Introduction to Programming

Class 36, 9 May 2017

SYNC

Sit where you like.

Introduction to Programming
Class 36, 9 May 2017

Goals

Goal 1: You will understand how variables are organized and grouped in objects.
Goal 2: You will know how to create and use objects.

Vocabulary

Objects
Properties
Methods
Dot Notation

Code

object.property
object.method()
object = {}

CODE WALKTHROUGH

Objectifying TooClose

Introduction to Programming
Class 36, 9 May 2017

CODE REVIEW

```
var gx;
var gy;

function setup() {
  gx = width/2;
  gy = height/2;
}

function draw() {
  if(tooClose(gx, gy, bx, by, 80)) {
    fill(255, 0, 0); // red
  }
  //…
  drawGrumpy(); // draw grumpy
  updateGrumpy(); // move grumpy
}

function drawGrumpy() {
  noStroke();
  ellipse(gx, gy, 40, 40);
}

function updateGrumpy() {
  // take a step in a random direction
  gx += random(-1, 1);
  gy += random(-1, 1);
}
```

Introduction to Programming
Class 36, 9 May 2017

CODE REVIEW

```
var gx;
var gy;

function setup() {
  gx = width/2;
  gy = height/2;
}

function draw() {
  if(tooClose(gx, gy, bx, by, 80)) {
    fill(255, 0, 0); // red
  }
  //…
  drawGrumpy(); // draw grumpy
  updateGrumpy(); // move grumpy
}

function drawGrumpy() {
  noStroke();
  ellipse(gx, gy, 40, 40);
}

function updateGrumpy() {
  // take a step in a random direction
  gx += random(-1, 1);
  gy += random(-1, 1);
}
```

```
var grumpy;
```

Introduction to Programming
Class 36, 9 May 2017

CODE REVIEW

```
var gx;
var gy;

function setup() {
  gx = width/2;
  gy = height/2;
}

function draw() {
  if(tooClose(gx, gy, bx, by, 80)) {
    fill(255, 0, 0); // red
  }
  //…
  drawGrumpy(); // draw grumpy
  updateGrumpy(); // move grumpy
}

function drawGrumpy() {
  noStroke();
  ellipse(gx, gy, 40, 40);
}

function updateGrumpy() {
  // take a step in a random direction
  gx += random(-1, 1);
  gy += random(-1, 1);
}
```

```
var grumpy;

function setup() {
  grumpy = {
    x: width/2,
    y: height/2,
  }
}
```

Introduction to Programming
Class 36, 9 May 2017

CODE REVIEW

```
var gx;
var gy;

function setup() {
  gx = width/2;
  gy = height/2;
}

function draw() {
  if(tooClose(gx, gy, bx, by, 80)) {
    fill(255, 0, 0); // red
  }
  //…
  drawGrumpy(); // draw grumpy
  updateGrumpy(); // move grumpy
}

function drawGrumpy() {
  noStroke();
  ellipse(gx, gy, 40, 40);
}

function updateGrumpy() {
  // take a step in a random direction
  gx += random(-1, 1);
  gy += random(-1, 1);
}
```

```
var grumpy;

function setup() {
  grumpy = {
    x: width/2,
    y: height/2,
    draw: function() {
      noStroke();
      ellipse(this.x, this.y, 40, 40);
    }
  }
}
```

Introduction to Programming
Class 36, 9 May 2017

CODE REVIEW

```
var gx;
var gy;

function setup() {
  gx = width/2;
  gy = height/2;
}

function draw() {
  if(tooClose(gx, gy, bx, by, 80)) {
    fill(255, 0, 0); // red
  }
  //…
  drawGrumpy(); // draw grumpy
  updateGrumpy(); // move grumpy
}

function drawGrumpy() {
  noStroke();
  ellipse(gx, gy, 40, 40);
}

function updateGrumpy() {
  // take a step in a random direction
  gx += random(-1, 1);
  gy += random(-1, 1);
}
```

```
var gr

functi
  grum
    x:
    y: height/2,
    draw: function() {
      noStroke();
      ellipse(this.x, this.y, 40, 40);
    }
  }
}
```

NOTE: No function name, ANONYMOUS FUNCTION

Introduction to Programming
Class 36, 9 May 2017

CODE REVIEW

```
var gx;
var gy;

function setup() {
  gx = width/2;
  gy = height/2;
}

function draw() {
  if(tooClose(gx, gy, bx, by, 80)) {
    fill(255, 0, 0); // red
  }
  //…
  drawGrumpy(); // draw grumpy
  updateGrumpy(); // move grumpy
}

function drawGrumpy() {
  noStroke();
  ellipse(gx, gy, 40, 40);
}

function updateGrumpy() {
  // take a step in a random direction
  gx += random(-1, 1);
  gy += random(-1, 1);
}
```

```
var grumpy;

function setup() {
  grumpy = {
    x: width/2,
    y: height/2,
    draw: function() {
      noStroke();
      ellipse(this.x, this.y, 40, 40);
    },
    update: function() {
      this.x += random(-1, 1);
      this.y += random(-1, 1);
    }
  }
}
```

Introduction to Programming
Class 36, 9 May 2017

CODE REVIEW

```
var gx;
var gy;

function setup() {
  gx = width/2;
  gy = height/2;
}

function draw() {
  if(tooClose(gx, gy, bx, by, 80)) {
    fill(255, 0, 0); // red
  }
  //…
  drawGrumpy(); // draw grumpy
  updateGrumpy(); // move grumpy
}

function drawGrumpy() {
  noStroke();
  ellipse(gx, gy, 40, 40);
}

function updateGrumpy() {
  // take a step in a random direction
  gx += random(-1, 1);
  gy += random(-1, 1);
}
```

```
var grumpy;

function setup() {
  grumpy = {
    x: width/2,
    y: height/2,
    draw: function (){
      noStroke();
      ellipse(this.x, this.y, 40, 40);
    },
    update: function () {
      this.x += random(-1, 1);
      this.y += random(-1, 1);
    }
  }
}

function draw() {
  if(tooClose(grumpy.x, grumpy.y, bx, by,
80)) {
    fill(255, 0, 0); // red
  }
  //…
  grumpy.draw(); // draw grumpy
  grumpy.update(); // move grumpy
}
```

Introduction to Programming
Class 36, 9 May 2017

CODE REVIEW

```
var grumpy;

function setup() {
  grumpy = {
    x: width/2,
    y: height/2,
    draw: function {
      noStroke();
      ellipse(this.x, this.y, 40, 40);
    },
    update: function() {
      this.x += random(-1, 1);
      this.y += random(-1, 1);
    }
  }
}

function draw() {
  if(tooClose(grumpy.x, grumpy.y, bx, by,
80)) {
    fill(255, 0, 0); // red
  }
  //…
  grumpy.draw(); // draw grumpy
  grumpy.update(); // move grumpy
}
```

Declare variable

Introduction to Programming
Class 36, 9 May 2017

CODE REVIEW

```
var grumpy;

function setup() {
  grumpy = {
    x: width/2,
    y: height/2,
    draw: function {
      noStroke();
      ellipse(this.x, this.y, 40, 40);
    },
    update: function() {
      this.x += random(-1, 1);
      this.y += random(-1, 1);
    }
  }
}

function draw() {
  if(tooClose(grumpy.x, grumpy.y, bx, by,
80)) {
    fill(255, 0, 0); // red
  }
  //…
  grumpy.draw(); // draw grumpy
  grumpy.update(); // move grumpy
}
```

Initialize object
`object = {};`

CODE REVIEW

```
var grumpy;

function setup() {
  grumpy = {
    x: width/2,
    y: height/2,
    draw: function {
      noStroke();
      ellipse(this.x, this.y, 40, 40);
    },
    update: function() {
      this.x += random(-1, 1);
      this.y += random(-1, 1);
    }
  }
}

function draw() {
  if(tooClose(grumpy.x, grumpy.y, bx, by,
80)) {
    fill(255, 0, 0); // red
  }
  //…
  grumpy.draw(); // draw grumpy
  grumpy.update(); // move grumpy
}
```

Set **PROPERTIES**

```
object = {
 prop1: value,
 prop2: value
};
```

Introduction to Programming
Class 36, 9 May 2017

CODE REVIEW

```
var grumpy;

function setup() {
  grumpy = {
    x: width/2,
    y: height/2,
    draw: function {
      noStroke();
      ellipse(this.x, this.y, 40, 40);
    },
    update: function() {
      this.x += random(-1, 1);
      this.y += random(-1, 1);
    }
  }
}

function draw() {
  if(tooClose(grumpy.x, grumpy.y, bx, by,
80)) {
    fill(255, 0, 0); // red
  }
  //…
  grumpy.draw(); // draw grumpy
  grumpy.update(); // move grumpy
}
```

Set **METHODS**

```
object = {
  method1: function {
  …
  this.prop1
  …
  }
};
```

Dot Notation

Introduction to Programming
Class 36, 9 May 2017

CODE REVIEW

```
var grumpy;

function setup() {
  grumpy = {
    x: width/2,
    y: height/2,
    draw: function {
      noStroke();
      ellipse(this.x, this.y, 40, 40);
    },
    update: function() {
      this.x += random(-1, 1);
      this.y += random(-1, 1);
    }
  }
}

function draw() {
  if(tooClose(grumpy.x, grumpy.y, bx, by
80)) {
    fill(255, 0, 0); // red
  }
  //…
  grumpy.draw(); // draw grumpy
  grumpy.update(); // move grumpy
}
```

Use **Properties**
object.prop1;

CODE REVIEW

```
var grumpy;

function setup() {
  grumpy = {
    x: width/2,
    y: height/2,
    draw: function {
      noStroke();
      ellipse(this.x, this.y, 40, 40);
    },
    update: function() {
      this.x += random(-1, 1);
      this.y += random(-1, 1);
    }
  }
}

function draw() {
  if(tooClose(grumpy.x, grumpy.y, bx, by,
80)) {
    fill(255, 0, 0); // red
  }
  //…
  grumpy.draw(); // draw grumpy
  grumpy.update(); // move grumpy
}
```

Call **Methods**
`object.method1();`

DEFINITION

OBJECTS are a way of grouping variables together.

DO

"Objectify" blissful.

Brickly

Complete the Handout

SAM

Download Install

HW

- **Grab That Ball!** A ball moves across the screen bouncing off the walls. When a mouse click is detected within the ball's radius, the ball moves to a new random location, changes color, and a point is awarded. The score is visible on the screen.

- **Pong**. The classic. Two paddles on either side of the screen, each with its own set of key controls. Detect when the ball collides with a paddle and have it reverse horizontal direction for that case.