The Object Class

- "Object" is the most general container
  - Object with a capital "O"
- Variables of type Object can hold any other type

```
Object o1 = new String("abc");
Object o2 = "abc";
Object o3 = new PImage(100, 100);
Object o4 = 123;
Object o5 = true;
```

The Object Class

- Constructors

```
Object o = new Object();
```

- Fields
- Methods

```
// Tests for equality with Object o2
o.equals(Object o2)

// Returns a String representation of Object
o.toString()
…
```

http://download.oracle.com/javase/7/docs/api/java/lang/Object.html

The Object Class

- Variables of type Object don't know the type they hold, so the compiler can't check for legal operations.

```
void setup() {
  Object o1 = "ABC";
  String o2 = "DEF";

  o2 = o2.toLowerCase();
  println(o2);

  //o1 = o1.toLowerCase();
  // Runtime Error
  // Object class does not have toLowerCase() method

}
```

Type Casting

- We learned about type-conversion functions

```
int( … ), float( … ), boolean( … ), …
```

- Another way to convert from one type to another is called "type casting," which works by preceding an expressing with the target type in parentheses.

```
float f = 12.0;
int i = (int)f;  // Will not work without type cast

Object o = new PImage(100, 100);
PImage p = (PImage)o;
```

Built-in Collection Classes

- ArrayList
  - A built-in object that stores and manages an *arbitrary* number of data items of any type (Objects).
  - Objects in an ArrayList are access by **index** [0..size-1]

- HashMap
  - A built-in object that stores and manages an *arbitrary* number of data items of any type (Objects).
  - Objects in a HashMap are access by a **key**, which can be another Object, frequently a String.

ArrayList

- Constructors

```
ArrayList myList = new ArrayList();
ArrayList myList = new ArrayList(initialSize);
```

- Fields
- Methods

```
myList.size()                    // Returns the num of items held.
myList.add(Object o)             // Appends o to end.
myList.add(int idx, Object o)    // Inserts o at pos idx.
myList.remove(int idx)           // Removes item at pos idx.
myList.get(int idx)              // Gets items at idx. No removal.
myList.set(int idx, Object o)    // Replaces item at idx with o.
myList.clear()                   // Removes all items.
myList.isEmpty()                 // Returns true if empty.
```

ArrayList Example – Box Dropper

```
// Box Dropper
ArrayList boxes = new ArrayList();

void setup() { size(500, 500); }

void draw() {
  background(0);

  for (int i = boxes.size()-1; i>=0; i--) {
    //boxes.get(i).draw();       // Fails. Why?
    Box b = (Box)boxes.get(i);   // Type cast Object->Box
    b.y = b.y + b.v;             // Physics
    b.v = b.v + 0.02;
    b.draw();

    // Remove Box from ArrayList if below sketch
    if (b.y > height) {
      boxes.remove(i);
      println(boxes.size() + " boxes remaining");
    }
  }
}

void mousePressed() {
  Box b = new Box(mouseX, mouseY)
  boxes.add( b );
  println( boxes.size() + " boxes in ArrayList" );
}
```

```
// A simple Box class
class Box {
  float x, y, v;

  Box(float tx, float ty) {
    x = tx;  // x position
    y = ty;  // y position
    v = 0.0; // y velocity
  }

  void draw() {
    fill(200);
    rect(x, y, 20, 20);
  }
}
```

- Why can we not call draw directly on item in ArrayList?
- Why do we loop over ArrayList backwards?

HashMap

- Constructors

```
HashMap myMap = new HashMap();
HashMap myMap = new HashMap(initialCapacity);
```

- Fields
- Methods

```
myMap.size()                        // Returns num of items held.
myMap.put(Object key, Object o)     // Puts o in map at key
myMap.remove(Object key)            // Remove Object at key
myMap.get(Object key)               // Get Object at key
myMap.containsKey(Object key)       // True if map contains key
myMap.containsValue(Object val)     // True if map contains val
myMap.clear()                       // Removes all items.
myMap.isEmpty()                     // Returns true if empty.
```

HashMap Example – High Score

```
// HighScore
HashMap scores = new HashMap();

void setup() {
  size(500, 500);

  // Init HashMap
  scores.put("Fred", 2);
  scores.put("Wilma", 4);
  scores.put("Barney", 10);
  scores.put("Betty", 5);
  scores.put("BamBam", 6);
  scores.put("Pebbles", 5);

  // Draw once
  noLoop();
  drawMap(scores);
}

void draw() { }
```

```
// Draw the HashMap to the sketch
void drawMap(HashMap hm) {
  background(0);
  fill(255);
  textSize(20);

  // Display all scores
  text( buildScore("Fred", scores), 100, 100);
  text( buildScore("Wilma", scores), 100, 150);
  text( buildScore("Barney", scores), 100, 200);
  text( buildScore("Betty", scores), 100, 250);
  text( buildScore("BamBam", scores), 100, 300);
  text( buildScore("Pebbles", scores), 100, 350);

  redraw();
}

// Build a return a String for displaying a Score
String buildScore(String name, HashMap hm) {
  String msg = name + ":" + hm.get(name).toString();
  return msg;
}
```

ArrayList Example - Fireworks