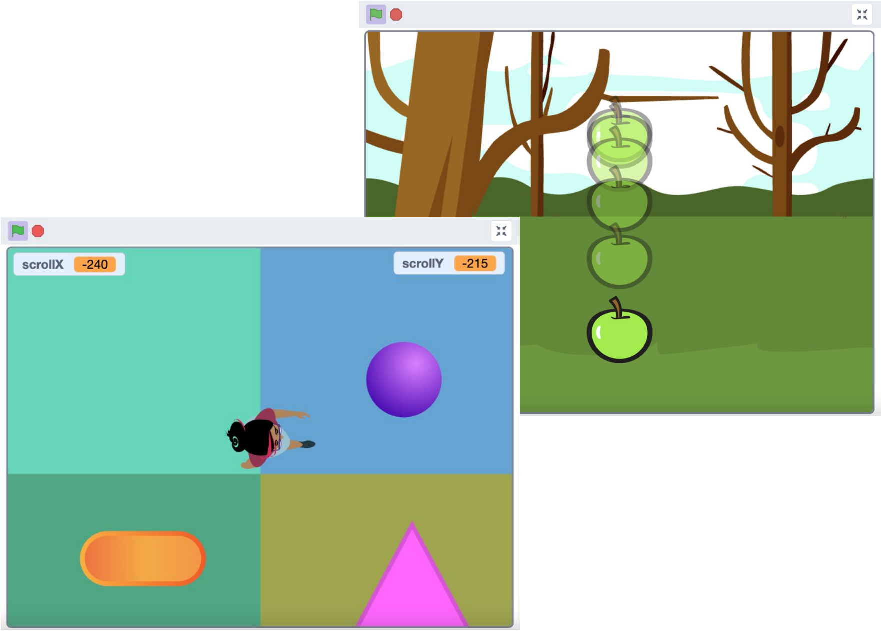
Advanced Topics: Graphic Effects

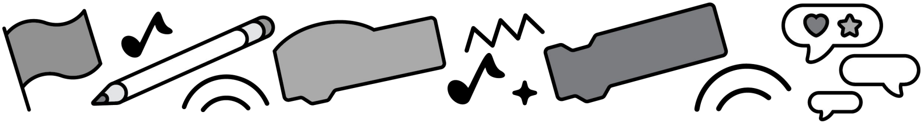


Use data to create more advanced scripts for games, stories, and more!

scratch.mit.edu

SCRATCH

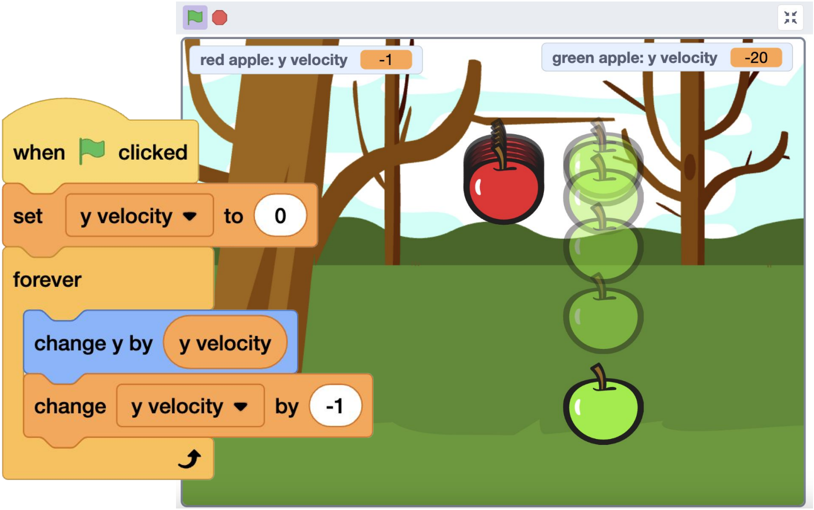
Set of 5 cards



Cards in This Pack

- Gravity in Scratch
- Gravity in Scratch: Stop or Bounce
- Gravity in Scratch: Reverse Gravity
- User-Controlled Scrolling Background
- Text Generator/Text Rendering [COMING SOON]

Gravity in Scratch



red apple: y velocity -1

green apple: y velocity -20

```
when ⚑ clicked
set y velocity ▾ to 0
forever
    change y by (y velocity)
    change y velocity ▾ by -1
```

Sir Isaac Newton is said to have discovered universal gravitation by observing the fall of an apple.

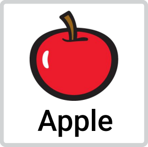
Creating gravity in Scratch means sprites fall to the bottom of the stage in a realistic way. **Velocity** is the speed of something in a given direction. In order to fall in a realistic-looking way, they should **accelerate** (gain speed as they fall). *(Example here: scratch.mit.edu/projects/964424785.)*

Gravity in Scratch

scratch.mit.edu

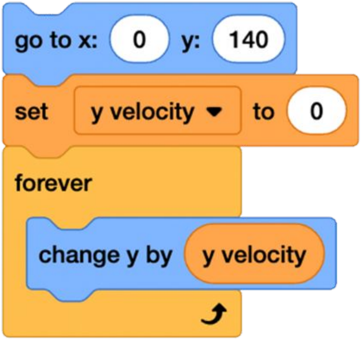
1. Create a variable that will hold the velocity of your sprite, like “y velocity.”

Choose a sprite.



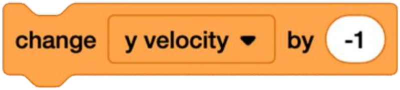
2. Set the velocity to zero at the start (so the object will start from a place of rest).

3. Then, have the sprite move (in this case, change y) by that velocity.

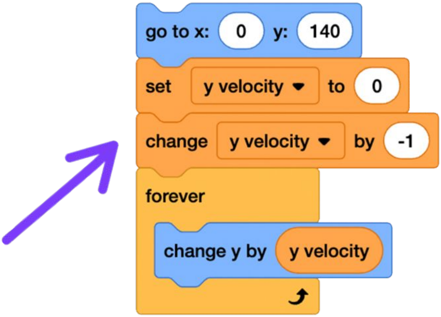


```
go to x: 0 y: 140
set y velocity to 0
forever
    change y by y velocity
```

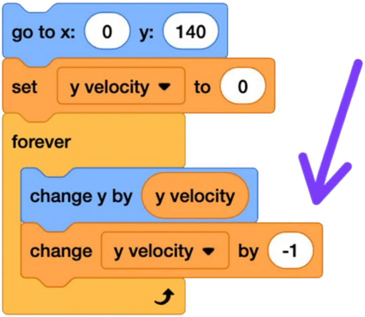
4. In order to gain speed (acceleration), the velocity will have to change. What is different if that change happens inside or outside your loop?



```
change y velocity by -1
```



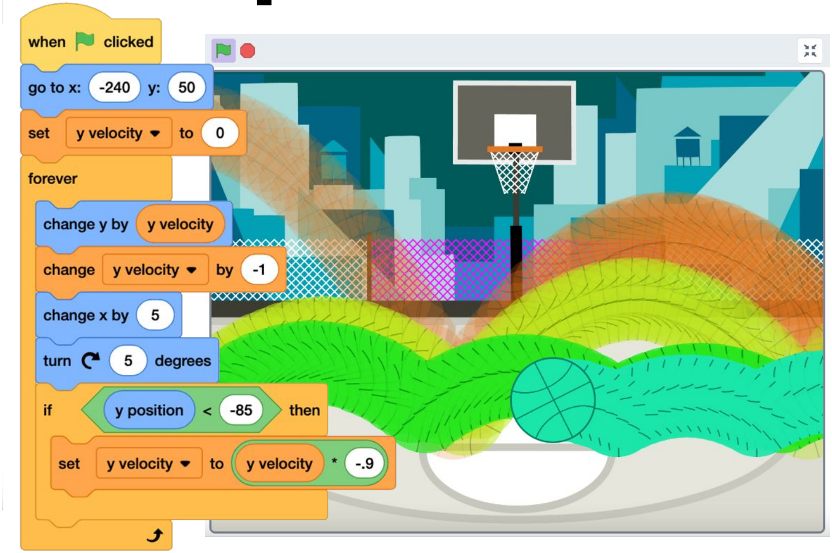
```
go to x: 0 y: 140
set y velocity to 0
change y velocity by -1
forever
    change y by y velocity
```



```
go to x: 0 y: 140
set y velocity to 0
forever
    change y by y velocity
    change y velocity by -1
```

Is it just moving down at a constant rate? Or is it also speeding up over time, adding realism?

Gravity in Scratch: Stop or Bounce



```
when ⚑ clicked
go to x: -240 y: 50
set y velocity ▾ to 0
forever
    change y by (y velocity)
    change y velocity ▾ by -1
    change x by 5
    turn ↻ 5 degrees
    if <(y position) < -85> then
        set y velocity ▾ to ((y velocity) * -.9)
```

When creating gravity in Scratch, you might also think about how the object should react when it hits the ground.

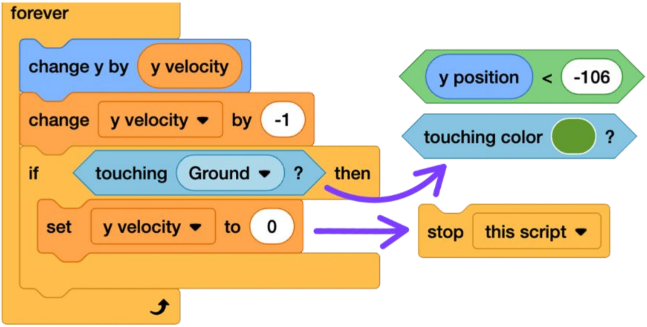
The object could come to a complete stop. Or sprites may bounce when they reach the ground (or the bottom of the stage), with those bounces getting smaller and smaller over time.

(Example here: scratch.mit.edu/projects/964194371.)

Gravity: Stop or Bounce

scratch.mit.edu

1. To stop the sprite from falling through to the bottom of the stage, you'll need to set up a condition that tells the program when to stop the sprite. You can choose from many conditions, like the position, touching a sprite, etc.

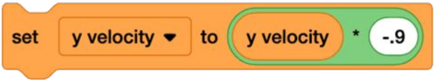


```
forever
    change y by (y velocity)
    change [y velocity] by (-1)
    if <touching [Ground] ?> then
        set [y velocity] to (0)
    end
end
```

```
<(y position) < (-106)>
<touching color [ ] ?>
stop [this script]
```

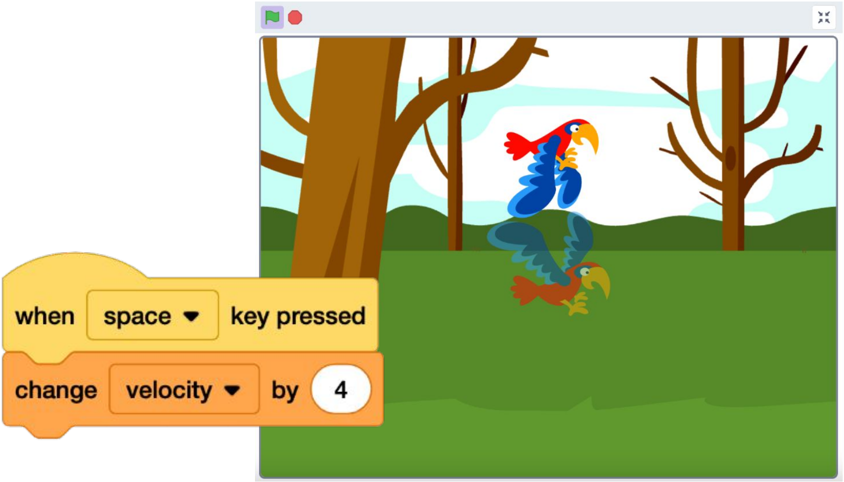
2. When the condition has been met, you could stop the script completely or stop the movement by setting the velocity to zero. This represents an inelastic or "sticky" collision, where the kinetic energy is absorbed by its surroundings and the object comes to an immediate stop.

3. In an elastic or "bouncy" collision the kinetic energy (and, thus, the bounces) gets smaller over time. Experiment to recreate this by setting the velocity to the velocity times a negative number, like - 0.9. A negative multiplied by a negative is a positive. A positive velocity means the object will move up, until the velocity becomes negative again and it starts to fall. As the loop repeats, the velocity when it hits the "ground" becomes smaller and smaller.



```
set [y velocity] to ((y velocity) * (-.9))
```

Gravity in Scratch: Reverse Gravity



```
when [space ▾] key pressed
change [velocity ▾] by (4)
```

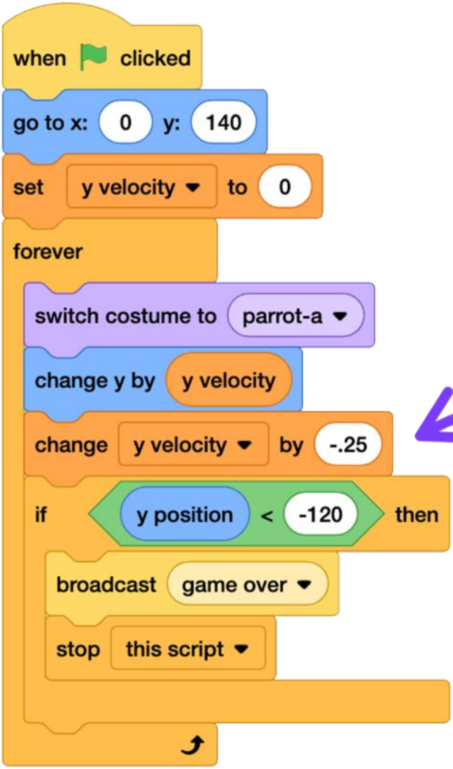
What if you wanted to, at least temporarily, reverse gravity? If you made a sprite bounce, you reversed gravity. Another example is in the case of a “flappy bird”-style game, where pressing a keyboard key temporarily reverses gravity and stops a sprite from hitting other objects or the ground in order complete the objectives of a game.

(Example here: scratch.mit.edu/projects/963989317.)

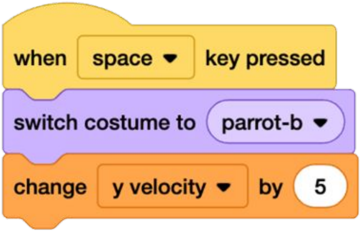
Gravity: Reverse Gravity

scratch.mit.edu

1. Create a script that sets your sprite to fall with gravity.
2. Then, add a second script so when, for instance, a keyboard key is pressed, the velocity will change by a larger positive number. (Versus the smaller negative number in the gravity script.)



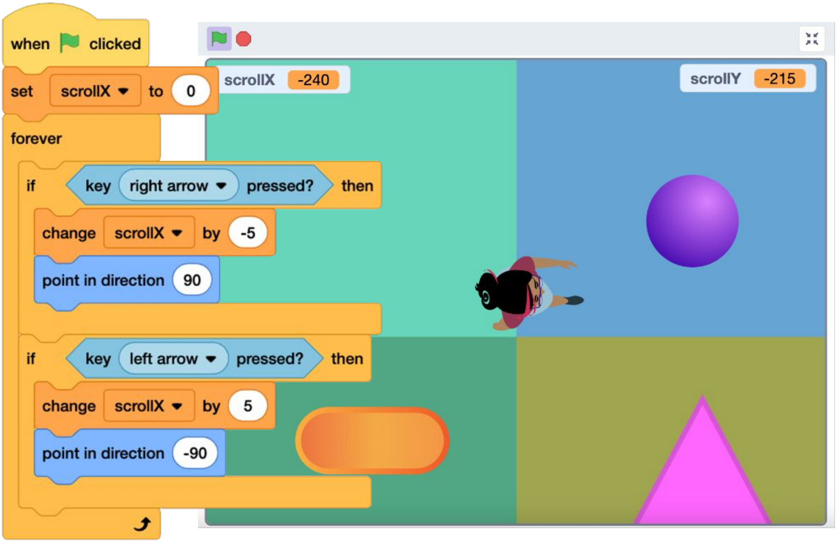
```
when 🏴 clicked
go to x: (0) y: (140)
set [y velocity ▾] to (0)
forever
    switch costume to (parrot-a ▾)
    change y by (y velocity)
    change [y velocity ▾] by (-.25)
    if <(y position) < (-120)> then
        broadcast (game over ▾)
        stop [this script ▾]
```



```
when [space ▾] key pressed
switch costume to (parrot-b ▾)
change [y velocity ▾] by (5)
```

Test and experiment with the numbers. For a moment, the velocity should be smaller, becoming positive if you hit the key enough times, momentarily slowing or reversing the gravity effect.

User-Controlled Scrolling Background



```
when ⚑ clicked
set scrollX ▾ to 0
forever
    if key right arrow ▾ pressed? then
        change scrollX ▾ by -5
        point in direction 90
    if key left arrow ▾ pressed? then
        change scrollX ▾ by 5
        point in direction -90
↺
```

scrollX -240

scrollY -215

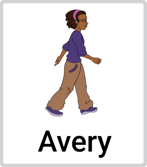
You can create a version of a scrolling background, where the background moves while the sprite stays in place. And users can use keyboard keys to control the scrolling/the position of the backgrounds in order to explore the scene! Scrolling backgrounds like this could be used in games or animations or informational projects.

User-Controlled Scrolling Background

scratch.mit.edu

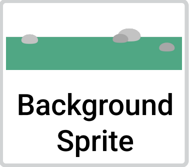
GET READY

Choose a sprite, perhaps with costumes to show movement.



Avery

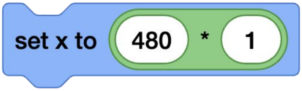
Draw a few backgrounds as sprites.



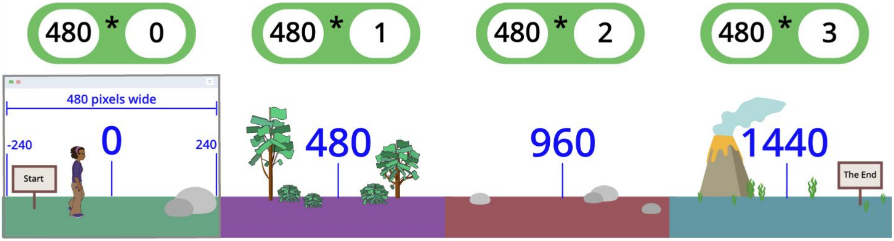
Background Sprite

ADD CODE

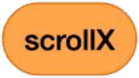
1. To line up all the background sprites end-to-end, knowing that the stage is 480 pixels wide, you can use a mathematical expression to quickly position each sprite. Set the x-position of the first background at 480 times zero (480 x 0 = 0). The next at 480 times one, etc.



```
set x to (480) * (1)
```



2. Now, create a variable that will allow you to change the position of all the background sprites simultaneously.



```
scrollX
```

(Continue to the next card.)

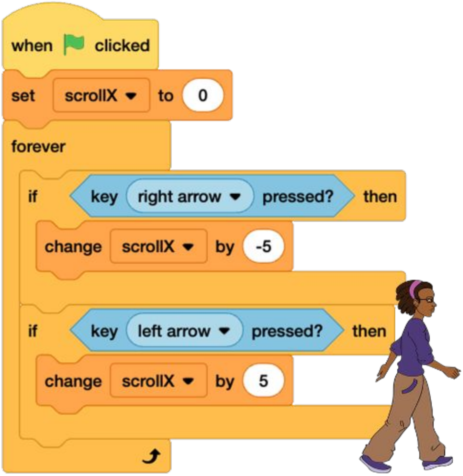
User-Controlled Scrolling Background

scratch.mit.edu

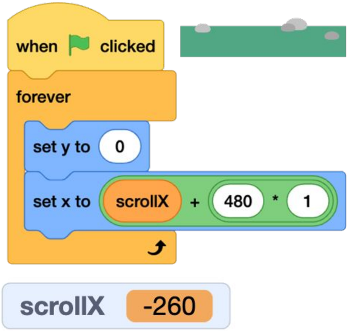
3. On the main walking sprite, create a script that changes the variable (scrollX) by a positive or negative amount when keys are pressed.

4. Then, on the background sprites, adjust the x position to add the variable, so backgrounds move simultaneously.

5. Now, you can add features and test for unexpected behavior!

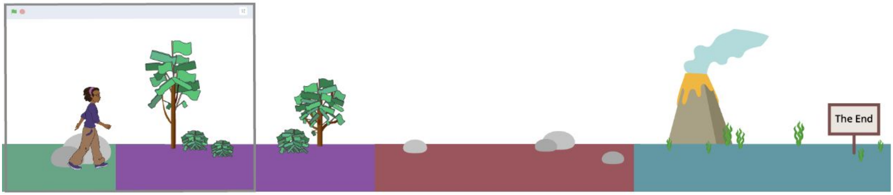


```
when ⚑ clicked
set scrollX ▾ to 0
forever
    if key right arrow ▾ pressed? then
        change scrollX ▾ by -5
    if key left arrow ▾ pressed? then
        change scrollX ▾ by 5
```



```
when ⚑ clicked
forever
    set y to 0
    set x to scrollX + 480 * 1
```

scrollX -260



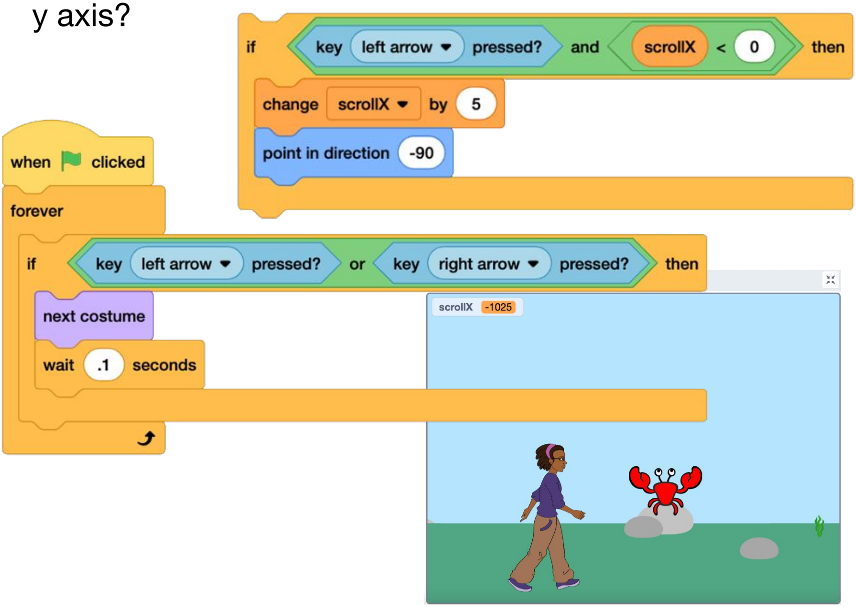
(Example here: scratch.mit.edu/projects/969838240.)

User-Controlled Scrolling Background

scratch.mit.edu

EXPERIMENT

- What if you want the sprite to look like it is walking by cycling through the costumes?
- What if you want to ensure that the walking sprite can't go off the edge of the first or last background sprite?
- What if you want to add other sprites to interact with your walking sprite?
- What if you want to move your sprite along the x and the y axis?



```
if <<key (left arrow ▾) pressed?> and <(scrollX) < (0)>> then
    change [scrollX ▾] by (5)
    point in direction (-90)
```

```
when ⚑ clicked
forever
    if <<key (left arrow ▾) pressed?> or <key (right arrow ▾) pressed?>> then
        next costume
        wait (.1) seconds
```

(Examples: scratch.mit.edu/projects/970482174 and 970438551)