Bonus: Useful Tools

Data Structures in Processing

Agenda

Data structures – tools you can use in Processing

Introduction to ArrayList

Introduction to HashMap

Data Structures in Processing are DYNAMIC

So what do we mean by **dynamic**?

Data Structures can be any size you want, and you can and remove items from them at **runtime** (while your program is running).

Data Structures in Processing

Flexible data structures
ArrayList
HashMap

“Shortcut” data structures
FloatList, StringList, IntList
FloatDict, StringDict, IntDict

Specialized data structures
XML
JSON
Table and TableRow

ArrayList

```
// put ints in a list, note Integer instead of int
ArrayList<Integer> myIntList = new ArrayList<Integer>();
myIntList.add(45); // [45]
myIntList.size(); // 1
myIntList.add(52); // [45, 52]
myIntList.size(); // 2
myIntList.remove(0); // [52]
myIntList.size(); // 1

// put floats in a list, note Float instead of float
ArrayList<Float> myFloatList = new ArrayList<Float>();

ArrayList<String> myStringList = new ArrayList<String>(); // or Strings

ArrayList<Ball> myBallList = new ArrayList<Ball>(); // objects too!
```

ArrayList vs Arrays

ArrayList stores a **variable** number of items

an Array stores a **static** number of items

```
ArrayList<int> numbers = new ArrayList<int>();
int[] otherNumbers = {5, 10};

void setup() {

  // we can't do this with an array
  numbers.add( 5 );
  numbers.add( 10 );

  println(numbers.get(0));
  println(numbers.get(1));
  numbers.remove(1);
}
```

ArrayList Example: Particle System

```
ArrayList<Particle> particles = new
ArrayList<Particle>();

void setup() {
  particles.add( new Particle() );
}

void draw() {
  for (int i = 0; i < particles.size(); i++)
  {
    Particle p = particles.get(i);
    if (p.isDead == true) {
      particles.remove(i);
    } else {
      p.draw();
    }
  }
}
```

ArrayList Example: Word Scrambler

```
ArrayList<String> words;

void setup() {
  String[] someWords = { “hello”, “world”, “of”,
“Processing” };
  words = new ArrayList<String>(someWords);
  words.shuffle(); // put words in random order
  for (int i = 0; i < words.size(); i++) {
    print(word.get(i) + “ “);
  }
}
```

ArrayList shortcuts in Processing

```
IntList myIntList; // same as ArrayList<Integer>
FloatList myFloatList; // same as ArrayList<Float>
StringList myStringList; // same as ArrayList<String>
```

HashMap

HashMap uses a **key** (how you look up an item - like the number in an array!) and a **value** (what you get back from the HashMap when you look something up)

```
HashMap<String, String> myStringMap = new HashMap<String, String>();
myStringMap.put("Hello", "World");
myStringMap.get("Hello"); // "World"

HashMap<String, Float> myFloatMap = new HashMap<String, Float>();
myFloatMap.put("smallNumber", 0.1);
myFloatMap.put("bigNumber", 9995320.338401);
myFloatMap.get("bigNumber"); // 9995320.338401
```

HashMap

Example: Get room descriptions for a text adventure game

```
import java.util.Map; // important!

// we use a String for the "key"
// and a String for the "value" as well
HashMap<String, String> roomDescriptions = new
HashMap<String, String>();
String currentRoomName = "entrance";

void setup() {
  roomDescriptions.put("entrance", "This looks
like an entrance to me.");
  roomDescriptions.put("hallway", "A simple
hallway with lamps against one wall.");

 if (currentRoomName == "entrance") {
  print("Entrance: ");
  println(roomDescription.get("entrance"));
 }
}
```

HashMap

Storing Objects in a HashMap

Example: Get a room name, description, and a picture for a text adventure game

```
import java.util.Map; // important!

class Room {
  String name;
  String description;
  PImage picture;
}

HashMap<String, Room> rooms = new
HashMap<String, Room>();
String currentRoomName = "Foyer";

void setup() {
 // pretend we created these room objects
 // already!
 rooms.put("Foyer", Foyer);
 rooms.put("Hallway", Hallway);
 rooms.put("Dining Room", DiningRoom);
 Room currentRoom = rooms.get(currentRoomName);
 println(room.name); // "Foyer"
 println(room.description); // "A big foyer."
 image(room.picture, 0, 0); // display picture
}
```

HashMap shortcuts in Processing

```
IntDict myIntDict = new IntDict(); // same as HashMap<String,Integer>
FloatDict myFloatDict = new FloatDict(); // same as HashMap<String,Float>
StringDict myStringDict = new StringDict(); // same as HashMap<String,String>
```

Read more

ArrayLists https://processing.org/reference/ArrayList.html

HashMaps https://processing.org/reference/HashMap.html