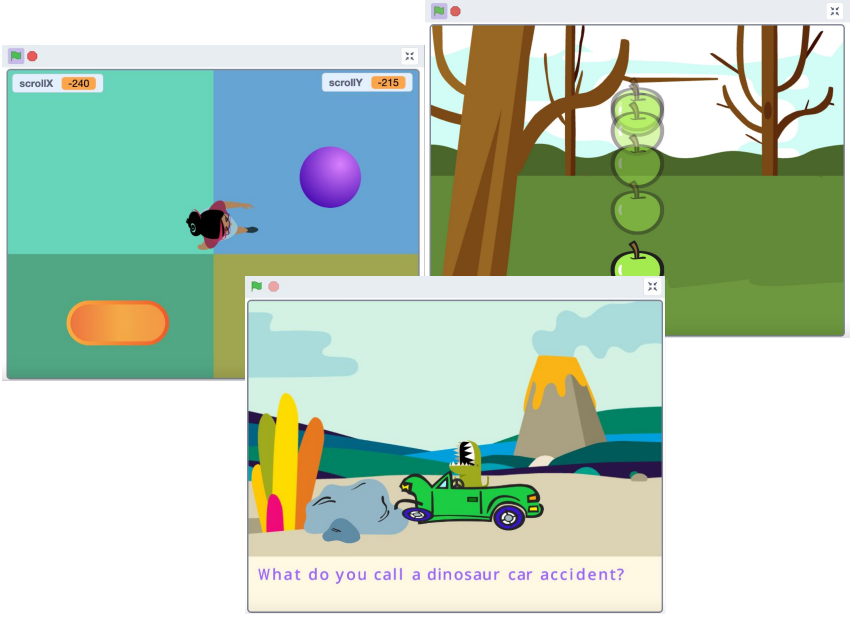
# Advanced Topics: Graphic Effects

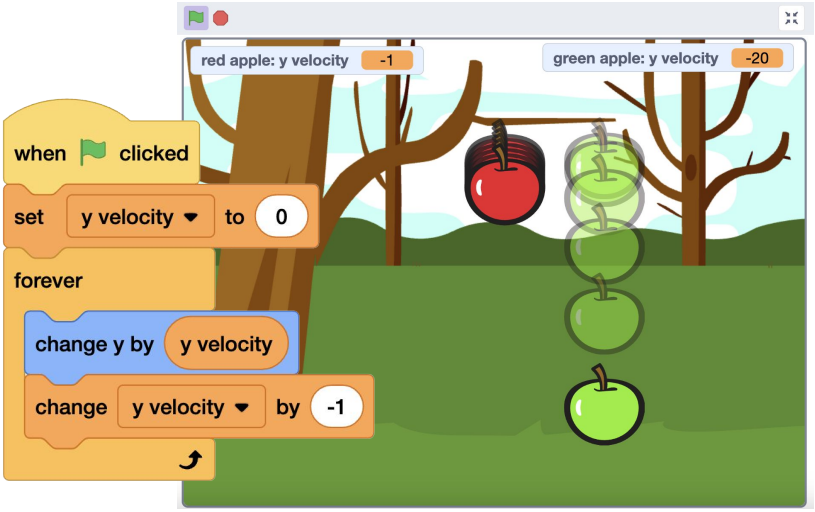



**Use data to create more advanced scripts for games, stories, and more!**

# Cards in This Pack

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

# Gravity in Scratch




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

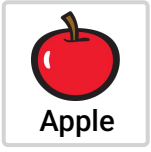
Creating gravity in Scratch means sprites fall to the bottom of the stage in a realistic way. **Velocity** is the speed of something in a given direction. In order to fall in a realistic-looking way, they should **accelerate** (gain speed as they fall). *(Example here: scratch.mit.edu/projects/964424785.)*

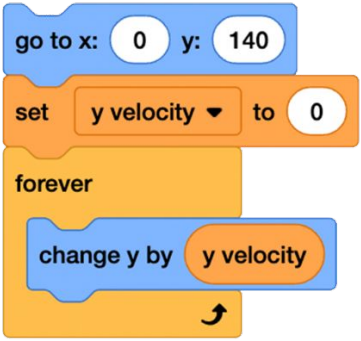
# Gravity in Scratch

scratch.mit.edu

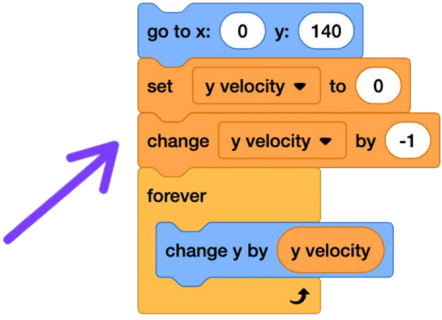
1. Create a variable that will hold the velocity of your sprite, like “y velocity.”
2. Set the velocity to zero at the start (so the object will start from a place of rest).
3. Then, have the sprite move (in this case, change y) by that velocity.
4. In order to gain speed (acceleration), the velocity will have to change. What is different if that change happens inside or outside your loop?

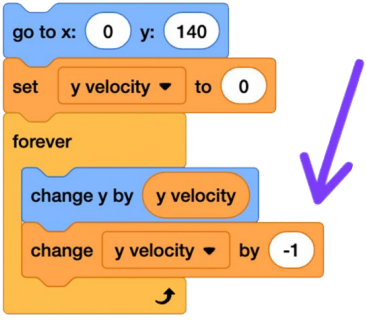
Choose a sprite.



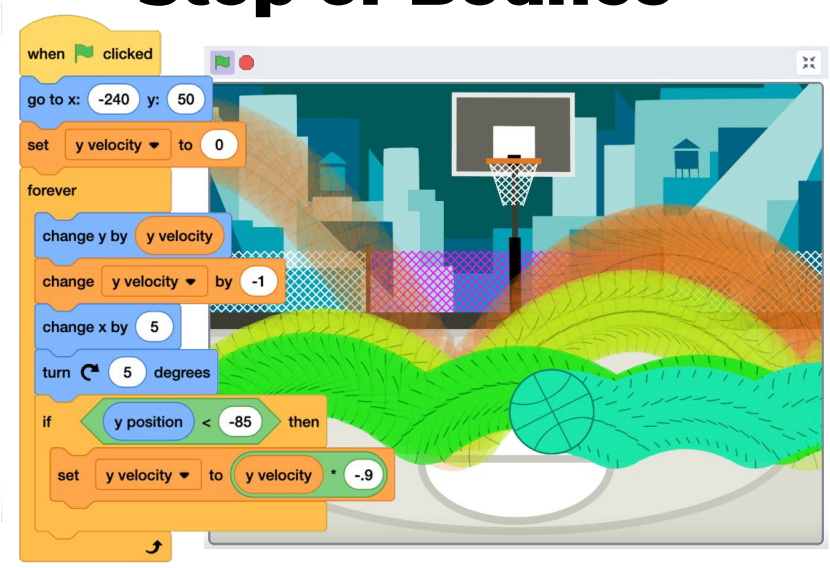





Is it just moving down at a constant rate? Or is it also speeding up over time, adding realism?

# Gravity in Scratch: Stop or Bounce




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

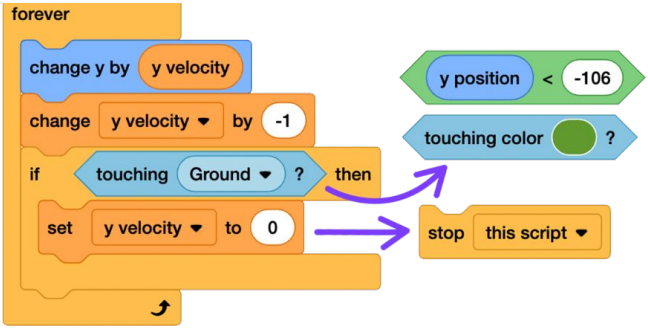
The object could come to a complete stop. Or sprites may bounce when they reach the ground (or the bottom of the stage), with those bounces getting smaller and smaller over time.

*(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.




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.

# Gravity in Scratch: Reverse Gravity

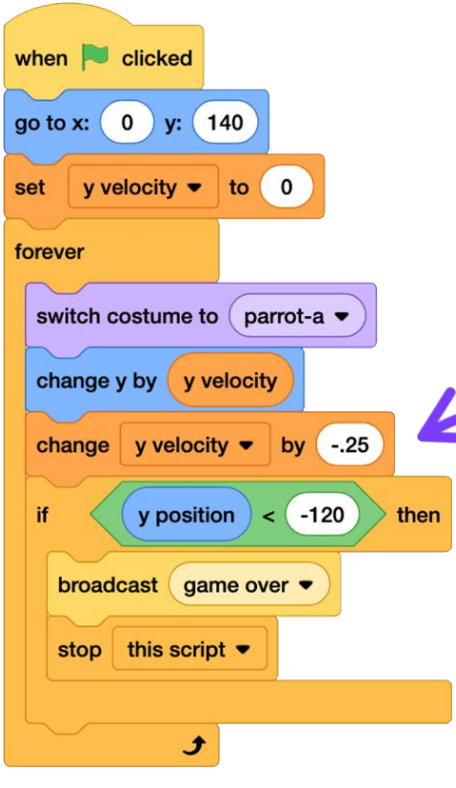

What if you wanted to, at least temporarily, reverse gravity? If you made a sprite bounce, you reversed gravity. Another example is in the case of a “flappy bird”-style game, where pressing a keyboard key temporarily reverses gravity and stops a sprite from hitting other objects or the ground in order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

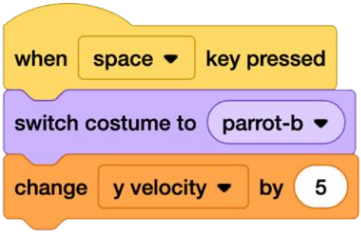
*(Example here: scratch.mit.edu/projects/963989317.)*

# Gravity: Reverse Gravity

scratch.mit.edu

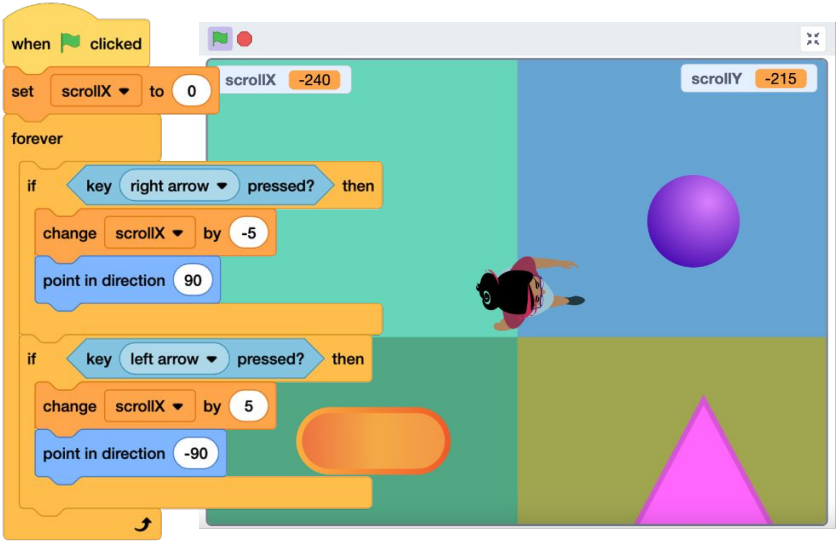
1. Create a script that sets your sprite to fall with gravity.
2. Then, add a second script so when, for instance, a keyboard key is pressed, the velocity will change by a larger positive number. (Versus the smaller negative number in the gravity script.)




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




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

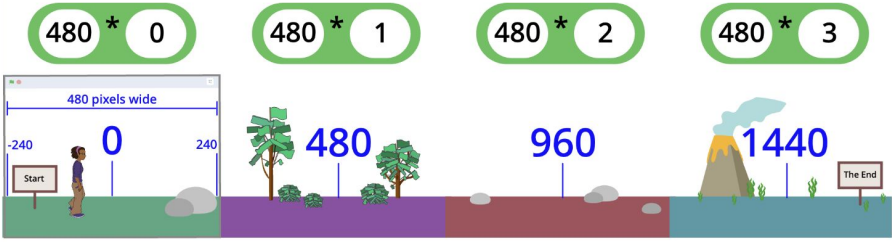
## GET READY

Choose a sprite, perhaps with costumes to show movement.


Draw a few backgrounds as sprites.


## 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.


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


*(Continue to the next card.)*

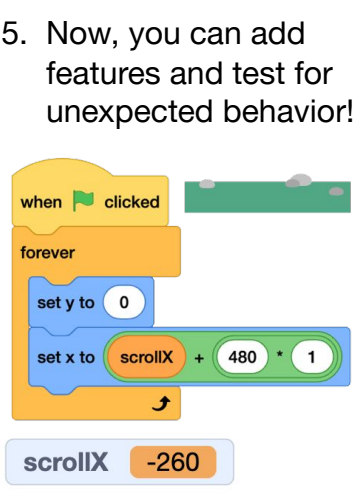
# User-Controlled Scrolling Background

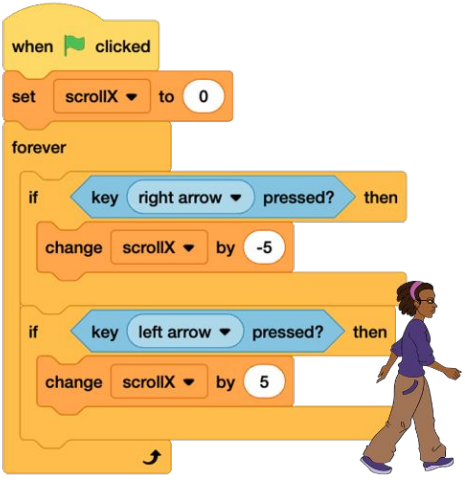
scratch.mit.edu

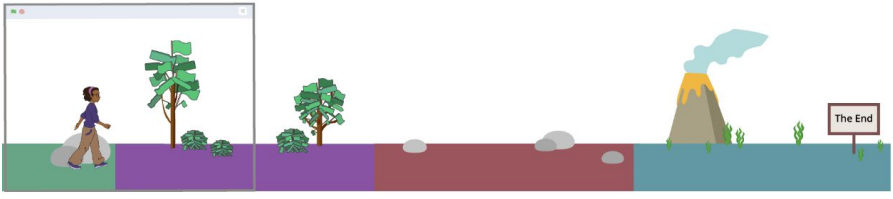
3. On the main walking sprite, create a script that changes the variable (scrollX) by a positive or negative amount when keys are pressed.

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!



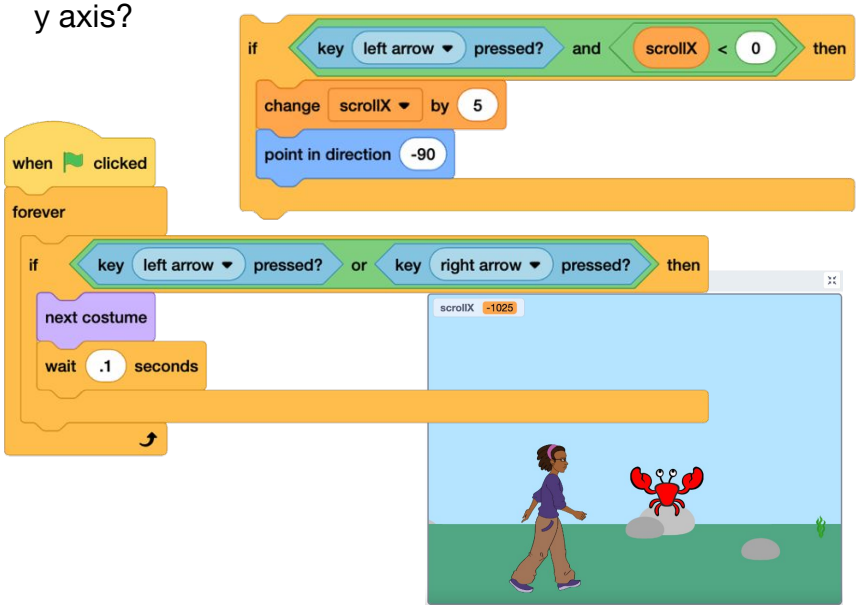

*(Example here: scratch.mit.edu/projects/969838240.)*

# User-Controlled Scrolling Background

scratch.mit.edu

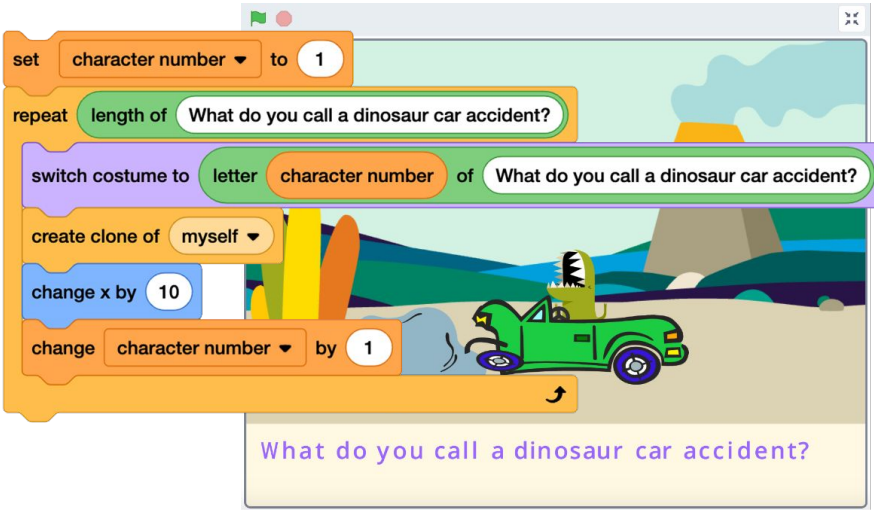
## EXPERIMENT!

- 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?




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

# Text Rendering / Text Generator




Do you want to create your own text generator to display changing text on the stage without creating sprite costumes for each line of text? Let's explore how!

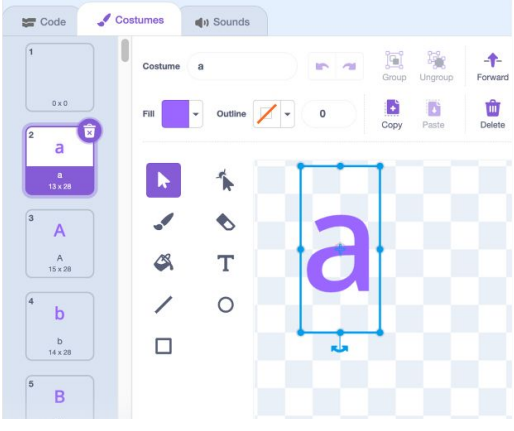
Then, you can customize the text, add effects, and more!

*(Example here: scratch.mit.edu/projects/985124543.)*

# Text Rendering

scratch.mit.edu

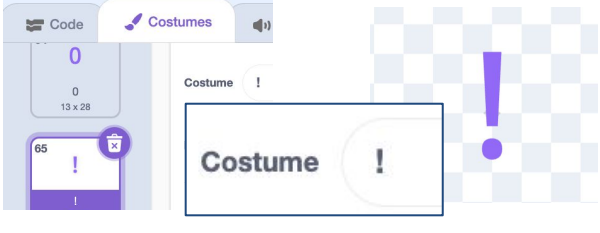
## GET READY

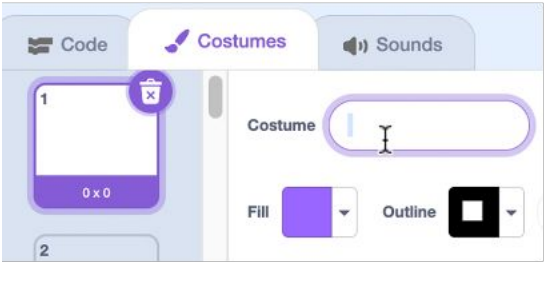



To start, create a sprite that has a costume for each letter in the alphabet (upper and lowercase), numbers 0-9, and punctuation marks you'll need.*

Choose any font or color. Center each.

The name of the costume should match the contents exactly.



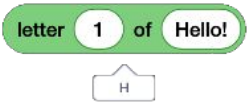




Important: Make sure your first costume is blank, but name it with a space. This will help the program recognize spaces when they appear.

**See the back of card 9 for additional letter sprite creation options.*

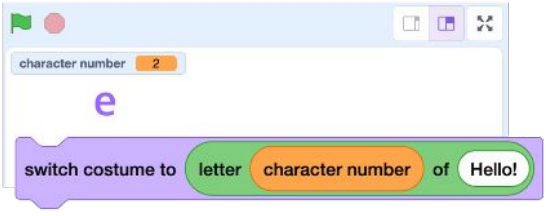
## ADD CODE

1. Locate the “letter of” reporter block. Enter some text, then click on the block. In programming, a sequence of characters that represents text is called a “string.” Strings can be used to store human-readable data, such as words or sentences. Experiment with the number and text string to see what this block reports.




2. Then, place the reporter block inside a “switch costume” block. If your costume list has a letter, number, or punctuation mark that matches the letter reported by this block, you should see it appear on the stage.

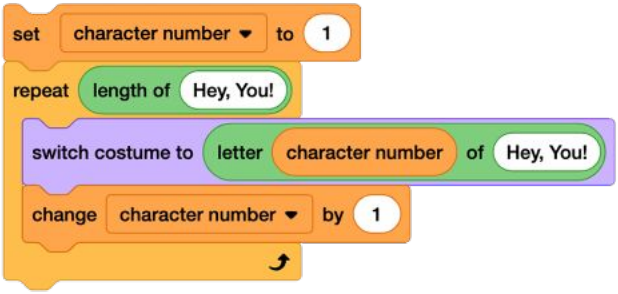

3. Now, create a variable, like “character number” to use in place of a fixed number in the “letter of” block.



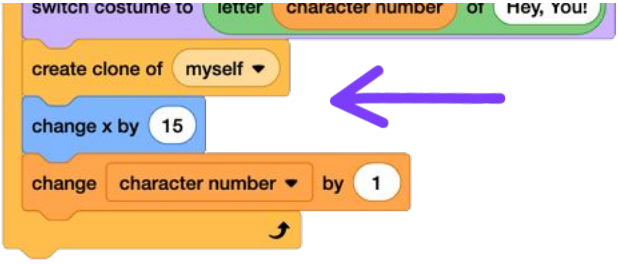
# Text Rendering

scratch.mit.edu

5. Time to create a script that cycles through each letter! You'll want to start at the first character. Then, for the "length of" your string (i.e. the number of letters in your text), increase the character number to show each letter costume in order.




6. We still only see one letter at a time. To print all the letters on the stage, we'll have to add a block to create a clone of each letter and then use a "move" or "change x" block so they appear in a line.

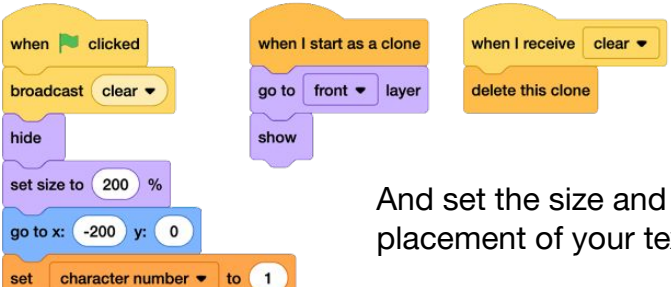



Tinker with the amount you move/change x by based on the size of your letters.

# Text Rendering

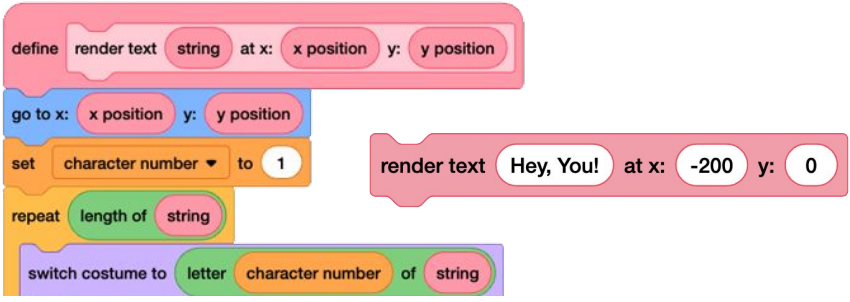
scratch.mit.edu

7. Now that you have clones, you might want to play with hiding and showing the sprite and the clones. You might also want to add a script to delete the clones at some point, if you want to start a fresh row of text.




And set the size and the placement of your text.

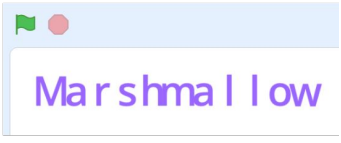
***OPTIONAL:*** Have multiple lines of text to display? Try creating a procedure using a My Block, so you can quickly define the string (and its placement, if desired) without duplicating the entire code stack. (See our My Blocks cards for more information.)

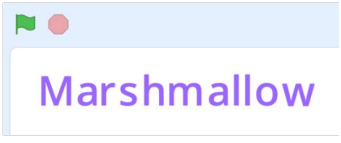


# Text Rendering

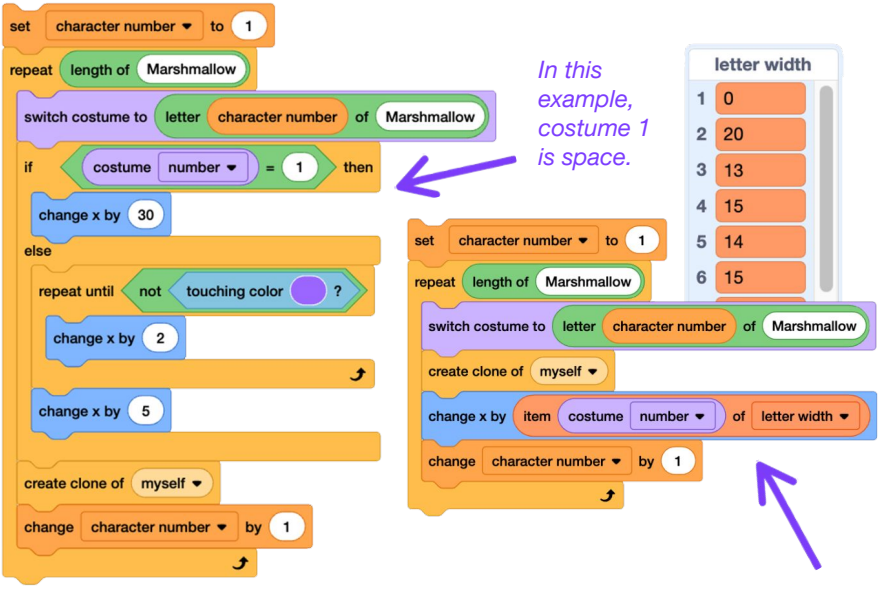
scratch.mit.edu

***OPTIONAL:*** Is your letter spacing a little uneven because each letter has a different width? In typesetting, adjusting the space between letters is called kerning.



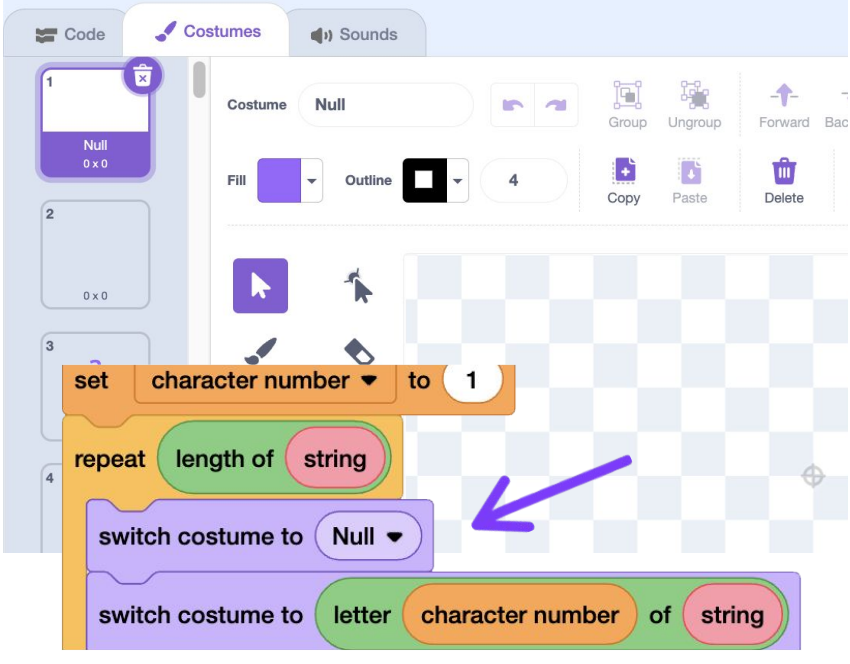

Scratch doesn't have a reporter block that stores the width of a costume, but you can try different solutions to even out the letter placement. The two different examples below are using color sensing or a list of widths (that was manually entered for each costume) to accomplish this.



# Text Rendering

scratch.mit.edu

***OPTIONAL:*** What if one of the letters doesn't have a costume? You could create a costume, like "Null," and first switch to that costume before switching to a letter costume. If there is no letter costume, the costume will remain on "Null" (which in this example is blank but could be anything you choose).



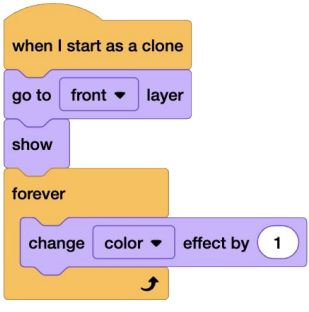
# Text Rendering

scratch.mit.edu

***OPTIONAL:*** Instead of creating a clone, you could use the "stamp" Pen block to record your letters. But what might be some advantages or disadvantages of creating a clone versus using a stamp?

Clones can appear in front of other sprites, for instance. Or what if you wanted to animate the text? You can animate clones.

Experiment! How else can you customize your text?




***OPTIONAL:*** To quickly get started with letters, backpack or export/import the "Letter - BACKPACK THIS" sprite in scratch.mit.edu/projects/1138479378. Or what if you want to use a font not available in the Paint Editor? You could upload SVG or transparent PNG images of letters, or use letters from the sprite library.