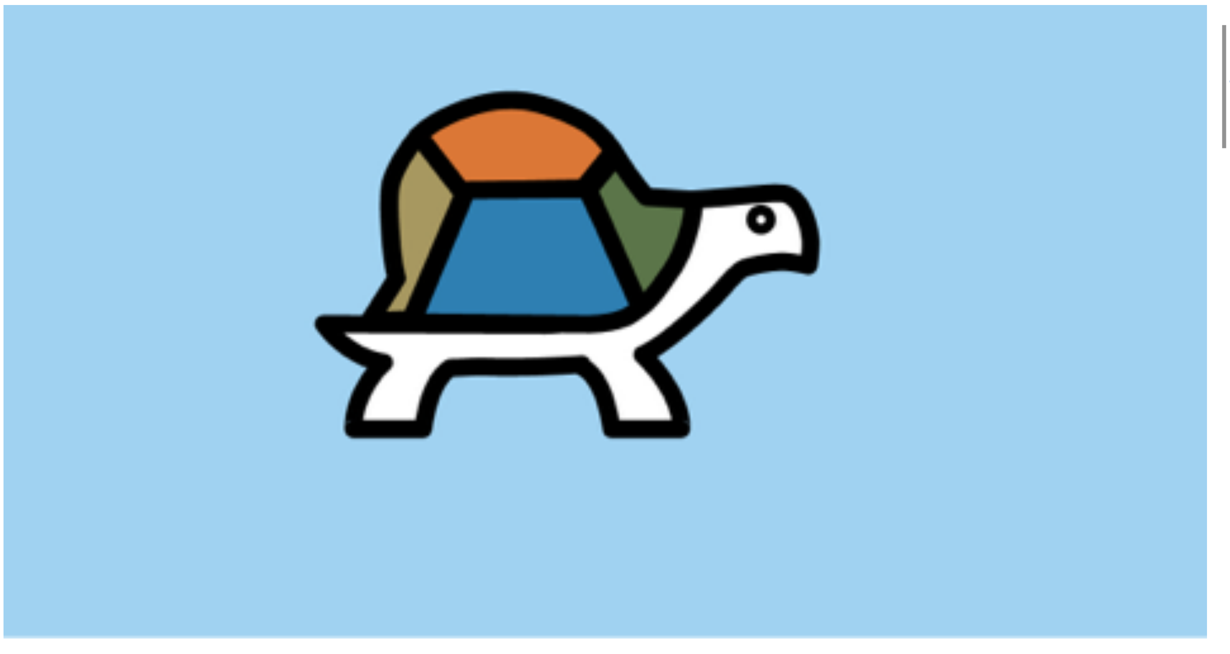


SUGAR LAB

Advanced Blocks Manual for Turtle Blocks

Sugar Labs

Prepared by Emily Ong Hui Qi

sugarlabs

Table of Contents

All About Turtle Blocks

What is Turtle Blocks?

Turtle Blocks is an activity with a Logo-inspired graphical "turtle" that draws colorful art based on snap-together visual programming elements. Its "low floor" provides an easy entry point for beginners. It also has "high ceiling" programming features, which will challenge the more adventurous student.

Where can we get Turtle Blocks?

Note

This manual is based on a JS version and there is a Python version that runs on Linux platforms and native to Sugar.

http://activities.sugarlabs.org/en-US/sugar/addon/4027

Note: There are two inter-compatible programs: Turtle Art and Turtle Blocks. Turtle Art, which closely parallels the Java version of Turtle Art maintained by Brian Silverman, offers a small subset of the functionality of Turtle Blocks. **Sugar users probably want to use Turtle Blocks rather than Turtle Art.** (Also see Turtle Confusion, a collection of programming challenges designed by Barry Newell; as well as the Activities/TurtleFlags, Activities/Tortuga de Mexico and Activities/Amazonas Tortuga variants.)

Debian (and Ubuntu) users can install Turtle Blocks from a repository maintained by Alan Aguiar (https://launchpad.net/~alanjas/+archive/turtleblocks):

1. sudo add-apt-repository ppa:alanjas/turtleblocks
2. sudo apt-get update
3. sudo apt-get install turtleblocks

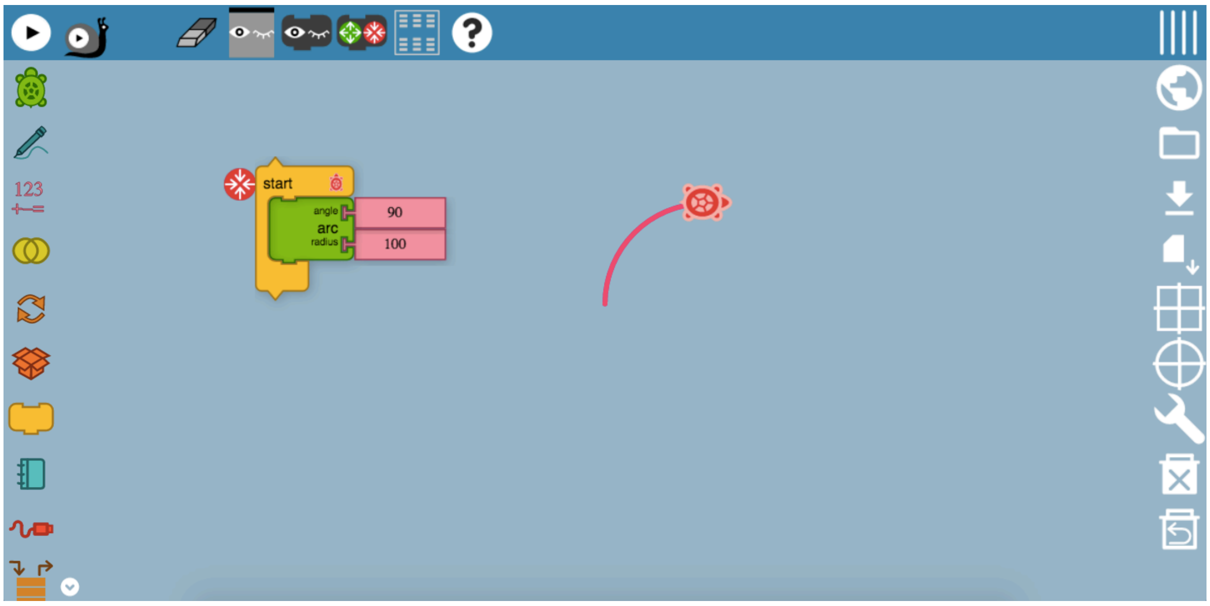
Fedora users can do:

1. sudo yum install sugar-turtleart

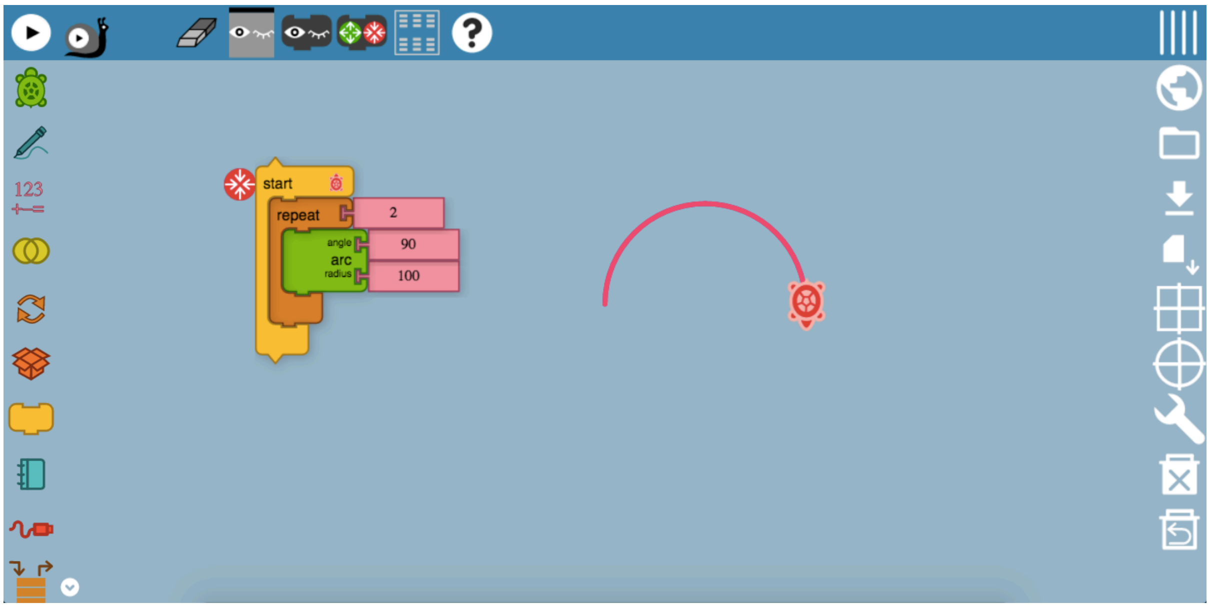
For those of you who would like to use Turtle Blocks in a browser, there is a mostly compatible version at Turtle Blocks JS. See the Guide (en ES) for more details.

Turtle Palette

Angle-Arc-Radius Block



As it can be seen, this block controls the angle that the Turtle would be moving. Have you noticed the 90° turning point? This is a result of the angle of the block being set to 90°. However, in this block, we do not expect a sharp 90° turn, unless we specify the arc radius to be 0 counts. In this case, we have set the arc radius to be 100 counts. The 100 counts specify the radius of this circle. By setting it to these values, we have created the arc of a quarter circle!

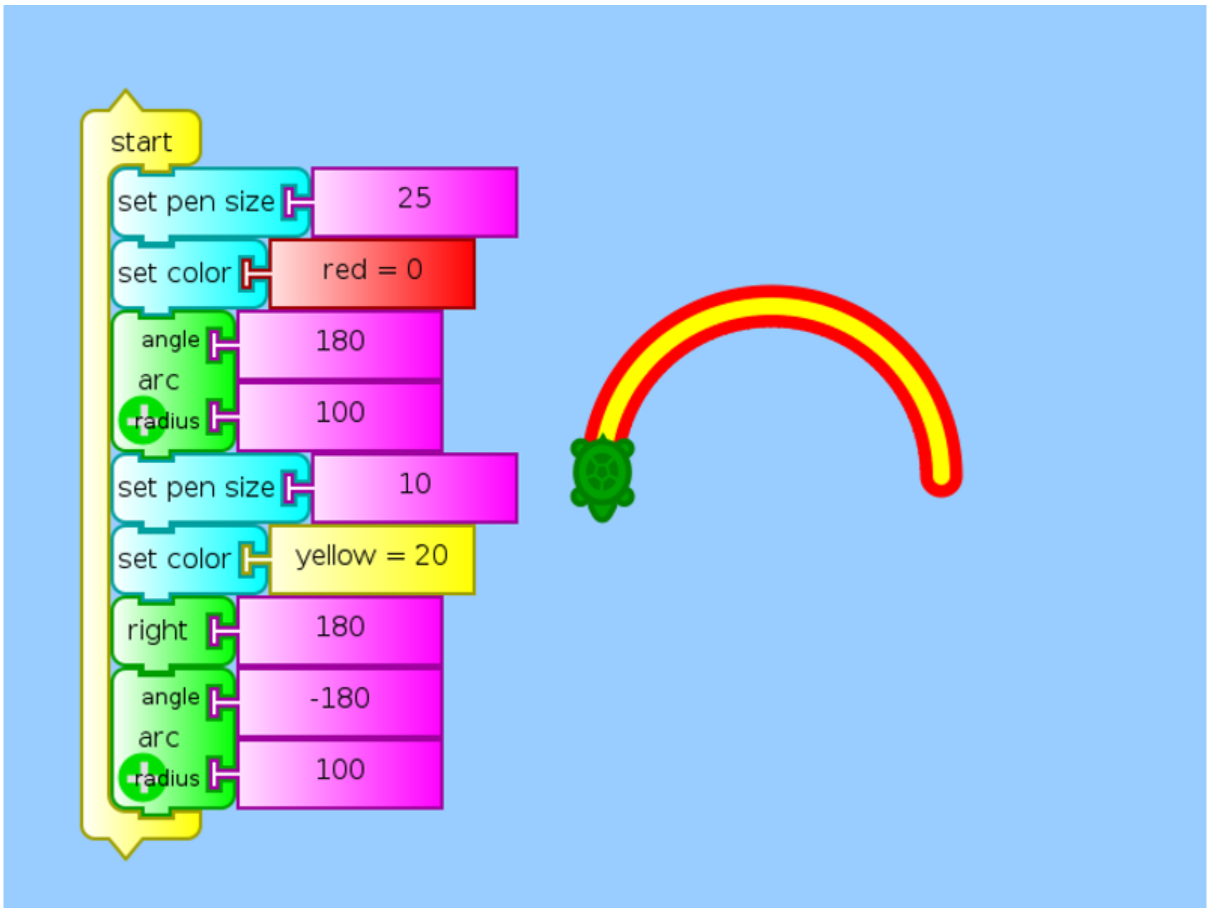


But, what about a semicircle? We could create another angle-arc-radius block with the same angle value, and combine them in a repeat block found in the flow palette? Why so? This will mean that the Turtle will be making another 90° turn and drawing the remaining radius of 100 units, setting the diameter of this semicircle to be 200 units.

Of course, we could change the angle and arc radius to different values to experiment with different designs! Moreover, try combining with different move blocks as well! But here's a tip, a 360° angle count at any radius of your choice, would form a full circle! Alternatively, you could repeat the above codes 4 times.

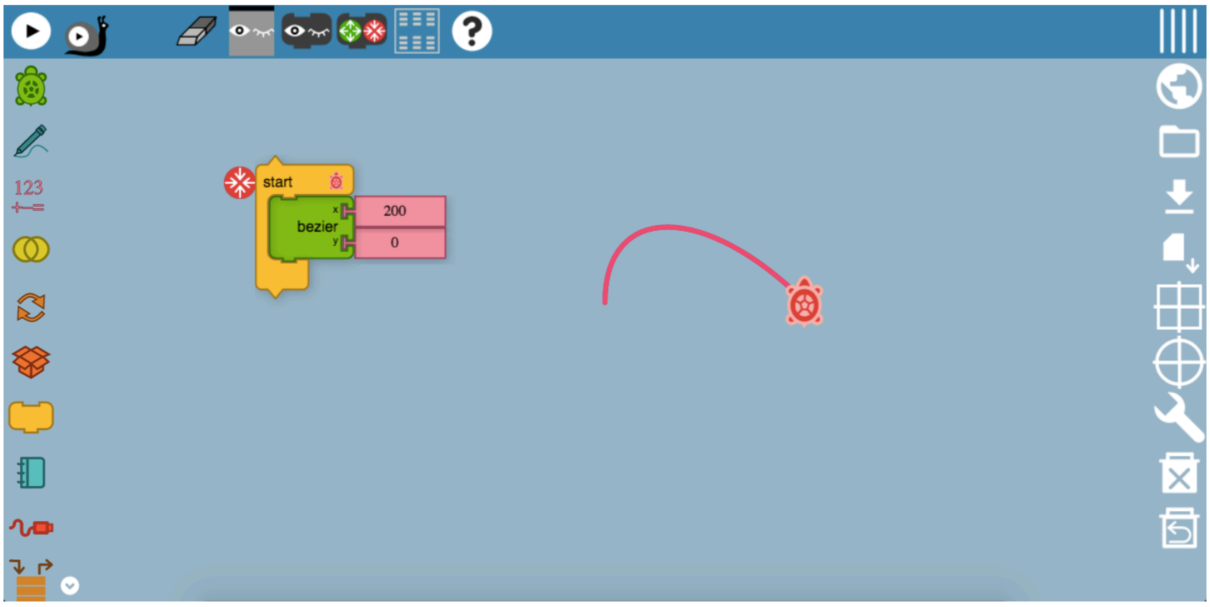
This would definitely be useful for visualizing geometric shapes and learning the concept of angles and radius.

This is an example of using the angle-arc-radius block to create a little rainbow! (Taken from the Turtle Art wiki page)

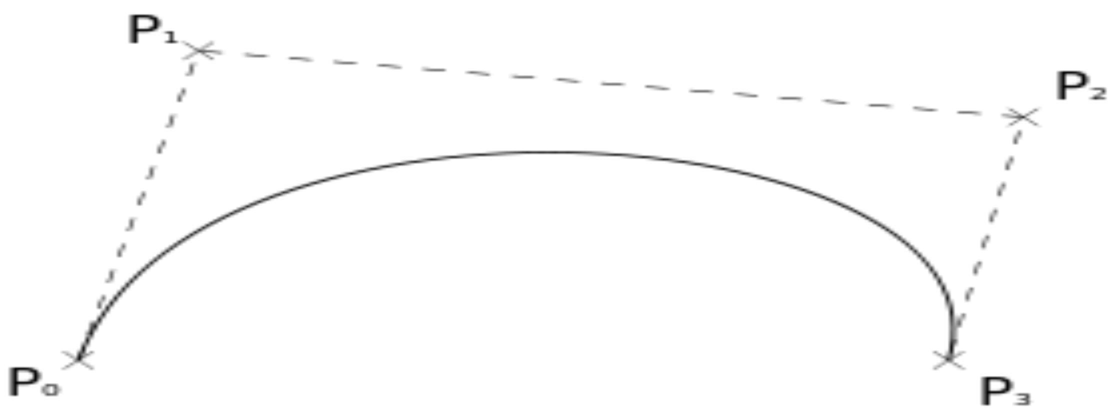


```
start
  set pen size 25
  set color red = 0
  arc angle 180
      radius 100
  set pen size 10
  set color yellow = 20
  right 180
  arc angle -180
      radius 100
```

Bezier Block

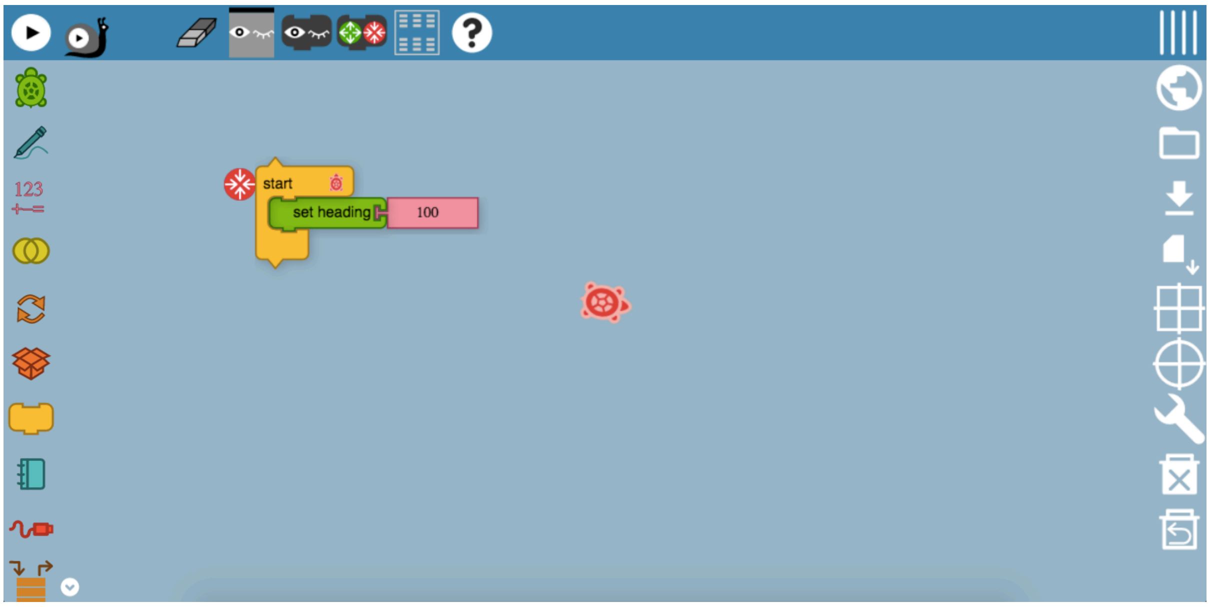


Well, the Bezier block may be slightly confusing. What is a Bezier? A Bézier curve is a parametric curve frequently used in computer graphics and related fields. Generalizations of Bézier curves to higher dimensions are called Bézier surfaces, of which the Bézier triangle is a special case. It looks something like this.



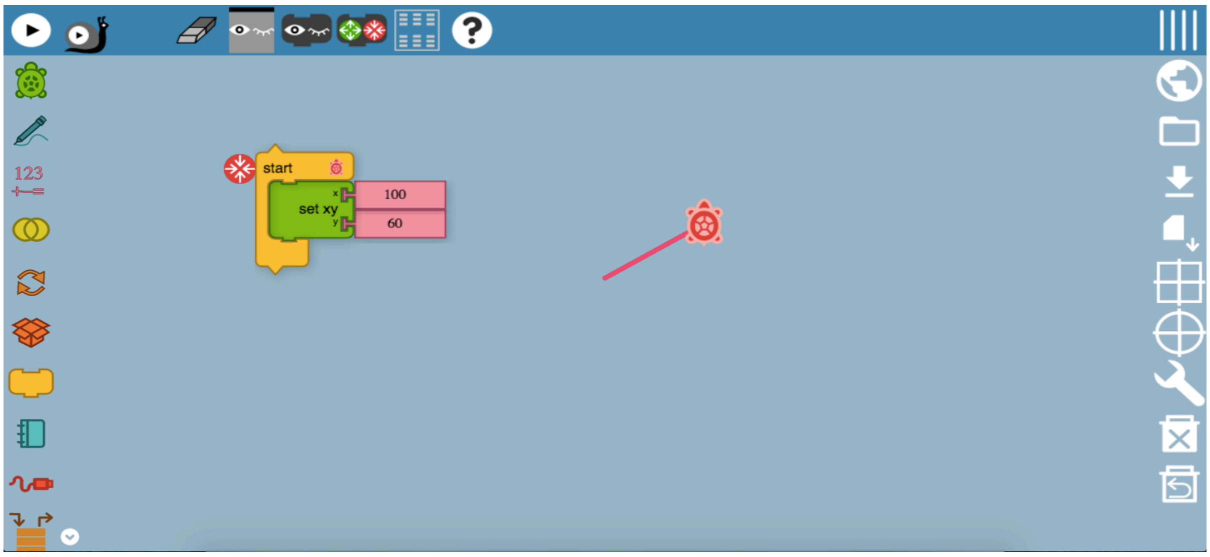
A Bezier block causes the Turtle to move in a very sharp angle. A smaller value gives a steeper and smaller curve, while a larger value gives a wider curve. The x represents the horizontal direction that the Turtle would be moving, while the y represents the vertical direction. Setting a higher y direction tends to create wider and longer curves, and curves would not tend to be too narrow. Try it yourself!

Set Heading Block

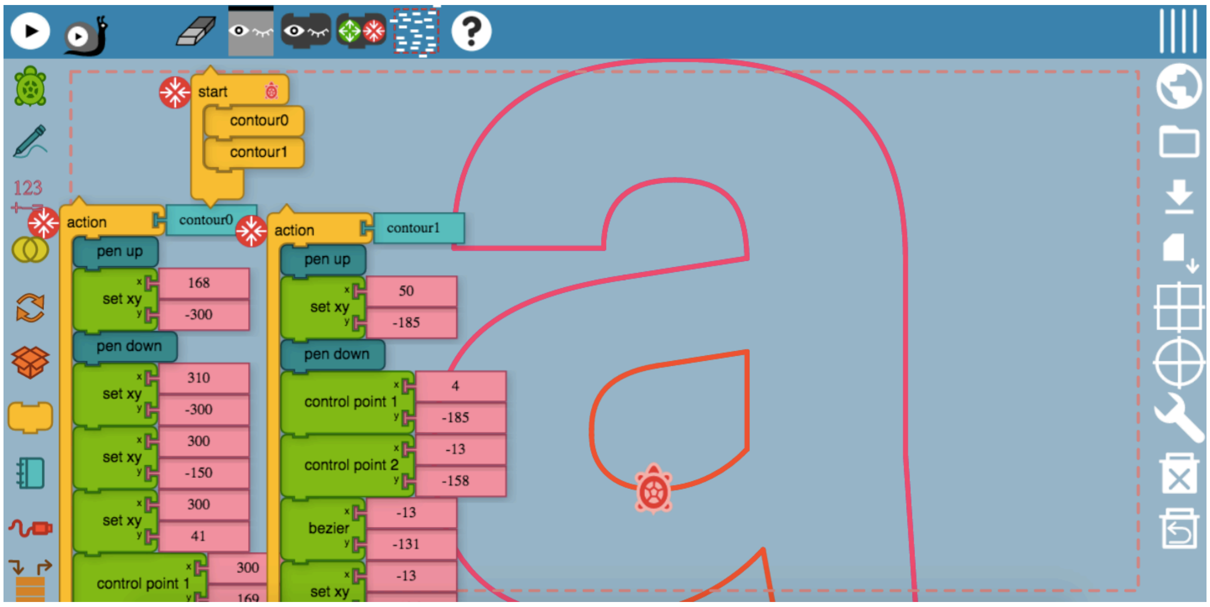


Well, the set-heading block really seems like a left and right movement block. As the name suggests, it specifies the direction that the Turtle would be heading to, based on the angle counts. In this case, assuming that the north direction is vertically upwards, the Turtle is in-between the east and south direction, but has not yet reached south-east. Always remember that the Turtle moves in a clockwise direction here!

Set xy Block



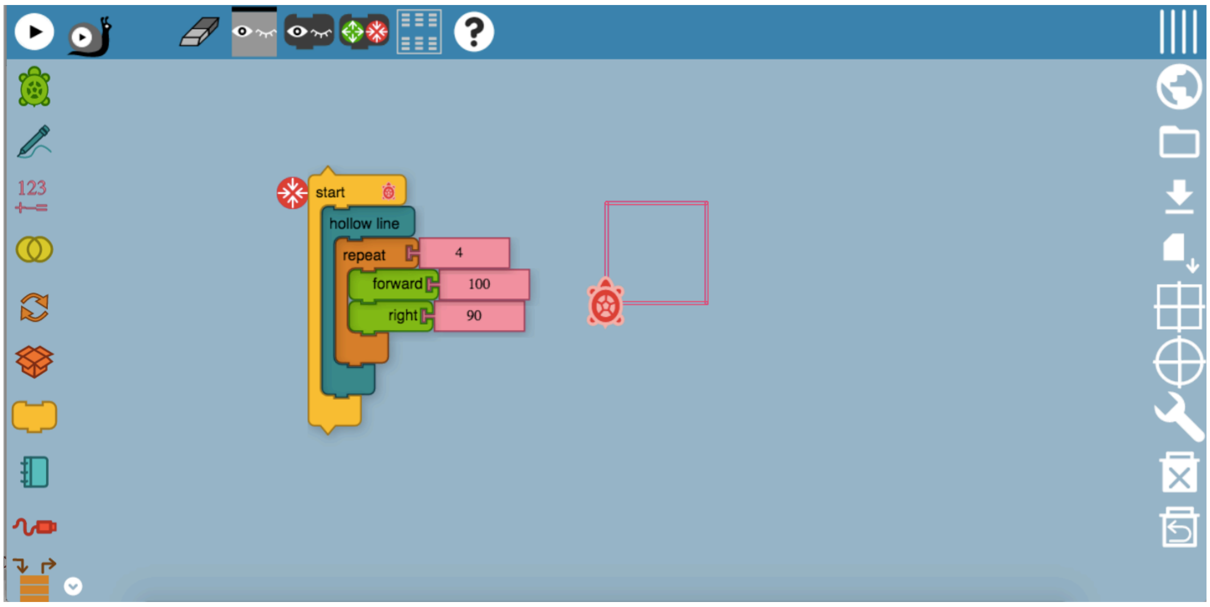
Doesn't this xy block reminds you of the straight-line graph? Yes, it indeed is! Like the Bezier block, this block sets the x and y direction of that the Turtle would be moving to. In this case, the Turtle has moved 100 counts in the horizontal direction, which means that it is 100 horizontal counts away from the starting point. Applying the same concept, it is 60 counts vertically away from the starting point. However, have you noticed how the line is slanted and it is not exactly 60 vertical counts away from the starting point? The y point is in fact 60 vertical counts away from the end of the horizontal line, which forms this straight-line graph. It will be useful for you to form shapes and even applying the concept of graph. Perhaps, you may even want to use Turtle Blocks to visualize graphical equations!



If you feel that you are up for it, you can even use the set xy block to create alphabets!

Pen Palette

Hollow Line Block

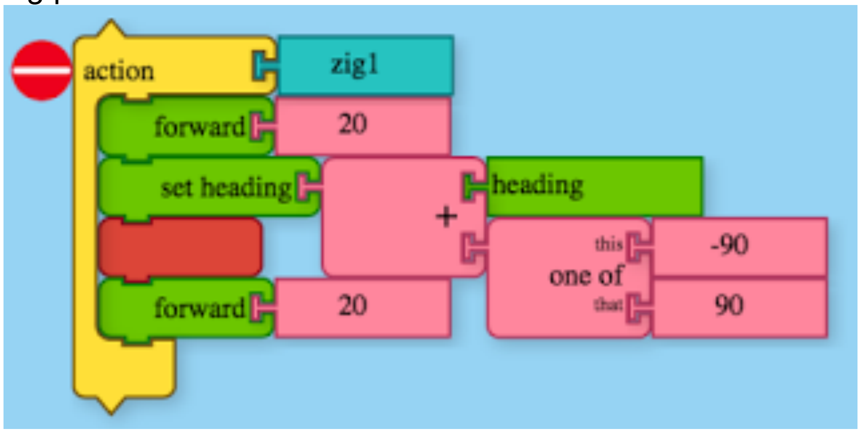


The hollow line block can be used for creating hollow lines throughout the drawing, and is useful for 3D printing.

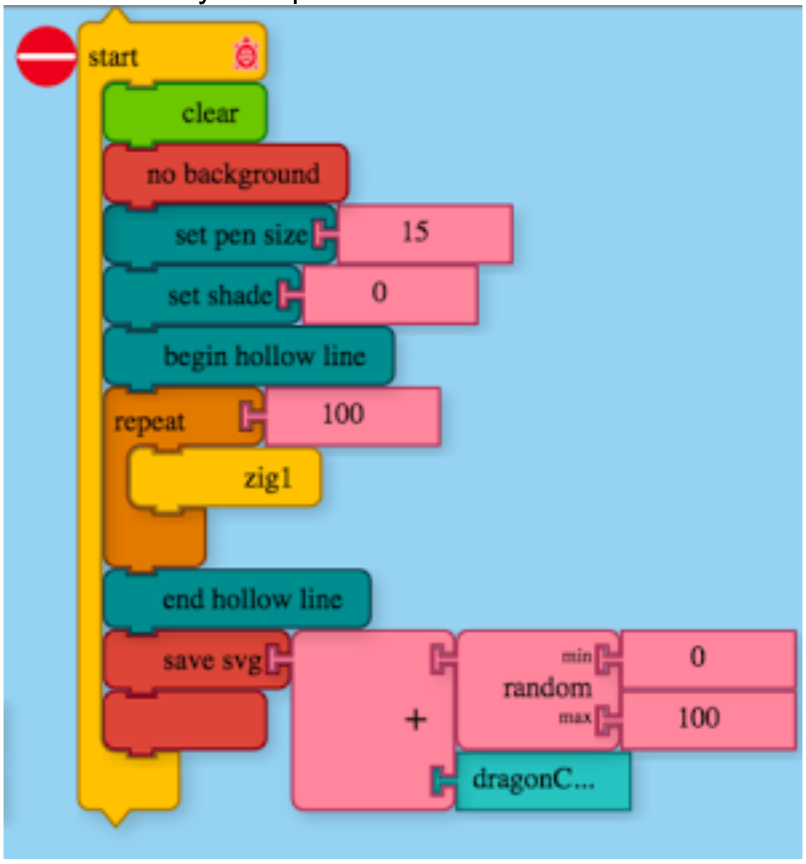
Below is a write-up from Josh Burker on Hollow lines for 3D Printing:

“

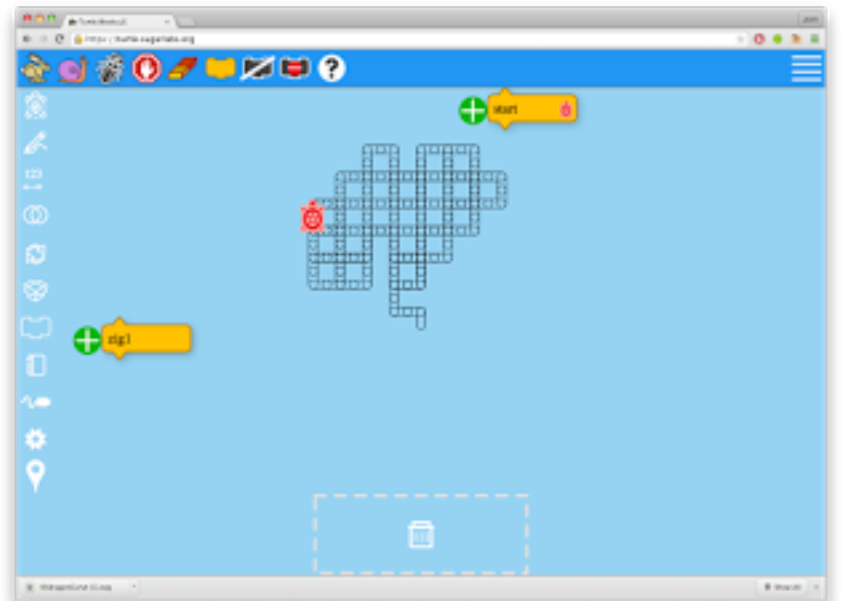
We can recreate the TurtleArt Dragon Curves procedure in Turtle Blocks. Walter also added a One of block to Turtle Blocks to make this possible. Start by creating a zig procedure.



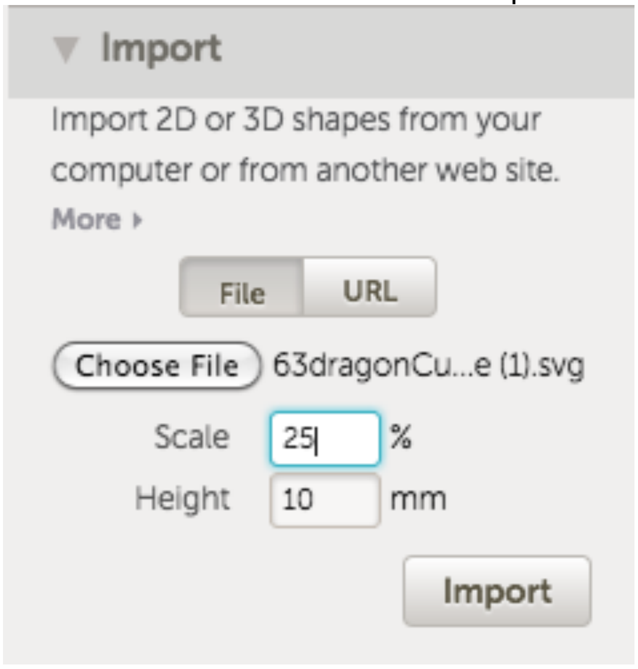
When you create your master procedure you need to add a couple of blocks to make the SVG ready to import into Tinkercad.



First, suppress the background by adding the red "no background" block. Next, add the begin hollow line block before the turtle starts drawing. Once the design is drawn, add the end hollow line block to the procedure. The design will look a little unusual when you run it. An SVG file downloads at the end.



The scale at which you import your design into Tinkercad depends on the size of the build plate on your 3D printer. I sized scaled my import to 25% and further reduced the size of the model once it was imported.

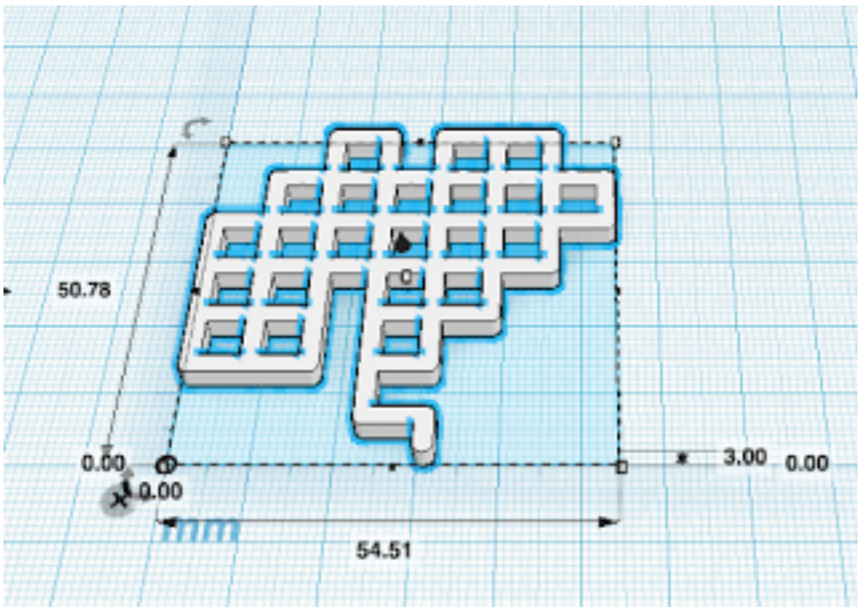


Import 2D or 3D shapes from your computer or from another web site.

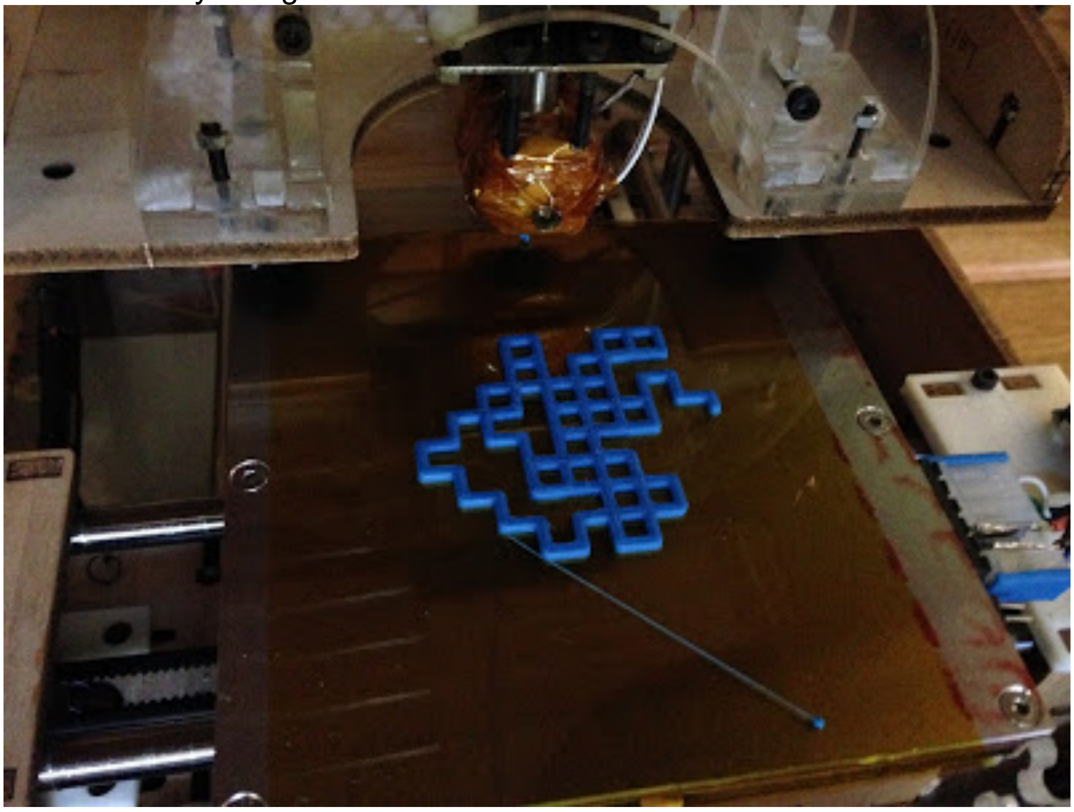
63dragonCu...e (1).svg

Scale 25 %

Height 10 mm



The resulting model sliced fine in ReplicatorG. I printed the model with 2 shells and 5% infill on my Thing-O-Matic.



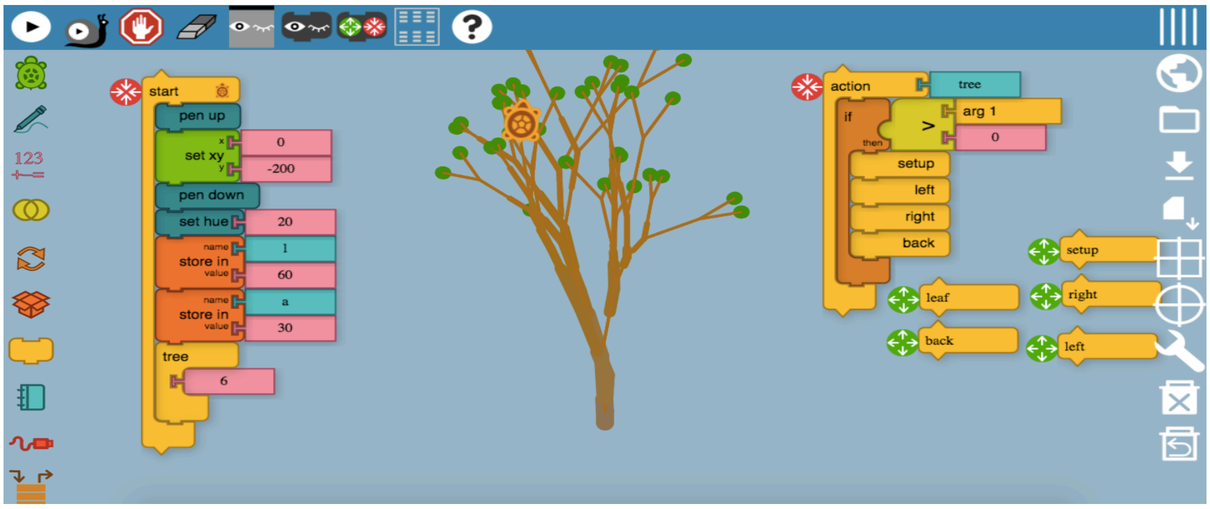
Turtle Blocks takes a few of the hurdles out of trying to get a turtle graphics project 3D printed. By streamlining the process and eliminating the intimidating step of getting Inkscape running and converting PNG files to SVG files, Turtle Blocks should help more people experiment with programming 3D printed designs.

”

1 Turtle Blocks

Set Pen Block

This is a fine example from the Turtle Blocks planet, where a user uses math functions to set pen direction and variations to draw a beautiful, realistic and nice tree.



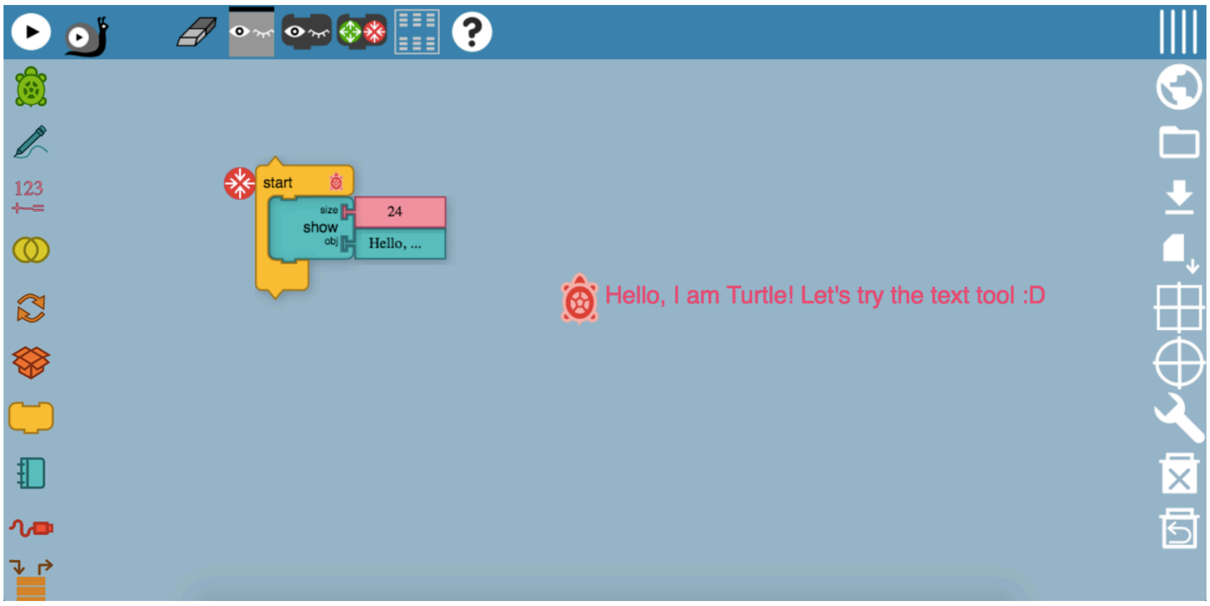
Media Palette

Turtle Blocks provides rich-media tools that enable the incorporation of sound, typography, images, and video.

At the heart of the multimedia extensions is the *Show* block. It can be used to show text, image data from the web or the local file system, or a web camera. Other extensions include blocks for synthetic speech, tone generation, and video playback.

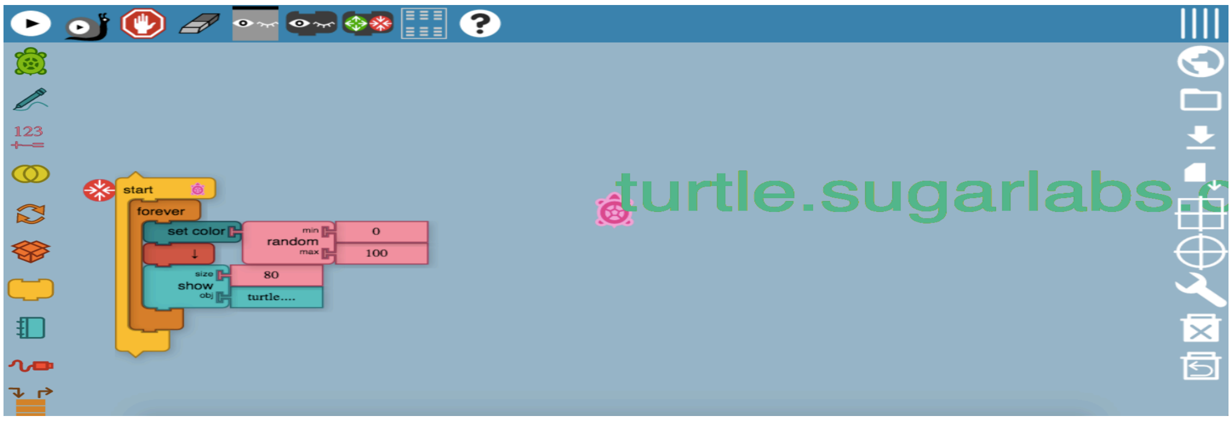
Show Block

Text Tool

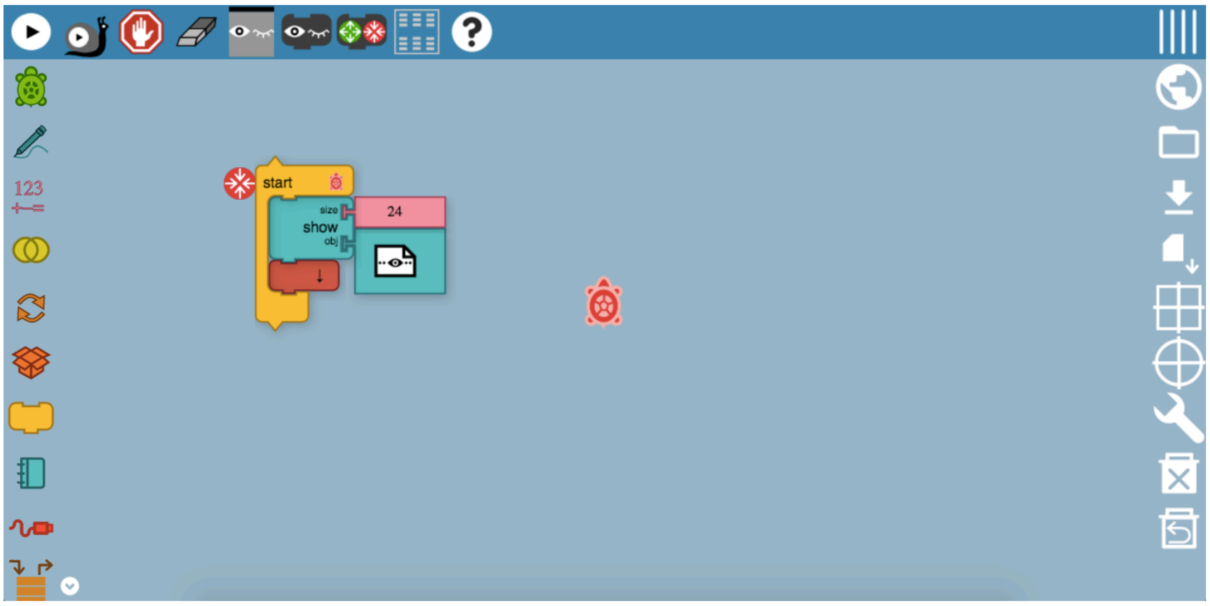


This is the show block! It displays the input that you have entered. For example, the size of the object here was specified to be 24, while the object, have been set to "Hello, I am Turtle! Let's try the text tool!". This explains what the Turtle is trying to show here. The object has been specified to be a text input. You could really try to experiment with font sizes and print out different texts on the screen!

This is an example from the TurtleBlocks planet, where the user randomize text colors:

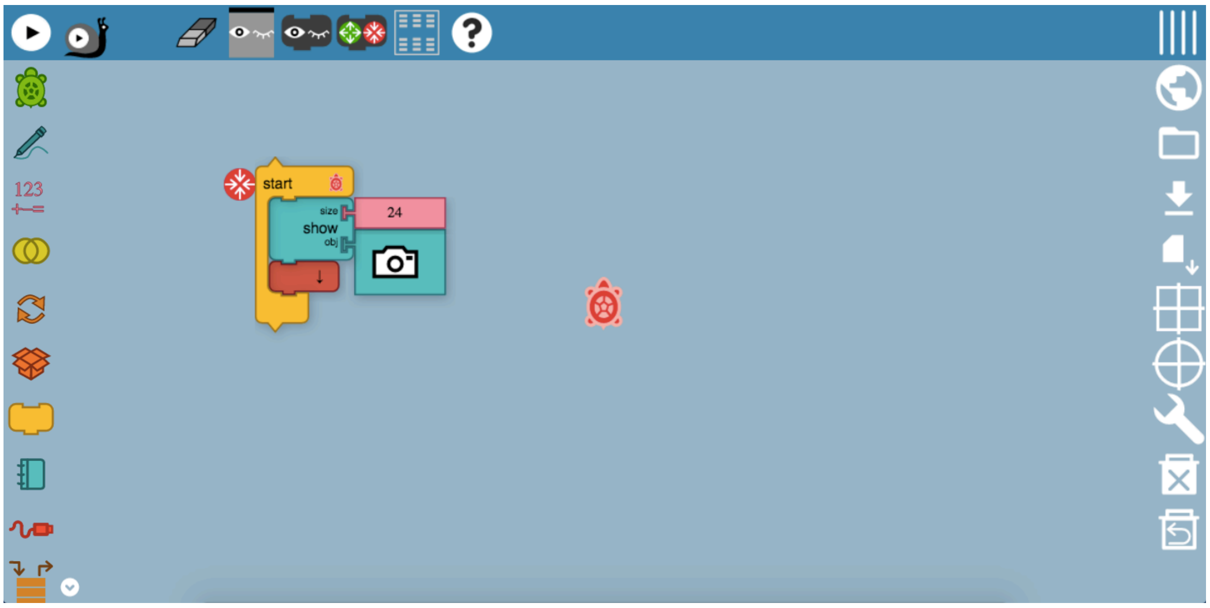


Video Tool



In the show block, one of the most amazing function is to display a live video camera on the canvas, which in this way, you can record anything that you want. To adjust the size of the video screen, you can also make edits to the size tool.

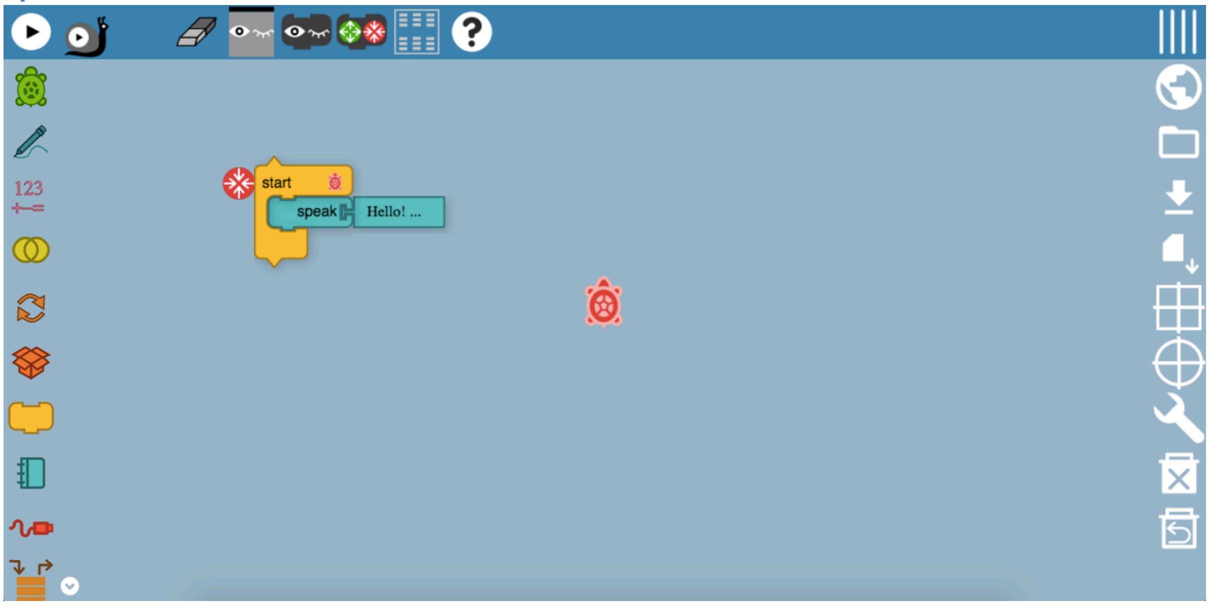
Camera Tool



Like the video tool, the camera tool displays a camera screen on the canvas. Incorporated with the loudness block, it can be used as an image capture after the loudness block starts a burglar alarm, or even a friendly "Hello" welcome. (See Sensors: Loudness Block)

Try it now!

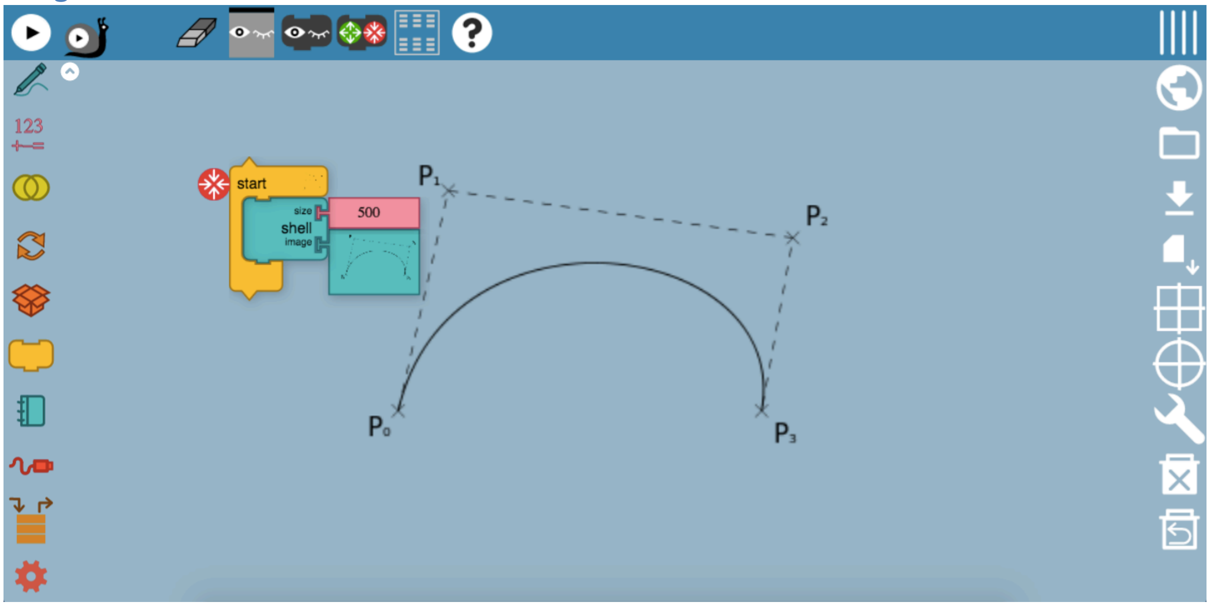
Speak Tool



The speak tool is used to program the Turtle to produce sounds, of whatever that you have typed. For example, in this program, the input for the speak block is “Hello! I am Turtle.” So, turn up your volume and you can hear the sound output! Just imagine how this great function can help you to pronounce words, and even make your art even cooler as the Turtle speaks while creating shapes.

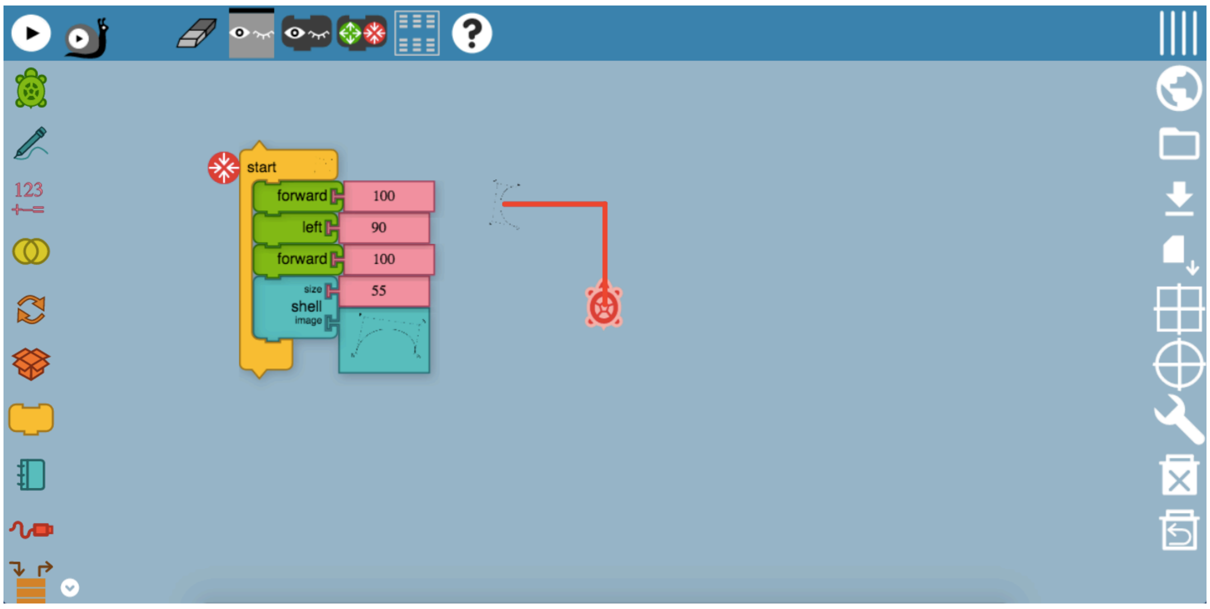
Shell Block

Image tool



The image tool is handy for displaying different kinds of images on the canvas. You could even use the image tool to display a little map and create a program for the Turtle to draw it out! It is also useful for displaying a Powerpoint, when incorporated with the keyboard function. (See Sensors: Keyboard Block)

Turtle Shell



```
start
  forward 100
  left 90
  forward 100
  shell size 55
        image
```

Perhaps you may be thinking, why am I using a Turtle to draw? Well, did you know that you could change the image of the Turtle? We call this, changing the image of the Turtle shell. Firstly, the Turtle has been programmed to more in a certain direction. This is to allow you to better visualize the shell image of the Turtle. Here, the Turtle's shell was replaced with a Bezier!

Number Palette

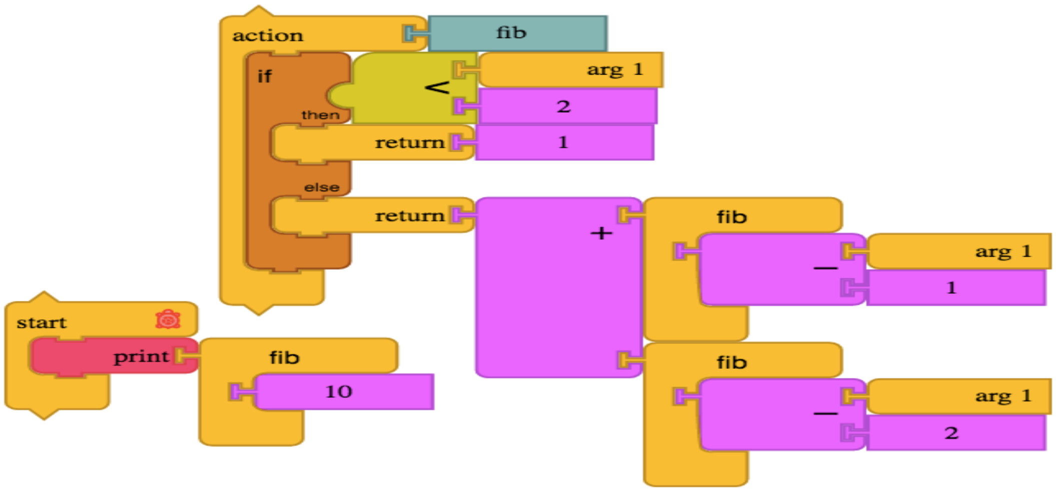
Perhaps you may have heard of Fibonacci numbers, perhaps you haven't.

Part of the Mathematics glossary: The **Fibonacci** sequence is a set of numbers that starts with a one or a zero, followed by a one, and proceeds based on the rule that each number (called a **Fibonacci** number) is equal to the sum of the preceding two numbers.

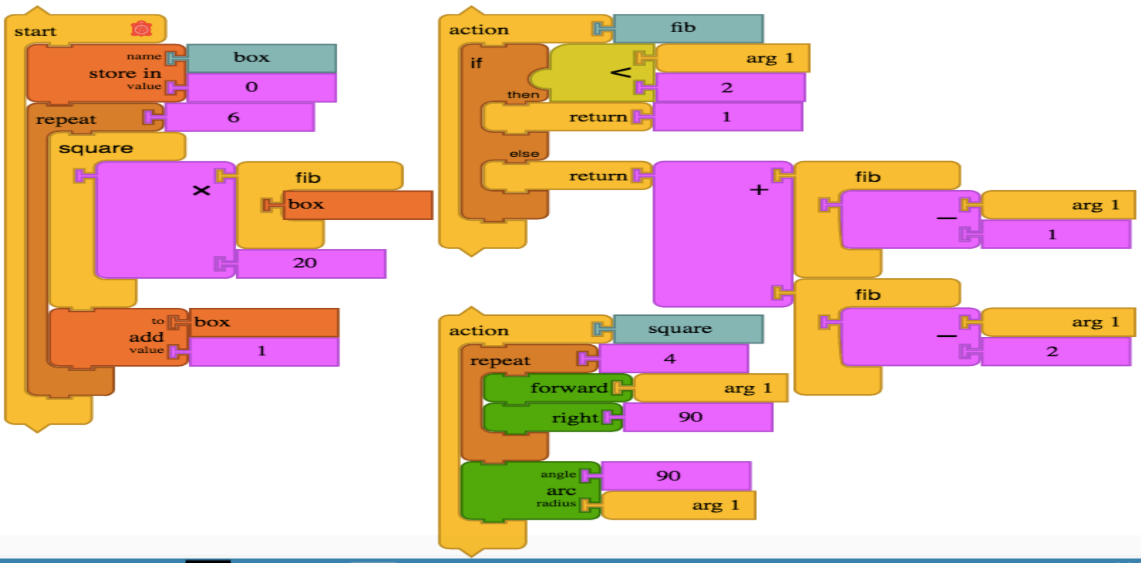
The number palette is often used with the Boolean palette to create artwork based on the values and math concepts, playing around with the numbers. Well, what about creating a nautilus based on this number palette?

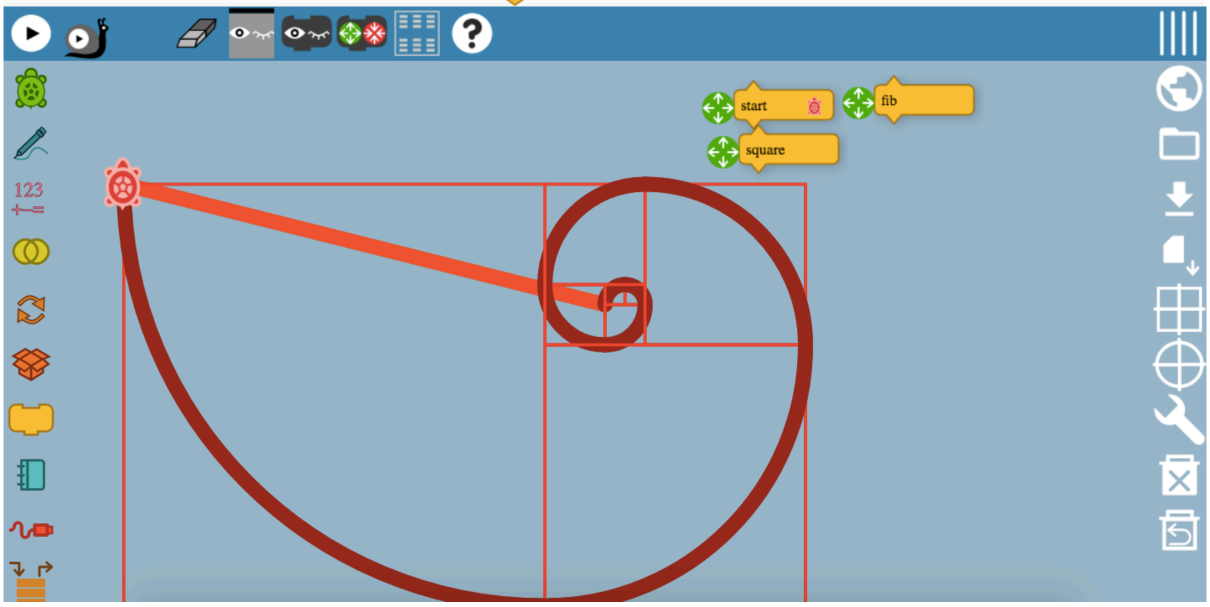
This is an example from the TurtleBlocks Guide:

Calculating the Fibonacci sequence is often done using a resursive method. In the example below, we pass an argument to the *Fib* action, which returns a value if the argument is < 2; otherwise it returns the sum of the result of calling the *Fib* action with argument - 1 and argument - 2.



In the second example, we use a *Repeat* loop to generate the first six Fibonacci numbers and use them to draw a nautilus.





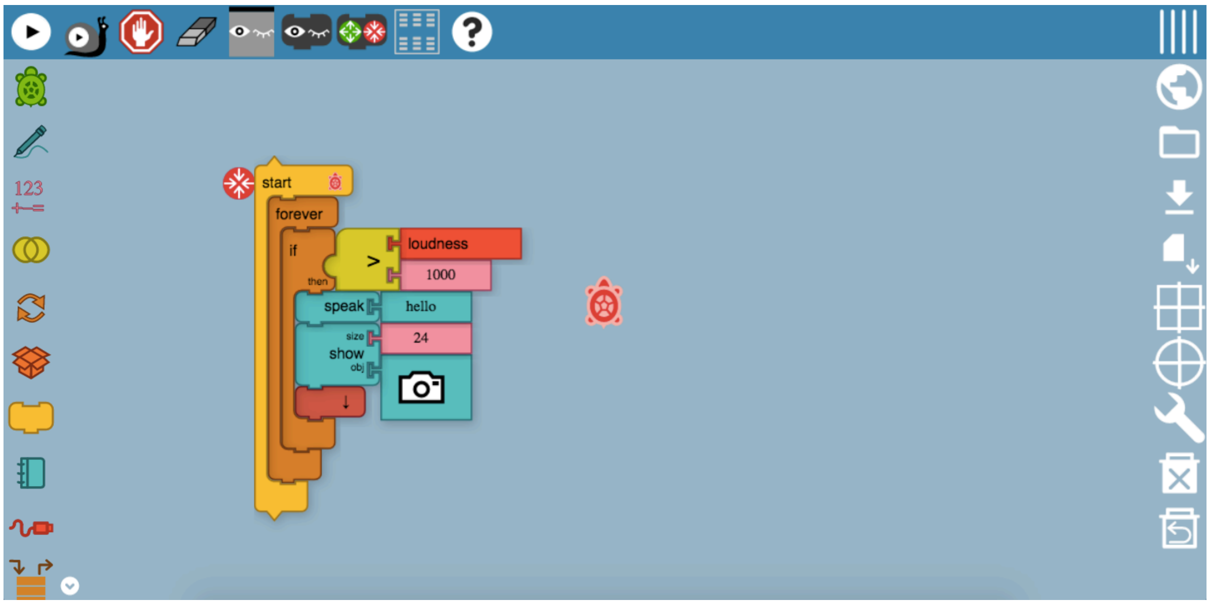
Sensors

Seymour Papert's idea of learning through making is well supported in Turtle Blocks. According to Papert, "learning happens especially felicitously in a context where the learner is consciously engaged in constructing a public entity, whether it's a sand castle on the beach or a theory of the universe". Research and development that supports and demonstrates the children's learning benefits as they interact with the physical world continues to grow. In similar ways, children can communicate with the physical world using a variety of sensors in Turtle Blocks. Sensor blocks include keyboard input, sound, time, camera, mouse location, color that the turtle sees. For example, children may want to build a burglar alarm and save photos of the thief to a file. Turtle Blocks also makes it possible to save and restore sensor data from a file. Children may use a "URL" block to import data from a web page.

Teachers from the Sugar community have developed extensive collection of examples using Turtle Block sensors. Guzmán Trinidad, a physics teacher from Uruguay, wrote a book, *Physics of the XO*, which includes a wide variety of sensors and experiments. Tony Forster, an engineer from Australia, has also made remarkable contributions to the community by documenting examples using Turtle Blocks. In one example, Tony uses the series of switches to measure gravitational acceleration; a ball rolling down a ramp trips the switches in sequence. Examining the time between switch events can be used to determine the gravitational constant.

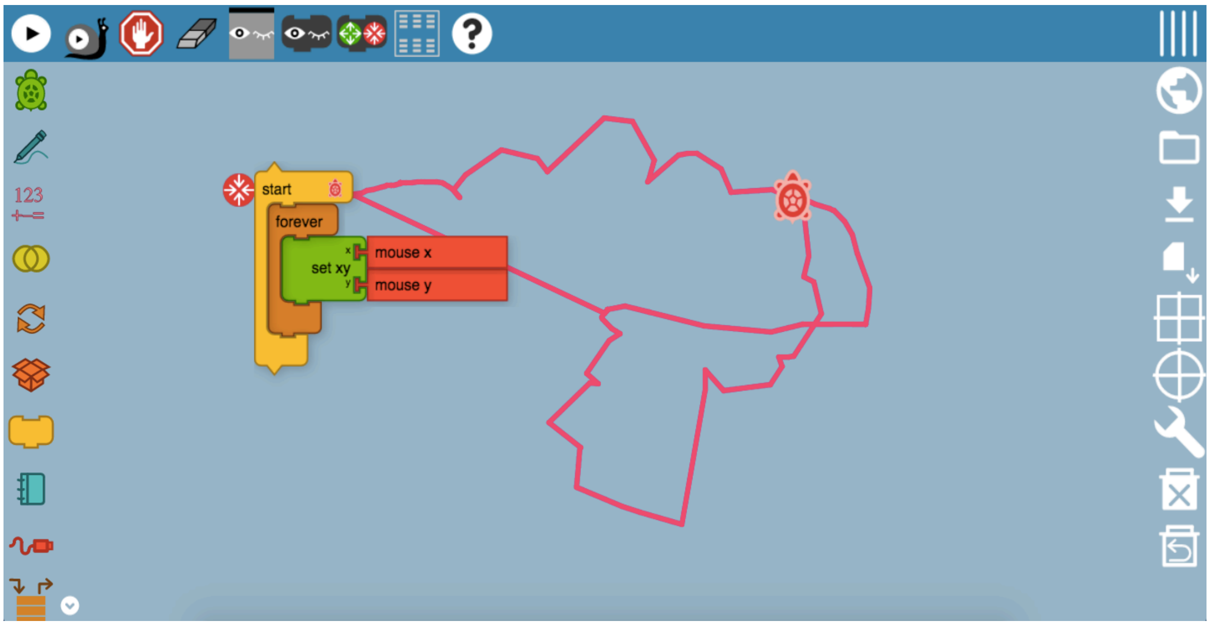
One of the typical challenges of using sensors is calibration. This is true as well in Turtle Blocks. The typical project life-cycle includes: (1) reading values; (2) plotting values as they change over time; (3) finding minimum and maximum values; and finally (4) incorporating the sensor block in a Turtle program.

Loudness Block



The loudness block is now working again, thanks to the constant updates and maintenance from the developers behind-the-scene. This program may seem slightly complicated, but don't worry! The whole program was enclosed within a forever block, which means that this program would continue running on forever and ever, with no breaks and interruptions. Next, the if-then block would carry out the action. Firstly, if the Turtle sensed a loudness value of 1000 counts from the surroundings, it will say, "Hello", and even take a snapshot of the surroundings!

Mouse Blocks

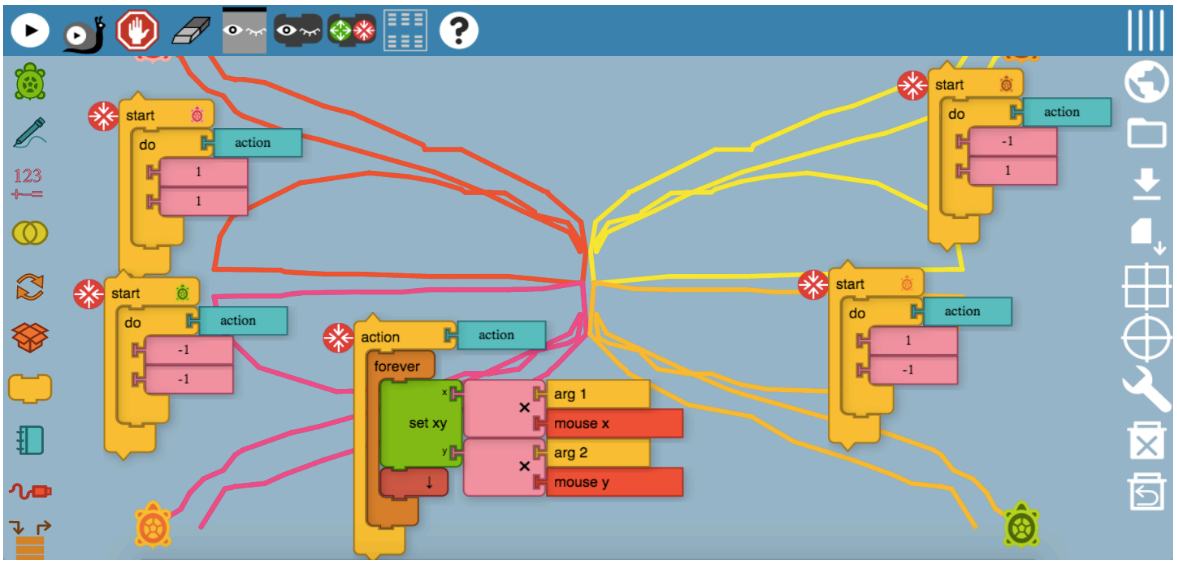


This mouse block is really nice to draw out different images. You can use a mouse-pad or even a mouse! Again, this program has been set to run forever without any interruptions, so one could continue drawing on and on unless the user stops the program. So, remember the set xy block? This block sets the x and y coordinates of the mouse and the Turtle will draw in the direction of wherever the mouse direction goes.

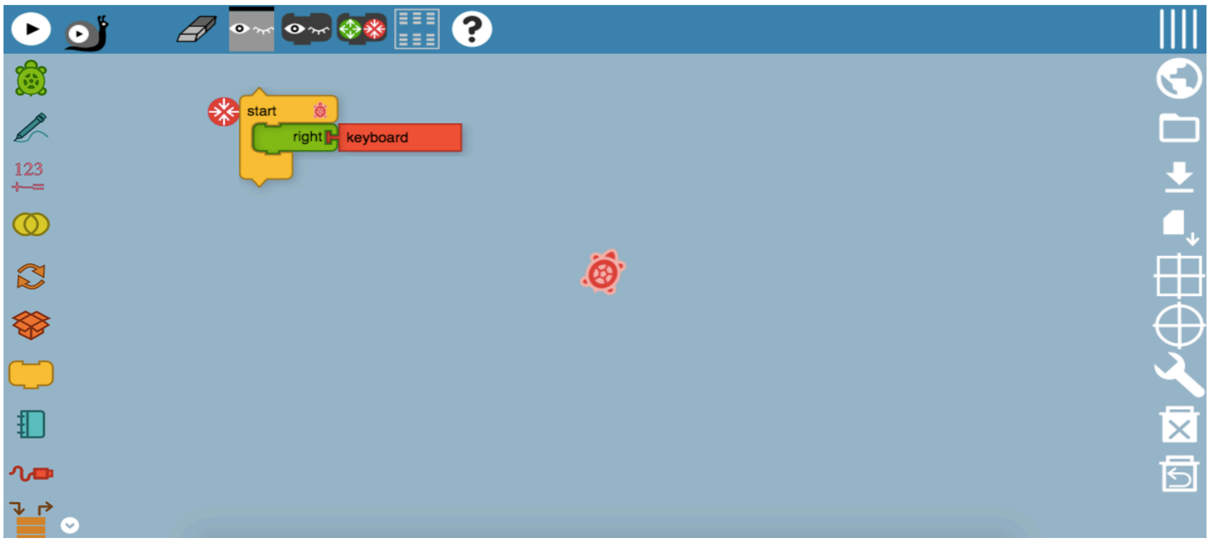
Writing your own paint program is empowering: it demystifies a commonly used tool. At the same time, it places the burden of responsibility on the programmer: once we write it, it belongs to us, and we are responsible for making it cool.

This is an example of using the mouse blocks together with math blocks to create reflection paint:

By combining multiple turtles and passing arguments to actions, we can have some more fun with paint. In the example below, the *Paint Action* uses *Arg 1* and *Arg 2* to reflect the mouse coordinates about the y and x axes. The result is that the painting is reflected into all four quadrants.



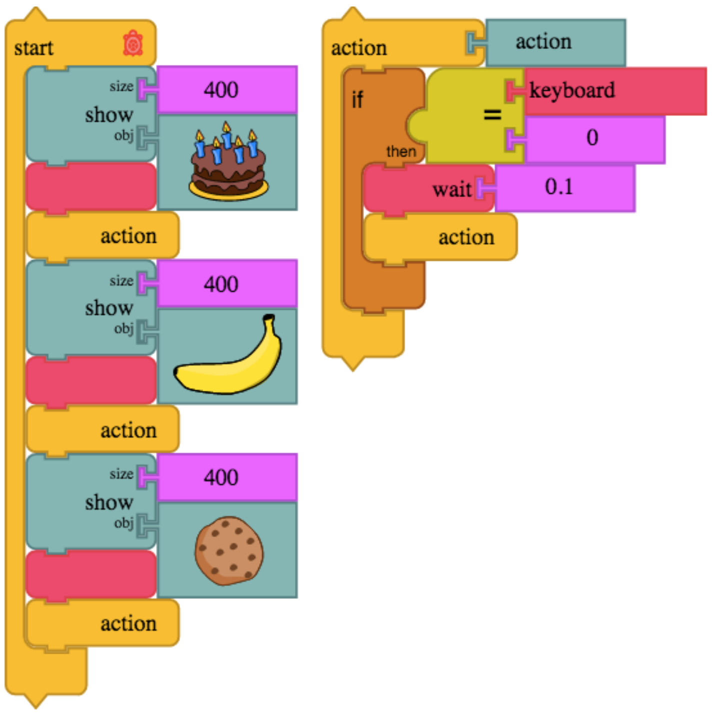
Keyboard Block



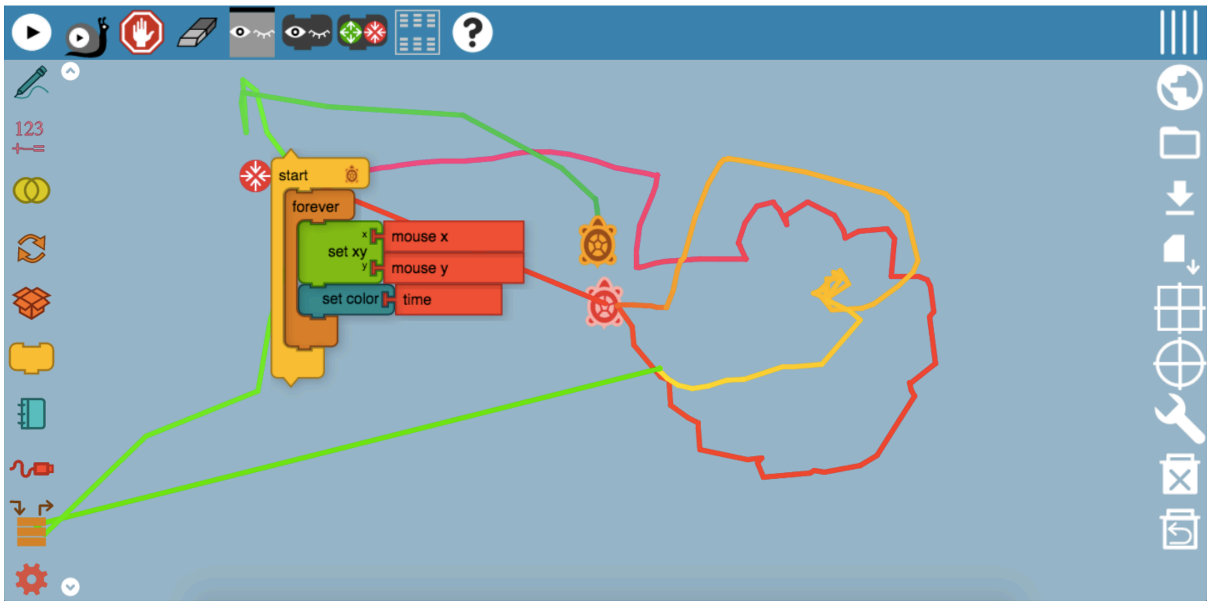
This is an amazing example of the sensor tool – the keyboard block! Just as how you would have used to control the Turtle by setting values, this will move the Turtle using the enter-key on the keyboard! In the image above, the Turtle is moving in a clockwise direction every time the enter-key is pressed.

Below is an example from TurtleBlock Guide on using keyboard for Powerpoint:

Why use Powerpoint when you can write Powerpoint? In this example, an Action stack is used to detect keyboard input: if the keyboard value is zero, then no key has been pressed, so we call the action again. If a key is pressed, the keyboard value is greater than zero, so we return from the action and show the next image.



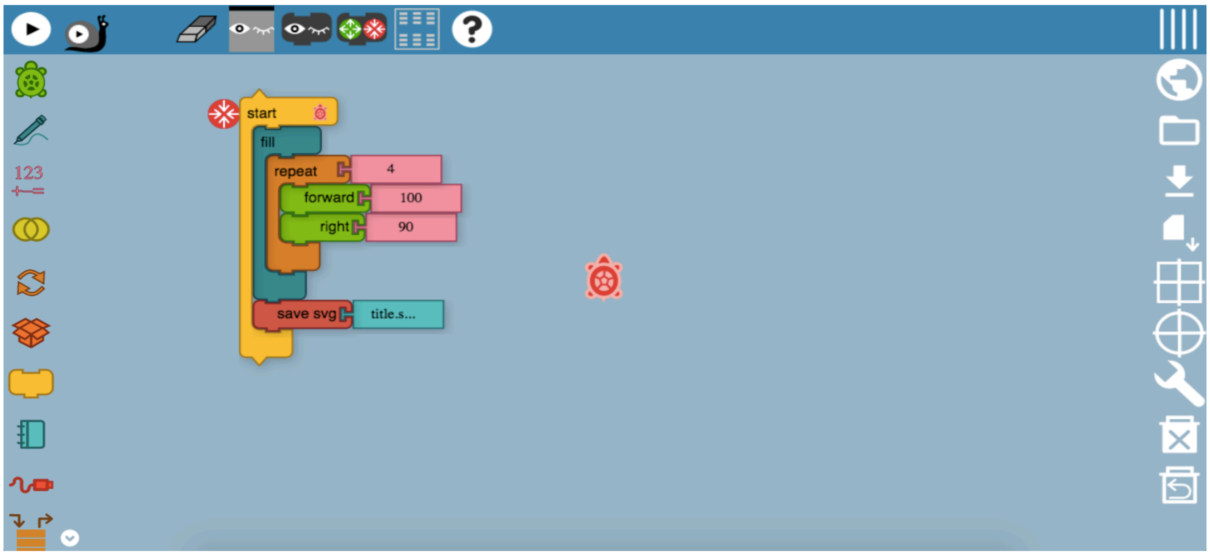
Time Block



In the previous mouse blocks write-up, the color of the drawings remain the same throughout, and this makes it really hard to beautify your drawing. Here, a time block was inserted as an addition to the mouse blocks, so that the color of the pen tool would change after a set period of time, giving your drawings the extra touch of creativity.

Extras Palette

Save SVG Tool



As it can be seen from the image above, I have created a colored square using the fill and movement tools. After which, the svg tool would be able to save the output of this program, which is the colored square into my files. It is useful in showing the output of the program without the palette and blocks.

Sources

http://whatis.techtarget.com/definition/Fibonacci-sequence

https://en.wikipedia.org/wiki/B%C3%A9zier_curve

https://upload.wikimedia.org/wikipedia/commons/d/d0/Bezier_curve.svg

http://wiki.sugarlabs.org/go/File:TurtleCard-7.png

http://wiki.sugarlabs.org/go/Activities/Turtle_Art

Credits

https://turtle.sugarlabs.org/

https://github.com/walterbender/turtleblocksjs/blob/master/guide/README.md

Mr Walter Bender