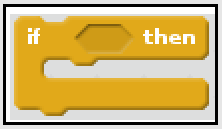
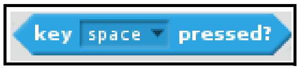
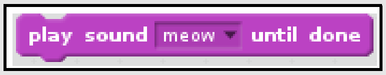
In the previous section, you learned how to interact with a sprite by making the sprite do something in response to mouse and keyboard input. So far, your sprites have been quiet, and you're probably curious about how to add sound to your Scratch programs. You can make a sprite make sounds by creating scripts using **Control**, **Sensing**, and **Sound** code blocks.

This section shows you how to make a sprite make some sounds. You will learn about:

- **Making sprites make a sound**
- **Making sprites make your sound**

The code blocks that you will be using are summarized in **Table 6**.

Table 6: Code Blocks Used in Making Sounds

Code Block	Name	Type
	when green flag clicked	Events
	forever	Control
	if then	Control
	key space pressed?	Sensing
	play sound meow until done	Looks

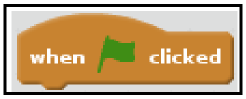
Making sprites make a sound

One of the ways to make a sprite make a sound is to choose a built-in sound from the **Sound** code blocks. You can make a sprite make a sound in response to a key being pressed in an existing project by creating a script with **Control**, **Sensing**, and **Sound** code blocks.

To make a sprite make a sound

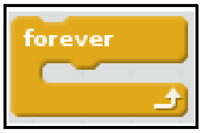
1. Create a new Scratch project with a cat sprite in it.
2. From the **Sprite list**, select the cat sprite that you want to make a sound.

   A blue box appears around the selected cat sprite.

3. On the **Scripts** tab of the **Blocks palette**, click **Events** and drag a **when green flag clicked** block 

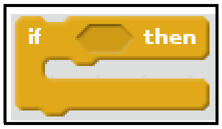
 to the **Scripts** area.

   *The **Events** code blocks appear and the **when green flag clicked** block appears in the **Scripts** area.*

4. On the **Blocks palette**, click **Control** and drag a **forever** block 

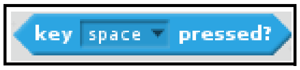
 to the **Scripts** area to connect to the **when green flag clicked** block.

   *The **forever** block is connected to the **when green flag clicked** block in the **Scripts** area.*

5. On the **Blocks palette**, drag an **if then** block 

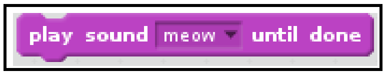
 to the **Scripts** area and connect it to the middle of the **forever** block.

   *The **if then** block is wrapped in the **forever** block in the **Scripts** area.*

6. On the **Blocks palette**, click **Sensing** and drag a **key space pressed?** block 

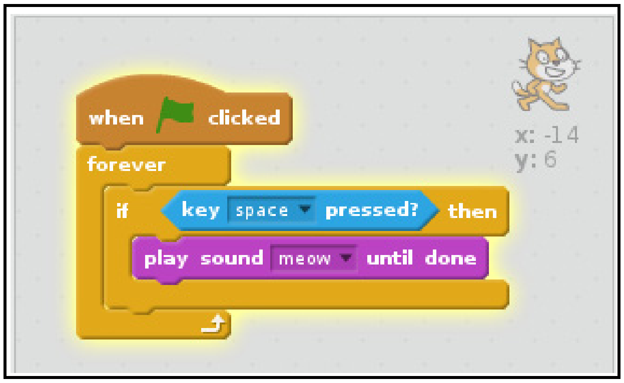
 to the **Scripts** area to place in the window of the **if then** block.

   *The **if then** block becomes an **if key space pressed? then** block in the **Scripts** area.*

7. On the **Blocks palette**, click **Sound** and drag a **play sound meow until done** block 

 to the **Scripts** area to connect to the middle of the **if key space pressed? then** block.

*The **play sound meow until done** block is wrapped in the **if key space pressed? then** block in the **Scripts** area and it becomes an **if key space pressed? then play sound meow until done** block. Your **Scripts** area should look similar to the one in the image below.*



```
when ⚑ clicked
forever
  if key space pressed? then
    play sound meow until done
```

8. In the **Stage area**, click the **green flag** icon ⚑ to run your script and press spacebar.

   Your sprite makes the meow sound.

   The sprite continues to make meow sounds if you hold down the spacebar.

9. In the **Stage area**, click the **red stop sign** icon 🛑 to stop your script and click **File > Save Now** to save your project.

   The script stops running and your project is saved.

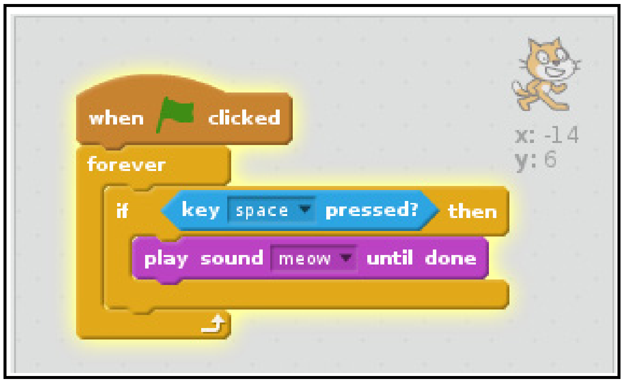
Now that you know how to add sound to your Scratch programs by choosing a built-in sound from the **Sound** code blocks, you can make the sprites in all of your Scratch projects make sounds. To make a Scratch program as unique as you, you might like to record your own sounds to use in your scripts. See **Making sprites make your sound** to learn how to record a sound for your sprite to make.

Making sprites make your sound

Another way to make a sprite make a sound is to record your own sound for your sprite to make by using your computer's microphone. You can make a sprite make your sound in response to keyboard input by recording your own sound and creating a script with **Control**, **Sensing**, and **Sound** code blocks.

To make a sprite make your sound

1. Open an existing Scratch project with a completed script for making your sprite make a sound as shown in the image below.



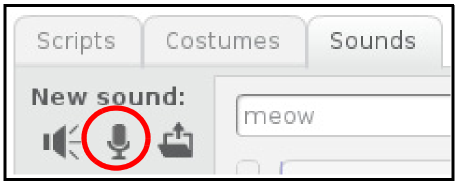
```
when [flag] clicked
forever
    if <key [space] pressed?> then
        play sound [meow] until done
```

To build a script like the one in the image above, see **Making sprites make a sound** on page 27.

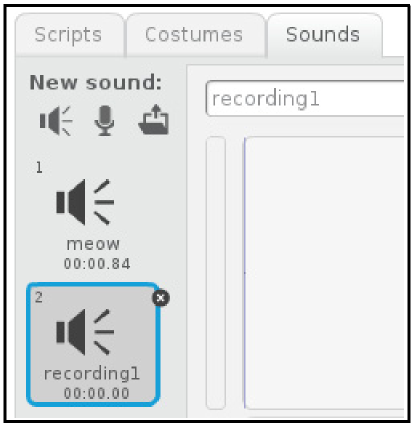
2. From the **Sprite list**, select the sprite that you want to make your sound.

A blue box appears around the selected sprite.

3. On the **Sounds** tab of the **Blocks palette**, in the **New sound list**, click the **Record new sound** icon.



A new sound, called ***recording1****, appears in the* ***New sound list*** *with a blue box around it.*



4. In the **Sounds area**, click the **record** icon ●.

   *An **Adobe Flash Player Settings** dialog opens, requesting that you allow the Scratch website access to your computer's camera and microphone for recording purposes.*

5. Click **Allow** and make your sound by speaking into your computer's microphone.

   *The **record** icon turns red and Scratch records your sound.*

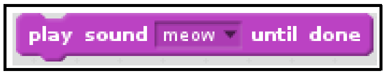
6. When you are finished making your sound, click the **stop** icon ■ to stop recording.

   *Recording stops and the **stop** icon returns to black.*

7. To test your recording, click the **play** icon ▶.

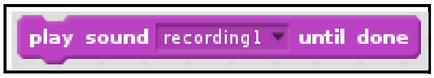
   *The **play** icon turns green and your sound plays back. When your recording ends, playback stops automatically and the **play** icon returns to black.*

8. On the **Scripts** tab of the **Blocks palett**e, in the **Scripts** area, on the **play sound meow until done** block, choose **recording1** from the drop-down combo box.

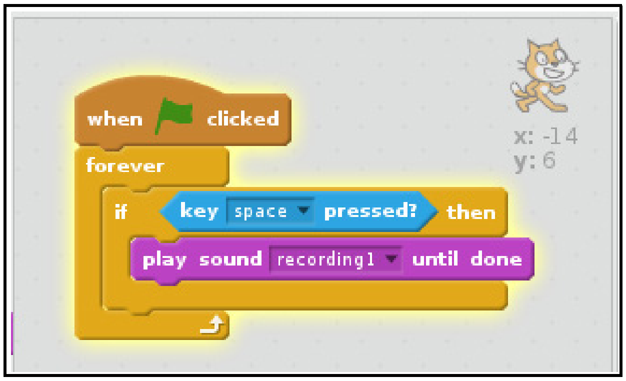
   

   play sound meow until done

   *The **play sound meow until done** block becomes a **play sound recording1 until done** block. Your **Scripts** area should look similar to the one in the image below.*

   play sound recording1 until done



```
when [flag] clicked
forever
    if <key space pressed?> then
        play sound recording1 until done
```

9. In the **Stage area**, click the **green flag** icon ⚑ to run your script and press spacebar.

   Your sprite makes your sound (i.e. the sound that you recorded as recording1).

10. In the **Stage area**, click the **red stop sign** icon 🛑 to stop your script and click **File > Save Now** to save your project.

    The script stops running and your project is saved.

If you would like to rename your sound, return to the **Sounds** tab and type a new name for your sound in the name box. When you return to the **Scripts** tab, your new sound name appears in your sprite's script. Resave your project.

If you would like to use a sound from the Scratch **Sound Library** instead of recording your own sound, go to the **Sounds** tab of the **Blocks palette**. In the **New sound list**, click the **Choose sound from library** icon 🔈. The **Sound Library** dialog appears. Choose a sound from the library and click OK. The **Sound Library** dialog closes and your chosen sound appears in the **New sound list** with a blue box around it. When you return to the **Scripts** tab, your chosen sound is available to be chosen from the drop-down combo box.

Now that you know how to record your own sound and make a sprite make *your* sound in response to a key being pressed, you can record your own sounds for the sprites to make in all of your Scratch projects.

Appendix A: Glossary

This Appendix lists definitions for frequently used technical terms in the Scratch User Manual.

Blocks palette The area of the Scratch user interface where you go to choose different types of code blocks to use in your **scripts**. With the **Scripts** tab selected, the **Blocks palette** is located down the middle of the Scratch user interface. See **Figure 1** on page 2.

costume A look or pose that defines a **sprite**'s appearance. See **Changing the appearance of sprites** on page 9.

keyboard input A method of providing instructions to a computer to tell it what to do by pressing keys on the keyboard.

mouse input A method of providing instructions to a computer to tell it what to do by clicking the mouse.

script A set of instructions that tell a computer what to do. In Scratch, you build **scripts** by connecting coloured blocks of code together like puzzle pieces. **Scripts** control the actions of **sprites** by instructing them to do things like move or speak.

Scripts area The area of the Scratch user interface where you build **scripts**. With the **Scripts** tab selected, the **Scripts area** is located on the right side of the Scratch user interface. See **Figure 1** on page 2.

sprite A movable character (e.g. person or animal) or object (e.g. car or tree) on the Scratch stage. A **sprite**'s actions are controlled by its **scripts**.

Sprite list The collection of **sprites** that are available on the stage. The **Sprite list** is located on the lower left of the Scratch user interface. See **Figure 1** on page 2.

Stage area The area of the Scratch user interface where **sprites** are active. The **Stage area** is located on the top left of the Scratch user interface. See **Figure 1** on page 2.

test statement An expression that is evaluated to be either true or false. If a **test statement** is found to be true, then the specified result happens. If the **test statement** is found to be false, then the specified result does not happen. See **Making sprites respond to mouse input** on page 21 for an example of a script that includes a test statement.

Appendix B: FAQ

This Appendix provides answers to some of the most frequently asked questions in Scratch programming.

How do I create a new project in Scratch?

> *If you do not have any Scratch projects open, use the **Create** button as described in **To create a sprite** on page 5 to create a new project. If you have a Scratch project open, click **File > New**.*

How do I save a project in Scratch?

> *To save your project, click **File > Save Now**.*

Do any sprites come with Scratch?

> *Yes, a number of sprites are available in the **Sprite Library**. See **To create a sprite** on page 5 for more information on using the **Sprite Library**.*

Where are the sprite costumes?

> *If you don't see your sprite's costumes, then you are probably on the **Scripts** tab. Click the **Costumes** tab at the top of the **Blocks palette** to switch to see your sprite's costumes.*

How do I change the name of a sprite?

> *You can change the name of a sprite in the **Sprite Information** panel. See **To rename a sprite** on page 6 for more information.*

Can I change the background of the stage?

> *Yes, in the **Stage list**, click the **New Backdrop** icon 🖼 to select a new backdrop from the **Backdrop Library**.*

What other kinds of moves can sprites make?

> *A sprite can be moved from one location on the stage to another by specifying new x, y coordinates for the sprite. Another way for a sprite to move is to glide from one location on the stage to another. Sprites*

*can also bounce off the edge of the stage. See the **Motion** blocks on the **Scripts** tab for more **Motion** block ideas to move your sprite.*

What keys should I use on the keyboard to control my sprite?

Popular keys that would be used to control a sprite are the spacebar (e.g. to make a sprite do something) and the arrow keys (e.g. to control the direction of a sprite’s moves). It's a good idea to use a letter related to the sprite action (e.g. 'b' to bark, 'm' to meow, ‘j’ for jump, or 'r' to rotate).

Do any sounds come with Scratch?

*Yes, a number of sounds are available in the **Sound Library**. If you would like to use a sound from the **Sound Library** instead of recording your own sound, go to the **Sounds** tab of the **Blocks palette**. In the **New sound list**, click the **Choose sound from library** icon 🔈. The **Sound Library** dialog appears. Choose a sound from the library and click OK.*

Where can I learn more about Scratch?

To learn more about Scratch, check out the wiki

http://wiki.scratch.mit.edu/wiki/Scratch

and Scratch’s own FAQ page

https://scratch.mit.edu/info/faq/

Appendix C: Troubleshooting Guide

This Appendix provides assistance for troubleshooting four common Scratch programming problems. For each problem in **Table 7**, a couple of questions are listed to help you identify the cause of the problem. Once you have identified the cause of the problem, you will find the corresponding recommended solution in the last column.

Table 7: Troubleshooting Common Problems in Scratch Programs

Type of Problem	Questions to Help Identify the Cause of the Problem	Solution to the Problem
Sprite won't move	Is the **green flag** icon in the **Stage** area bright green? If not, then the script is not running.	Click the **green flag** icon. Your script runs and your sprite moves according to your script. See the last step of **To move a sprite to the right** on page 13 for an example of clicking the **green flag** icon.
	Is there a **Motion** block in your script? If not, then you need to use at least one in your script.	Include a **Motion** block in your script. See **To repeat a sprite move** on page 17 for an example of a completed script that uses **Motion** blocks.
No sound	Check the task bar on your computer for the speaker icon. Is the sound muted? If so, then you need to turn the sound on.	Unmute your computer's sound. The next time that you run your script, the sound plays.
	Is there a **Sound** block in your script? If not, then you need to use at least one in your script.	Include a **Sound** block in your script. See **To make a sprite make a sound** on page 28 for an example of a completed script that uses **Sound** blocks. The next time that you run your script, the sound plays.

Table 7: Troubleshooting Common Problems in Scratch Programs

Type of Problem	Questions to Help Identify the Cause of the Problem	Solution to the Problem
Sprite doesn't face the direction of motion	Is there a **point towards mouse-pointer** block in your script? If not, then you need to use at least one in your script.	Include a **point towards mouse-pointer** block in your script. The next time that you run your script, your sprite will move in the direction that it is facing. See **To make sprites follow the mouse pointer** on page 14 for an example of a completed script that uses **point towards mouse-pointer** blocks.
	Does your **move n steps** block have a -10 in it? If so, then you need to change it from a **move -10 steps** block to a **move 10 steps** block.	Type 10 in place of the -10 to change your block into a **move 10 steps** block. The next time that you run your script, your sprite will move in the direction that it is facing.
Sprite disappeared	Check the edges of the **Stage** area. Can you see any part of your sprite? If so, then your sprite has probably moved just off of the edge of the stage. You need to drag your sprite back into the **Stage** area.	To drag your sprite back into the **Stage** area, carefully select the part of your sprite that is in the **Stage** area and drag your sprite back onto the stage.
	Check the properties of your sprite by clicking the blue 'i' button in the **Sprite list**. Is the **Show** option enabled? (i.e. is the "show:" box checked off?) If not, then you need to enable the **Show** option.	Enable the **Show** option (a check mark should appear in the box). Your sprite reappears.

Appendix D: Toolbar Reference

This Appendix provides a description of the toolbar buttons on the Scratch software application's **Cursor** and **Graphics** toolbars. The **Cursor** toolbar buttons are named and described in **Table 8**. The **Graphics** toolbar buttons are named and described in **Table 9** on page 39.

Table 8: Description of Cursor Toolbar Buttons

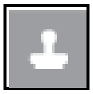
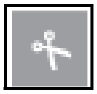
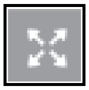
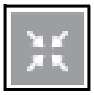
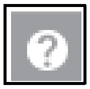
Button and Name	Description
 **Duplicate**	**Copy** a sprite or script.
 **Delete**	**Delete** a sprite or script.
 **Grow**	**Enlarge** a sprite.
 **Shrink**	**Shrink** a sprite.
 **Block help**	**Get help** on a block.
To use the **Cursor** tools, first click on the tool that you want to use. Next, click on the sprite or script that you want to use the tool on.	

Table 9: Description of Graphics Toolbar Buttons

Button and Name	Description
Select	**Select** a graphic object.
Reshape	**Change** the shape of a graphic object.
Pencil	**Draw** a line freehand (i.e. the line follows the mouse pointer).
Line	**Draw** a straight line between two points (i.e. the line starts where the mouse pointer is first clicked and ends where the mouse pointer is next clicked).
Rectangle (Shift: Square)	**Draw** a rectangle. Or, hold shift to **draw** a square.
Ellipse (Shift: Circle)	**Draw** an ellipse. Or, hold shift to **draw** a circle.
Text	**Write** on the backdrop.
Color a shape	**Fill** the selected shape.
Duplicate (Shift: Multiple)	**Copy** a graphic. Or, hold shift to make **multiple** copies of a graphic object.
To use the **Graphics** tools, first click on the tool that you want to use. Next, click on the graphic object that you want to use the tool on.	

Index