

15-104 Introduction to Computing for Creative Practice

Fall 2021

27 More Turtle Graphics

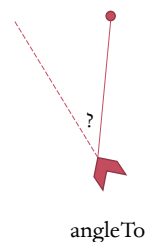
Instructor: Tom Cortina, tcortina@cs.cmu.edu, GHC 4117, 412-268-3514

1



Turtle API

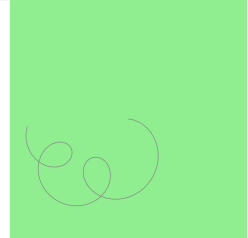
- `makeTurtle(x, y)` -- make a turtle at x, y, facing right, pen down
- `left(d)` -- turn left by d degrees
- `right(d)` -- turn right by d degrees
- `forward(p)` -- move forward by p pixels
- `back(p)` -- move back by p pixels
- `lowerPen()` -- set pen down
- `raisePen()` -- pick pen up
- `goto(x, y)` -- go straight to this location
- `setColor(color)` -- set the drawing color
- `setWidth(w)` -- set line width to w
- `face(d)` -- turn to this absolute direction in degrees
- `angleTo(x, y)` -- what is the angle from my heading to location x, y?
- `turnToward(x, y, d)` -- turn by d degrees toward location x, y
- `distanceTo(x, y)` -- how far is it to location x, y?



2

Using the mouse

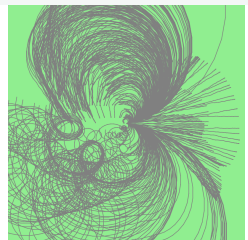
```
function setup() {
  createCanvas(400, 400);
}
function draw() {
  background("lightgreen");
  var ttl = makeTurtle(200, 200);
  for (var i = 0; i < 100; i++) {
    ttl.turnToward(mouseX, mouseY, 10);
    var d = ttl.distanceTo(mouseX, mouseY) / 10;
    ttl.forward(d);
  }
}
```



3

Using the mouse

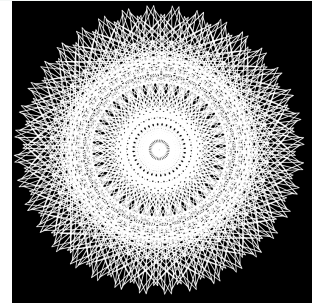
```
function setup() {
  createCanvas(400, 400);
  background("lightgreen");
}
function draw() {
  var ttl = makeTurtle(200, 200);
  for (var i = 0; i < 100; i++) {
    ttl.turnToward(mouseX, mouseY, 10);
    var d = ttl.distanceTo(mouseX, mouseY) / 10;
    ttl.forward(d);
  }
}
```



4

Static Drawing

```
function setup() {
  createCanvas(400, 400);
  background(0);
  var turtle = makeTurtle(130, 80);
  turtle.lowerPen();
  turtle.setColor(255);
  for (var i = 0; i < 1000; i++) {
    turtle.forward(150);
    turtle.right(141.5);
    turtle.forward(60);
    if (i % 20 === 0) { turtle.forward(70); }
  }
}
function draw() { }
```



5

Dynamic Drawing

```
var turtle;
var count = 0;

function setup() {
  createCanvas(400, 400);
  background(0);
  turtle = makeTurtle(130, 80);
  turtle.lowerPen();
  turtle.setColor(255);
}

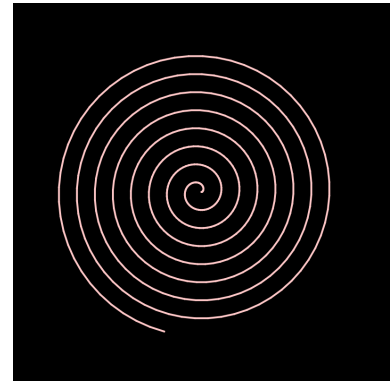
function draw() {
  turtle.forward(150);
  turtle.right(141.5);
  turtle.forward(60);
  if (count % 20 === 0) {
    turtle.forward(70);
  }
  count++;
  if (count == 1000) {
    noLoop();
  }
}
```

6



Spiral

Start in the center.
 Move forward and turn a bit to the left.
 Move forward a little more than last time
 and turn a bit to the left.
 Repeat...
 If the spiral touches the outside of
 the canvas, reset and repeat.



7



Spiral

```
var myTurtle;
var startFrame; // frame number when spiral begins

function setup() {
  createCanvas(400, 400);
  background(0);
  myTurtle = makeTurtle(width / 2, height / 2);
  myTurtle.setColor(color(255, 200, 200));
  myTurtle.setWeight(2);
  myTurtle.lowerPen();
  resetCanvas();
  frameRate(10);
}
```

8



Spiral

```
function resetCanvas() {
    background(0);
    startFrame = frameCount;
    myTurtle.raisePen();
    myTurtle.goto(width / 2, height / 2);
    myTurtle.lowerPen();
}

function draw() {
    var step = (frameCount - startFrame)/30.0;
    myTurtle.forward(step);
    myTurtle.left(6.0);
    if (myTurtle.y > height) resetCanvas();
}
```

How will the spiral
change if you
increase 30.0?
Is there another
way to get this effect?

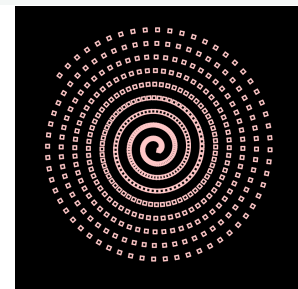
9



Spiraling Squares

```
function draw() {
    var step = (frameCount - startFrame)/30.0;
    drawSquare(myTurtle);
    myTurtle.raisePen();
    myTurtle.forward(step);
    myTurtle.lowerPen();
    myTurtle.left(6.0);
    if (myTurtle.y > height) resetCanvas();
}

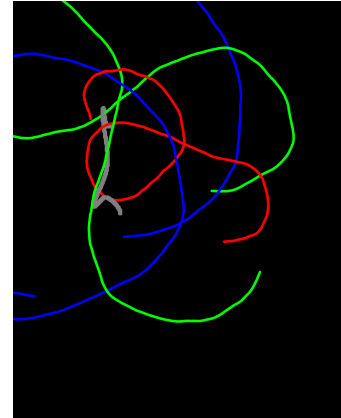
function drawSquare(t) {
    for (var i = 0; i < 4; i++) {
        t.forward(5); t.right(90);
    }
}
```



10

Turtle Race

Three Turtles (red, green and blue) are racing after a gray “target”, whose location is based on Perlin noise. The turtles “overshoot” the target because they always move forward but can only turn 1 degree per frame (plus or minus a random 5 degrees to introduce some wandering into their paths).



11

Turtle Race

```
var t1, t2, t3;      // three turtles
var x, y;            // position of target

function setup() {
  createCanvas(400, 500);
  background(0);
  t1 = makeTurtle(width/2 + random(-100, 100), height/2 + random(-100, 100));
  t2 = makeTurtle(width/2 + random(-100, 100), height/2 + random(-100, 100));
  t3 = makeTurtle(width/2 + random(-100, 100), height/2 + random(-100, 100));
  t1.lowerPen();
  t1.setColor(color(255, 0, 0));
  t1.setWeight(3);
  t2.lowerPen();
  t2.setColor(color(0, 255, 0));
  t2.setWeight(3);
  t3.lowerPen();
  t3.setColor(color(0, 0, 255));
  t3.setWeight(3);
  frameRate(40);
}
```

12

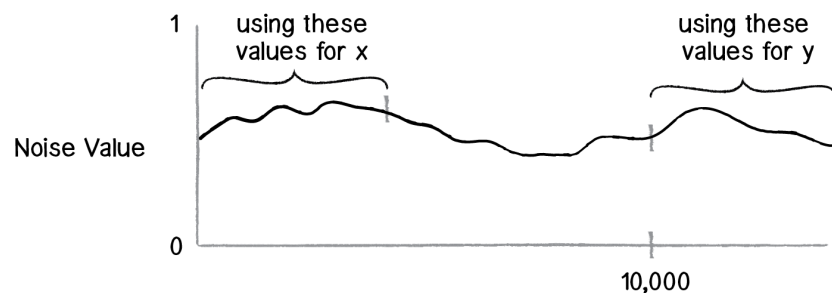
Turtle Race

```
function draw() {
  t1.forward(1);
  t2.forward(2);
  t3.forward(3);
  x = width * noise(frameCount / 1000);
  y = height * noise(10000 + frameCount / 1000);
  strokeWeight(5);
  stroke(128);
  point(x, y);
  t1.turnToward(x, y, 1);
  t2.turnToward(x, y, 1);
  t3.turnToward(x, y, 1);
  t1.left(random(-5, 5));
  t2.left(random(-5, 5));
  t3.left(random(-5, 5));
}
```

Why add 10000?
(see next slide)

13

Review: Perlin Noise



<https://www.khanacademy.org/computing/computer-programming/programming-natural-simulations/programming-noise/a/perlin-noise>

14

