



# Original Cards in This File

- Animate a Name Coding Cards
- Imagine a World Coding Cards
- Chase Game Coding Cards
- Make Music Coding Cards (see *new Sound and Music Cards for more*)
- Animate a Character Coding Cards
- Create a Story Coding Cards
- Pong Game Coding Cards
- Let's Dance Coding Cards
- Jumping Game Coding Cards
- Virtual Pet Coding Cards
- Catch Game Coding Cards
- Video Sensing Coding Cards
- Make It Fly Coding Cards



# New Cards in This File

- Sprite Creation Cards
- Sound and Music Cards
- Conditional Statements Coding Cards
- Variables and Lists Coding Cards
- My Blocks Coding Cards
- Face Sensing Coding Cards
- Advanced Topics: Graphic Effects Coding Cards
- Advanced Topics: Clones Coding Cards

## Links to Other Available Cards (not in this file)

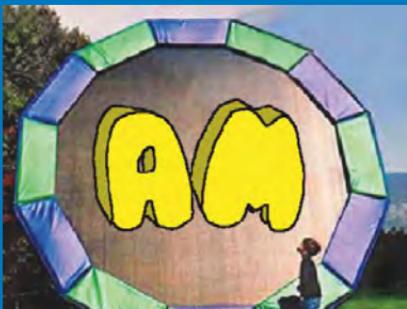
- [micro:bit Coding Cards](#)
- [Makey Makey Coding Cards](#)
- [Vernier Go Direct Force & Acceleration Coding Cards](#)



# Links to Other Available Cards (not in this file)

- [Bring Yourself In Lesson Coding Cards](#)
- [Paper Planes, Turtle Graphics, and Computational Concepts Lesson Coding Cards](#)
- [Build the Change Coding Cards](#)
- [Designing for Creative Learning Prototype to Scratch Cards](#)
- [Hour of Code 2024 with Scratch: Invention Station Coding Cards](#)
- [Hour of Code 2024 with Scratch: Spreading Kindness Coding Cards](#)

# Animate a Name Cards



Animate the letters of your name,  
initials, or favorite word.

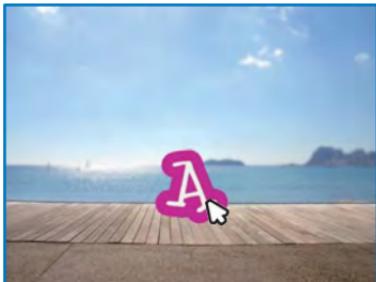
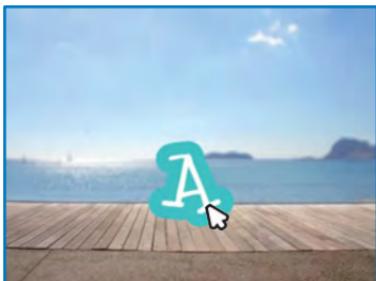
# Animate a Name Cards

Try these cards in any order:

- Color Clicker
- Spin
- Play a Sound
- Dancing Letter
- Change Size
- Press a Key
- Glide Around

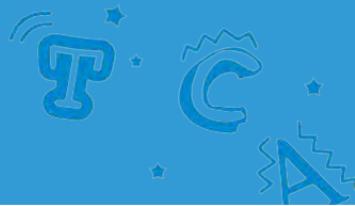
# Color Clicker

Make a letter change color when you click it.



# Color Clicker

scratch.mit.edu



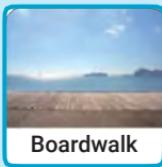
## GET READY



Choose a letter from the Sprite Library.



Choose a backdrop.



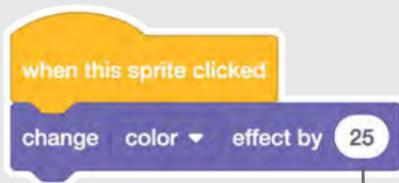
Food

Fashion

Letters

To see just the letter sprites, click the Letters category at the top of the Sprite Library.

## ADD THIS CODE



Try different numbers.

## TRY



Click your letter.

# Spin



Make a letter turn when you click it.

A green letter 'B' is centered within a light blue rectangular frame. A small white cursor arrow points to the exact center of the letter 'B'.

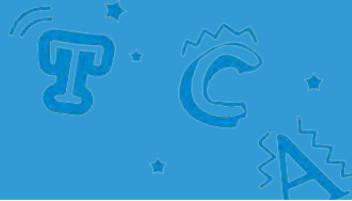
The same green letter 'B' is shown again, but this time it is rotated approximately 45 degrees clockwise relative to the first frame.

The green letter 'B' is shown again, rotated further, this time appearing almost vertically oriented (about 85 degrees clockwise from the original position).

The green letter 'B' is shown again, rotated even more, appearing at an angle of about 135 degrees clockwise from the original position.

# Spin

scratch.mit.edu



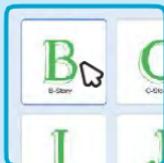
## GET READY



Go to the Sprite Library.

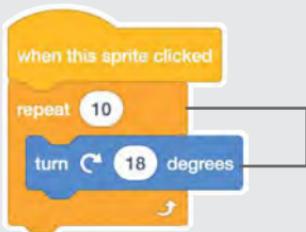


Click the Letters category.



Choose a letter sprite.

## ADD THIS CODE



Try different numbers.

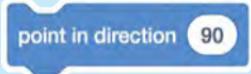
## TRY IT

Click your letter.

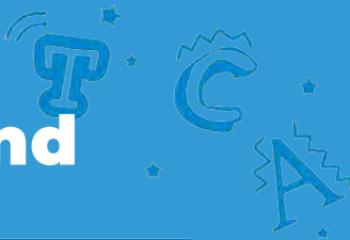


## TIP

Click this block to reset the sprite's direction.



# Play a Sound



Click a letter to play a sound.



# Play a Sound

scratch.mit.edu



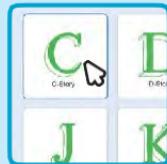
## GET READY



Go to the Sprite Library.



Click the Letters category.



Choose a letter sprite.



Choose a backdrop.



Click the Sounds tab.



Choose a sound.

## ADD THIS CODE



Click the Code tab.



Choose a sound from the menu.

## TRY IT

Click your letter.



# Dancing Letter



Make a letter move to the beat.



# Dancing Letter

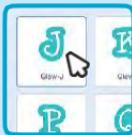
scratch.mit.edu



## GET READY



Choose a backdrop.



Choose a letter from the Sprite Library.



Click the Extensions button (at the bottom left).



Then click Music to add the music blocks.

## ADD THIS CODE



Type a minus sign to move backward.

Choose a drum from the menu.

## TRY IT

Click your letter.



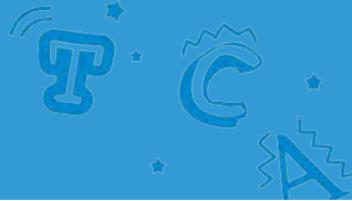
# Change Size

Make a letter get bigger and then smaller.



# Change Size

scratch.mit.edu



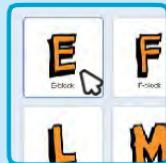
## GET READY



Go to the Sprite Library.



Click the Letters category.



Choose a letter sprite.

## ADD THIS CODE



Type a minus sign to get smaller.

## TRY IT

Click your letter.

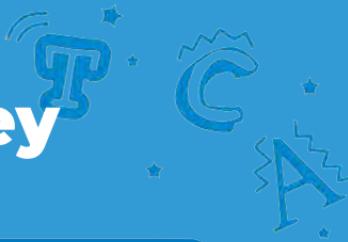


## TIP

Click this block to reset the size.



# Press a Key

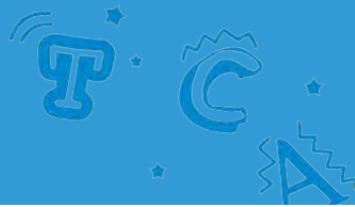


Press a key to make your letter change.



# Press a Key

scratch.mit.edu



## GET READY



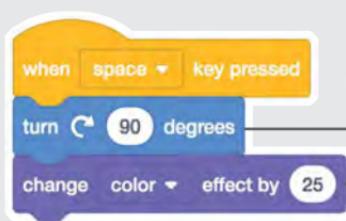
Choose a backdrop.



Choose a letter from the Sprite Library.



## ADD THIS CODE



Try different numbers.

## TRY IT



Press the space key.

## TIP



You can choose a different key from the menu.  
Then press that key!

# Glide Around

Make a letter glide smoothly from place to place.



# Glide Around

scratch.mit.edu



## GET READY



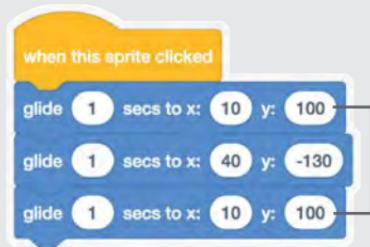
Choose a backdrop.



Choose a letter from the Sprite Library.



## ADD THIS CODE



Try different numbers.

## TRY IT

Click your letter to start.



## TIP



When you move a sprite, you can see the numbers for x and y update.

x is the position from left to right.

y is the position up and down.

SCRATCH

# Imagine a World



Imagine a world where anything is possible!



# Imagine a World

Try these cards in any order:

- Say Something
- Fly Around
- Go Right and Left
- Go Up and Down
- Change Costumes
- Glide from Here to There
- Grow and Shrink
- Change Backdrops
- Add a Sound

# Say Something



Type what you want your sprite to say.



Imagine if...



# Say Something

scratch.mit.edu

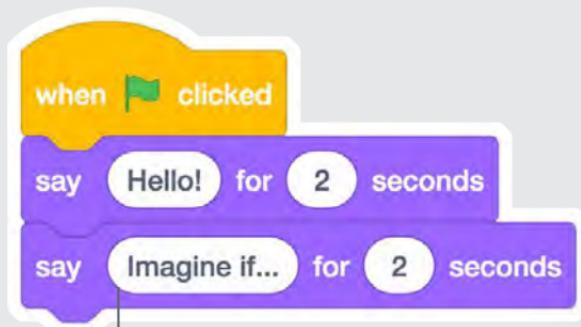


**GET READY**



Select the sprite you want to talk.

**ADD THIS CODE**



**TRY IT**

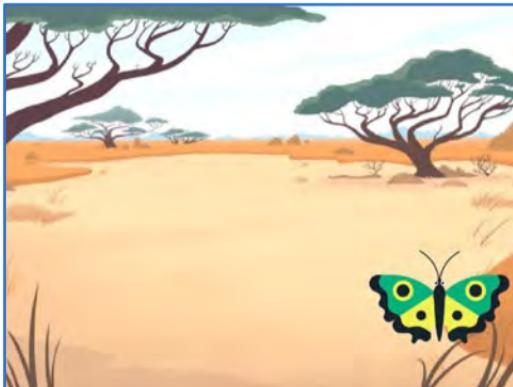
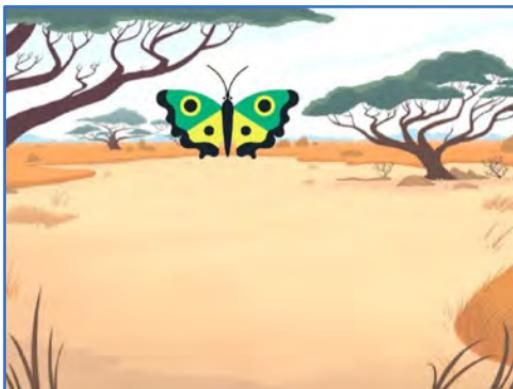
Click the green flag to start.



# Fly Around



Press the space key to glide glide.



# Fly Around

scratch.mit.edu



## GET READY



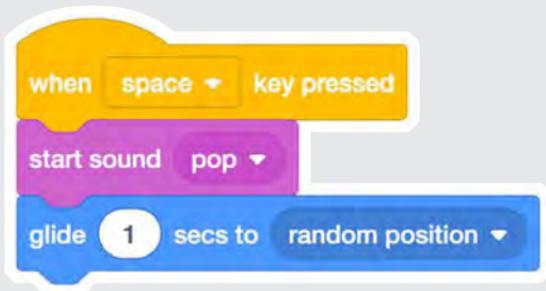
Choose a sprite.



Choose a backdrop.



## ADD THIS CODE



## TRY IT



Press the space key to glide.

# Go Right and Left



Press arrow keys to move right and left.



# Go Right and Left

scratch.mit.edu



## GET READY



Choose a sprite.



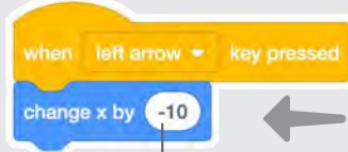
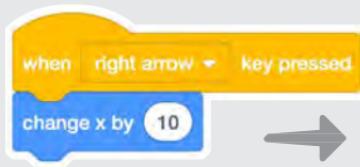
Choose a backdrop.



Playing Field

## ADD THIS CODE

Change **x** to move your character *side to side*.



Type a minus sign to move *left*.

## TRY IT



Press the right and left arrow keys on your keyboard.

# Go Up and Down



Press arrow keys to move up and down.



# Go Up and Down

scratch.mit.edu



## GET READY



Choose a sprite.



Hedgehog



Choose a backdrop.



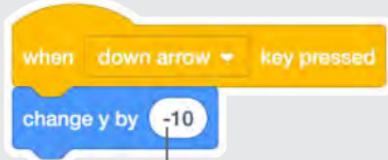
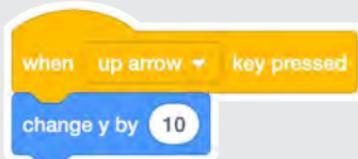
Woods and Bench

## ADD THIS CODE

Change **y** to move your character *up and down*.



Hedgehog



Type a minus sign to move *down*.

## TRY IT



Press the up and down arrow keys on your keyboard.

# Change Costumes



Animate a sprite when you click it.



# Change Costumes

scratch.mit.edu



## GET READY



Choose a sprite.



Rooster



Choose a backdrop.

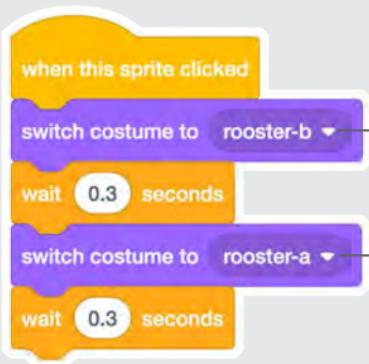


Blue Sky

## ADD THIS CODE



Rooster



Choose one costume.

Choose another.



## TRY IT

Click your sprite.

# Glide From Here to There

Make a sprite glide from one point to another.



# Glide From Here to There

scratch.mit.edu



## GET READY



Choose a sprite.



Earth



Choose a backdrop.



Stars

## ADD THIS CODE

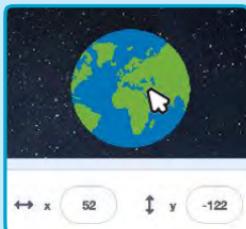


Set the starting point.

Set the end point.

## TRY IT

Click the green flag to start.



## TIP

When you move a sprite, you can see the numbers for **x** and **y** update.

**x** is the position from left to right.

**y** is the position up and down.

# Grow and Shrink



Make a sprite change size when you click it.



# Grow and Shrink

scratch.mit.edu



## GET READY



Choose a backdrop.



Theater 2



Choose a sprite.



Drums Tabla

## ADD THIS CODE



Drums Tabla



Type a larger number  
to make it bigger.

Type 100 to return to  
original size.

## TRY IT



Click your sprite.

# Change Backdrops



Change scenes by switching backdrops.



# Change Backdrops

scratch.mit.edu



## GET READY



Choose two backdrops.



Savanna



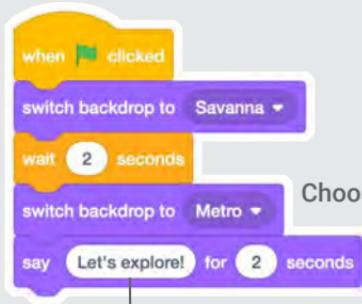
Metro



Choose a sprite.



## ADD THIS CODE



Choose the backdrop you want to start with.

Choose the second backdrop.

Type what you want to say.

## TRY IT

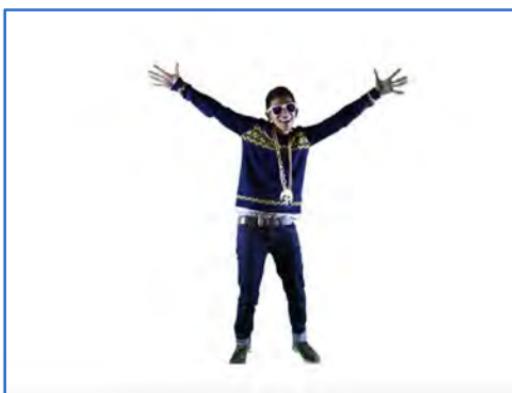
Click the green flag to start.



# Add a Sound



Add your voice or other sounds  
to your project.



# Add a Sound

scratch.mit.edu



## GET READY

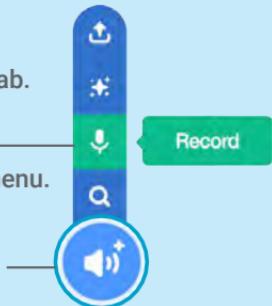


Choose a sprite.

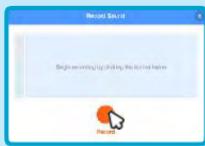


Click the Sounds tab.

Then click Record—  
from the pop-up menu.



Or, click here to  
choose a sound  
from the  
library.



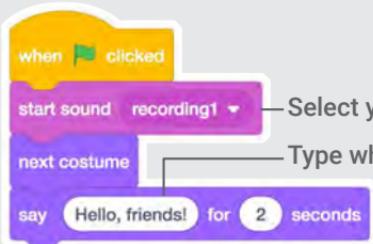
To record your voice or other  
sound, click the red button.

If your device is unable to  
record, you can choose a sound.

## ADD THIS CODE



Click the Code tab.



Select your sound.

Type what you want to say.

## TRY IT.

Click the green flag to start.



# Chase Game Cards



Make a game where you chase a character to score points.



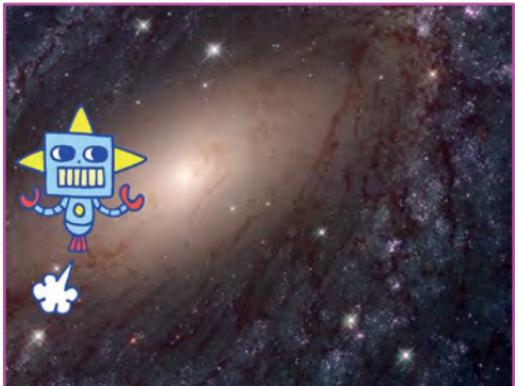
# Chase Game Cards

Use these cards in this order:

1. Move Left and Right
2. Move Up and Down
3. Chase a Star
4. Play a Sound
5. Add a Score
6. Level Up!
7. Victory Message

# Move Left and Right

Press arrow keys to move left and right.



# Move Left and Right

scratch.mit.edu



## GET READY



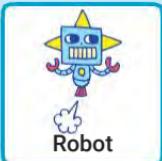
Choose a backdrop.



Galaxy



Choose a character.



Robot

## ADD THIS CODE



Robot



Choose right arrow.



Choose left arrow.

Type a minus sign to move left.

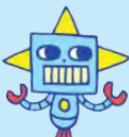
## TRY IT

Press the arrow keys.



## TIP

Type a negative number to move to the left.



Type a positive number to move to the right.



# Move Up and Down

Press arrow keys to move up and down.

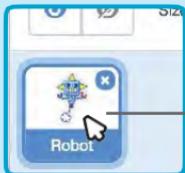


# Move Up and Down

scratch.mit.edu



## GET READY



Click your character to select it.

## ADD THIS CODE



Choose up arrow.



Choose down arrow.

Use the **change y by** block to move up.

Type a minus sign to move down.

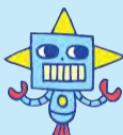
## TRY IT

Press the arrow keys.

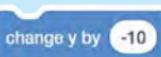


## TIP

y is the position on the Stage from top to bottom.



Type a positive number to move up.



Type a negative number to move down.

# Chase a Star

Add a sprite to chase.



# Chase a Star

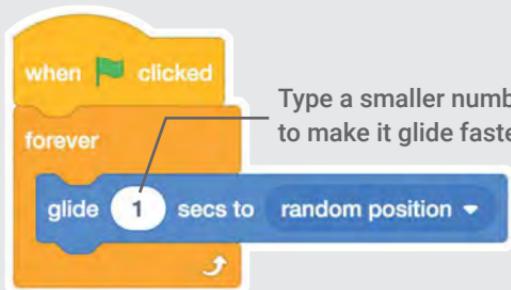
scratch.mit.edu

## GET READY



Choose a sprite to chase, like Star.

## ADD THIS CODE



Type a smaller number (like 0.5) to make it glide faster.

## TRY IT

Click the green flag to start.



Click the stop sign to stop.

# Play a Sound

Play a sound when your character touches the star.

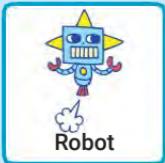


# Play a Sound

scratch.mit.edu



## GET READY



Click to select the Robot sprite.



Click the **Sounds** tab.



Choose a sound from the Sounds Library, like Collect.

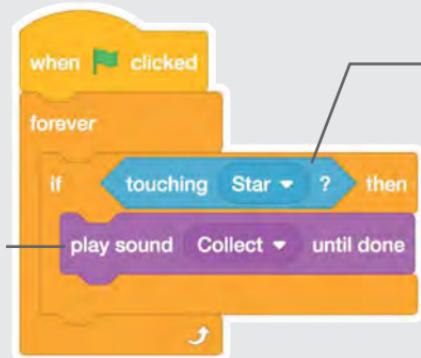
## ADD THIS CODE



Click the **Code** tab and add this code.



Choose your sound from the menu.



Insert the touching block into the if then block.



## TRY IT

Click the green flag to start.



# Add a Score

Score points when you touch the star.



# Add a Score

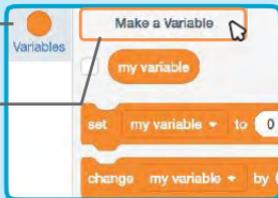
scratch.mit.edu



## GET READY

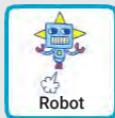
Choose Variables.

Click the Make a Variable button.

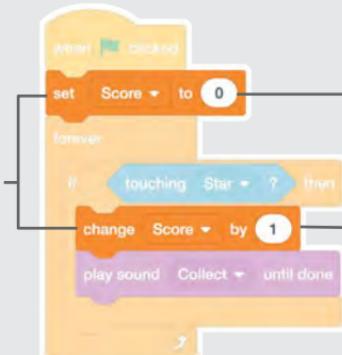


Name this variable Score and then click OK.

## ADD THIS CODE



Select Score from the menu.



Add this block to reset the score.

Add this block to increase the score.

## TIP



Use the **set variable** block to reset the score to zero.

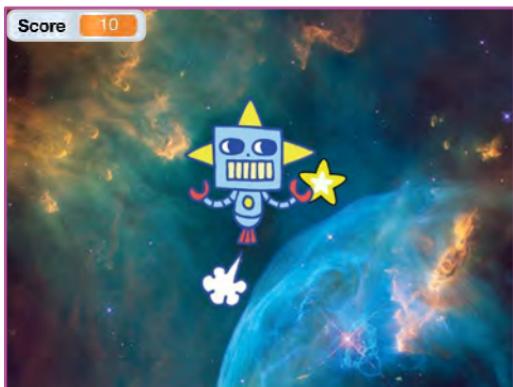


Use the **change variable** block to increase the score.

# Level Up!



Go to the next level.



# Level Up!

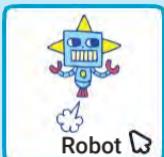
scratch.mit.edu



## GET READY



Choose a second backdrop, like Nebula.

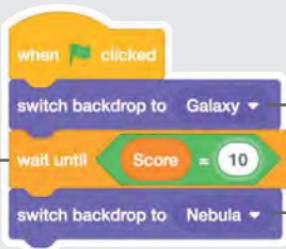


Select the Robot

## ADD THIS CODE

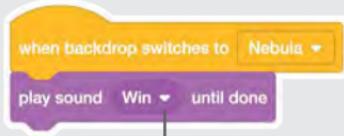


Insert the Score block into the equals block (from the Operators category).



Choose your first backdrop.

Choose the backdrop to switch to.



Choose a sound.

## TRY IT

Click the green flag to start the game!



# Victory Message

Show a message when you go to  
the next level.



# Victory Message

scratch.mit.edu



## GET READY



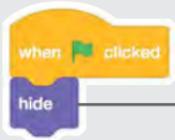
Click the **Paint** icon to make a new sprite.

Use the **Text** tool to write a message, like "Level Up!"

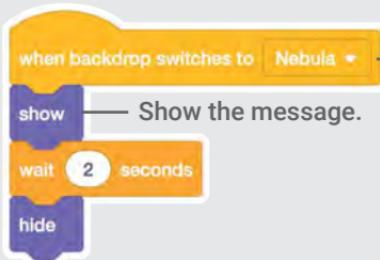


You can change the font color, size, and style.

## ADD THIS CODE



Hide the message at the beginning.



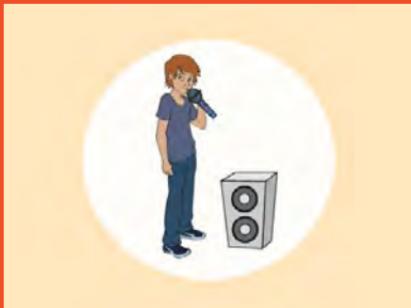
Choose the backdrop for the next level.

## TRY IT

Click the green flag to play your game.



# Make Music Cards



Choose instruments, add sounds,  
and press keys to play music.

# Make Music Cards

Try these cards in any order:

- Play a Drum
- Make a Rhythm
- Animate a Drum
- Make a Melody
- Play a Chord
- Surprise Song
- Beatbox Sounds
- Record Sounds
- Play a Song

# Play a Drum

Press a key to make a drum sound.



# Play a Drum

scratch.mit.edu

## GET READY



Choose a  
backdrop.



Theater 2



Choose a drum.



Drum

## ADD THIS CODE



Select the sound you want from the menu.

## TRY IT



Press the **space** key on your keyboard.

# Make a Rhythm

Play a loop of repeating drum sounds.



# Make a Rhythm

scratch.mit.edu

## GET READY



Choose a backdrop.



Choose a drum from the Music category.



Dance

Music

Sports

To see just the music sprites, click the **Music** category at the top of the Sprite Library.

## ADD THIS CODE



Type how many times you want to repeat.  
Try different numbers to change the rhythm.

## TRY IT



Press the **space** key on your keyboard.

# Animate a Drum

Switch between costumes to animate.



# Animate a Drum

scratch.mit.edu

## GET READY



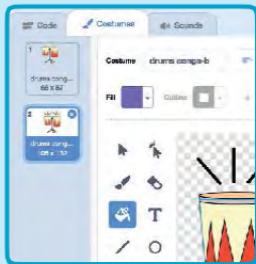
Choose  
a drum.



### Costumes

Click the **Costumes** tab  
to see the costumes.

You can use the paint  
tools to change colors.



## ADD THIS CODE

### Code

Click the **Code** tab.



Choose a sound  
from the menu.

## TRY IT



Press the **left arrow** key on your keyboard.

# Make a Melody

Play a series of notes.



# Make a Melody

scratch.mit.edu

## GET READY



Choose an instrument, like Saxophone.



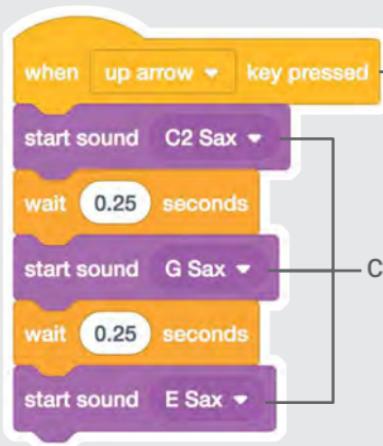
Dance

Music

Sports

To see just the music sprites, click the **Music** category at the top of the Sprite Library.

## ADD THIS CODE



Choose up arrow (or another key).

Choose different sounds.

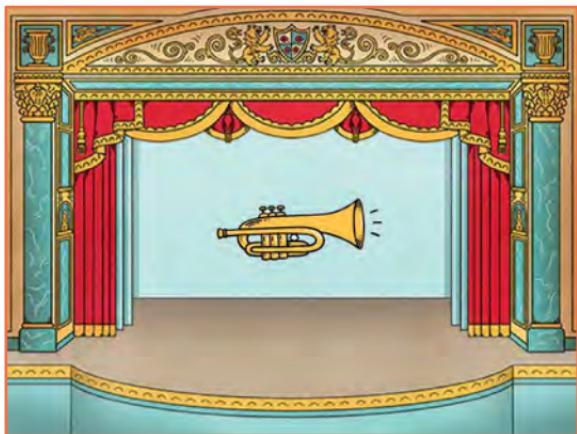
## TRY IT



Press the up arrow key.

# Play a Chord

Play more than one sound at a time to make a chord.



# Play a Chord

scratch.mit.edu

## GET READY



Choose an instrument, like Trumpet.



Trumpet

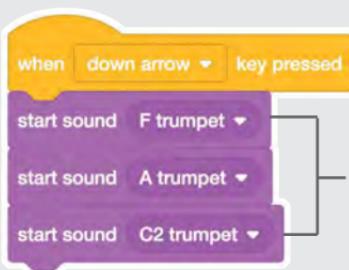
Dance

Music

Sports

To see just the music sprites, click the **Music** category at the top of the Sprite Library.

## ADD THIS CODE



Choose down arrow (or another key).

Choose different sounds.

## TRY IT



Press the **down arrow** key.

## TIP

Use to play sounds play at the same time.

Use to play sounds one after another.

# Surprise Song

Play a random sound from a list of sounds.



# Surprise Song

scratch.mit.edu

## GET READY



Choose an instrument,  
like Guitar.



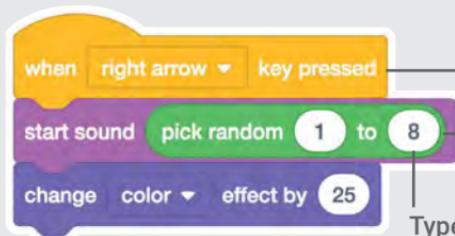
Click the **sounds** tab to see how many sounds are in your instrument.



## ADD THIS CODE



Click the **Code** tab.



Choose **right arrow**.

Insert a **pick random** block.

Type the number of sounds in your instrument.

## TRY IT



Press the **right arrow** key.

# Beatbox Sounds

Play a series of vocal sounds.



# Beatbox Sounds

scratch.mit.edu

## GET READY



Choose the Microphone sprite.



Microphone



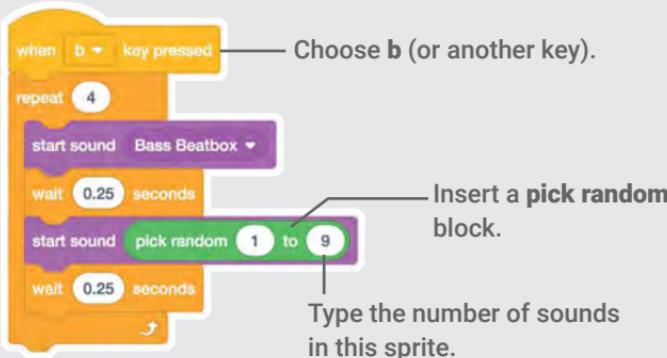
Click the **Sounds** tab to see how many sounds are in your instrument.



## ADD THIS CODE



Click the **Code** tab.



## TRY IT



Press the **B** key to start.

# Record Sounds

Make your own sounds to play.



# Record Sounds

scratch.mit.edu

## GET READY



Choose a backdrop.



Beach Malibu



Choose any sprite.

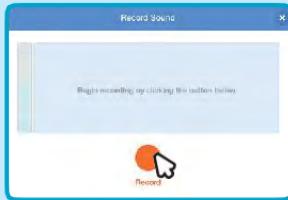
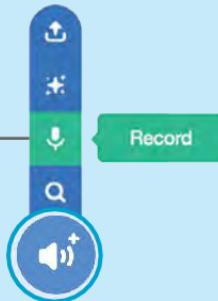


Beachball



Click the **Sounds** tab.

Then choose **Record** from the pop-up menu.

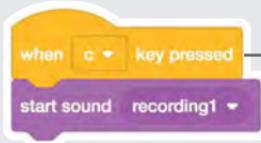


Click the **Record** button to record a short sound.

## ADD THIS CODE



Click the **Code** tab.



Choose **c** (or another key).

## TRY IT



Press the **C** key to start.

# Play a Song

Add a music loop as background music.



# Play a Song

scratch.mit.edu

## GET READY



Choose a sprite,  
like Speaker.



Click the **Sounds** tab.



Choose a sound from  
the **Loops** category,  
like Drum Jam.

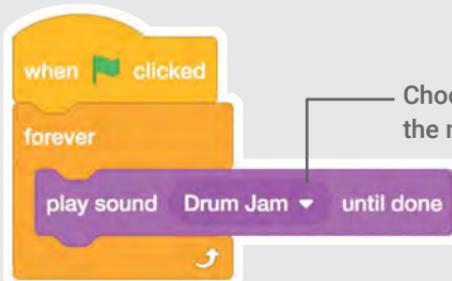


To see just the music loops, click the **Loops** category at the top of the Sounds Library.

## ADD THIS CODE



Click the **Code** tab.



Choose your sound from  
the menu.

## TRY IT

Click the green flag to start.



# Animate a Character Cards



Bring characters to life with animation.



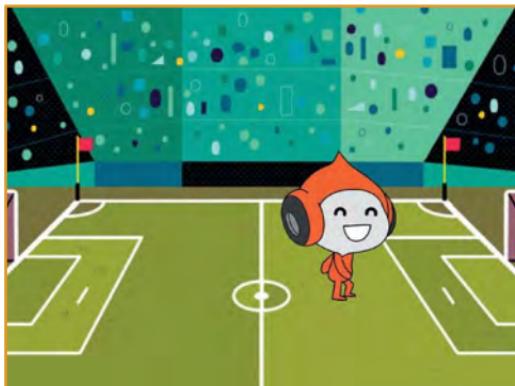
# Animate a Character Cards

Try these cards in any order:

- Move with Arrow Keys
- Make a Character Jump
- Switch Poses
- Glide from Point to Point
- Walking Animation
- Flying Animation
- Talking Animation
- Draw an Animation

# Move with Arrow Keys

Use the arrow keys to move your character around.



# Move with Arrow Keys

scratch.mit.edu

## GET READY



Choose a backdrop.



Soccer 2



Choose a character.

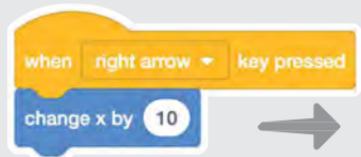


Pico Walking

## ADD THIS CODE

### Change x

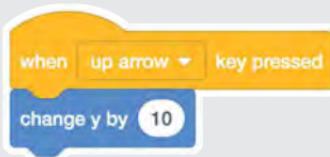
Move your character side to side.



Type a minus sign to move left.

### Change y

Move your character up and down.



Type a minus sign to move down.

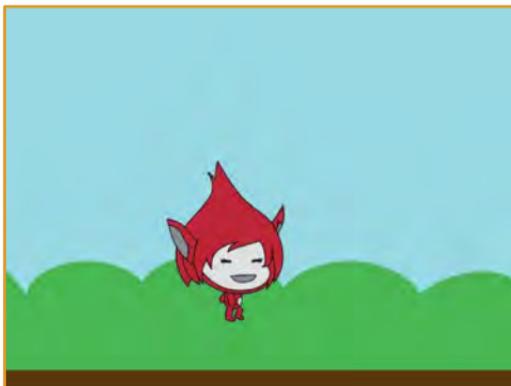
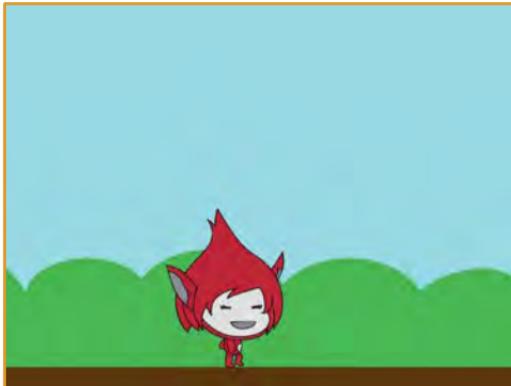
## TRY IT



Press the arrow keys on your keyboard to move your character around.

# Make a Character Jump

Press a key to jump up and down.



# Make a Character Jump

scratch.mit.edu

## GET READY



Choose a backdrop.



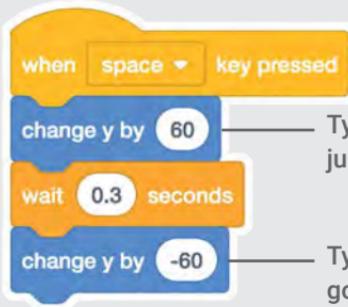
Choose a character.



## ADD THIS CODE



Giga



Type how high to jump.

Type a minus sign to go back down.

## TRY IT



Press the space key on your keyboard.

# Switch Poses

Animate a character when you press a key.



# Switch Poses

scratch.mit.edu

## GET READY

Choose a character with multiple costumes, like Max.



Scroll over sprites in the Sprite Library to see if they have different costumes.



Click the Costumes tab to view all of your sprite's costumes.

## ADD THIS CODE



Click the Code tab.



Choose a costume.

Choose a different costume.

## TRY IT



Press the space key on your keyboard.

# Glide from Point to Point

Make a sprite glide from point to point.



# Glide from Point to Point

scratch.mit.edu

## GET READY



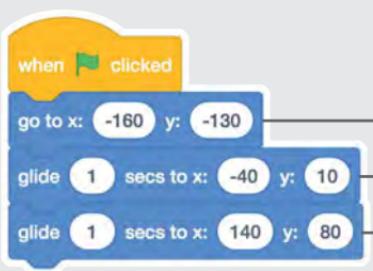
Choose a backdrop.



Choose a character.



## ADD THIS CODE



Set the starting point.

Set another point to glide to.

Set the end point.

## TRY IT

Click the green flag to start.



## TIP



When you drag a sprite, its x and y positions will update in the blocks palette.

# Walking Animation

Make a character walk or run.



# Walking Animation

scratch.mit.edu

## GET READY



Choose a backdrop.



Jungle



Choose a walking or running sprite.



Unicorn Running

## ADD THIS CODE

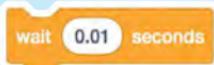


## TRY IT



Click the green flag to start.

## TIP



If you want to slow down the animation, try adding a wait block inside the repeat block.

# Flying Animation

Have a character flap its wings as it moves across the stage.

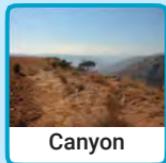


# Flying Animation

scratch.mit.edu



Choose a backdrop.



Canyon



Choose Parrot  
(or another flying sprite).



Parrot

## GET READY

## ADD THIS CODE

### Glide across the screen

```
when green flag clicked
go to x: -170 y: 120
glide (1) secs to x: 150 y: 50
```

Set the starting point.  
Set the end point.

### Flap the wings

```
when green flag clicked
repeat (5)
  switch costume to [parrot-a v]
  wait (0.1) seconds
  switch costume to [parrot-b v]
  wait (0.1) seconds
```

Choose one costume.  
Choose another.

## TRY IT

Click the green flag to start.



# Talking Animation

Make a character talk.



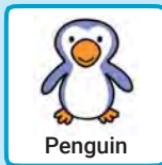
# Talking Animation

scratch.mit.edu

## GET READY



Choose Penguin 2.



Penguin

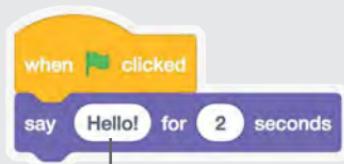


Click the Costumes tab to view the penguin's other costumes.

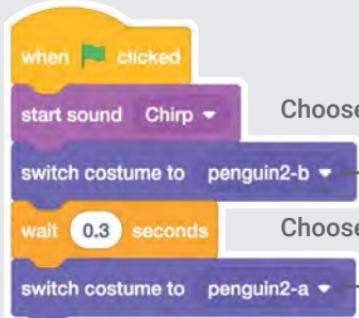
## ADD THIS CODE



Click the Code tab.



Type what you want your character to say.



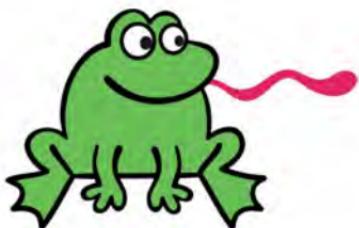
## TRY IT

Click the green flag to start.



# Draw an Animation

Edit a sprite's costumes to create your own animation.



# Draw an Animation

scratch.mit.edu



Choose a character.



Frog

## GET READY



Click the Costumes tab.



Right-click (on a Mac, control-click) a costume to duplicate it.

Now you should have two identical costumes.



Click a costume to select and edit it.

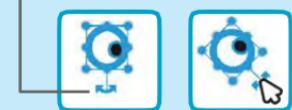
Click the Select tool.



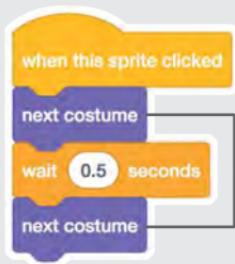
Select a part of the costume to squeeze or stretch it.



Drag the handle to rotate an object you've selected.



## ADD THIS CODE



Click the Code tab.

Use the **next costume** block to animate your character.

## TRY IT



Click the green flag to start.

# Create a Story Cards



Choose characters, add conversation,  
and bring your story to life.

# Create a Story Cards

Start with the first card, and then try the other cards in any order:

- Start a Story
- Start a Conversation
- Switch Backdrops
- Click a Character
- Add Your Voice
- Glide to a Spot
- Walk onto the Stage
- Respond to a Character
- Add a Scene

# Start a Story

Set the scene and have a character  
say something.



# Start a Story

scratch.mit.edu

## GET READY



Choose a backdrop.



Witch House



Choose a character.

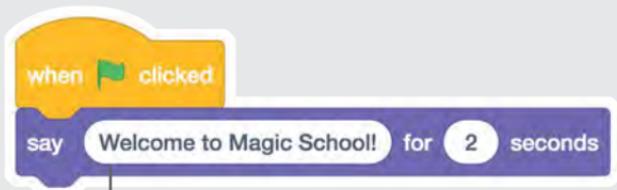


Wizard

## ADD THIS CODE



Wizard



Type what you want your character to say.

## TRY IT

Click the green flag to start.



# Start a Conversation

Make two characters talk to each other.



# Start a Conversation

scratch.mit.edu

## GET READY

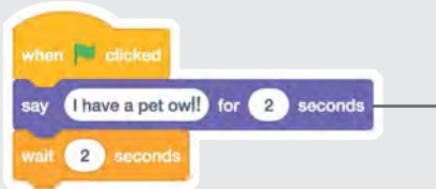


Choose two characters,  
like Witch and Elf.

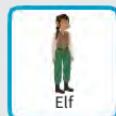


## ADD THIS CODE

Click the thumbnail for each character, and then add its code.



Type what you want  
each character to say.



## TIP



To change the direction a character is facing, click the **Costumes** tab, then click **Flip Horizontal**.



# Switch Backdrops

Change from one backdrop to another.



# Switch Backdrops

scratch.mit.edu

## GET READY



Choose a character.



Choose two backdrops.



## ADD THIS CODE



Choose the backdrop you want to start with.

Choose the second backdrop.

## TRY IT

Click the green flag to start.



# Click a Character

Make your story interactive.



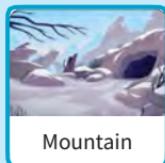
# Click a Character

scratch.mit.edu

## GET READY



Choose a backdrop.



Choose a character.



## ADD THIS CODE



You can choose different effects.

Select a sound from the menu.

## TRY IT

Click your character.



# Add Your Voice

Record your voice to make a character talk.



# Add Your Voice

scratch.mit.edu



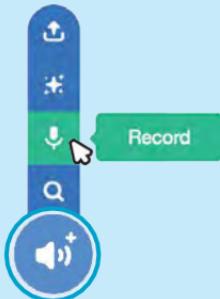
Choose a sprite.



Princess



Click the Sounds tab.



Choose Record from the pop-up menu.

Click Record.

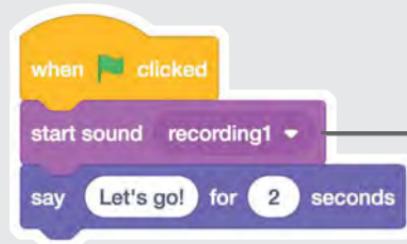


When you're done, click Save.

## ADD THIS CODE



Click the Code tab.



Select your recording from the menu.

## TRY IT

Click the green flag to start.



# Glide to a Spot

Make a character move across the Stage.



# Glide to a Spot

scratch.mit.edu

## GET READY



Choose a backdrop.



Mountain



Choose a character.



Owl

## ADD THIS CODE



Owl

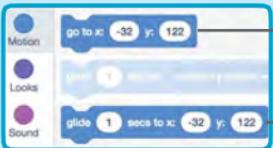


## TRY IT

Click the green flag to start.



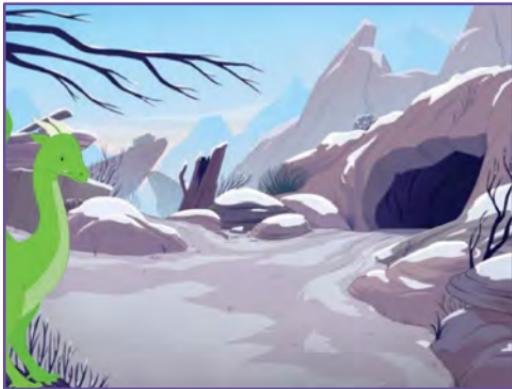
## TIP



When you drag a sprite, the numbers for x and y will update in the blocks palette.

# Walk onto the Stage

Have a character enter the scene.



# Walk onto the Stage

scratch.mit.edu

## GET READY



Choose a backdrop.



Mountain

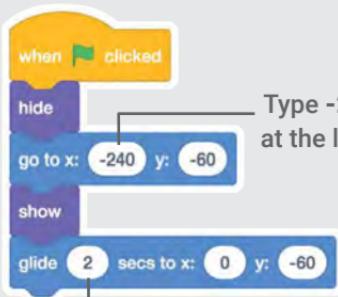


Choose a character.



Dragon

## ADD THIS CODE

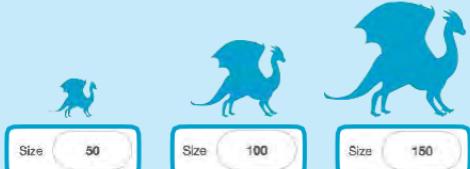
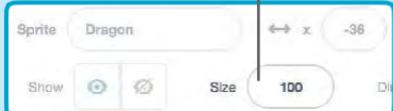


Type **-240** to place your sprite at the left edge of the Stage.

Change this number to glide faster or slower.

## TIP

Change the size of a sprite by typing a smaller or larger number.



# Respond to a Character

Coordinate a conversation so that one character talks after another.



# Respond to a Character

scratch.mit.edu

## GET READY



Choose a backdrop.



Mountain



Choose two characters.



Goblin



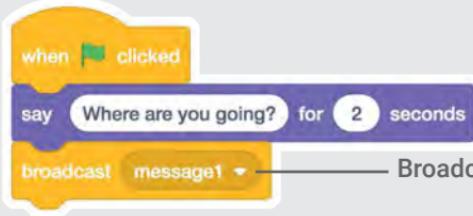
Princess

## ADD THIS CODE

Click the thumbnail for each character, and then add its code.



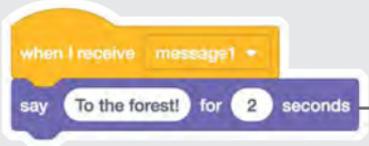
Goblin



Broadcast a message.



Princess



Tell this character what to do when it receives the broadcast.

## TIP



You can click the menu to add a new message.

# Add a Scene

Create multiple scenes with different backdrops and characters.



# Add a Scene

scratch.mit.edu



## GET READY



Choose two backdrops.



Witch House



Mountain

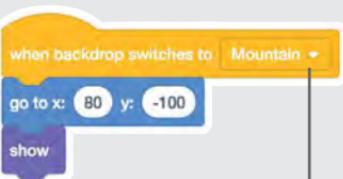
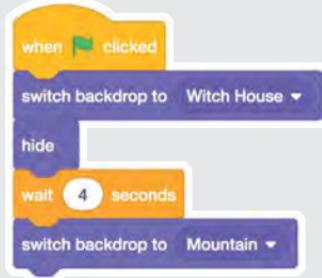


Choose a character.



Fox

## ADD THIS CODE



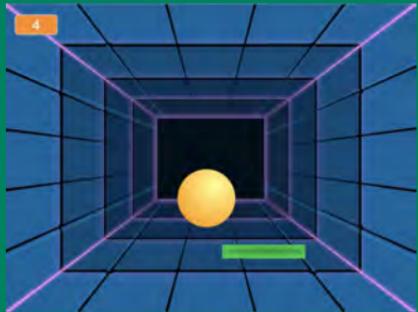
Choose the backdrop name from the menu.

## TRY IT

Click the green flag to start.



# Pong Game Cards



Make a bouncing ball game and score points to win!



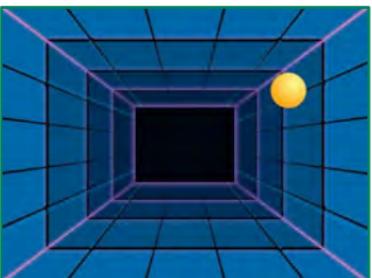
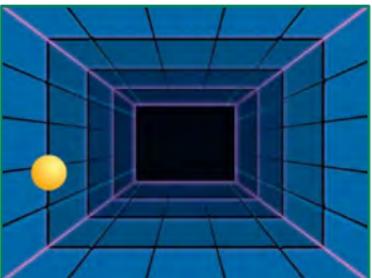
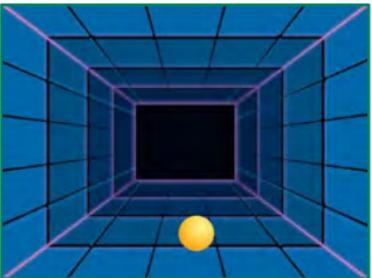
# Pong Game Cards

Use these cards in this order:

1. **Bounce Around**
2. **Move the Paddle**
3. **Bounce off the Paddle**
4. **Game Over**
5. **Score Points**
6. **Win the Game**

# Bounce Around

Make a ball move around the Stage.



# Bounce Around

scratch.mit.edu

## GET READY



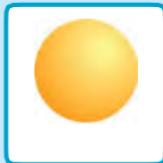
Choose a backdrop.



Neon Tunnel



Choose a ball.



## ADD THIS CODE



Type a larger number to move faster.

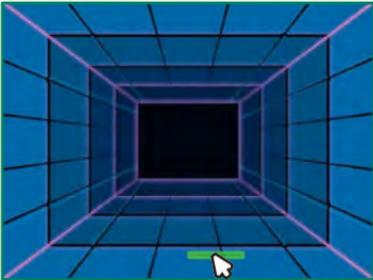
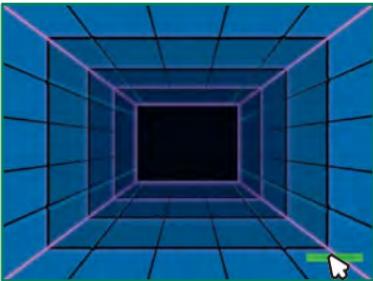
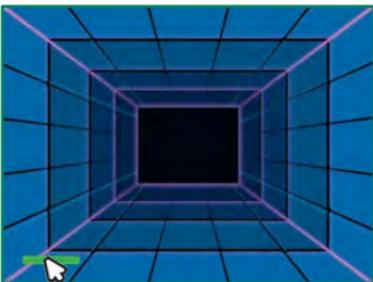
## TRY IT

Click the green flag to start.



# Move the Paddle

Control a paddle by moving  
your mouse pointer.



# Move the Paddle

scratch.mit.edu

## GET READY

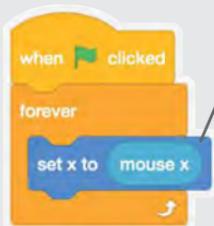


Choose a sprite for hitting the ball, like Paddle.



Then, drag your paddle to the bottom of the Stage.

## ADD THIS CODE



Insert the **mouse x** block into the **set x to** block.



## TRY IT

Click the green flag to start.

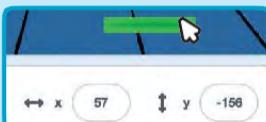


Move your mouse pointer to move the paddle.



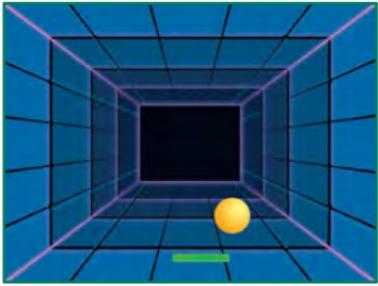
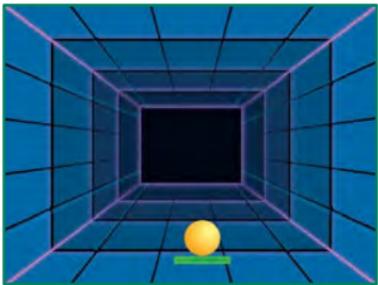
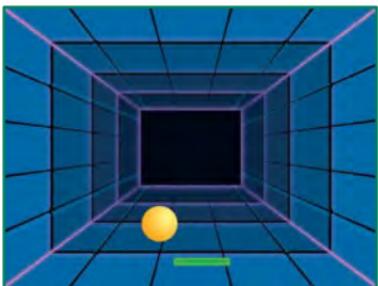
## TIP

You can see the **x** position of the paddle change as you move the mouse pointer across the Stage.



# Bounce Off the Paddle

Make the ball bounce off the paddle.



# Bounce Off the Paddle

scratch.mit.edu

## GET READY

Click to select the Ball sprite.



## ADD THIS CODE

Add this new stack of blocks to your Ball sprite.



The image shows a Scratch script for the Ball sprite. It starts with a 'when green flag clicked' hat block, followed by a 'forever' control loop. Inside the loop is an 'if touching Paddle then' control block. This is followed by a 'turn (pick random 170 to 190 degrees)' motion block, a 'move (15 steps)' motion block, and a 'wait (0.5 seconds)' control block. A callout box points to the 'if touching Paddle then' block with the text: 'Choose Paddle from the menu.' Another callout box points to the 'pick random 170 to 190' block with the text: 'Insert the pick random block and type in 170 to 190'.

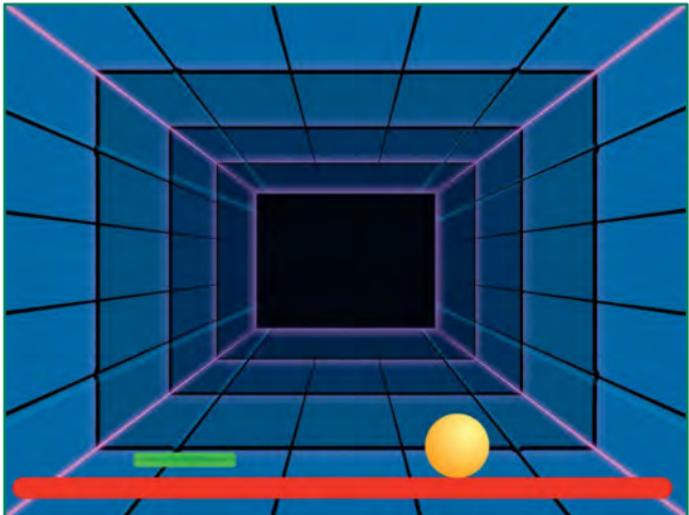
## TRY IT

Click the green flag to start.



# Game Over

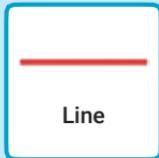
Stop the game if the ball hits the red line.



# Game Over

scratch.mit.edu

## GET READY

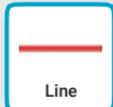


Choose the sprite called Line.



Drag the Line sprite to the bottom of the Stage.

## ADD THIS CODE



Set the position of the Line.

Choose Ball from the menu.

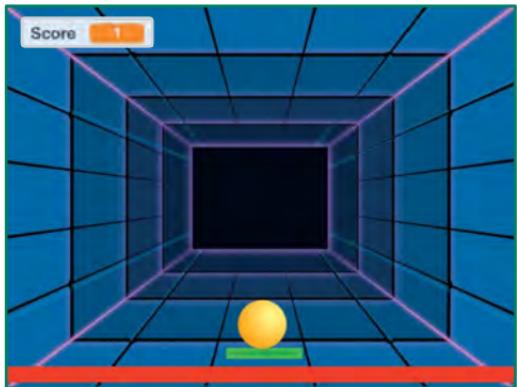
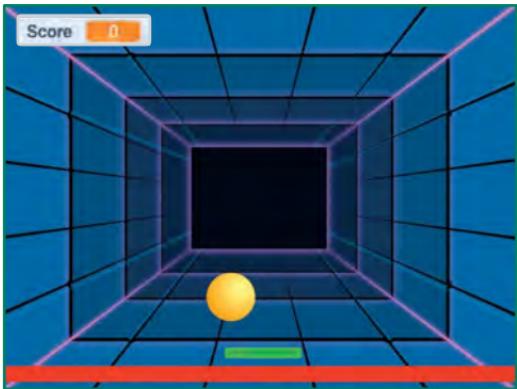
## TRY IT

Click the green flag to start.



# Score Points

Add a point each time you hit the ball  
with the paddle.



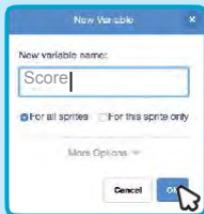
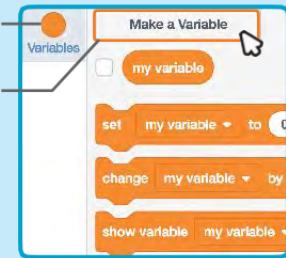
# Score Points

scratch.mit.edu

## GET READY

Choose Variables.

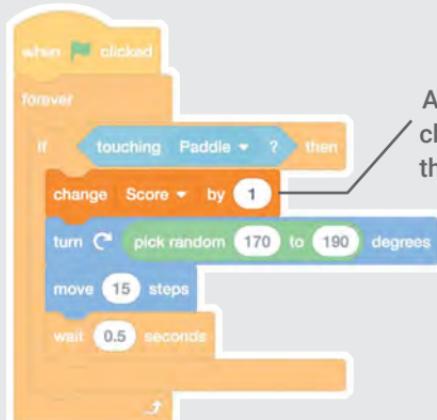
Click the Make a Variable button.



Name this variable **Score** and then click **OK**.

## ADD THIS CODE

Click to select the Ball sprite.



Add this block and choose **Score** from the menu.



Use this block to reset the score. Choose **Score** from the menu.

# Win the Game

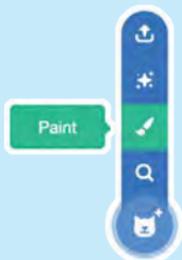
When you score enough points, display a winning message!



# Win the Game

scratch.mit.edu

## GET READY



Click the Paint icon  
to make a new sprite.

Use the Text tool to write a message, like "You Won!"

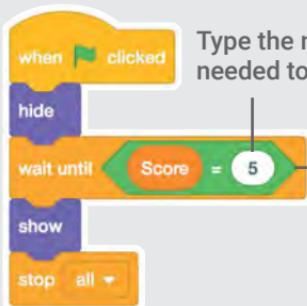


You can change the font color, size, and style.

## ADD THIS CODE



Click the Code tab.



Type the number of points needed to win the game.



Insert the Score block into the equals block from the Operators category.

## TRY IT

Click the green flag to start.



Play until you score enough points to win!

# Let's Dance Cards



Design an animated dance scene  
with music and dance moves.



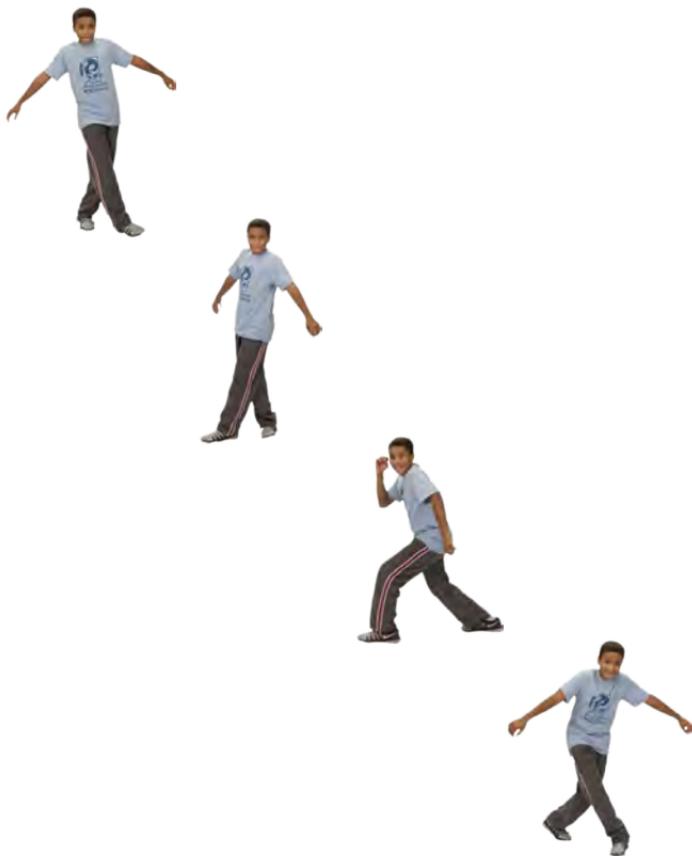
# Let's Dance Cards

Try these cards in any order:

- **Dance Sequence**
- **Dance Loop**
- **Play Music**
- **Take Turns**
- **Starting Position**
- **Shadow Effect**
- **Interactive Dance**
- **Color Effect**
- **Leave a Trail**

# Dance Sequence

Make an animated dance.



Let's Dance

1

SCRATCH

# Dance Sequence

scratch.mit.edu



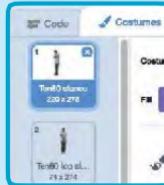
## GET READY



Choose a dancer.



Ten80 Dance



Costumes

Click the Costumes tab to see the different dance moves.



To see just the dance sprites, click the Dance category at the top of the Sprite Library.

## ADD THIS CODE



Click the Code tab.



Type how long to wait between dance moves.



Pick different dance moves.

## TRY IT

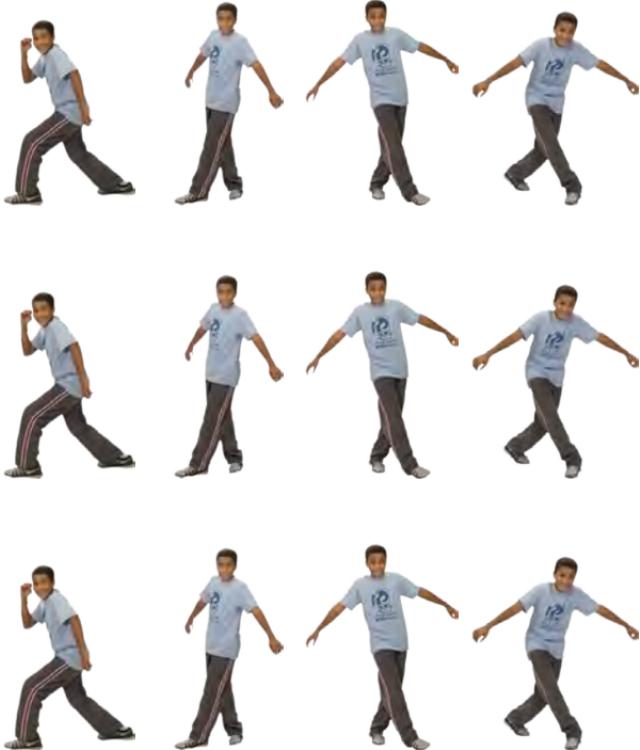
Click the green flag to start



# Dance Loop!



Repeat a series of dance steps.



# Dance Loop

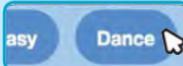
scratch.mit.edu



## GET READY



Go to the  
Sprite Library.



Click the Dance category.



Choose a dancer.

## ADD THIS CODE



Add a repeat  
block around your  
dance sequence.



Choose a dance pose.

Type how many times  
you want to repeat the  
dance.

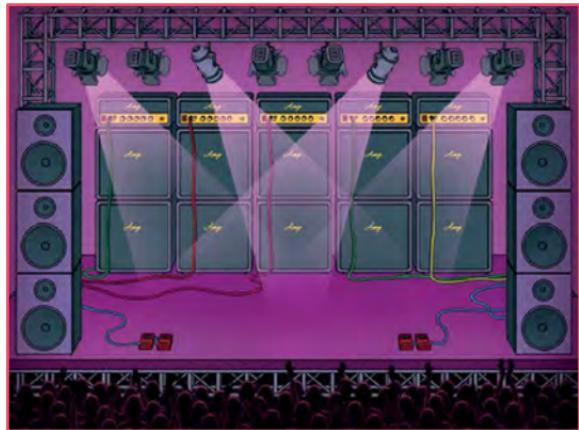
## TRY IT

Click the green flag to start.



# Play Music

Play and loop a song.



Let's Dance

3

SCRATCH

# Play Music

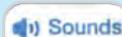
scratch.mit.edu



## GET READY



Choose a backdrop.



Click the Sounds tab.

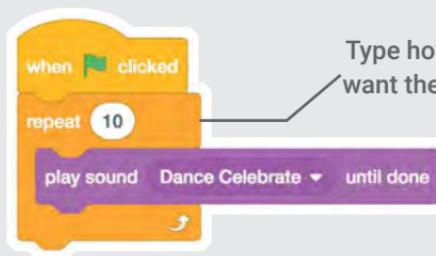


Choose a song from the Loops category.

## ADD THIS CODE



Click the Code tab.



Type how many times you want the song to repeat.

## TIP

Make sure to use

`play sound [Dance Celebrate v] until done`

(not

`start sound [Dance Celebrate v]`

)

or else the music won't finish playing before it begins again.

# Take Turns♪



Coordinate dancers so that one begins after the other finishes.



# Take Turns

scratch.mit.edu



## GET READY



Choose two dancers from the Dance category.



## ADD THIS CODE



```
when green flag clicked
  switch costume to [anina top L step v]
  wait [0.3 seconds]
  switch costume to [anina top R step v]
  wait [0.3 seconds]
  switch costume to [anina stance v]
  broadcast [message1 v]
```

Broadcast a message.



```
when I receive [message1 v]
  say [My turn to dance!] for [1] seconds
  repeat (4)
    next costume
    wait [0.3 seconds]
```

Tell this dancer sprite what to do when it receives the message.

## TRY IT

Click the green flag to start.



# Starting Position

Tell your dancers where to start.



# Starting Position

scratch.mit.edu



## GET READY



Go to the  
Sprite Library.

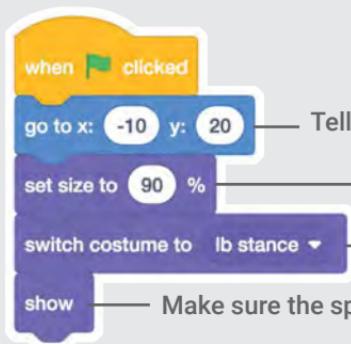


Click the Dance category.



Choose a dancer.

## ADD THIS CODE



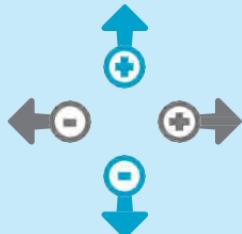
Tell your sprite where to start.

Set the sprite's size.

Choose a starting costume.

Make sure the sprite is showing.

## TIP



Use `go to x: [ ] y: [ ]` to set a sprite's position on the Stage.

**x** is the position on the Stage from left to right.

**y** is the position on the Stage from top to bottom.

# Shadow Effect

Make a dancing silhouette.



# Shadow Effect

scratch.mit.edu



## GET READY



Go to the  
Sprite Library.



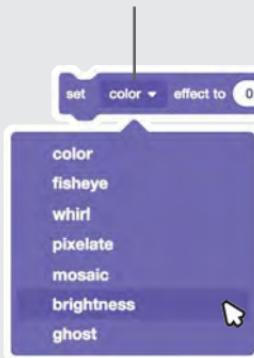
Click the **Dance** category.



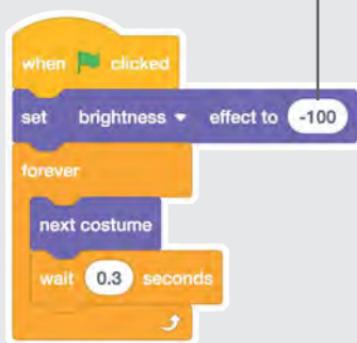
Choose a dancer.

## ADD THIS CODE

Choose **brightness**  
from the menu.



Set the brightness to **-100** to  
make the sprite completely dark.



## TRY IT

Click the green flag to start.



Click the stop sign to stop.



# Interactive Dance

Press keys to switch dance moves.



# Interactive Dance

scratch.mit.edu



## GET READY



Go to the  
Sprite Library.



Click the Dance category



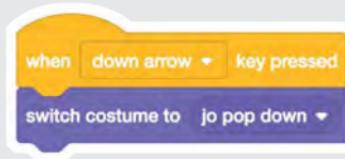
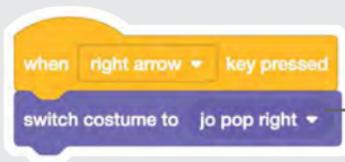
Choose a dancer.

## ADD THIS CODE

Choose a different key to press  
for each dance move.



Pick a dance move from the menu.



## TRY IT



Press the arrow keys on your keyboard.

# Color Effect!



Make the backdrop change colors.



# Color Effect

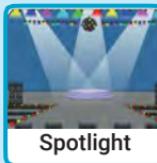
scratch.mit.edu



## GET READY

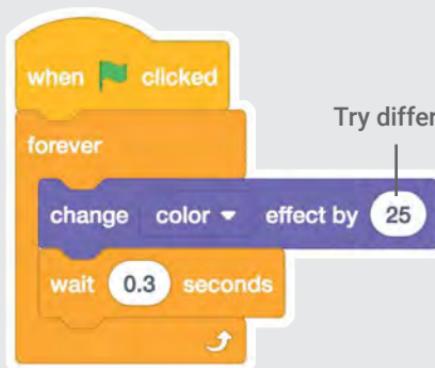


Choose a backdrop.



Spotlight

## ADD THIS CODE



Try different numbers.

## TRY IT

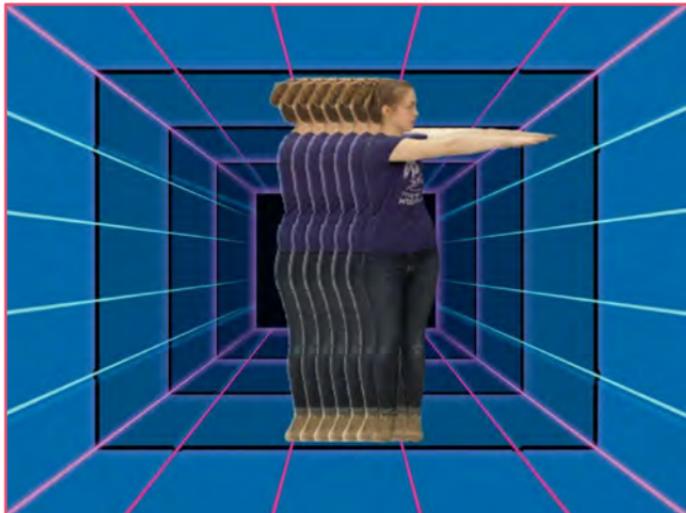
Click the green flag to start.



# Leave a Trail



Stamp a trail as your dancer moves.



# Leave a Trail

scratch.mit.edu



## GET READY

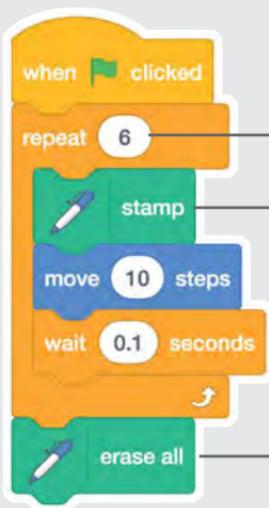
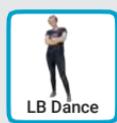


Choose a dancer from the Dance category.



Click the **Extensions** button, and then click **Pen** to add the blocks.

## ADD THIS CODE



Type how many times to repeat.

Stamp an image of the sprite on the Stage.

Clear all the stamps.

## TRY IT

Click the green flag to start.



# Jumping Game Cards



Make a character jump over moving obstacles.

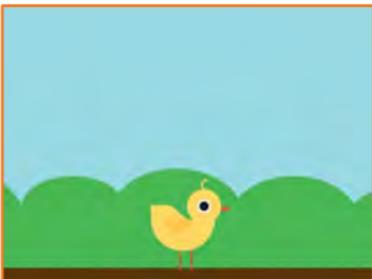
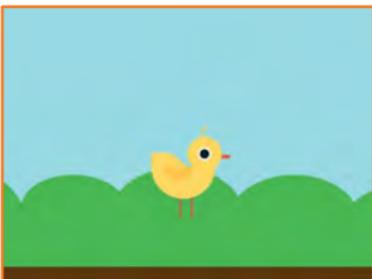
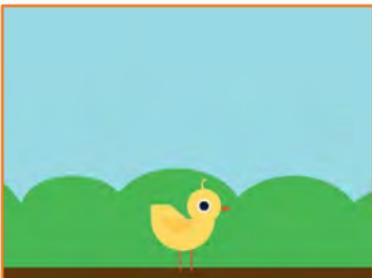
# Jumping Game Cards

Use these cards in this order:

- 1. Jump**
- 2. Go to Start**
- 3. Moving Obstacle**
- 4. Add a Sound**
- 5. Stop the Game**
- 6. Add More Obstacles**
- 7. Score**

# Jump

Make a character jump.



# Jump

scratch.mit.edu

## GET READY



Choose a backdrop.



Blue Sky



Choose a character,  
like Chick.



Chick

## ADD THIS CODE



Type a minus sign  
to go back down.

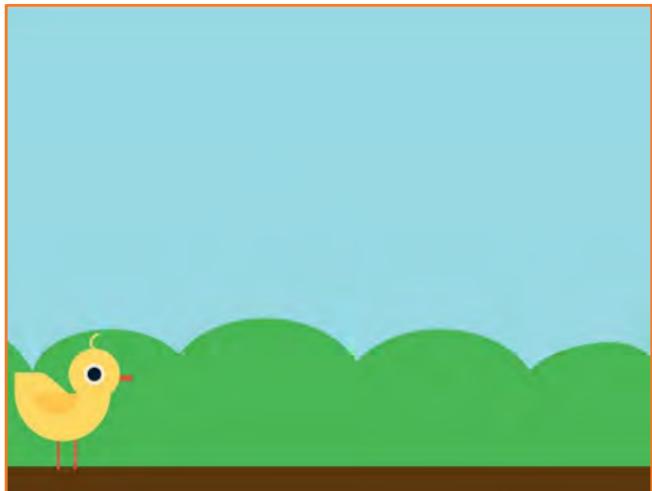
## TRY IT



Press the **space** key on your keyboard.

# Go to Start

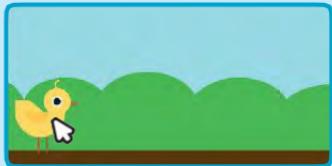
Set the starting point for your sprite.



# Go to Start

scratch.mit.edu

## GET READY



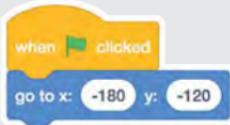
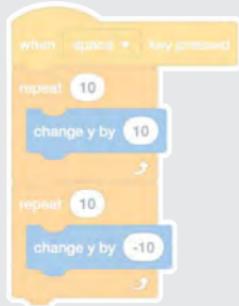
Drag your character to where you want it.



When you move your character, its **x** and **y** position will update in the blocks palette.

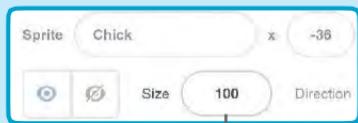
Now when you drag out a **go to** block, it will set to your character's new position.

## ADD THIS CODE



Set the starting position.  
(Your numbers may be different.)

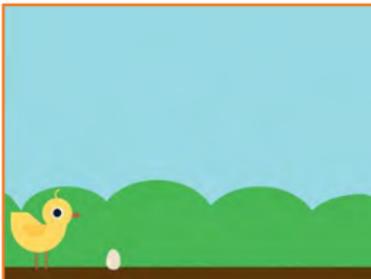
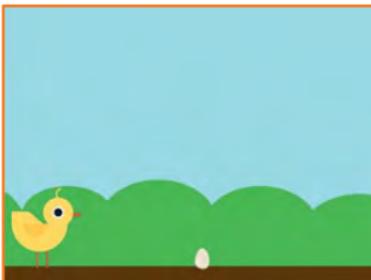
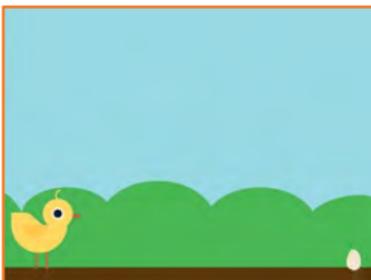
## TIP



Change the size of a sprite by typing a smaller or larger number.

# Moving Obstacle

Make an obstacle move across the Stage.



# Moving Obstacle

scratch.mit.edu

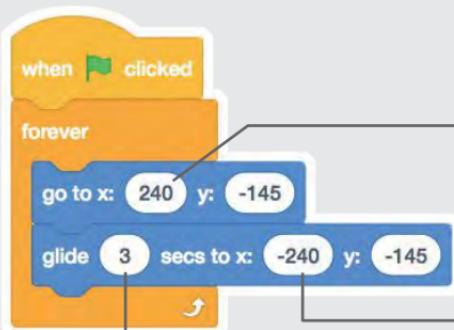
## GET READY



Choose a sprite to be an obstacle, such as Egg.



## ADD THIS CODE



Start at the right edge of the Stage.

Type a smaller number to go faster.

Glide to the left edge of the Stage.

## TRY IT

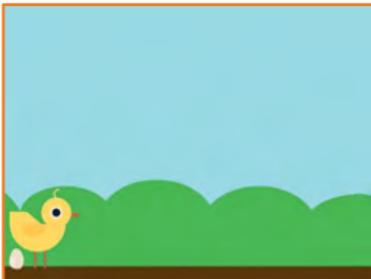
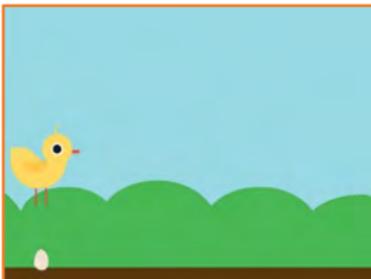
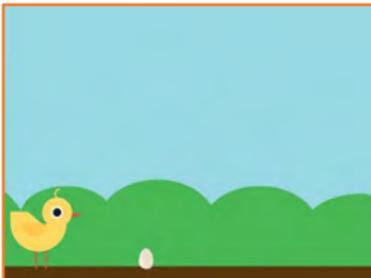
Click the green flag to start.



Press the space key on your keyboard.

# Add a Sound

Play a sound when your sprite jumps.

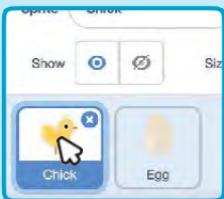


# Add a Sound

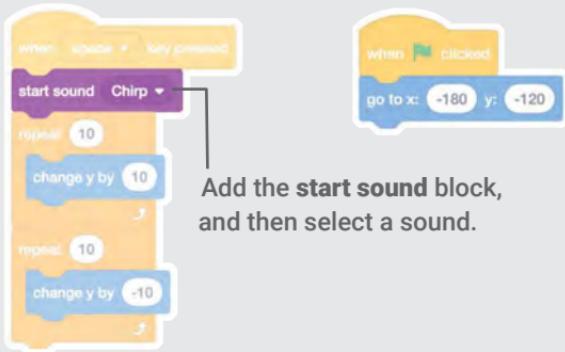
scratch.mit.edu

## GET READY

Click to select the Chick sprite.



## ADD THIS CODE



## TRY IT

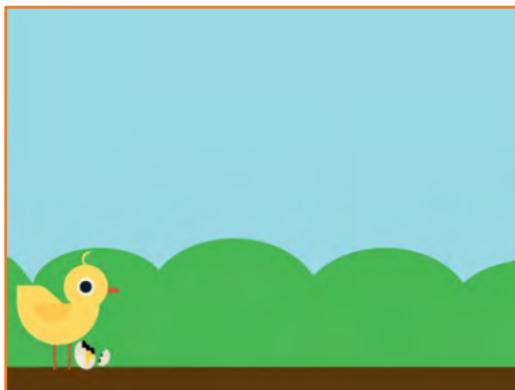
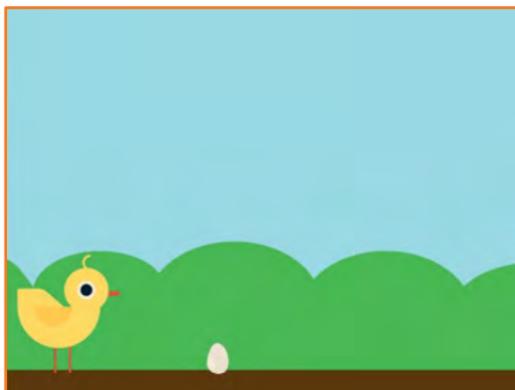
Click the green flag to start.



Press the **space** key on your keyboard.

# Stop the Game

Stop the game if your sprite touches the egg.

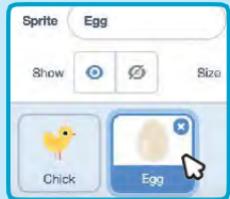


# Stop the Game

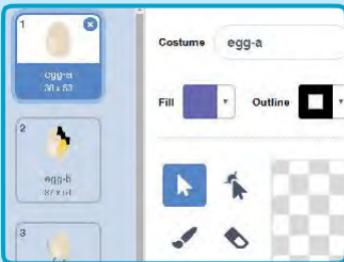
scratch.mit.edu

## GET READY

Click to select the Egg sprite.



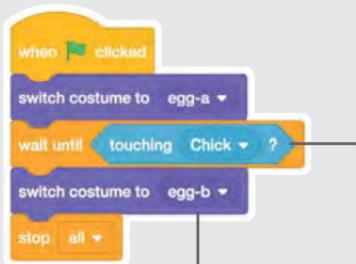
Click the Costumes tab to see the Egg sprite's costumes.



## ADD THIS CODE

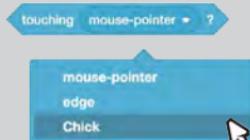


Click the Code tab and add this code.



Choose a second costume for the Egg sprite to change to.

Insert the touching block and choose Chick from the menu.



Click the green flag to start.

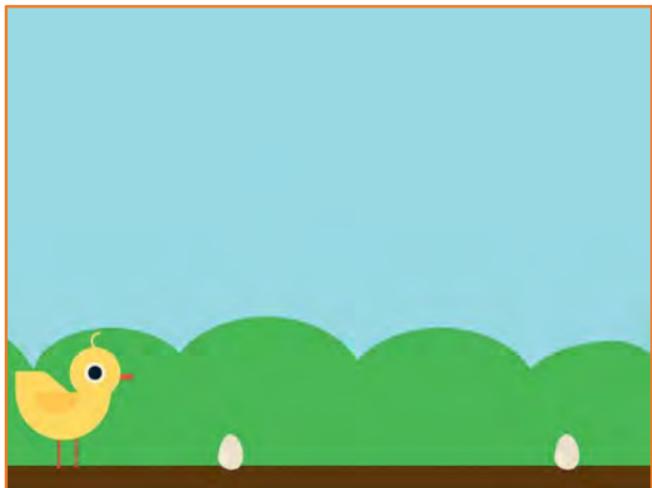


Press the space key on your keyboard.

# Add More Obstacles

v

Make the game harder by adding more obstacles.



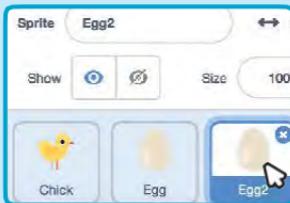
# Add More Obstacles

scratch.mit.edu

## GET READY

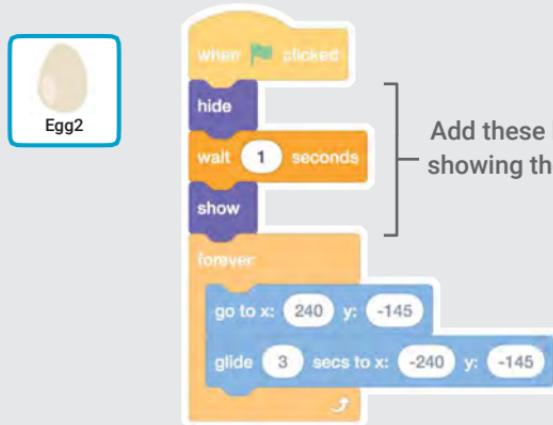


To duplicate the Egg sprite, right-click (Mac: control-click) on the thumbnail, and then choose **duplicate**.



Click to select Egg2.

## ADD THIS CODE



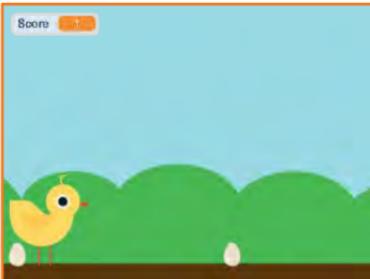
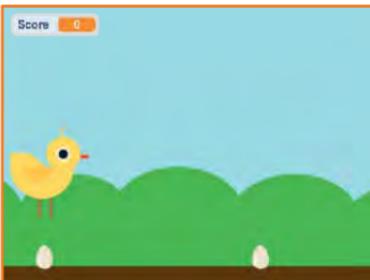
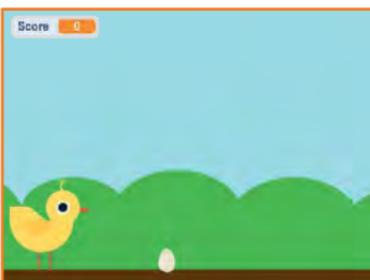
## TRY IT

Click the green flag to start.



# Score

Add a point each time your sprite jumps over an egg.



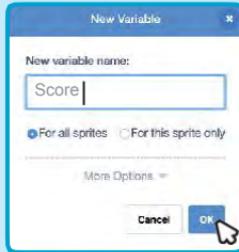
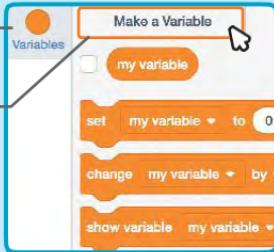
# Score

scratch.mit.edu

## GET READY

Choose Variables.

Click the Make a Variable button.



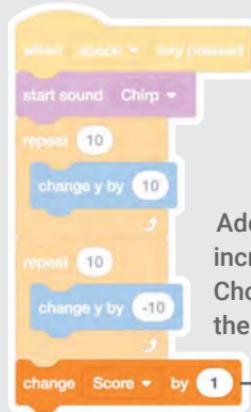
Name this variable **Score** and then click **OK**.

## ADD THIS CODE

Click the Chick sprite and add two blocks to your code:



Add this block to reset the Score. Choose **Score** from the menu.



Add this block to increase the score. Choose **Score** from the menu.

## TRY IT

Jump over the eggs to score points!



# Virtual Pet Cards



Create an interactive pet that can  
eat, drink, and play.



# Virtual Pet Cards

Use these cards in this order:

- 1. Introduce Your Pet**
- 2. Animate Your Pet**
- 3. Feed Your Pet**
- 4. Give Your Pet a Drink**
- 5. What Will Your Pet Say?**
- 6. Time to Play**
- 7. How Hungry?**

# Introduce Your Pet

Choose a pet and have it say hello.



# Introduce Your Pet

scratch.mit.edu



## GET READY



Choose a backdrop,  
like Garden Rock.



Garden



Choose a sprite to be  
your pet, like Monkey.



Monkey

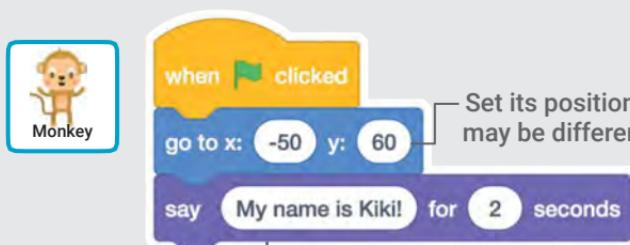
Pick a sprite with more  
than one costume.



Scroll over sprites in the  
Sprite Library to see their  
different costumes.

## ADD THIS CODE

Drag your pet to where you want it on the Stage.



Type what you want your pet to say.

## TRY IT

Click the green flag to start.



# Animate Your Pet

Bring your pet to life.



# Animate Your Pet

scratch.mit.edu

## GET READY



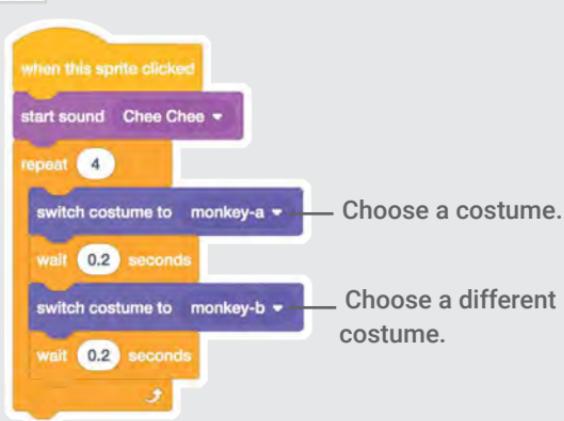
Click the **Costumes** tab to see your pet's costumes.



## ADD THIS CODE



Click the **Code** tab and add this code.



## TRY IT

Click your pet.



# Feed Your Pet

Click the food to feed your pet.



# Feed Your Pet

scratch.mit.edu



## GET READY

### Sounds

Click the Sounds tab.



Monkey

Choose a sound from the Sounds Library, like Chomp.



Bananas

Choose a food sprite, like Bananas.

## ADD THIS CODE

### Code

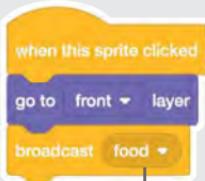
Click the Code tab.



Bananas

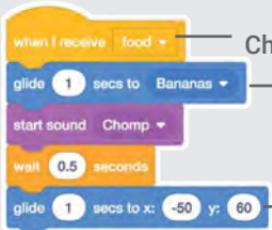


Select New message and name it food.



Broadcast the food message.

Select your pet.



Choose food from the menu.

Choose Bananas from the menu.

Glide to the starting position.

## TRY IT

Click the food.



# Give Your Pet a Drink

Give your pet some water to drink.



# Give Your Pet a Drink

scratch.mit.edu



## GET READY



Choose a drink sprite, like Glass Water.



## ADD THIS CODE



when this sprite clicked

go to front ▾ layer

broadcast drink ▾

wait 1 seconds

switch costume to glass water-b ▾

start sound Water Drop ▾

wait 1 seconds

switch costume to glass water-a ▾

Broadcast a new message.

Switch to the empty glass.

Switch to the full glass.

Tell your pet what to do when it receives the message.



when I receive drink ▾

glide 1 secs to Glass Water ▾

wait 1 seconds

glide 1 secs to x: -50 y: 60

Choose drink from the menu.

Choose Glass Water from the menu.

Glide to the starting position.

## TRY IT

Click the drink to start.



# What Will Your Pet Say?

Let your pet choose what it will say.



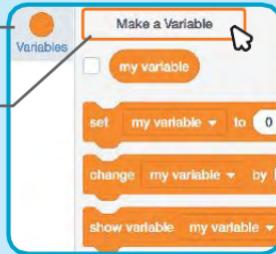
# What Will Your Pet Say?

scratch.mit.edu

## GET READY

Choose Variables.

Click the Make a Variable button.

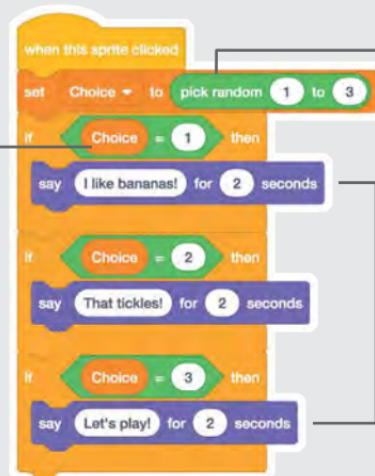


Name this variable Choice and then click OK.

## ADD THIS CODE



Insert the Choice block into the equals block from the Operators category.



Insert the pick random block.

Type things for your pet to say.

## TRY IT

Click your pet to see what it says.



# Time to Play

Have your pet play with a ball.



# Time to Play

scratch.mit.edu

## GET READY



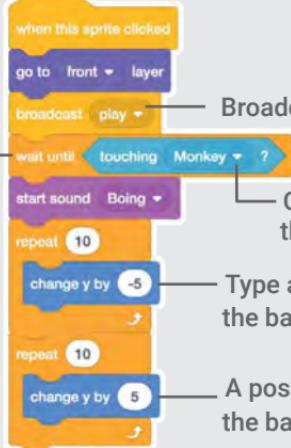
Choose a sprite,  
like Ball.



## ADD THIS CODE



Insert the touching block  
into the wait until block.



Broadcast a new message.

Choose Monkey from  
the menu.

Type a minus sign to make  
the ball move down.

A positive number makes  
the ball move up.



Choose play from the menu.

Pick Ball from the menu.

## TRY IT

Click the ball.



# How Hungry?

Keep track of how hungry your pet is.

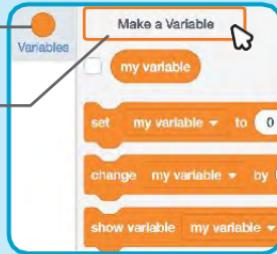


# How Hungry?

scratch.mit.edu

## GET READY

Choose Variables.

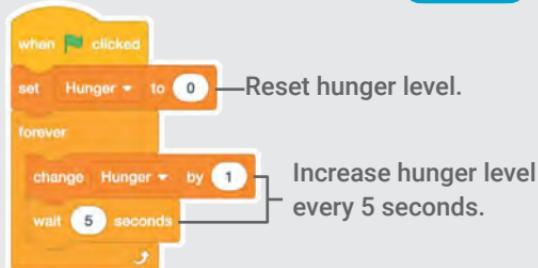


Click the Make a Variable button.



Name this variable Hunger and then click OK.

## ADD THIS CODE



Choose food from the menu.



Type a minus sign to make your pet less hungry when it gets food.

## TRY IT

Click the green flag to start.



Then click the food.



# Catch Game Cards



Make a game where you catch things falling from the sky.



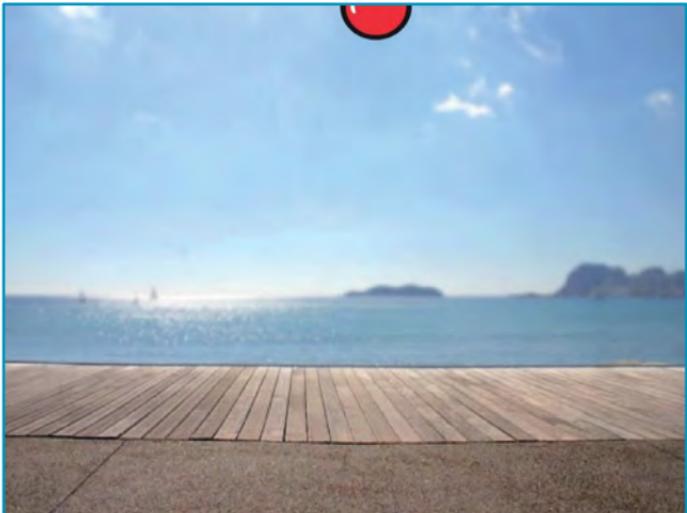
# Catch Game Cards

Use these cards in this order:

1. Go to the Top
2. Fall Down
3. Move the Catcher
4. Catch It!
5. Keep Score
6. Bonus Points
7. You Win!

# Go to the Top

Start from a random spot at the top of the Stage.



# Go to the Top

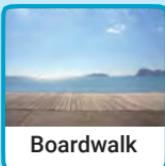
scratch.mit.edu



## GET READY



Choose a backdrop,  
like Boardwalk.



Boardwalk



Choose a sprite,  
like Apple.



Apple

## ADD THIS CODE



Type 180 to go to the top of the stage.

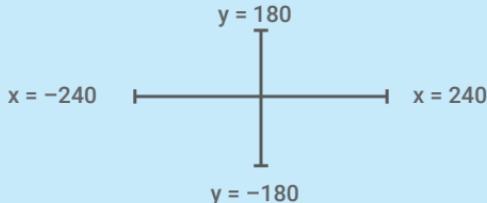
## TRY IT

Click the green flag to start.



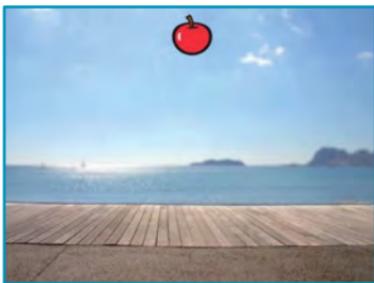
## TIP

y is the position on the Stage from top to bottom.



# Fall Down

Make your sprite fall down.



# Fall Down

scratch.mit.edu

## GET READY



Click to select the Apple sprite.

## ADD THIS CODE

Keep the previous code as is, and add this second stack of blocks:



Insert the **y position** block into this block from the Operators category.



Type a minus sign to fall down.

Check if near the bottom of the Stage.

Go back to the top of the Stage.

## TRY IT

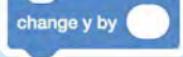
Click the green flag to start.



Click the stop sign to stop.

## TIP

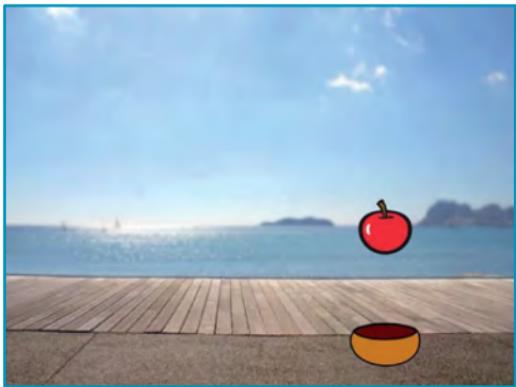
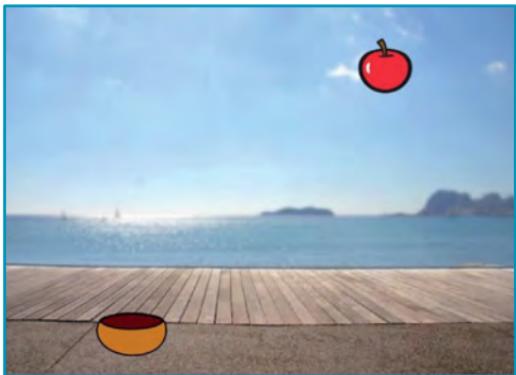
Use



to move up or down.

# Move the Catcher

Press the arrow keys so that the catcher moves left and right.



# Move the Catcher

scratch.mit.edu



Choose a catcher like Bowl.



Drag the bowl to the bottom of the Stage.

## ADD THIS CODE



```
when green flag clicked
forever
  if key right arrow pressed? then
    change x by 10
  end
  if key left arrow pressed? then
    change x by -10
  end
```

Choose the **right arrow** from the menu.

Choose the **left arrow** from the menu.

## TRY IT

Click the green flag to start.



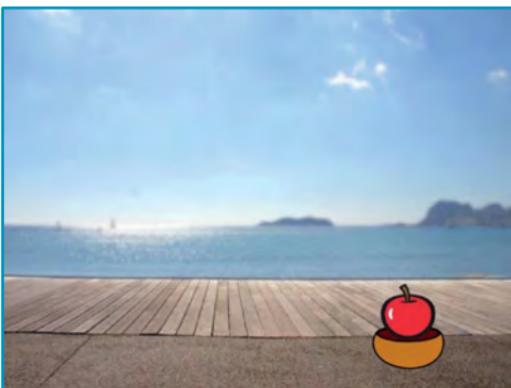
Press the arrow keys to move the catcher.

# Catch It!

Catch the falling sprite.



□)

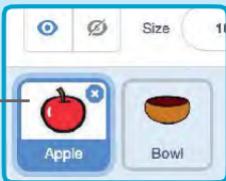


# Catch It!

scratch.mit.edu

## GET READY

Click to select  
the Apple.



## ADD THIS CODE



Choose Bowl from the menu.

Choose a sound.

## TIP



Click the **Sounds** tab if  
you want to add a  
different sound.



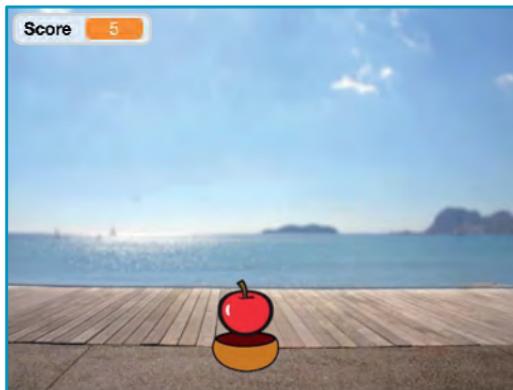
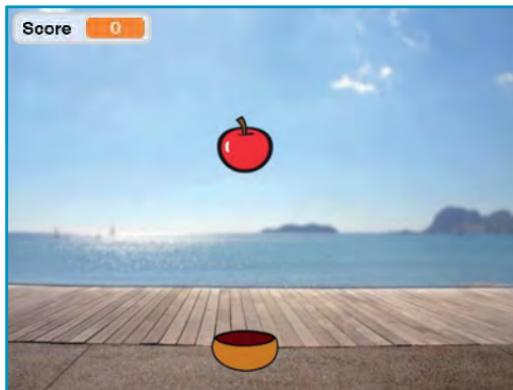
Then choose a sound  
from the Sounds Library.



Click the **Code** tab  
when you want to  
add more blocks.

# Keep Score

Add a point each time you catch the falling sprite.



# Keep Score

scratch.mit.edu

## GET READY

Choose Variables.

Click the Make a Variable button.



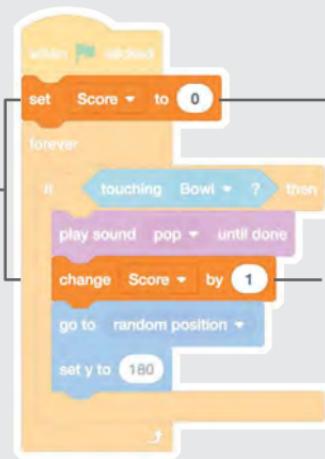
Name this variable Score and then click OK.

## ADD THIS CODE

Add two new blocks to your code:



Choose Score from the menu.



Add this block to reset the score.

Add this block to increase the score.

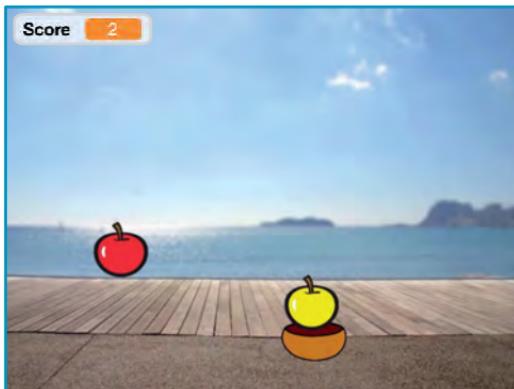
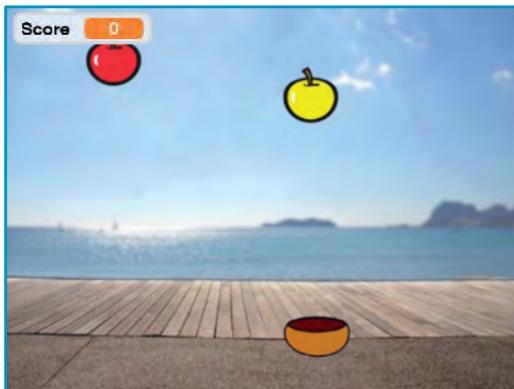
## TRY IT



Click the green flag to start.  
Then, catch apples to score points!

# Bonus Points

Get extra points when you catch a golden sprite.



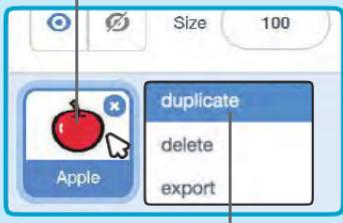
# Bonus Points

scratch.mit.edu



## GET READY

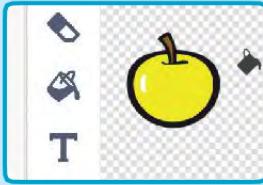
To duplicate your sprite,  
right-click (Mac: control-click).



Choose **duplicate**.

Costumes

Click the Costumes tab.

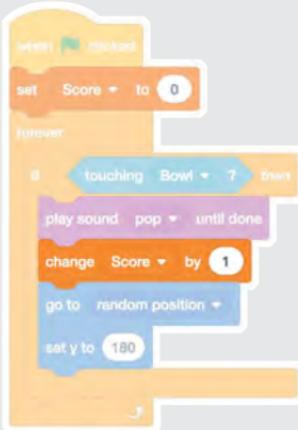


You can use the paint tools to make  
your bonus sprite look different.

## ADD THIS CODE

Code

Click the Code tab.



Type how many points you get  
for catching a bonus sprite.

## TRY IT

Catch the bonus sprite to increase your score!

# You Win!

When you score enough points, display a winning message!

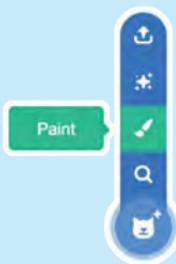


# You Win!

scratch.mit.edu



## GET READY



Click the **Paint** icon  
to make a new sprite.

Use the **Text** tool to write a message, like "You Win!"

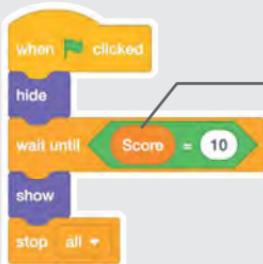


You can change the font color, size, and style.

## ADD THIS CODE



Click the **Code** tab.



Insert the **Score** block from the Variables category.

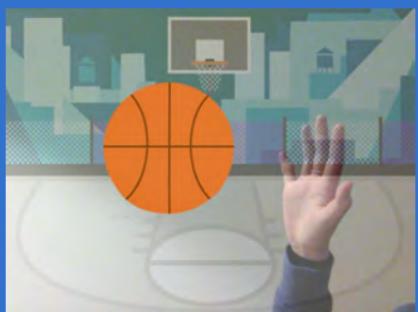
## TRY IT

Click the green flag to start.



Play until you score enough points to win!

# Video Sensing Cards



Interact with projects using video sensing.



# Video Sensing Cards

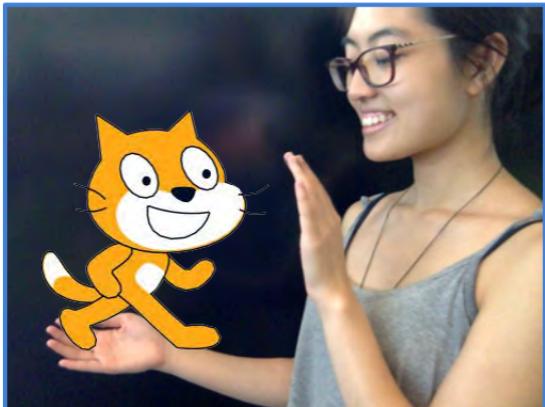
Try these cards in any order:

- Pet the Cat
- Animate
- Pop a Balloon
- Play the Drums
- Keep Away Game
- Play Ball
- Start an Adventure

# Pet the Cat



Make the cat meow when you touch it.



# Pet the Cat

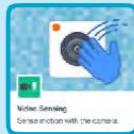
scratch.mit.edu



## GET READY

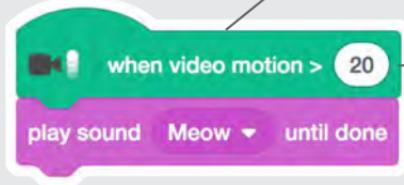


Click the Extensions button  
(at the bottom left of the screen).



Choose Video Sensing  
to add the video blocks.

## ADD THIS CODE



This will start when it  
senses video motion on  
a sprite.

Type a number between  
1 and 100 to change the  
sensitivity.

1 will start with very little  
movement, 100 requires  
a lot of movement.

## TRY IT

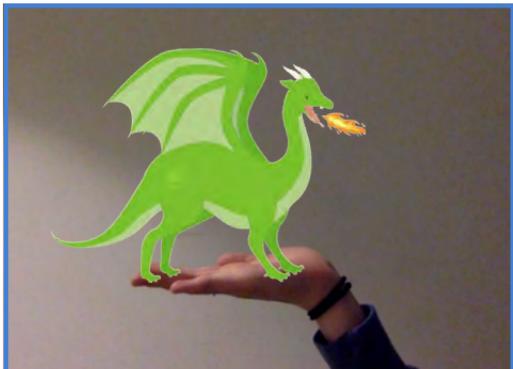
Move your hand to pet the cat.



# Animate



Move around to bring a sprite to life.



# Animate

scratch.mit.edu



## GET READY



Click the Extensions button, then choose Video Sensing.



Choose a sprite to animate.



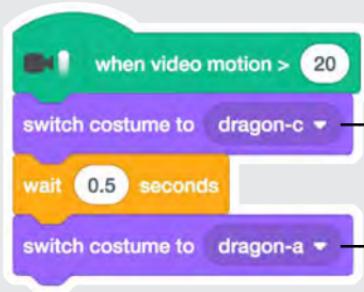
Dragon

Pick a sprite with more than one costume.



Scroll over sprites in the Sprite Library to see their different costumes.

## ADD THIS CODE



Choose one costume.

Choose a different costume.

## TRY IT

Move around to animate the dragon.



# Pop a Balloon



Use your finger to pop a balloon.



# Pop a Balloon

scratch.mit.edu



## GET READY

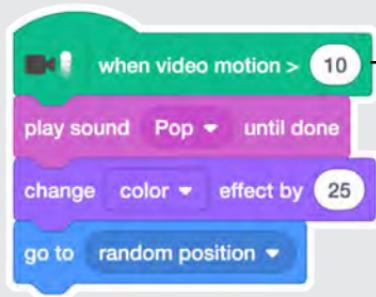


Click the Extensions button,  
then choose **Video Sensing**.



Choose a sprite, like Balloon1.

## ADD THIS CODE



Type a larger number to  
make it harder to pop.

## TRY IT

Use your finger to pop the balloon.



# Play the Drums



Interact with sprites that play sounds.



# Play the Drums

scratch.mit.edu



## GET READY



Click the Extensions button,  
then choose Video Sensing.



Choose two sprites,  
like Drum and Drum-cymbal.

## ADD THIS CODE

Click on a drum to select it, then add its code.



```
when video motion > 10
set size to 100 %
change size by 20
start sound [High Tom v]
wait [0.1] seconds
change size by -20
```

Type a minus sign to get  
smaller.



```
when video motion > 10
switch costume to [drum-cymbal-a v]
start sound [Crash Cymbal v]
wait [0.1] seconds
switch costume to [drum-cymbal-b v]
```

Choose a different costume.

## TRY IT

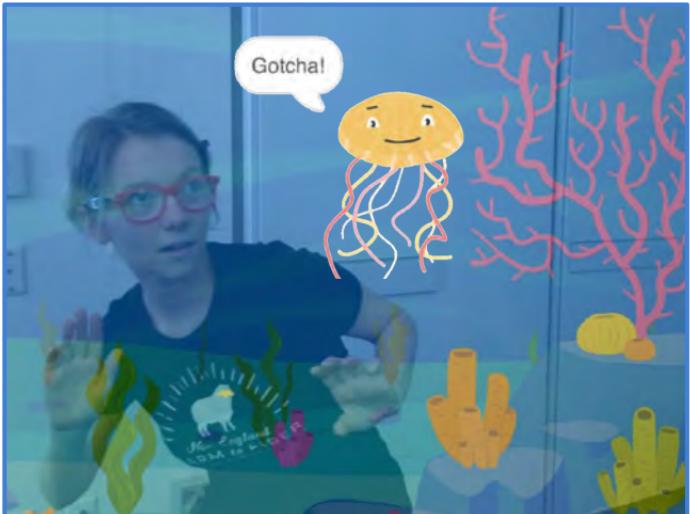
Use your hands to play the drums!



# Keep Away Game



Move around to avoid a sprite.

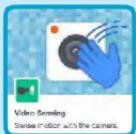


# Keep Away Game

scratch.mit.edu



## GET READY



Click the **Extensions** button, then choose **Video Sensing**.

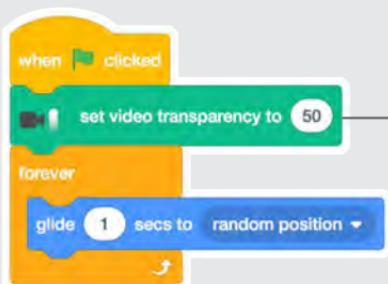


Choose a backdrop, like Ocean.

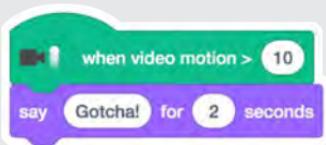


Choose a sprite, like Jellyfish.

## ADD THIS CODE



Type a number between 0 and 100.  
(0 to show the video, 100 to make the video transparent.)



## TRY IT

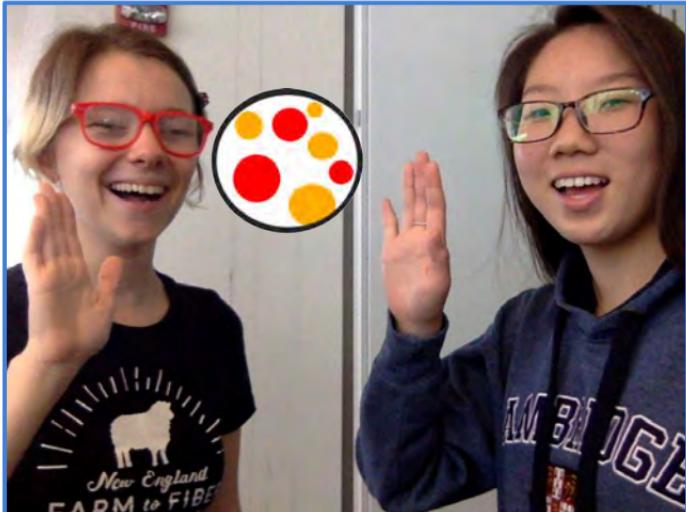
Move around to avoid the jellyfish.



# Play Ball



Use your body to move a sprite across the screen.

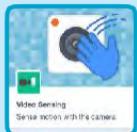


# Play Ball

scratch.mit.edu



## GET READY

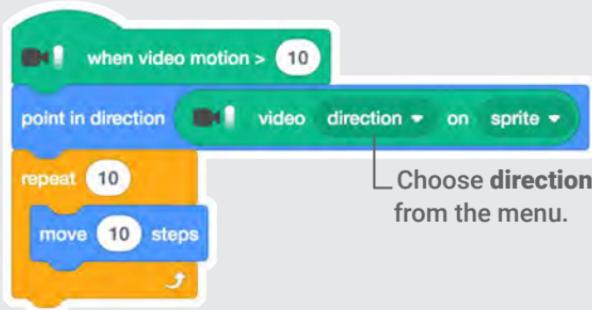
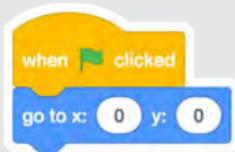


Click the Extensions button,  
then choose Video Sensing.



Choose a sprite, like Beachball.

## ADD THIS CODE



Choose direction  
from the menu.

## TRY IT

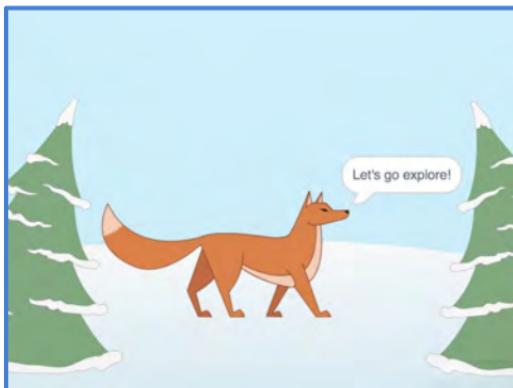
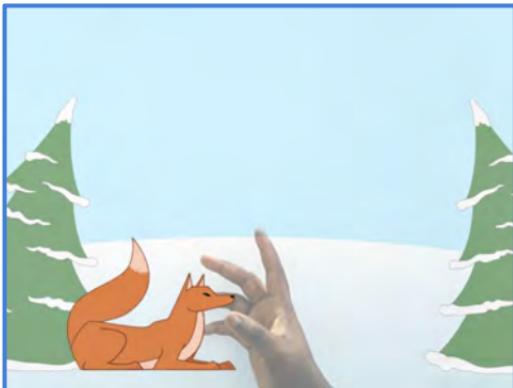


Use your hands to push the beach ball around the screen.  
Try it with a friend!

# Start an Adventure!



Interact with a story by  
moving your hands.



# Start an Adventure!

scratch.mit.edu



## GET READY



Click the Extensions button.



Choose Video Sensing.



Choose a backdrop.



Winter



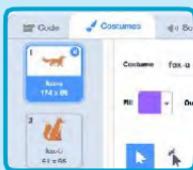
Choose a sprite.



Fox



Click the Costumes tab to see your sprite's other costumes.



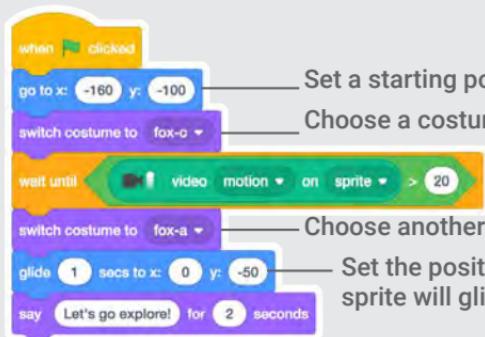
## ADD THIS CODE



Click the Code tab.



Insert the video motion on sprite block into the greater than block from the Operators category.



Set a starting point.

Choose a costume.

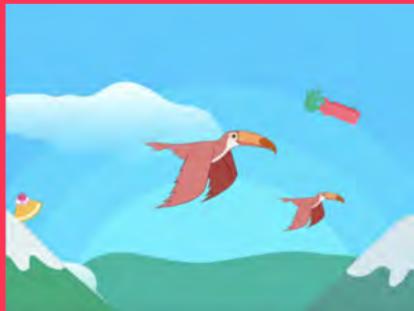
Choose another costume.

Set the position your sprite will glide to.

## TRY IT

Click the green flag. Then wave to wake up the fox.

# Make it Fly Cards



Choose any character and make it fly!



# **Make it Fly**

## **Cards**

Use these cards in this order:

- 1. Choose a Character**
- 2. Start Flying**
- 3. Switch Looks**
- 4. Make it Interactive**
- 5. Floating Clouds**
- 6. Flying Hearts**
- 7. Collect Points**

# Choose a Character

Choose a character to fly.



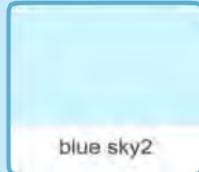
# Choose a Character

scratch.mit.edu

## GET READY



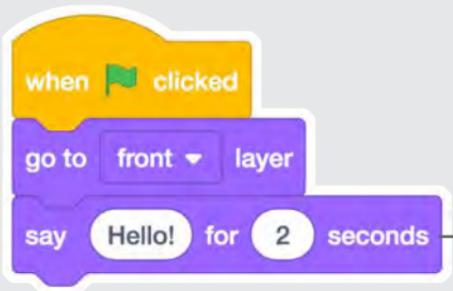
Choose a backdrop,  
such as "blue  
sky2".



Choose a sprite from  
the Flying theme.



## ADD THIS CODE



Type what you want  
your sprite to say.

## TRY IT

Click the green flag to start



# Start Flying

Move the scenery so your character looks like it's flying.



# Start Flying

scratch.mit.edu



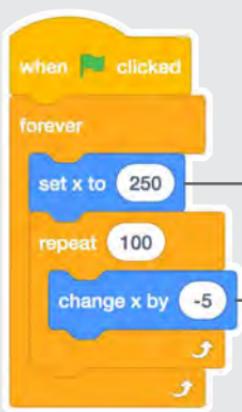
## GET READY



Choose a sprite to fly by, such as Buildings



## ADD THIS CODE

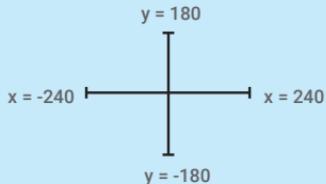


Start from the right end of the stage.

Type a negative number to move left.

## TIP

x is the position on the Stage from left to right.



# Switch Looks

Add variety to your scenery.



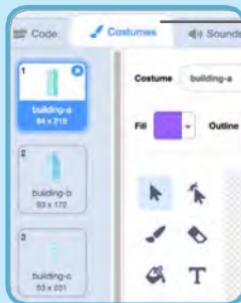
# Switch Looks

scratch.mit.edu



## GET READY

Click to select the Buildings sprite.

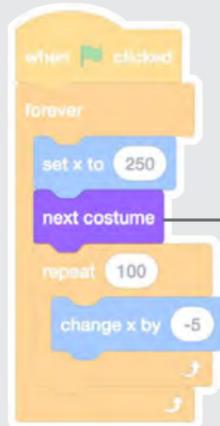


Then, click the Costumes tab to see the different building costumes.

## ADD THIS CODE



Click the tab.



Add this block to switch costumes.

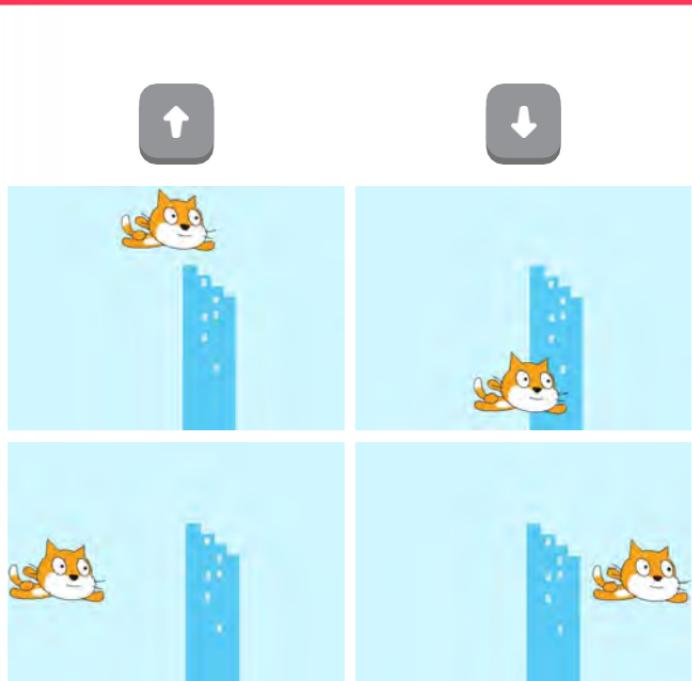
## TRY IT

Click the green flag to start



# Make It Interactive

Make your character move  
when you press a key.

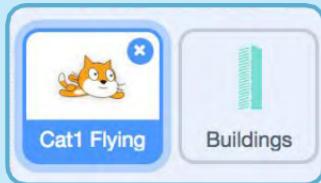


# Make It Interactive

scratch.mit.edu

## GET READY

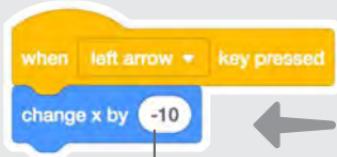
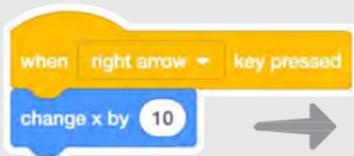
Click to select your flying sprite.



## ADD THIS CODE

### Change x

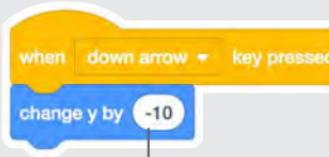
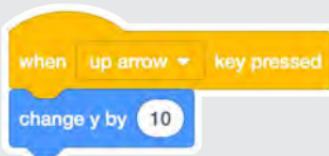
Move your character *side to side*.



Type a minus sign to move *left*.

### Change y

Move your character *up and down*.



Type a minus sign to move *down*.

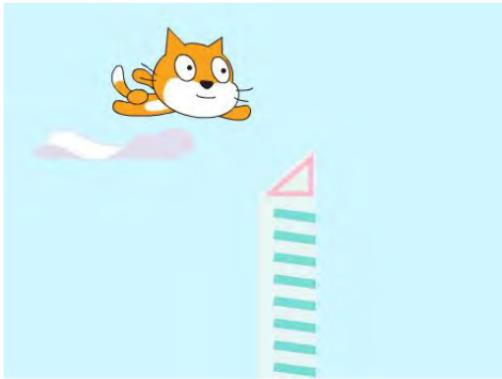
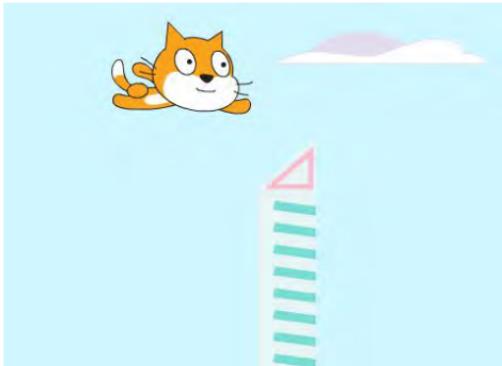
## TRY IT



Press the arrow keys on your keyboard to move your character around.

# Floating Clouds

Make clouds float by in the sky!



# Floating Clouds

scratch.mit.edu

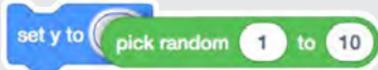
## GET READY



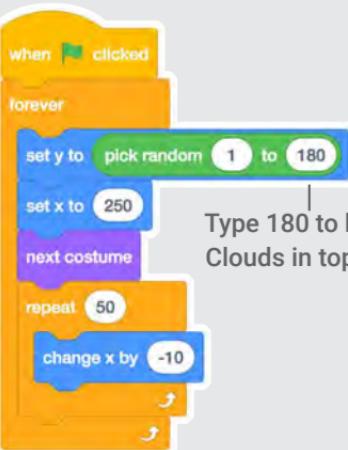
Choose Clouds from the library.



## ADD THIS CODE



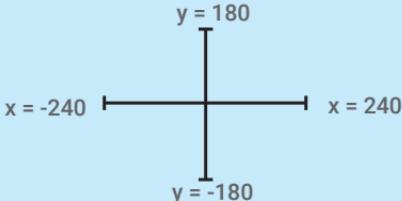
Drag the **pick random** block into the **set y to** block.



Type 180 to keep  
Clouds in top half.

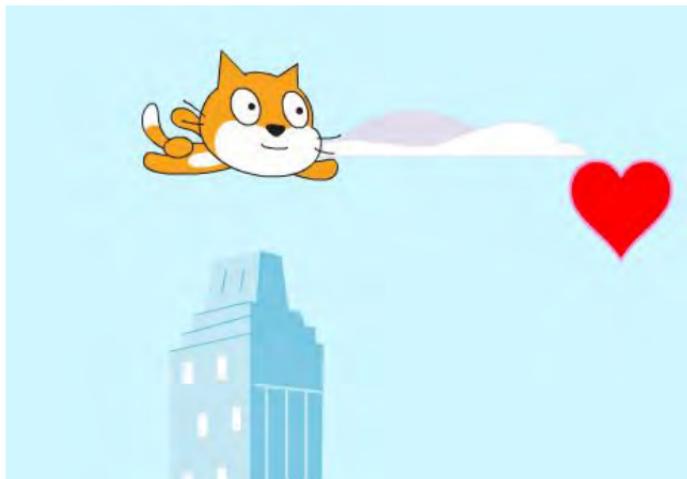
## TIP

y is the position on the Stage from top to bottom.



# Flying Hearts

Add hearts or other floating objects to



# Flying Hearts

scratch.mit.edu



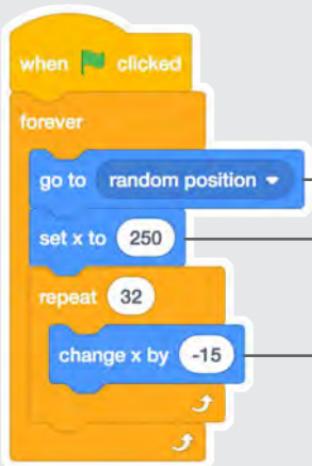
## GET READY



Choose a sprite, such as Heart.



## ADD THIS CODE



Moves the sprite up and down

Sets your sprite's position at the far right of the stage

Moves the sprite across the stage

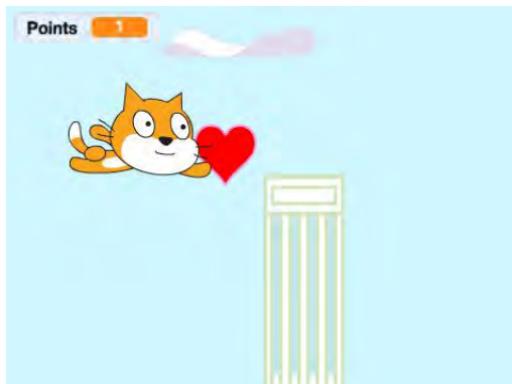
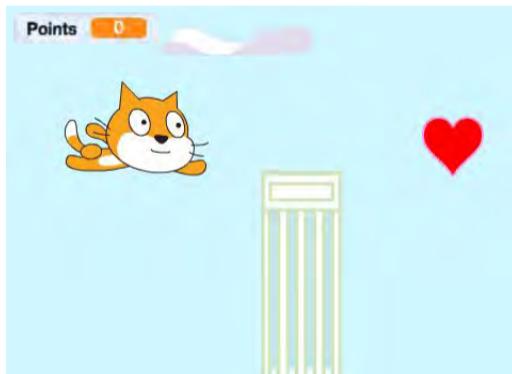
## TRY IT

Click the green flag to start



# Collect Points

Add a point each time you touch a heart or other object.



# Collect Points

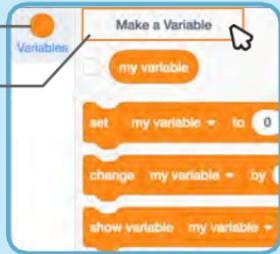
scratch.mit.edu



## GET READY

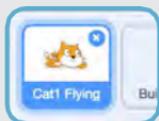
Select Variable

Click the Make a Variable button.



Name this variable Points and then click OK.

## ADD THIS CODE



Select your flying sprite.



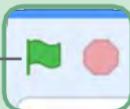
Resets points at the start.

Choose Heart from the menu.

Add a point.

## TRY IT

Click the green flag to start





# Create a Sprite



Explore digital drawing, remixing, or uploading to create original sprites



# Cards in This Pack

- Design Your Sprite
- Using the Paint Editor
- Options to Customize Sprites
- Create a Sprite by Remixing
- Bring Your Drawings into Scratch
- Animate Your Sprite
- Code the Sprite
- Create an Asset Pack
- Collaborate: Export or Backpack /  
Collaborate: Remix

# Design Your Sprite



What sprite do you want to create? When you are brainstorming ideas, ask yourself:

- What items are unique to your culture, community, language, or location that would be fun to animate in Scratch or share with your peers?
- What is your favorite activity or hobby? Food? Native animal or family pet? Native plant? Item of dress? Book character?
- Is there already a sprite in the library that you'd want to remix or change?

*Sprite examples by pondermake, SaffronChai, Chumie, algorithmar, and watse166.*

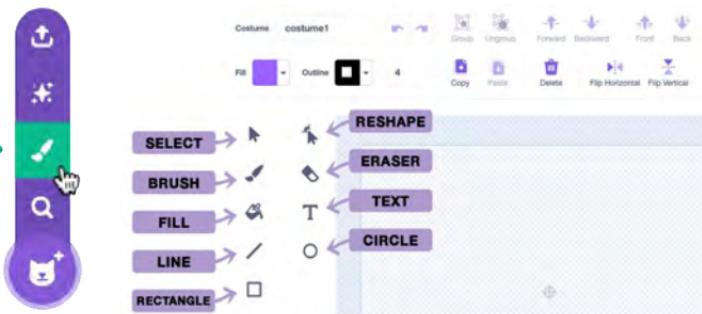
# Design Your Sprite



- There are two modes for using the Paint Editor in Scratch:
  - Vector-mode allows you to create and edit shapes (Scratch default).
  - Bitmap-mode allows you to edit photos and paint with pixels.
- We recommend using vector-mode, when drawing sprites, as it allows other users to make adjustments and add and remove elements if they remix your creations.

# Using the Paint Editor

## TOOLS TO TRY



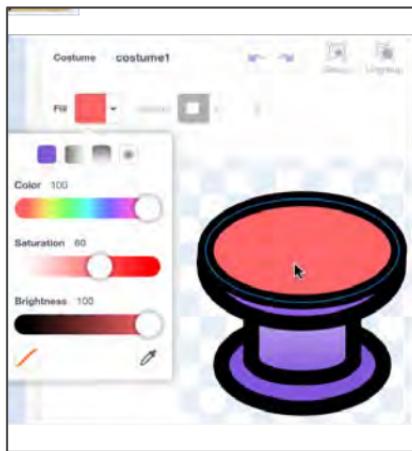
	Click and drag with the Line, Circle, or Rectangle tools to <b>create a shape</b> . Hold down the Shift key while dragging to create equal sides, or 45 and 90 degree angles with lines.
	Using the Select tool, select a shape and click and drag one of the corner points to <b>resize</b> it.
	To <b>rotate</b> a shape once you've made it, use the Select tool to grab the anchor under the shape and drag it. Hold down the Shift key while dragging to rotate at 45 degree angles.
	Using the Reshape tool, click on one of the points of a shape and <b>move the point</b> around to alter the shape. Click + Shift key to select and move multiple points at once.
	Using the Reshape tool, click on a part of the shape that doesn't have a point to <b>add a new point</b> , or click on a point and press "Delete" to <b>remove a point</b> .

# Using the Paint Editor

scratch.mit.edu

 Curved	Using the Reshape tool, click on a point and choose whether it is <b>curved or pointed</b> . Click on a point and drag rotate the handles attached to the point to <b>alter the shape of a curve</b> .
 Copy	Using the Select tool, select a shape and click the buttons on the top menu to <b>copy and paste</b> a duplicate.
 Flip Vertical	Using the Select tool, select a shape and click the flip horizontal or flip vertical buttons on the top menu to <b>flip</b> a shape.
 Forward	Using the Select tool, select a shape and click the Forward, Backward, Front, or Back buttons to change the <b>layer order</b> .
	Select the fill from the dropdown and use the fill (paint bucket) tool to adjust a shape's color. Or using the Select tool, select a shape and then use the Fill and Outline dropdowns to adjust the <b>color, saturation, brightness, and outline</b> . You can also choose to use a <b>gradient</b> . Use the eyedropper to select a color from another shape. Use the red strikethrough to fill with no color.
 Group	Using the select tool and holding down the "Shift" key, select multiple shapes to <b>group</b> them (helpful to move several shapes together).
	Use the brush tool for <b>freehand line drawing</b> . The example to the right shows hand drawn whiskers.
	Use the <b>eraser</b> tool to remove parts of the drawing from <i>all</i> shapes and layers it comes into contact with when clicking and dragging. You can use the reshape tool to then adjust the new points created.
 T	The <b>text</b> tool comes with a dropdown list of font options to choose from, and Fill and Outline dropdowns to change text color and outline.

# Options to Customize Sprites



point in direction 90

set [color ▾] effect to 50

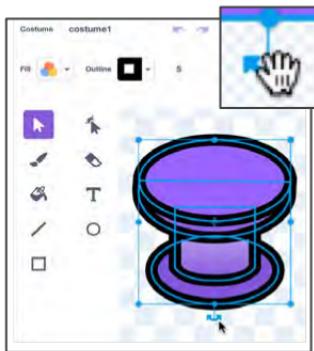
set size to 300 %

- Let's explore different ways to rotate sprites, change their size, change their color or brightness, etc.
- We can adjust the look of sprite costumes using the Paint Editor tools.
- Or we can adjust sprites using code blocks.

# Options to Customize Sprites

scratch.mit.edu

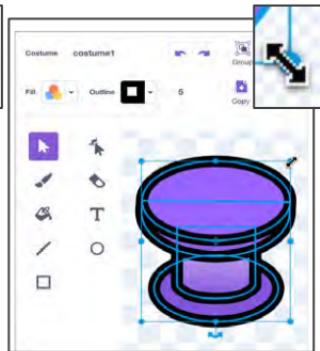
## EDIT THE COSTUME IN THE PAINT EDITOR



Rotate with Select



Recolor with Fill



Resize with Select

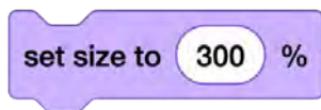
## ADJUST THE SPRITE WITH CODE



Experiment!



Do you notice any differences between using these code blocks to adjust a sprite versus using the Paint Editor tools above?



What happens if you use both methods?

# Create a Sprite by Remixing



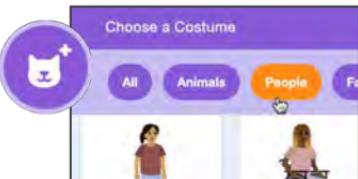
- Remixing parts of existing sprites could make creating a new unique sprite faster and easier.
- The sprite library contains a mix of bitmap and vector sprites. You can remix and re-imaging either type of sprite, but for this exercise, we are going to focus on vector sprites because they are easier to edit and customize, mix and match.

*Sprite remix examples by algorithmar, bgordi0077, RealAimkidBunni, and Chumie.*

# Create a Sprite by Remixing

scratch.mit.edu

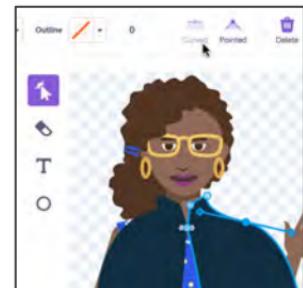
Choose two or more vector sprites with elements you like. Remember, some sprites have multiple costumes with elements/poses.



Recolor with Fill



Resize with Select



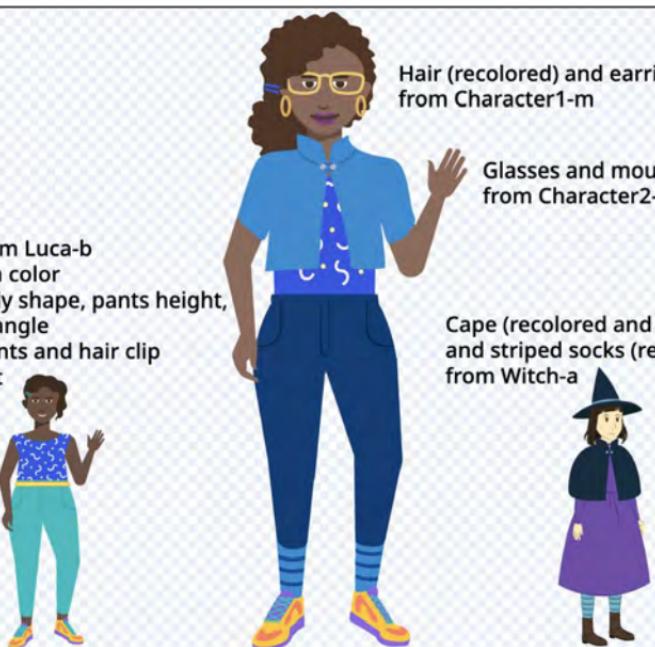
Use Reshape

Main Body from Luca-b  
- changed skin color  
- adjusted body shape, pants height, and hairclip angle  
- recolored pants and hair clip  
- removed belt

Hair (recolored) and earrings from Character1-m

Glasses and mouth (recolored) from Character2-a

Cape (recolored and reshaped) and striped socks (recolored) from Witch-a



# Bring Your Drawings into Scratch



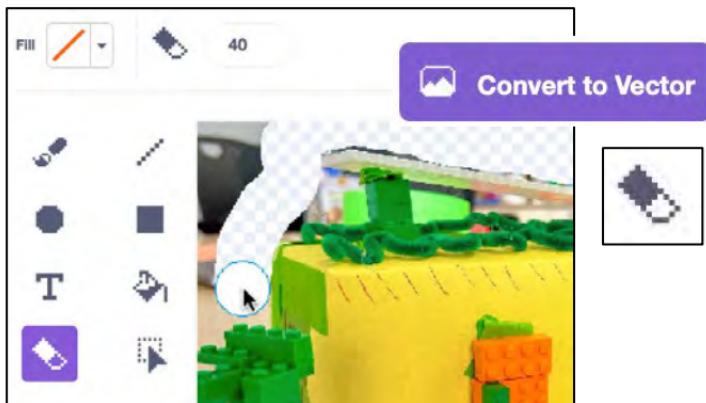
You can upload an original hand-drawn image or photograph to create a sprite. Keep in mind:

- You can choose a JPG, PNG, or SVG file.
- Keep each of your files under 10MB.
- Do not upload materials under copyright.
- Be sure that your upload follows the Community Guidelines and does not reveal personal information (like a photo with your face).

*Sprite example drawings by algorithmar's two daughters.*

# Bring Your Drawings In

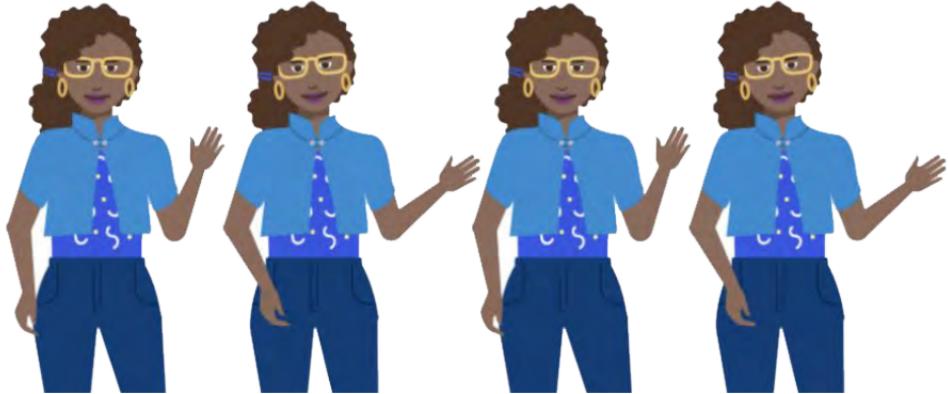
scratch.mit.edu



*Options to Remove the Background:*

- Before you upload the file, use **online tools or software**
- Use the **tools in the Scratch Paint Editor** after a file has been uploaded
  - In bitmap-mode, use the eraser tool to remove the image background or other pieces you don't want from your image.
  - You'll know you are in bitmap-mode when you see the "Convert to Vector" button at the bottom of the screen.
- You can choose to convert it to vector when done using the "Convert to vector" button to more easily rotate or resize, if desired.

# Animate Your Sprite



There are many ways you might choose to animate your character. For instance:

- Try moving or gliding your character to a new location.
- Try changing the direction of the character to tilt back and forth.
- Or try adding additional costume drawings to change the position of certain elements and create movement as costumes are changed. Flip this card over for more.

# Animate Your Sprite

scratch.mit.edu

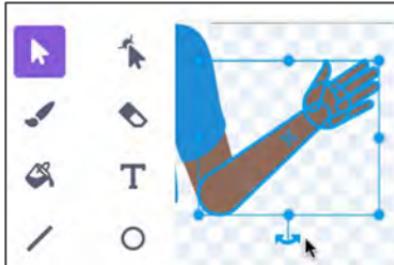
## GET READY

Duplicate your sprite costume on the costume tab.  
(Right click, Duplicate.)

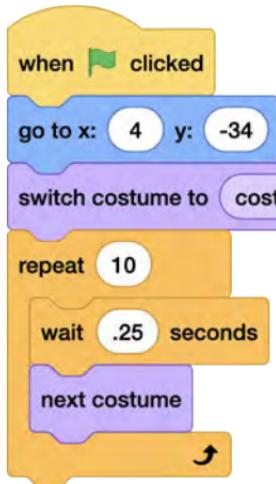


Use the select tool, then click and drag on the canvas to select multiple items.

Try rotating and moving incrementally.



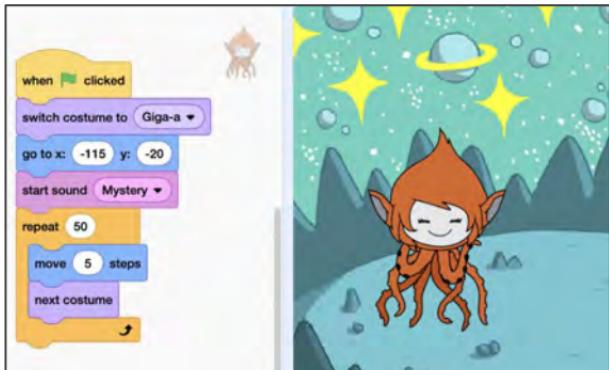
## ADD CODE



There are many ways to animate. Try looping through the costumes.



# Code Your Sprite



Click the Code tab, then try adding a few blocks! A great place to find tips for getting started, tutorials, Scratch Coding Cards, and more is the Scratch Ideas page ([scratch.mit.edu/ideas](http://scratch.mit.edu/ideas)). Try:

- using blocks to hear or see what you want to say on the stage
- adding text or custom backgrounds
- using motion blocks to give the sprite movement
- using event blocks (like “broadcast” or “when clicked”) to trigger action or make the project interactive.

# Code Your Sprite

scratch.mit.edu

## BLOCKS TO TRY

when green flag clicked

next costume

next backdrop

when this sprite clicked

when space key pressed

when I receive

message1

broadcast

message1

play sound

My Recording

until done

say

Hello!

for

2

seconds

speak

hello

think

Hmm...

for

2

seconds

turn



15 degrees

glide

1

secs to x:

0

y:

0

change

color

effect by

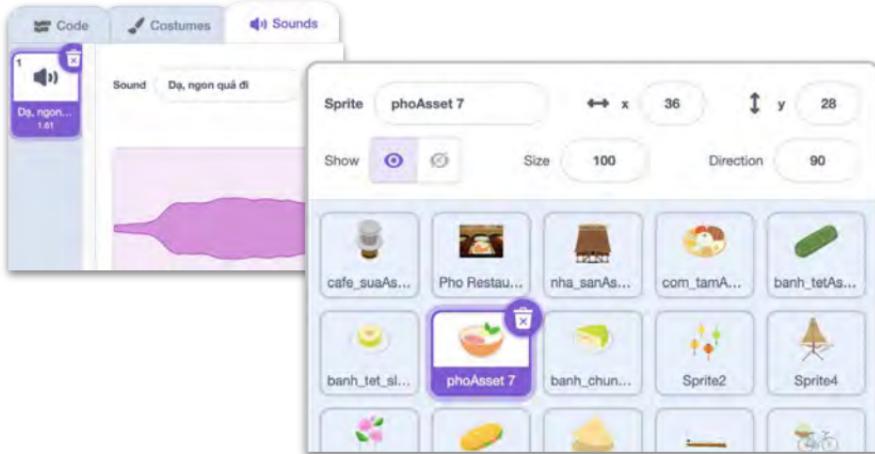
25

set size to

100

%

# Create an Asset Pack



Assets, in Scratch, can include:

- sprites
- costumes
- sounds
- backdrops
- code snippets

An **asset pack** is a collection of assets related to a specific theme, project type, cultural event, cultural symbols or customs, geographical region, or idea.

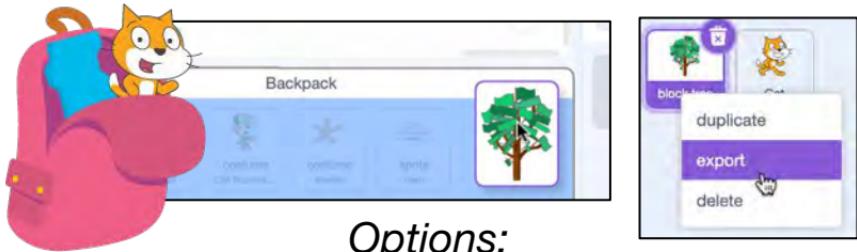
*Asset Pack example made by STEAM for Vietnam.*



# Create an Asset Pack

- **Name your sprite** and costumes with something descriptive.
- Consider creating **multiple costumes** for your sprite to show animation or variation.
- Consider adding at least one **related sound** for each sprite you create. Upload a sound or create an original sound by recording yourself, or noises in your environment.
- When creating an asset pack to share, we recommend creating your **backdrop as a sprite** instead, for easy backpacking or exporting.
- If you did not make a sound or an image yourself or you remixed someone else's creation, it is important to **provide credit** in the Notes and Credits section.

# Collaborate: Export or Backpack



*Options:*

- **Export a sprite, costume, or sound:**  
Right-click the asset. Choose “export.” To add the asset to a project, choose the **upload** option in the sprite, costume, or sound menu to upload from your files.
- **Backpack a sprite, costume, or sound:**  
You must be logged in to access the backpack at the bottom of the editor screen. Click it to open the backpack and drag-and-drop a sprite, costume, or sound inside. To add the asset to a different project, open the backpack and drag-and-drop the asset into the sprite, costume, or sound area.



# Collaborate: Remix

⟳ **Remix**

Scratch embraces remix culture. Remixing is when you build upon someone else's projects, code, ideas, images, or anything else shared on Scratch to make your own unique creation.

When remixing an asset, **make changes** like:

- adding code to animate the asset
- placing it in a new scene with other assets or add related sounds
- using the tools in the paint or sound editor to make adjustments to it
- adding additional elements you felt were missing

Just make sure that you **give credit** to whomever created the original asset in the Notes and Credits section.



# Sound and Music Cards



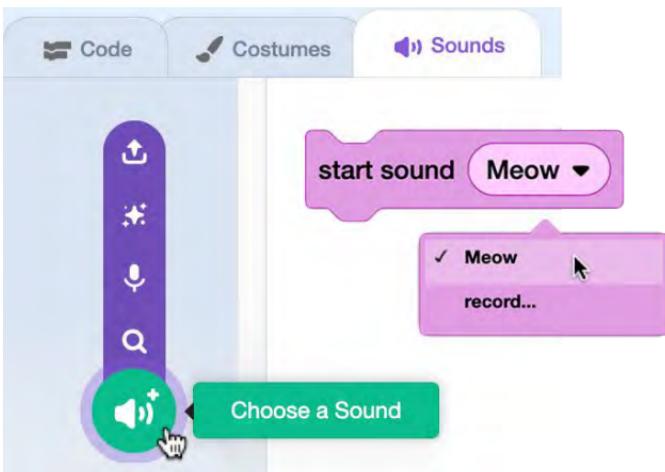
**Make some noise while exploring the sound and music extension blocks in Scratch.**



# Cards in This Pack

- Sound Blocks
- Pitch and Volume
- Text to Speech
- Loudness
- Create a Face Sensing Sound Board
- Music Extension Blocks
- Alternative Pianos
- Makey Makey Foil Piano
- Make or Re-Create a Song
- My Block: Music
- Musical List
- Generate a Melody: Repeat through a List
- From the original Scratch Coding Cards:
  - Animate a Drum
  - Surprise Song
  - Play the Drums (video sensing)
  - Squeak (using the micro:bit)

# Sound Blocks



- Click on a sprite or the backdrop and select the “Sounds” tab. Hover over the Sounds menu at the bottom of the tab, and select “Choose a Sound,” “Record,” or “Upload Sound.”
- Rules for uploaded sounds:
  - You can choose a MP3 or WAV file.
  - Please keep each of your files under 10MB.
  - Do not upload materials under copyright.
  - Uploads must follow the Community Guidelines.

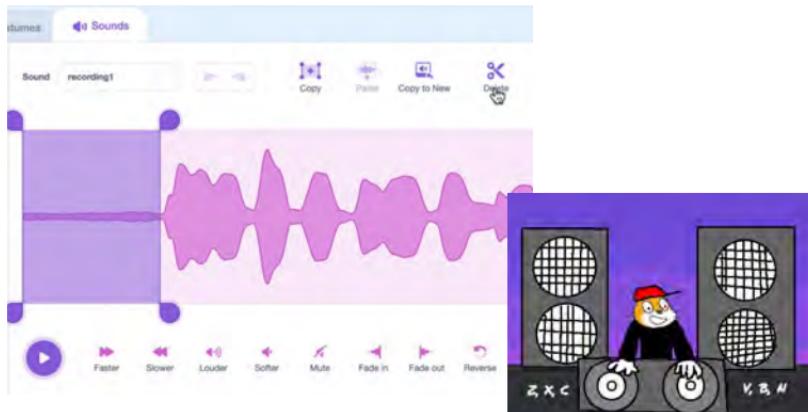
# Sound Blocks

scratch.mit.edu

1. Explore the difference between “start sound” and “play sound until done.” Try each in a “forever” loop or add another block, like a “say” block, after the sound block to see the difference. Note when the script moves to the next block after the sound block.

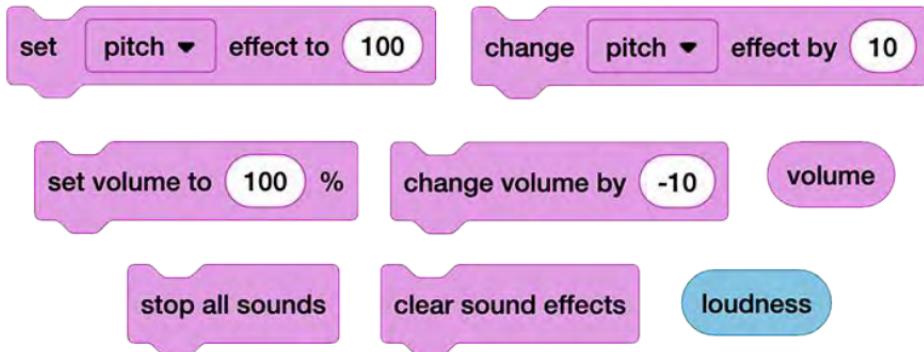


2. Try using the sound editor tools to make edits, like shortening the length or adjusting the volume or adding an effect like fade in and out or reversing it.



3. Check out our starter project “DJ Scratch Cat” ([scratch.mit.edu/projects/11640429](https://scratch.mit.edu/projects/11640429)). Explore and remix this project to think about how to pair sounds or layer sounds to create something new.

# Pitch and Volume

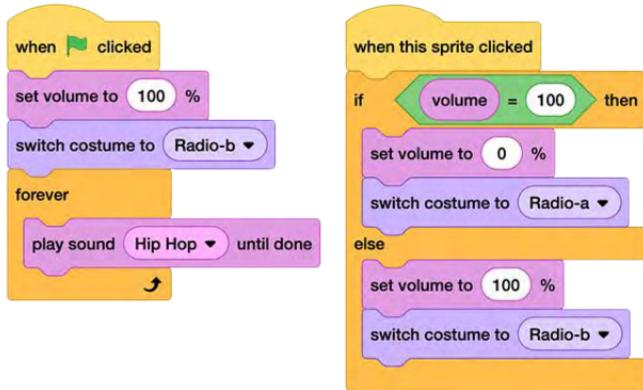


- One way is to customize sounds is to use the sound editor tools to make edits. Another way to customize and manipulate sound in Scratch is via code blocks.
- Explore blocks in the Sounds category that can set or change the pitch or the volume. There are also blocks to stop all sounds currently playing or clear sound effects, like pitch. How might you use such blocks in a project?

# Pitch and Volume

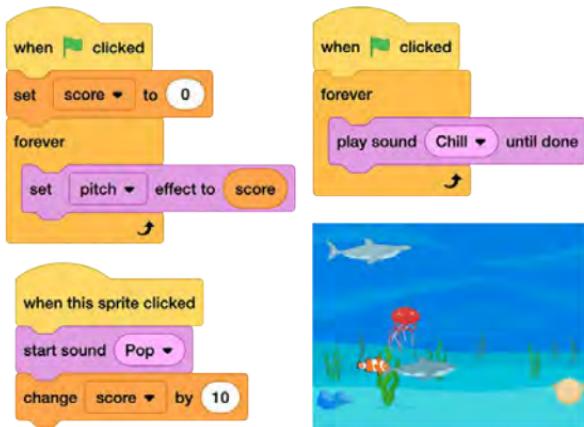
scratch.mit.edu

1. You could give users control over hearing or muting continuous background sound in a project. There are a number of ways to approach this. Here is one to try:



2. Or add emotion and excitement to a game by playing with the pitch of a sound. Check out our starter project “Catch the Fish, Increase the Pitch” ([scratch.mit.edu/projects/1106268602](https://scratch.mit.edu/projects/1106268602)). Explore and remix this project.

Notice as you click on all 30 fish that the music gets higher in pitch creating a feeling of urgency. How can sound add an emotional component to a project?



# Text to Speech

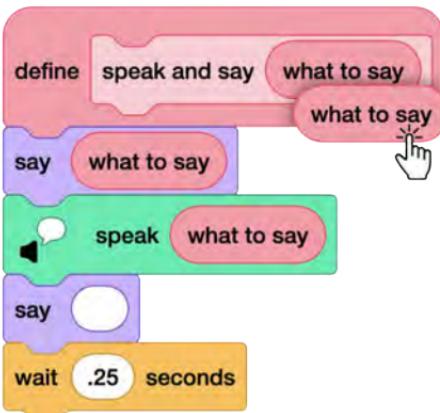
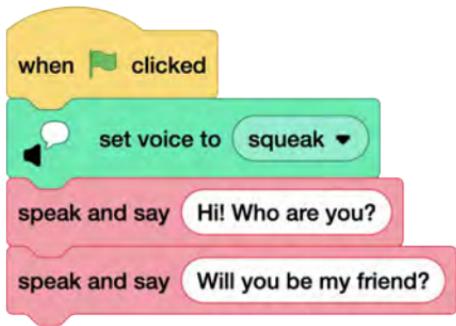


- Add the Text to Speech extension by clicking on the extension menu in the lower-left corner of the project editor. Note: You must be connected to the internet to use it.
- Set a voice. Type words into the block to hear them read aloud. Adjust the language to match the text for better pronunciation.
- Pair with blocks like “say” or the Translation extension blocks to make your projects more accessible.

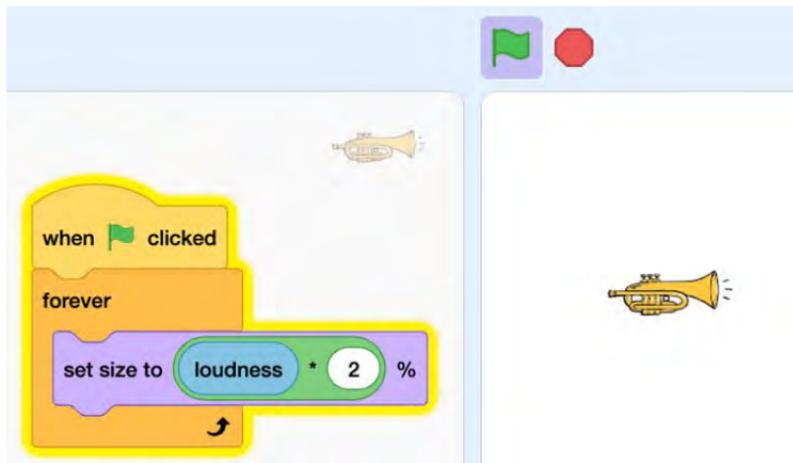
# Text to Speech

scratch.mit.edu

1. Check out our starter project “Text to Speech” ([scratch.mit.edu/projects/1106234816](https://scratch.mit.edu/projects/1106234816)). Explore and remix this project. Think about how to combine text to speech with “say” blocks to make text heard and seen.
2. See what happens when you adjust the language, or use with translate blocks.
3. Optional: Create a custom My Block to speed up the process of creating dialogue that can be heard and seen. (See our My Blocks resources for more.)



# Loudness



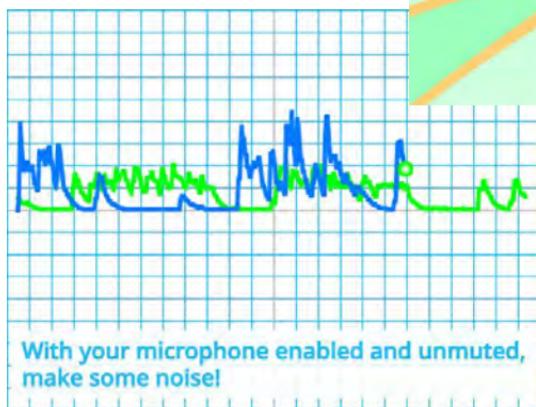
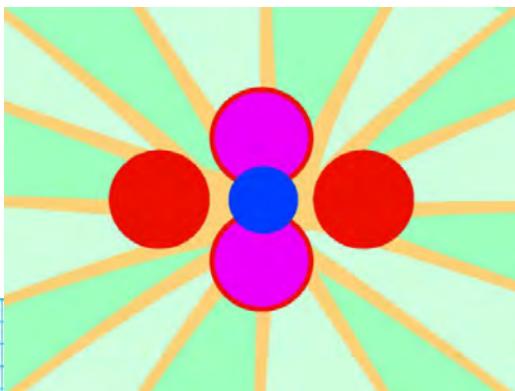
- Did you know there is also a “loudness” reporter block under the Sensing category that records the “loudness” of the noise that a microphone receives, on a scale of 0 to 100, to control things in Scratch?
- You must enable your microphone in the browser (nothing will be recorded or stored).
- How could you use this in a project? Try using it in the “set size” block inside a “forever” loop. Then, make some noise!

# Loudness

scratch.mit.edu

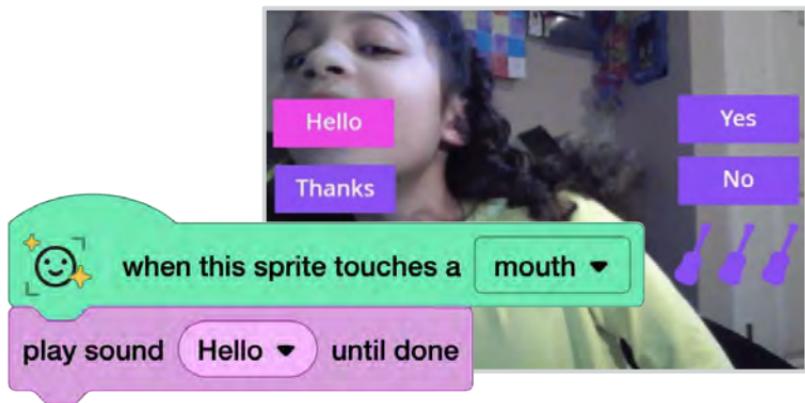
1. Check out our starter projects “Sound Graph” ([scratch.mit.edu/projects/1105532968](https://scratch.mit.edu/projects/1105532968)) and “SoundFlower” ([scratch.mit.edu/projects/1111537402](https://scratch.mit.edu/projects/1111537402)).
2. Explore! See how singing, playing music, or just making fun noises at different volumes creates an effect.
3. Remix and change the sprites, or adjust the numbers to see the effects.

How might you create an interactive art piece to accompany a musical performance?



With your microphone enabled and unmuted, make some noise!

# Create a Face Sensing Sound Board



- Choose a variety of fun sounds or record your own.
- Code a sound board that uses parts of your face to play sounds.
- Code effects that add visual confirmation, like color changes.
- Create a mystery button that picks a sound to play at random.

# Create a Sound Board

scratch.mit.edu

## GET READY

 Add the Face Sensing Extension.

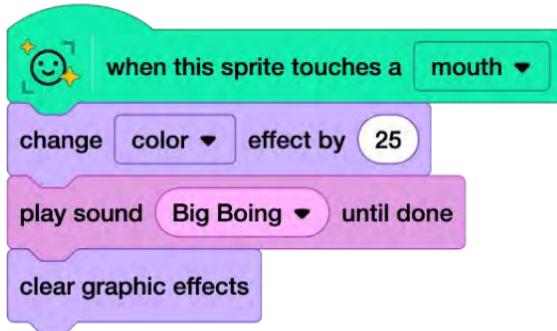


Choose a sound from the sound library for each sprite, or record your own.

Choose a Sound

## ADD CODE

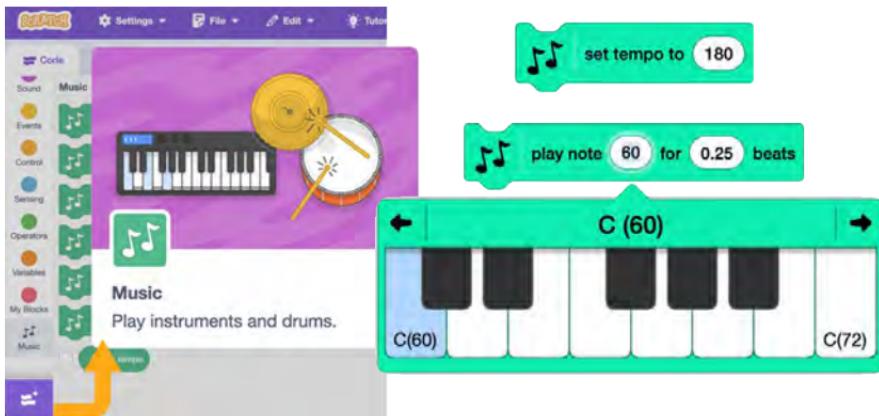
1. Add code to each sprite to play a sound, change an effect, or perform another animation when parts of your face touch them.



2. Try adding multiple sounds to a sprite. Use the “pick random” operator so each time is a surprise.



# Music Extension Blocks



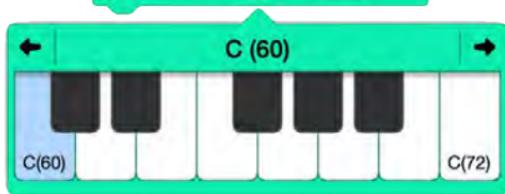
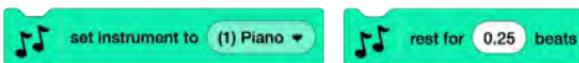
- Add the Music extension by clicking on the extension menu in the lower-left corner of the project editor and choosing “Music.”
- The beat or BPM (beats per minute) is a basic rhythmic unit of a measure. You can make the beat faster or slower by changing the number in that input bubble.
- A standard tempo is 60 BPM, which means one beat will be played each second. What happens if you create a sequence of “play note” blocks and use the same beat but change the tempo?

# Music Extension Blocks

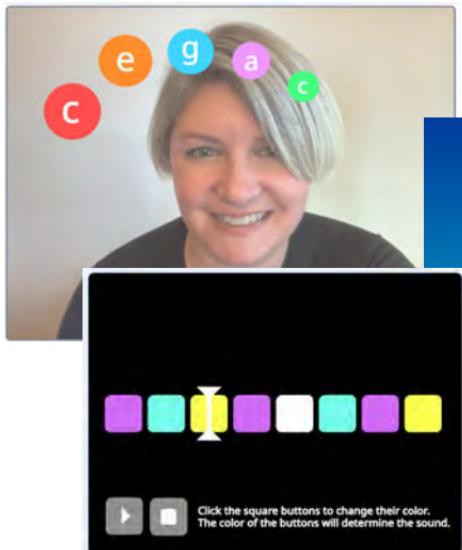
scratch.mit.edu

1. Check out our starter project “Piano” ([scratch.mit/projects/1106245381](https://scratch.mit/projects/1106245381)). We’ve set up a basic piano using the “play note \_ for \_ beats” block.
2. Click in the note input bubble to see the piano keys that appear so that you can choose a note attached to a number.
3. Adjust the beat using different numbers and test the difference.
4. Try changing the instrument, either via the slider we have provided or by changing the script. Note that changing the instrument on one sprite does not change it for all sprites in a project, so you could have a separate instrument for each key. How does using a variable make it easier to adjust the instrument for all keys at once?

5. In your remix, you could change what the piano looks like, add computer keyboard shortcuts, change the tempo, or add higher and lower notes.



# Alternative Pianos

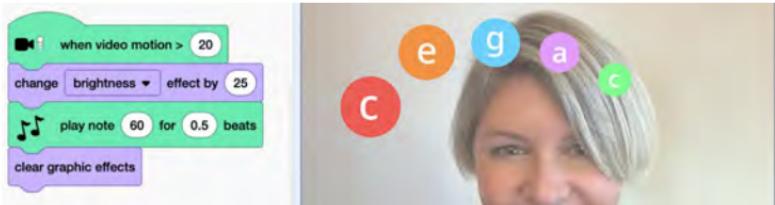


- What about creating an alternate piano keyboard? For instance, you could use video motion or the position of the mouse to play notes.
- You could also make your piano keyboard more accessible by adding visual effects when notes play for those hard of hearing or deaf. By making your musical projects visual as well as audio, more people can experience them, or experience them in different ways.

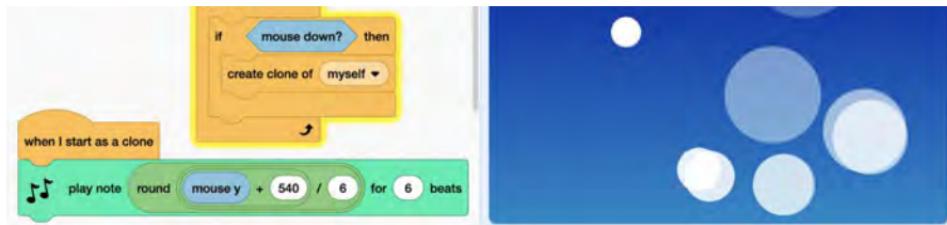
# Alternative Pianos

scratch.mit.edu

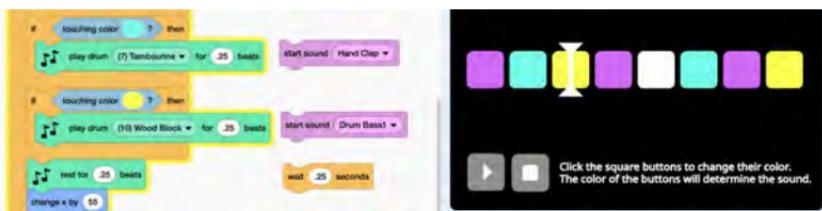
1. Check out our starter project “Musical Buttons using Video” ([scratch.mit.edu/projects/1105110383](https://scratch.mit.edu/projects/1105110383)).



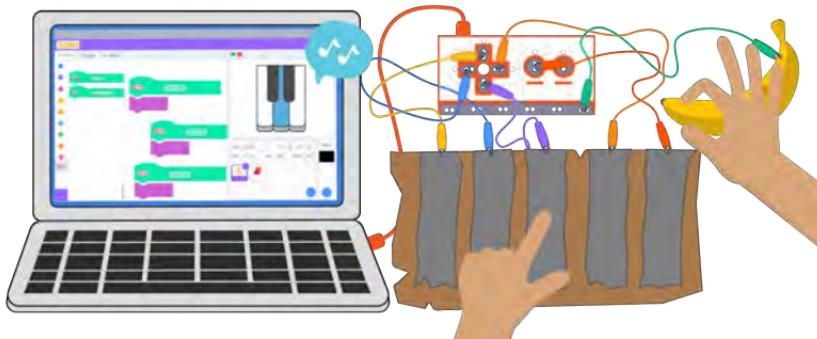
2. Check out our starter project “Musical Droplets” ([scratch.mit.edu/projects/1111576868](https://scratch.mit.edu/projects/1111576868)) that uses mouse y-position.



3. Check out our starter project “Drum Sequencer” ([scratch.mit.edu/projects/1111562971](https://scratch.mit.edu/projects/1111562971)) where the user creates the beat. The starter project uses sounds from the library, but you could experiment with additional drum sounds by using the “play drum \_ for \_ beats” block from the music extension.



# Makey Makey Foil Piano



You can make a physical piano with some foil and a Makey Makey (see our Makey Makey Coding Cards for more information).

## *Instructions:*

1. Connect one alligator clip to EARTH and various alligator clips to multiple keyboard keys, which will represent various musical notes.
2. Code a project so key presses play different notes.
3. Use foil, bananas, Play-doh, or other conductive materials as external keys.

# Makey Makey Foil Piano

scratch.mit.edu

## GET READY



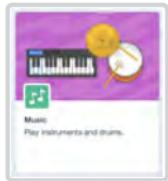
Choose any sprite or draw your own.



Keyboard



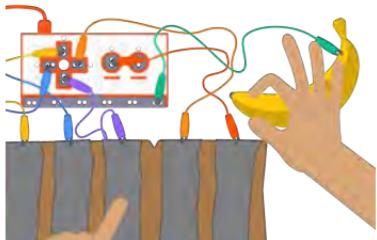
Optional: add the Music Extension.



## ADD CODE AND TEST



Select note sounds in the Sound library to play when different keyboard keys are pressed. (You can use the Makey Makey extension hat block or the Event hat block.)



Or add the Music Extension and select notes to play when different keyboard keys are pressed. Notes can be customized for beat count and instrument.

Close the circuit to register each keyboard press by touching EARTH and a keyboard input.

# Make or Re-Create a Song

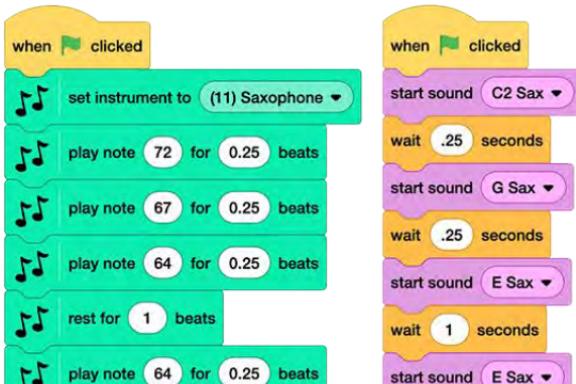


- Think about the structure of a piece of music. Typically there are verses and choruses that are repeated throughout the song.
- You can use the music or sound blocks to compose your own original composition or recreate a song!
- Are you creating a song with a simple melody, or are you creating chords and layering sounds? There are different approaches you can try using sound blocks and instrument sounds from the library or music blocks. Experiment!

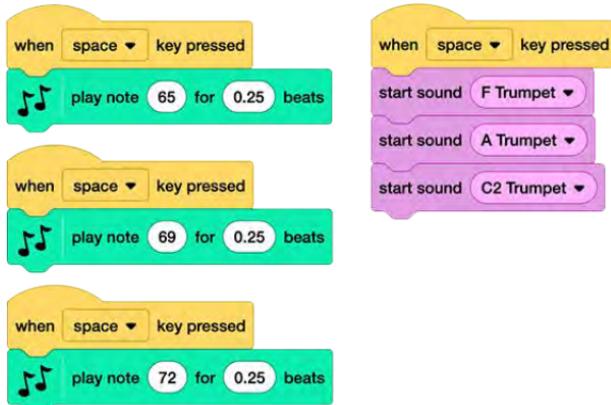
# Make or Re-Create a Song

scratch.mit.edu

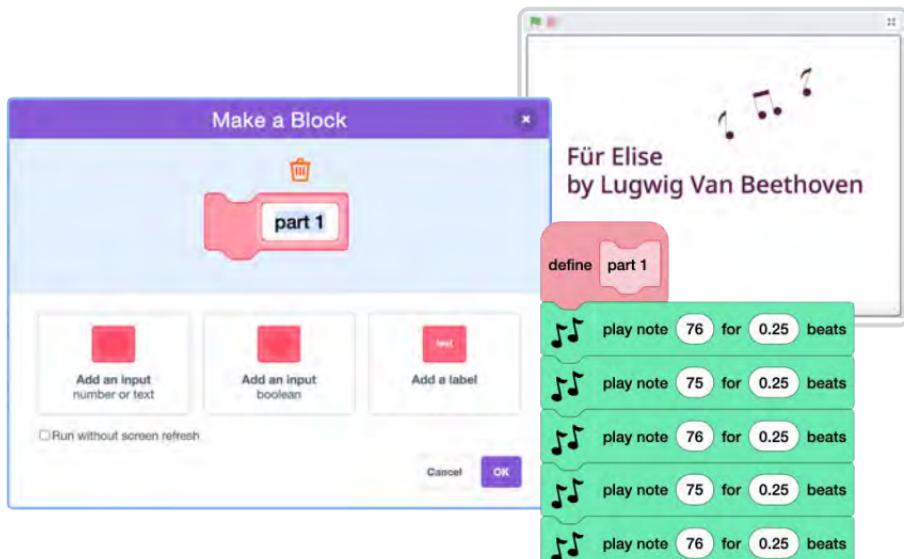
Here is an example of two different ways to create a melody. How to choose? Explore which instruments are available in the sound library vs the music blocks. Think about how much control you need over the beat.



Are you creating chords/layering sounds or notes to play at the same time? Here is an example of two different ways.



# My Block: Music



- You can use music blocks from the Music extension to create a song in Scratch.
- Rather than write the same sequence of notes over and over when they repeat in your song, you can place those notes in a My Block and simply call that block each time you need it, for instance each time a chorus is called.

# My Block: Music

scratch.mit.edu

## GET READY



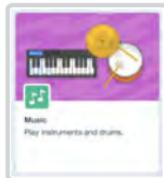
Choose any sprite.



Keyboard



Add Music extension.

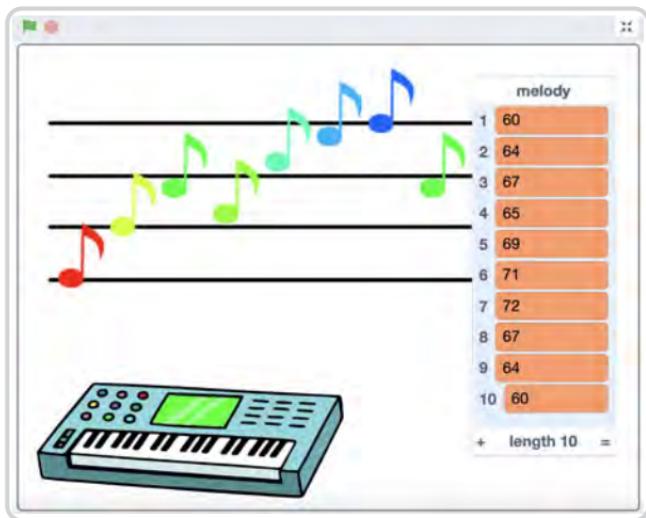


## ADD CODE

The Scratch script starts with a green `when green flag clicked` hat block. Inside, there's a yellow `repeat (2)` control block. The first iteration of the loop contains a pink `define part 1` control block, which contains a green `set instrument to [Piano v]` sound block and a series of green `play note [76 v] for [0.25 v] beats` sound blocks. The second iteration of the loop contains a pink `define part 2` control block, which contains a green `rest for [0.25 v] beats` sound block followed by a series of green `play note [60 v] for [0.25 v] beats` sound blocks. The third iteration of the loop contains a pink `define first section` control block, which contains a pink `part 1` control block and a pink `part 2` control block, each containing a series of green `play note [64 v] for [0.25 v] beats` sound blocks. The fourth iteration of the loop contains a pink `define` control block, which contains a pink `part 1` control block and a pink `part 2` control block, each containing a series of green `play note [64 v] for [0.25 v] beats` sound blocks.

1. Compose the sections of your song. Create multiple My Blocks for different parts (such as verse and chorus).
2. My Blocks can also be placed within other My Blocks to further simplify the code.
3. Use My Blocks in the main program, along with repeat blocks (if applicable) to compose a whole song. Set the instrument and the tempo.

# Musical List



- You can use a predefined list to determine animation.
- Try creating a melody project, storing song notes in a list that creates a musical score that can be played.
- As a bonus, you can use the Pen extension to stamp notes on a scale and produce a visual representation of your musical score.

# Musical List

scratch.mit.edu

## GET READY



Add the Music extension.



Choose any sprite.



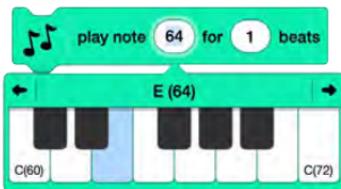
Keyboard

## ADD CODE

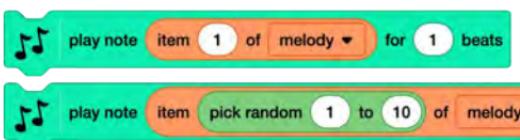
1. Create a list. Add song notes to the list via the stage monitor (add rows manually and type note numbers in) or by using the “add to [list]” block.



You can find note numbers by clicking on the input of the “play note” music block.

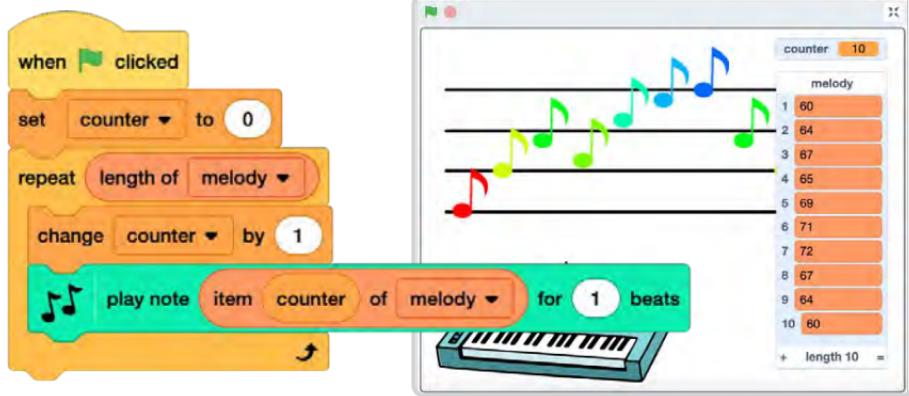


2. Write a script to play each note on the list by item number, or let the program pick the note to play randomly.



See the next card to learn how to create a “counter” variable to automate moving/repeating through the list in order.

# Generate a Melody: Repeat through a List



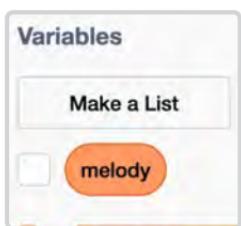
- While there is no “next item of list” block, you can create a script that loops through the items of a list in order.  
The ability to automate moving or repeating through a list can speed up your coding process and make editing scripts quicker.
- This can be useful if you want to add items of a list together, speak or say items in a list, etc.

# Generate a Melody

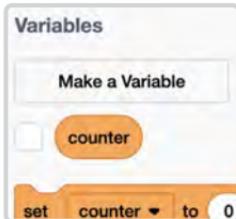
scratch.mit.edu

## GET READY

Create a list.



Create a variable.



## ADD CODE

Step through the code on the card front to see what it does:

1. Changes the “counter” variable (that stores a number to represent an item number on the list) by one.



2. Plays the note number associated with that item number (the number entered on that line of the list). Note: This is why it is important to first set “counter” to zero first each time the program runs.



3. Repeat as many times as there are rows in the list/for the length of the list.



# Animate a Drum

Switch between costumes to animate.



# Animate a Drum

scratch.mit.edu

## GET READY



Choose  
a drum.



### Costumes

Click the **Costumes** tab to see the costumes.  
You can use the paint tools to change colors.



## ADD THIS CODE

### Code

Click the **Code** tab.



Choose a sound from the menu.

## TRY IT



Press the **left arrow** key on your keyboard.

# Surprise Song

Play a random sound from a list of sounds.



# Surprise Song

scratch.mit.edu

## GET READY



Choose an instrument,  
like Guitar.



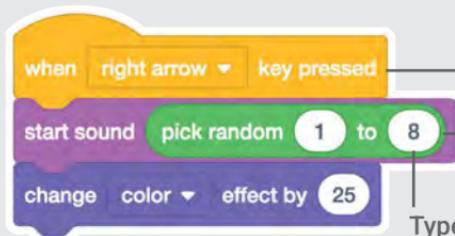
Click the **sounds** tab to see how many sounds are in your instrument.



## ADD THIS CODE



Click the **Code** tab.



when right arrow key pressed

start sound pick random 1 to 8

change color by 25

Choose **right arrow**.

Insert a **pick random** block.

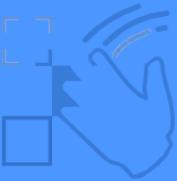
Type the number of sounds in your instrument.

## TRY IT



Press the **right arrow** key.

# Play the Drums



Interact with sprites that play sounds.



# Play the Drums

scratch.mit.edu



## GET READY



Click the Extensions button, then choose Video Sensing.

Choose two sprites, like Drum and Drum-cymbal.

## ADD THIS CODE

Click on a drum to select it, then add its code.



```
when video motion > 10
set size to [100 %]
change size by [20]
start sound [High Tom v]
wait [0.1 seconds]
change size by [-20]
```

Type a minus sign to get smaller.

```
when video motion > 10
switch costume to [drum-cymbal-a v]
start sound [Crash Cymbal v]
wait [0.1 seconds]
switch costume to [drum-cymbal-b v]
```

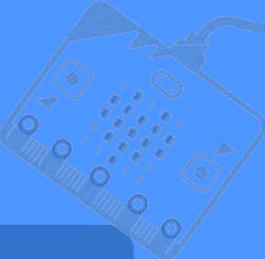
Choose a different costume.

## TRY IT

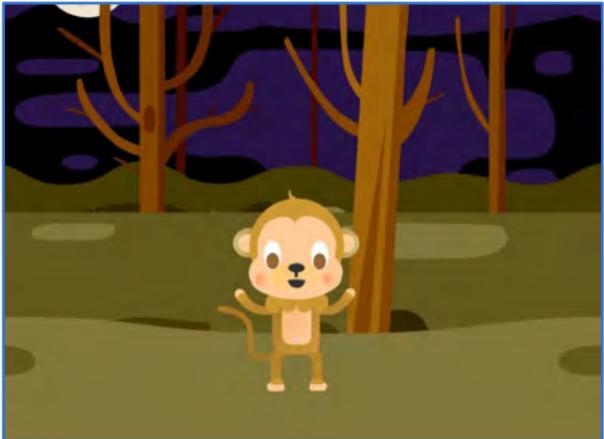
Use your hands to play the drums!



# Squeak



Make a sound when you  
shake the micro:bit.



# Squeak

scratch.mit.edu/microbit

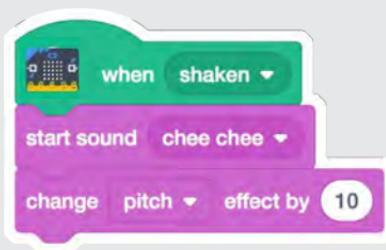


## GET READY



Choose a sprite, like  
Monkey.

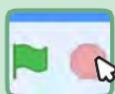
## ADD THIS CODE



## TRY IT



Shake the  
micro:bit to start.



Click the stop sign  
to reset the pitch.



You can click the Sounds tab to  
view your character's sounds.

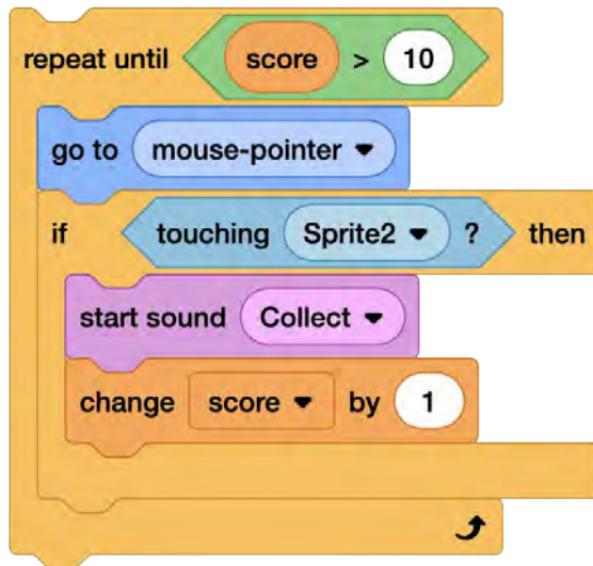
## TIP



Click this button to add a sound  
from the Sound library.



# Conditional Statements



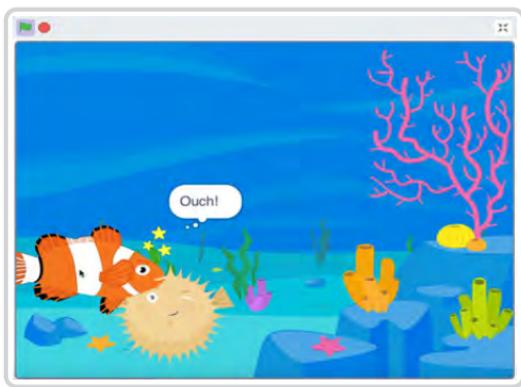
Create dynamic programs that are interactive or offer multiple outcomes



# Cards in This Pack

- Fish Game: Code the Fish
- Different Code, Similar Results
- Conditional Statements: “If Then”
- Conditional Statements: “Until”
- Operators in Conditional Statements
- Fish Game: Nested Conditional Statements
- Nested Conditional Statements
- Create a Maze Game
- Create a Math Game

# Fish Game: Code the Fish



- Have you ever wanted to create a Scratch program that is interactive or offers multiple outcomes? Let's create a fish game that the user can interact with, and code different animations triggered by conditional statements.
- First, let's code a sprite to be controlled by the user's mouse. Then, use a conditional statement to trigger costume changes when sprites touch.
- See additional cards to code other sprites and trigger additional animations.

# Fish Game: Code the Fish

scratch.mit.edu

## GET READY



Choose two  
sprites.



Fish and  
Pufferfish



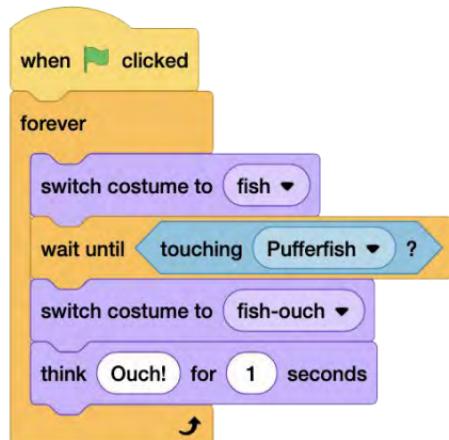
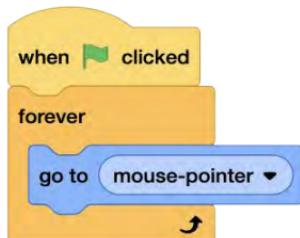
Choose any  
backdrop.



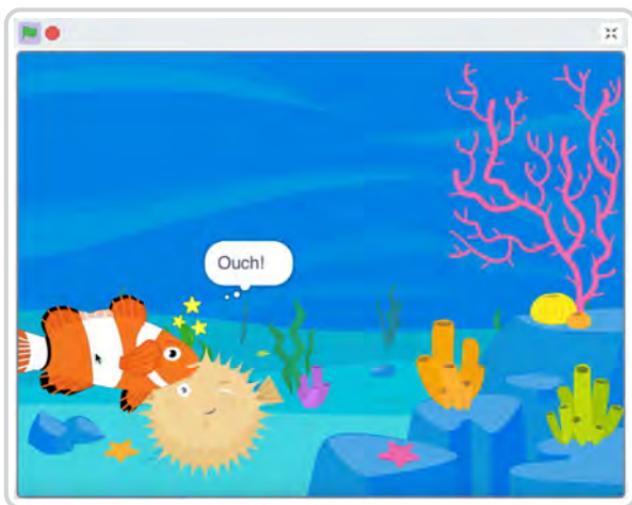
Underwater

## ADD CODE

1. Add code to make the fish continuously follow your mouse.
2. Create an additional costume for the fish that will show when it touches the pufferfish. You can duplicate and edit the fish costume to show stars or a funny face, etc.
3. Create a second script that uses a conditional statement to make the fish change costumes when it touches the pufferfish.  
Test and debug!



# Different Code, Similar Results



- There is often more than one solution/more than one way to code a program to get a similar result.
- Experiment with using different types of conditional statements (like “wait until” versus “if then else”). What differences, if any, do you notice?

# Different Code, Similar Results

scratch.mit.edu

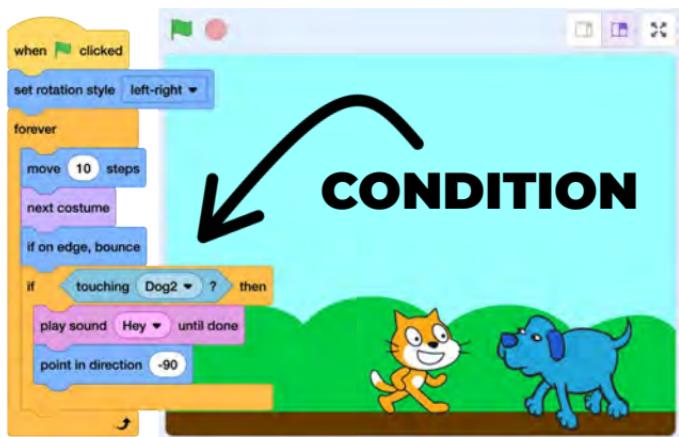
## COMPARE CODE

1. The first code stack on the right is the code tried on the “Fish Game: Code the Fish” card.
2. Compare it to the second code stack below that uses an “if than else” statement instead of a “wait until” conditional statement. What is the same and what is different?
3. Test each code stack. Make the fish touch the pufferfish and then move it away quickly, what do you observe happening?
4. Experiment and customize! What solution works best for your game?

```
when green flag clicked
forever
  switch costume to [fish v]
  wait until [touching [Pufferfish v] ?]
  switch costume to [fish-ouch v]
  think [Ouch!] for (1) seconds
```

```
when green flag clicked
forever
  if [touching [Pufferfish v] ?] then
    switch costume to [fish-ouch v]
    think [Ouch!]
  else
    switch costume to [fish v]
    think [ ]
```

# Conditional Statements: “If Then”



Boolean blocks that report "true" or "false" are used in conditional statement blocks. Try using:

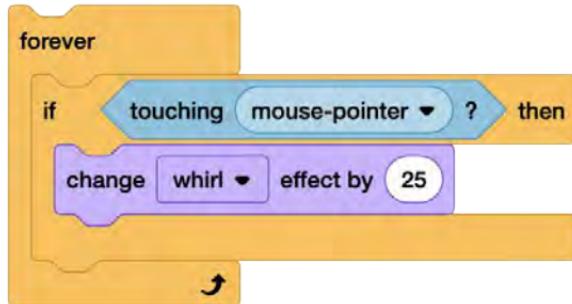
- user actions, such as pressing keyboard keys or mouse positioning or clicking
- sprite interactions (touching another sprite), comparisons (distance between sprites), and touching colors of sprites or backdrops
- data input by users, data stored in variables and lists, or data stored in reporter blocks

# Conditional Statements: “If Then”

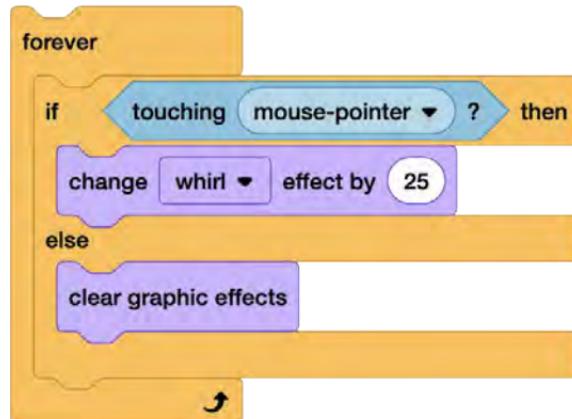
scratch.mit.edu

Experiment with “if (something) is true or false then” conditional statement blocks. What is the difference between these scripts below?

## IF THEN

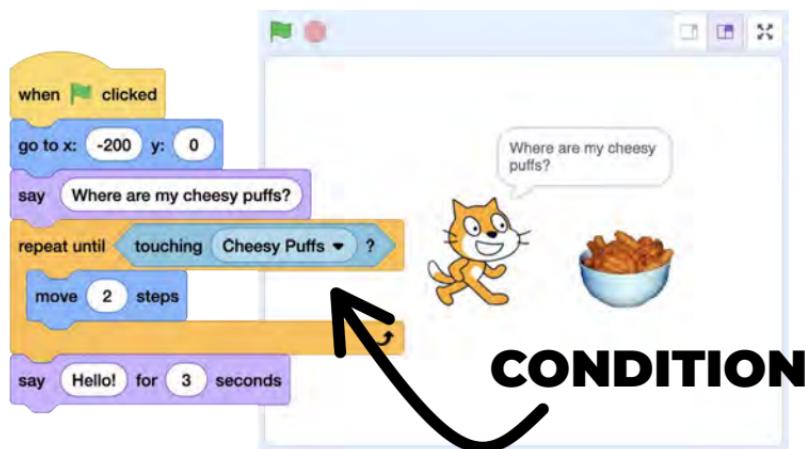


## IF THEN ELSE



Try different conditions and customize the results. What is the difference if the condition block is not in a forever loop?

# Conditional Statements: “Until”



Boolean blocks that report "true" or "false" are used in conditional statement blocks. Try using:

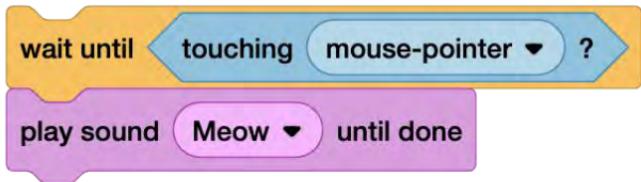
- user actions, such as pressing keyboard keys or mouse positioning or clicking
- sprite interactions (touching another sprite), comparisons (distance between sprites), and touching colors of sprites or backdrops
- data input by users, data stored in variables and lists, or data stored in reporter blocks

# Conditional Statements: “Until”

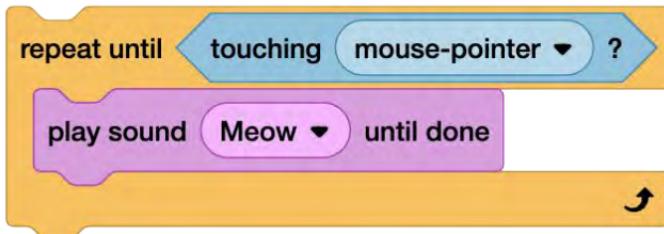
scratch.mit.edu

Experiment with “until (something) is true or false” conditional statement blocks. What is the difference between these scripts below?

## WAIT UNTIL

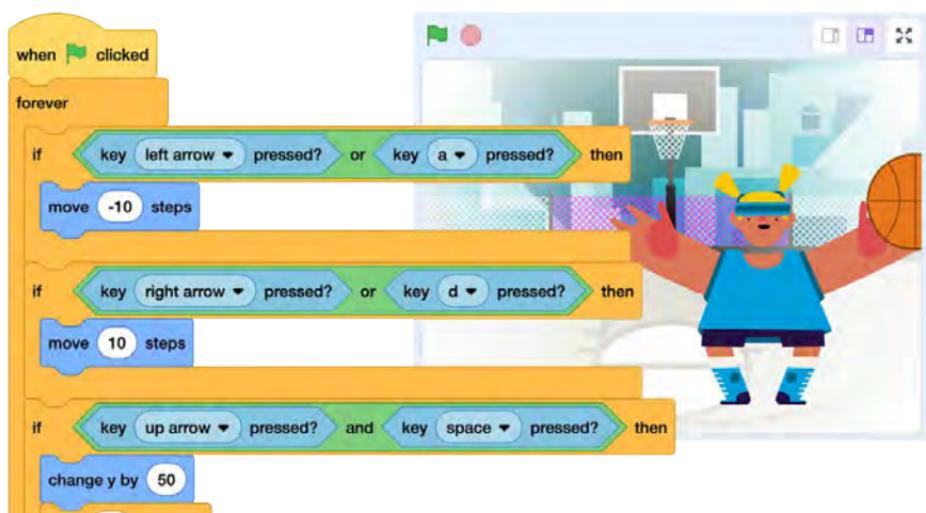


## REPEAT UNTIL



Try different conditions and customize the results. What is the difference if the condition block is inside a forever loop?

# Operators in Conditional Statements



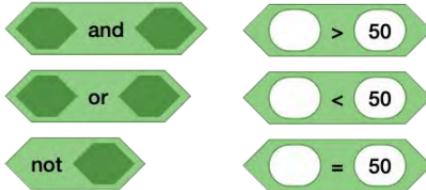
Using operators in conditional statements with sensing blocks or reporter blocks can give you a wider range of conditions to choose from.

Scratch comes with some built-in reporter blocks that store information, like the x or y position of a sprite, the volume in the project, the sprite's size, etc. Looks for these oval blocks under a number of the block categories.

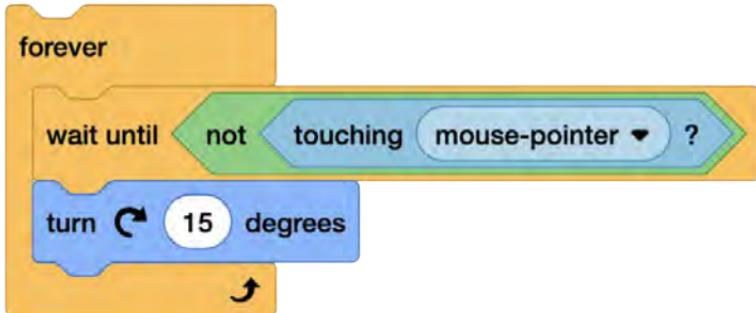
# Operators in Conditions

scratch.mit.edu

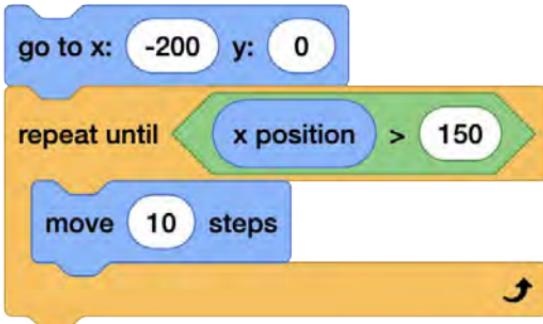
Experiment with different operators in your condition, such as:



## “NOT” OPERATOR



## COMPARISON OPERATORS



Try using “and” or “or.” What is the effect of two conditions needing to be true, or multiple options that make it true?

# Fish Game: Nested Conditional Statements



- Imagine a project where multiple effects can be triggered based on the distance between sprites, colors sprites are touching, or other conditions.
- Nested conditional statements are where one conditional statement is placed inside another. Use them to add additional complexity to your program.
- You could use nested statements to add scoring, or use color as another condition.

# Fish Game: Nested Condition

scratch.mit.edu

## GET READY



Choose two  
sprites.



Fish and  
Pufferfish



Choose any  
backdrop.

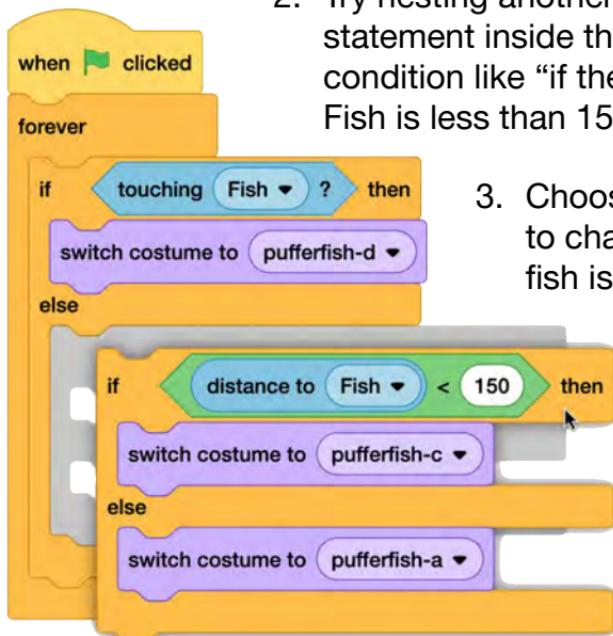


Underwater

## ADD CODE

1. Add an “if then else” block to the pufferfish so it changes costumes when it touches the fish.

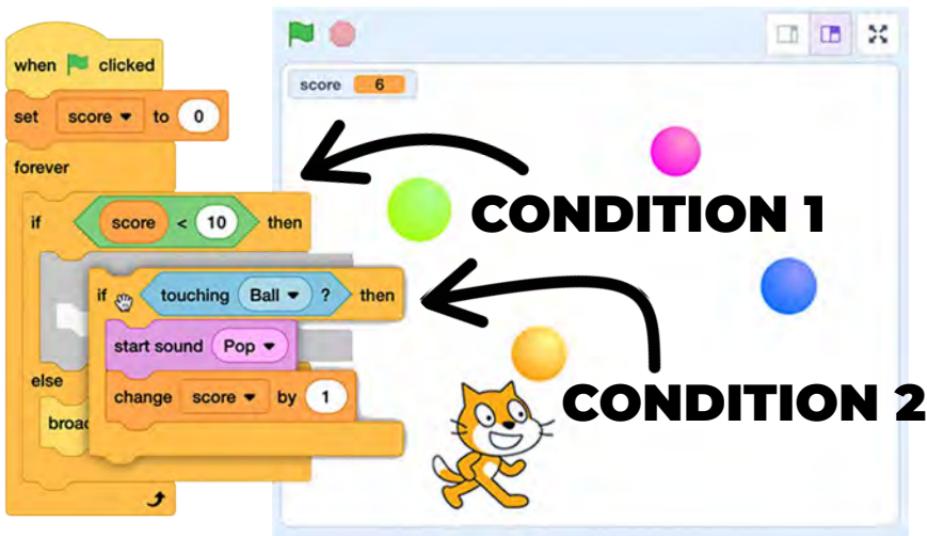
2. Try nesting another “if then else” statement inside the first. Add a condition like “if the distance to the Fish is less than 150 pixels then...”



3. Choose a third costume to change to when the fish is close.

Test the code.  
What is the  
difference if the  
sequence is  
reversed and it  
checks the  
distance first?

# Nested Conditional Statements



Nested conditional statements are where one conditional statement is placed inside another. The first condition is checked and, depending on if it is true or false, the program may then move forward to check the nested condition inside.

This means that the sequence of the nested statements is important.

# Nested Conditional Statements

scratch.mit.edu

Experiment with nested conditional statements using two or more “until (something) is true or false” or “if (something) is true or false then” conditional statement blocks.

## GET READY



Choose two  
sprites.



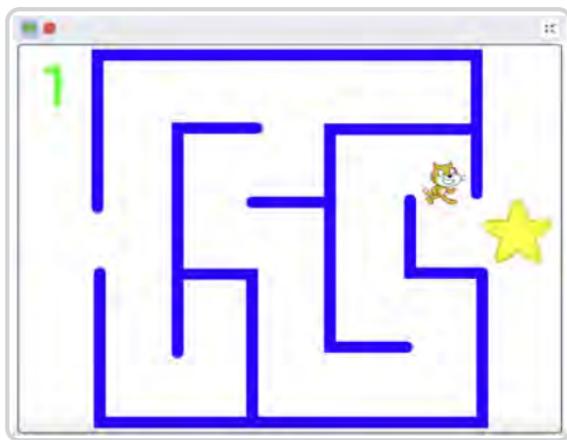
## ADD CODE

```
forever
  if [touching mouse-pointer?]
    then
      turn (2 degrees)
      if [mouse down?]
        then
          change color by (25)
        else
          turn (-2 degrees)
    end
  end
```

```
forever
  if [mouse down?]
    then
      change color by (25)
      if [touching mouse-pointer?]
        then
          turn (2 degrees)
        else
          turn (-2 degrees)
    end
  end
```

These code stacks use the same blocks. The difference is the sequence (which conditional statement is nested). Code each sprite with a different stack. Click the mouse when touching each sprite and when not touching each sprite. What differences do you notice? Customize and experiment!

# Create a Maze Game



- Use the line tool to create maze walls with color. Use the “touching color” condition to stop sprite movement through the walls. To select the color, use the eyedropper tool. 
- Make sure your sprite can fit through all passages and around all corners.
- Bonus: Try using another conditional statement to code a winning animation when the end is reached!

# Create a Maze Game

scratch.mit.edu

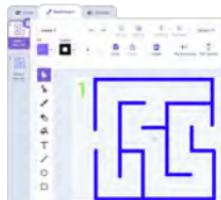
## GET READY



Choose any sprite.

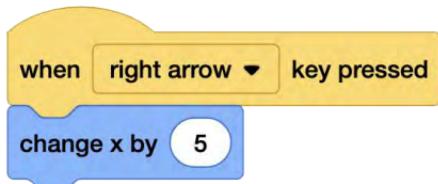


Create maze backdrops using the paint editor line tool.

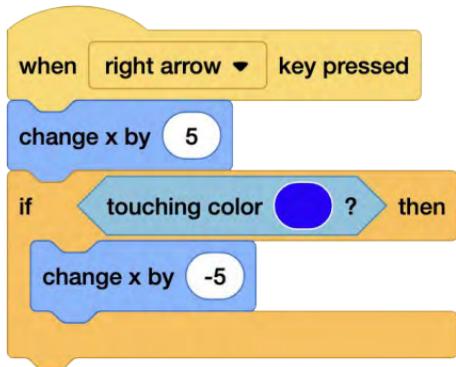


## ADD CODE

1. Add code to move the sprite up and down when arrow keys are pressed by changing y, and move left and right by changing x. Use positive and negative numbers.

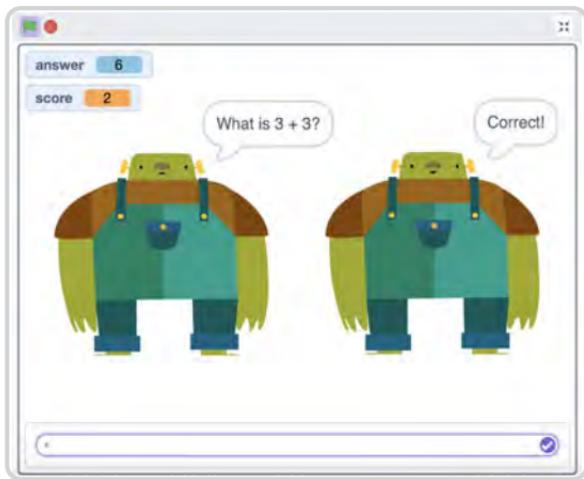


2. Add an “if then” conditional statement to reverse the move if touching the maze wall color.
3. If necessary, add code to change maze backdrops.



You could add a sprite at maze end and use a conditional statement to change the backdrop when touched.

# Create a Math Game



- Choose addition, subtraction, division, or multiplication questions to ask a user.
- Use a conditional statement to check the answer given, and have the program respond differently if the answer is correct or incorrect.
- Use a score variable to record points.
- Bonus: Try using My Blocks to make your program more efficient to write and edit.

# Create a Math Game

scratch.mit.edu

## GET READY



Choose any sprite.



Create a “score” variable

Variables

Make a Variable



## ADD CODE

1. Use the ask block to ask a math question.
2. Add an “if then else” conditional statement block. Then, use an equals operator block to set the condition to “answer” equals the correct answer.
3. Add blocks to say “Correct” if the answer is correct, else “Wrong.” Customize with sounds or other effects.
4. Use variable blocks to set and change the score.

```
when green flag clicked
set [score v] to [0]
ask [What is 2 + 2? and wait]
if [answer = 4] then
  change [score v] by [1]
  start sound [Ya v]
  say [Correct!] for [2] seconds
else
  start sound [Big Boing v]
  say [Wrong!] for [2] seconds
```



# Variables and Lists



Store and recall information to create customized animations, stories, and games



# Cards in This Pack

- Reporter Blocks in Scratch
- Brightness Slider
- Interactive Storytelling
- Clone Piano: Using Local Variables
- Clone Piano: Using Global Variables
- Musical List
- Generate a Melody: Repeat through a List
- Generate a Sum: Repeat through a List

*Perhaps you have used a variable to store a game score, but did you know a variable can hold numbers or text (also known as a “string”)? See these cards for examples of non-score uses for variables and lists. See our in-editor tutorials or other coding cards for instructions on how to set up a basic score.*

# Reporter Blocks in Scratch



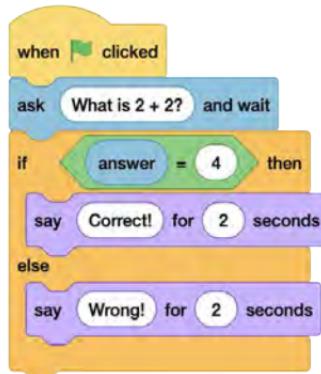
- Variables and lists hold information you can use in your program, but Scratch comes with some built-in reporter blocks that also store information.
- Unlike a stack block, reporter blocks go inside another block to serve as an input.
- You can click on a reporter block in the block palette or in the script area to see the piece of data it currently holds or the value it reports. Or check a box next to many of these reporter blocks to display them on the stage via a stage monitor.

# Reporter Blocks in Scratch

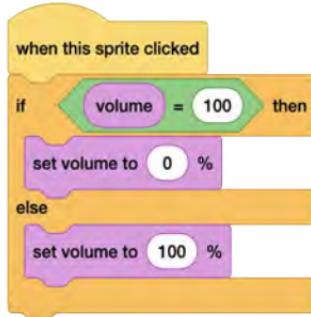
scratch.mit.edu

## EXPERIMENT WITH SOME REPORTERS

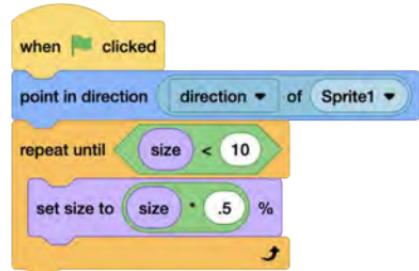
When you use the “ask” block to pose a question to a user, the answer they enter into the dialogue box is stored in a reporter block called “answer.”



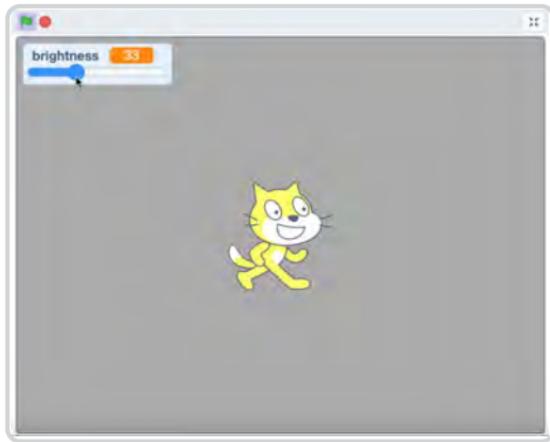
The “volume” reporter block stores the number representing the current volume of the sprite, clone, or stage.



You can use reporter blocks that store the position, direction, and size of sprites to perform calculations or mirror properties.



# Brightness Slider



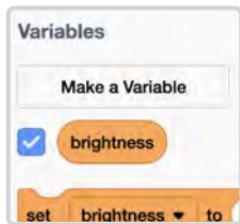
- Scratch comes with some built-in reporter blocks that store information, but what if you want to store and recall information for which there is no reporter block, like the sprite color or brightness?
- Let's create a project where a variable controls the brightness of the sprite.
- And let's put the power in the hands of users, by letting them control the value in the variable with a slider.

# Brightness Slider

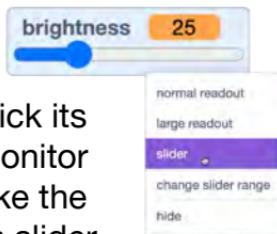
scratch.mit.edu

## GET READY

Create a variable called "brightness."

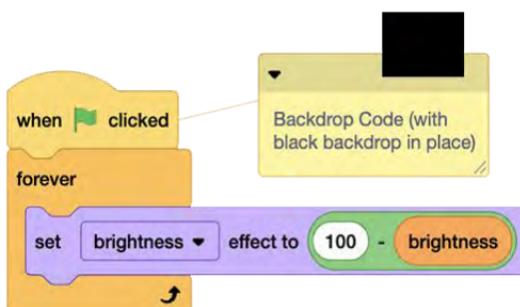
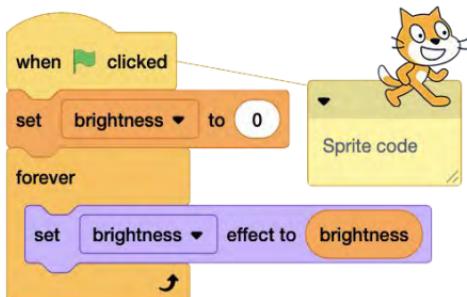


Right-click its stage monitor and make the readout a slider.

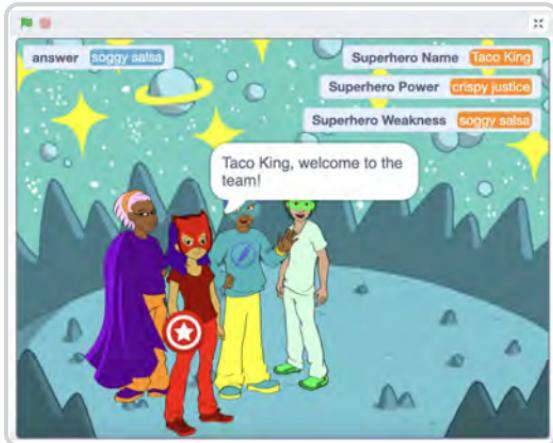


## ADD CODE

1. Create a short script that sets the "brightness" variable to 0.
2. Then, have the program forever set the brightness effect to the number in the "brightness" variable. Test it out!
3. Try adding code to make another sprite or the backdrop have the opposite effect.



# Interactive Storytelling



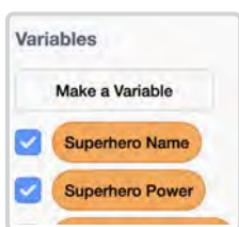
- You can pass information to a variable or list by clicking or moving a sprite, adjusting a slider, via code blocks, and more!
- You can also pass information from one reporter block, variable, or list to another. This could be helpful because variables and reporter blocks can only hold one piece of information at a time.
- Let's create a project that collects the user's answers to multiple questions, and repeats them back in the form of a story!

# Interactive Storytelling

scratch.mit.edu

## GET READY

Create multiple variables.

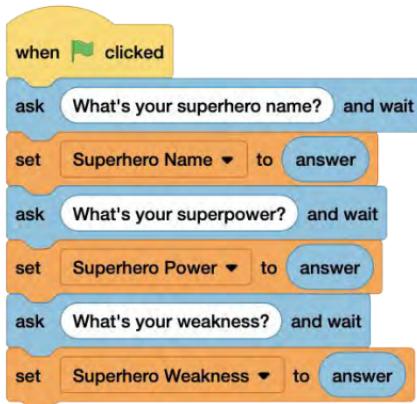


Choose any sprite.



## ADD CODE

1. Create individual variables for each answer you are collecting. The “answer” reporter block can only hold one piece of data at a time, so use the “set [variable] to” block to pass the “answer” into a variable for storage after each related question is asked.

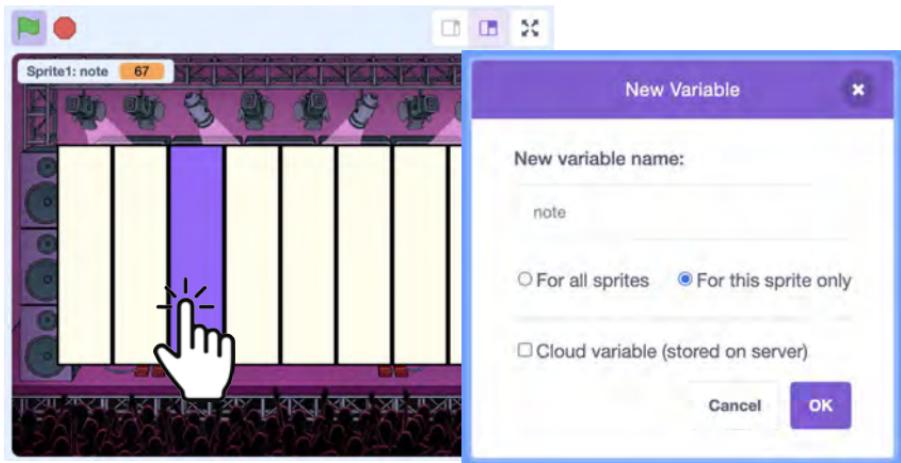


2. Use a “say” block and a “join” block to weave each answer stored in a variable into a story sentence.



*Alternative: Store and retrieve answers from a list!*

# Clone Piano: Using Local Variables



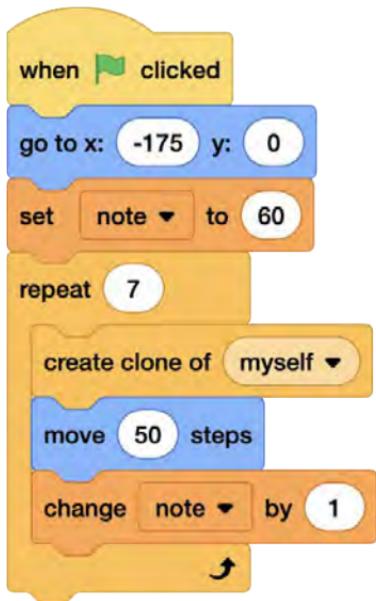
A local variable (“For this sprite only”) is individual to a single sprite or a single clone. (Versus a global variable that applies to all sprites in the project and all their clones.)

The value stored in each clone’s local variable is the value that was present at the moment the clone was created.

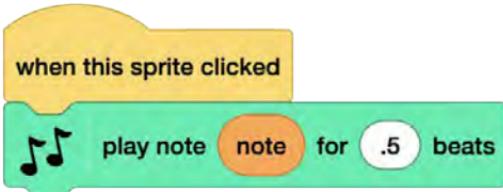
Local variables show the sprite name followed by the variable name in the stage monitor.

# Clone Piano: Using Local Variables

scratch.mit.edu

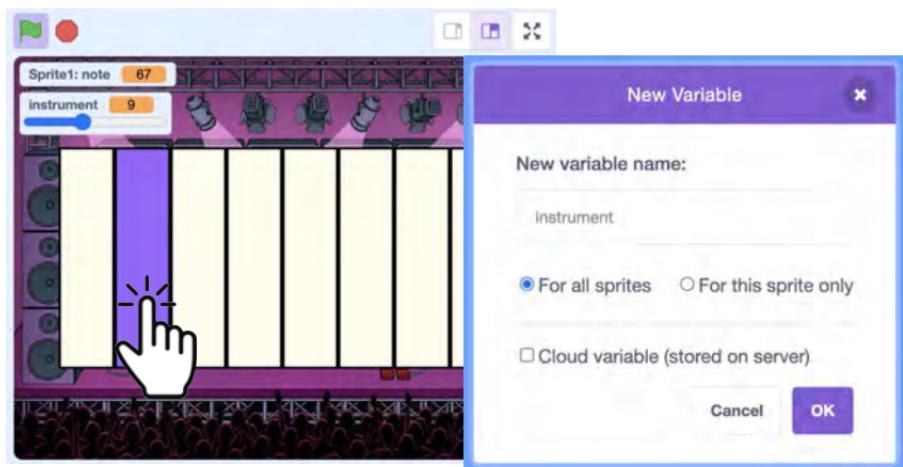


1. Create a piano key sprite by using the Rectangle drawing tool in the Paint Editor.
2. Assemble a script that creates clones of the piano key sprite in a row.
3. Create a local variable (“For this sprite only”) to store the note for each clone and the original sprite.
4. Set the initial note, and then change the note after each clone is created. Use the “note” variable in the “play note” block. Test and debug!



*Optional:* Adjust your program so it changes the color of the piano key or shows a different costume for the piano key as the note is played, so you can hear and see when a piano key is pressed.

# Clone Piano: Using Global Variables



A global variable (“For all sprites”) applies to all sprites in the project and all their clones.  
(Versus a local variable that is individual to a single sprite or a single clone.)

The value stored can be changed, and the variable is updated for all sprites in a project.

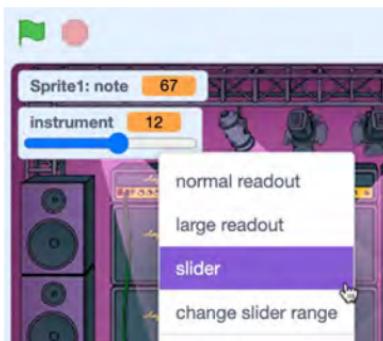
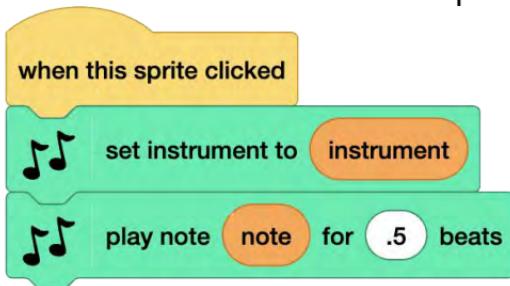
Global variables just show the variable name in the stage monitor.

# Clone Piano: Using Global Variables

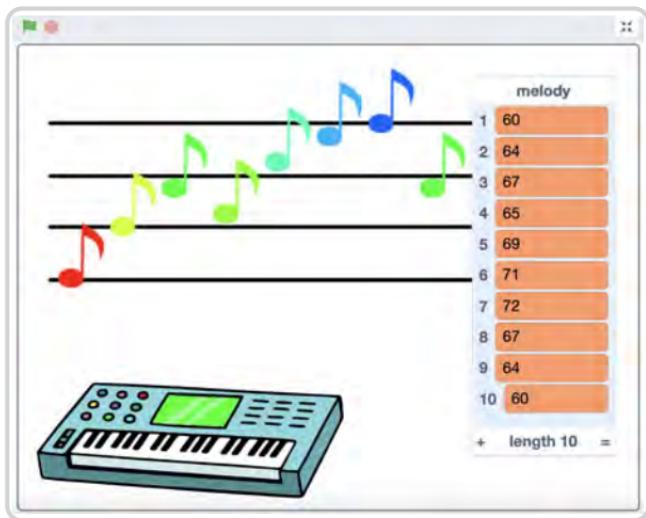
scratch.mit.edu



1. Create a global variable (“For all sprites”) to store the instrument for each clone and the original sprite.
2. Set the initial instrument, and then use the “instrument” variable in the “set instrument” block.
3. Right click on the “instrument” stage monitor to change it to a slider. Then, right click again to set the range to 1-21 (the number of instruments available). Now, use the slider to change the instrument globally, for all piano keys. Test and debug!



# Musical List



- You can use a predefined list to determine animation.
- Try creating a melody project, storing song notes in a list that create a musical score that can be played.
- As a bonus, you can use the Pen extension to stamp notes on a scale and produce a visual representation of your musical score.

# Musical List

scratch.mit.edu

## GET READY



Add the Music extension.



Choose any sprite.



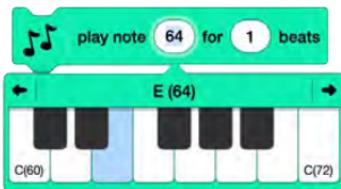
Keyboard

## ADD CODE

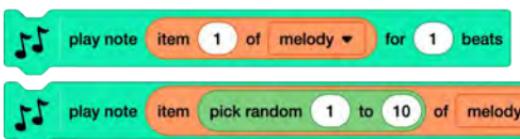
1. Create a list. Add song notes to the list via the stage monitor (add rows manually and type note numbers in) or by using the “add to [list]” block.



You can find note numbers by clicking on the input of the “play note” music block.

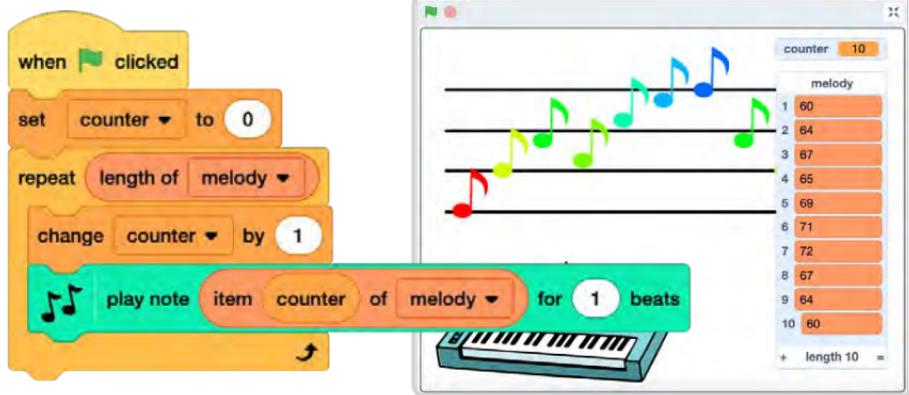


2. Write a script to play each note on the list by item number, or let the program pick the note to play randomly.



See the next card to learn how to create a “counter” variable to automate moving/repeating through the list in order.

# Generate a Melody: Repeat through a List



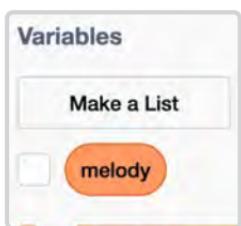
- While there is no “next item of list” block, you can create a script that loops through the items of a list in order.  
The ability to automate moving or repeating through a list can speed up your coding process and make editing scripts quicker.
- This can be useful if you want to add items of a list together, speak or say items in a list, etc.

# Generate a Melody

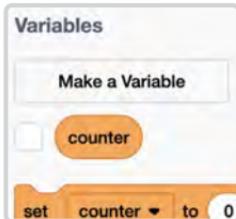
scratch.mit.edu

## GET READY

Create a list.



Create a variable.



## ADD CODE

Step through the code on the card front to see what it does:

1. Changes the “counter” variable (that stores a number to represent an item number on the list) by one.



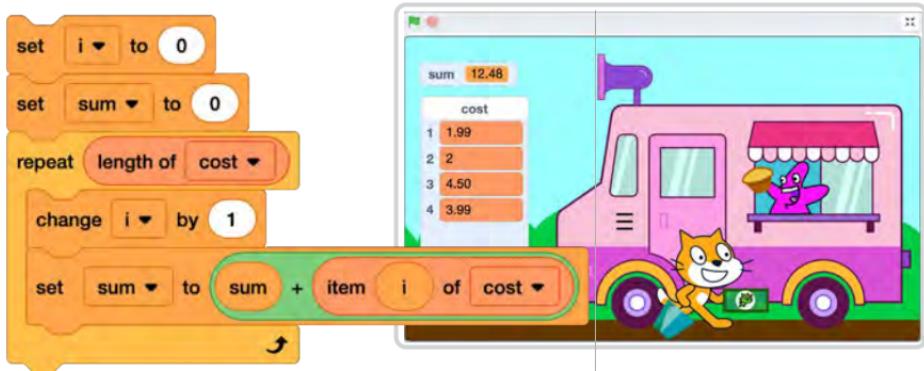
2. Plays the note number associated with that item number (the number entered on that line of the list). Note: This is why it is important to first set “counter” to zero first each time the program runs.



3. Repeat as many times as there are rows in the list/for the length of the list.



# Generate a Sum: Repeat through a List



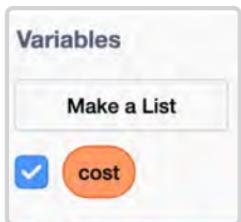
- While there is no “next item of list” block, you can create a script that loops through the items of a list in order.  
The ability to automate moving or repeating through a list can speed up your coding process and make editing scripts quicker.
- This can be useful if you want to add items of a list together, speak or say items in a list, etc.

# Generate a Sum

scratch.mit.edu

## GET READY

Create a list.



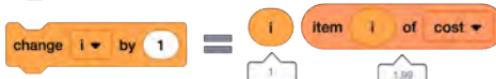
Create two variables.



## ADD CODE

Step through the code on the card front to see what it does:

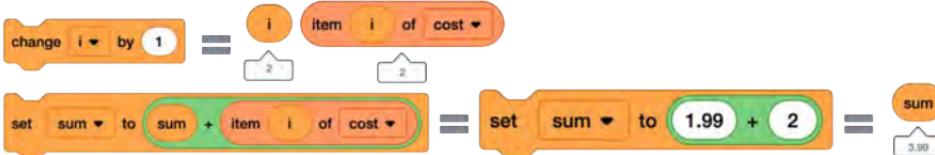
1. Changes the “i” variable (that stores a number to represent an item number on the list) by one.



2. Adds the amount associated with that item number (the number entered on that line of the list) to the value already stored in the “sum” variable, creating a new “sum” value. Note: This is why it is important to set “sum” and “i” to zero first each time the program runs.

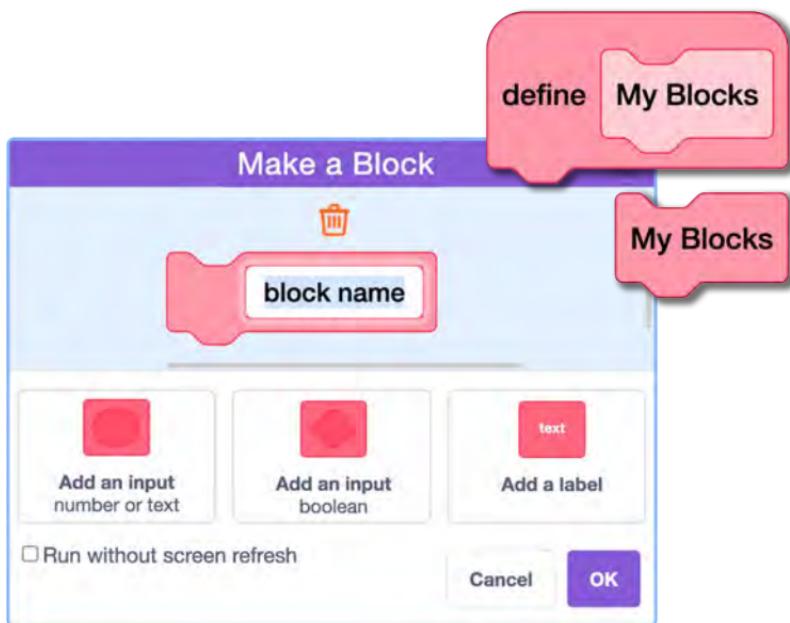


3. Repeat for the length of the list.





# My Blocks



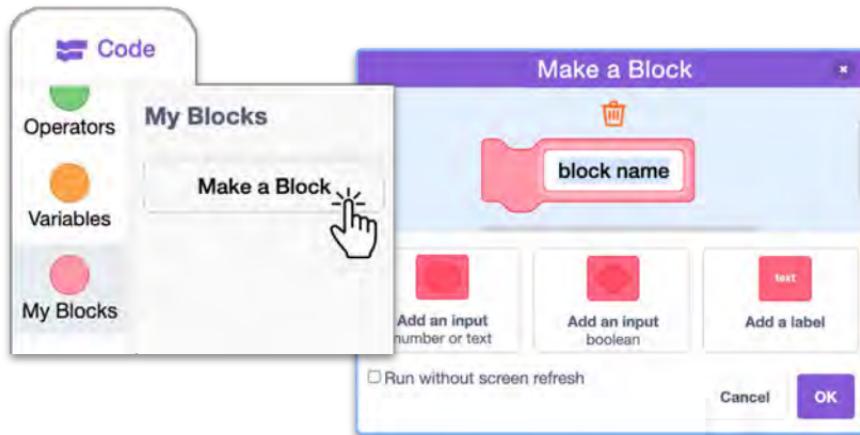
Explore procedures/routines and modularizing your code for efficiency



# Cards in This Pack

- Create a My Block
- My Block: Fade In and Out
- My Block: Music
- My Block with Parameters:  
Speak and Say
- My Block with Parameters: Move,  
Move
- My Block vs Broadcast

# Create a My Block



- Click on the “Make a block” button under the “My Blocks” category.
- Give your block any name you want, but it is best if it is a descriptive name so you can recall later what the block does.
- For a basic block, once you’ve provided a name, simply click “OK.”
- For an advanced block, add any additional inputs or labels needed.
- You can always edit blocks later.



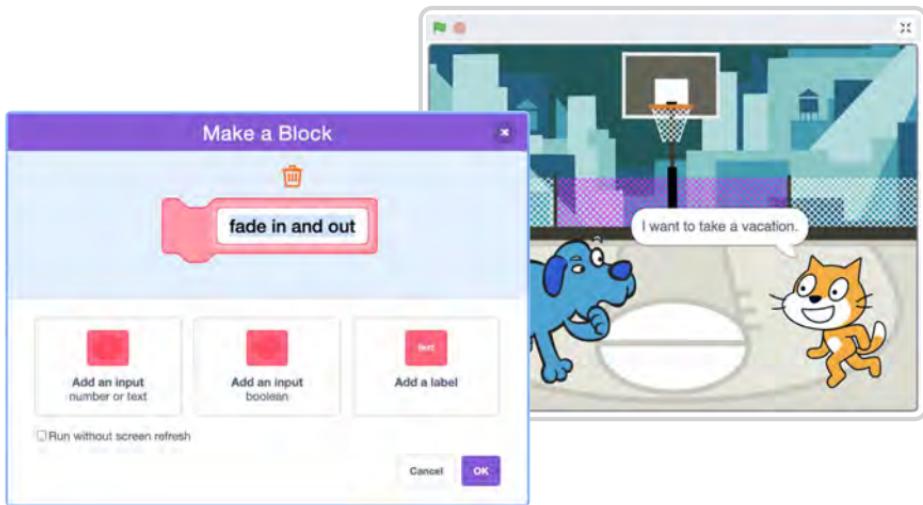
The image shows a Scratch script consisting of two parts. The left part is triggered by a green flag click, setting the stage to 'Space City 2' at coordinates (0, -40) and waiting 0.5 seconds. It then calls a custom block named 'cat spin'. The right part defines the 'cat spin' block, which repeats 20 times, changing the whirl effect by 50 and turning 36 degrees each time.

```
when green flag clicked
  go to [Space City 2 v]
  wait (0.5) seconds
  cat spin
end

define [cat spin v]
  repeat (20)
    change [whirl v] by (50)
    turn (36) degrees
  end
end
```

- Once a new block is created, a “define” event handler block will appear on the scripts area. Place all the blocks to make up your steps (procedure) under it.
- Once defined, you can use your custom block in your main program.
- Creating separate procedures as custom blocks makes the code faster to write and read, and easier or quicker to edit.
- Note: A custom block is specific to the sprite where it was defined.

# My Block: Fade In and Out



- Say you are creating an animation and you want a sprite to fade in and out as the scene changes/change opacity (also known as the “ghost” effect in Scratch).
- Rather than write the same steps over multiple times in a program, you can place those steps in a My Block and simply call that block each time you need it.

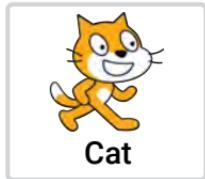
# My Block: Fade In and Out

scratch.mit.edu

## GET READY



Choose any sprite.

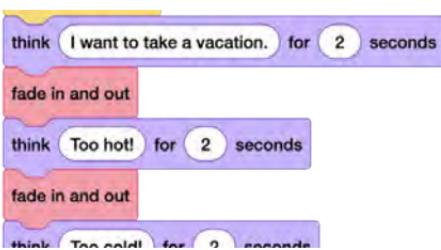
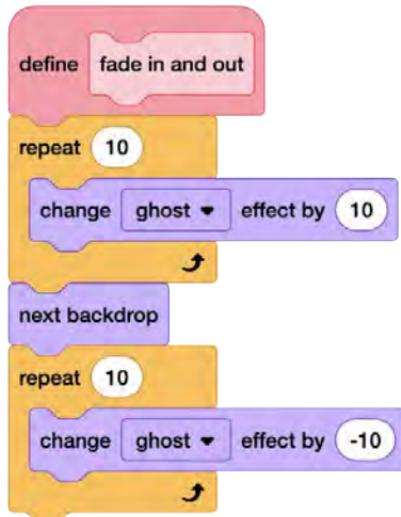


Choose any backdrop.

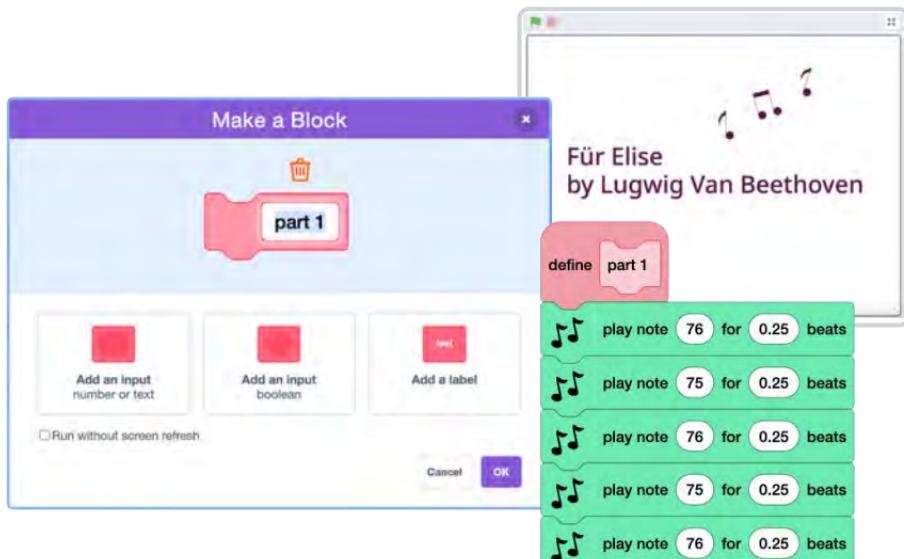


## ADD CODE

1. Create a My Blocked called something like “fade in and out.”
2. Under the “define” block that appears on the stage, add steps to change the ghost effect to 100, switch the backdrop, and then change the ghost effect to 0.
3. Use the My Block in your main program each time you want the sprite to fade in and out and the backdrop to change.



# My Block: Music



- You can use music blocks from the Music extension to create a song in Scratch.
- Rather than write the same sequence of notes over and over when they repeat in your song, you can place those notes in a My Block and simply call that block each time you need it, for instance each time a chorus is called.

# My Block: Music

scratch.mit.edu

## GET READY



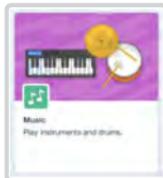
Choose any sprite.



Keyboard



Add Music extension.



## ADD CODE



1. Compose the sections of your song. Create multiple My Blocks for different parts (such as verse and chorus).
2. My Blocks can also be placed within other My Blocks to further simplify the code.
3. Use My Blocks in the main program, along with repeat blocks (if applicable) to compose a whole song. Set the instrument and the tempo.

# My Block with Parameters: Speak and Say



- What if you want to perform the same procedure (set of steps) each time the custom block is called in the main program, but with a small modification each time (like the text shown or spoken)? Create a My Block with parameters!
- With an input in place, the custom block will use the parameter (the data provided in the input bubble) when running.

# My Block with Parameters: Speak and Say

scratch.mit.edu

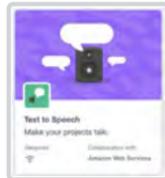
## GET READY



Choose any  
sprite.

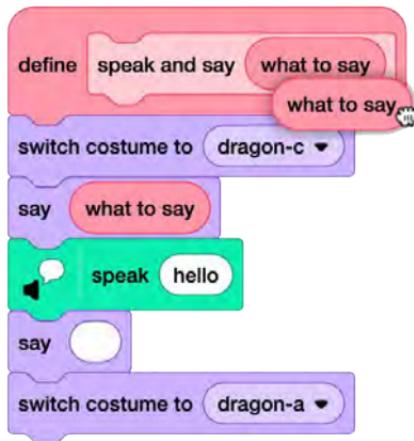


Add Text to  
Speech  
extension.



## ADD CODE

1. Add an input when creating your My Block.
2. When defining the block steps, click on the input label and drag it out to place it in a code block. In this case, add it to the first “say” and “speak” blocks.
3. Now, when you use this custom block in the main program, you can see the blank input bubble where you can enter the parameter. In this case, the parameter is the text to say and speak.



# My Block with Parameters: Move, Move



- What if you want to perform the same procedure (set of steps) each time the custom block is called in the main program, but with a small modification each time (like the sprites coordinates)? Create a My Block with parameters!
- With an input in place, the custom block will use the parameter (the data provided in the input bubble) when running.

# My Block with Parameters: Move, Move

scratch.mit.edu

## GET READY



Choose any sprite.

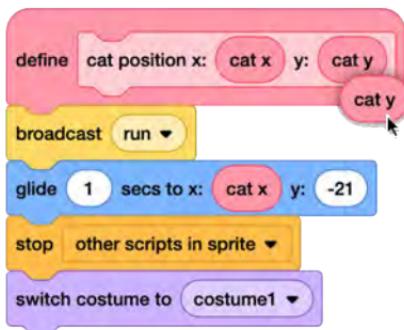


Choose any backdrop.



## ADD CODE

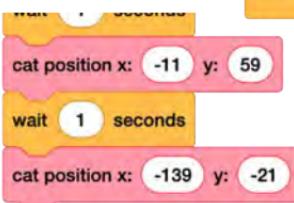
1. Add two inputs when creating your My Block (for x and y position). Add a label (descriptive text) between the inputs to help you remember what each input bubble is for when you use it in your main program.



2. When defining the block steps, click on the input label and drag it out to place it in a code block. In this case, x and y position. (Note, this My Block also sends a broadcast.)



3. Now, when you use this custom block in the main program, enter the new parameters each time.



# My Block vs Broadcast



- My Block: the program pauses and runs through all the steps under the “define” block before preceding.
- Broadcast: the program sends the message and then proceeds with the next steps in the program, so code sequences may run simultaneously.
- Note: Unlike a broadcast that can be sent globally between all sprites and backdrops, a My Block is local, usable only by the sprite it is defined on. The call for the custom block isn’t received by any other sprites, even if their custom block has the same name.

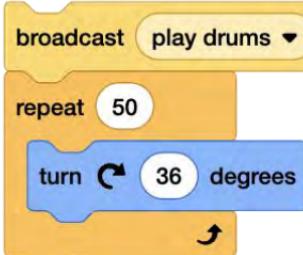
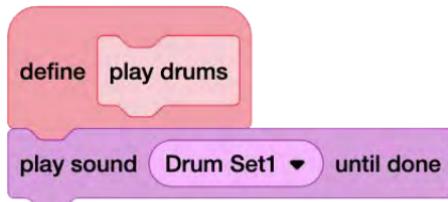
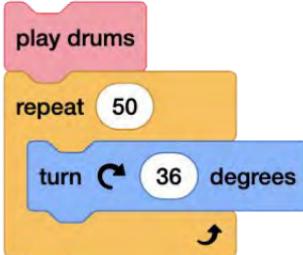
# My Block vs Broadcast

scratch.mit.edu

## EXPERIMENT

Try these two code sequence pairings to see the difference between calling for a custom block and calling for a broadcast.

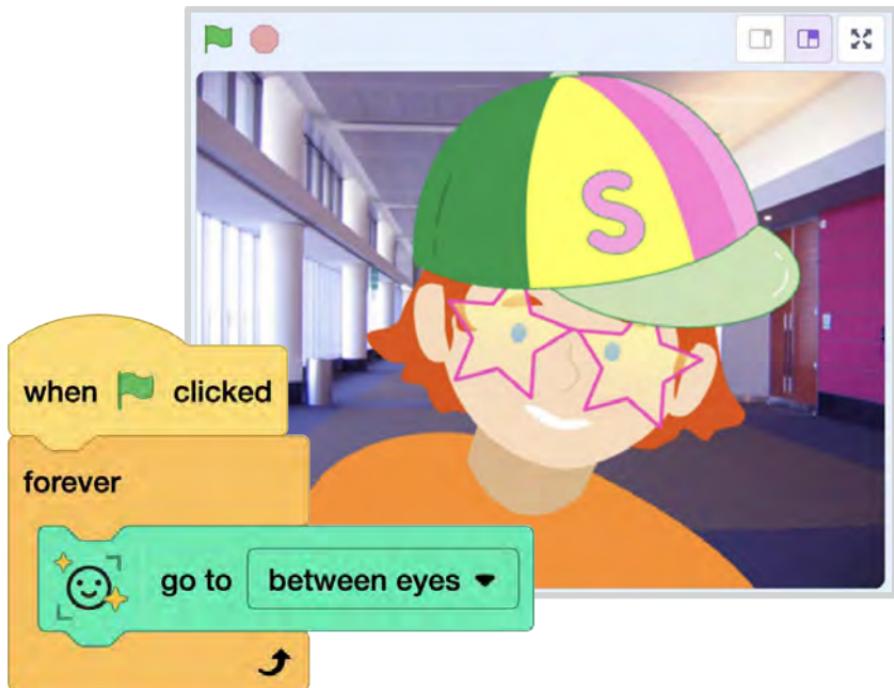
What else could you try? Costume changes? Motion?



What if you used  
“broadcast and wait”?



# Scratch and AI: Face Sensing



Use AI face detection to make interactive  
Scratch projects



# Cards in This Pack

- Try Out Face Sensing
- Create a Face Filter
- Create a Face Sensing Game
- Create a Face Sensing Sound Board
- Use Your Nose As a Pen
- Say It and Spray It
- Fool the AI

*Face Sensing blocks use a device's camera, so you may be prompted to give permission to use the camera in your browser. **These blocks do not record or store your video.***

# Try Out Face Sensing



This AI-powered extension uses a machine learning model to detect if it sees a face and where a nose, eyes, ears, mouth, etc., are. The model was trained using a large data set of images of faces.

**Click a “go to nose” block while your face is visible on the stage.** Did the sprite go to your nose? What happens if you choose another feature?

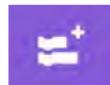
# Try Out Face Sensing

scratch.mit.edu

## GET READY



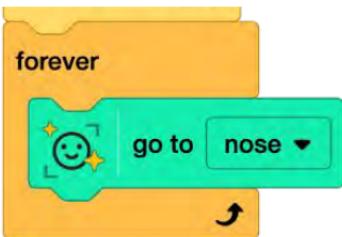
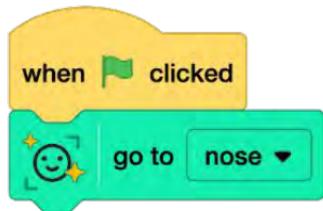
Choose any sprite.



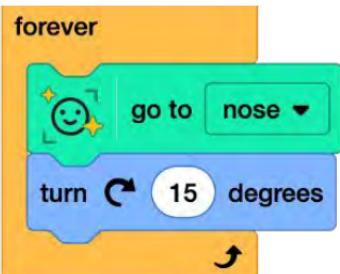
Add the Face Sensing Extension.



## ADD CODE



1. Add a “go to [nose]” block. Click it to see the result.

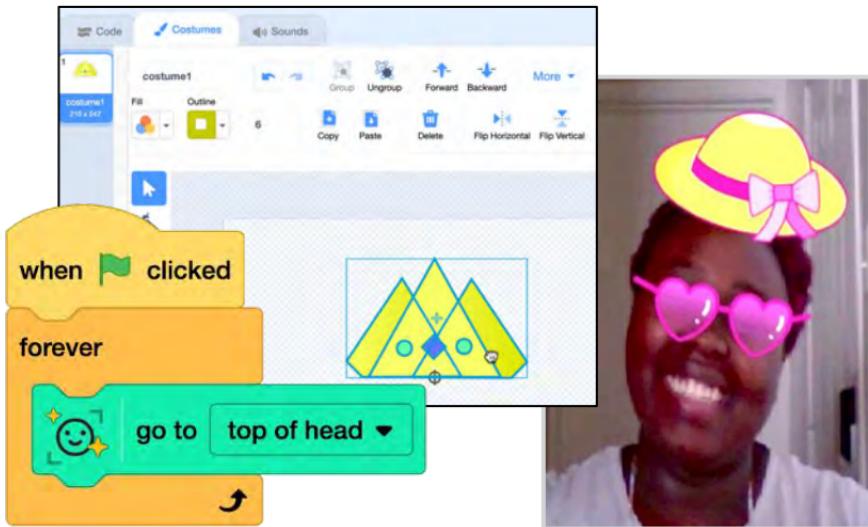


2. Next, add a “forever” loop to have the sprite stick to your chosen feature.

3. Try adding additional blocks from the Motion or Looks category to animate the sprite.

*What if more than one face is on screen? The blocks can only interact with one face at a time.*

# Create a Face Filter



- Use the Fashion sprites in the sprite library or draw your own hat, glasses, or other accessory with the **Scratch Paint Editor** tools.
- **Code a face filter** that follows your face as you move it around the stage.

# Face Filter

scratch.mit.edu

## GET READY



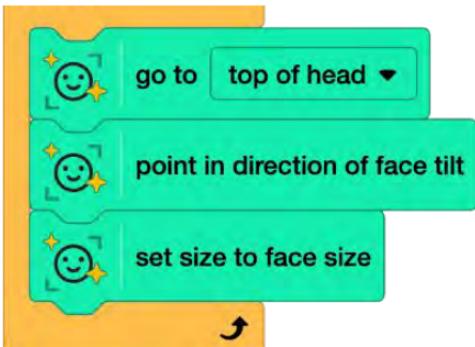
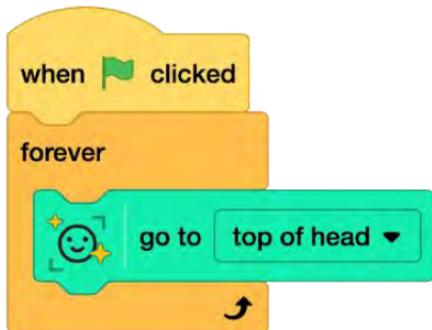
Choose any sprite.



Or use “Paint” to create your own costume or two.



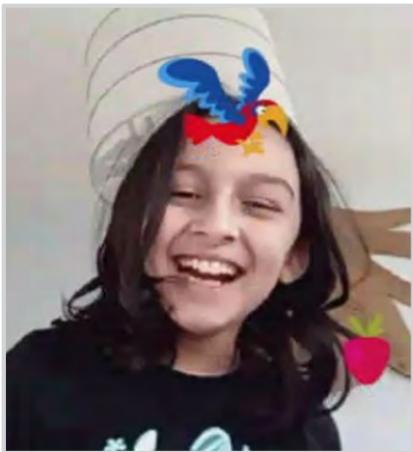
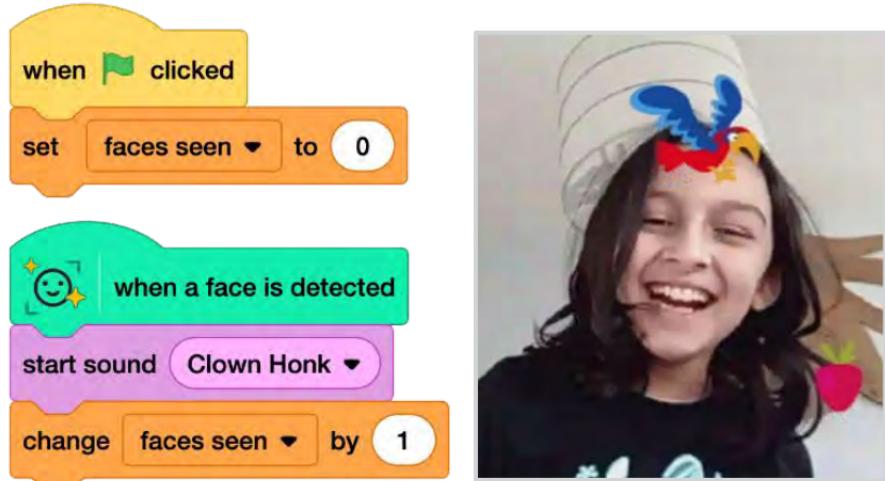
## ADD CODE



1. Add code so it sticks to a feature of your face, like the top of the head.
2. Experiment with other blocks to make the sprite scale to match the size of your face and point in the direction of your face.

*Need to adjust the sprite’s position on your face? Try moving on the Costumes tab relative to the costume center. You could also add code to switch costumes.*

# Create a Face Sensing Game



- Create a **face detection counter**
- Code a game that uses your face to **score points**, like a clicker game that uses your nose instead of a mouse
- Code a game that uses your face to **control a player sprite**, triggering jumps or helping it move around the stage or point in a direction

# Face Sensing Game

scratch.mit.edu

## GET READY



Choose a first sprite.



Parrot



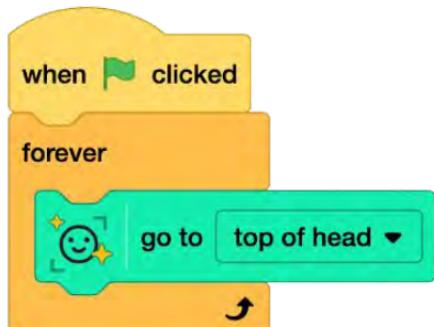
Choose a second sprite.



Strawberry

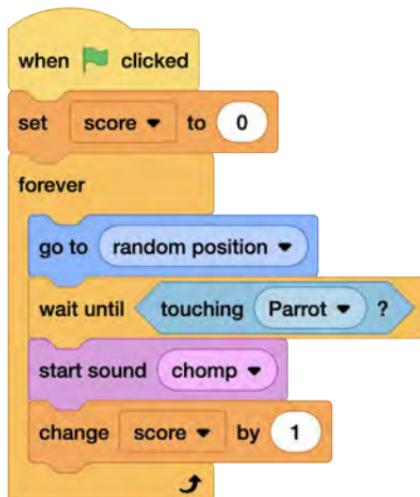
## ADD CODE

1. Add code to the first sprite so you can control it with your face. This will be the player.



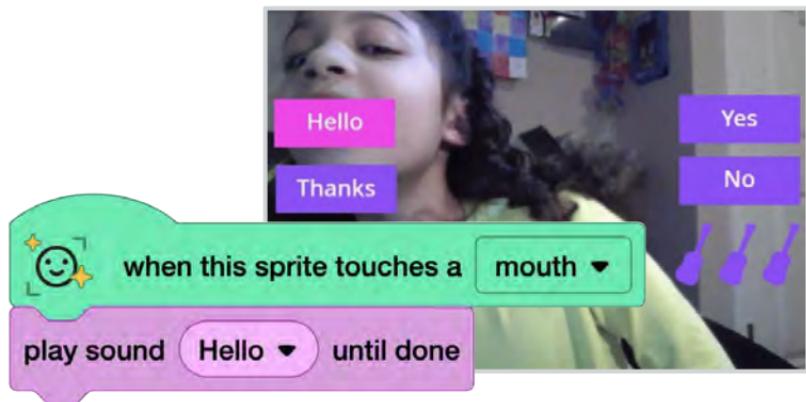
2. Create a score variable to track points. Don't forget the program will need to reset it each time a new game is started.

3. Add code to the second sprite so it moves to a random position on the stage and gives the player a point when sprites touch.



*Try making the game harder by moving the second sprite after a time, if not touched yet.*

# Create a Face Sensing Sound Board



- Choose a variety of fun sounds or record your own.
- Code a sound board that uses parts of your face to play sounds.
- Code effects that add visual confirmation, like color changes.
- Create a mystery button that picks a sound to play at random.

# Sound Board

scratch.mit.edu

## GET READY



Choose a few sprites, or draw your own.

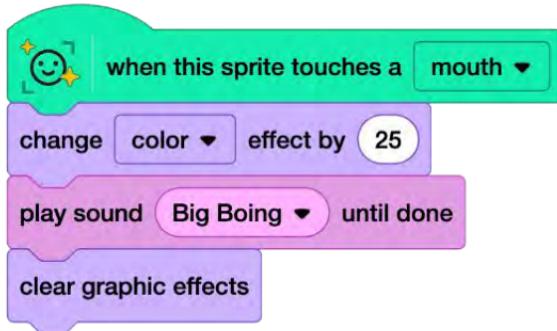


Choose a sound from the sound library for each sprite, or record your own.

[Choose a Sound](#)

## ADD CODE

1. Add code to each sprite to play a sound, change an effect, or perform another animation when parts of your face touch them.



2. Try adding multiple sounds to a sprite. Use the “pick random” operator so each time is a surprise.



# Use Your Nose As a Pen



- Combine blocks from the Face Sensing and the **Pen Extension** for a wild way to draw!
- Stick a sprite to your nose or eyes, put the **pen down**, and move it around the stage to create a unique art piece.
- Want more control? Try using your head **tilt to put the pen up and down**. Or try adjusting the **pen size based on your face size**.

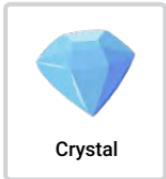
# Use Your Nose As a Pen

scratch.mit.edu

## GET READY



Choose any sprite to act as the Pen.

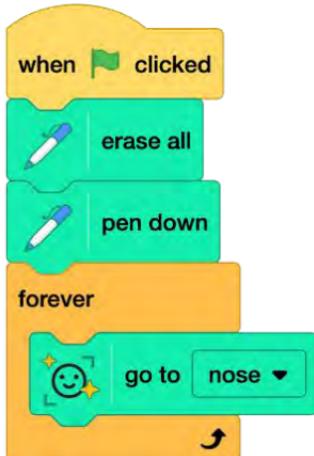
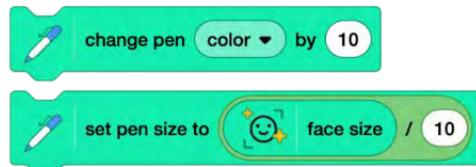


Add Pen Extension.

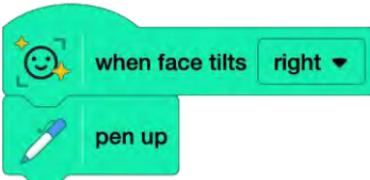
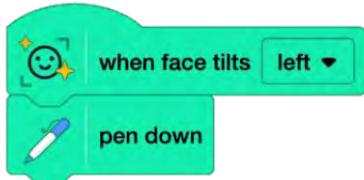


## ADD CODE

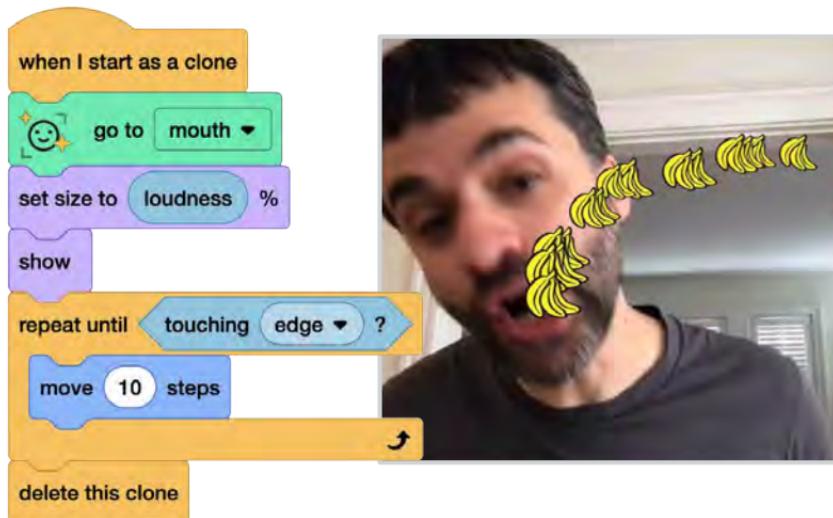
1. Add a Pen block to put the pen down. Then have the pen follow your nose.
2. Try variations like changing the pen color or setting the pen size based on your face size.



3. Want more control? Use “when face tilts” to control when the pen is up and when it is down.



# Say It and Spray It



- Code sprite **clones** to spray from your mouth!
- Control the size and visibility of clones using the “**loudness**” block.
- For even more fun, set up different sprite **costumes that are chosen at random**, so what sprays out is a continual surprise.

# Say It and Spray It

scratch.mit.edu

## GET READY

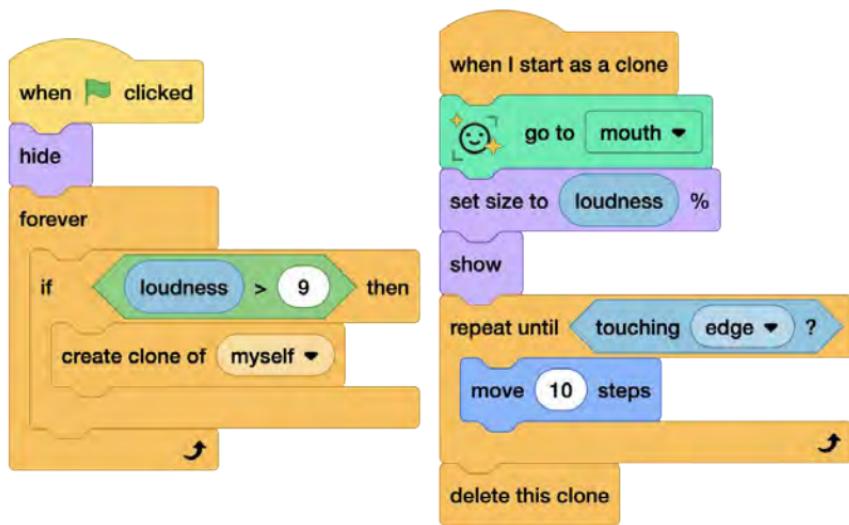


Choose any sprite.



## ADD CODE

1. To have multiple copies of a sprite spray from your mouth, you can create clones and use “loudness” to trigger their creation.
2. Have clones go to your mouth, then move repeatedly. You could delete clones when they reach the edge of the stage, and set or change size based on loudness.



# Fool the AI

*Face Sensing blocks try to detect if a face exists, but they are not able to identify who the face is, or even if it is a human face! That means sometimes the AI makes interesting mistakes. Identifying these mistakes can help us see the difference between our own human intelligence and AI.*

Can the AI find the parts of a face if:

- you are in disguise, your face is covered, or your face is tilted or upside down?
- the lighting in the room is very bright or very dark?
- you step out of frame and hold up a drawing of a smiley face? a stuffed animal? a pet? two googly eyes attached to fingertips? or another facelike object made of different materials or from nature?

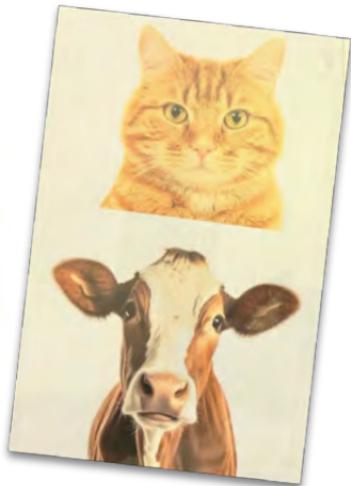
What variables can you change to try to fool it into thinking it sees a face? What limitations can you find?

# Fool the AI

scratch.mit.edu

The AI that we're using is trained to detect human faces.

**False positives** are things that are not actually a human face (false) that it detects as a face (positive). What are some false positives you can find?

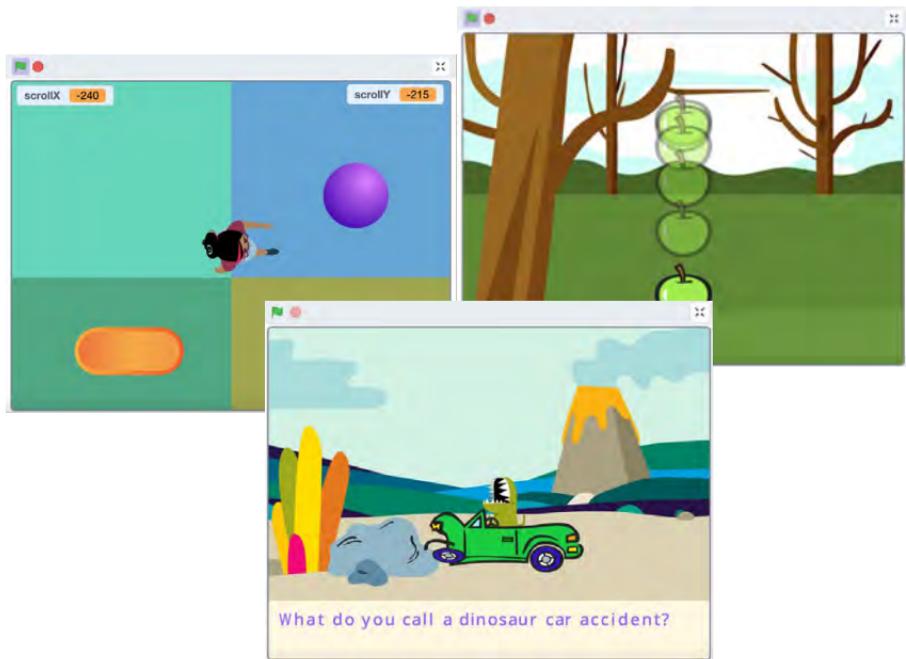


Why might some animal faces (like cats) be recognized while others (like cows or dogs) may not be seen as a face? Think about the proportions of a human face and how those are different from a cow's longer snout.

A **false negative** when working with the Face Sensing blocks is when a human face is in the frame, but the AI does not detect it (perhaps because it is too small or upside down). What are some false negatives you can find? What might be causing their detection failure?



# Advanced Topics: Graphic Effects



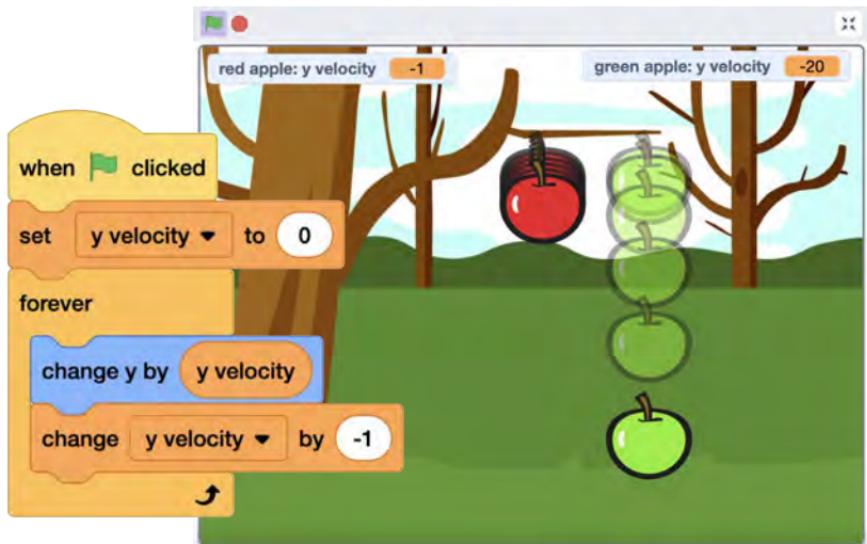
Use data to create more advanced scripts for games, stories, and more!



# 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

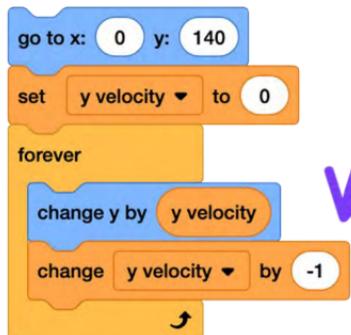
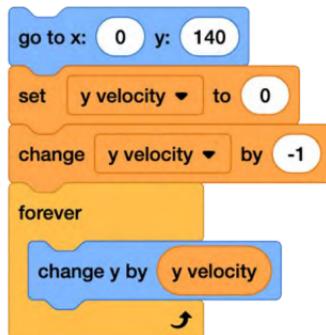
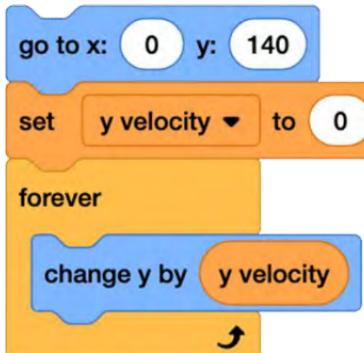
1. Create a variable that will hold the velocity of your sprite, like “y velocity.”
2. Set the velocity to zero at the start (so the object will start from a place of rest).
3. Then, have the sprite move (in this case, change y) by that velocity.
4. In order to gain speed (acceleration), the velocity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



Choose a sprite.

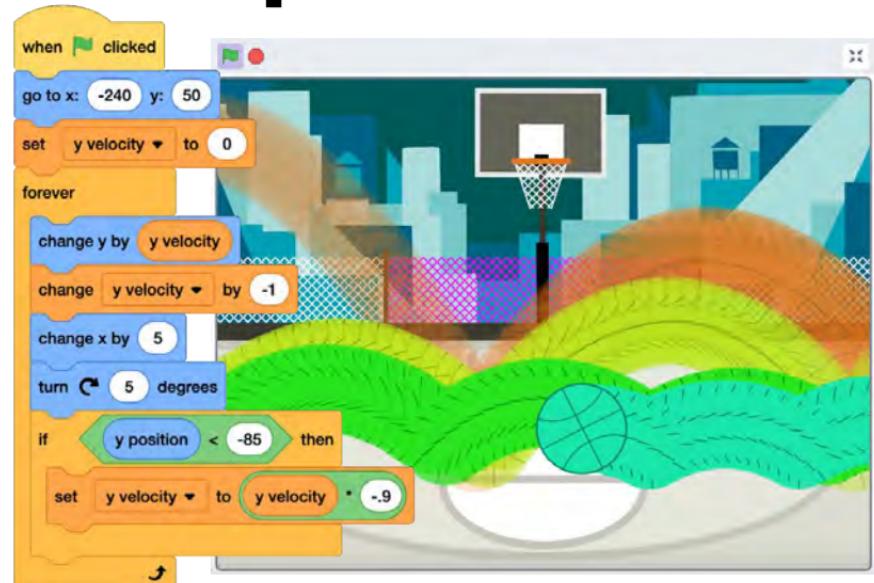


Apple



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.

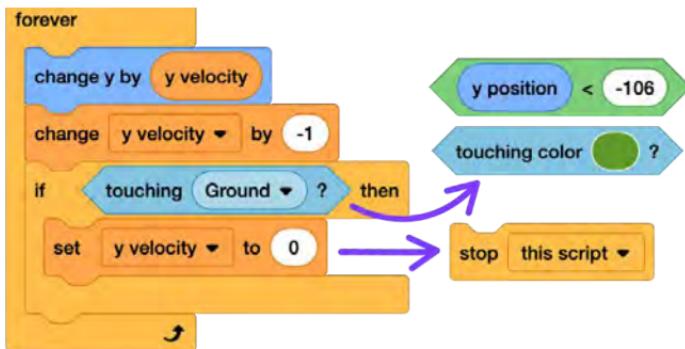
The object could come to a complete stop. Or sprites may bounce when they reach the ground (or the bottom of the stage), with those bounces getting smaller and smaller over time.

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

# Gravity: Stop or Bounce

scratch.mit.edu

- 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.



- 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.
- 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



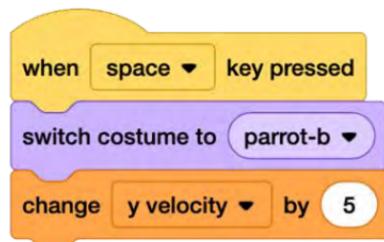
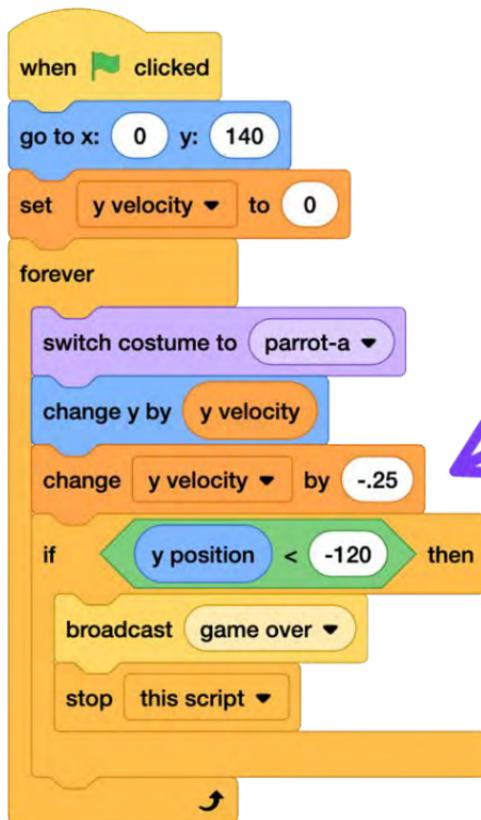
What if you wanted to, at least temporarily, reverse gravity? If you made a sprite bounce, you reversed gravity. Another example is in the case of a “flappy bird”-style game, where pressing a keyboard key temporarily reverses gravity and stops a sprite from hitting other objects or the ground in order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

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

# Gravity: Reverse Gravity

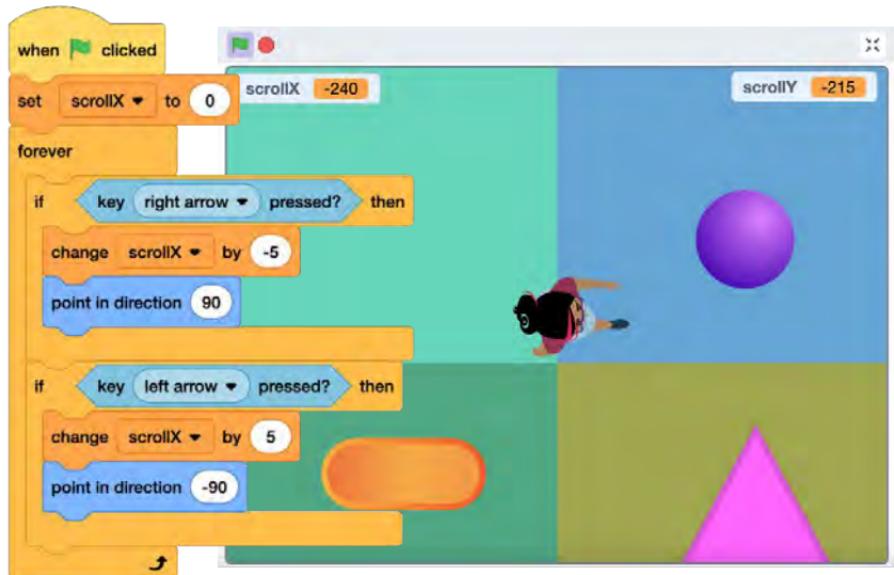
scratch.mit.edu

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



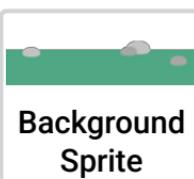
Choose a sprite, perhaps with costumes to show movement.



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 ( $480 \times 0 = 0$ ). The next at 480 times one, etc.

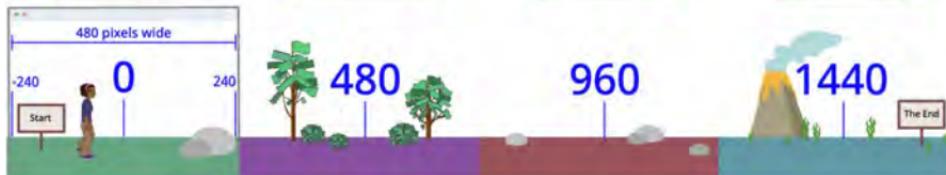


480 \* 0

480 \* 1

480 \* 2

480 \* 3



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

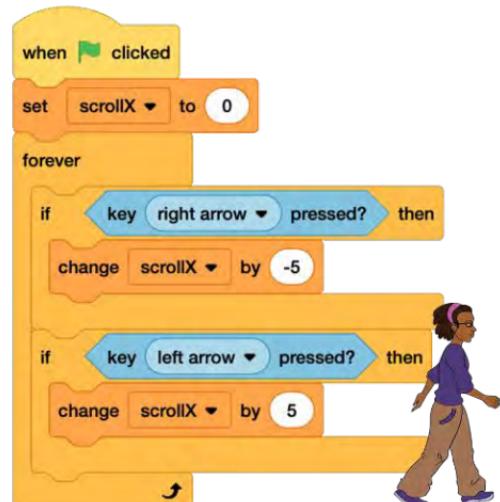
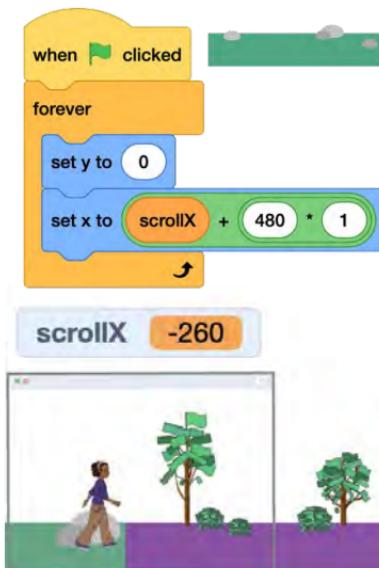
scrollX

(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!



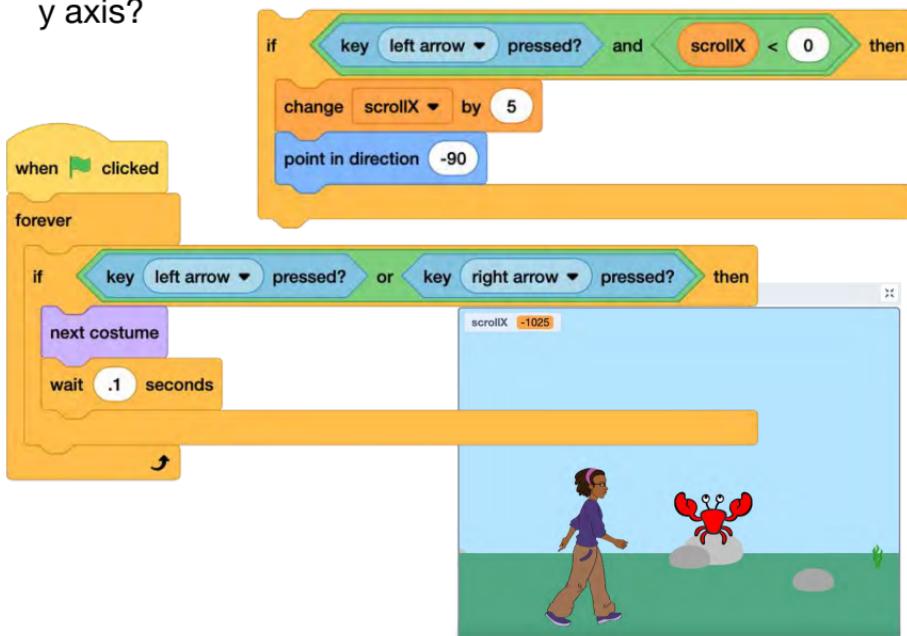
(Example here: [scratch.mit.edu/projects/969838240](https://scratch.mit.edu/projects/969838240).)

# User-Controlled Scrolling Background

scratch.mit.edu

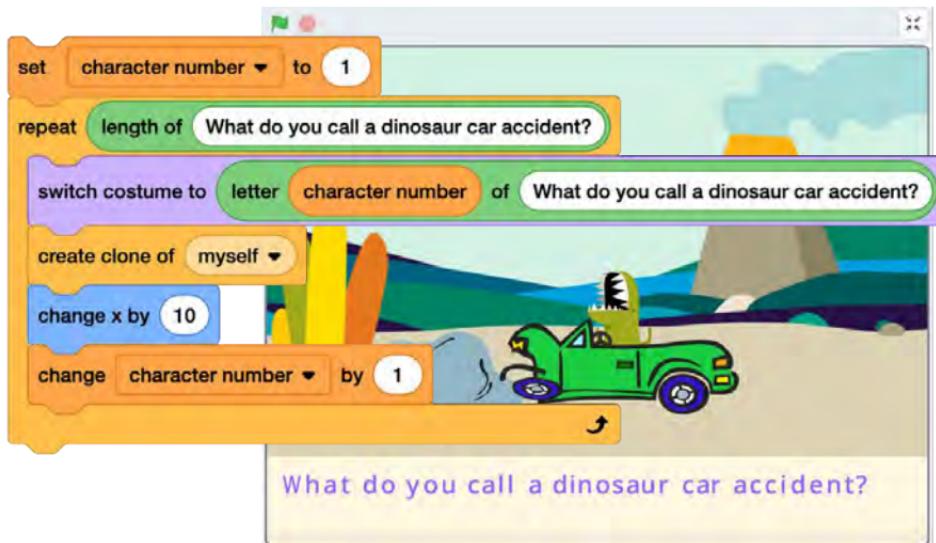
## EXPERIMENT!

- 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](https://scratch.mit.edu/projects/970482174) and [970438551](https://scratch.mit.edu/projects/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!

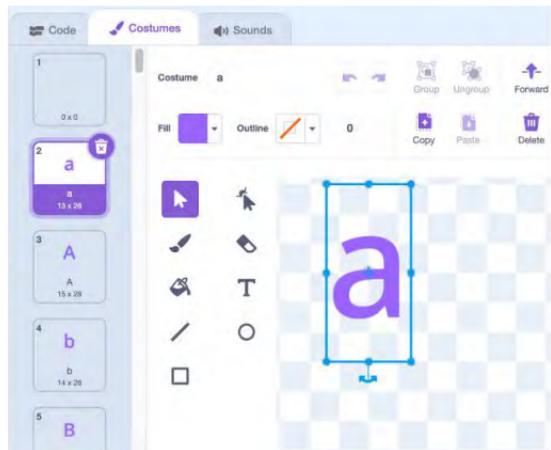
Then, you can customize the text, add effects, and more!

(Example here: [scratch.mit.edu/projects/985124543](https://scratch.mit.edu/projects/985124543).)

# Text Rendering

scratch.mit.edu

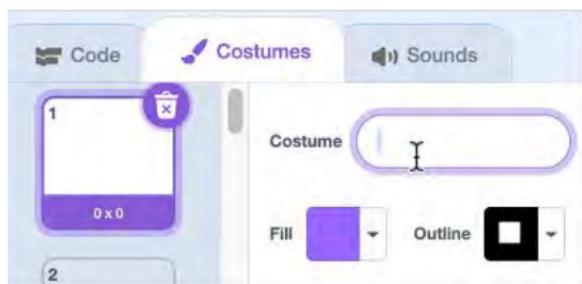
## GET READY



The name of the costume should match the contents exactly.

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

Choose any font or color. Center each.



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

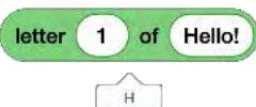
\*See the back of card 9 for additional letter sprite creation options.

# Text Rendering

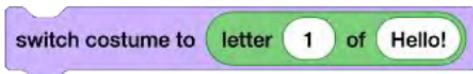
scratch.mit.edu

## ADD CODE

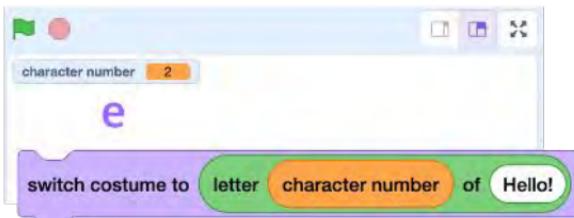
1. Locate the “letter of” reporter block. Enter some text, then click on the block. In programming, a sequence of characters that represents text is called a “string.” Strings can be used to store human-readable data, such as words or sentences. Experiment with the number and text string to see what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



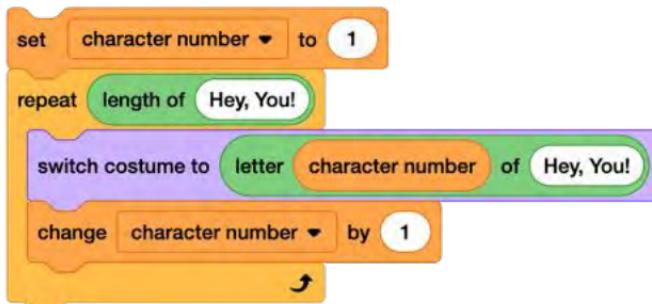
3. Now, create a variable, like “character number” to use in place of a fixed number in the “letter of” block.



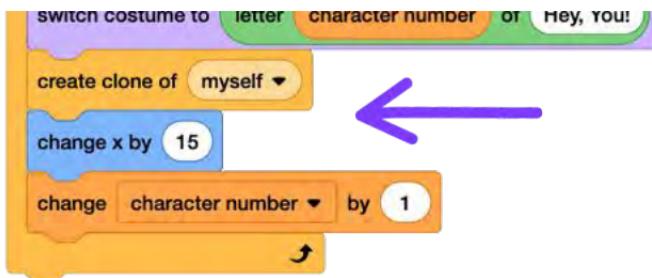
# Text Rendering

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.

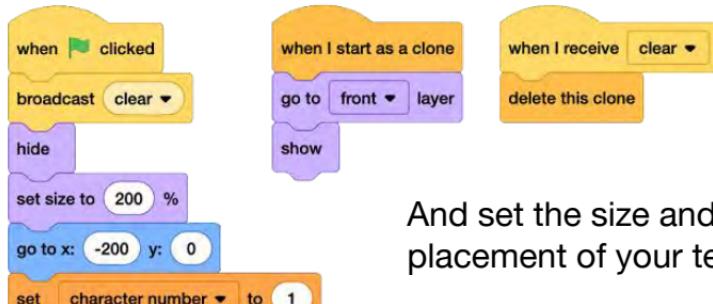


Tinker with the amount you move/change x by based on the size of your letters.

# Text Rendering

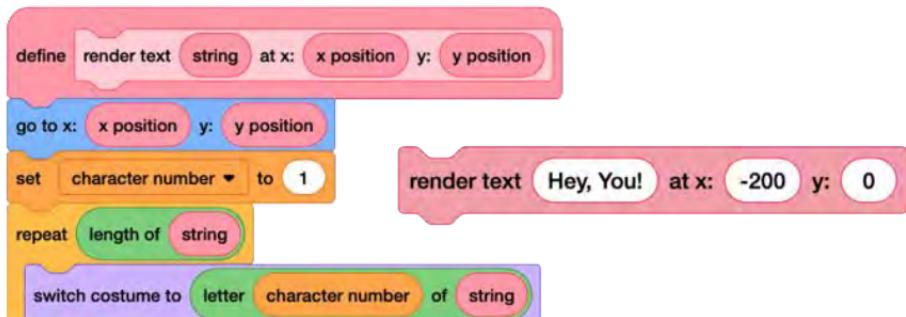
scratch.mit.edu

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.

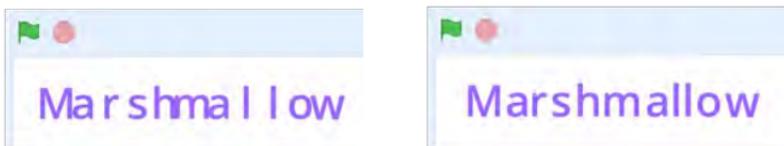
**OPTIONAL:** Have multiple lines of text to display? Try creating a procedure using a My Block, so you can quickly define the string (and its placement, if desired) without duplicating the entire code stack. (See our [My Blocks cards](#) for more information.)



# 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.

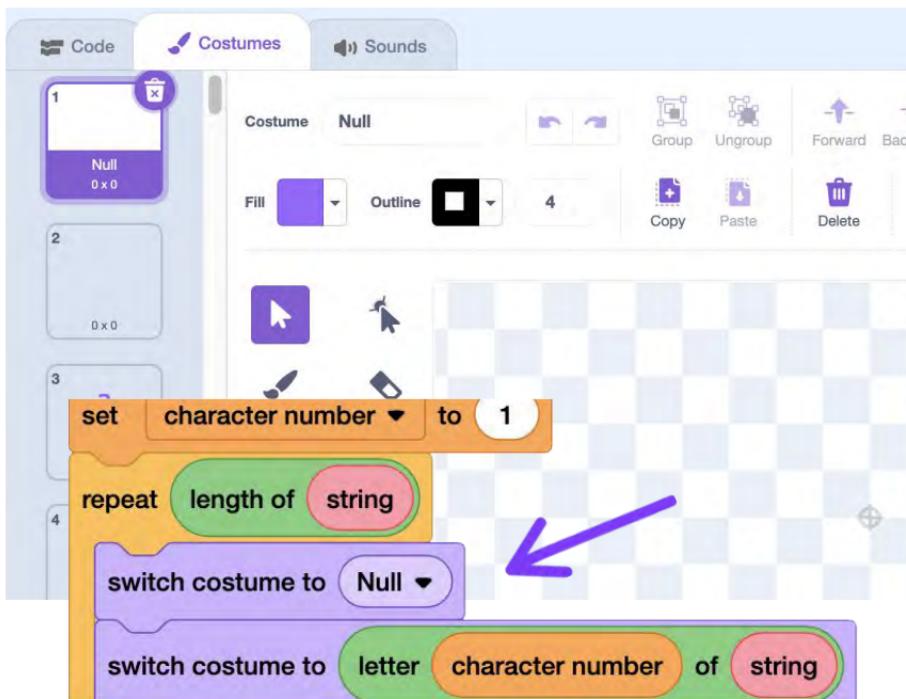
The image shows a Scratch script and a table of letter widths. The script uses a combination of color sensing and a list of widths to achieve even letter spacing. It starts with a 'set [character number v] to [1]' block, followed by a 'repeat [length of <Marshmallow>]' loop. Inside the loop, it uses a 'switch costume to [letter <character number> of <Marshmallow>]' block to set the current character's width. It then checks if the current character is a space ('costume number = 1') and, if so, changes the x-position by 30 pixels. If not, it repeats until it finds a non-space character ('not [touching color [purple v]]?'). It then changes the x-position by 2 pixels for each non-space character. Finally, it creates a clone of itself and changes the character number by 1 for the next iteration. A callout box points to the 'if costume number = 1 then' block with the text: "In this example, costume 1 is space." Another callout box points to the 'change character number by 1' block with the text: "change character number by 1".

	letter width
1	0
2	20
3	13
4	15
5	14
6	15

# 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).



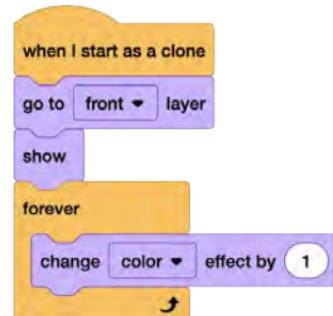
# Text Rendering

scratch.mit.edu

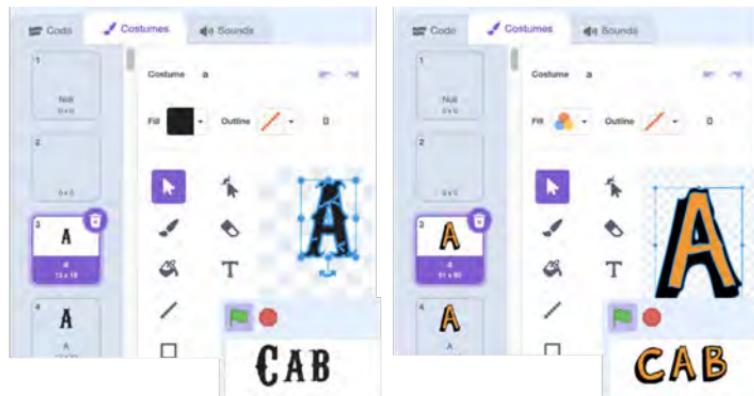
**OPTIONAL:** Instead of creating a clone, you could use the "stamp" Pen block to record your letters. But what might be some advantages or disadvantages of creating a clone versus using a stamp?

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?

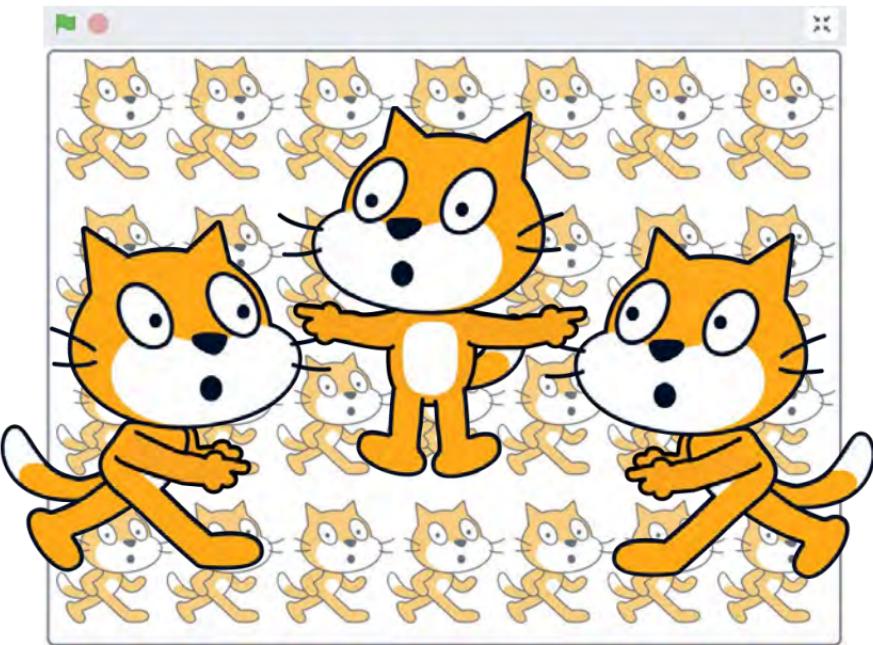


**OPTIONAL:** To quickly get started with letters, backpack or export/import the "Letter - BACKPACK THIS" sprite in [scratch.mit.edu/projects/1138479378](https://scratch.mit.edu/projects/1138479378). Or what if you want to use a font not available in the Paint Editor? You could upload SVG or transparent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





# Advanced Topics: Clones



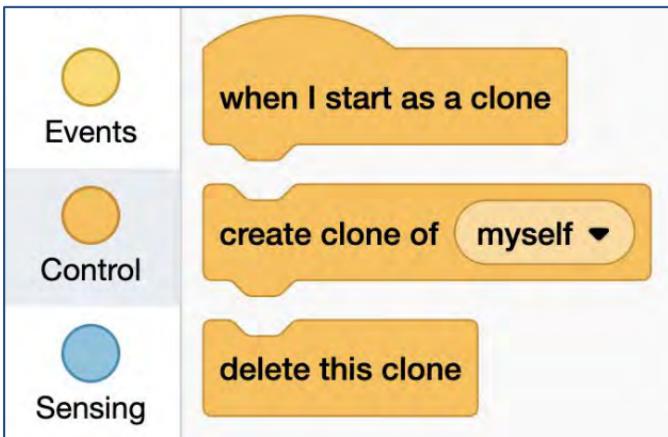
Use clones to create multiple sprites for more efficient and advanced projects.



# Cards in This Pack

- Clone 101
- More Clone 101
- Balloon Pop Clones: Quick Game Creation
- Clone Piano: Using Local Variables
- Clone Piano: Using Global Variables
- Fractal Tree: Clones Making Clones
- Clone IDs: Generating and Using

# Clone 101



Cloning lets you create multiple copies of your sprite while your project is running. When each clone is produced, it has the **same costumes, sounds, scripts, and variables as the original, but it is otherwise independent**.

There are **three blocks that are specific to clones, which can be found in the Control category**: “create clone,” “delete this clone,” and “when I start as a clone” (where you can define code that only applies to the clones and not the original sprite).

# Clone 101

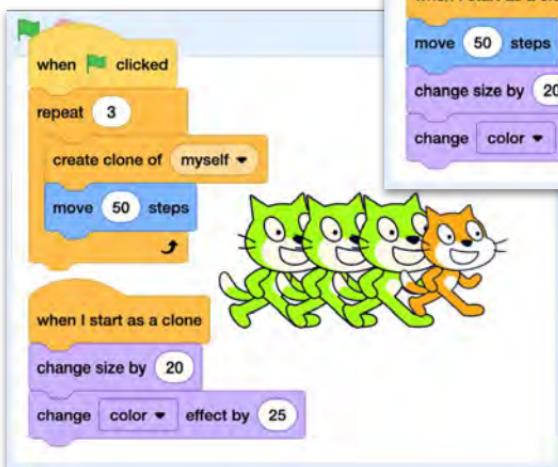
scratch.mit.edu

What differences do you notice in the code and the results between these three similar scripts?



Sequence and project goals are important.

In the first case, the original sprite moves and changes, creating clones along the way.



In other cases, the clone is created and then controls its own movement and/or effects.

# More Clone 101



When a clone is created, the **scripts for the original sprite also apply to a clone**. That means that scripts like “when this sprite clicked” or “when I receive [broadcast]” apply to them, too.

One exception is scripts that have started running before the clone was created (like a “when green flag clicked” script). Code that should be applied to a clone needs to be triggered after their creation (via a broadcast, click, key press, “when I start as a clone,” etc.).

# More Clone 101

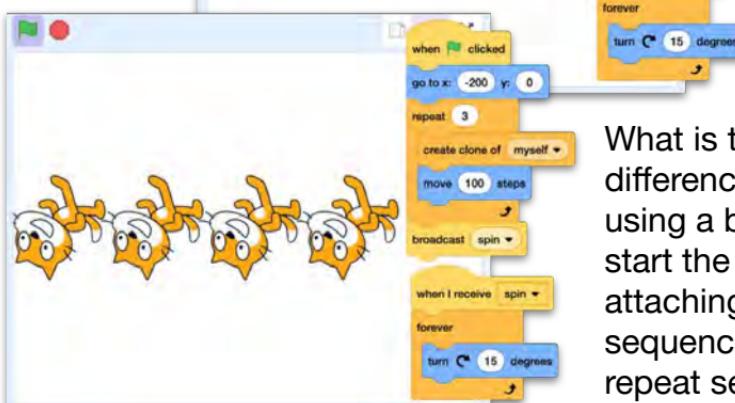
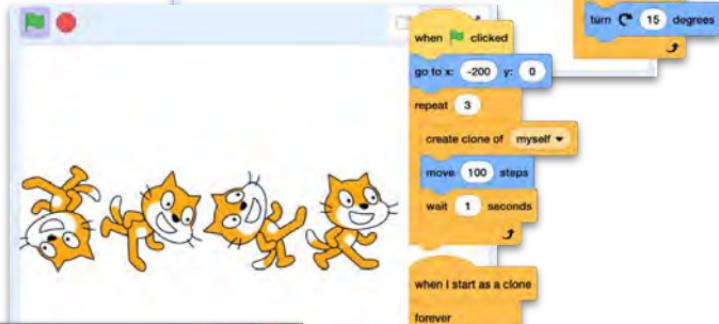
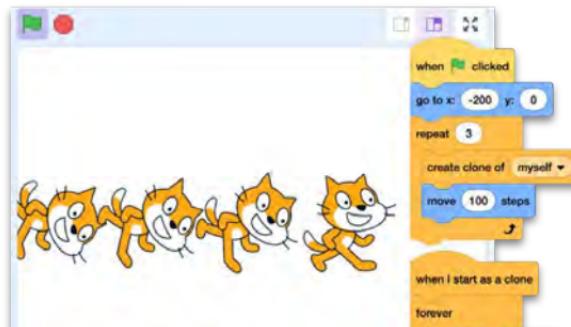
scratch.mit.edu

What differences do you notice in the code and the results between these three similar scripts?

How does timing and sequence affect when the clones start turning?

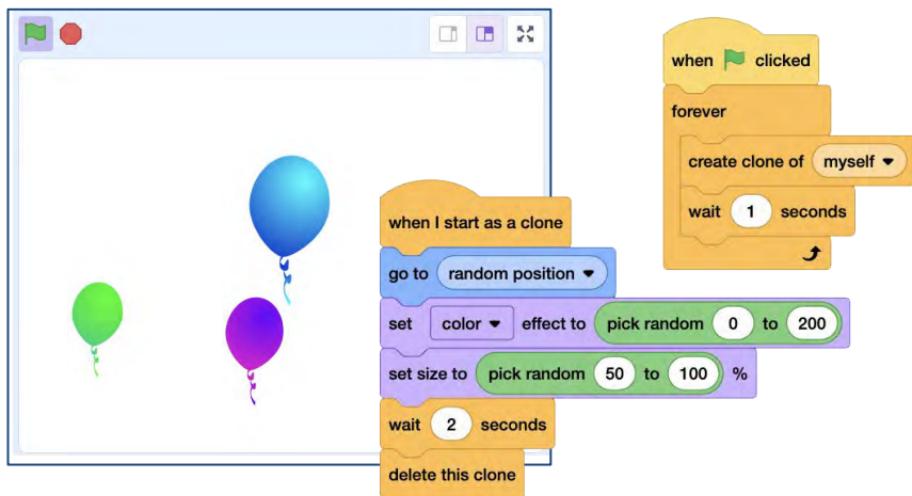
Why is the original sprite turning in one example and not the others?

What if the spin was triggered by clicking the sprite instead?



What is the difference between using a broadcast to start the turning vs attaching the forever sequence after the repeat sequence?

# Balloon Pop Clones: Quick Game Creation



You can **use clones to create a repeating animation or game with objects that repeatedly appear to interact with**.

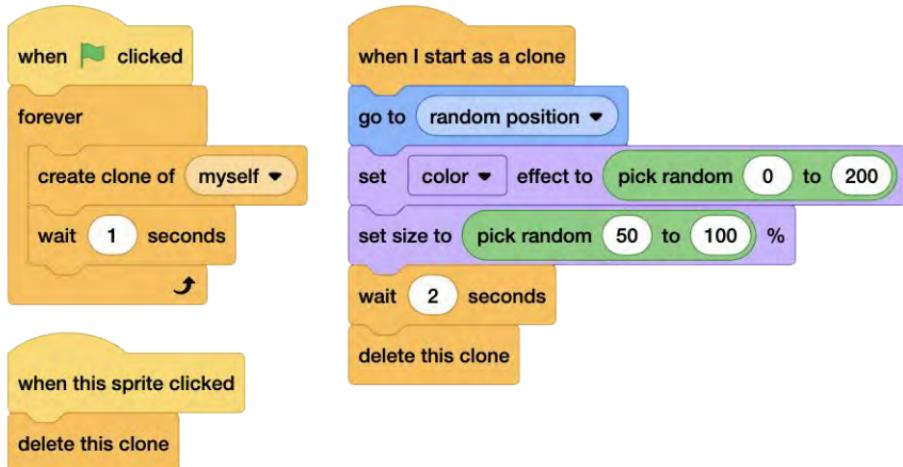
For instance, you could make an animation where balloon clones are created every few seconds and then disappear or can be popped by the user.

(Example here: [scratch.mit.edu/projects/1154244503](https://scratch.mit.edu/projects/1154244503).)

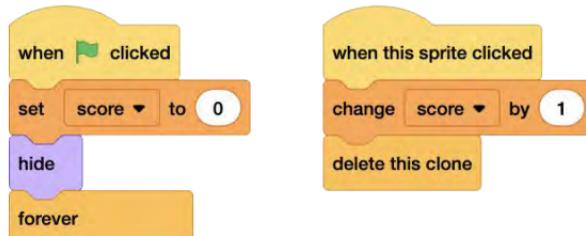
# Balloon Pop Clones

scratch.mit.edu

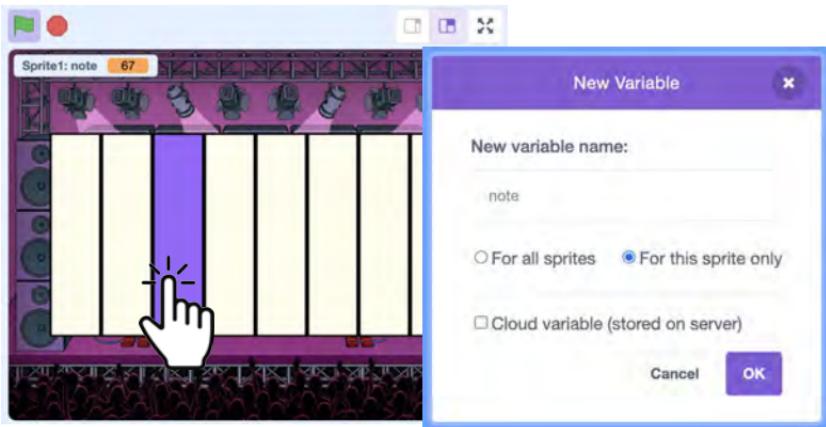
1. Create a script that makes a clone every few seconds forever when the green flag is clicked. You could...
  - o have the original sprite move around and change color or size before creating a clone
  - o have the clones adjust their own settings after they have been created



2. What if you want to add a score? Where would you add blocks to set and change the score? You may also need to hide the original and show the clones.



# Clone Piano: Using Local Variables



A **local variable** ("For this sprite only") is individual to a single sprite or a single clone. (Versus a global variable that applies to all sprites in the project and all their clones.)

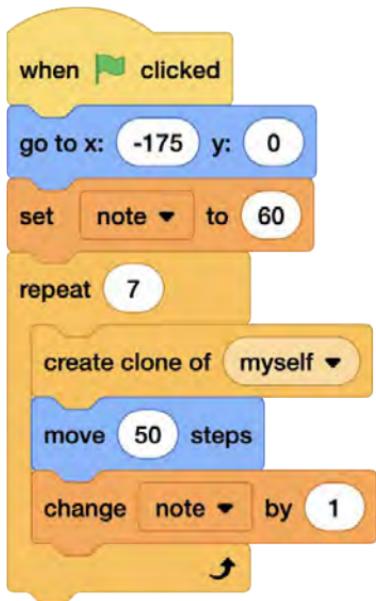
The value stored in each clone's local variable is the value that was present at the moment the clone was created.

Local variables show the sprite name followed by the variable name in the stage monitor.

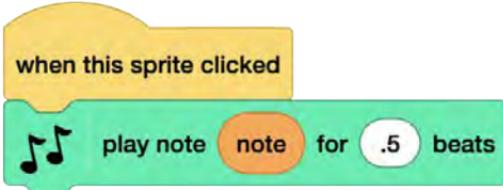
(Example here: [scratch.mit.edu/projects/1181518635](https://scratch.mit.edu/projects/1181518635).)

# Clone Piano: Using Local Variables

scratch.mit.edu

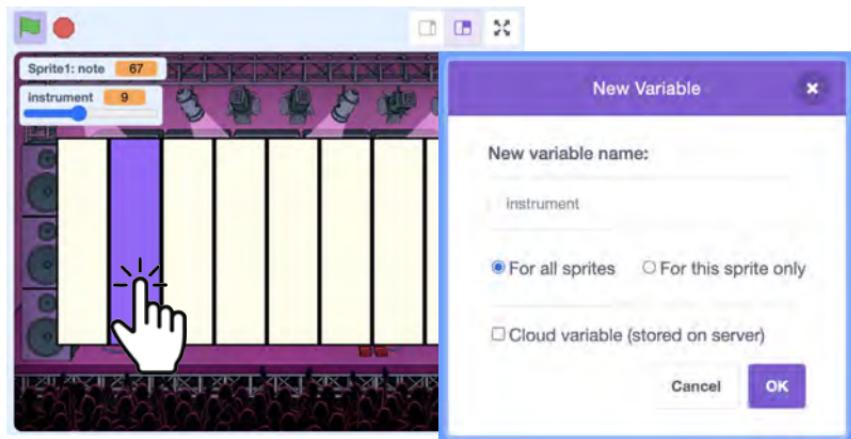


1. Create a piano key sprite by using the Rectangle drawing tool in the Paint Editor.
2. Assemble a script that creates clones of the piano key sprite in a row.
3. Create a local variable (“For this sprite only”) to store the note for each clone and the original sprite.
4. Set the initial note, and then change the note after each clone is created. Use the “note” variable in the “play note” block. Test and debug!



*Optional:* Adjust your program so it changes the color of the piano key or shows a different costume for the piano key as the note is played, so you can hear and see when a piano key is pressed.

# Clone Piano: Using Global Variables



**A global variable (“For all sprites”) applies to all sprites in the project and all their clones.** (Versus a local variable that is individual to a single sprite or a single clone.)

The value stored can be changed, and the variable is updated for all sprites in a project.

Global variables just show the variable name in the stage monitor.

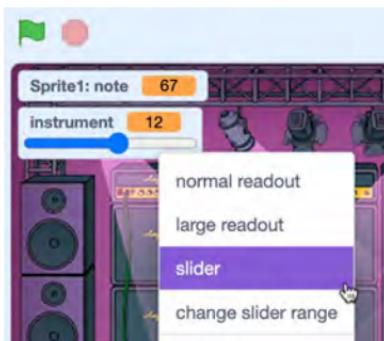
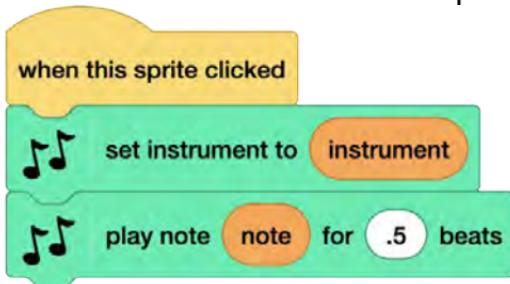
(Example here: [scratch.mit.edu/projects/1181518635](https://scratch.mit.edu/projects/1181518635).)

# Clone Piano: Using Global Variables

scratch.mit.edu

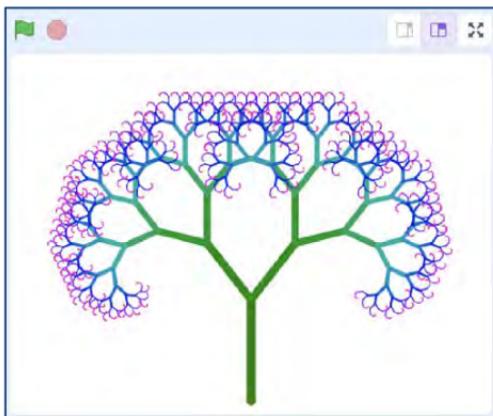


1. Create a global variable (“For all sprites”) to store the instrument for each clone and the original sprite.
2. Set the initial instrument, and then use the “instrument” variable in the “set instrument” block.
3. Right click on the “instrument” stage monitor to change it to a slider. Then, right click again to set the range from 1-21 (the number of instruments available). Now, use the slider to change the instrument globally, for all piano keys. Test and debug!



# Fractal Tree: Clones

## Making Clones



When a clone is created, the scripts for the original sprite also apply to a clone. That means that **clones can be coded to create clones**.

There is a limit to the number of clones a program can create. At this time, each program can create a maximum of 300 clones. The limit is important, because issues could arise with too many clones (for instance, it could cause the program to lag).

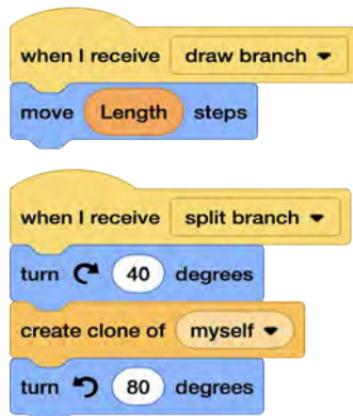
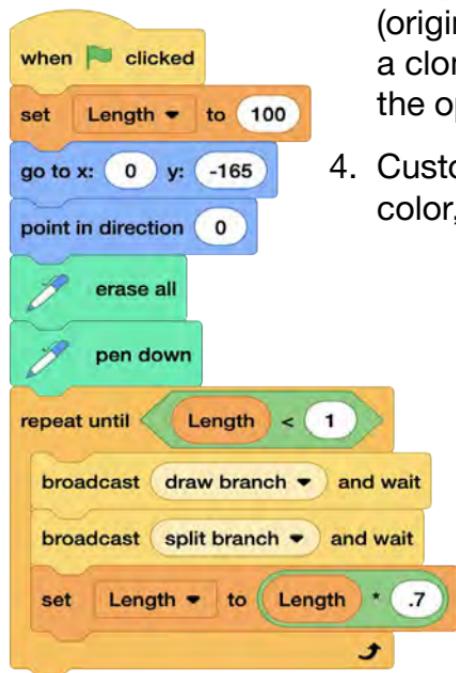
(Example here: [scratch.mit.edu/projects/1145558837](https://scratch.mit.edu/projects/1145558837).)

# Fractal Tree

scratch.mit.edu

The fractal tree uses the pen tool and an army of clones to draw each branch.

1. Set the initial length, position, and direction of the first branch (which will actually be the trunk of the tree). Then, place the pen down. (Pen blocks can be found in the Extensions.)
2. Next, repeatedly have the program draw a branch by moving, split the branch into two, and divide the length, so branches get shorter and more numerous.
3. To split the branch, have the sprite (original and all clones) turn, create a clone of itself, and then turn in the opposite direction.
4. Customize by changing the pen color, brightness, size, etc.



# Clone IDs: Generating and Using



Do you want to have more control over an individual clone's behavior? **Assign each clone an ID as it is created, then use that ID in a conditional statement to set unique code sequences for each clone.**

Generate clone IDs using a local variable ("For this sprite only"). The value stored in each clone's local variable is the value that was present at the moment the clone was created.

(Example here: [scratch.mit.edu/projects/1184917851](https://scratch.mit.edu/projects/1184917851).)

# Clone IDs

scratch.mit.edu

1. Create a local variable (“For this sprite only”) to store the clone ID.
2. Set the clone ID to 1, then change the clone ID by 1 after each clone is created.

*Optional:* Have each clone say its clone ID, so you can easily identify them on the stage. You can always remove this code later, when finalizing the project.

3. Create a conditional statement and use the “equals” Operator block to identify individual clones by their clone ID.

For example, use a series of “if then” blocks or nested “if then else” blocks with different scripts under each one.

