CREATIVE COMPUTING

Harvard Graduate School of Education

UNIT 4
GAMES

YOU ARE HERE

0 1 2 3 **4** 5 6

WHAT'S INCLUDED

UNIT 4 OVERVIEW

THE "BIG IDEA"

Personalization is an important guiding principle in the design of the creative computing experience. By "personalization", we mean both connecting to personal interests and acknowledging that personal interests can vary considerably. There are many ways of knowing and doing – and exploring these multiple ways can help support interest, motivation, and persistence among young learners. In this unit, learners explore some of the advanced concepts and challenging problems associated with game design. An advanced concept or challenging problem can be made more accessible if rooted in activities that are personally meaningful. As an example of the power of context, we turn to a story shared by Mitch Resnick – the director of the Scratch project at MIT.

A few years ago I was at one of our Computer Clubhouse after school centers and I saw a 13-year-old boy working on creating his own game. He was able to control a character, in this case, a fish. He wanted the game to keep track of the score, so you could see how many little fish had been eaten by the big fish, but he didn't know how.

I saw this as an opportunity to introduce the idea of variables. I showed this to him and he immediately saw how he could use this block to keep track of how many fish had been eaten in his game. He took the block and put it in the script right where the big fish eats the little fish. He quickly tried it. Sure enough, every time the big fish ate a little fish, the score goes up by 1.

I think that he really got a deep understanding of variables because he really wanted to make use of it. That's one of our overall goals of Scratch. It's not just about variables, but for all types of concepts. We see that kids get a much deeper understanding of the concepts they learn when they are making use of the concepts in a meaningful and motivating way.

LEARNING OBJECTIVES

Students will:

+ be introduced to the computational concepts of conditionals, operators, and data (variables and lists)
+ become more familiar with the computational practices of experimenting and iterating, testing and debugging, reusing and remixing, and abstracting and modularizing by building and extending a self-directed maze, pong, or scrolling game project
+ identify and understand common game mechanics

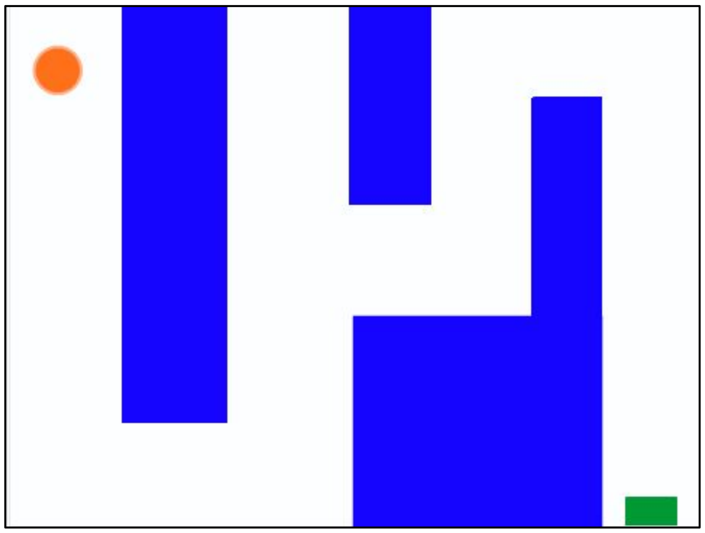
KEY WORDS, CONCEPTS, & PRACTICES

+ abstracting and modularizing
+ conditionals
+ operators
+ data
+ variables and lists
+ sensing
+ feedback fair
+ arcade day
+ puzzle jar
+ brain dump

NOTES

+ Many new concepts are explored in this unit, so we've included added support in the form of example project studios, new programming puzzles for extra practice, and starter game projects that we encourage you to remix and reuse as needed.

CHOOSE YOUR OWN ADVENTURE



In this unit, learners will become game designers and experience creating their own game project. Guided by the activities in this unit, students will be introduced to game mechanics and game development while building understandings of computational concepts (conditionals, operators, data) and computational practices (abstracting and modularizing).

You could get students started on their game projects with the Starter Games activity and then support further development through other activities. From learning common game mechanics such as keeping score and side-scrolling, to the creation of multiplayer games (e.g., Pong), Unit 4 activities offer students multiple opportunities to practice game development.

POSSIBLE PATH

SESSION 1 ↓ **DREAM GAME LIST** — What do all games have in common?

SESSIONS 1 - 5 ↓ **STARTER GAMES** — How can you use Scratch to build an interactive game?

SESSION 2 ↓ **SCORE** — How can you add score to a game using variables?

SESSION 3 ↓ **EXTENSIONS** — What are different ways of extending and increasing difficulty in a game?

SESSION 4 ↓ **INTERACTIONS** — Tackle nine Scratch programming puzzles.

SESSION 5 ↓ **DEBUG IT!** — Help! Can you debug these five Scratch programs?

UNIT 4 ACTIVITY

DREAM GAME LIST

SUGGESTED TIME
15–30 MINUTES

OBJECTIVES
By completing this activity, students will:
+ identify common design elements of games

ACTIVITY DESCRIPTION

- ❑ Divide students into small groups of 2-3 people.
- ❑ In their small groups, ask students to generate a list of games that they enjoy playing. They can compose the list using their design journals or a sheet of paper. We suggest facilitating the brain dump brainstorming activity: give students a short time period (1-2 minutes) to write down as many games as they can. Then, have students narrow down their favorites from the brain dump list.
- ❑ After a few minutes, ask groups about their list of games:
  What do the games have in common?
  What features of their design make them a game?
- ❑ Facilitate a class discussion about what characteristics make up a game and generate a class list of common game mechanics. Next, ask students to imagine their dream game and write a list of design elements for that game.
- ❑ Invite students to share their dream game lists in their small groups or critique groups (see Unit 0 Critique Group activity) to get feedback and suggestions.

RESOURCES

- ❑ paper to write down game design elements
- ❑ things to sketch with (pencils, pens, markers, etc.)

REFLECTION PROMPTS

+ Make a list of your favorite games.
+ What do the games have in common?
+ What features of their design make them a game?
+ Create a list of design elements for your dream game.

REVIEWING STUDENT WORK

+ Do the dream game lists include features of games?
+ What design elements are similar or different from the class group list?
+ What do the lists tell you about the kinds of games and the types of play your students enjoy?

NOTES

+ Invite students to refer back to this dream game list while programming games in other Unit 4 activities.

NOTES TO SELF

- ❑ ______________________________
- ❑ ______________________________
- ❑ ______________________________
- ❑ ______________________________

STARTER GAMES

SUGGESTED TIME
45–60 MINUTES

OBJECTIVES

By completing this activity, students will:

+ develop greater fluency with computational concepts (conditionals, operators, data) and practices (experimenting and iterating, testing and debugging, reusing and remixing, abstracting and modularizing) by working on a self-directed game project

ACTIVITY DESCRIPTION

❑ In this activity, students will create a starter game project that can be revisited and extended during the Score, Extensions, and Interactions activities. Optionally, show the Maze, Pong, and Scrolling example starter projects, and have the Maze, Pong, and Scrolling handouts available to guide students.

❑ Choose one game project to facilitate as a class or let students choose which game they want to create: maze, pong, or scrolling. Give students time to start building their games or let them remix one of the starter projects.

❑ Encourage students to get feedback on their games-in-progress. We suggest the feedback fair activity: half of the students stay in their seats with their projects open while the other half walks around exploring projects, asking questions, and giving feedback, then switch sides. Optionally, have students add their final game projects to the Games studio or a class studio.

❑ Ask students to respond to the reflection prompts in their design journals or in a group discussion.

RESOURCES

❑ Maze handout
❑ Maze example starter project
http://scratch.mit.edu/projects/11414041
❑ Pong handout
❑ Pong example starter project
http://scratch.mit.edu/projects/10128515
❑ Scrolling handout
❑ Scrolling example starter project
http://scratch.mit.edu/projects/22162012
❑ Games studio
http://scratch.mit.edu/studios/487504

REFLECTION PROMPTS

+ What was challenging about designing your game?
+ What are you proud of?

REVIEWING STUDENT WORK

+ Do games include conditionals, operators, and data?

NOTES

+ To celebrate and share final game creations, we recommend hosting an Arcade Day. Final game projects are placed in presentation mode; students walk around and play each other's games.
+ The Scrolling game option introduces cloning. Help students learn more about the cloning blocks with the Cloning handout from Unit 5 Advanced Features.

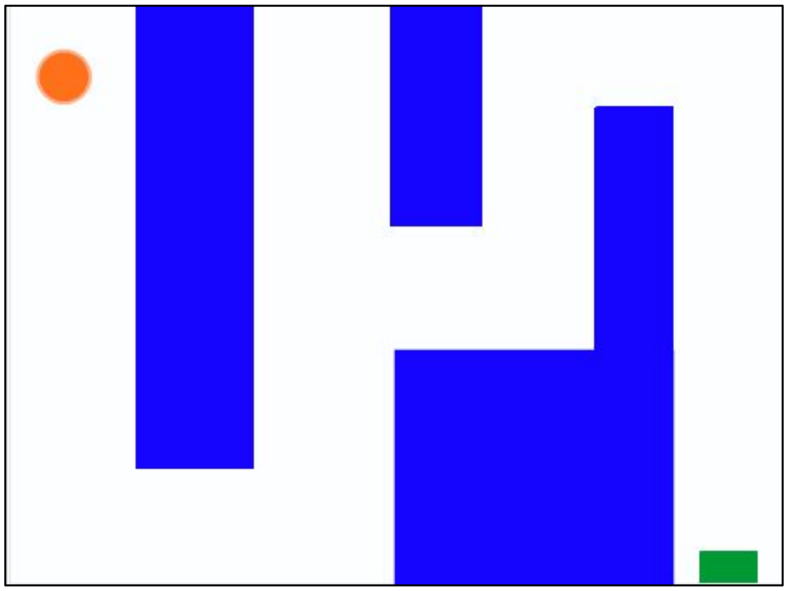
NOTES TO SELF

❑ ______________________________
❑ ______________________________
❑ ______________________________
❑ ______________________________

MAZE

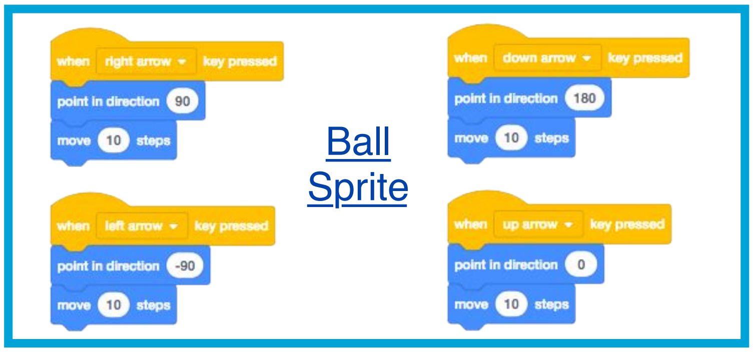
HOW CAN YOU USE SCRATCH TO BUILD AN INTERACTIVE GAME?

In this project, you will create a game. This game includes interactions between sprites, score, and levels. You move a sprite from the start of a maze to the end without touching the walls.



START HERE

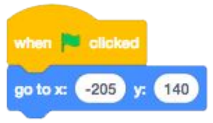
- ❑ Draw a maze-like background and use different colors for the walls and end-of-maze marker.
- ❑ Add a sprite.
- ❑ Make your game interactive!



Ball Sprite

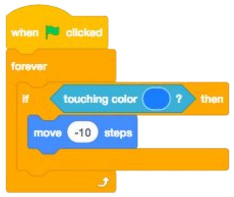
```
when [right arrow] key pressed
point in direction (90)
move (10) steps

when [left arrow] key pressed
point in direction (-90)
move (10) steps

when [down arrow] key pressed
point in direction (180)
move (10) steps

when [up arrow] key pressed
point in direction (0)
move (10) steps
```

These scripts give the player control over sprite movement in the maze.



```
when flag clicked
go to x: (-205) y: (140)
```

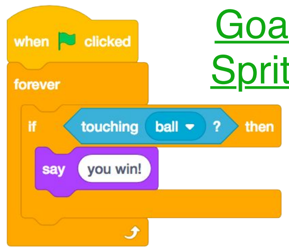
This tells your sprite where to begin and marks the start of the maze.



Ball Sprite

```
when flag clicked
forever
    if <touching color [ ]?> then
        move (-10) steps
```

This will cause your sprite to bounce off the blue walls of the maze.



Goal Sprite

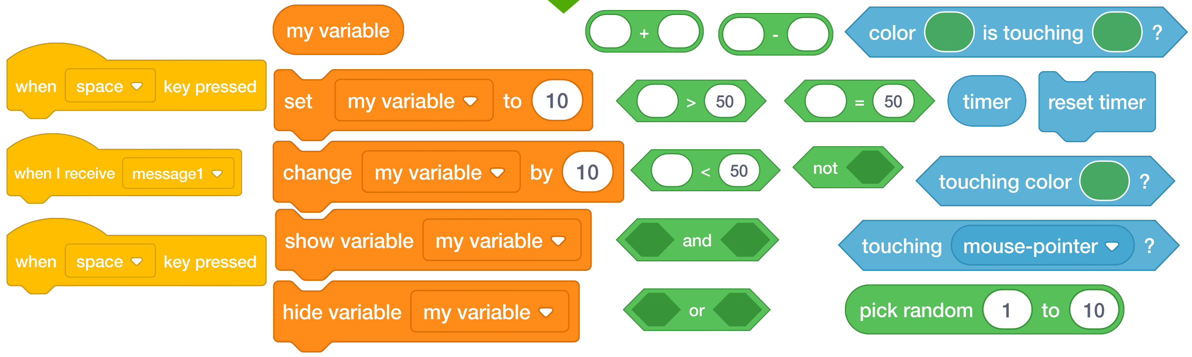
```
when flag clicked
forever
    if <touching [ball]?> then
        say [you win!]
```

This tells the end-of-maze sprite that players win when the ball touches this sprite.

THINGS TO TRY

- ❑ Add multiple levels to your game! This can be done through the use of different backdrops and using broadcast blocks to trigger the next level.
- ❑ Use the make a variable block to keep score!
- ❑ Experiment with timer blocks to add new challenges to your maze!

BLOCKS TO PLAY WITH



```
when [space] key pressed
when I receive [message1]
when [space] key pressed
(my variable)
set [my variable] to (10)
change [my variable] by (10)
show variable [my variable]
hide variable [my variable]
( ) + ( )
( ) - ( )
< ( ) > (50) >
< ( ) < (50) >
< ( ) = (50) >
<not < >>
< < > and < > >
< < > or < > >
<color [ ] is touching [ ]?>
(timer)
reset timer
<touching color [ ]?>
<touching [mouse-pointer]?>
(pick random (1) to (10))
```

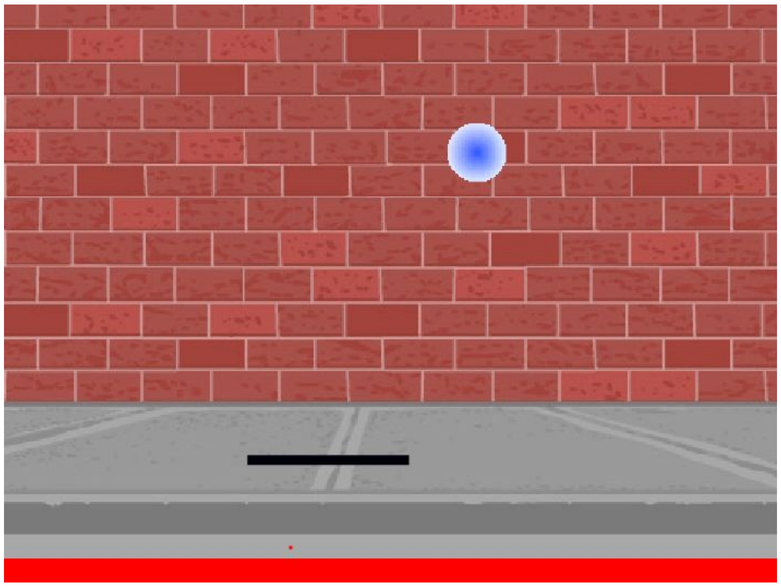
FINISHED?

- + Add your project to the Games Studio: http://scratch.mit.edu/studios/487504
- + Swap games with a partner and walk each other through your creations.

PONG

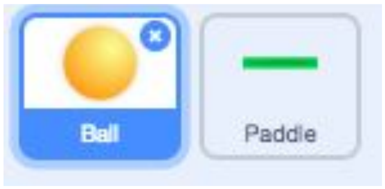
HOW CAN YOU USE SCRATCH TO BUILD AN INTERACTIVE GAME?

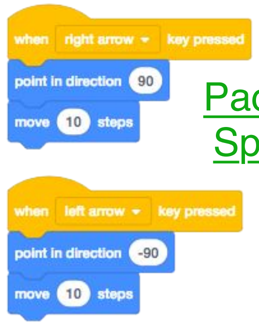
In this project, you will create a game. This game includes interactions between sprites, score, and levels. The game is similar to the classic game of pong, where the goal is to keep the sprite from getting past you.



START HERE

- ❑ Create two sprites: a paddle for the user to control and a ball the user will be playing with.
- ❑ Make your paddle sprite interactive.
- ❑ Bring your game to life!



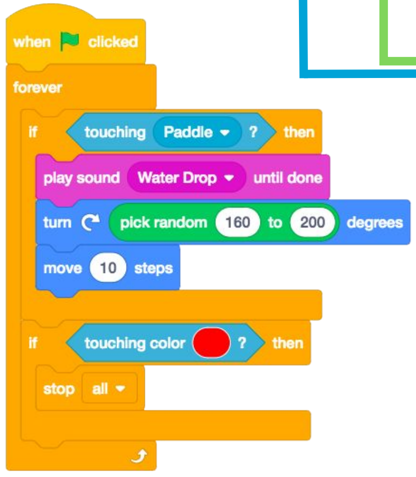


```
when right arrow key pressed
point in direction 90
move 10 steps

when left arrow key pressed
point in direction -90
move 10 steps
```

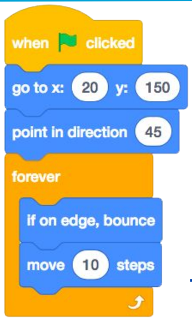
Paddle Sprite

THINGS TO TRY

- ❑ How do you add difficulty to your game? Creating different levels, using a timer, or keeping score are a few examples of things you could do.
- ❑ Experiment with changing the look of your game by editing the backdrops!
- ❑ Explore using different key presses to control your sprites!



```
when 🏁 clicked
forever
    if touching Paddle ? then
        play sound Water Drop until done
        turn ↻ pick random 160 to 200 degrees
        move 10 steps
    if touching color 🟥 ? then
        stop all
```



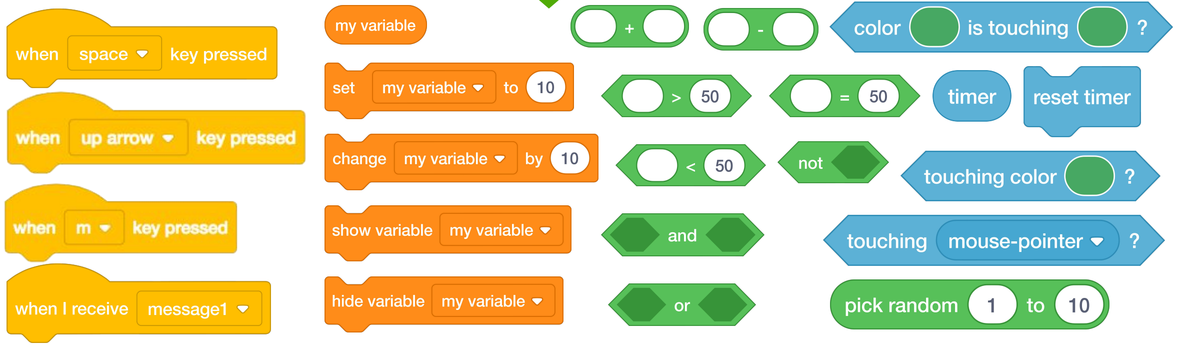
```
when 🏁 clicked
go to x: 20 y: 150
point in direction 45
forever
    if on edge, bounce
    move 10 steps
```

Ball Sprite

Interacts with the walls
Interacts with the paddle

These control the ball - if touching the paddle or a wall, it continues moving. If touching red (meaning the ball moved past the paddle) the game ends.

BLOCKS TO PLAY WITH



```
when space key pressed
when up arrow key pressed
when m key pressed
when I receive message1
my variable
set my variable to 10
change my variable by 10
show variable my variable
hide variable my variable
( ) + ( )
( ) - ( )
( ) > 50
( ) = 50
( ) < 50
not ( )
( ) and ( )
( ) or ( )
color ( ) is touching ( ) ?
timer
reset timer
touching color ( ) ?
touching mouse-pointer ?
pick random 1 to 10
```

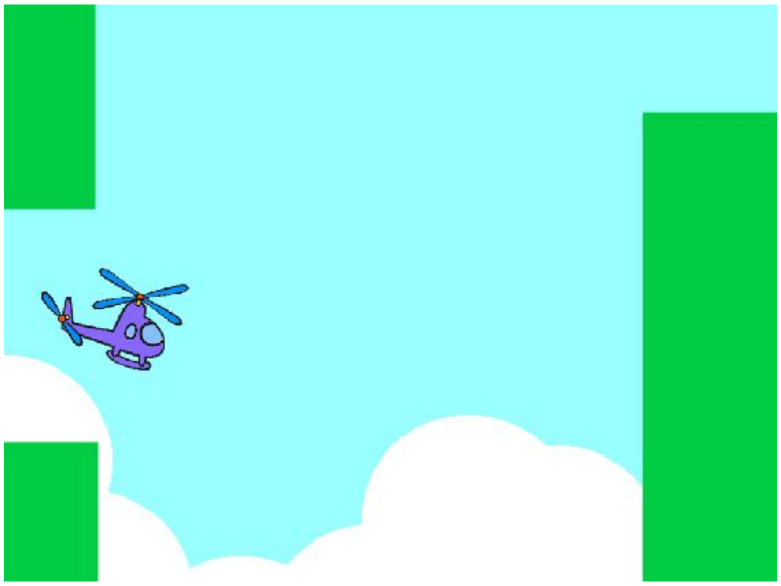
FINISHED?

- + Add your project to the Games Studio: http://scratch.mit.edu/studios/487504
- + Swap games with a partner and walk each other through your creations.

SCROLLING

HOW CAN YOU USE SCRATCH TO BUILD AN INTERACTIVE GAME?

In this project, you will create a game. This game includes interactions between sprites, score, and levels. The game is similar to Flappy Bird, where the goal is to keep an object from falling to the ground or touching certain objects.

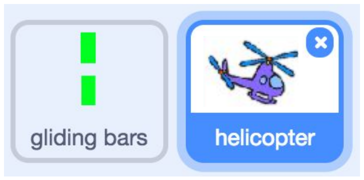


START HERE

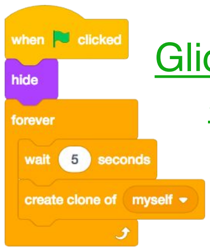
- ❏ Create two sprites: one for the player to control (helicopter) and one to avoid (gliding bars).
- ❏ Make the helicopter interactive.
- ❏ Bring your game to life by adding scripts to make the gliding bars scroll across the stage!

THINGS TO TRY

- ❏ How do you add difficulty to your game? Creating different levels, using a timer, or keeping score are a few examples of things you could do.
- ❏ Experiment with changing the look of your game by editing the backdrops!
- ❏ Explore using different key presses to control your sprites!

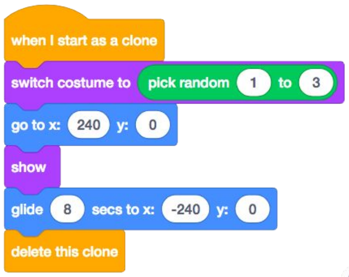


Gliding bars Sprite



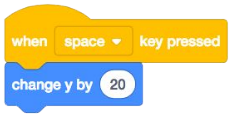
```
when [flag] clicked
hide
forever
    wait (5) seconds
    create clone of [myself ▾]
```

This creates clones, which are used in the script below to make the bars scroll across the screen:



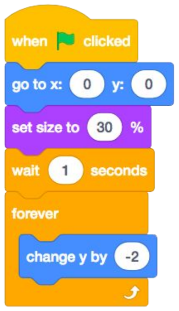
```
when I start as a clone
switch costume to (pick random (1) to (3))
go to x: (240) y: (0)
show
glide (8) secs to x: (-240) y: (0)
delete this clone
```

Helicopter Sprite



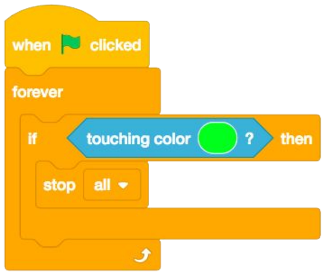
```
when [space ▾] key pressed
change y by (20)
```

Controls sprite movement



```
when [flag] clicked
go to x: (0) y: (0)
set size to (30) %
wait (1) seconds
forever
    change y by (-2)
```

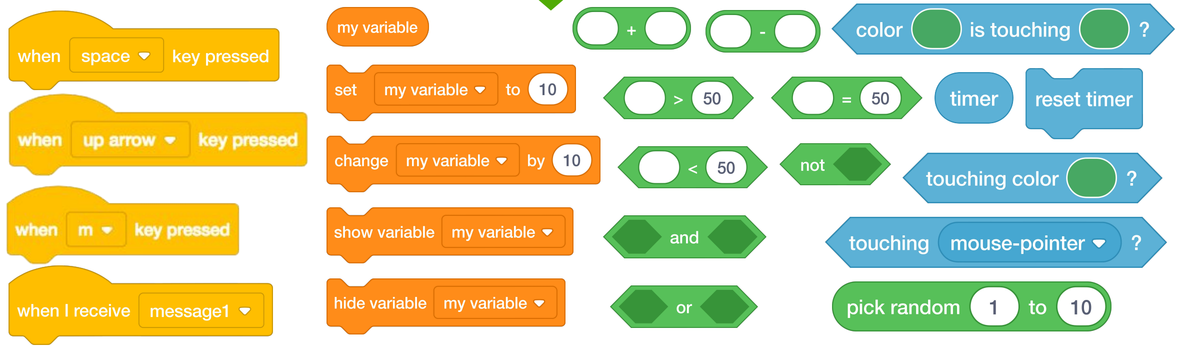
Causes sprite to constantly fall downward



```
when [flag] clicked
forever
    if <touching color [ ] ?> then
        stop [all ▾]
```

Specifies when the game ends

BLOCKS TO PLAY WITH



FINISHED?

- + Add your project to the Games Studio: http://scratch.mit.edu/studios/487504
- + Swap games with a partner and walk each other through your creations.

UNIT 4 ACTIVITY

SCORE

SUGGESTED TIME
30–45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ be able to describe what a variable is and why variables are useful
+ be introduced to the computational concept of data
+ experience remixing and reusing a project or part of a project

ACTIVITY DESCRIPTION

- ❑ Optionally, explore the Fish Chomp starter project as a group and have the Score handout available to guide students.
- ❑ Help students open the Fish Chomp starter project. Give students time to explore variables by remixing the Fish Chomp Starter Project to add score to the game. Optionally, give students time to incorporate score into their previously started maze, pong, or scrolling game projects.
- ❑ Allow students to share their Fish Chomp remixes or game projects with added score. We suggest the Design Demo activity: invite a few students to present their projects to the group and demonstrate how they implemented score using variables. Optionally, have students add their remixes to the Fish Chomp Remix studio or a class studio.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Score handout
- ❑ Score examples studio
  http://scratch.mit.edu/studios/218313
- ❑ Fish Chomp starter project
  http://scratch.mit.edu/projects/10859244
- ❑ Fish Chomp remix studio
  http://scratch.mit.edu/studios/475615

REFLECTION PROMPTS

+ How would you explain variables to someone else?
+ What are variables good for?

REVIEWING STUDENT WORK

+ Can students explain what a variable is and what variables are good for?

NOTES

+ Encourage students to clarify their understanding of variables by exploring code from sample projects in the Score examples studio.
+ Variables are an important mathematical and computational concept. Students are taught about variables in their math and science classes, but many students have a difficult time learning them. Games are one way to make the usefulness of variables more concrete.

NOTES TO SELF

- ❑ ______________________________
- ❑ ______________________________
- ❑ ______________________________
- ❑ ______________________________

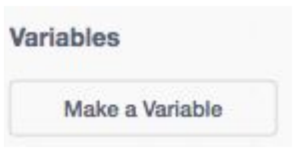
SCORE

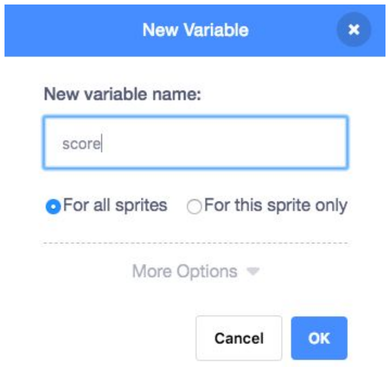
HOW CAN YOU KEEP SCORE IN A SCRATCH PROJECT?

Fish Chomp is a game where players try to catch as many fish as they can by guiding a sprite with the mouse. In this activity, you will remix Fish Chomp by adding a score with variables.

START HERE

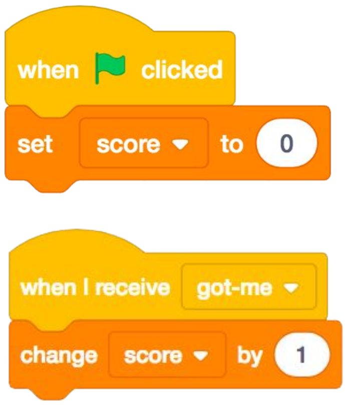
- ❑ Go to the Fish Chomp project page: http://scratch.mit.edu/projects/10859244
- ❑ Click on the Make a Variable button in the Data category to create and name a variable for score.
- ❑ Experiment with your new variable blocks to incorporate score into your project!

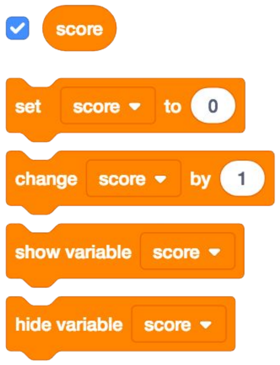



New Variable

New variable name: score

For all sprites [✓] / For this sprite only


```
when green flag clicked
set score to 0

when I receive got-me
change score by 1
```


```
score
set score to 0
change score by 1
show variable score
hide variable score
```

FEELING STUCK?

THAT'S OKAY! TRY THESE THINGS...

- ❑ Not sure how to work with variables? Check out this project for more information: http://scratch.mit.edu/projects/2042755
- ❑ Or take a look at this video: http://youtu.be/uXq379XkhVw
- ❑ Explore and study code in games that use score to learn more about creating variables and incorporating score into a project.

FINISHED?

- + Add your project to the Fish Chomp Remix studio: http://scratch.mit.edu/studios/475615
- + Challenge yourself to do more! How can you use score to add difficulty to your game design?
- + Find a game you are inspired by and remix it!

EXTENSIONS

SUGGESTED TIME
30–45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ become more familiar with the concepts of conditionals, operators, and data by exploring programs that illustrate common game mechanics

ACTIVITY DESCRIPTION

- ❑ Optionally, show example projects from the Extensions studio and have the Extensions handout available to guide students.
- ❑ Give students time to explore the code of programs in the Extensions studio to investigate different ways games can be increased in difficulty or extended. Ask students to select one or more extensions to add to their previously started maze, pong, or scrolling game projects. Give students time to experiment and incorporate the extension(s) into their games.
- ❑ Allow students to share their extended game projects with one another. We suggest facilitating the pair-share or design demo activity to let students share their games and demonstrate what they learned.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Extensions handout
- ❑ Extensions studio
  http://scratch.mit.edu/studios/475619

REFLECTION PROMPTS

+ What are different ways of increasing difficulty in a game?
+ Which extensions did you add to your game project?
+ Describe your process for including the extension(s) in your game?

REVIEWING STUDENT WORK

+ Were students able to incorporate extensions into their original game projects?

NOTES

+ To provide more scaffolding for students needing extra support, we suggest walking through one extension sample program (e.g., levels) as a class and helping students add the extension to their game projects.
+ The backpack tool is one way students can incorporate parts of the extension projects into their starter games. Learn more about backpack at http://bit.ly/scratchbackpack

NOTES TO SELF

- ❑ ________________________
- ❑ ________________________
- ❑ ________________________
- ❑ ________________________

EXTENSIONS

HOW CAN YOU EXTEND AND REIMAGINE GAMES IN SCRATCH?

Get into game design by adding extended features within your Scratch project! Choose at least one (or more!) of the following extensions and add it to your previously started maze, pong, or scrolling games.

START HERE

- ❑ Go to the Extensions studio: http://scratch.mit.edu/studios/475619
- ❑ Choose one (or more) of the extensions to explore.
- ❑ Incorporate your choice into your previously started game projects!

+ **SCORE** http://scratch.mit.edu/projects/1940443
  Demonstrates how to set and change a score. Receive 10 points every time the Scratch cat is clicked.
+ **LEVELS** http://scratch.mit.edu/projects/1940453
  Demonstrates how to change levels. Score increases by 1 every time the space bar is pressed. Level increases by 1 for every 10 points.
+ **TIMER** http://scratch.mit.edu/projects/1940445
  Demonstrates how to use a timer. Use the mouse to navigate the Scratch cat to Gobo.
+ **ENEMIES** http://scratch.mit.edu/projects/1940450
  Demonstrates how to add an enemy. Avoid the tennis ball by using the up and down arrow keys.
+ **REWARDS** http://scratch.mit.edu/projects/1940456
  Demonstrates how to collect items. Use the arrow keys to move the Scratch cat around to collect quest items.
+ **MOUSE** http://scratch.mit.edu/projects/25192659
  Demonstrates how to program the mouse to control game play. Move the mouse to move the paddle.
+ **RESTART** http://scratch.mit.edu/projects/25192935
  Demonstrates how to make a button to restart the game. Click on the RESTART button to restart.
+ **MENU** http://scratch.mit.edu/projects/25192991
  Demonstrates how to display a menu screen at the beginning of the game. Click START or DIRECTIONS on the menu screen.
+ **MULTIPLAYER** http://scratch.mit.edu/projects/25192711
  Demonstrates how to add another player to the game. Player 1 uses the arrow keys to navigate Pico through the maze, and player 2 uses the W, A, S, D keys to navigate Nano through the maze.

THINGS TO TRY

+ The backpack can be an extremely useful tool while programming in Scratch. It can store everything from lines of code, to music files, to sprites, and more. Try using it to incorporate extensions into your game projects.
+ Alternatively, sketching out ideas and bits of code in your design journal is another great method for planning how to incorporate your extensions.

FINISHED?

+ Add another extension to your maze, pong, or scrolling game.
+ Challenge yourself to do more! Continue going through each of the extensions and add them to your games.
+ Help a neighbor!
+ Share your project with a neighbor and give each other feedback on your games.