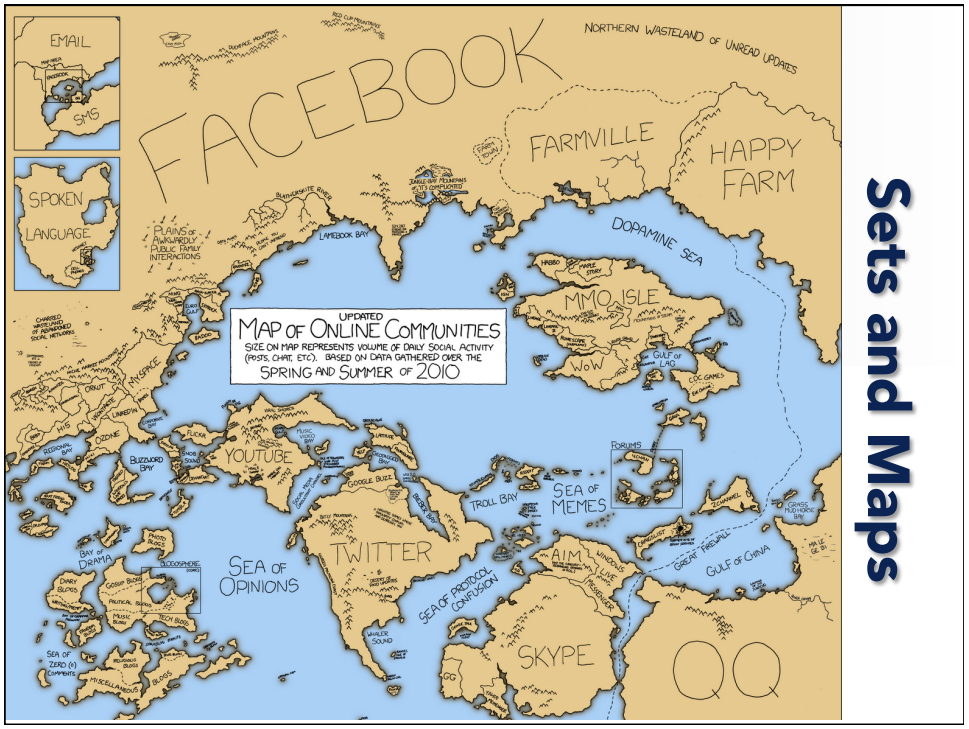
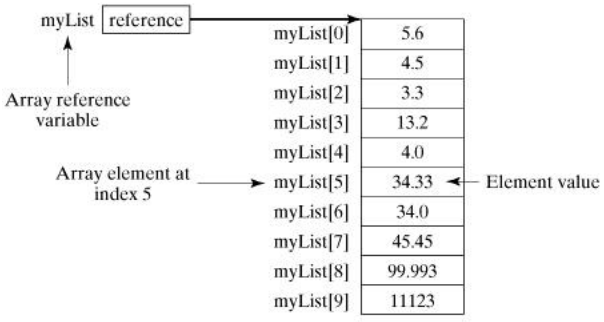


For those who were wondering

- anArray.length
  - length - public final attribute
    - you can access the data
    - you cannot change the data

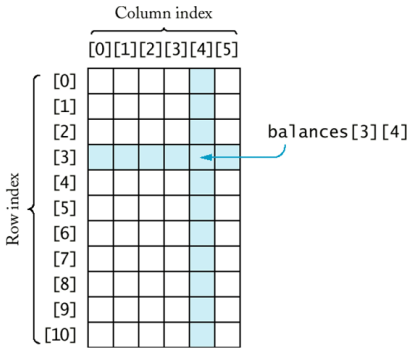


For those who were wondering

- 2D array

```
int[][] twoD = new int[11][6];

twoD[3][4] = 7;
```

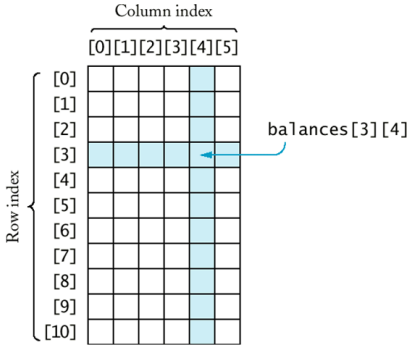


For those who were wondering

- 2D array

```
int[][] twoD = new int[11][6];

twoD[3][4] = 7;
```



Some Data Structures

- Array – ordered, indexed, fixed length
- List – ordered, indexed, adjustable length
- Set
- Map

Array vs. List

- Which is faster?
- Code example

```
double start = System.currentTimeMillis();
array.makeArray(size);
double end = System.currentTimeMillis();
System.out.printf("Array: total time = %f\n",
                  (end - start) / 1000);
```

Some Data Structures

- Array – ordered, indexed, fixed length
- List – ordered, indexed, adjustable length
- Set
- Map

Set

- Unordered collection of unique values

java.util

Interface Set<E>

Method Summary

boolean	**add**(E e) Adds the specified element to this set if it is not alrea
boolean	**addAll**(Collection<? extends E> c) Adds all of the elements in the specified collection to
void	**clear**() Removes all of the elements from this set (optional o
boolean	**contains**(Object o) Returns true if this set contains the specified elemen
boolean	**containsAll**(Collection<?> c) Returns true if this set contains all of the elements o
boolean	**equals**(Object o) Compares the specified object with this set for equali
int	**hashCode**() Returns the hash code value for this set.

Set

```
Set<Double> set = new HashSet<Double>();
```

or

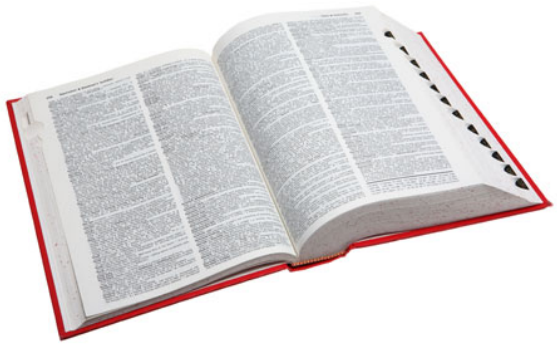
```
Set<Double> set = new TreeSet<Double>();

boolean added = set.add(3.0);

boolean inSet = set.contains(3.0);
```

Map

- Unordered collection of values mapped to keys
  - dictionary
    - key – word
    - value - definition



Map

```
Map<Double, Integer> map =
                    new HashMap<Double, Integer>();

for(double d: map.keySet()){
    System.out.println(d + ": " + map.get(d));
 }
```

http://docs.oracle.com/javase/6/docs/api/java/util/HashMap.html

Practice

- Snarf today's code
  - Change the method `buildCircles` so that it builds an array of circles with length `numCircles` in random locations with random colors.
    - Hint: `(int)(Math.Random() * 500)` will give you a random integer between 0 - 499
  - Complete the method `countColors` so that it displays the number of circles of each color
    - Hint: use a Map