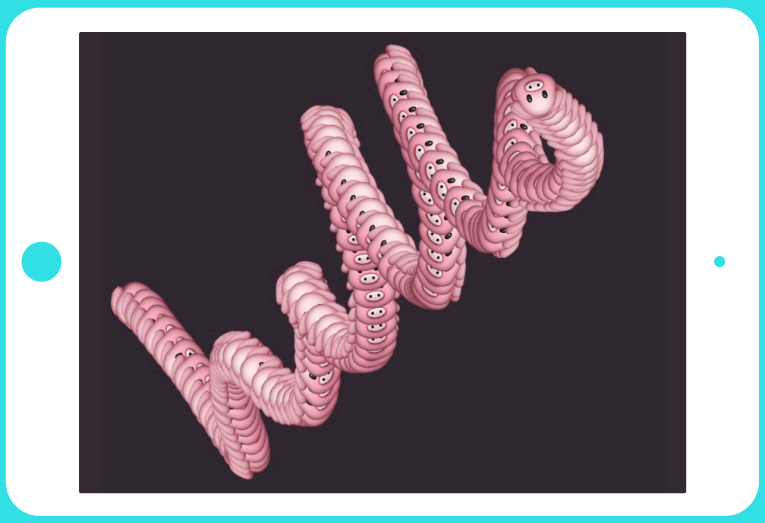
HOPSCOTCH HOUR OF CODE



Teacher notes for student-led tutorial

TIME

45-60 minutes (+15 minutes of optional, free code time)

BIG IDEA

Coding is creative. You can express yourself and your ideas through code.

SKILL FOCUS

– Debugging
– Make sense of problems and persevere in solving (CCSS.MATH.PRACTICE.MP1)
– Look for and make use of structure (CCSS.MATH.PRACTICE.MP7)
– Designing solutions (NGSS Practice 6)

KEY VOCABULARY

Sequence: The order in which instructions are given to the computer
Event: When something happens
Rule: Instructions that tell your computer what to do (the command) and when to do it (the event)
Loop: Code that repeats
Value: A number that can change

TRANSFER GOALS

1. Students will understand that coding requires giving a computer explicit directions
2. Students will become familiar with creating and editing rules
3. Students will practice testing their programs to find bugs.
4. Students will abstract a problem to design a solution.
5. Students will develop confidence and persistence.

MATERIALS

– 1 iPad or iPhone per student, or 1 device per 2 students, for pair programming
– Video tutorial available in Hopscotch and on http://hop.sc/hoc_emojidraw

Hi!

We're *really* excited that, this Hour of Code, you're programming with your students—both for them and for you. Kids have remarkable imaginations, and creating computer programs is an amazing way for them to express themselves. We've seen kids create astonishing things using our simple but powerful tool. We know you'll see the same when using Hopscotch, and hope you share what your students create.

Anyone, regardless of their experience in programming, can teach this Hour of Code lesson. Just as Hopscotch was built on the principle that anyone can become a great programmer, this lesson is designed on the premise that anyone can teach basic programming, including you!

In this lesson, students will build a drawing app that traces the movement of their finger on the screen and creates a trail of spinning emojis. It's simple but fun, and very quickly allows them to experience the creativity inherent in coding.

You can teach this Hour of Code in several ways:

1. Students independently complete their games, following along with the video tutorial in the app. The tutorial is designed to be used without any outside help (though we encourage kids to pause it as they're coding). The tutorial is available at http://hop.sc/hoc_emojidraw.
2. Students indepently complete their games but you ask them to pause their own work for discussion and group work. We offer discussion ideas, as well as differentiation techniques and reflection questions, in the following pages.
3. You project the video to the class and use it as a supplement to your instruction. You lead discussion and group work, and adjust the directions for the project based on the following.

After a discussion of what needs to be built and, if desired, how it might be coded, students can start coding. Depending on how many iPads you have, you can have students work independently or in pairs. At Hopscotch, we do a lot of pair programming (two programmers share one computer) because it helps us write smarter, less-buggy code. We recommend trying it! All students should get into the habit of testing their code frequently by running (playing) it. It is much easier to find and solve mistakes when you're constantly testing.

Have fun and we can't wait to see what your students build. Share their projects on social media and tag us either with #madeonhopscotch or @hopscotch on Twitter and @gethopscotch on Instagram :)

Yours,
Jocelyn Leavitt
Co-founder and CEO, Hopscotch

0. Discussion: Creative Coding
In this activity, students will create a drawing app that they and others can use to create art. They will be able to choose the medium—the "ink" used to draw. In this exercise, they will experience coding's power as a creative medium of expression. No two apps will be the same.

To start the activity, ask students to discuss examples of how technology can be a form of creativity. Possible starting places include software for creativity, like Word, Photoshop, or Garageband, or being able to change the way an app looks through code. Challenge students to discuss why this powerful.

Second, help orient students to think of themselves as coders. What does this mean? What kind of responsibilities and powers might they have? One is that they will tell the computer what to do and when to do it. In Hopscotch, we use **events** to represent this idea of "when to do something." To create programs, we combine an event (or a trigger) with the resuling action ("what to do") to create a **rule**.

They will have to work hard to make sure their rules function the way they intend, and work together to debug, or find and fix problems in code, when they don't.

Students will also explore the responsibility that comes with building a creative tool. They should talk about what it means to create something that other people will use, and how this relates to digital citizenship.

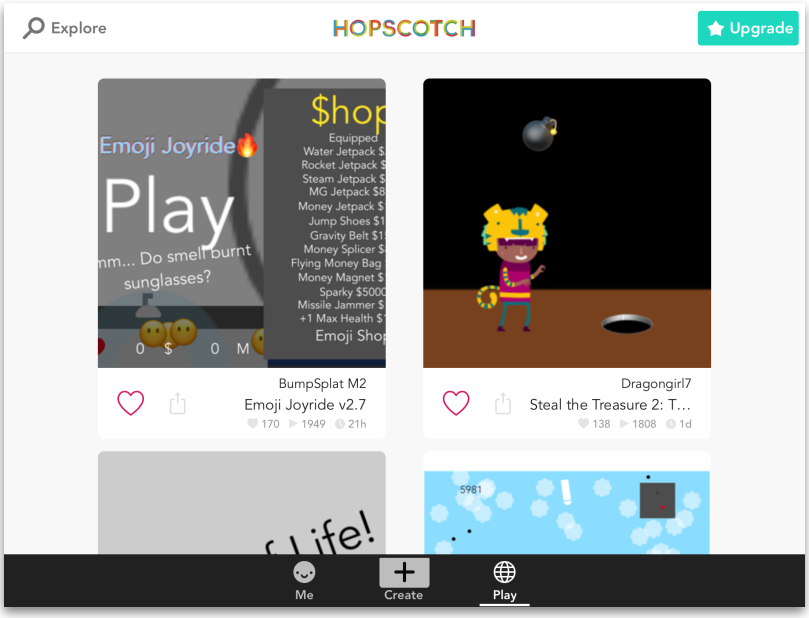
1. Using Hopscotch (5 minutes)

First, get your students acquainted with Hopscotch.

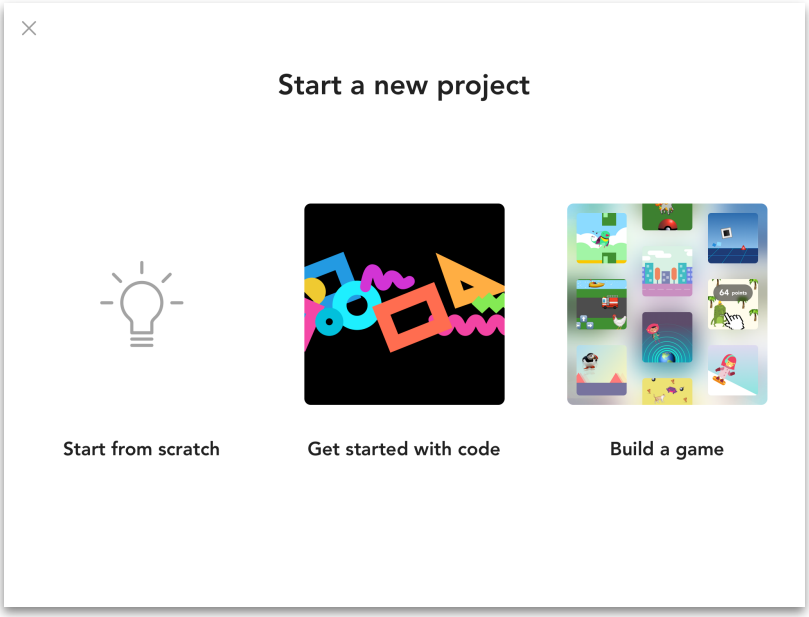
1.1 Finding the Hopscotch app on your iPad

1.2 Signing into your account (students may need to create accounts)

1.3 Making a new project: Tap on the highlighted + on the bottom of the screen



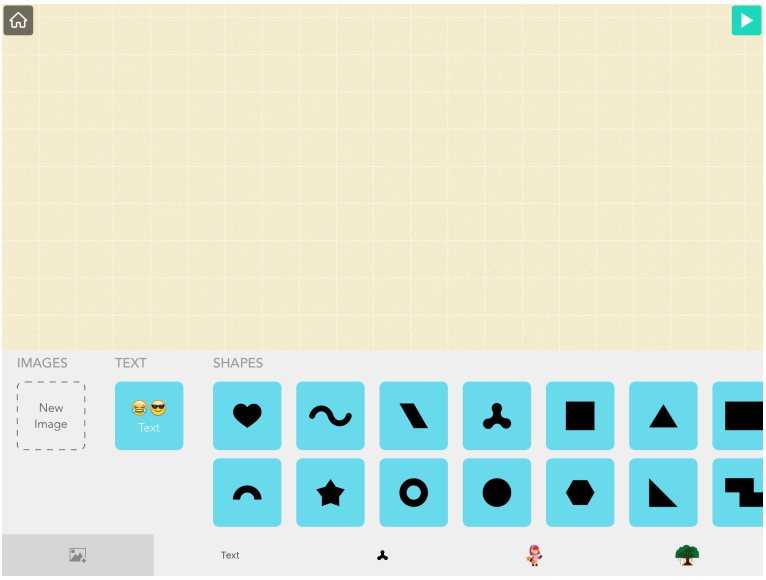
1.4 Choose Blank Project



2. Choose your object

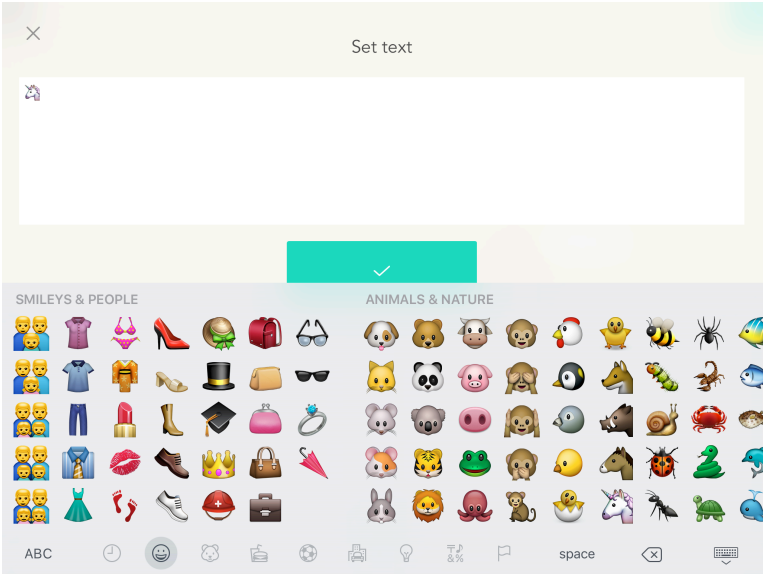
Since this app will create a drawing using emoji, students should choose text as their object. They then have the freedom to customize the "ink" of their drawing app by choosing whatever combination of emoji they like. This can always be changed later, so encourage students to think of this step as a starting place.

2.1 Press the "+" button at the bottom of the stage and choose a text object.



Make sure the emoji keyboard is enabled, which you can do in your iPad's settings.

2.2 Set the text to whatever emoji you like (I picked the unicorn here)



After setting your text to an emoji, you can rename the object.

By default, Hopscotch names text objects "Text", "Text 2", etc. Clearly named objects make it easier for you and others to read your code, however, and students should rename each text object according to the role it will serve in the project. Here, since we are using it to draw, let's rename the text object "Drawer."

3. Events and rules
Take this opportunity to revist the premise of the game with your students: they are creating an app that will leave a trail of emojis wherever the player touches the iPad. If they play their game at this stage, nothing will happen. This is because they haven't told the computer what to do!

This is the perfect time to discuss the bones behind Hopscotch code: events and rules.

An **event** is a trigger that the computer recognizes and causes it to do some action. In Hopscotch, all events start with the word "When" and are the first thing you choose when you write a rule. Think of it as completing a "WHEN….., THEN….." sentence.

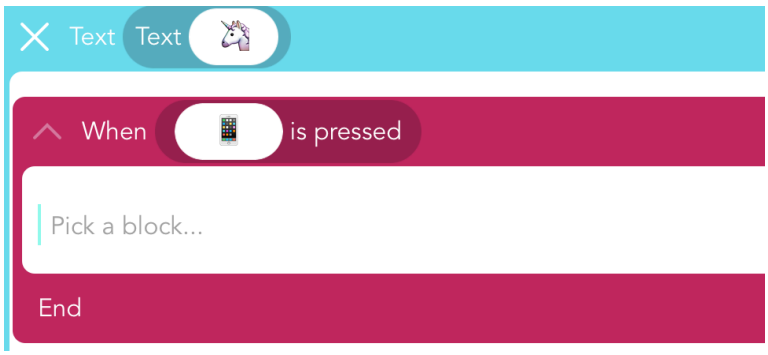
Events are deeply important for computer engineers because they tell the computer when it should do something. When you touch the phone icon on your home screen, then your phone brings up the interface to make calls. When an Angry Bird hits a block, then the block falls down.

Discuss some events (triggers) that happen in the classroom. Identify the trigger and resulting action: When I raise my hand (trigger), then stop talking (action), when the bell rings (trigger), then put down your pencil and turn in your test (action).

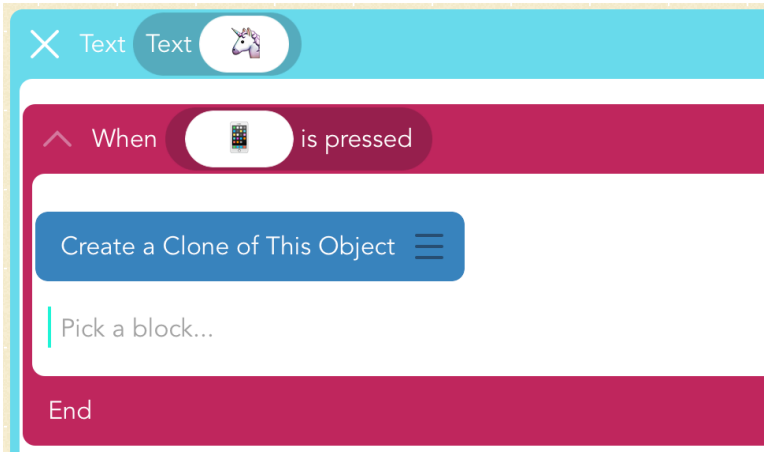
Together, a trigger and a result form a rule in Hopscotch — an event plus the code that says "what the computer should do" and "when it should do it." Ask students to brainstorm the rules of a drawing app in pairs.

To make Spiral Draw, we need to create two rules. For the first, we want the object to clone itself when the iPad is pressed (this will create the "ink" in the drawing).

3.1 Tap on "Add Code" and then add the "When iPad is pressed" event.



3.2 Add a "Create a clone" block.



4. Program clones

In this step, we will tell the clones what to do after they are created. They should appear wherever the player has touched the iPad to make it appear like the object is leaving a trail of "ink" (clones) behind it. Ask your students to brainstorm ways that the game can track where to leave the "ink".

One solution is to track wherever the player is touching the iPad. When you move an object around on the stage in editing mode, you'll see two numbers following it. These numbers are the (x,y) coordinates and tell you where the object is located on the screen. The first number, or x-position, tells us how far to the left (a number closer to 0) or the right of the screen (a bigger number) the object is. The second number, or y-position, tells us how close to the bottom (a number closer to 0) or top of the screen (a bigger number) the object is.

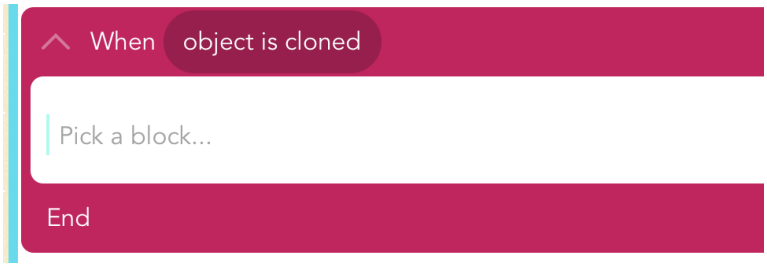
So to keep track of where the "ink" should appear, we can use a value to track the last place the player has touched the iPad. Values, also popularly known as variables, hold pieces of information. Values can be set, changed, or checked. They can be used inside events and other blocks, and can stand in any place you can use a number.

What makes values so powerful is that you can actually change them programmatically. For example, think of your score in a game like Angry Birds or the number of unread emails in your inbox. Both of these numbers are actually values (variables). As you score more points in Angry Birds your score goes up; the value changes. As you read your email the number of unread emails you have goes down. Values are used to represent some kind of information.

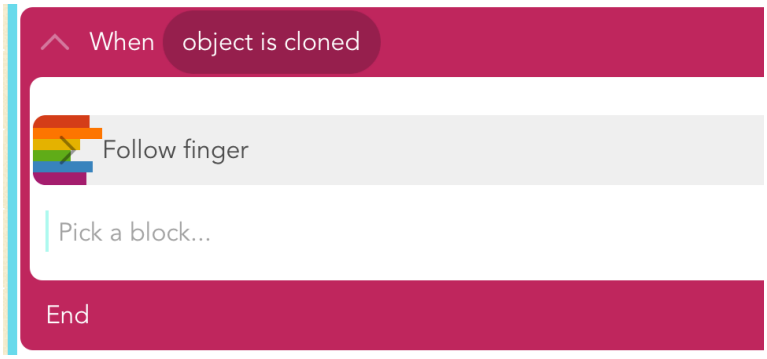
Ask your students to discuss examples of values—or numbers that change—in their lives.

In this game, we're going to set the location of clones to value that represents the last place the iPad was touched.

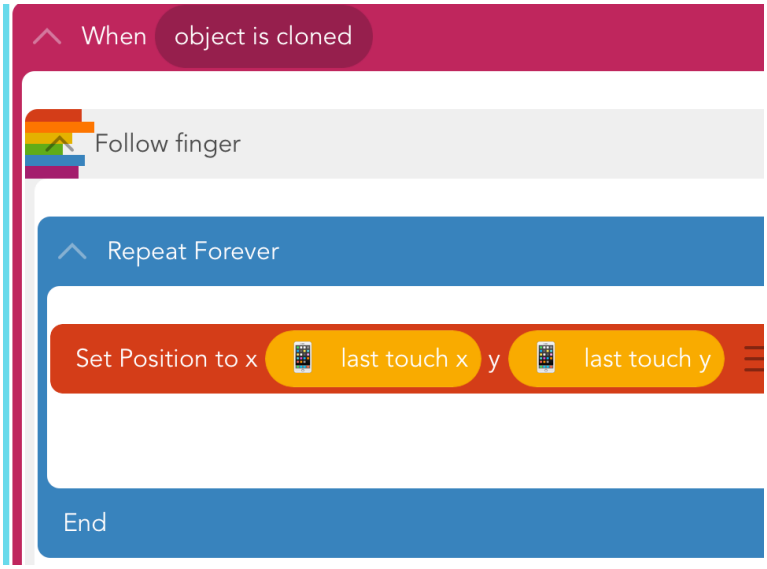
4.1 Add a new "When object is cloned" When



4.2 Add a "Follow finger" ability to the "When object is cloned" rule



4.3 Open the "Follow finger" ability by tapping the arrow on the left



At this stage, ask the class to the play the game and then discuss how it is working in pairs or small groups. They should notice that something is not exactly right. In their small groups, challenge them to articulate the problem and then, as a class, discuss possible ways to solve it.

5. Debugging and loops

Currently, when the player presses the screen, the object is told to clone itself. A millisecond later, if the iPad is still being pressed, both objects are told to clone themselves and go to the same location. This happens over and over until we end up with a huge number of objects cloning themselves on top of each other. Ah!

Instead, we want to create one clone every time we press the screen. To do that, we can use a loop to effectively "stop" the code. In computer science, a "loop" is code that repeats, and loops are one of the basic building blocks of all software. Ask your students to identify and discuss example of loops in their lives (e.g. a stop light).

In Hopscotch, a "Repeat forever" loop will repeat whatever code is inside of it, from start to finsh, until the program ends.

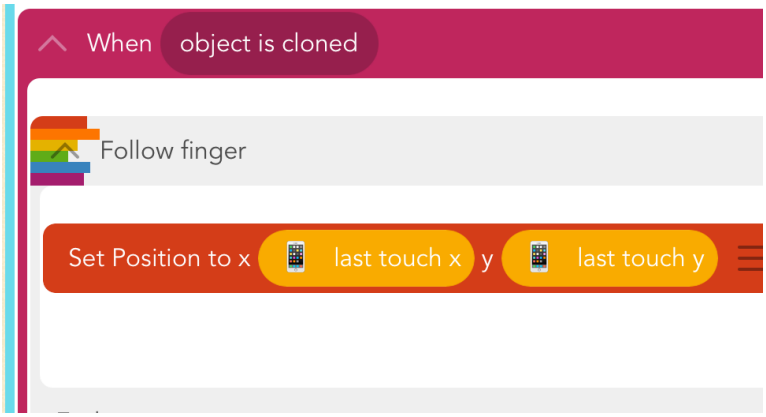
In this step, students will remove the loop from the "Follow finger" ability so that the clones do not constantly follow the player's finger but rather stay in one place. Then, they will add a repeat forever loop underneath the "Create a clone of this character" block.

By dropping a "Turn by" block in this loop, they can set newly made clones on an endless task of spinning, which will prevent them from cloning themselves indefinitely each time the iPad is pressed.

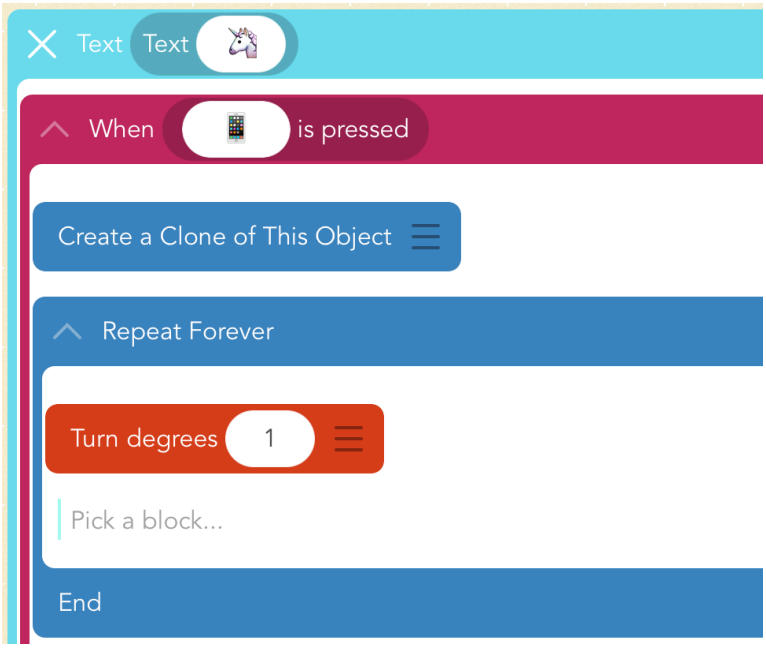
(If these steps seem complicated, remember you can always follow along with the instructional video linked on page 02.)

When discussing these steps with your students, return to the debugging framework to indicate how each line of code they add represents a solution to a problem they had created.

5.1 Drag the "Set Position" block above the Repeat forever but below the top of the Follow Finger block. Then delete the repeat forever block.



5.2 Add a "Repeat Forever" block below the "Create a clone of this character" in the "When iPad is pressed" rule. Add a "Turn by" block set to 1 degree inside the loop.



5.3 Play and publish your project!

DIFFERENTIATION

(15 minutes, optional)

- Add a background
- Change the emojis
-
-

REFLECTION

(5 minutes, optional)

- Why is coding creative? How can you use code to express yousrelf?
- What are values? Why are they useful?
- Is drawing with a computer easier or harder than drawing with pencil and paper? Why? If it is harder, why do we still do it?

GLOSSARY FOR YOUNGER STUDENTS

Ability: Code that can be reused

Algorithm: A recipe for a program

Coding: Telling computers what to do

Concurrence: Two things happening at the same time

Conditional: Statements of the form "IF (something is true) THEN (do an action)".

Debugging: Finding mistakes in your code and fixing them

Event: When something happens

Iteration: Having ideas and making mistakes, over and over

Logic: The process of making decisions

Loop: Code that repeats

Operator: A mathematical symbol that makes an equation

Program: A set of instructions a computer can understand

Programmer: A person who writes programs

Programming Language: A set of rules or blocks that can be used to write any program

Random: When there's no pattern

Range: The highest and lowest number random can choose between

Rule: Instructions that tell your computer what to do (the command) and when to do it (the event)

Sequence: The order in which instructions are given to the computer

Object: A character or text with its own rules

Value/Variable: A holder for a number

GLOSSARY FOR OLDER STUDENTS

Ability/Function/Procedure/Subroutine: A saved set of blocks. What we call abilities in Hopscotch are known as functions or subroutines in other programming languages. Easily replicable routines are a key concept in computer programming, and allow you to scale your code and create complex programs.

Algorithm: Algorithms are at the heart of computer science; they are the recipes that computers follow to solve problems.

Bug: An error that a programmer has made in their code

Coding: Writing the rules of behavior for a computer to follow automatically; programming

Concurrency: Two or more things happening at the same time, or triggered by the same event

Conditional: Statements of the form "IF (something is true) THEN (do an action)"

Debugging: Finding mistakes in your code (bugs) and fixing them

Event: A trigger that the computer recognizes and causes it to do some action. In Hopscotch, events include "When the iPad is tapped" or "When the play button is tapped"

Iteration: the repetition of a process

Logic: the science of the formal processes of thinking and reasoning

Loop: a repeating set of instructions

Operator: a mathematical symbol that produces a value

Program: a set of instructions a computer can understand

Programmer: a person who writes programs

Programming Language: a set of words, rules, blocks or instructions that can be used to write a program.

Random: Any number or item among a set. The lack of a pattern among items in a set.

Range: The highest and lowest number random can choose between

Rule: Rules tell your object what to do and when to do it. When you make an ability and pair it with an event, you create a rule.

Sequence: An ordered list of things (instructions, blocks, numbers, etc) which can be triggered by an event or repeated

Object: A character or text with its own rules on screen

Value: A holder for a number. Also known as a variable