PAT 204/504 (Fall 2025)

Creative Coding

Lecture 6: Objects

Instructor: Hao-Wen Dong

UNIVERSITY OF MICHIGAN

Objects

What is an Object?

- **Variables** store **data**
- **Functions** provide **functionality**
- **Objects** **store data** and **provide functionality** within its scope

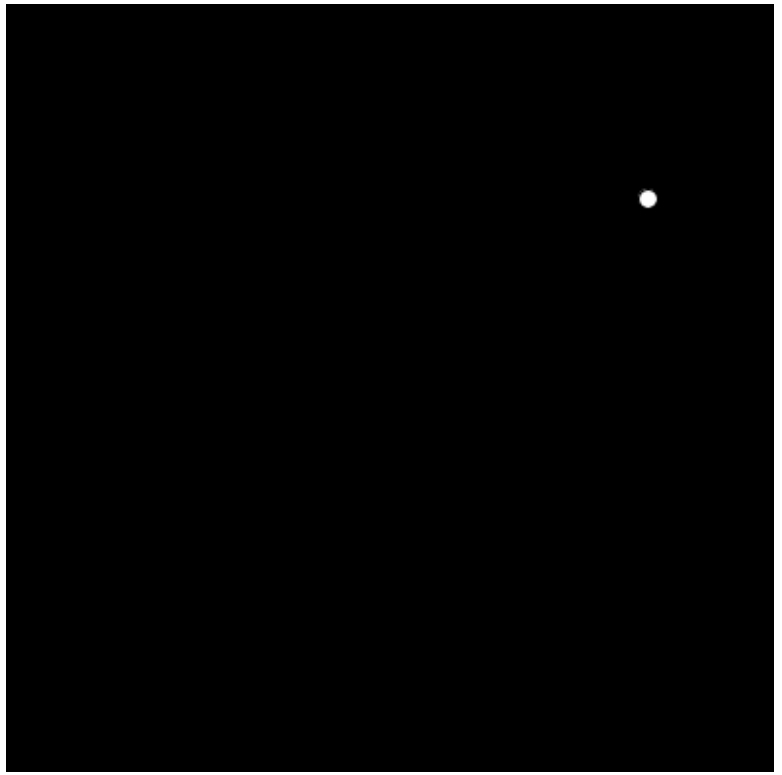
Example: Bouncing Ball

```
class Ball {
  float x, y;
  float size;
  float speedX, speedY;

  Ball() {
    // Constructor
  }

  void show() {
    // Show the ball
  }

  void move() {
    // Move the ball
  }

  void checkWalls() {
    // Check if the ball hit the walls
  }
}
```

Fields

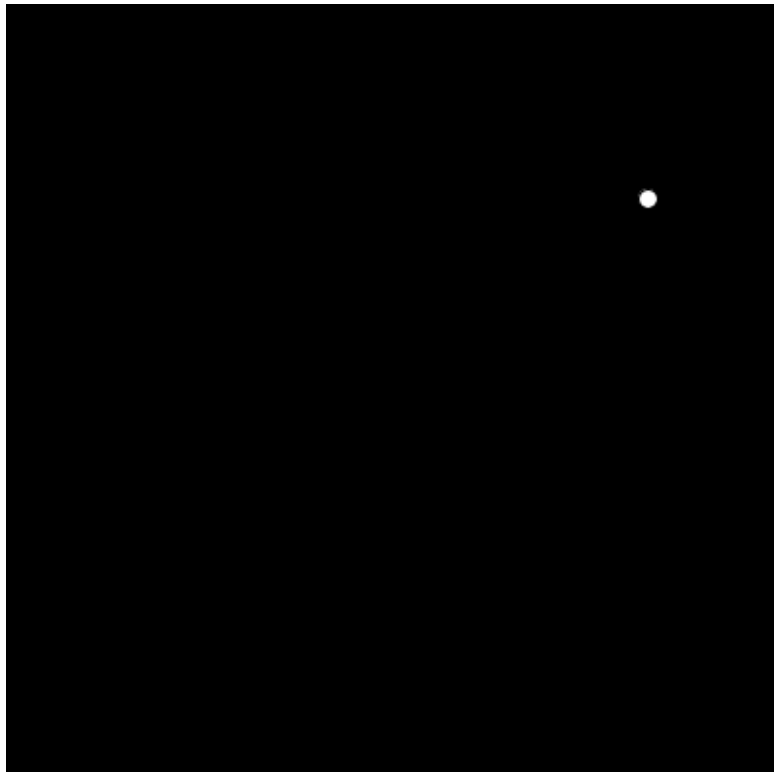
Constructor

Methods



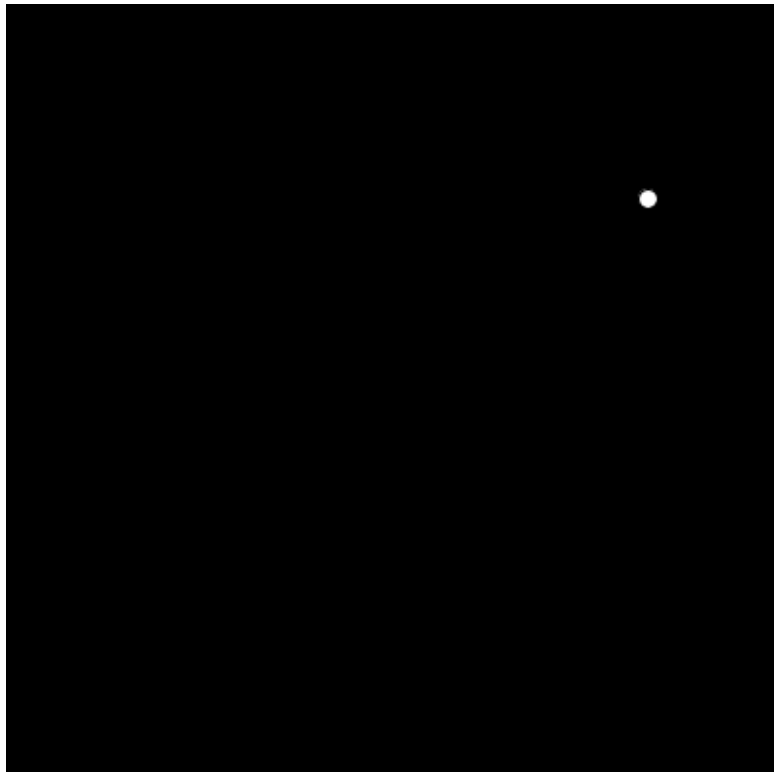
Example: Bouncing Ball

```
class Ball {
  float x, y;
  float size;
  float speedX, speedY;

  // Constructor
  Ball() {
    x = 200;
    y = 200;
    size = 10;
    speedX = 3;
    speedY = 3;
  }

  ...
}
```



Example: Bouncing Ball

```
class Ball {
  float size = 10;
  float speed = 5;
  float x, y, speedX, speedY;

  Ball() {
    ...
  }

  // Methods
  void show() {
    circle(x, y, size);
  }

  void move() {
    x += speedX;
    y += speedY;
  }

  ...
}
```

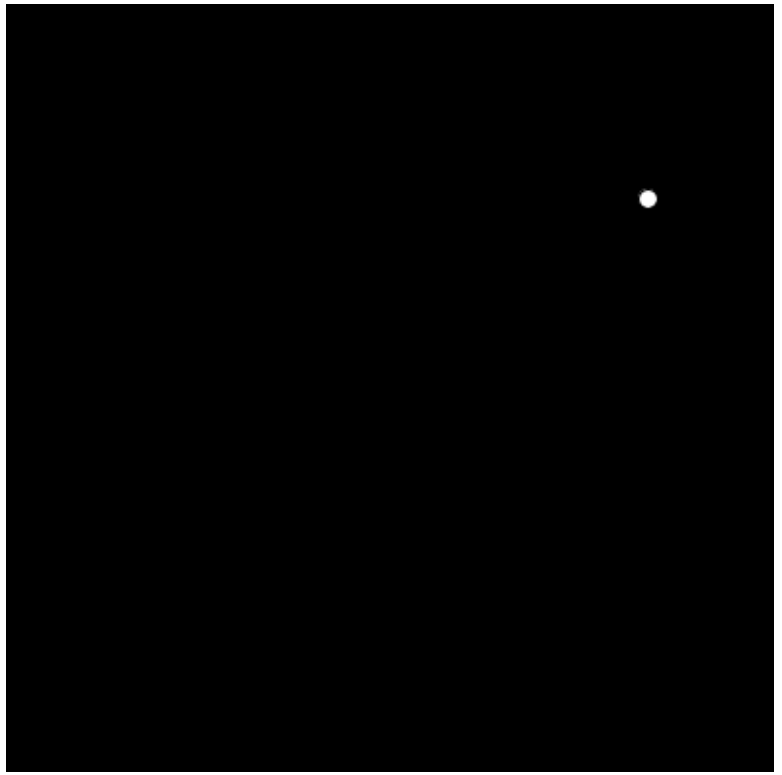


Example: Bouncing Ball

```
class Ball {
  ...

  void checkWalls() {
    float radius = size / 2;

    if (x > width - radius || x < radius) {
      speedX = -speedX;
    }

    if (y > height - radius || y < radius) {
      speedY = -speedY;
    }
  }

  ...
}
```

Check if the ball hit the left and right walls

Check if the ball hit the left and right walls



Example: Bouncing Ball

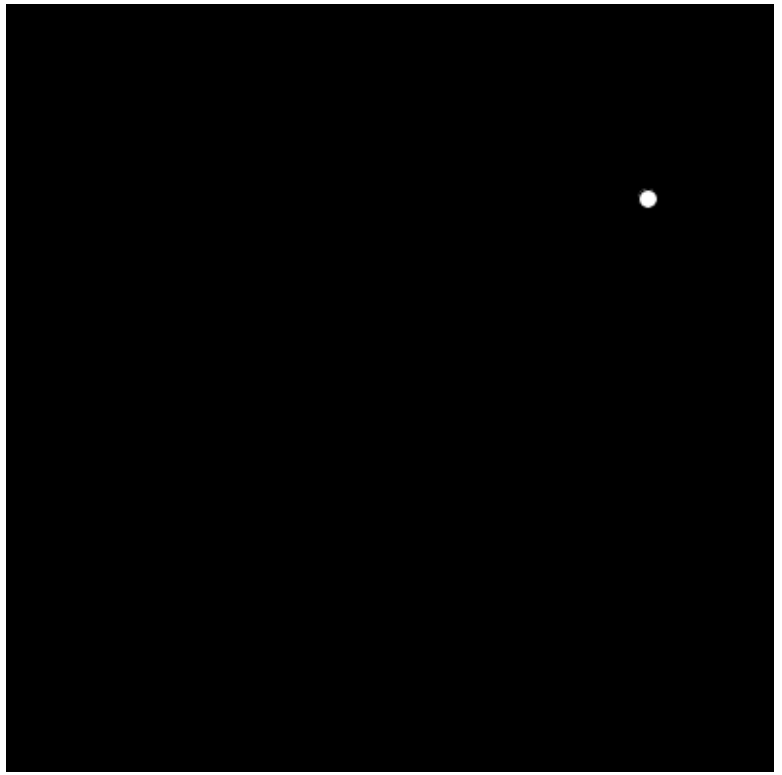
```
Ball ball;
```

Declaration

```
void setup() {
  size(400, 400);

  ball = new Ball();
}

void draw() {
  background(0);

  ball.move();
  ball.checkWalls();
  ball.show();
}
```

Initialization

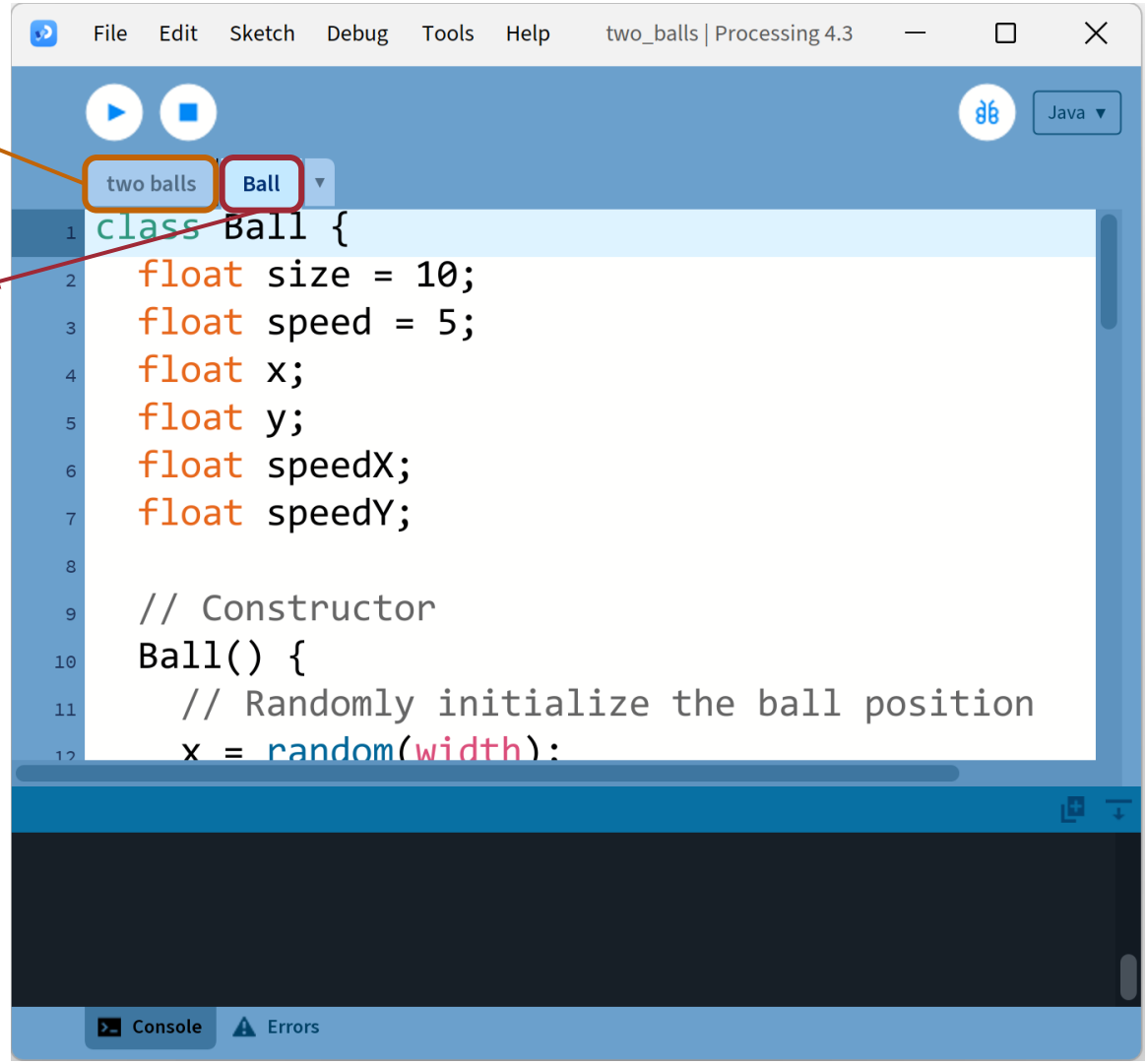
Call the methods!



Standalone Files for Classes

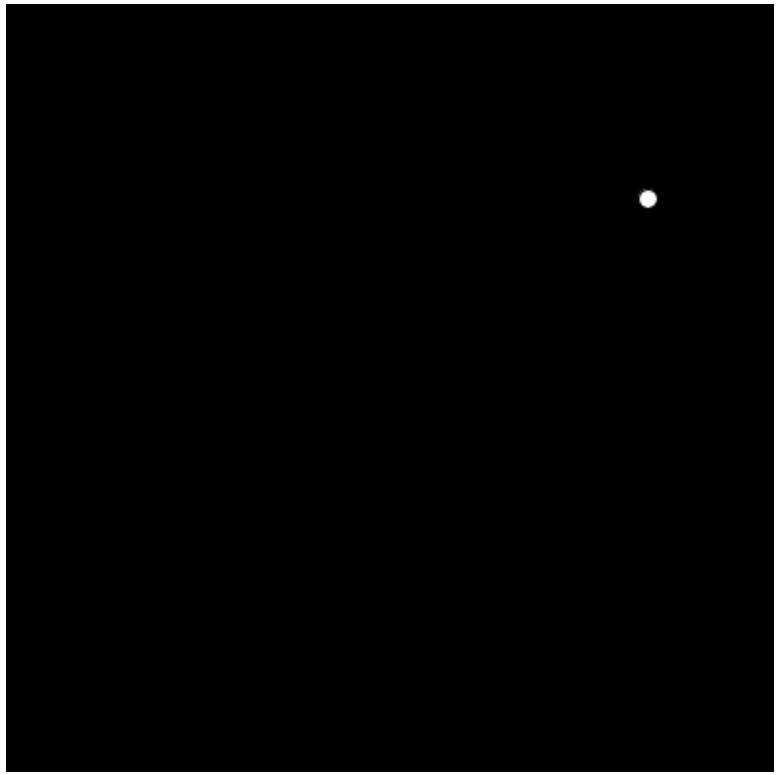
Main file

Define your class



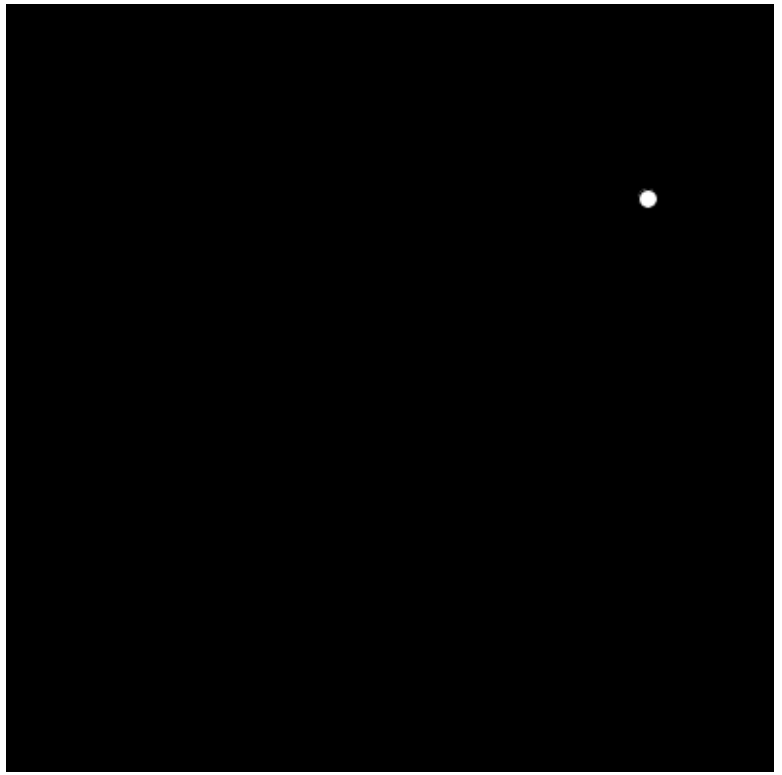
Example: Bouncing Ball

```
class Ball {
  float x, y;
  float size;
  float speedX, speedY;

  // Constructor
  Ball(float x, float y, float size,
       float speedX, float speedY) {
    this.x = x;
    this.y = y;
    this.size = size;
    this.speedX = speedX;
    this.speedY = speedY;
  }

  ...
}
```



Example: Bouncing Ball

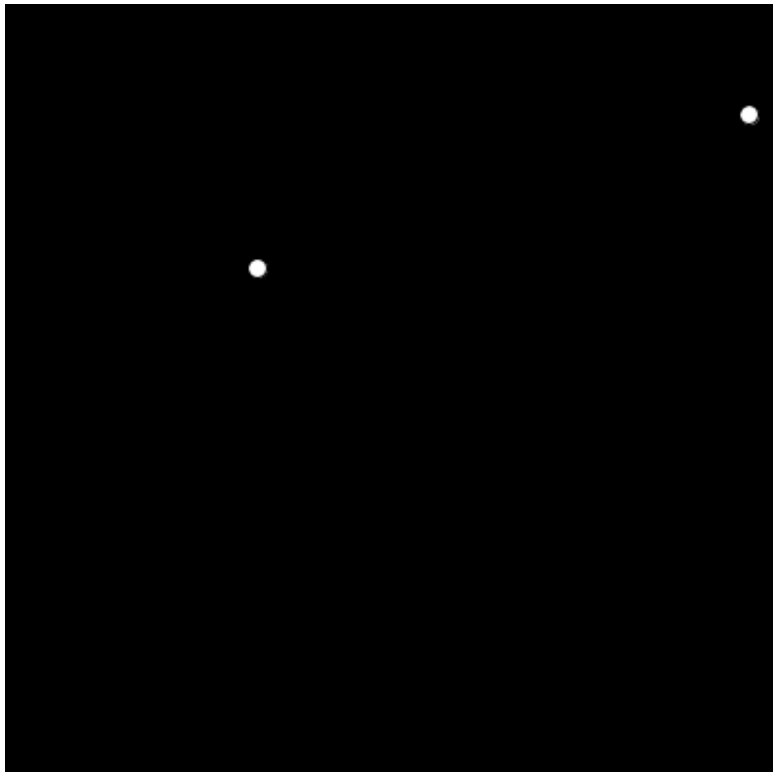
```
class Ball {
  float size = 10;
  float speed = 5;
  float x, y, speedX, speedY;

  // Constructor
  Ball() {
    // Randomly initialize the ball position
    x = random(width);
    y = random(height);

    // Randomly initialize the ball speed
    float theta = random(0, TWO_PI);
    speedX = speed * cos(theta);
    speedY = speed * sin(theta);
  }

  ...
}
```



🧐 **Exercise**: Two Bouncing Balls

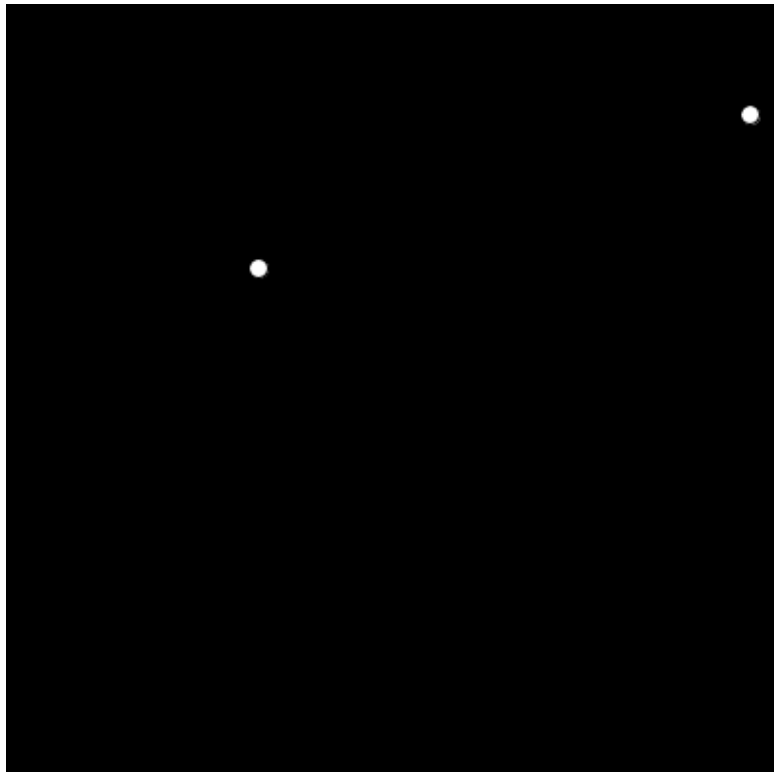
- Find the Ball class definition in the link
- Create **two** bouncing balls

Ball Class



🧐 **Exercise**: Two Bouncing Balls

```
Ball ball, ball2;

void setup() {
  size(400, 400);

  ball = new Ball();
  ball2 = new Ball();
}

void draw() {
  background(0);

  ball.move();
  ball.checkWalls();
  ball.show();

  ball2.move();
  ball2.checkWalls();
  ball2.show();
}
```



Example: Bouncing Balls

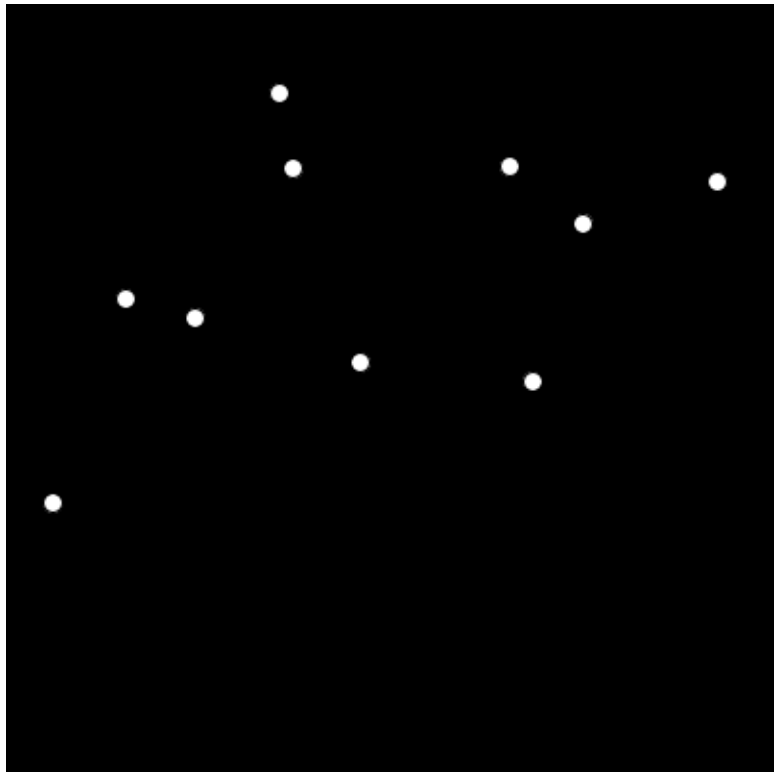
```
Ball[] balls = new Ball[20];
```

An array of objects

```
void setup() {
  size(400, 400);

  for (int i = 0; i < balls.length; i++) {
    balls[i] = new Ball();
  }
}

void draw() {
  background(0);

  for (int i = 0; i < balls.length; i++) {
    balls[i].move();
    balls[i].checkWalls();
    balls[i].show();
  }
}
```

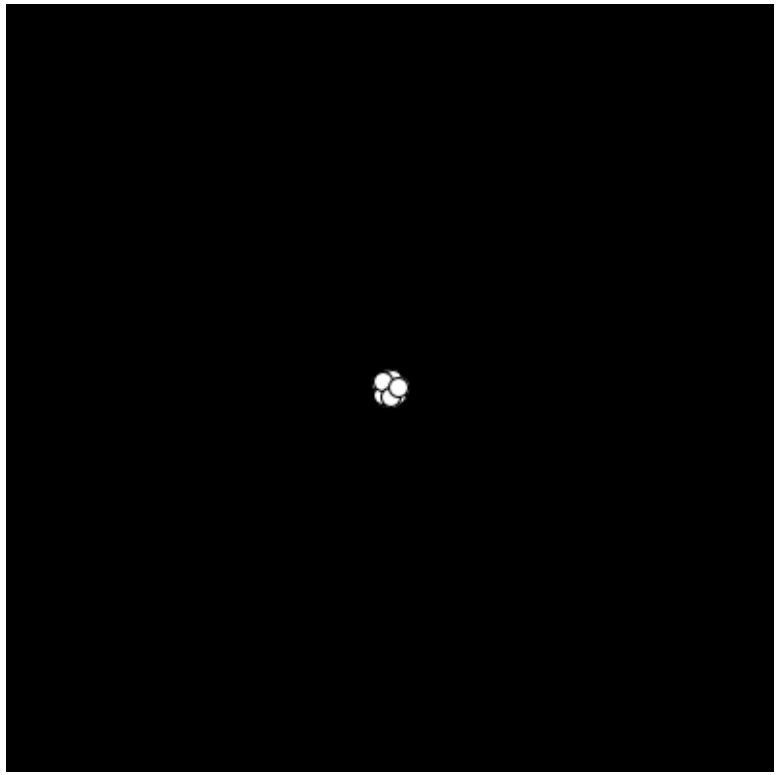
Initialization

Call the methods!



Signature Polymorphism

```
class Ball {
  float size = 10;
  float speed = 5;
  float x, y, speedX, speedY;

  Ball() {
    x = random(width);
    y = random(height);

    float theta = random(0, TWO_PI);
    speedX = speed * cos(theta);
    speedY = speed * sin(theta);
  }

  Ball(float x, float y) {
    this.x = x;
    this.y = y;

    float theta = random(0, TWO_PI);
    speedX = speed * cos(theta);
    speedY = speed * sin(theta);
  }
}
```



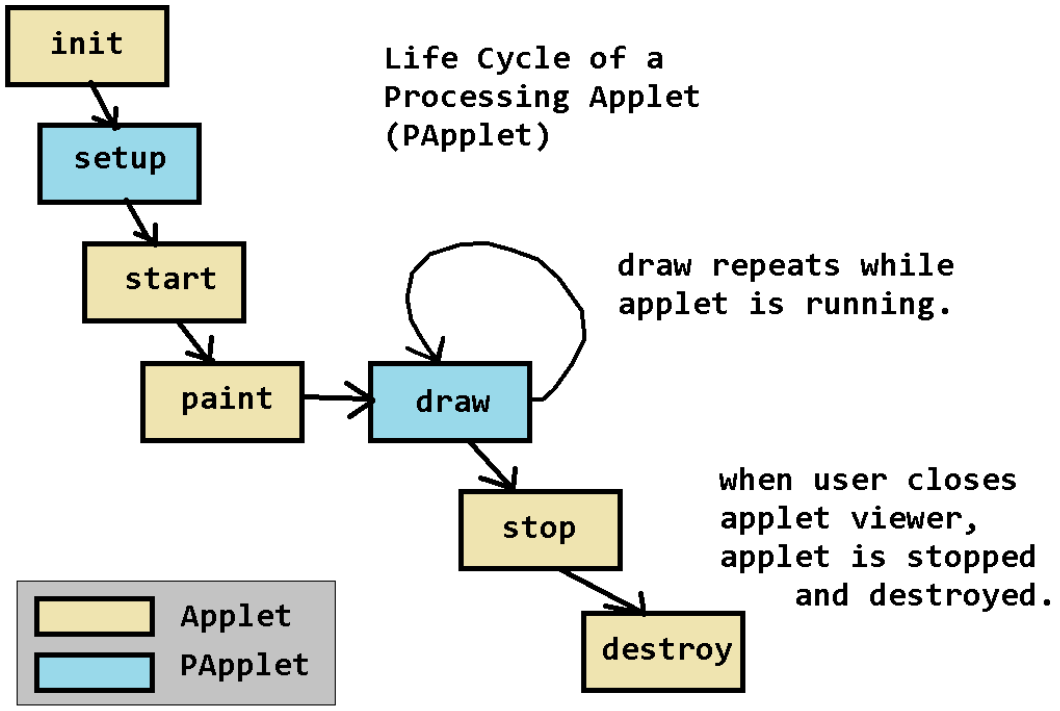
Why Objects?

- **Organization**
  - Naturally organized into files or blocks
- **Re-usability**
  - A well-written class can be reused in many projects (e.g., FFT, PImage, PVector)
- **Ease of maintenance**
  - Each team member can work on different part of the code without less conflicts
- **Abstraction & Encapsulation**
  - What does FFT do internally? **Do we really need to know every detail?**
  - **Define the interface** rather than exposing everything

Some Example Classes

A Processing Scratch is an Object in Java

- A processing scratch creates an **PApplet** object in Java
- NOT directly accessible in Processing



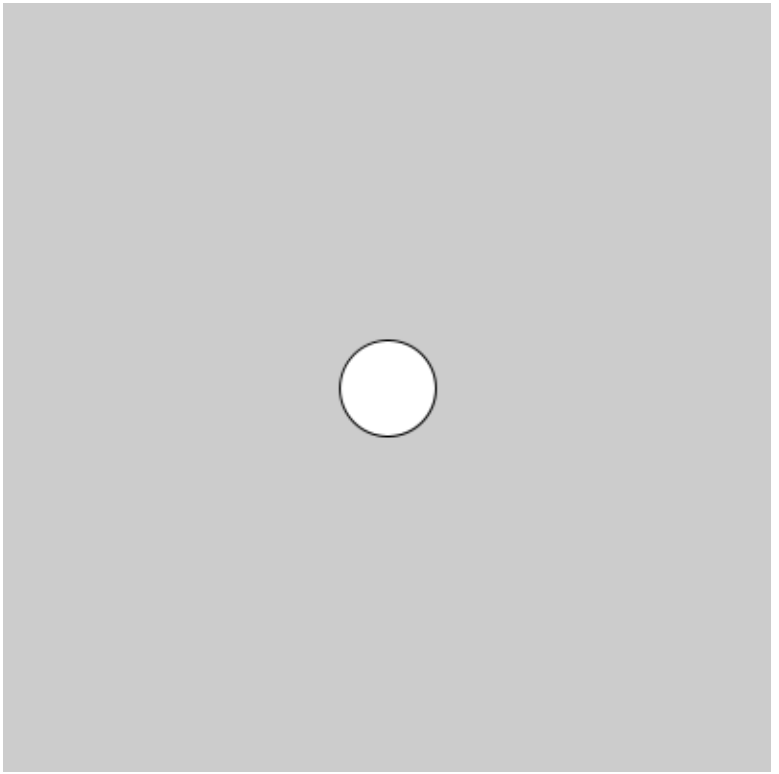
(Deb Deppeler, 2015)

pages.cs.wisc.edu/~cs302/labs/EclipseTutorial_Processing/Step_04.html

PVector Class

- A class for 2D or 3D vectors in the Euclidean space

```
PVector pos = new PVector(200, 200);

void setup() {
  size(400, 400);
  noLoop();
}

void draw() {
  circle(pos.x, pos.y, 50);
}
```

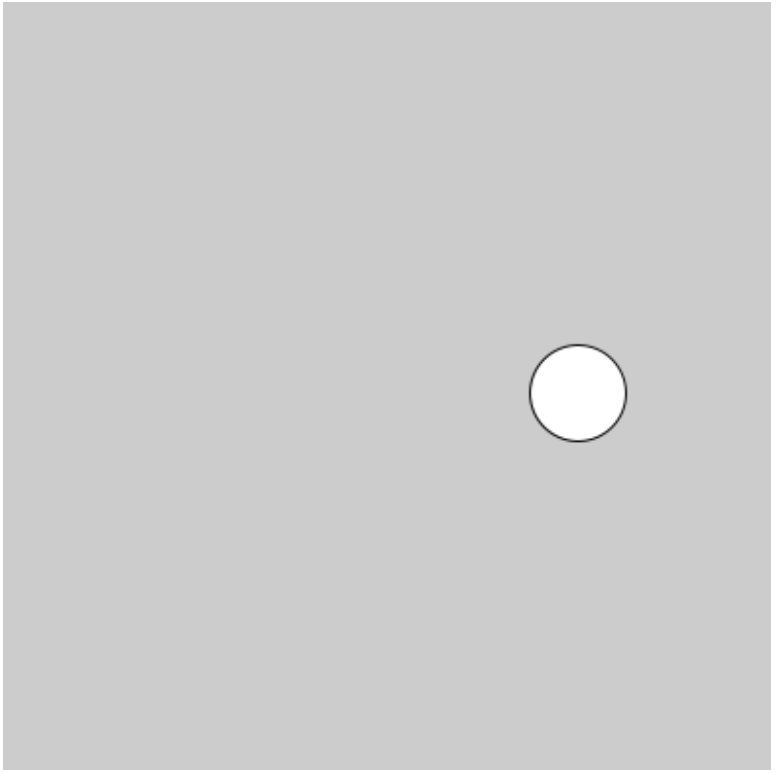


`PVector` Methods

- **`PVector`** offers many handy methods
  - Analysis **`mag, heading, dist`**
  - Manipulation **`rotate, setMag, normalize, limit`**
  - Operations **`add, sub, mult, div`**
  - Vector operations **`dot, cross`**

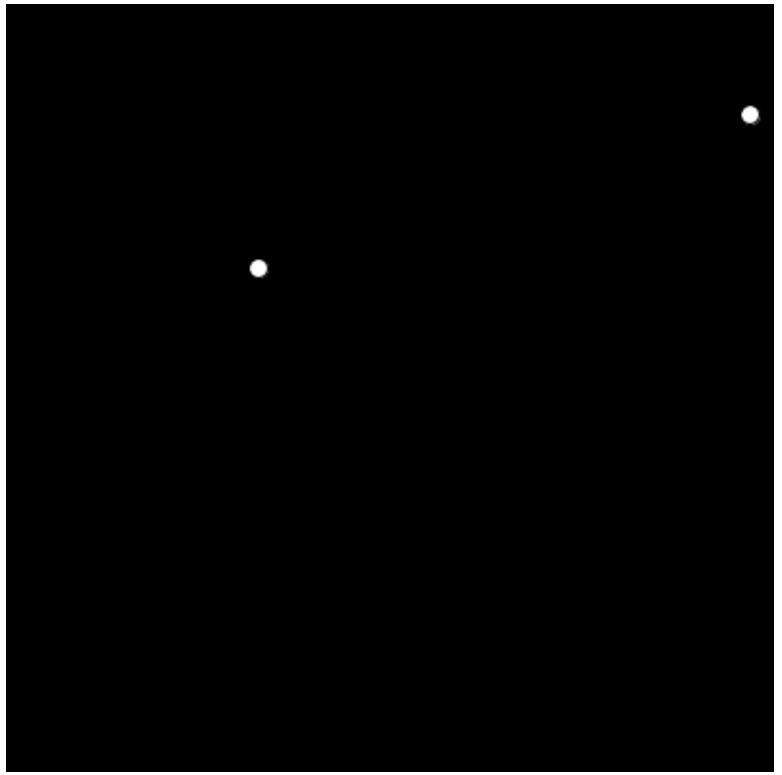
Example: Rotating Ball

```
PVector vec = new PVector(100, 0);

void setup() {
  size(400, 400);
}

void draw() {
  // Rotate the vector by a fixed angle
  vec.rotate(PI * 0.01);

  // Draw the circle
  circle(200 + vec.x, 200 + vec.y, 50);
}
```



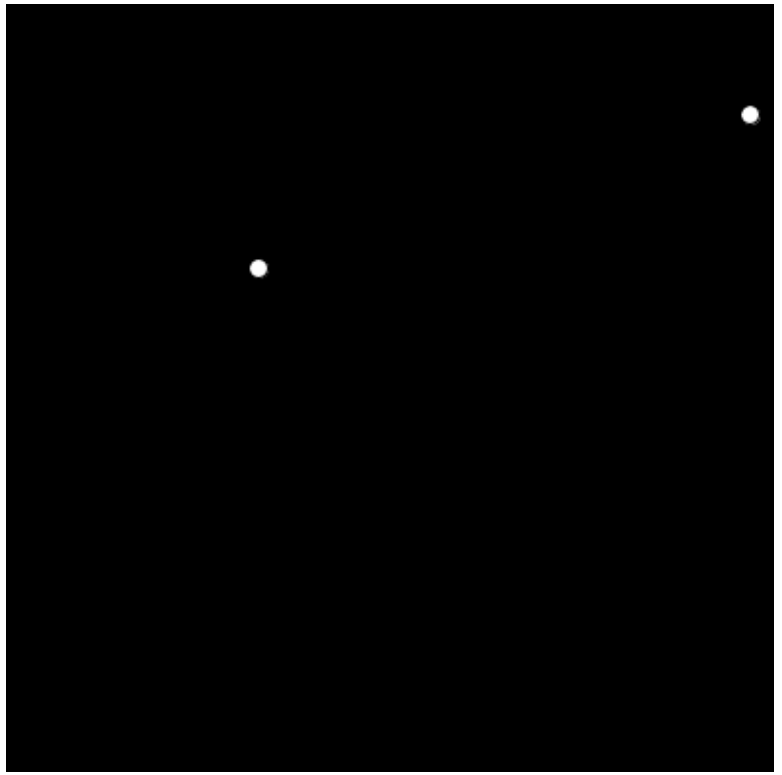
🧐 **Exercise**: Rewrite the `Ball` Class using `PVector`

```
class Ball {
  float size = 10;
  float speed = 5;
  float x, y, speedX, speedY;

Ball() {
    // Randomly initialize the ball position
    x = random(width);
    y = random(height);

    // Randomly initialize the ball speed
    float theta = random(0, TWO_PI);
    speedX = speed * cos(theta);
    speedY = speed * sin(theta);
  }

  ...
}
```



🧐 **Exercise**: Rewrite the `Ball` Class using `PVector`

```
class Ball {
  float size = 10;
  float speed = 5;
  PVector pos = new PVector();
  PVector vel = new PVector();

  Ball() {
    // Randomly initialize the ball position
    pos.x = random(width);
    pos.y = random(height);

    // Randomly initialize the ball speed
    float theta = random(0, TWO_PI);
    vel.x = speed * cos(theta);
    vel.y = speed * sin(theta);
  }

  ...
}
```



PVector Static Methods

- **Static methods** are methods that belong to a class (rather than an instance)
  - `PVector.random2D` Create a 2D unit vector with a **random direction**
  - `PVector.random3D` Create a 3D unit vector with a **random direction**
  - `PVector.fromAngle` Create a 2D unit vector with the **specified direction**

Instance method

```
PVector v = new PVector(1, 0);
v.rotate(PI / 4);
println(v);
```

Static method

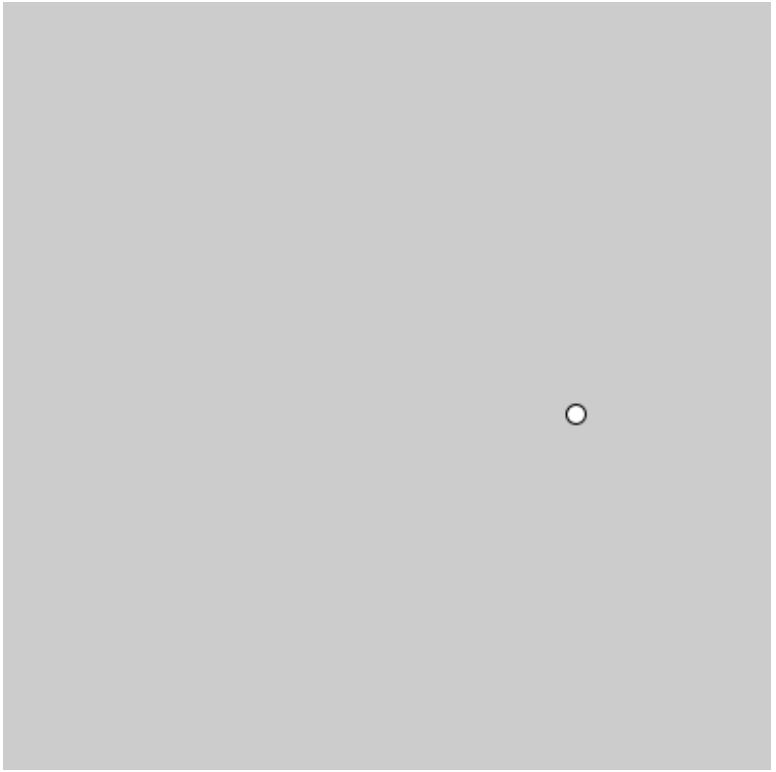
```
PVector v = PVector.fromAngle(PI / 4);
println(v);
```

Example: PVector.random2d

```
PVector pos = new PVector();

void setup() {
  size(400, 400);
  frameRate(30);
}

void draw() {
  pos = PVector.random2D().mult(100);
  circle(200 + pos.x, 200 + pos.y, 10);
}
```

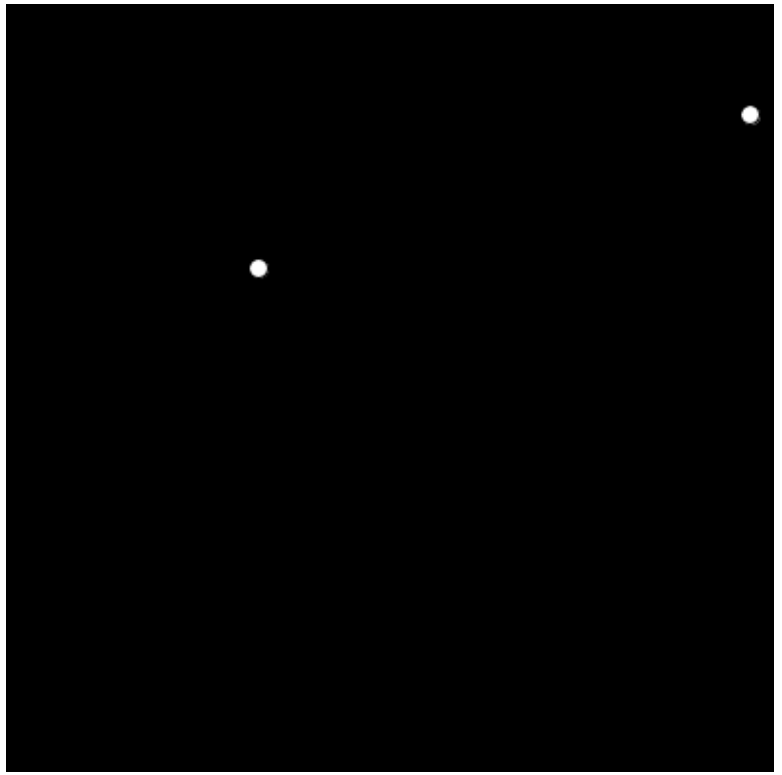
Get a random 2D unit vector

Scale the vector by 100



🧐 **Exercise**: Rewrite `Ball` Class using `Pvector.random2d`

```
class Ball {
  float size = 10;
  float speed = 5;
  PVector pos = new PVector();
  PVector vel = new PVector();

  Ball() {
    // Randomly initialize the ball position
    pos.x = random(width);
    pos.y = random(height);

    // Randomly initialize the ball speed
    float theta = random(0, TWO_PI);
    vel.x = speed * cos(theta);
    vel.y = speed * sin(theta);
  }

  ...
}
```

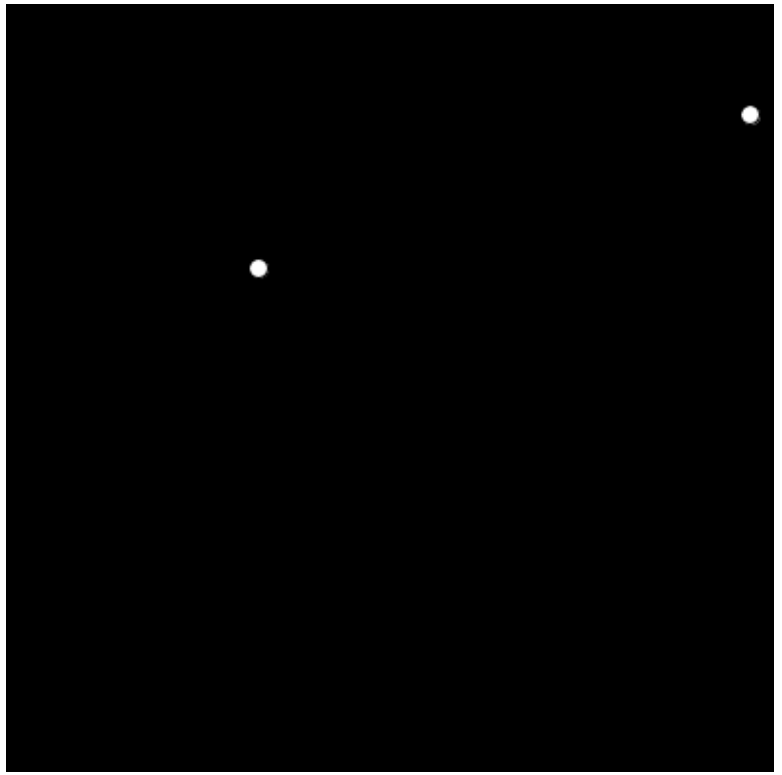


🧐 **Exercise**: Rewrite `Ball` Class using `Pvector.random2d`

```
class Ball {
  float size = 10;
  float speed = 5;
  PVector pos = new PVector();
  PVector vel = new PVector();

  Ball() {
    // Randomly initialize the ball position
    pos.x = random(width);
    pos.y = random(height);

    // Randomly initialize the ball speed
    vel = PVector.random2D().mult(speed);
  }

  ...
}
```

Get a random 2D unit vector

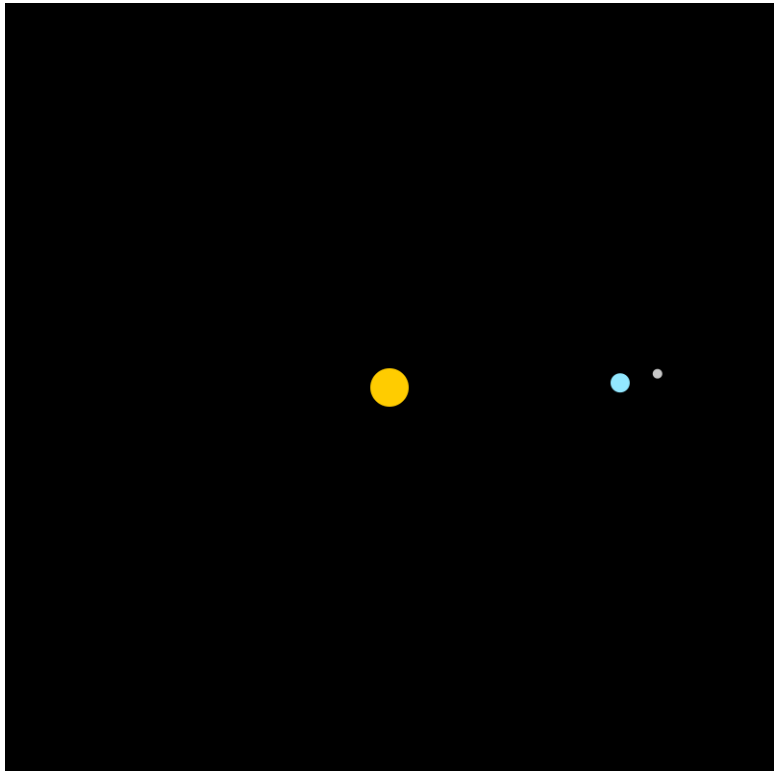
Scale the vector by speed



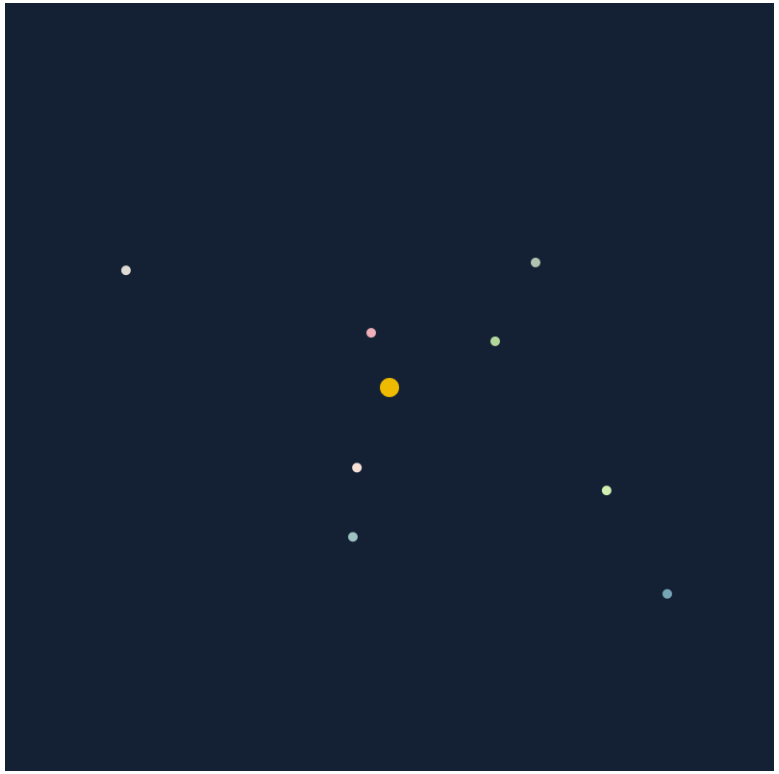
🔥 **HW 3**: Solar System

- Instructions will be sent by **emails** and released on the **course website**

Q1: Sun, Earth, and Moon



Q2: Solar System



Recap

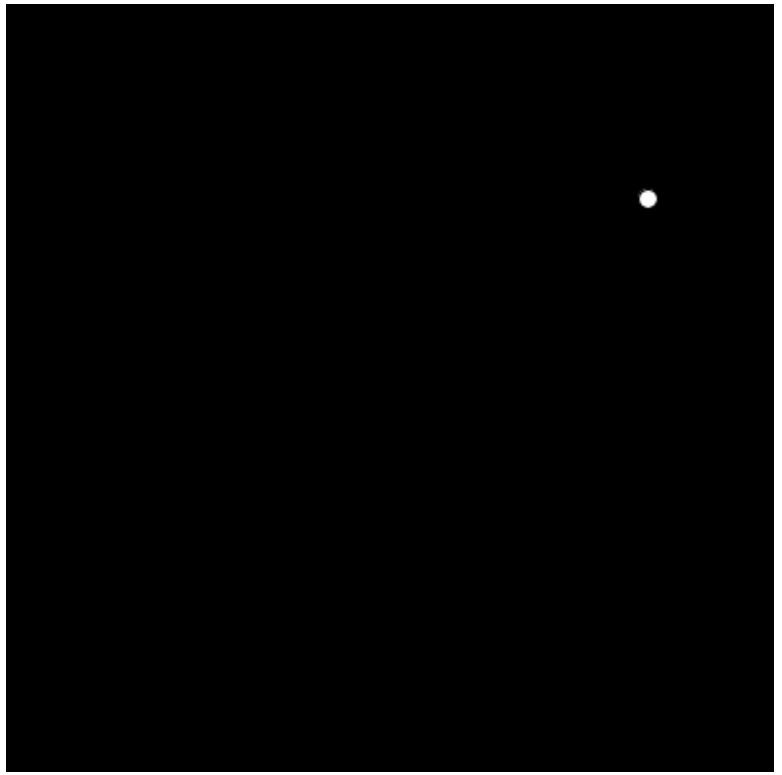
Example: Bouncing Ball

```
class Ball {
  float x, y;
  float size;
  float speedX, speedY;

  Ball() {
    // Constructor
  }

  void show() {
    // Show the ball
  }

  void move() {
    // Move the ball
  }

  void checkWalls() {
    // Check if the ball hit the walls
  }
}
```

Fields

Constructor

Methods



Example: Bouncing Ball

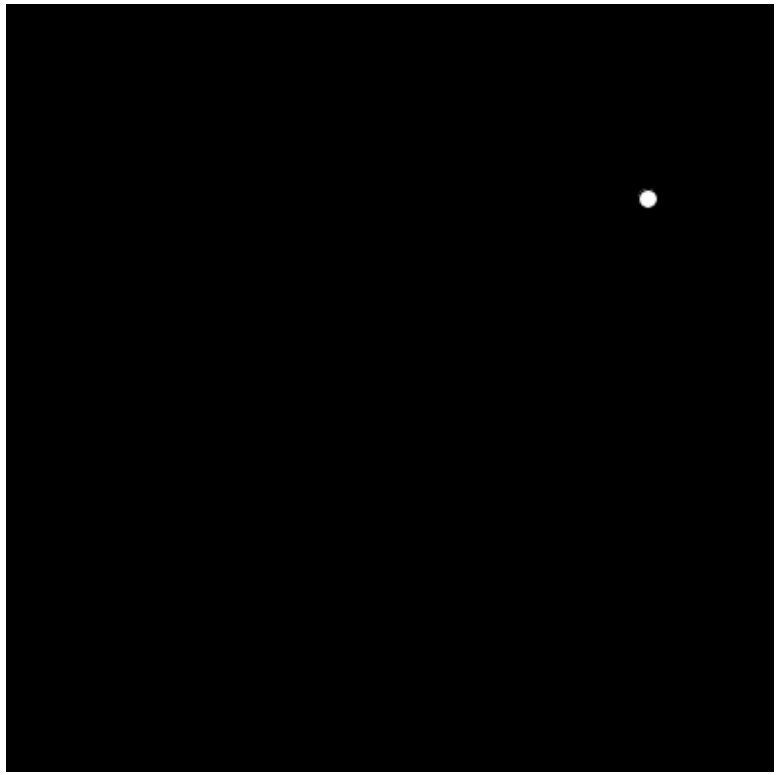
```
Ball ball;
```

Declaration

```
void setup() {
  size(400, 400);

  ball = new Ball();
}

void draw() {
  background(0);

  ball.move();
  ball.checkWalls();
  ball.show();
}
```

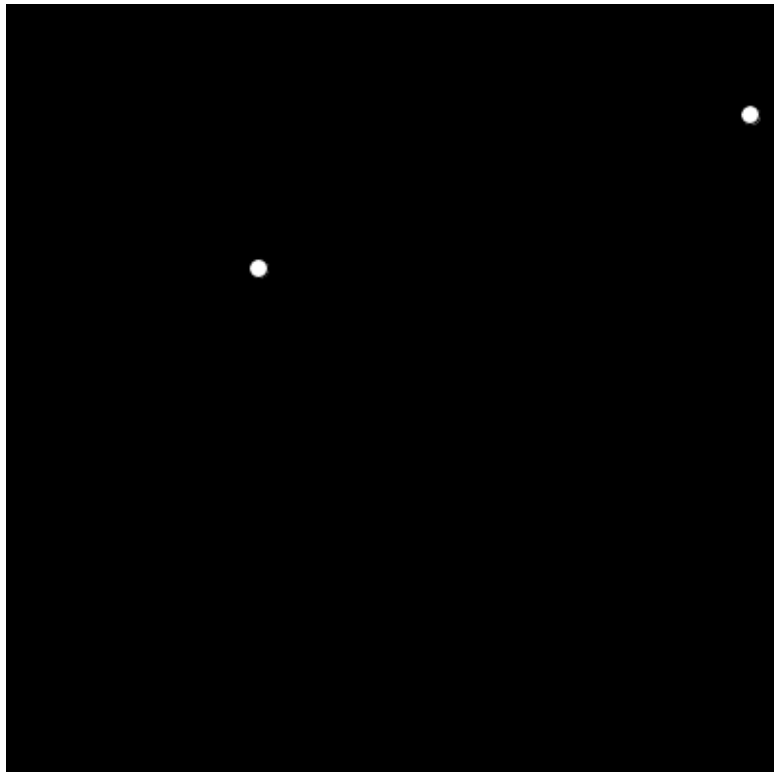
Initialization

Call the methods!



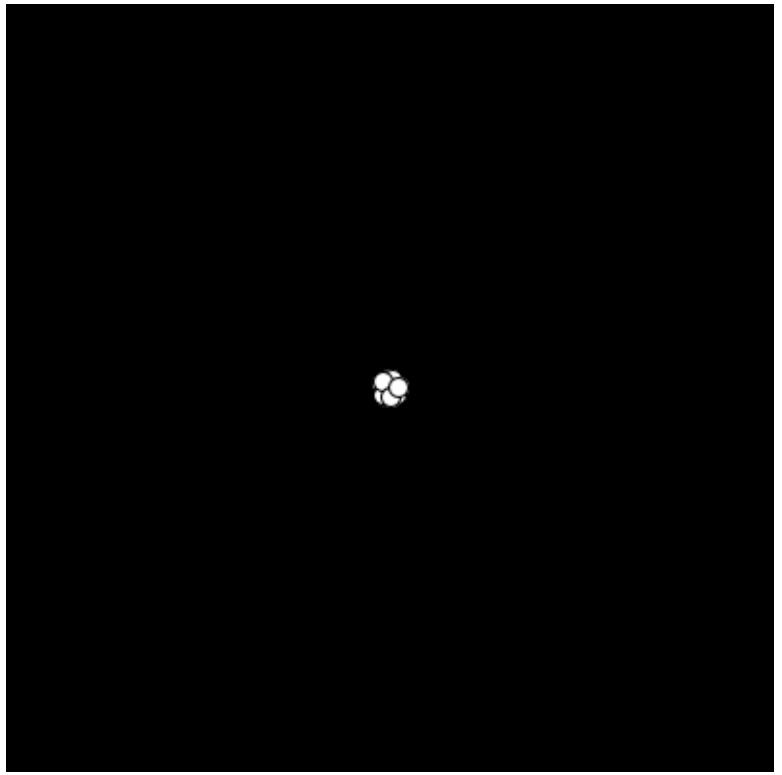
🧐 **Exercise**: Two Bouncing Balls

```
Ball ball, ball2;

void setup() {
  size(400, 400);

  ball = new Ball();
  ball2 = new Ball();
}

void draw() {
  background(0);

  ball.move();
  ball.checkWalls();
  ball.show();

  ball2.move();
  ball2.checkWalls();
  ball2.show();
}
```



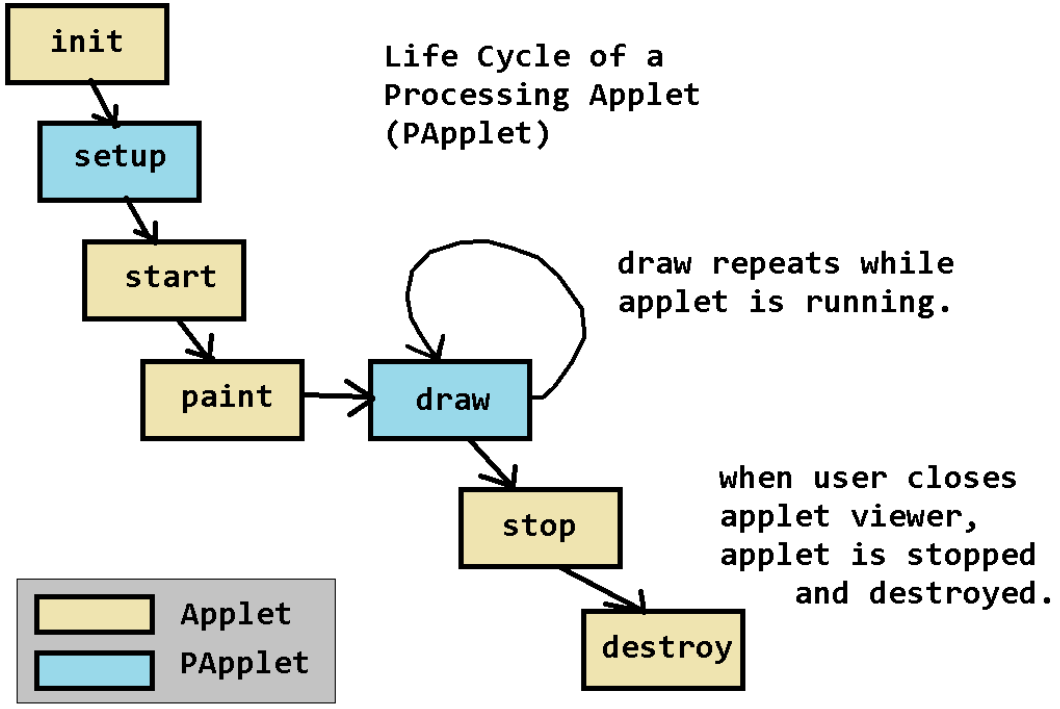
Signature Polymorphism

```
class Ball {
  float size = 10;
  float speed = 5;
  float x, y, speedX, speedY;

  Ball() {
    x = random(width);
    y = random(height);

    float theta = random(0, TWO_PI);
    speedX = speed * cos(theta);
    speedY = speed * sin(theta);
  }

  Ball(float x, float y) {
    this.x = x;
    this.y = y;

    float theta = random(0, TWO_PI);
    speedX = speed * cos(theta);
    speedY = speed * sin(theta);
  }
}
```



A Processing Scratch is an Object in Java

- A processing scratch creates an **PApplet** object in Java
- NOT directly accessible in Processing

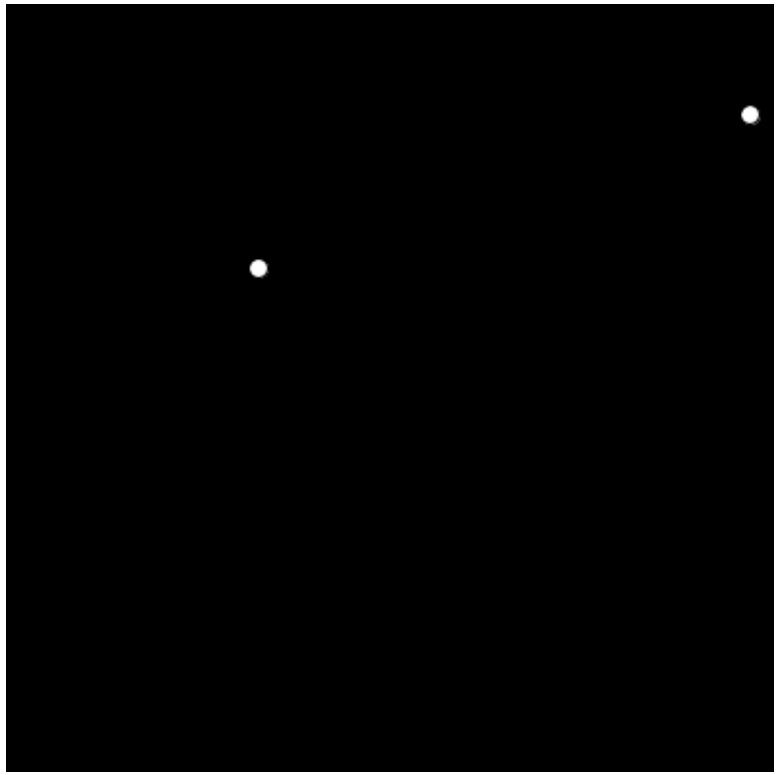


(Deb Deppeler, 2015)

pages.cs.wisc.edu/~cs302/labs/EclipseTutorial_Processing/Step_04.html

🧐 **Exercise**: Rewrite the `Ball` Class using `PVector`

```
class Ball {
  float size = 10;
  float speed = 5;
  PVector pos = new PVector();
  PVector vel = new PVector();

  Ball() {
    // Randomly initialize the ball position
    pos.x = random(width);
    pos.y = random(height);

    // Randomly initialize the ball speed
    float theta = random(0, TWO_PI);
    vel.x = speed * cos(theta);
    vel.y = speed * sin(theta);
  }

  ...
}
```

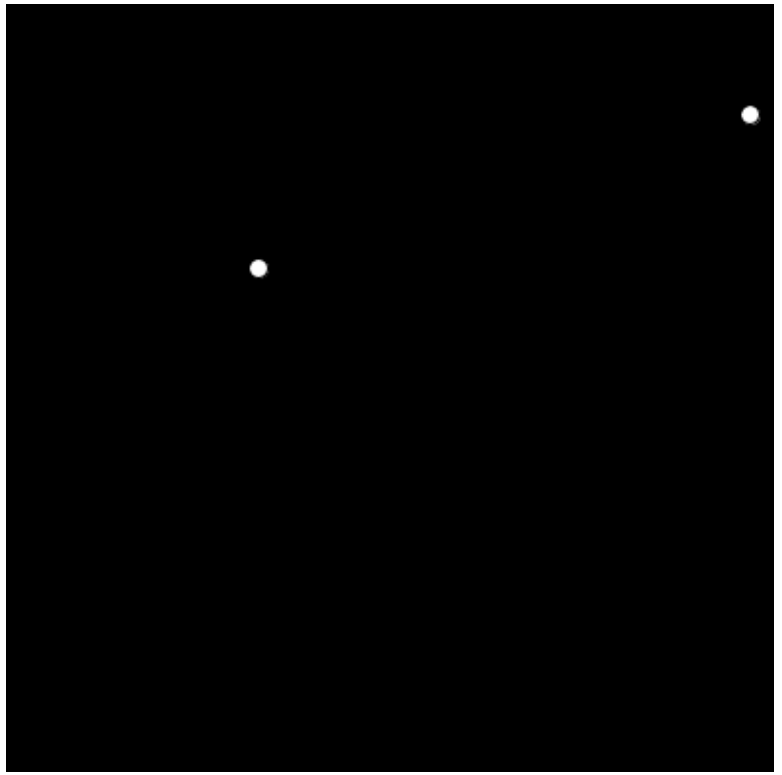


🧐 **Exercise**: Rewrite `Ball` Class using `Pvector.random2d`

```
class Ball {
  float size = 10;
  float speed = 5;
  PVector pos = new PVector();
  PVector vel = new PVector();

  Ball() {
    // Randomly initialize the ball position
    pos.x = random(width);
    pos.y = random(height);

    // Randomly initialize the ball speed
    vel = PVector.random2D().mult(speed);
  }

  ...
}
```

Get a random 2D unit vector

Scale the vector by speed



Next Lecture

Lists & Data I/O