Presto momento

```
float x, y;
float ballSize = 40;
float xSpeed = 5, ySpeed = 5;
float gravity = 0.1;
float braking = 0.09;

// keep setup() the same still

void draw() {

    // first lines of draw() are the same too

    y = y + ySpeed;
    x = x + xSpeed;
    ySpeed = ySpeed + gravity;

    if (y + ballSize/2 > height) {
        ySpeed *= -(1 - braking);

        y = height - ballSize/2;
    }

    if (x + ballSize/2 >= width) {

        xSpeed *= -(1 - braking);
        x = width - ballSize/2;

    } else if (x - ballSize/2 <= 0) {

        xSpeed *= -(1 - braking);
        x = ballSize/2;
    }
}
```

To the vector go the spoils

~~float x, y;~~
~~float xSpeed = 5, ySpeed = 5;~~

PVector location, velocity;

(Instead of a bunch of floats, we now just have two PVector variables.)

To the vector go the spoils

```
float ballSize = 40;
PVector location, velocity;
float gravity = 0.1;
float braking = 0.09;

void setup() {

    size(640, 240);
    location = new PVector(width/2, ballSize * 1.5);
    velocity = new PVector(2.5, 5);
}

void draw() {

    background(#92CFED);
    fill(216, 7, 21);
    ellipse(location.x, location.y, ballSize,
ballSize);

    // location = location + velocity;
    location.add(velocity);
    velocity.y += gravity;

    if ((location.x > width) || (location.x < 0)) {
        velocity.x = velocity.x * -1;
    }
    if ((location.y > height) || (location.y < 0)) {
        velocity.y = velocity.y * -1;
    }
```

```
    if (location.y + ballSize/2 > height) {
        velocity.y *= -(1 - braking);

        location.y = height - ballSize/2;
    }

    if (location.x + ballSize/2 >= width ||
location.x - ballSize/2 <= 0) {

        velocity.x *= -(1 - braking);
        velocity.y *= (1 - braking);
    }

    if (location.x > width + ballSize/2) {
        location.x = width - ballSize/2;
    } else if (location.x < ballSize/2) {
        location.x = ballSize/2;
    }
}
```

What IS a vector, really?

```
Vector {
  float x;
  float y;
}
```

What IS a vector, really?

```
Vector {
  float x;
  float y;

  Vector(inX, inY) {
     x = inX;
     y = inY;
  }
}
```

What IS a vector, really?

```
Vector {
  float x;
  float y;

  Vector(inX, inY) {
     x = inX;
     y = inY;
  }

  add(Vector otherVector) {
    x = x + otherVector.x;
    y = y + otherVector.y;
  }
}
```

Declaration of… int-dependence?

```
// Variable Declaration
int var;  // type name


// Variable Initialization
var = 10;  // var equals 10


// Object Initialization
myCar = new Car();
// The ‘new’ operator is used to make a new object.
```

It's always cars. Why always cars?

```
// Step 1. Declare an object.
Car myCar;

void setup()  {
  // Step 2. Initialise object.
  myCar = new Car();
}

void draw()  {
  background(255);
  // Step 3. Call methods on the object.
  myCar.drive();
  myCar.display();
}
```

```
// Simple non OOP Car
```

```
color c;
float xpos;
float ypos
float xspeed;
```

```
void setup() {
  size(200,200);
  c = color(255);
  xpos = width/2;
  ypos = height/2;
  xspeed = 1;
}
```

```
void draw() {
  background(0);
  display();
  drive();
}
```

```
void display () {
  rectMode(CENTER);
  fill(c);
  rect(xpos,ypos,20,10);
}
```

```
void drive() {
  xpos = xpos + xspeed;
  if (xpos > width) {
    xpos = 0;
  }
}
```

→

```
class Car {
  color c;
  float xpos;
  float ypos;
  float xspeed;
```

→ The class name

→ Data

```
  Car() {
    c = color(255);
    xpos = width/2;
    ypos = height/2;
    xspeed = 1;
  }
```

→ Constructor

```
  void display() {
    rectMode(CENTER);
    fill(c);
    rect(xpos,ypos,20,10);
  }

  void drive() {
    xpos = xpos + xspeed;
    if (xpos > width) {
      xpos = 0;
    }
  }
}
```

→ Functionality

Ultra classy

```
class Ball {

    PVector location, velocity;
    int ballSize = 40;
    float gravity = 0.1;
    float braking = 0.1;

    void Ball(inX, inY, inVelX, inVelY) {
        location = new PVector(inX, inY);
        velocity = new PVector(inVelX, inVelY);
    }

    void update() { }

    void move() { }

    boolean shouldBounce() { }

    void bounce() { }

    void draw() { }
}
```

Ultra classy

```
Ball bouncy;

void setup() {

    size(640, 240);
    bouncy = new Ball(width/2, 60, 2.5, 5);
}

void draw() {

    background(#92CFED);
    bouncy.update();
}
```

Ultra classy

```
class Ball {

    PVector location, velocity;
    int ballSize = 40;
    float gravity = 0.1;
    float braking = 0.1;

    Ball(float inX, float inY, float inVelX,
float inVelY) {
        location = new PVector(inX, inY);
        velocity = new PVector(inVelX, inVelY);
    }

    void update() {

        move();

        if (shouldBounceX()) {
            bounceX();
        }
        if (shouldBounceY()) {
            bounceY();
        }

        draw();
    }

    void move() {
        location.add(velocity);
        velocity.y += gravity;
    }

    boolean shouldBounceX() {
        return ((location.x > width) ||
(location.x < 0));
    }

    boolean shouldBounceY() {
        return ((location.y > height) ||
(location.y < 0));
    }

    void bounceX() {

        velocity.x = velocity.x * -1;

        if (location.x < 0) {
            location.x = 0;
        } else if (location.x > width) {
            location.x = width;
        }
    }

    void bounceY() {
        velocity.y *= -(1 - braking);
        location.y = height - ballSize/2;
    }

    void draw() {
        fill(216, 7, 21);
        ellipse(location.x, location.y,
ballSize, ballSize);
    }
}
```