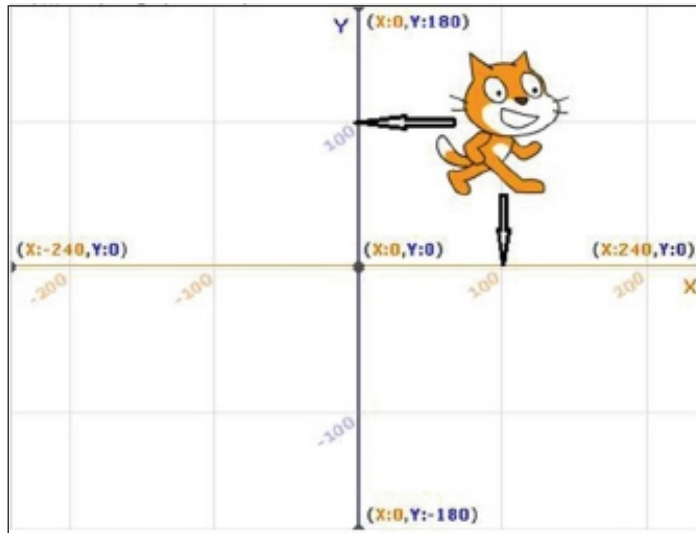


Chapter 2



Scratchy Moves

After completing this chapter you will be able to know how to use the Motion Blocks and make the sprite:

- move forward, move backward, move up;
- trigger motion, turn & backflip;
- glide and bounced;
- move to the nearest mouse position and
- move then point toward and follow the mouse pointer.

Make the Cat Move

This is the chapter that you've been waiting for. After reading the previous chapters, you now have all the background required to start programming in Scratch. In this chapter, you will learn about the blocks of code and how to use them to create scripts. You will also learn where to create scripts and where to run them. The focus of this chapter is specifically on the **Motion** category of code blocks. The blocks in this category instruct the sprite to move in various ways, and you'll learn how these blocks operate and how you can use them. Get ready, because you will start creating scripts in the second half of the chapter.

The Stage

You can think of the code blocks that you combine into a script as a series of instructions that tell your sprite what to do, as well as when and where to do it. To help you identify the point on the stage that your sprite currently occupies and specify exactly where to move it, Scratch uses a two-dimensional coordinate plane. To understand what that means, imagine the stage like a piece of graph paper with an **X** axis (a horizontal line) and a **Y** axis (a vertical line) crossing in the middle (see Figure 3-1). You can specify a location on the stage just like you'd plot a point on a graph, because every location on the stage is defined by a pair of **X** and **Y** positions, or coordinates. The **X** and **Y** axis cross, or intersect, in the middle of the stage, so the center of the stage is the point (0, 0).

While the **X**, **Y** coordinates tell you where the sprite is located, a measurement of degrees specifies in which direction the sprite is facing. Sprites can rotate 360 degrees on the stage. Facing to the right is 90 degrees. Facing toward the left is -90 degrees. Facing upward is 0 degree and downward is 180 degrees. So, in Figure 3-2, the cat is at position (100, 100) and facing 90 degrees.

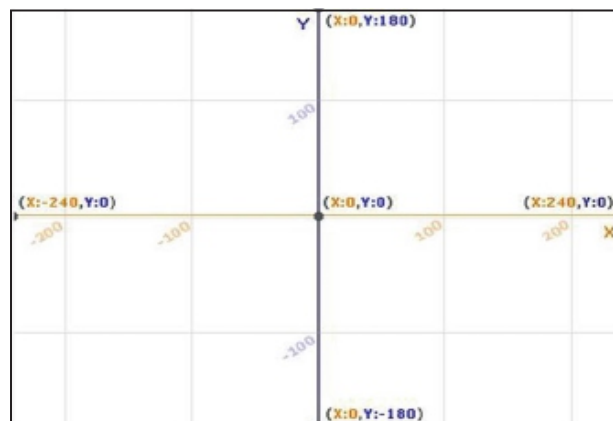


Figure 3-1. Two-dimensional coordinate plane

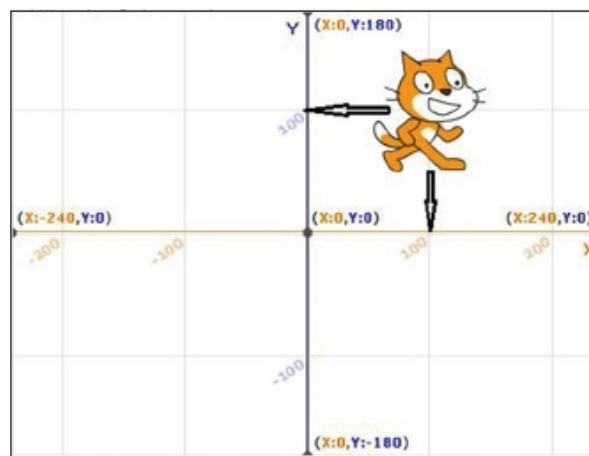


Figure 3-2. The center of the sprite is at the position (100, 100)

Scripts

In Scratch, scripts are the instructions that determine what happens on the stage. A script can have just one instruction (one block of code) or many. To create a script, you simply drag blocks of code from the block palette and snap them together in the scripts area. To snap one block to another, make sure that the notch of one block is aligned with the bump of the other one. When the blocks are close enough and ready to connect to each other, a white line (kind of like a glow) appears between them (see Figure 3-4). When you see it, let go of the block that you are dragging and it will snap automatically to the nearby block (see Figure 3-5).

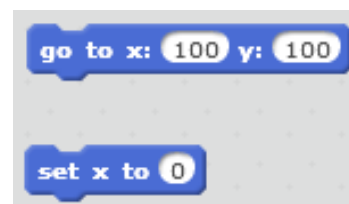


Figure 3-4. Before snapping together



Figure 3-5. After snapping together

There are six different block shapes that you can combine to create scripts. The shape of a block gives you a clue as to what the block does. The following describes the six shapes:

Hat blocks : Sometimes called *triggers*, these blocks activate the script and start it running. Hat blocks are rounded at the top and sit on top of everything, like a real hat. You can snap other blocks below them, but never above them. Most are found in the *Events* block category in the block palette. For example, the hat block in Figure 3-6 instructs the script to start when the green flag above the stage is clicked.

Stack blocks : These blocks are the main instructions in a script. They have a notch at the top and a bump at the bottom, like a jigsaw puzzle piece, so you can snap other blocks below and above them (see Figure 3-7).

Boolean blocks : Found in several categories in the block palette, these blocks contain conditions that are evaluated and then reported as either true or false. For example, the block in Figure 3-8 compares two items then reports true if the first is less than the second, and false if it isn't. In programming, conditions are used when a decision needs to be made; the result of the condition, whether true or false, determines the action to be taken. This may sound complicated, but you use the same Boolean logic in your daily life. For example, when you leave your office, whether you go left to the bus stop or right to walk home, depends on a condition: the weather. If it's raining, you dash to the left; if it's not, you go right and stroll home. You can embed Boolean blocks inside other blocks. You will find more about Boolean blocks in Chapter 6.

Reporter blocks : These blocks hold values. The value can be either a number or a string of characters. Like the Boolean blocks, these blocks can only be embedded inside other blocks. No blocks can be snapped below or above them. The *direction* block (see Figure 3-9), found in the *Motion* blocks category, is an example of a reporter block. This block holds the current direction of the sprite. For example, if you want to know the current direction of the sprite, you can use this block in your script to display the direction on the stage. You will find more about reporter blocks, including how to use them, in Chapter 6.

C blocks : These blocks are found in the *Control* blocks category. Other blocks can be snapped inside them. For example, Figure 3-10 shows a C block that will repeat the sequence of the actions or blocks within it four times. You will use C blocks throughout this book, but Chapter 6 focuses on the *Control* blocks category.

Cap blocks : These blocks are snapped at the end of a script. Like a cap on the end of a pipe, cap blocks stop the flow of instructions (see Figure 3-11). There are only two cap blocks, both of which are found in the *Control* blocks category.

To start a script to run, just click it. A yellow glow appears around the script to indicate that it is activated and running (see Figure 3-12).



Figure 3-6. Example hat block



Figure 3-7. Example stack block



Figure 3-8. Example Boolean block



Figure 3-9. Example reporter block



Figure 3-10. Example C block




Figure 3-11. Example cap block




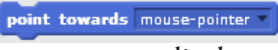


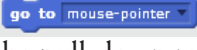

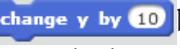
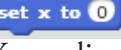





Figure 3-12. Activated script

Motion Blocks in Detail



Okay, so now that you have an understanding of the different block shapes and how to create scripts, let's take a closer look at the various Motion blocks, and then we'll start creating scripts. The Motion blocks category consists mostly of stack blocks. There are also some reporter blocks. With the blocks in the Motion category, you can make the sprite move in several ways. You can try out the following blocks by clicking to activate.

The  block moves the sprite the number of steps you specify. Each step is equal to the length of a pixel. If the number is positive, the sprite moves in the direction that it is pointing toward. If the number is negative, the sprite moves in the opposite direction from which it's pointing. For instance, the example block moves the sprite 10 steps (10 pixels) in the direction that it's pointing.

Several blocks change the orientation of the sprite. The  block turns the sprite clockwise according to the specified number of degrees. The  block works the same way, but turns the sprite counterclockwise by the specified number of degrees. The examples here turn the sprite 15 degrees, but you can change the number in the blocks. Click each block to activate it and then watch the sprite turn in each direction. The  block makes the sprite face toward a certain direction that you specify using the block's pull-down menu. Choose 90 to face the sprite to the right, -90 to face the sprite to the left, 0 to face it up, and 180 to face it down. The block points  the sprite toward the mouse-pointer or another sprite in the project. Use the block's pull-down menu to display your options and choose what to point to. You can select the mouse-pointer or another sprite. Sprites can move several ways. One way is by gliding, which gives the illusion of the sprite floating or gliding to a position. The  block makes the sprites glide for a specified number of seconds to a specified location. For instance, this block instructs the sprite to take 1 second to glide to the coordinates (-7, -283). Depending on the number of seconds, the sprite can move slower or faster. The go to blocks make the sprite move faster from one position to another. The  block moves the sprite to the specified location on the stage. Here the example block moves the sprite to the point (-7, -283). The  block moves the sprite toward the mouse-pointer or another sprite in the project. You can use the pull-down menu to select another sprite or a random position for the sprite to go to. The  block changes the current X coordinate by the specified number. For example, the block shown here adds 10 to the current X coordinate and moves the sprite to that X coordinate. If you use a negative number, the block will subtract that number from the current X coordinate. Similarly, the  block changes the current Y coordinate by the specified number. The  block enables you to set only the X coordinate of the sprite to a specified value and move the sprite to that X coordinate. Likewise, the  block enables you to set only a Y coordinate and move the sprite to it. If the sprite reaches the edge of the stage, the  block makes the sprite bounce back in the opposite direction from where it was traveling. In other words, when the sprite reaches an edge of the stage, it will change orientation and travel back in the opposite direction from which it was traveling before hitting the edge.

The  block sets the rotation style of the sprite. Using the pull-down menu in the block, you can choose one of three rotation styles: left-right means that the sprite can face only left or right; all around means that the sprite can face in any direction; and do not rotate means that the sprite always faces to the right (90 degrees) and cannot rotate. Three blocks open monitor windows on the stage to report information about your sprite.

The  block holds the value of the current X coordinate of the sprite. If you select the box in front of it, a window opens on the stage and displays the current X coordinate of the sprite.

The  block works the same way for the Y coordinate of the sprite. Similarly, the  block holds the value of the current direction of the sprite. Select the box in front of it to display the current direction of the sprite in a window on the stage.

ACTIVITIES

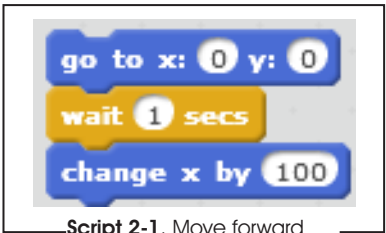
It's time to start snapping some blocks together to create scripts that get your cat sprite moving. If you accidentally drag the wrong code block from the block palette to the scripts area, you can always unsnap the blocks from each other by clicking them and dragging them apart. To clean up the scripts area, you can also drag a block of code or a complete script back to the block palette. This is one way to erase a block of code or a complete script from the scripts area. If you want to start a new project for each script, you can always choose File ➤ New from the menu bar to create a new project with a clean scripts area.

Activity 2-1: Move Forward

In this first activity let's take the cat on a forward walk. There are many ways to make the sprite move, and Script 2-1 demonstrates one of the simpler ways to do that.

In the Sprites pane, select the thumbnail of Sprite1 (the cat) to tell Scratch to send instructions from the script that you're about to create to this sprite. In the block palette, click the **Motion** category to display its contents, and then drag the `go to x: 0 y: 0` block to the scripts area. If the block is not set to the coordinates (0, 0), please set the X value to 0 and the Y value to 0. Next, click the **Control** category in the block palette and drag `wait 1 secs` to the scripts area and snap it to the previous block. If the block is not set to 1 second, please click in the number field to change it to 1 second. Finally, return to the **Motion** category, and drag and snap the `change x by 100` block to the script. If it's not set to 100, please change it so.

To activate and run the script, click it in the scripts area. The top block moves the sprite to the center of the stage. The next block, `wait 1 secs`, pauses the script for 1 second. The last code block adds 100 to the X coordinate of the sprite and moves the sprite to this new position. Because the sprite's X coordinate starts at 0, adding 100 changes it to 100. The Y coordinate stays the same, so the sprite moves to coordinates (100, 0). Table 3-1 lists the blocks and describes the actions used in this activity.



OUTPUT

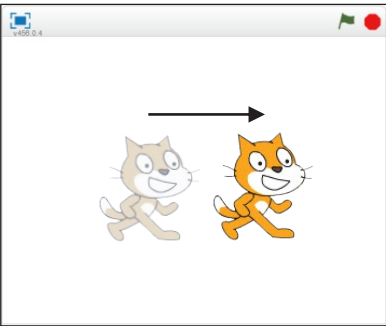





Table 2-1. Code Blocks in Move Forward




Blocks	Actions
	Move the sprite to coordinates (X = 0, Y = 0), which is the middle of the stage.
	The script waits 1 second. No actions are performed for 1 second.
	Add 100 to the current value of the X coordinate and move the sprite to this new X coordinate.

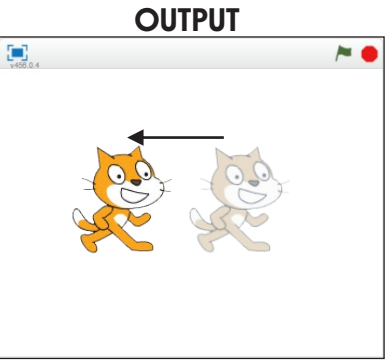
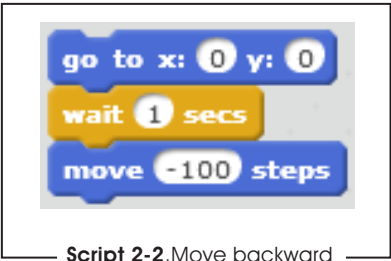
Activity 2-2: Move Backward

This activity shows you how to create a script to make the sprite move backward. You can create Script 2-2 next to the previous one in the scripts area, or you can create it in a clean scripts area all by itself. Remember to select the thumbnail of the cat sprite first, and then drag the three blocks shown from the Motion and Control categories to snap them together in the scripts area. Be sure to set the values shown for each, as well. Again, the top block of code will move the sprite to the center of the stage and the next pauses the script for 1 second.


The last block of code moves the sprite 100 pixels in the opposite direction from which the sprite is facing, because the value specified is -100 . Run the script, and then try changing the value for the final block to 100. Click to run the script again. This time, because the value is positive, the sprite moves in the same direction that it is facing. Table 2-2 lists the blocks and describes the actions used in this activity.

Table 2-2. Code Blocks in Move Backward

Blocks	Actions
	Move the sprite to coordinates (X = 0, Y = 0), which is the middle of the stage.
	The script waits 1 second. No actions are performed for 1 second.
	Move the sprite 100 pixels to the opposite direction from which it's facing.



Activity 2-3: Move Up

Script 2-3 shows yet another way of moving the sprite. Because this script is the same as the previous ones, except for the last block of code, you can just remove the last block from Script 2-3, replace it with the  block from the Motion category in the block palette, and set its value to 100.





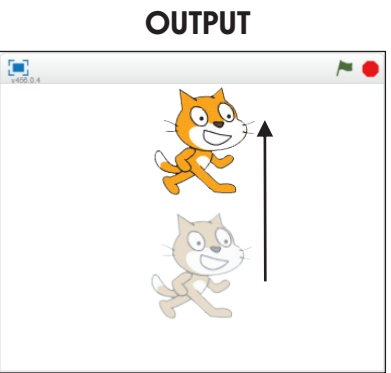
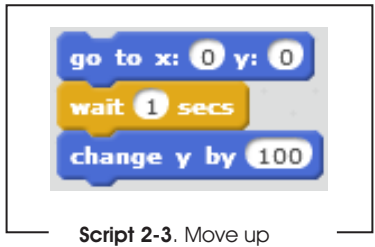
Click the script to run it. This time, the cat moves up from the center of the stage to the coordinates (0, 100) because the final block adds 100 to the original Y coordinate value of 0. Remember, you can always change the values in the code blocks' fields. Experiment with a new start position by changing the values in the  block. Pause the script for a longer length of time by increasing the second block's value or by moving the cat down by using a negative value in the final block. Run the script with your new values and watch how the sprite reacts. Table 2-3 lists the blocks and describes the actions used in this activity.

Table 2-3. Code Blocks in Move Up

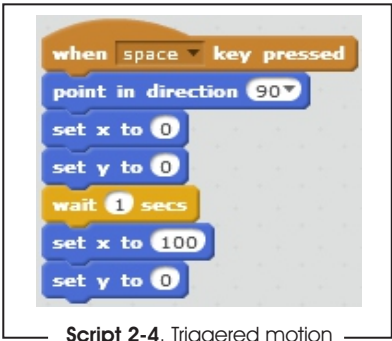
Blocks	Actions
	Move the sprite to coordinates (X = 0, Y = 0), which is the middle of the stage.
	The script waits 1 second. No actions are performed for 1 second.
	Add 100 to the current value of the Y coordinate and move the sprite to this new Y coordinate.



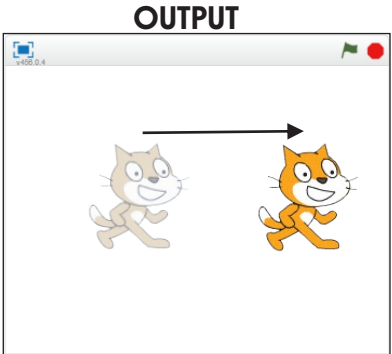
Activity 2-4: Triggered Motion

Script 2-4 introduces a hat block as a trigger to activate the script when the user performs a certain action. You can find the hat blocks in the **Events** category in the block palette. Drag the **when space key pressed** block from the **Events** category to the scripts area. Then choose **space** from the pull-down menu, if it's not already selected. Drag and snap together the remaining **Motion** and **Control** blocks needed to re-create Script 2-4, setting the parameters as shown.

The first block activates the script when the user presses the space bar (or you can choose a different key from the pull-down menu if you prefer). The next block makes the sprite face to the right (90 degrees). The third block of code sets the X coordinate to 0 and the next block sets the Y coordinate to 0. So the sprite begins by facing right and at the center of the stage. After a pause of 1 second, the last two blocks set the X and Y coordinates to 100 and 0, respectively, so that the sprite will move from the center to coordinates (100, 0). **Table 2-4** lists the blocks and describes the actions used in this activity.










Script 2-4. Triggered motion



OUTPUT

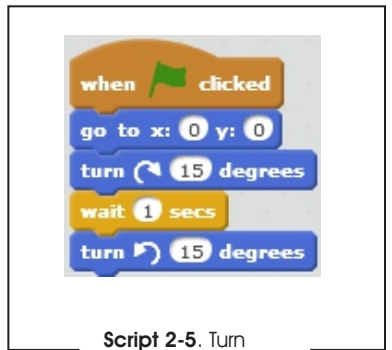
Table 2-4. Code Blocks in Triggered Motion

Blocks	Actions
	This block triggers the script. The script gets activated by pressing the space bar.
	Point the direction of the sprite to the right.
	Set the value of the X coordinate to 0 and move the sprite to that location.
	Set the value of the Y coordinate to 0 and move the sprite to that location.
	The script waits 1 second. No actions are performed for 1 second.
	Set the value of the X coordinate to 100 and move the sprite to that location.
	Set the value of the Y coordinate to 0 and move the sprite to that location.

Activity 2-5: Turn

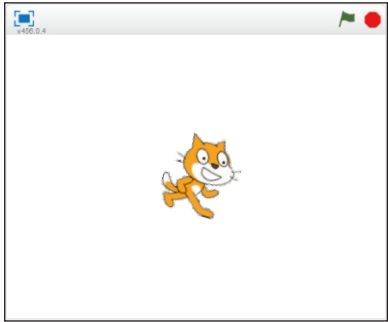
Now try rotating your sprite. Drag the blocks from the **Events**, **Motion**, and **Control** categories of the block palette and snap together, as shown in Script 2-5. (Remember to select the thumbnail of the sprite in the sprites area first). This script uses the new hat block that activates the script when the user clicks the green flag above the stage. You still can activate the script by simply clicking it, though.

After the user clicks the green flag, the next block moves the sprite to the center of the stage.








Script 2-5. Turn

OUTPUT







The next block is new. It rotates the sprite 15 degrees clockwise from its current orientation. After a familiar 1 second pause, the last block rotates the sprite 15 degrees counterclockwise—back in its original orientation. Remember that you can always change the values in the fields of the blocks to experiment and see how the sprite reacts. **Table 2-5** lists the blocks and describes the actions used in this activity.

Table 2-5. Code Blocks in Turn

Blocks	Actions
	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
	Move the sprite to coordinates (X = 0, Y = 0), which is the middle of the stage.
	Turn the sprite clockwise 15 degrees.
	The script waits 1 second. No actions are performed for 1 second.
	Turn the sprite counterclockwise 15 degrees.

Activity 2-6: Backflip

This activity rotates the sprite 360 degrees counterclockwise, creating the illusion of the sprite performing a backflip. Take a look at Script 2-6 . By now, you can probably tell what's going to happen. This time, the hat block activates the script when the user clicks the sprite on the stage. The next block makes the sprite face to the right. The third block moves the sprite to the center of the stage. Next is something new. It's a C block. Give it a try.

After dragging and snapping the first three blocks, go to the **Control** category and drag the  block into place on the stage. Now snap the  and  blocks inside it. Make sure that each block's setting matches Script 2-6 . After positioning the sprite at (0, 0), the script performs the first instruction inside the C block and turns the sprite counterclockwise 90 degrees from its current orientation. Next, the script pauses 0.1 second, and then follows the arrow back to the top of the C block. Because it is set to 4, the  block repeats the sequence of instructions within it four times, flipping the cat 90 degrees from its current orientation each time.

Run the script and watch it make the cat do a backflip. You may think you're only snapping blocks together, but you're writing a real computer program. A Scratch script and a traditional computer program's lines of code are both lists of instructions telling the computer what to do. In this case, the blocks are telling the computer to move the sprite. **Table 2-6** lists the blocks and describes the actions used in this activity.



OUTPUT

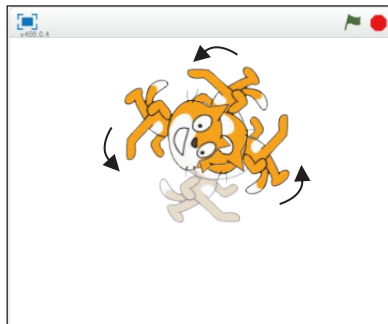








Table 2-6. Code Blocks in Backflip

Blocks	Actions
	Clicking the sprite activates the script.
	Point the direction of the sprite to the right.
	Move the sprite to coordinates (X = 0, Y = 0), which is the middle of the stage.
	Repeat the actions represented by the blocks within this block four times.
	Turn the sprite counterclockwise 90 degrees.
	The script waits 1 second. No actions are performed for 1 second.

Activity 2-7: Square Pattern Motion

This time, you’ll send the cat in a square pattern, from the center stage to coordinates (100, 0), (100, 100), and (0, 100), and then back to the center of the stage in its original position, facing to the right. Think about which blocks you might use to accomplish this, and then take a look at Script 2-7 . Was your idea close? Drag and snap the blocks and adjust their settings to match the example script to try it.

According to the hat block, when the user clicks the green flag, the script starts. The next block makes the sprite face to the right. The third block moves the sprite to the center of the stage. The pull-down menu in the fourth block is set to give the sprite a rotation style of all around . This means that the sprite will be able to rotate all around (a 360-degree radius). The next block pauses the script for 1 second.

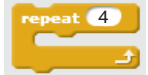


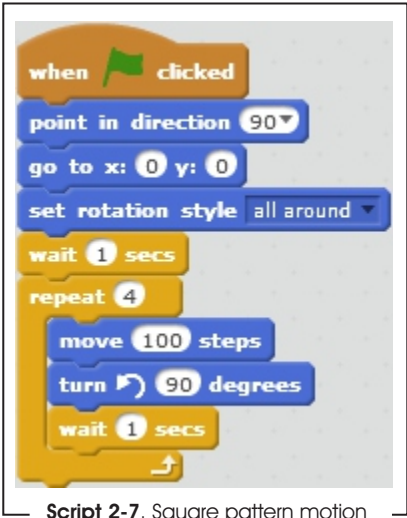
The  block executes the blocks within it four times. The first block within it moves the sprite 100 pixels in the direction that it’s facing. The next rotates the sprite 90 degrees counterclockwise, and the last block within the C block pauses the script for 1 second before repeating the three-block sequence. **Table 2-7** lists the blocks and describes the actions used in this activity.

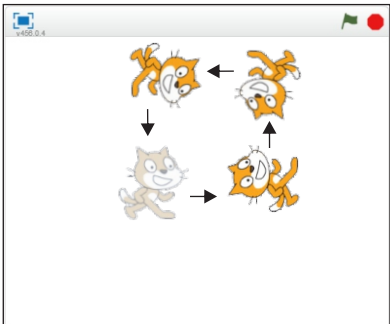
Table 2-7. Code Blocks in Square Pattern Motion

Blocks	Actions
	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
	Point the direction of the sprite to the right.



Script 2-7. Square pattern motion

OUTPUT





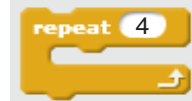
Move the sprite to coordinates (X = 0, Y = 0), which is the middle of the stage.



Set the rotation style of the sprite so that it can rotate all around its axis.



The script waits 1 second. No actions are performed for 1 second.



Repeat the actions represented by the blocks within this block four times.



Move 100 pixels.



Turn the sprite counterclockwise 90 degrees.

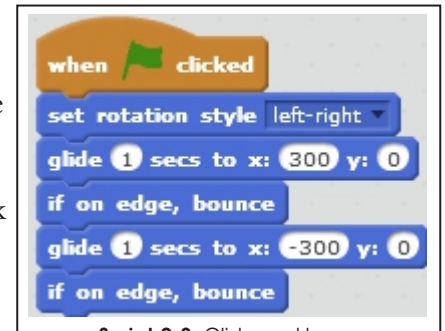


The script waits 1 second. No actions are performed for 1 second.

Activity 2-8: Glide and Bounce

The stage area has limits. If the sprite reaches any of the edges of the stage, it can get stuck. The **if on edge, bounce** block prevents this, as you'll see in Script 2-8. Drag and snap the blocks together, and then watch what happens to the cat when it tries to glide past the edge of its world.

The script starts running when the user clicks the green flag. The next block sets the rotation style of the sprite to **left-right**. This means that the sprite can only face left or right. The third block glides the block for 1 second to coordinates (300, 0). Remember that the limit of the X coordinate for the stage is 240, so the sprite will hit the edge (see Figure 2-13). The fourth block makes the sprite bounce when it hits the edge and travel in the opposite direction (see Figure 2-14). The next block will glide the sprite for 1 second to coordinates (-300, 0). Remember the limit of the X coordinate in that direction is -240. The last block will bounce the sprite when it hits the edge and make it travel in the opposite direction. **Table 2-8** lists the blocks and describes the actions used in this activity.



Script 2-8. Glide and bounce

OUTPUT

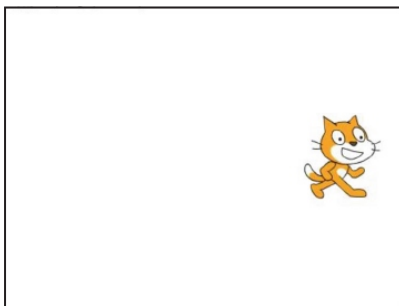
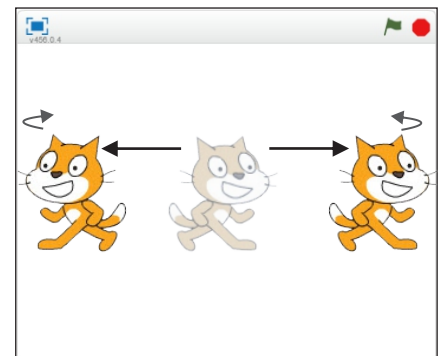


Figure 2-13. Before reaching edge

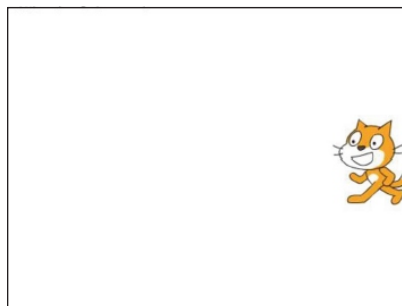




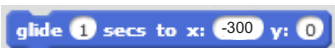



Figure 2-14. After reaching the edge

Table 2-8. Code Blocks in Glide and Bounce






Blocks	Actions
	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
	Set the rotation style of the sprite so that it can only turn left or right.
	Glide the sprite for 1 second to coordinates, X = 300 and Y = 0.
	If the sprite reaches the edge of the stage, bounce in the opposite direction.
	Glide the sprite for 1 second to coordinates, (X = -300, Y = 0).
	If the sprite reaches the edge of the stage, bounce in the opposite direction.

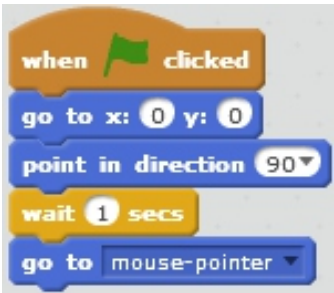
Activity 2-9: Go to the Mouse-Pointer

It’s time to let your cat have some fun with your mouse. Specifically, Script 2-9 demonstrates how to use your mouse-pointer to control the sprite. Drag and snap the necessary blocks together to try out this technique. The script starts when the user clicks the green flag. The next block moves the sprite to the center of the stage. The third block makes the sprite face to the right. The next block pauses the script for 1 second. The last block makes the sprite move toward the mouse-pointer.

Click the green flag, and then change the position of your mouse-pointer. Watch the sprite move toward it automatically. Table 2-9 lists the blocks and describes the actions used in this activity.

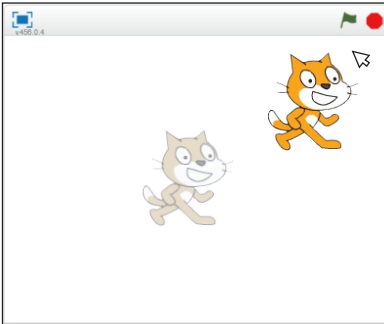
Table 2-9. Code Blocks in Go to the Mouse-Pointer

Blocks	Actions
	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
	Move the sprite to coordinates (X = 0, Y = 0), which is the middle of the stage.
	Point the direction of the sprite to the right.
	The script waits 1 second. No actions are performed for 1 second.
	Move the sprite toward the mouse-pointer.




Script 2-9. Go to the mouse-pointer

OUTPUT



Activity 2-10: Move with the Mouse-Pointer

With the addition of a C block, you can keep your cat chasing your mouse forever—or at least until you click the stop button to stop the script. As you snap together Script 2-10, you’ll notice that it is identical to Script 2-9 until you reach the  block.







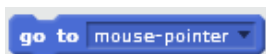
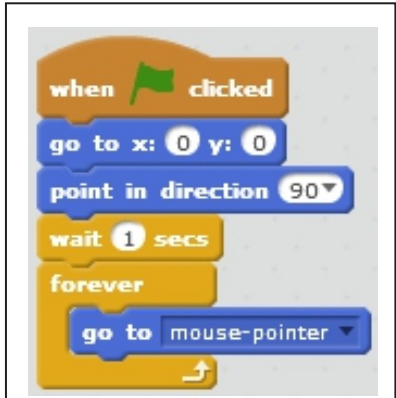
Found in the Control category, this block will continue to execute the sequence of blocks within it until the script is stopped manually. In this case, there’s only one block inside it: the  block. Click the green flag to run the script, and then move the mouse-pointer to watch how the sprite moves to it. To stop the script, click the red icon next to the green flag. **Table 2-10** lists the blocks and describes the actions used in this activity.

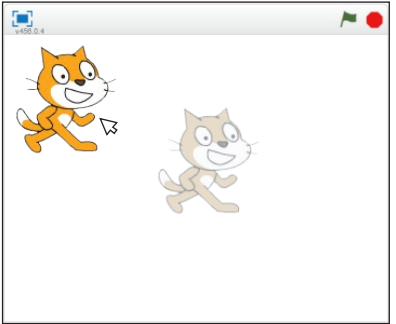
Table 2-10. Code Blocks in Move with the Mouse-Pointer

Blocks	Actions
	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
	Move the sprite to coordinates (X = 0, Y = 0), which is the middle of the stage.
	Point the direction of the sprite to the right.
	The script waits 1 second. No actions are performed for 1 second.
	Repeat the actions represented by the blocks within this block forever—or until the script is stopped manually.
	Move the sprite toward the mouse-pointer.




Script 2-10. Move with the mouse-pointer

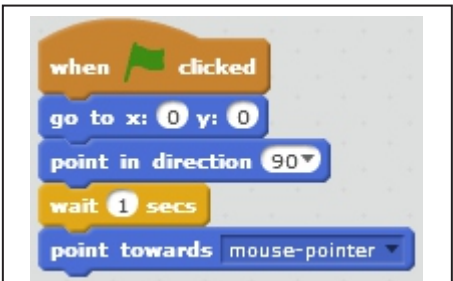
OUTPUT



Activity 2-11: Point Toward the Mouse-Pointer


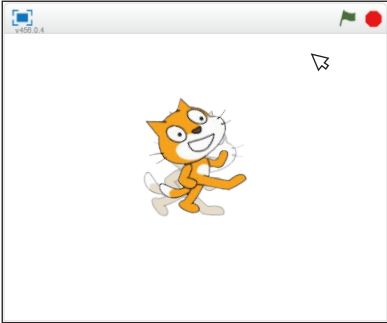
What if you want your sprite to turn to face the mouse-pointer rather than move to its position? The answer is to use the  block, which makes the sprite face the direction of the mouse-pointer instead of moving toward it. Give Script 2-11 a try.

The actions of the first four blocks should be familiar. Clicking the green flag runs the script, the sprite moves to the center of the stage and faces to the right, and then the script pauses for 1 second. The last block of code faces the sprite toward the direction of the mouse-pointer. Activate the script and move the mouse-pointer to see how the sprite rotates to face its direction. **Table 2-11** lists the blocks and describes the actions used in this activity.




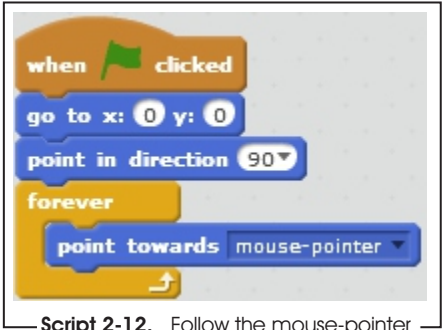
Script 2-11. Point toward the mouse-pointer

Table 2-11. Code Blocks in Point Toward the Mouse-Pointer

Blocks	Actions	OUTPUT
	<p>Clicking the green flag activates the script. The green flag is the trigger to start the script running.</p> <p>Move the sprite to coordinates (X = 0, Y = 0), which is the middle of the stage.</p> <p>Point the direction of the sprite to the right.</p> <p>The script waits 1 second. No actions are performed for 1 second.</p> <p>Point the direction of the sprite toward the mouse-pointer.</p>	


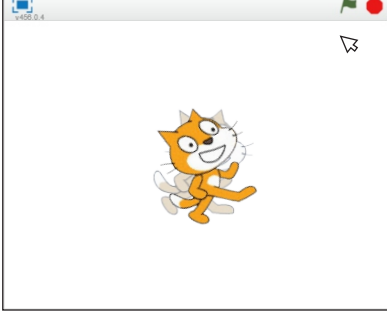
Activity 2-12: Follow the Mouse-Pointer

Script 2-12 combines techniques from Scripts 2-10 and 2-11. Instead of making the sprite rotate to face the direction of the mouse-pointer once, it makes the sprite face the mouse-pointer continuously until the script is stopped. The script starts running when the user clicks the green flag. The next two blocks of code will move the sprite to the center of the stage and make it face to the right. The next block, the C block, will execute the block within it forever until the script is stopped manually. The block within the  block will make the sprite face the direction of the mouse-pointer. Create the script, activate it, and move the mouse-pointer around to watch how the sprite rotates. **Table 2-12** lists the blocks and describes the actions used in this activity.



Script 2-12. Follow the mouse-pointer

Table 2-12. Code Blocks in Follow the Mouse-Pointer

Blocks	Actions	OUTPUT
	<p>Clicking the green flag activates the script. The green flag is the trigger to start the script running.</p> <p>Move the sprite to coordinates (X = 0, Y = 0), which is the middle of the stage.</p> <p>Point the direction of the sprite to the right. Repeat the actions represented by the blocks within this block forever—or until the script is stopped manually.</p> <p>Point the direction of the sprite toward the mouse-pointer.</p>	

Summary

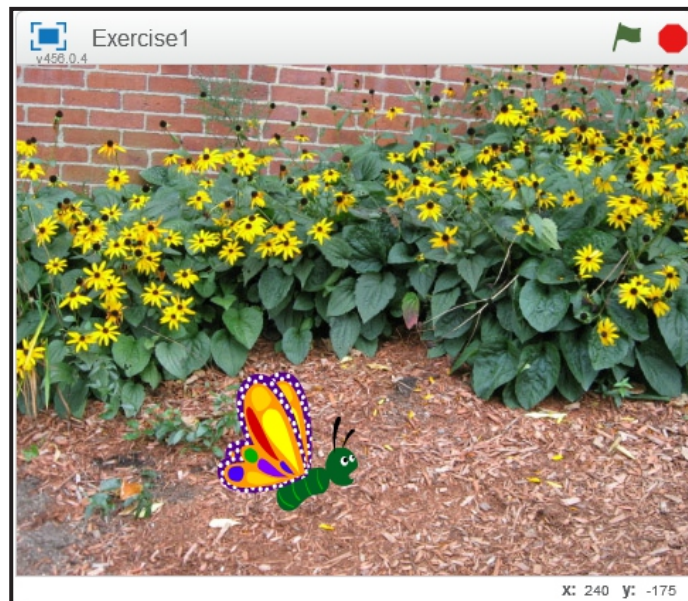
Congratulations, you now know how to build a basic script and move a sprite around the stage. You're a Scratch programmer! In this chapter, you learned how to create a script and how to activate it. You also learned about the various types of blocks of code and you specifically practiced using blocks in the Motion blocks category, among others. In the next chapter, you will continue using the knowledge that you gained here, but you focus on how to make the sprite draw on the stage.

Snap Script

a Short hands-on activity

1. Create a script to make the sprite go to positions (0, 0), (100, 100), and (−50, −100).

Output:



2. Create a script that makes the sprite rotate 360 degrees on its axis. The user should activate the script by clicking the green flag.

Output:

