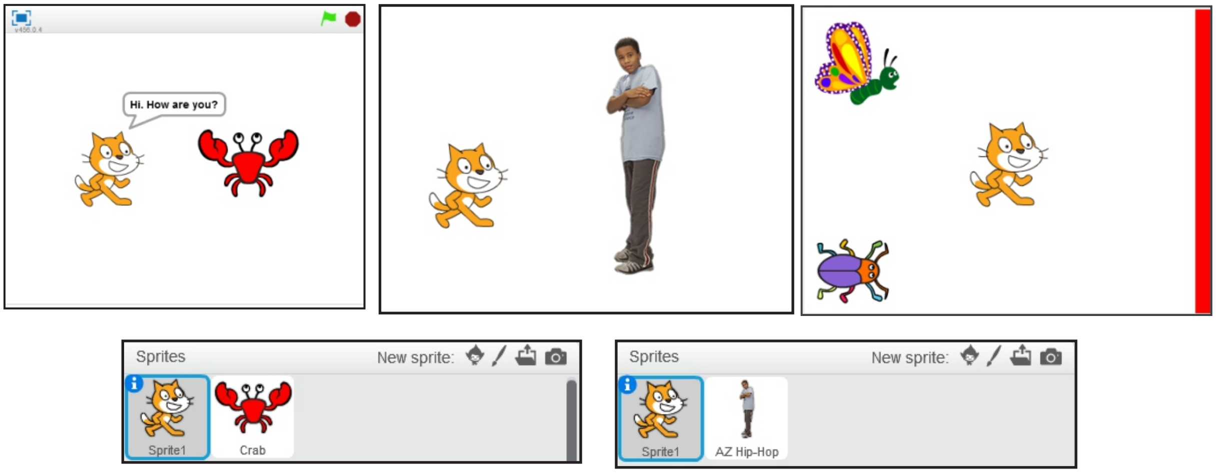
Chapter 9



Broadcast Interaction with Scratchy & the Gang

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

- do basic dialog using the "broadcast message" block;
- have an advanced conversation using "when I receive message" broadcast blocks;
- conversation on Math using both basic and advance broadcast blocks; and
- do the race and scenery change using broadcast blocks.

Broadcast Interaction

By now, you've learned several ways to activate a script besides simply clicking it in the scripts area. For example, you can use hat blocks in the **Events** block category to instruct a script to start when the user clicks the green flag or presses the space bar. But what if you could activate one script automatically from inside another script with no user clicking or key presses required? That's what this chapter teaches you how to do: use a block in one script to broadcast an instruction to another script, telling it to activate.

If you have two sprites in a project, for example, you can create one script for each and make the script you created for one sprite instruct the other sprite's script to start. On the stage, it looks as if the sprites are interacting with each other, which comes in handy when you want to create a project that tells a story. In a game, for example, you could have a sprite's script instruct a backdrop's script to start after the sprite performs a specific action. In this chapter, you will learn about the blocks that allow this interaction by broadcasting instructions between scripts, and then you'll create some scripts that use these blocks.

Broadcast Blocks

The activation of one script by another script consists of two parts: one script broadcasts the start instruction, which Scratch calls a **message**, and the other script receives the message and then activates. Each activation message sent between scripts has a unique name that is specified in both scripts, enabling you to send multiple messages to multiple scripts in a single project, with certainty that the various receiving scripts will start at the appropriate times. The blocks of code you need for both broadcasting and receiving messages are found in the **Events** blocks category (see Figure 9-1).

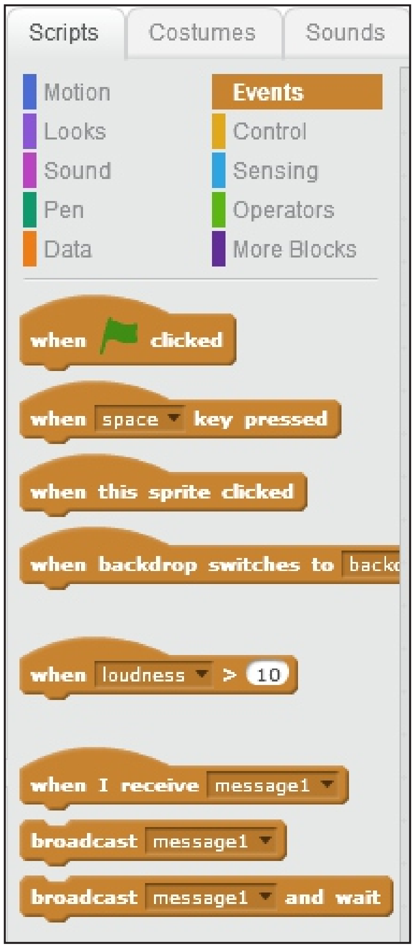


Figure 9-1. Events

To broadcast a message, you have the choice of two blocks. The broadcast message1 ▾ block broadcasts the specified message, and then the action progresses to the next block in the script. As you can see in the block's pull-down menu, the default message name is **message1**. You can create your message name, as well. Simply select **new message…** from the menu (see Figure 9-2), and when the New Message window (see Figure 9-3) opens, give your message a descriptive name and click the **OK** button. Remember, the message you create doesn't contain any value; it's a signal to trigger another script. The name simply helps you differentiate between messages that are being broadcasted and received in larger projects. The broadcast message1 ▾ and wait block also broadcasts the message you specify, but Scratch waits until the receiving script finishes running before performing the action of the next block in the broadcasting script.

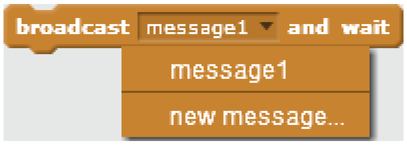


Figure 9-2. Choose to broadcast the default message or to create a new one

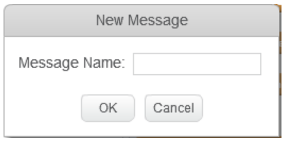


Figure 9-3. Name your message in the New Message window

Remember, for each broadcast block that sends an instruction to activate, you need to add a receiving block at the top of the script to be activated. The when I receive message1 block "listens for" the specified message. When it receives that message, the block triggers the script snapped below it and Scratch starts going through the next blocks of code in the script. Let's go next through a couple of examples showing these blocks in action.

Activities

By creating and running the scripts in the next examples, you will get a better understanding of how broadcasting works and how it can give sprites the illusion of reacting to each other's actions. For instance, you'll set up one sprite's script to broadcast a trigger to another sprite's script to mimic a conversation, as well as create a script for a sprite that sends a message to activate a backdrop-changing script at a key time. Remember, try to figure out what the script does before you create and run it. This is a good way to learn a programming language.

Activity 9-1: Basic Dialog

Without broadcasting messages, setting up interactions between sprites requires building in pauses and guesswork to ensure that the timing of the two sprites' scripts is in sync. To provide a contrast for the broadcast-based examples that follow, this first example sets up a short conversation between two sprites by adding carefully placed wait 2 secs blocks. Before you create the scripts, add one more sprite to the project. I'll use the Crab sprite, but you can choose whichever one you prefer.

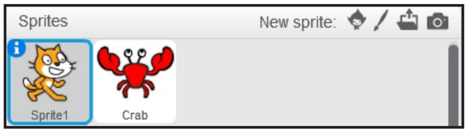
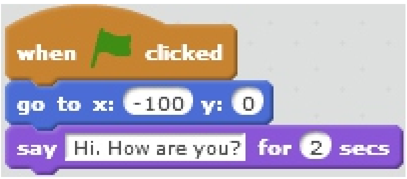


Figure 9-4. Sprites pane



```
when [green flag] clicked
go to x: -100 y: 0
say Hi. How are you? for 2 secs
```

Script 9-1. Dialog: Sprite 1 speech

Select the first sprite in the Sprites pane (see Figure 9-4), and snap together Script 9-1 in the scripts area. The script activates when the user clicks the green flag. The second block moves the sprite to the position on the stage where the coordinates are (–100, 0). The last block of code creates a speech bubble that displays Hi. How are you? for 2 seconds (see Figure 9-5). **Table 9-1** lists the blocks and describes the actions used in this activity.

OUTPUT

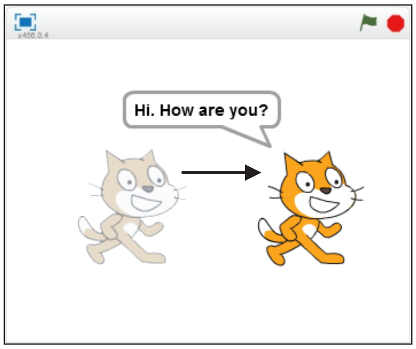
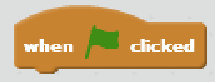
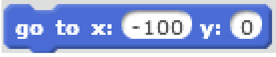
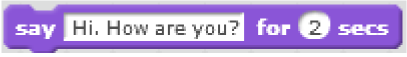
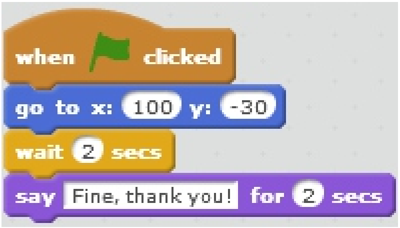


Table 9-1. Dialog: Sprite 1 Speech Code Blocks

Blocks	Actions
 when [green flag] clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
 go to x: -100 y: 0	Move the sprite to the position where X = –100 and Y = 0.
 say Hi. How are you? for 2 secs	The sprite gets a speech bubble that displays Hi. How are you? for 2 seconds.

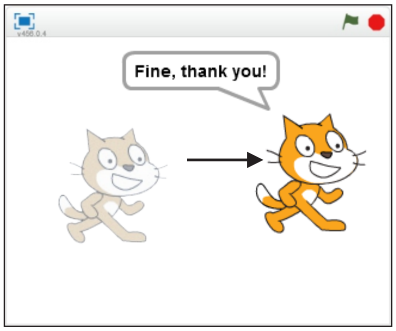
Now, set up your second sprite's answer. Select the second sprite and create Script 9-2 in the scripts area. The script starts running when the user clicks the green flag. The next block moves the selected sprite to (100, –30). The third block pauses the script for 2 seconds to stay in sync with the first sprite's script, and the last block creates a speech bubble that displays Fine, thank you! for 2 seconds.



```
when ⚑ clicked
go to x: 100 y: -30
wait 2 secs
say Fine, thank you! for 2 secs
```

Script 9-2. Dialog Sprite 2 speech

OUTPUT



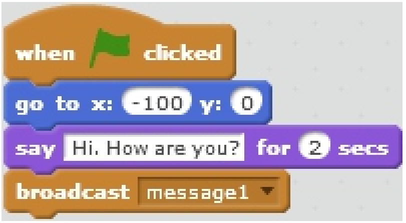
Remember, the first sprite's speech bubble displayed for 2 seconds. Because both scripts start at the same time (when the user clicks the green flag), the second sprite's script must pause for 2 seconds before its say Fine, thank you! for 2 secs block so that the second speech bubble displays only after the first one disappears from the stage. Although coordinating speech bubbles and pauses is manageable in this simple activity, imagine trying to coordinate the script timing for a longer, more complex conversation—or one among three sprites. Broadcasting messages makes the task much easier, as you'll learn in the next activity. **Table 9-2** lists the blocks and describes the actions used in this activity.

Table 9-2. Dialog: Sprite 2 Speech Code Blocks

Blocks	Actions
when ⚑ clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
go to x: 100 y: -30	Move the sprite to the position where X = 100 and Y = –30.
wait 2 secs	The script waits 2 seconds. No actions are performed for 2 seconds.
say Fine, thank you! for 2 secs	The sprite gets a speech bubble that displays Fine, thank you! for 2 seconds.

Activity 9-2: Advanced Conversation

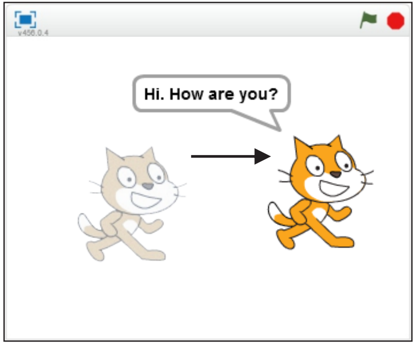
This more efficient activity produces the same result as Activity 9-1 by using the broadcast message1 ▾ and when I receive message1 ▾ blocks. Once again, make sure that you have two sprites added to the project. Select the first sprite in the Sprites pane, and create Script 9-3. The script starts running when the user clicks the green flag.



```
when ⚑ clicked
go to x: -100 y: 0
say Hi. How are you? for 2 secs
broadcast message1 ▾
```

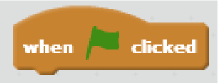
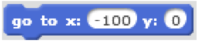
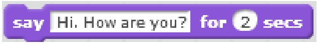
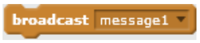
Script 9-3. Conversation: Sprite 1 speech

OUTPUT



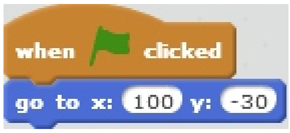
The next block moves the sprite to the position (100, 0). The third block creates a speech bubble that displays Hi. How are you? for 2 seconds. The last block of code broadcasts **message1** to trigger Script 9-5 , which controls the second sprite's speech.
Table 9-3 lists the blocks and describes the actions used in this activity.

Table 9-3. Conversation: Sprite 1 Speech Code Blocks

Blocks	Actions
 when ⚑ clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
 go to x: -100 y: 0	Move the sprite to the position where X = –100 and Y = 0.
 say Hi. How are you? for 2 secs	The sprite gets a speech bubble that displays Hi. How are you? for 2 seconds.
 broadcast message1 ▾	Broadcast message1 .

Select the second sprite. Create Script 9-4 to control its position and Script 9-5 to control its speech.

Script 9-4 starts running when the user clicks the green flag and moves the sprite to the position (100, –30).



```
when ⚑ clicked
go to x: 100 y: -30
```

Script 9-4. Conversation: Sprite 2 position

OUTPUT

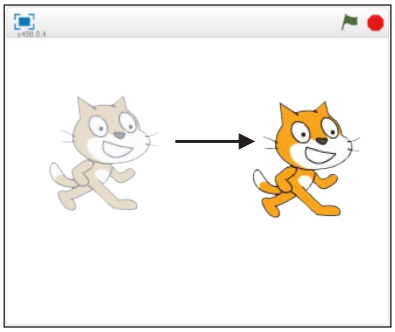
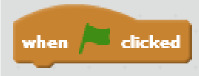
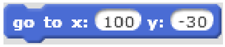
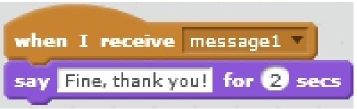


Table 9-4 lists the blocks and describes the actions used in this activity.

Table 9-4. Conversation: Sprite 2 Position Code Blocks

Blocks	Actions
 when ⚑ clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
 go to x: 100 y: -30	Move the sprite to the position where X = 100 and Y = –30.

Script 9-5 activates when it receives the message broadcast by the first sprite. Specifically, the first block activates the script when it receives **message1** , which Script 9-3 broadcasts after it displays the first sprite's speech bubble. The next block creates a speech bubble for the second sprite that displays Fine, thank you! for 2 seconds. The timing of the speech bubbles is coordinated automatically by the scripts—no more figuring out and manually adding pauses. Broadcasting and receiving a message makes coordinating the interaction between scripts easier and more efficient. **Table 9-5** lists the blocks and describes the actions used in this activity.



```
when I receive message1 ▾
say Fine, thank you! for 2 secs
```

Script 9-5. Conversation: Sprite 2 speech

OUTPUT

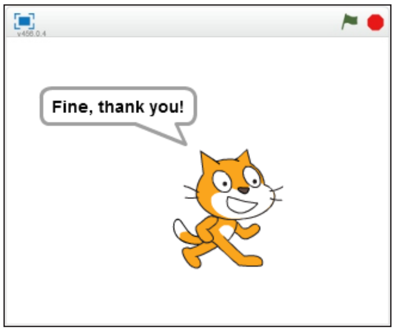
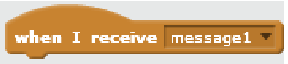
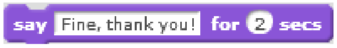


Table 9-5. Conversation: Sprite 2 Speech Code Blocks

Blocks	Actions
 when I receive message1 ▾	Receiving message1 activates the script.
 say Fine, thank you! for 2 secs	The sprite gets a speech bubble that displays Fine, thank you! for 2 seconds.

Activity 9-3: Dance

This activity builds on the concepts in Activity 9-2. You will create a conversation between the two sprites in Figure 9-6 , and then instruct one of them to dance. Before you create the scripts for this activity, make sure that you add the **AZ Hip-Hop** to the Sprites pane (see Figure 9-7). You also need to add the **dance celebrate** sound file from the Sound Library.

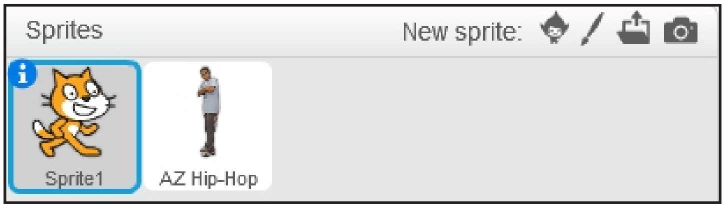


Figure 9-7. The Sprites pane

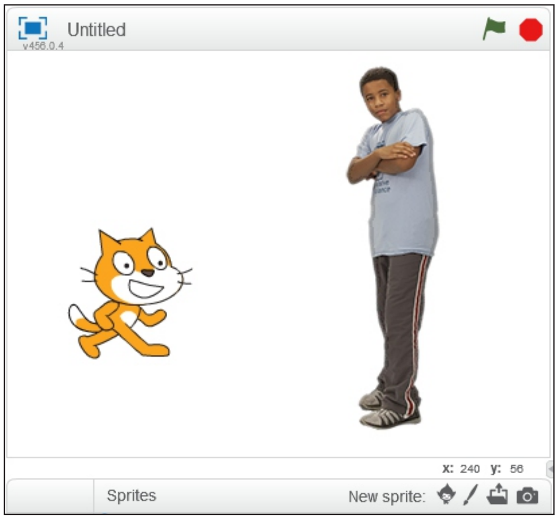
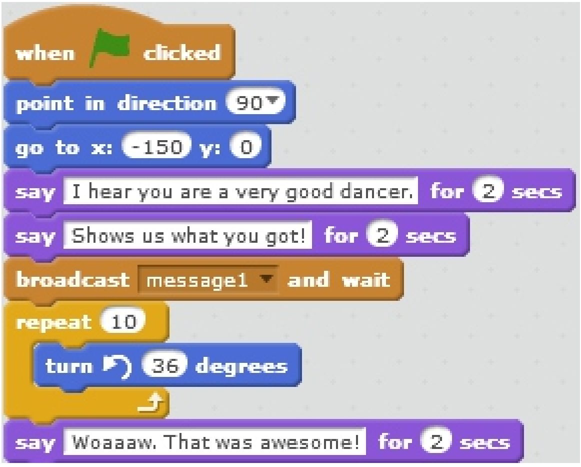


Figure 9-6. Dance sprites

To begin, select **Sprite1** and create Script 9-6 .

Script 9-6 starts running when the user clicks the green flag. The next two blocks make the sprite face to the right and move it to the position (–150, 0). The next two blocks each create a speech bubble that lasts for 2 seconds. The first one displays I heard you are a very good dancer. , and then the second one displays Show us what you got! The next block broadcasts **message1** to trigger Script 9-7 (the script that you will create to receive the broadcast) and waits until that script completes. Scratch then executes the next code block in the broadcasting script, which is a repeat 10 block that repeats the single block within it 10 times. That block rotates the sprite 36 degrees counterclockwise. After the loop completes, the last block creates a speech bubble that displays Woaaaw. That was awesome! for 2 seconds. Why rotate the sprite by repeating a 36-degree rotation 10 times instead of rotating 360 degrees at once? Because rotating 360 degrees at once wouldn't show the sprite rotating, since the starting and ending position of the sprite would be the same. **Table 9-6** lists the blocks and describes the actions used in this activity.



```
when clicked
point in direction 90
go to x: -150 y: 0
say I hear you are a very good dancer. for 2 secs
say Shows us what you got! for 2 secs
broadcast message1 and wait
repeat 10
    turn ↺ 36 degrees
say Woaaaw. That was awesome! for 2 secs
```

Script 9-6. Dance: Sprite 1 speech

OUTPUT

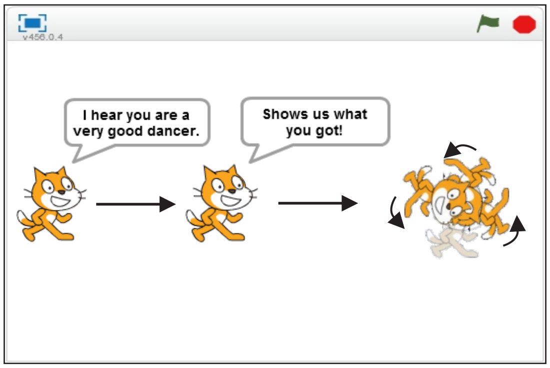
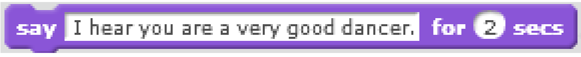
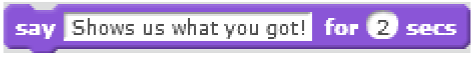
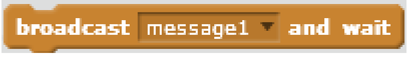
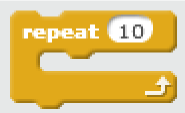
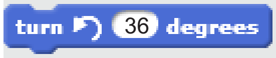
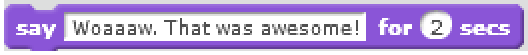


Table 9-6. Dance: Sprite 1 Speech Code Blocks

Blocks	Actions
when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
point in direction 90	Make the sprite face to the right.
go to x: -150 y: 0	Move the sprite to the position where X = –150 and Y = 0.

 say I hear you are a very good dancer. for 2 secs	The sprite gets a speech bubble that displays I hear you are a very good dancer. for 2 seconds.
 say Shows us what you got! for 2 secs	The sprite gets a speech bubble that displays Show us what you got! for 2 seconds.
 broadcast message1 and wait	Broadcast message1 and wait until all the scripts that received the message have finished.
 repeat 10	Repeat the actions represented by the blocks within this block ten times.
 turn ↺ 36 degrees	Turn the sprite counterclockwise 36 degrees.
 say Woaaaw. That was awesome! for 2 secs	The sprite gets a speech bubble that displays Woaaaw. That was awesome! for 2 seconds.

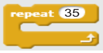
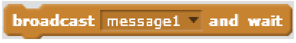
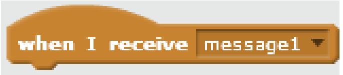
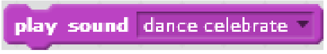
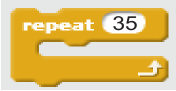
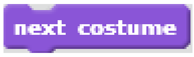
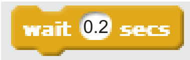
The broadcasting script is finished, so now you need to set up the receiving script for the second sprite. Select the **AZ Hip-Hop** sprite and create Script 9-7 . The hat block indicates that Script 9-7 activates when message1 is received. The next block plays the **dance celebrate** sound. The  repeat 35 block repeats the two actions within it 35 times. First, the sprite changes to its next costume, and then the script pauses for 0.2 seconds. The sprite dances while the music plays. Notice how  broadcast message1 and wait affects the interaction between the scripts. Because the first sprite's script waits for the second script to complete before it continues, on the stage it appears that the cat prompts and then reacts to the dancer sprite's actions. If you had used instead, the cat would have started spinning and speaking at the same time the dancer began to change costumes. **Table 9-7** lists the blocks and describes the actions used in this activity.

Table 9-7. Dance: Sprite 2 Actions Code Blocks

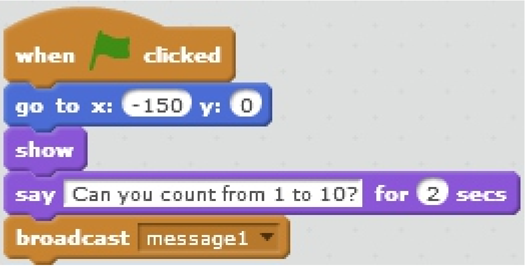
Blocks	Actions
 when I receive message1	Receiving message1 activates the script.
 play sound dance celebrate	Play the sound dance celebrate .
 repeat 35	Repeat the actions represented by the blocks within this block 35 times.
 next costume	Change the sprite to the next costume in the costumes tab.
 wait 0.2 secs	The script waits 0.2 seconds. No actions are performed for 0.2 seconds.

Activity 9-4: Math Test

Now that you've mastered broadcasting one message, try broadcasting three. This next activity shows interaction amongst five scripts that control a back-and-forth conversation about math between two sprites. One script asks a math question and triggers the script that answers, which in turn triggers a second question that then triggers a second answer. That's a lot of broadcasting and receiving, so you'd better get started.

The first step is to assemble all the elements you need. Make sure that you have two sprites in the Sprites pane. Next, create two additional messages and name them **message2** and **message3** . You can create the new message name, either while the block is still in the block palette or after you have dragged and dropped it in the scripts area. Last but not least, create a variable called **number**.

Next, let's create the scripts that control the sprite that asks the math questions. Select the Cat1 sprite (I chose the cat), and create Script 9-8 , which asks the first math question and broadcasts message1 to activate the second sprite's first answer script (see Script 9-11).



when clicked
go to x: -150 y: 0
show
say Can you count from 1 to 10? for 2 secs
broadcast message1

Script 9-8. Math Test: Question1

The script starts running when the user clicks the green flag. The next block moves the sprite to the position (–150, 0). The next block displays the sprite on the stage if it was hidden. If the sprite was not hidden, nothing will happen. The next block creates a speech bubble that displays Can you count from 1 to 10? for 2 seconds (see Figure 9-8). The last block of code broadcasts **message1**.

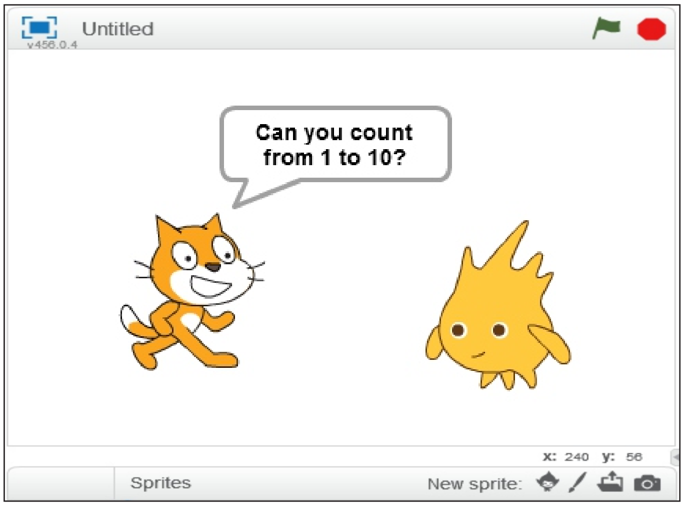
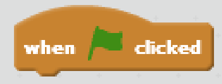
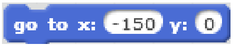
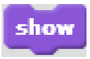
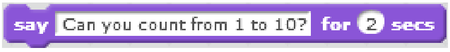
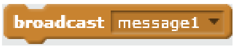


Figure 9-8. The Math test's first question

Table 9-8 lists the blocks and describes the actions used in this activity.

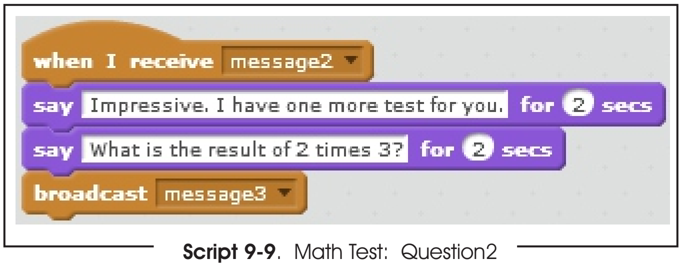
Table 9-8. Math Test: Question1 Code Blocks

Blocks	Actions
 when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
 go to x: -150 y: 0	Move the sprite to the position where X = –150 and Y = 0.
 show	Make the sprite appear on the stage.
 say Can you count from 1 to 10? for 2 secs	The sprite gets a speech bubble that displays Can you count from 1 to 10? for 2 seconds.
 broadcast message1	Broadcast message1 .

With the Cat1 sprite still selected, create the Script 9-9 , which receives message2 from the second sprite's first answer script (see Script 9-11), asks a second question, and then broadcasts message3 to the second sprite's second answer script (see Script 9-12).

Script 9-9 starts running when it receives message2 . The next two blocks create speech bubbles. The first bubble displays Impressive. I have one more test for you. for 2 seconds, and the second displays What is the result of 2 times 3? I for 2 seconds. The last block of code broadcasts message3 to activate the second sprite's second answer in Script 9-12 .

Table 9-9 lists the blocks and describes the actions used in this activity.



```
when I receive message2
say Impressive. I have one more test for you. for 2 secs
say What is the result of 2 times 3? for 2 secs
broadcast message3
```

Script 9-9. Math Test: Question2

OUTPUT

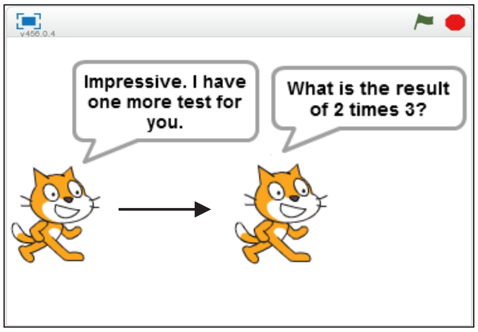
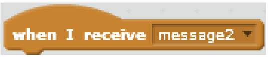
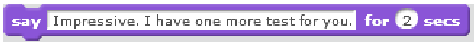
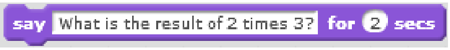
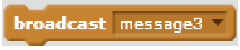
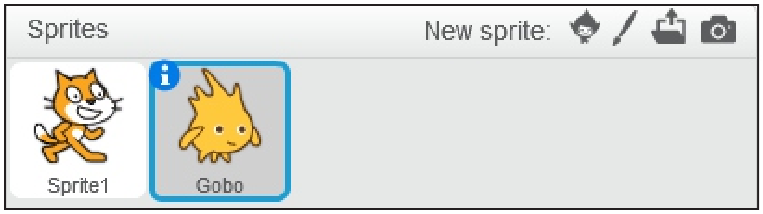
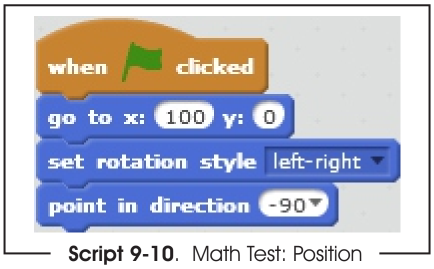

Impressive. I have one more test for you.
What is the result of 2 times 3?

Table 9-9. Math Test: Question2 Code Blocks

Blocks	Actions
 when I receive message2	Receiving message2 activates the script.
 say Impressive. I have one more test for you. for 2 secs	The sprite gets a speech bubble that displays Impressive. I have one more test for you. for 2 seconds.
 say What is the result of 2 times 3? for 2 secs	The sprite gets a speech bubble that displays What is the result of 2 times 3? for 2 seconds.
 broadcast message3	Broadcast message3 .

The Cat1 sprite's scripts are finished, so you're ready to move on to the answering sprite's scripts. Go ahead and select the second sprite. All the scripts you create now in the scripts area will send instructions to this sprite. Start with Script 9-10, which positions the second sprite on the stage.

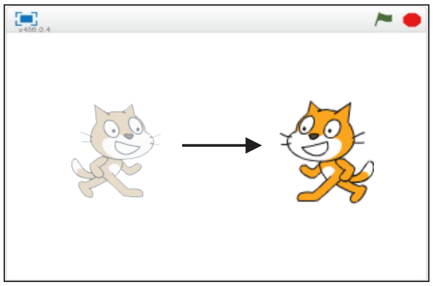




```
when [green flag] clicked
go to x: 100 y: 0
set rotation style left-right
point in direction -90
```

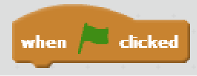
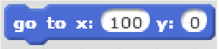
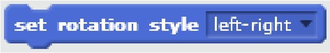
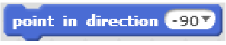
Script 9-10. Math Test: Position

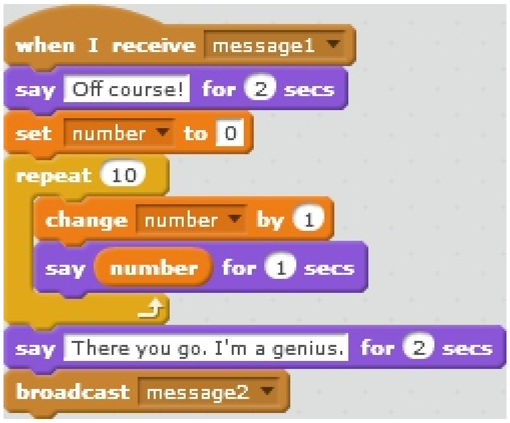
OUTPUT



Script 9-10 starts running when the user clicks the green flag. The next block moves the sprite to the position (100, 0). The next block sets the rotation style of the sprite to **left-right**. The last block makes the sprite face to the left. **Table 9-10** lists the blocks and describes the actions used in this activity.

Table 9-10. Math Test: Position Code Blocks

Blocks	Actions
 when 🏴 clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
 go to x: 100 y: 0	Move the sprite to the position where X = 100 and Y = 0.
 set rotation style left-right ▾	Set the rotation style of the sprite, so that it can only turn left or right.
 point in direction -90▾	Make the sprite face to the left.



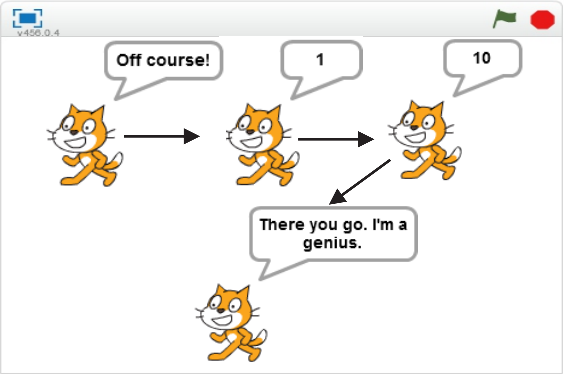
```
when I receive message1 ▾
say Off course! for 2 secs
set number ▾ to 0
repeat 10
    change number ▾ by 1
    say number for 1 secs
say There you go. I'm a genius. for 2 secs
broadcast message2 ▾
```

Script 9-11. Math Test: Answer1

Keeping the second sprite selected, create Script 9-11 with its first math answer, which starts running upon receiving **message1** from Script 9-8 and activates Script 9-9 by broadcasting **message2**. Script 9-11 starts running when it receives message1 broadcast from Script 9-8 .

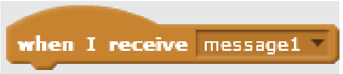
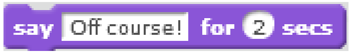
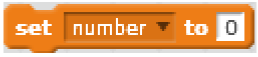
The next block creates a speech bubble that displays Of course! for 2 seconds. The next block assigns the value 0 to the **number** variable. Next, there's a repeat 10 block that repeats the sequence of actions within it 10 times. The first block in the sequence adds 1 to the current value of **number** , and the second creates a speech bubble that displays the value of number for 1 second.

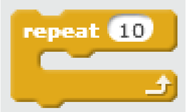
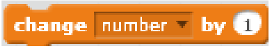
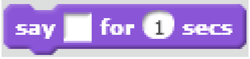
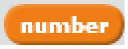
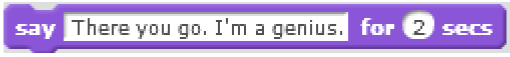
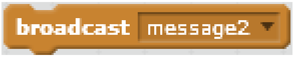
OUTPUT



After the second sprite counts from 1 to 10 in the loop, the next block creates a speech bubble that displays There you go. I'm a genius. for 2 seconds. Finally, the last block of code broadcasts **message2** to trigger Script 9-9.
Table 9-11 lists the blocks and describes the actions used in this activity.

Table 9-11. Math Test: Answer1 Code Blocks

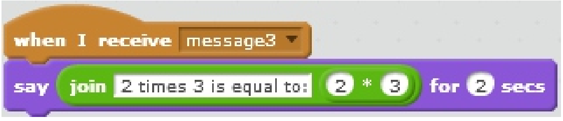
Blocks	Actions
 when I receive message1 ▾	Receiving message1 activates the script.
 say Off course! for 2 secs	The sprite gets a speech bubble that displays Of course! for 2 seconds.
 set number ▾ to 0	Set the current value of the variable called number to 0.

 repeat 10	Repeat the actions represented by the blocks within this block ten times.
 change number by 1	Add 1 to the current value of the number variable.
 say [ ] for 1 secs	The sprite gets a speech bubble that displays the specified text or value.
 number	Hold and report the current value of the number variable.
 say There you go. I'm a genius. for 2 secs	The sprite gets a speech bubble that displays There you go. I'm a genius. for 2 seconds.
 broadcast message2	Broadcast message2 .

You're almost done. You just need to create Script 9-12 , which receives **message3** from Script 9-9 and gives the second sprite's final math answer.

Make sure that the second sprite is still active and snap together the final two blocks in the scripts area.

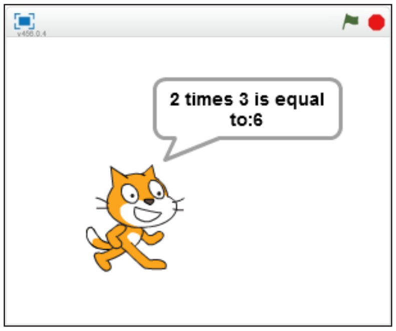
Table 9-12 lists the blocks and describes the actions used in this activity.



```
when I receive message3
say join 2 times 3 is equal to: (2 * 3) for 2 secs
```

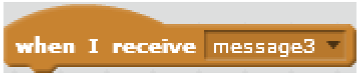
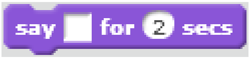
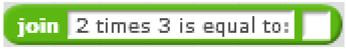
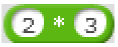
Script 9-12. Math Test: Answer2

OUTPUT



2 times 3 is equal to:6

Table 9-12. Math Test: Answer2 Code Blocks

Blocks	Actions
 when I receive message3	Receiving message3 activates the script.
 say [ ] for 2 secs	The sprite gets a speech bubble that displays the specified text or value.
 join 2 times 3 is equal to: [ ]	Join the two values that have been specified in the block and report the result.
 2 * 3	Multiply two values (2 * 3) and report the result.

So now that you have learned how to make the scripts and sprites interact with each other automatically, you can create your own stories using the same methods.

Activity 9-5: Race

This project illustrates how you can use the broadcasting technique to create a story in which the sprites race each other to a finish line (see Figure 9-9).

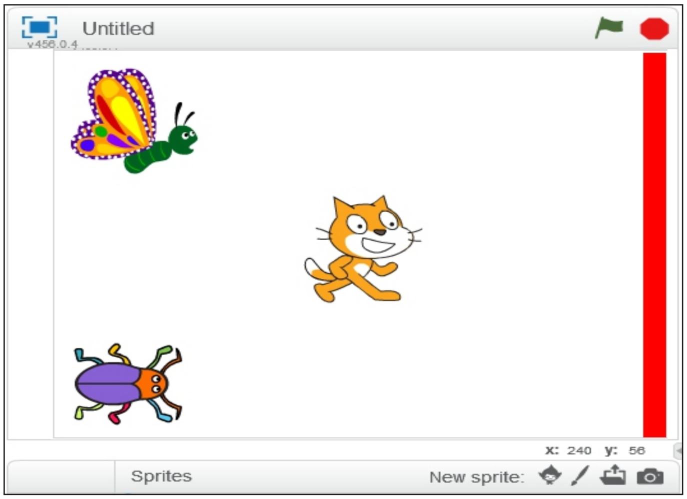


Figure 9-9. Race

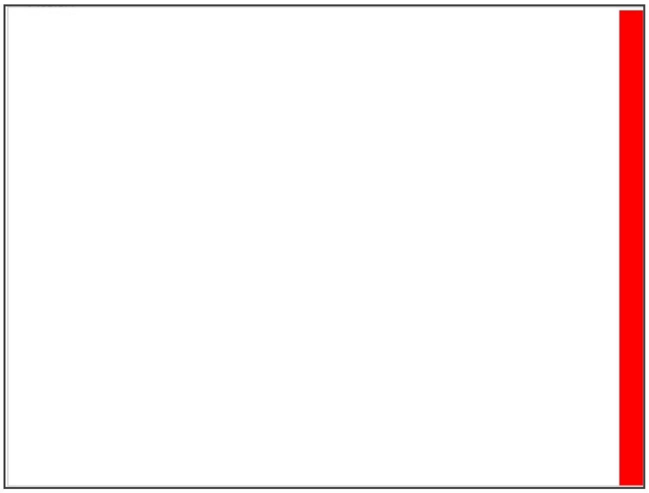


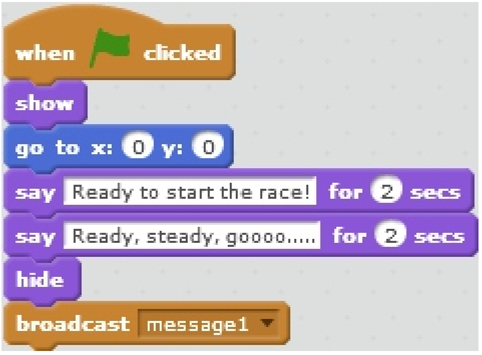
Figure 9-11. Race backdrop

Before creating the scripts, you need to make a few preparations. You'll need three sprites for this project. I added **Sprite1** , **Beetle** , and **Butterfly3** to the Sprites pane from the Scratch library (see Figure 9-10). You will also need to have **message1** , **message2** , and **message3** available for broadcast. Finally, add a vertical red line to the white backdrop (see Figure 9-11) to act as the finish line for the race. (Consult Chapter 2 if you don't remember how to do this.)

The cat, Sprite1 , directs the race between the beetle and the butterfly, so let's begin with the cat's three scripts. First, select **Sprite1** and create Script 9-13 , which starts the racers by broadcasting **message1**.

Script 9-13 starts running when the user clicks the green flag. The next two blocks show the sprite on the stage and move it to the center of the stage.

The next pair of blocks creates speech bubbles for 2 seconds each, first displaying Ready to start the race! , and then Ready, steady, gooooo.... . The next block hides the sprite from the stage. The last block broadcasts **message1** , which triggers Scripts 9-17 and 9-19. **Table 9-13** lists the blocks and describes the actions used in this activity.



```
when 🏳 clicked
show
go to x: 0 y: 0
say Ready to start the race! for 2 secs
say Ready, steady, goooo..... for 2 secs
hide
broadcast message1 ▾
```

Script 9-13. Race: Ready

OUTPUT

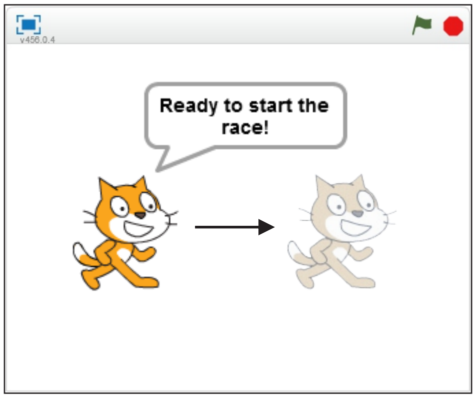
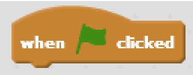
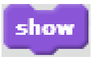
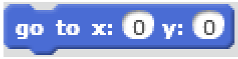
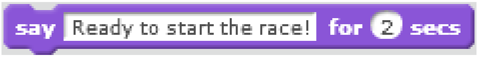
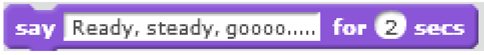
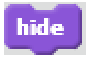
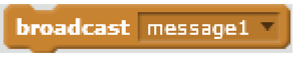


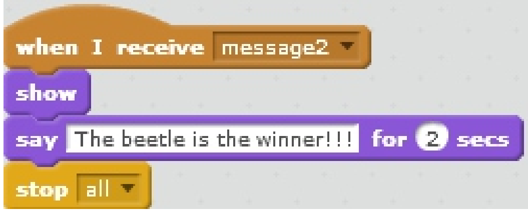
Table 9-13. Race: Ready Code Blocks

Blocks	Actions
 when ⚑ clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
 show	Make the sprite appear on the stage.
 go to x: 0 y: 0	Move the sprite to coordinates (X = 0, Y = 0), which is the middle of the stage.
 say Ready to start the race! for 2 secs	The sprite gets a speech bubble that displays Ready to start the race! for 2 seconds.
 say Ready, steady, goooo..... for 2 secs	The sprite gets a speech bubble that displays Ready, steady, gooooo…. for 2 seconds.
 hide	Make the sprite disappear from the stage.
 broadcast message1 ▾	Broadcast message1 .

Beneath the first script and with the cat still selected, create Script 9-14 , which announces the beetle as the winner upon receiving message2 . It then stops all the scripts in the project.

The script starts running after it receives message2 from Script 9-17 . The next block shows the sprite on the stage.

The third block creates a speech bubble that displays The beetle is the winner!!! for 2 seconds. The last block of code stops all scripts in this project.
Table 9-14 lists the blocks and describes the actions used in this activity.



```
when I receive message2 ▾
show
say The beetle is the winner!!! for 2 secs
stop all ▾
```

Script 9-14. Race: Announcement1

OUTPUT

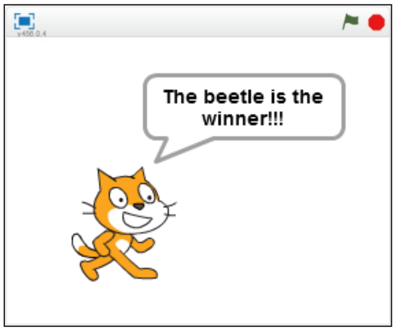
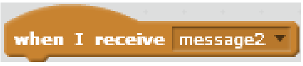
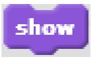
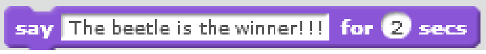
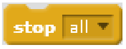


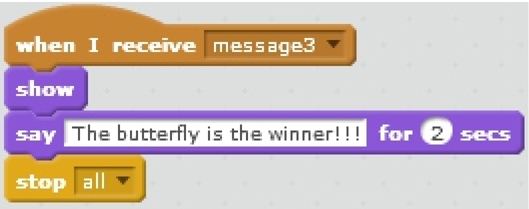
Table 9-14. Race: Announcement1 Code Blocks

Blocks	Actions
 when I receive message2 ▾	Receiving message2 activates the script.
 show	Make the sprite appear on the stage.
 say The beetle is the winner!!! for 2 secs	The sprite gets a speech bubble that displays The beetle is the winner!!! for 2 seconds.
 stop all ▾	Stop all scripts in this project from running.

The last script for Sprite1 announces the butterfly as the winner if it reaches the finish line first. Script 9-15 starts running if **message3** is received from Script 9-19 .

The next block shows the sprite on the stage, and the third creates a speech bubble that displays The butterfly is the winner!!! for 2 seconds. The last block of code stops all scripts in this project.

Table 9-15 lists the blocks and describes the actions used in this activity.



```
when I receive message3
show
say The butterfly is the winner!!! for 2 secs
stop all
```

Script 9-15. Race: Announcement2

OUTPUT

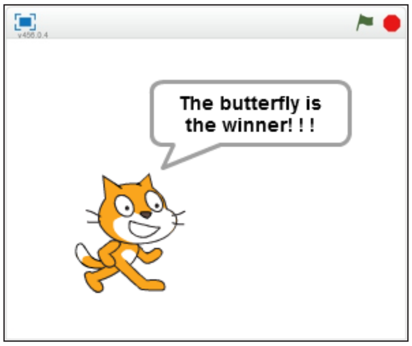
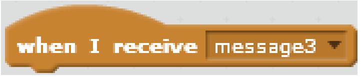
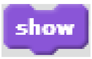
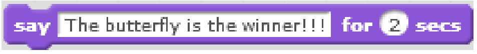
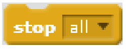


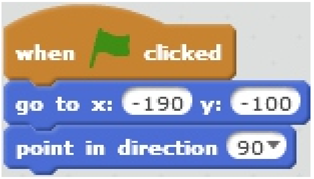
Table 9-15. Race: Announcement2 Code Blocks

Blocks	Actions
 when I receive message3	Receiving message3 activates the script.
 show	Make the sprite appear on the stage.
 say The butterfly is the winner!!! for 2 secs	The sprite gets a speech bubble that displays The butterfly is the winner!!! for 2 seconds.
 stop all	Stop all scripts in this project from running.

Next, select the beetle and create Scripts 9-16 and 9-17 , one beneath the other. This first script just positions the beetle for the start of the race.

Script 9-16 starts running when the user clicks the green flag. The next two blocks move the beetle to its starting position for the race, which is (–190, –100), and make the beetle face to the right.

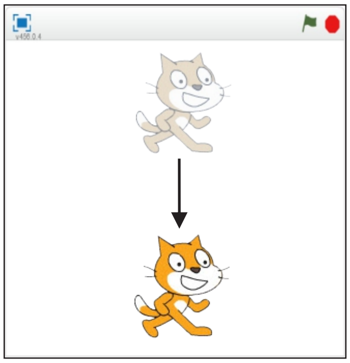
Table 9-16 lists the blocks and describes the actions used in this activity.

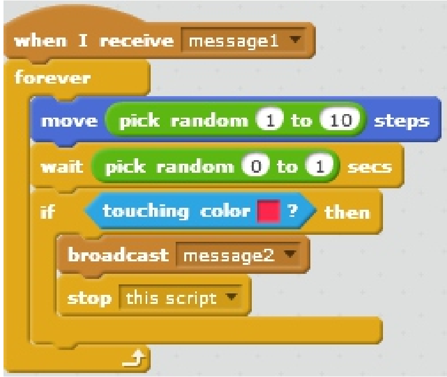


```
when 🏁 clicked
go to x: -190 y: -100
point in direction 90
```

Script 9-16. Race start Beetle

OUTPUT





```
when I receive message1
forever
  move pick random 1 to 10 steps
  wait pick random 0 to 1 secs
  if touching color 🟥 ? then
    broadcast message2
    stop this script
```

Script 9-17. Race: Race finish Beetle

OUTPUT

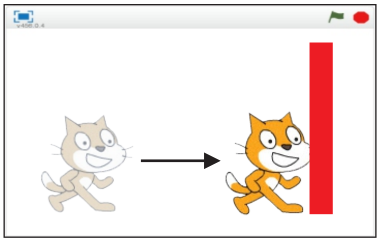
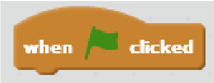
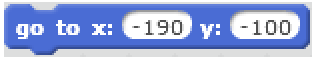
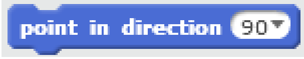


Table 9-16. Race: Race Start Beetle Code Blocks

Blocks	Actions
 when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
 go to x: -190 y: -100	Move the sprite to coordinates (X = –190, Y = –100).
 point in direction 90▾	Point the direction of the sprite to the right.

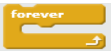
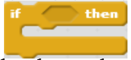
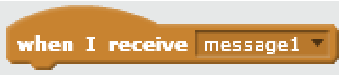
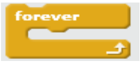
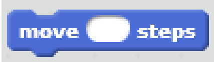
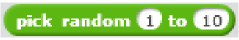
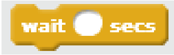
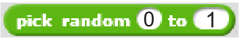
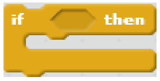
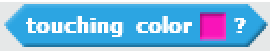
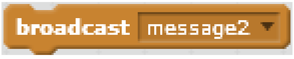
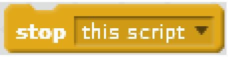
Upon receipt of message1 , Script 9-17 starts the Beetle racing toward the finish line and broadcasts message2 when it reaches the finish line. Script 9-17 starts running when it receives message1 , which is broadcast in Script 9-13 as the cat's start sign. The next block is a  forever block that repeats the sequence of actions within it forever. The first block in the sequence moves the sprite a random number of pixels (1 to 10) in the direction that it's facing. The next block pauses the script a random number of seconds between 0 and 1. Next there is the If/Then conditional statement that checks whether the sprite touches the red color, which is the color of the finish line. If it does, then the condition is true and the sequence of actions within the  if then block are executed. The first block within it broadcasts message2 to Script 9-14 , and the last block within it stops this script. **Table 9-17** lists the blocks and describes the actions used in this activity.

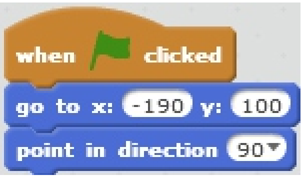
Table 9-17. Race: Race Finish Beetle Code Blocks

Blocks	Actions
 when I receive message1 ▾	Receiving message1 activates the script.
 forever	Repeat the actions represented by the blocks within this block forever, until the script is stopped manually.
 move steps	Move the sprite the amount of specified pixels in the direction that it's facing.
 pick random 1 to 10	Pick a random value between the two specified numbers and report the result.
 wait secs	The script waits a specified number of seconds. No actions are performed in that timespan.
 pick random 0 to 1	Pick a random value between the two specified numbers and report the result.
 if then	Check if the condition is true. If the condition is true, execute the actions within it. If the condition is false, skip to the next block.
 touching color ■ ?	Check if the sprite is touching the red color. If it is, then this condition is true; otherwise, it's false.
 broadcast message2 ▾	Broadcast message2 .
 stop this script ▾	Stop this script from running.

Next, select the butterfly, and create Scripts 9-18 and 9-19 , one beneath the other. As you did for the beetle, you will first create a script that positions the butterfly for the start of the race.

Script 9-18 starts running when the user clicks the green flag. The next two blocks move the butterfly to its starting position in the race, which is (–190, 100), and make the butterfly face to the right.

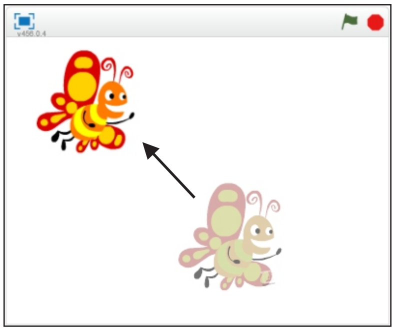
Table 9-18 lists the blocks and describes the actions used in this activity.

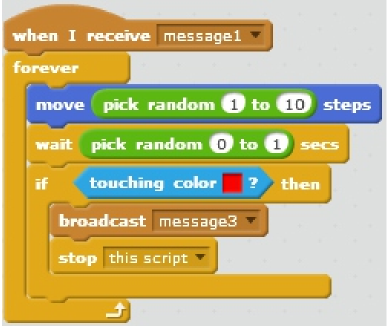


```
when ⚑ clicked
go to x: -190 y: 100
point in direction 90▾
```

Script 9-18. Race: Race start Butterfly

OUTPUT





```
when I receive message1 ▾
forever
    move pick random 1 to 10 steps
    wait pick random 0 to 1 secs
    if touching color 🟥 ? then
        broadcast message3 ▾
        stop this script ▾
```

Script 9-19. Race finish Butterfly

OUTPUT

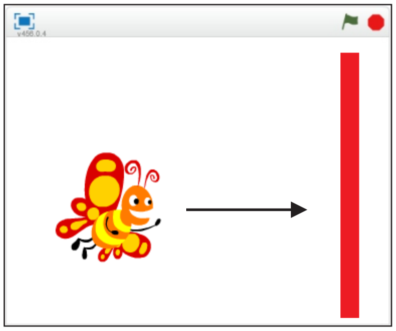
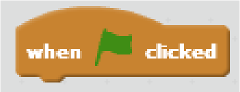
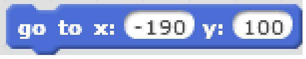
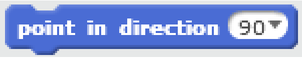
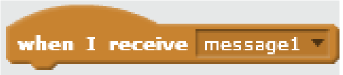
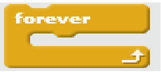
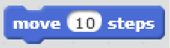


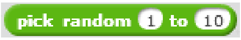
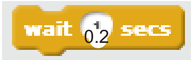
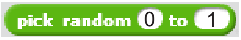
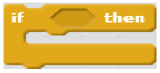
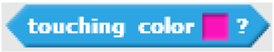
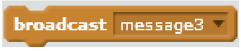
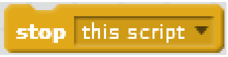
Table 9-18. Race: Race Start Butterfly Code Blocks

Blocks	Actions
 when ⚑ clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
 go to x: -190 y: 100	Move the sprite to the position where X = –190 and Y = 100.
 point in direction 90▾	Make the sprite face to the right.

The next script starts the Butterfly racing toward the finish line when it receives message1 and broadcasts message3 to when it reaches the finish line. Script 9-19 starts running when it receives message1 , which is the start sign broadcast from Sprite1 's Script 9-13 . The next block is a forever block that repeats the sequence of actions within it forever. The first block in the sequence moves the sprite a random number of pixels (1 to 10) in the direction that it's facing. The next block pauses the script a random number of seconds between 0 and 1. The if then block then uses the embedded sensing block to check whether the sprite is touching the red color (the finish line). If it is, then the condition is true and the sequence of actions within the block is executed. The first block in the sequence broadcasts message3 to trigger Script 9-16 . The second block stops the butterfly's script and forever loop. **Table 9-19** lists the blocks and describes the actions used in this activity.

Table 9-19. Race: Race Finish Butterfly Code Blocks

Blocks	Actions
 when I receive message1 ▾	Receiving message1 activates the script.
 forever	Repeat the actions represented by the blocks within this block forever, until the script is stopped manually.
 move 10 steps	Move the sprite the amount of specified pixels in the direction that it's facing.

 pick random 1 to 10	Pick a random value between the two specified numbers and report the result.
 wait 0.2 secs	The script waits a specified number of seconds. No actions are performed in that timespan.
 pick random 0 to 1	Pick a random value between the two specified numbers and report the result.
 if then	Check if the condition is true. If the condition is true, execute the actions within it. If the condition is false, skip to the next block.
 touching color ?	Check if the sprite is touching the red color. If it is, then this condition is true; otherwise, it's false.
 broadcast message3	Broadcast message3 .
 stop this script	Stop this script from running.

Notice that multiple messages are used in this activity. Each message that is broadcasted and received executes different actions and produces different results. This activity shows that it would be very difficult to create this project without the broadcasting and receiving blocks. Using the broadcasting block, one script sends a message to activate two scripts; whichever sprite wins the race has a script that broadcasts a message to announce the winner.

Activity 9-6: Scenery Change

So far, the example scripts have set up interactions between sprites. This project shows you how you can trigger a backdrop change by broadcasting a message from a sprite's script. Before creating this project, add two backdrops— **desert** and **slopes** —to the Backdrops pane from the Scratch library (see Figure 9-12). You can keep the default **backdrop1** or you can delete it. You'll also need two messages called **message1** and **message2**.

Select the sprite, and create Script 9-20 . The sprite tells a story, and then the script broadcasts message1 to Script 9-21 , which changes the backdrop.

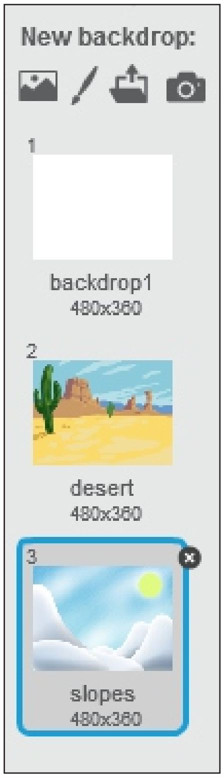
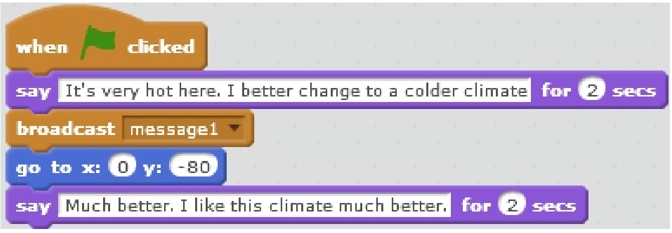


Figure 9-12. Backdrops



```
when [green flag] clicked
say It's very hot here. I better change to a colder climate for 2 secs
broadcast message1
go to x: 0 y: -80
say Much better. I like this climate much better. for 2 secs
```

Script 9-20. Scenery Change: story

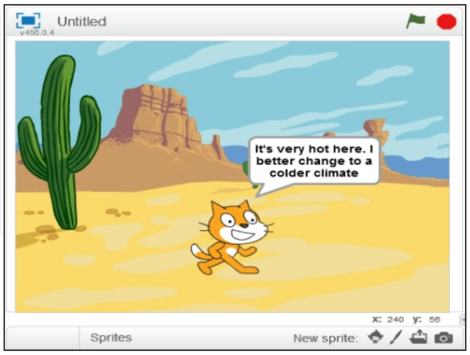
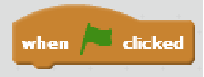
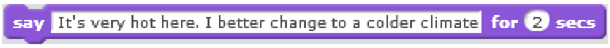
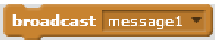
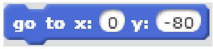
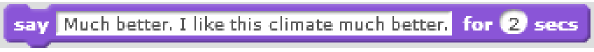


Figure 9-13. Scenery change

Script 9-20 starts when the user clicks the green flag. The next block creates a speech bubble that displays It's very hot here. I better change to a colder climate. for 2 seconds (see Figure 9-13). The next block broadcasts message1 to trigger Script 9-22 and the backdrop change. The next block moves the sprite to (0, –180) on the stage. The last block of code creates a speech bubble that displays Much better. I like this climate much better. for 2 seconds. **Table 9-20** lists the blocks and describes the actions used in this activity.

Table 9-20. Scenery Change: Story Code Blocks

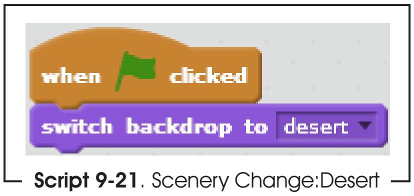
Blocks	Actions
 when [flag] clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
 say It's very hot here. I better change to a colder climate for 2 secs	The sprite gets a speech bubble that displays It's very hot here. I better change to a colder climate. for 2 seconds.
 broadcast message1	Broadcast message1 .
 go to x: 0 y: -80	Move the sprite to the position where X = 0 and Y = –80.
 say Much better. I like this climate much better. for 2 secs	The sprite gets a speech bubble that displays Much better. I like this climate much better. for 2 seconds.

Now for the backdrops' scripts. Select a backdrop in the Backdrops pane; it doesn't matter which one is showing as a thumbnail, just select one (see Figure 9-14).

Next, create Script 9-21 , which sets the starting backdrop for the project. Script 9-21 starts running when the user clicks the green flag.

The next block switches the backdrop to desert . If the backdrop of the stage is already set to desert , then nothing happens.

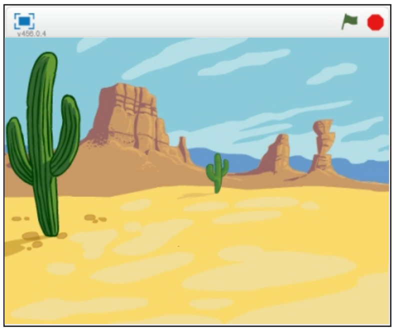
Table 9-21 lists the blocks and describes the actions used in this activity.



```
when [flag] clicked
switch backdrop to desert
```

Script 9-21. Scenery Change:Desert

OUTPUT



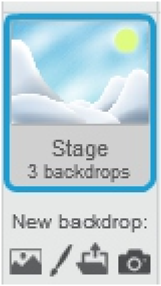
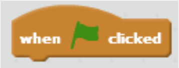
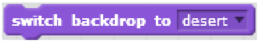
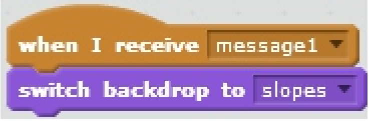


Figure 9-14. Backdrop thumbnail

Table 9-21. Scenery Change: Desert Code Blocks

Blocks	Actions
 when [flag] clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
 switch backdrop to desert	Switch the backdrop of the stage area to desert .

With the backdrop thumbnail still selected in the Backdrops pane, create Script 9-22 beneath the previous script. Script 9-22 starts running as soon as it receives message1 from Script 9-20 . The next block switches the backdrop to slopes.

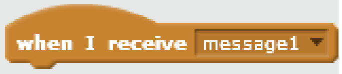
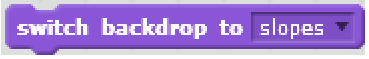


```
when I receive message1
switch backdrop to slopes
```

Script 9-22. Scenery Change: Slopes

Table 9-22 lists the blocks and describes the actions used in this activity.

Table 9-22. Scenery Change: Slopes Code Blocks

Blocks	Actions
 when I receive message1	Receiving message1 activates the script.
 switch backdrop to slopes	Switch the backdrop of the stage area to slopes .

Although this project doesn't illustrate it, a sprite can also receive a message from a backdrop. As an extra challenge, modify this activity so that instead of the sprite's script broadcasting a message to the backdrop, the backdrop's script broadcasts a message to the sprite's script to trigger a specific action.

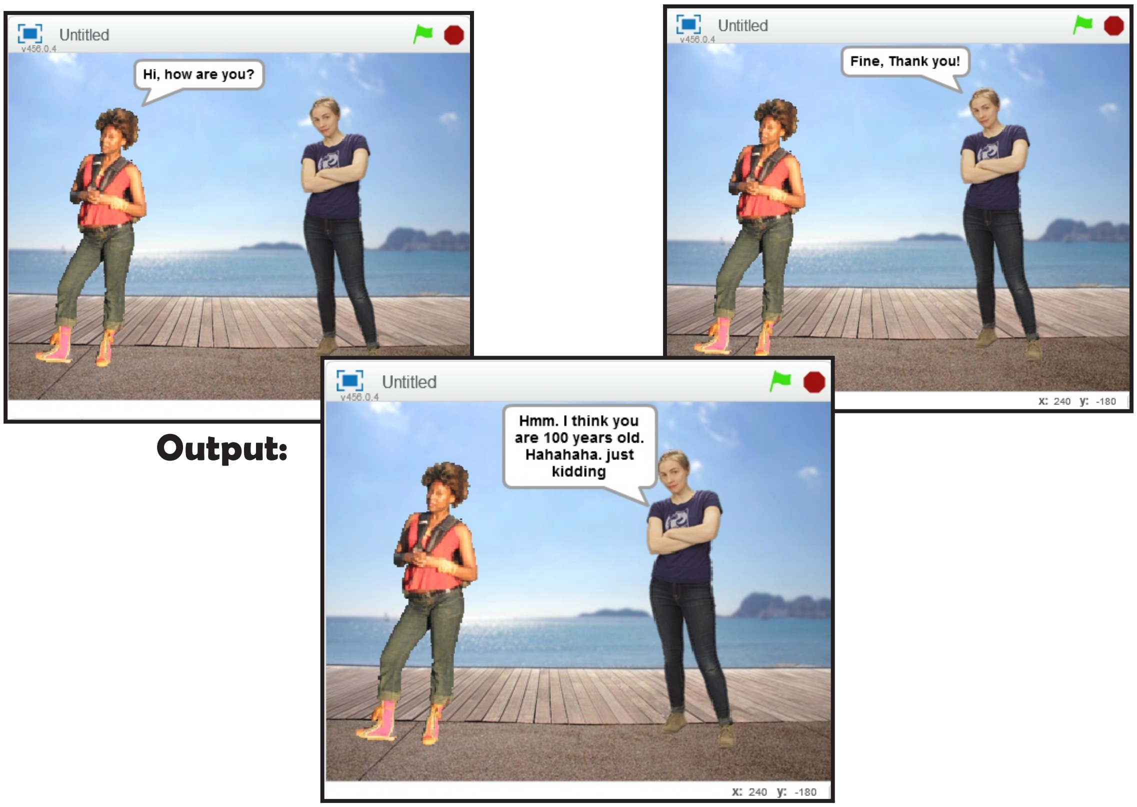
Summary

Now you know how to broadcast messages to trigger scripts. You can make your sprites interact with each other and with backdrops to tell your own stories—with or without user interaction. This is very useful if you want to create a Scratch project that tells a story for a game, movie, or simulation. You have gathered so much knowledge about Scratch that you can create some great projects. In the next chapter, I will show you how to create your own blocks of code.

Snap Script

a Short hands-on activity

1. Using the broadcast and receive blocks, create a project that shows two sprites having a conversation with each other.



Hi, how are you?

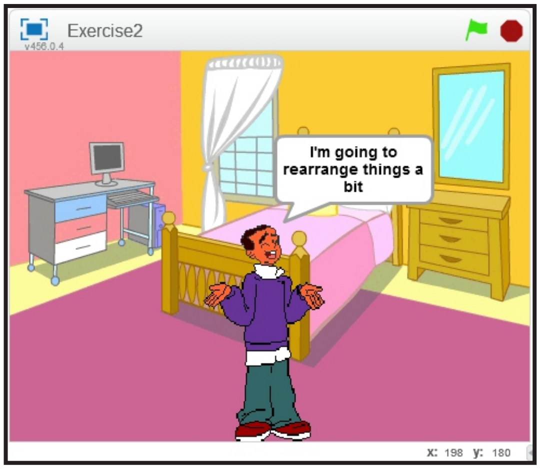
Fine, Thank you!

Hmm. I think you are 100 years old. Hahahaha. just kidding

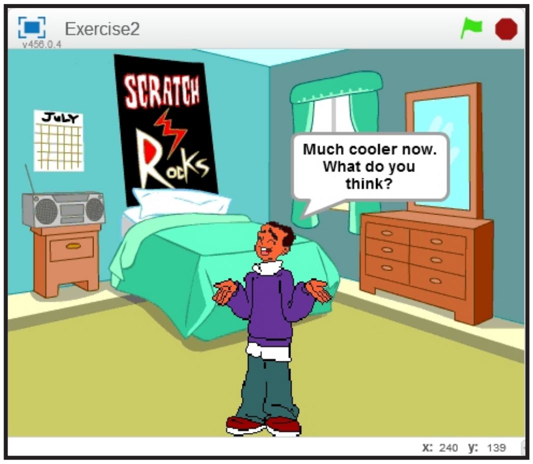
Output:

2. Using the broadcast and receive blocks, create a project that switches backdrops.

Output:



I'm going to rearrange things a bit



Much cooler now. What do you think?