GEORGE LUCAS EDUCATIONAL FOUNDATION

CODING IN THE CLASSROOM

Easy Projects to Get Started with Scratch: Drawing 2D Shapes

By *Dylan Ryder*

August 11, 2014

I had my first computer programming experience in ninth grade. I had just moved, and my new guidance counselor persuaded me to take a computer programming class because there were no other open electives. I was dubious. It appeared that coding was either not fun, not cool, not easy, or perhaps all three.

We used a language called BASIC to learn about managing data sets, sorting lists, and computing algebraic expressions. I wasn't exactly frustrated, but I certainly wasn't hooked by line after line of green text on the solemn black screen. I found more enjoyment hypnotizing myself with the animated screensavers and playing the onboard video games rather than doing what I thought was the "real programming."

So I didn't code again until a college intro course in the language C++. It was another, even more challenging experience for me. The computing was more powerful, but the script was even more cryptic! I often felt adrift in a sea of code words and keyboard symbols -- much like a natural language such as French or Spanish, programming languages come with their own vocabulary, grammar, and syntax. Adding foreign language lessons to the extra dose of math wasn’t going to help me love programming.

This high entry barrier to the world of computer programming is not uncommon for most kids. Programming experiences typically don't appear until late in high school if not college, and perhaps the whole thing looks like a mess of semicolons and brackets to them. No wonder the U.S. is experiencing a dearth of young programming talent.

Things have changed, however. A wave of new "gateway" programming languages have become available to teachers and students over the last few years, lowering the barriers for learning how to code while also broadening the view of what it means to program a computer. Building off the pioneering ***Logo*** **(http://el.media.mit.edu/logo-foundation/logo/turtle.html)** language, the MIT Media Lab's ***Lifelong Kindergarten Group*** **(http://llk.media.mit.edu/)** developed ***Scratch*** **(http://scratch.mit.edu/)**, a programming language with graphic drag-and-drop code blocks that snap together like LEGO bricks. Code blocks are categorized by color and shape according to their function, and designed to prevent users from making dreaded syntax errors.

Scratch has quickly become the most popular programming language for kids today, partly because it was designed for learning, and partly because it's built around sharing with a huge online community. Scratchers are challenged to **imagine, program, and share** their coded creations with over ***3.5 million other users*** **(http://scratch.mit.edu/statistics/)** who are eager to comment, compliment, and help each other. New stories, animations, and games are added every day, because Scratchers see programming as an interesting, playful way to express themselves.

Scratch also helps distill programming into its most pleasant and non-threatening core: just giving instructions to a computer -- that's all programming really is. My students start with Scratch in second grade, and they don't look back. By the time they're ready to graduate to more traditional languages like Python or Java, they aren't scared off by encountering hundreds of lines of text.

It was only after I was introduced to Scratch that my *love* of computer programming was finally born. It's fun and easy to use. The following tutorial is a brief introduction to coding with Scratch and can help young learners explore core math concepts like shape attributes and angles. We don't have to subtract anything from our curriculum to fit in computer programming and computational thinking. This is an absolute beginner's activity, so don't be shy! (If video tutorials are more your style you can ***watch the tutorial*** **(https://www.youtube.com/watch?v=XhIFLPUGfOY)** instead.)

Here's How

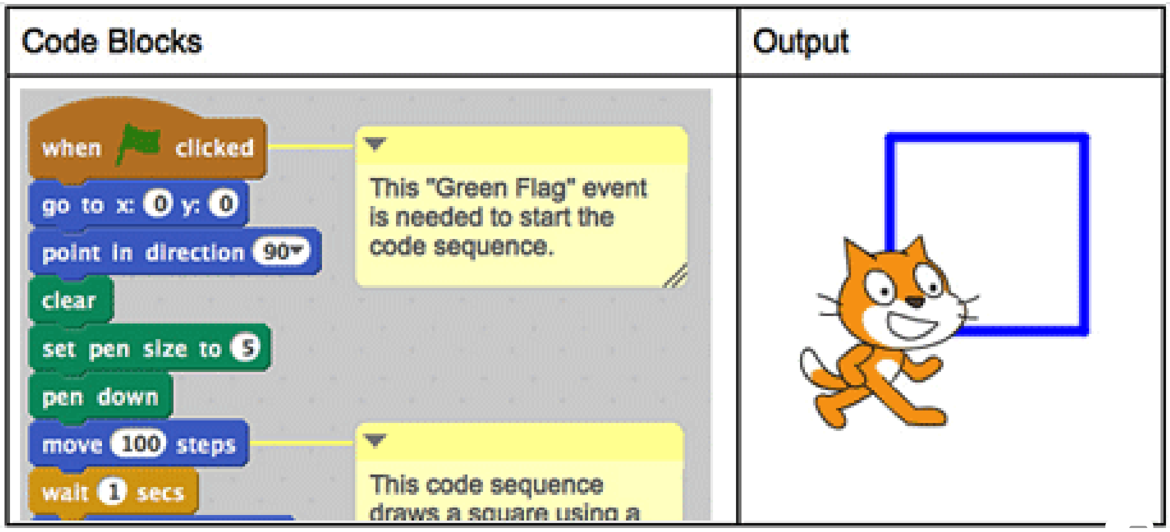
Getting to Know Scratch

- Sign up for an account by clicking on "Join Scratch" at the top right of your browser window. This isn't required to actually use Scratch, but it is necessary if you want to save your work.
- Click the "Create" button near the top left of your browser window to start a new project.
- Let's get to know the most important areas of the Scratch interface so we can make our own project:
- The area in the top left quadrant of the window is called the **Stage**, and it's where you can see the audio and visual output when you run a program. Make note of the **Green Flag** icon at the top right corner of the Stage -- it's used to start your programs.
- The area below the Stage to the bottom left is called the **Sprite List** and is where the characters and objects in your project are listed for selection and editing. Scratch refers to these objects as Sprites.
- The central area that divides the interface is called the **Palette**, and it holds all the code blocks that you can use to control your Sprites. Notice that there are ten color-coded sections. Clicking on each one reveals a new set of code blocks for programming your Sprites.
- Finally, the large area that's the whole right side of window is called the **Scripts Area**. It's used for building your programs. Drag code blocks from the Palette into the Scripts Area and stack them in sequence to make your Sprites do things.

1st Challenge: Draw a square.

For our first task, we'll teach our cat Sprite to draw a square on the Stage. Add the following code blocks on the left to the Scripts Area, and then click the Green Flag to run your program and see the visual output on the right.

(/pdfs/blogs/edutopia-ryder-scratch-square.png)



Code Blocks | Output

```
when [flag] clicked
go to x: 0 y: 0
point in direction 90
clear
set pen size to 5
pen down
move 100 steps
wait 1 secs
```

This "Green Flag" event is needed to start the code sequence.

This code sequence draws a square using a

(/pdfs/blogs/edutopia-ryder-scratch-square.png)

Scratch is a project of the Lifelong Kindergarten Group at the MIT Media Lab.

(/pdfs/blogs/edutopia-ryder-scratch-square.png)

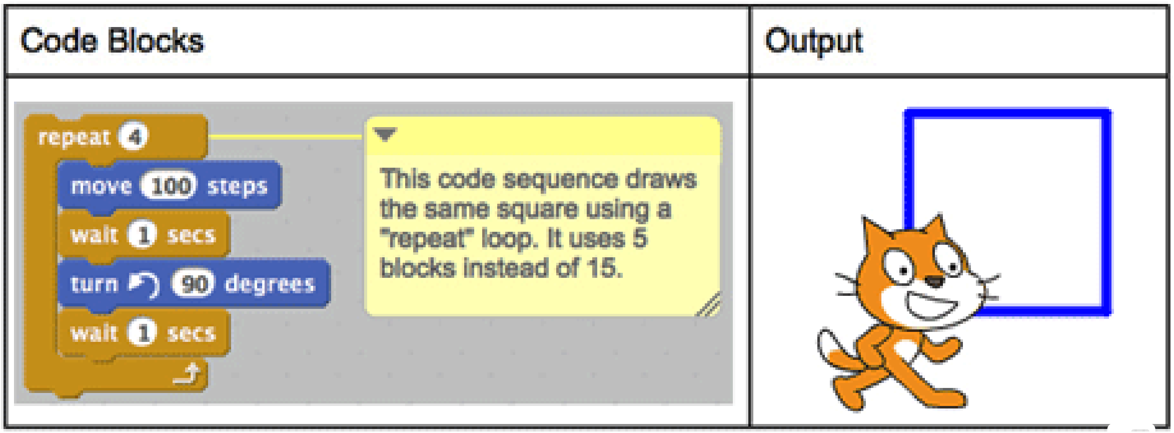
Did your Sprite draw a square? Guess what -- you are now a programmer! It can be as simple as that. But wait, can we make it simpler? Let's learn a clever way to get the same output with less effort.

2nd Challenge: Draw the same square using only ⅓ as many code blocks.

When you're working with students, ask them if they notice a pattern of "Move" and "Turn" blocks. We can teach the computer a simplified sequence with a "Repeat" loop.

- Click on the top-most Move block and drag it away from the rest of the sequence.
- Attach the code blocks below after the "Pen Down" block and click the Green Flag to run your program.

(/pdfs/blogs/edutopia-ryder-scratch-repeatloop.png)



Code Blocks	Output

```
repeat 4
    move 100 steps
    wait 1 secs
    turn ↺ 90 degrees
    wait 1 secs
```

This code sequence draws the same square using a "repeat" loop. It uses 5 blocks instead of 15.

(/pdfs/blogs/edutopia-ryder-scratch-repeatloop.png)

Scratch is a project of the Lifelong Kindergarten Group at the MIT Media Lab.

(Click to enlarge.)

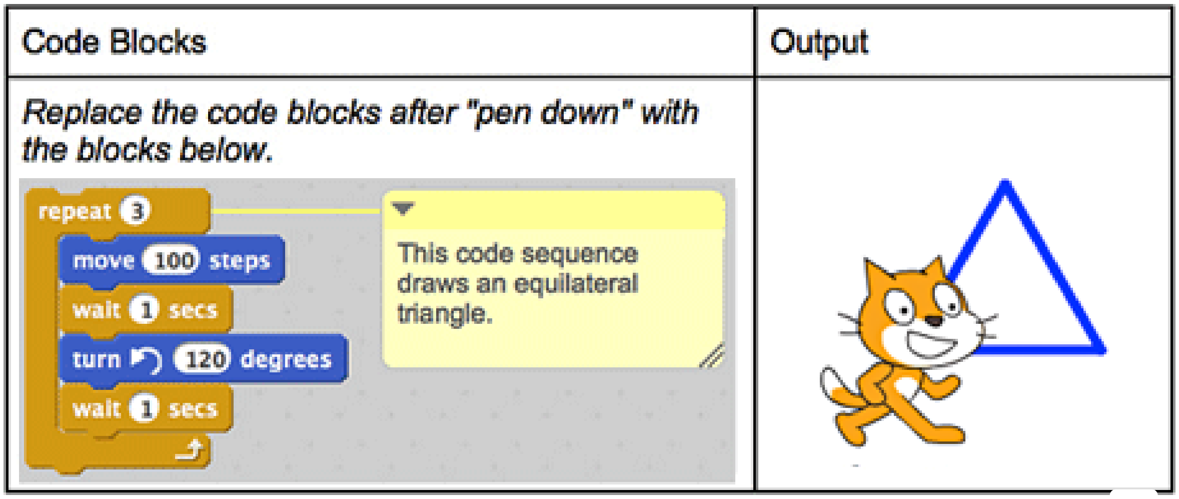
(/pdfs/blogs/edutopia-ryder-scratch-repeatloop.png)

Now we've optimized this square drawing program with a loop. That's something that computers do really well -- they can execute repetitive tasks without getting tired or bored. Let's keep going and teach the Sprite to draw a new shape.

3rd Challenge: Draw a triangle.

- If you were working with your students, you might ask them for the properties of a triangle. Hopefully they'd tell you that a triangle has three sides and three angles. We can draw an equilateral triangle by editing our square program.
- Change the "Repeat" loop to repeat three times.
- It's OK to leave the "Move" block as is (but it *is* fun to change the value).
- Change the "Turn" block's value to 120 and click the Green Flag to run your program.

(/pdfs/blogs/edutopia-ryder-scratch-triangle.png)



Code Blocks	Output
Replace the code blocks after "pen down" with the blocks below. repeat 3 move 100 steps wait 1 secs turn ↺ 120 degrees wait 1 secs This code sequence draws an equilateral triangle.	

(/pdfs/blogs/edutopia-ryder-scratch-triangle.png)

Scratch is a project of the Lifelong Kindergarten Group at the MIT Media Lab.

(Click to enlarge.)

(/pdfs/blogs/edutopia-ryder-scratch-triangle.png)

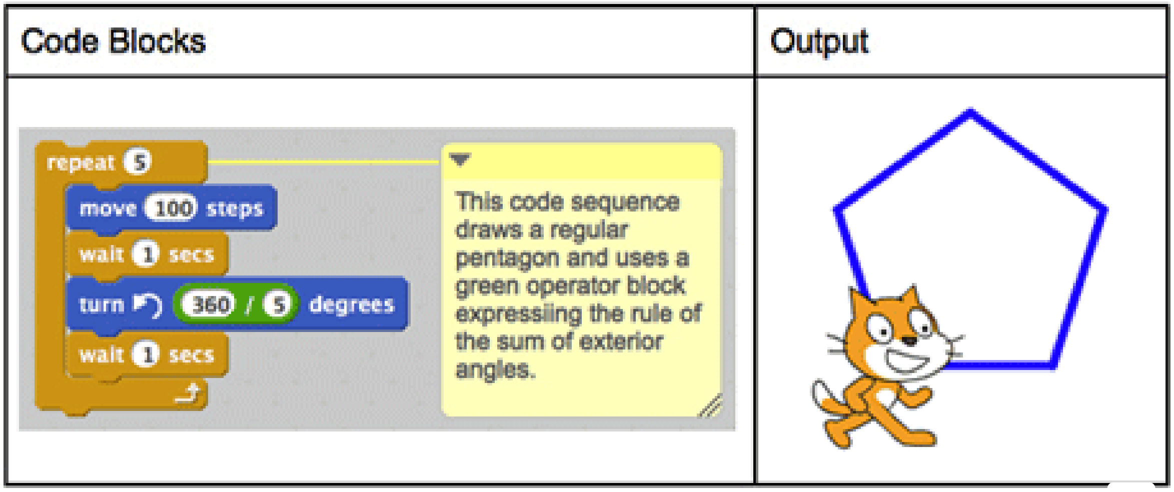
This is a great opportunity to talk to your students about measuring angles and degrees if they haven't yet been introduced to these concepts. You'll notice that the "Turn" value is a measurement of the exterior angle and that all three exterior angles of a triangle will add up to 360 degrees. Let's do another!

Final Challenge: Draw a simple, regular pentagon and use a mathematical "Operator" block

- Now let's incorporate a little more computation and take advantage of the mathematical operator blocks in the Palette. To draw another simple, regular polygon (like a pentagon), we can write our understanding of the rule of exterior angles as a coded expression.
- Go to the green "Operators" section of the Palette and locate the block for division, represented by the forward slash symbol (/). Insert the whole block into the space provided in the "Turn" block.

- Edit the "Repeat" loop to repeat five times, once for each angle or side, and run the program.

(/pdfs/blogs/edutopia-ryder-scratch-pentagon.png)



Code Blocks	Output
repeat 5 move 100 steps wait 1 secs turn ↺ 360 / 5 degrees wait 1 secs This code sequence draws a regular pentagon and uses a green operator block expressiing the rule of the sum of exterior angles.	

(/pdfs/blogs/edutopia-ryder-scratch-pentagon.png)

Scratch is a project of the Lifelong Kindergarten Group at the MIT Media Lab.

(Click to enlarge.)

(/pdfs/blogs/edutopia-ryder-scratch-pentagon.png)

Wrapping Up

Remember, programming is just giving instructions to a computer. It's a simple concept but it has big implications. Teaching kids to program gives them a new sense of agency over their technology. Soon they'll grow from seeing themselves as simple consumers of technology into creative makers of technology. If they can teach their computer to draw, what else can they make it do? Think of programming not as a dry task but as a new medium for students to learn with and express their ideas.

With this first step into programming, you've explored four core computational thinking concepts: sequences, events, loops, and operators. Further posts in this series will include parallelism, conditionals, data, and more.

Here are some possible extensions for this activity:

- What other shapes can your students draw?

- Challenge students to draw their initials or messages.
- Challenge students to draw using X/Y coordinates rather than angle turns.
- Challenge students to draw patterns of these shapes with nested loops (loops inside loops) that create beautiful geometric artwork.

Finally, some other Scratch educator resources:

- Visit the ***ScratchEd*** **(http://scratched.gse.harvard.edu/)** website, a community of educators that help each other learn and use Scratch. You can find lessons, activities, project ideas, or simply have your questions answered by a friendly fellow educator.
- ***Creative Computing Workshop*** **(https://creative-computing.appspot.com/preview)** is a free online workshop where you can learn more about using Scratch and supporting computational thinking.
- ***Scratch Day*** **(http://day.scratch.mit.edu/)** is a worldwide network of gatherings, where Scratchers meet up, share projects and experiences, and learn more about Scratch. Great for kids and adults!
- ***Geometry Cat*** **(http://scratch.mit.edu/projects/24582971/)** is the link where you can find the full code for this project.

FILED UNDER

Coding in the Classroom

Education Trends

Lesson Plans

Technology Integration

3-5 Upper Elementary