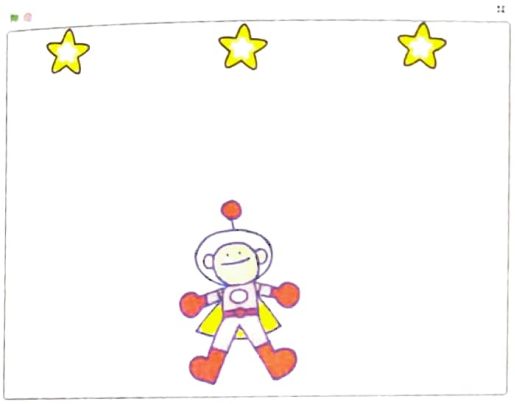
Activity 3—Dodge the Stars

LEVEL UP!

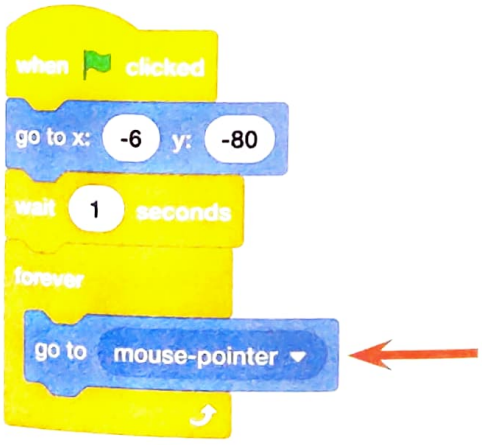
Ripley the astronaut has to watch out for falling stars when he's flying through space. Can you use some loops to help him out?

STEP 1



Bring in the Ripley sprite and center him on the bottom of the stage. Then bring in the Star sprite three times so that you get three different star sprites. (You can also do this by right-clicking the star icon and choosing *duplicate*.) Arrange them in a row across the top of the stage.

STEP 2

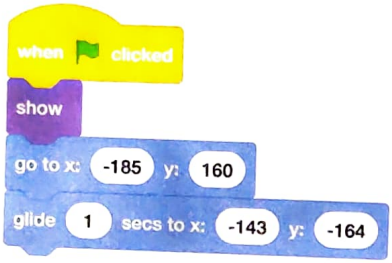


Go to Ripley's script editor. Start the script with a *when flag clicked* event block, and then add a *go to x: (__) y: (__)* motion block to set his start position. Add a *wait (1) seconds* control block.

We can do something really cool now—we can make Ripley follow our mouse *wherever* it goes! All you need to do is add a *forever* control block and place a *go to (random position)* motion block inside it. But instead of going to random positions, click the little arrow in the motion block to choose "mouse-pointer."

Now try running your script. Experiment with taking out the *wait (1) seconds* control block. See how Ripley gets stuck in the top-left corner at first? That's because the program starts the instant you click the flag. And where is your mouse when you click the flag? Yep, that's why Ripley pops up in that corner! Using the wait gives a smoother start to your game.

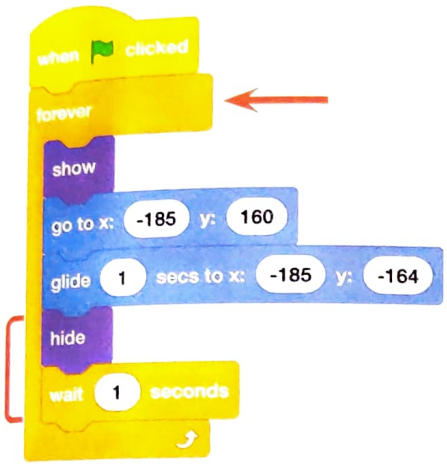
STEP 3



Go to the first star's script editor. Start your script with a *when flag clicked* event block, add a *show* looks block to make it visible, and then add a *go to x: (__) y: (__)* motion block to give it a starting point. To make the star actually fall, click and drag the star straight down until it's at the bottom of the screen. That will update your motion code blocks with the new coordinates. Then add a *glide (__) secs to x: (__) y: (__)* motion block to your script.

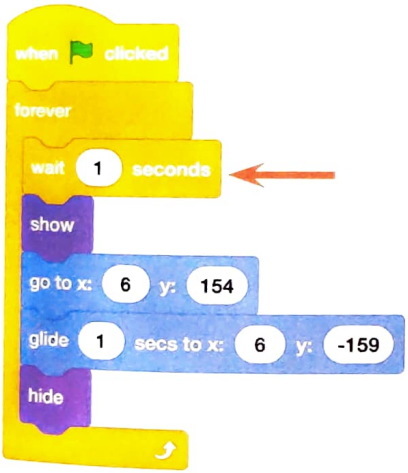
5

STEP 4



To make the star look like it has fallen *past* Ripley, add a *hide* looks block. Then add a *wait (__) seconds* control block. Finally, place all of the code inside a *forever* control block so that it will constantly repeat. That wait we added helps make it look like it's a *new* star falling from the top of the screen. Without the wait, it looks like the same star is just jumping back up to the top of the screen.

STEP 5



Go to Star2's script editor. You can build your code for Star2 the exact same way, with one small adjustment. Since Star2 is right above Ripley's head, move the *wait (__) seconds* control block to the top of the script instead of the bottom. That way, the user has enough time to move Ripley out of the way before the star begins to fall. This will also change the timing for Star2 so it feels more random. Otherwise, it will fall down right next to the first star. If all three stars fall down in a straight line, Ripley never gets a chance to dodge them!

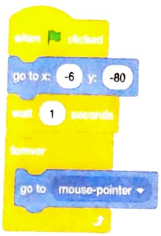
STEP 6: CODE COMPLETE!

Can you figure out how to code your last star? Try following the same strategy we used for the other stars, but this time change the *wait (__) seconds* control block to two seconds so Star3 will have different timing from the other two. This should make the falling stars feel more random.

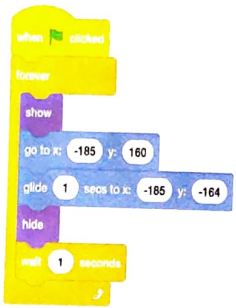
If you're quick with your mouse, you can make Ripley an expert star dodger!

5

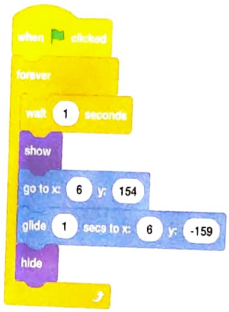
RIPLEY'S CODE



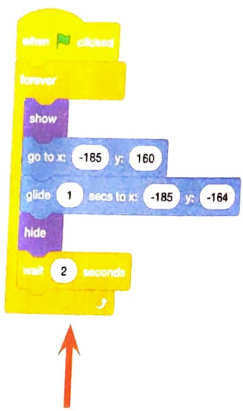
STAR CODE



STAR2 CODE



STAR3 CODE



YOUR TURN!

Now that you've coded this activity, try challenging yourself to make the following changes:

- ☐ Adjust your waits to make the star movements even more random.
- ☐ Add more sprites for Ripley to dodge.
- ☐ Make one of the new sprites shoot across the stage from left to right instead of falling from the top of the screen.

Loops Off-Screen

Did you know that you use loops all the time in your normal daily life? Think about it! You're constantly doing things over and over without thinking about every little step. For example, think about when you are eating popcorn at the movie theater. If you were a computer, you would use the following directions: Pick up a piece of popcorn, place in your mouth, chew, swallow. And repeat!

Exploring More!

Loops are great for saving you time and making your code much shorter. But they are also really important for setting up situations where the computer can always check for a certain condition, or rule, that you make. For example, if you create a rule that you should score a point every time the ball sprite goes into the basket, then you will need a forever control block around that code to tell the computer to constantly check for that condition.

It's a good thing you understand loops now because we will need them in our next chapter, when we learn how to use variables!