

Arrays and Classes

Produced Dr. Siobhán Drohan
by: Mr. Colm Dunphy
 Mr. Diarmuid O'Connor



Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

Department of Computing and Mathematics
<http://www.wit.ie/>

Let's Look at arrays of different types

Arrays can store any type of data

Let's look at some examples:

1. Array of primitives - **int**
2. Array of objects – **String**
3. Array of objects - **Spot**

An array can store any type of data.

Primitive Types

```
int numbers[] = new int[10];
```

```
byte smallNumbers[] = new byte[4];
```

```
char characters[] = new char[26];
```

Object Types

```
String words = new String[4];
```

```
Spot spots[] = new Spot[10];
```

1) Array of Primitives

e.g. int

Structure of an **int** primitive array

```
int[] numbers;
```

numbers

```
null
```

Structure of an **int** primitive array

```
int[] numbers;
```

```
numbers = new int[4];
```

numbers

0	0
1	0
2	0
3	0

Structure of an **int** primitive array

```
int[] numbers;
```

```
numbers = new int[4];
```

```
numbers[2] = 18;
```

We are directly
accessing the
element at index **2**
and setting it to a
value of **18**.

numbers

0	0
1	0
2	18
3	0

Structure of an **int** primitive array

```
int[] numbers;
```

```
numbers = new int[4];
```

```
numbers[2] = 18;
```

```
numbers[0] = 12;
```

We are setting the element at index **0** to a value of **12**.

numbers

0	12
1	0
2	18
3	0

Structure of an **int** primitive array

```
int[] numbers;
```

```
numbers = new int[4];
```

```
numbers[2] = 18;
```

```
numbers[0] = 12;
```

```
print(numbers[2]);
```

numbers

0	12
1	0
2	18
3	0

Here we are printing the contents of index location 2
i.e. 18 will be printed to the console.

2) Array of Objects
e.g. String

An array can store any type of data.

Primitive Types

```
int numbers[] = new int[10];
```

```
byte smallNumbers[] = new byte[4];
```

```
char characters[] = new char[26];
```

Object Types

```
String words = new String[4];
```

```
Spot spots[] = new Spot[10];
```

Structure of a **String** object array

String[] words;

words

null

Structure of a **String** object array

```
String[] words;
```

```
words = new String[4];
```

words

0	null
1	null
2	null
3	null

Structure of a **String** object array

```
String[] words;
```

```
words = new String[4];
```

```
words[1] = "Dog";
```

words

0	null
1	
2	null
3	null

"Dog"

Structure of a **String** object array

```
String[] words;
```

```
words = new String[4];
```

```
words[1] = "Dog";
```

We are directly accessing the element at index **1** and setting it to a value of “Dog”.

words

0	null
1	
2	null
3	null

“Dog”

Structure of a **String** object array

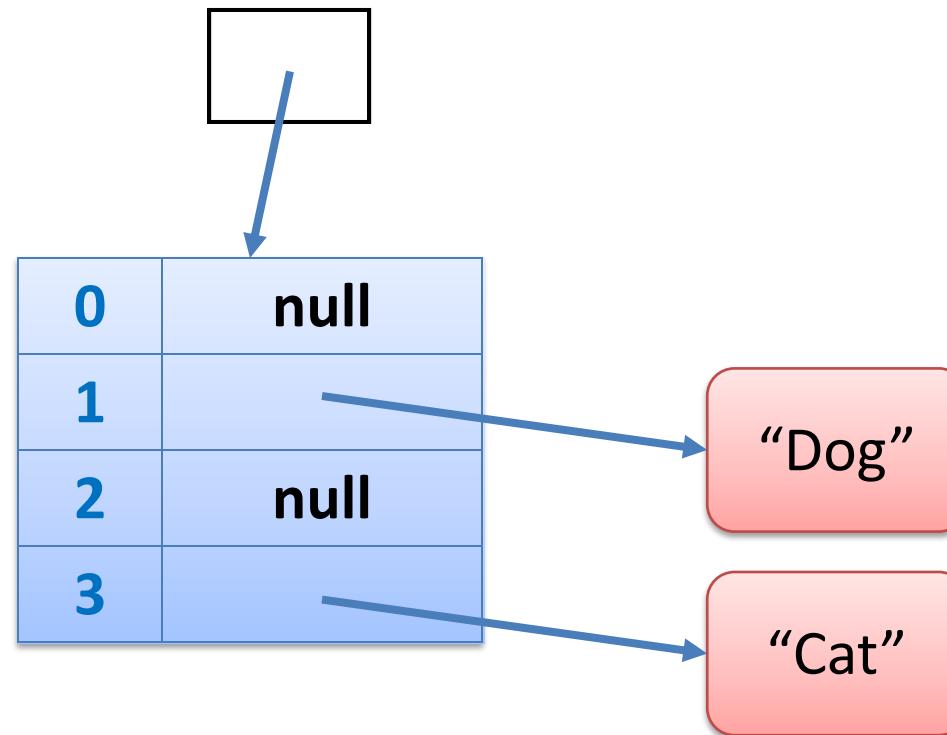
```
String[] words;
```

```
words = new String[4];
```

```
words[1] = "Dog";
```

```
words[3] = "Cat";
```

words



Structure of a **String** object array

```
String[] words;
```

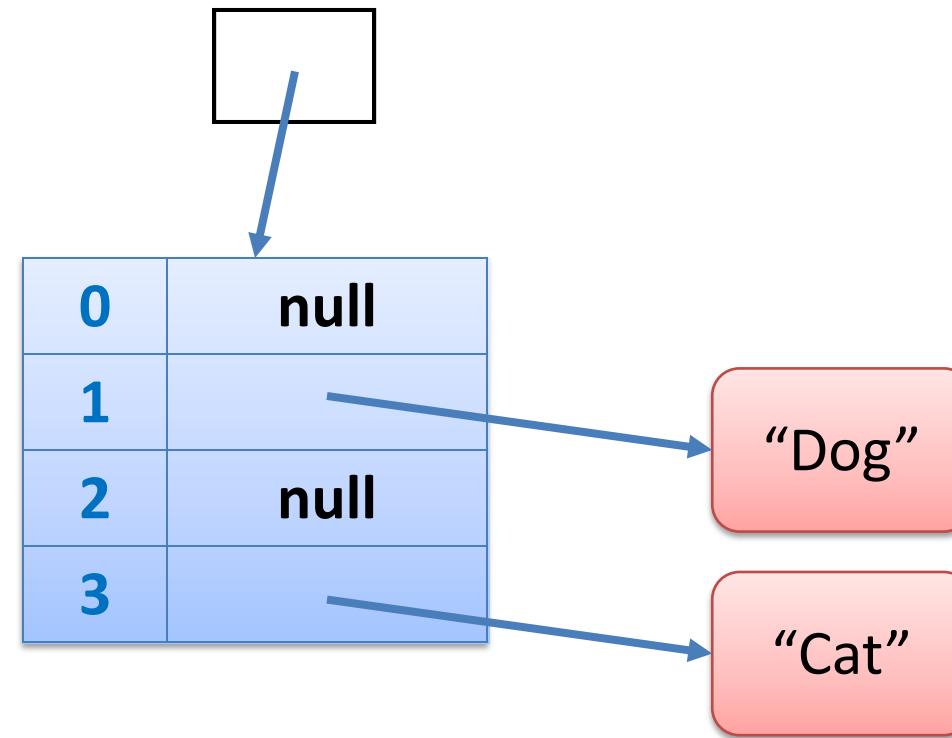
```
words = new String[4];
```

```
words[1] = "Dog";
```

```
words[3] = "Cat";
```

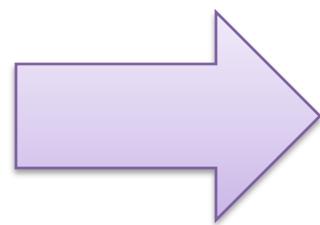
The element at index
3 is set to "Cat".

words



Structure of a **String** object array

```
String words[];  
  
words = new String[4];  
  
words[1] = "Dog";  
words[3] = "Cat";  
  
for (int i=0; i < words.length; i++)  
{  
    println(words[i]);  
}
```



```
null  
Dog  
null  
Cat
```

3) Array of Objects
e.g. Spot

An array can store any type of data.

Primitive Types

```
int numbers[] = new int[10];
```

```
byte smallNumbers[] = new byte[4];
```

```
char characters[] = new char[26];
```

Object Types

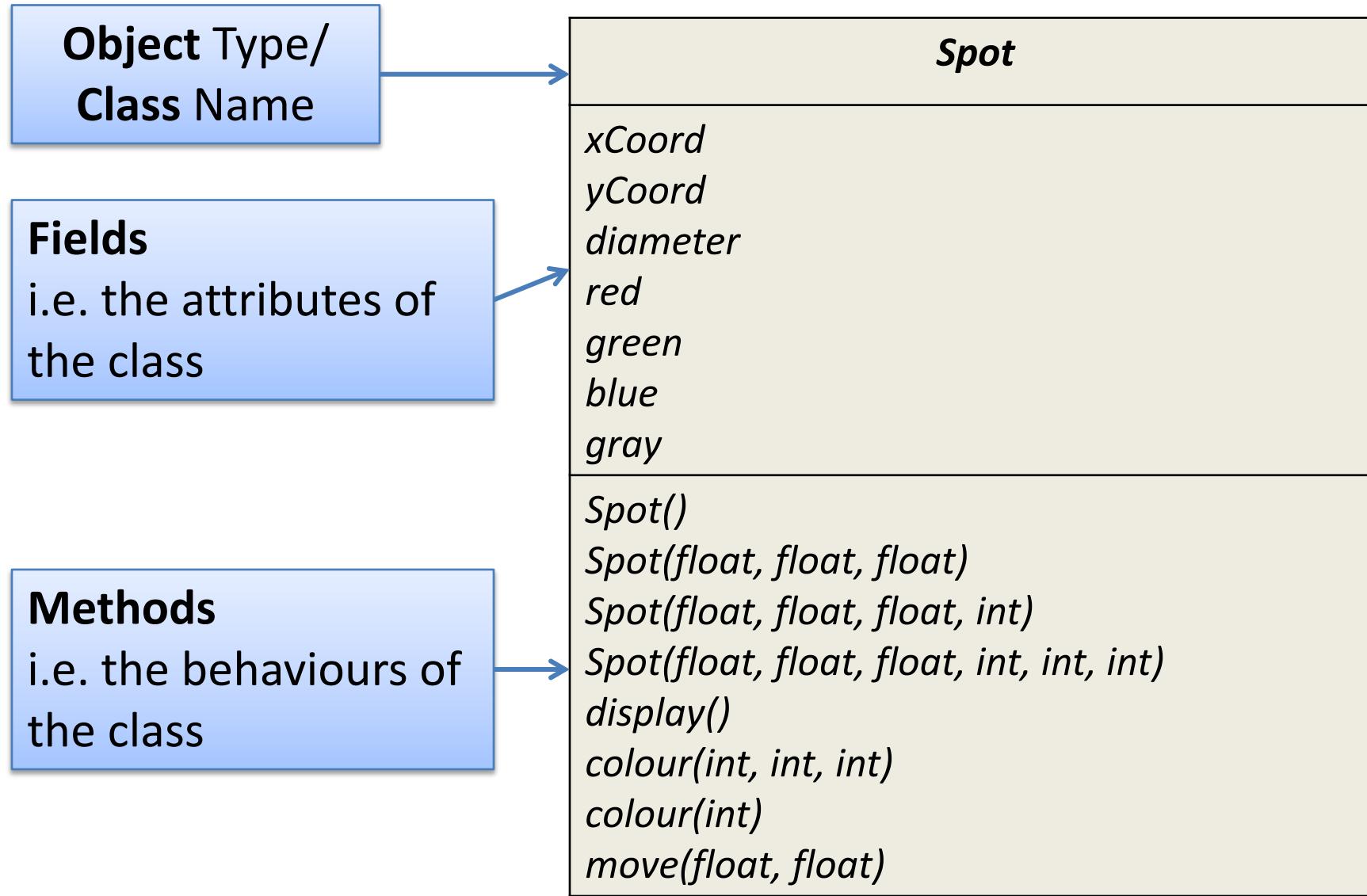
```
String words = new String[4];
```

```
Spot spots[] = new Spot[10];
```

Remember our **Spot** class?

Lets look at one of the versions
we worked on.

Class Diagram for Spot Version 6.1



Spot Class

– Version 6.1



```
class Spot{  
    float xCoord, yCoord;  
    float diameter;  
    int red, green, blue;  
  
    Spot()  
    {  
    }  
  
    Spot(float xCoord, float yCoord, float diameter)  
    {  
        this.xCoord = xCoord;  
        this.yCoord = yCoord;  
        this.diameter = diameter;  
    }  
  
    // colour methods...  
    // display method...  
    // move method...  
}
```

Spot Class

– Version 6.1



```
class Spot{  
// fields and constructors...  
  
void display()  
{  
    ellipse(xCoord, yCoord, diameter, diameter);  
}  
  
void colour(int red, int green, int blue)  
{  
    this.red = red;  
    this.green = green;  
    this.blue = blue;  
    fill (red, green, blue);  
}  
  
void colour(int gray){  
    this.gray = gray;  
    fill (this.gray);  
}  
}
```

Structure of a **Spot** primitive array

Spot[] spots;

spots

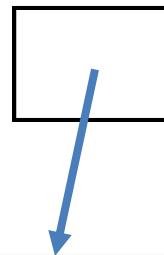
null

Structure of a **Spot** primitive array

```
Spot[] spots;
```

```
spots = new Spot[4];
```

spots



0	null
1	null
2	null
3	null

Structure of a **Spot** primitive array

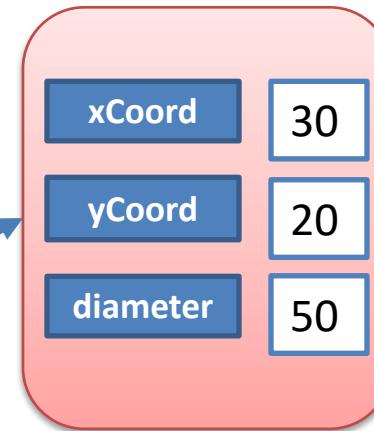
```
Spot[] spots;
```

```
spots = new Spot[4];
```

```
spots[1] = new Spot(30,20,50);
```

spots

0	null
1	
2	null
3	null



Structure of a **Spot** primitive array

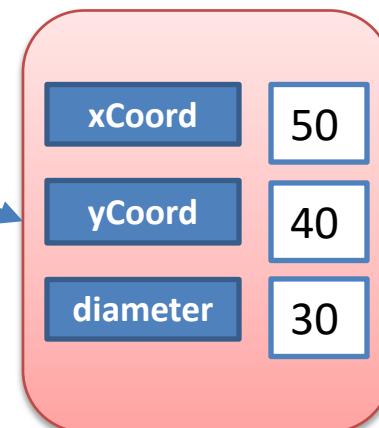
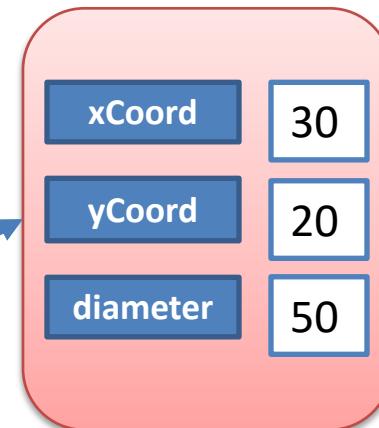
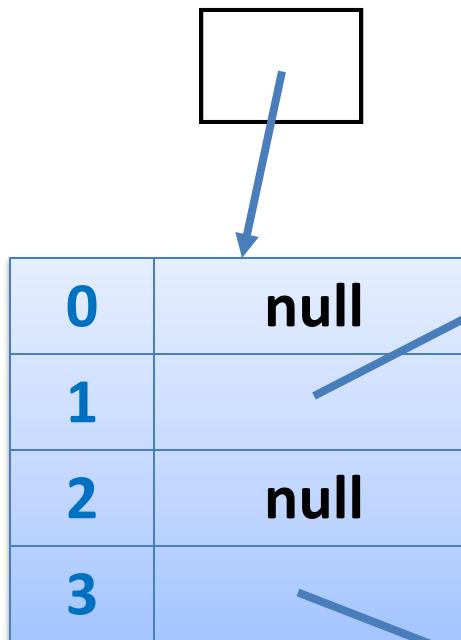
```
Spot[] spots;
```

```
spots = new Spot[4];
```

```
spots[1] = new Spot(30,20,50);
```

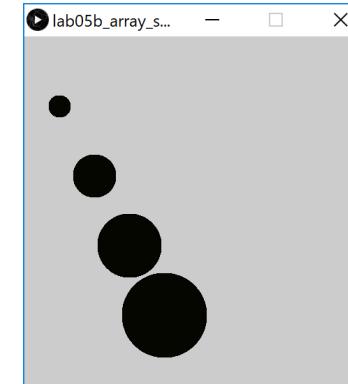
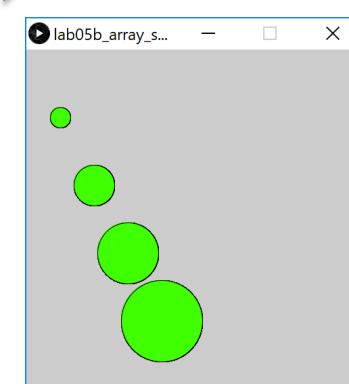
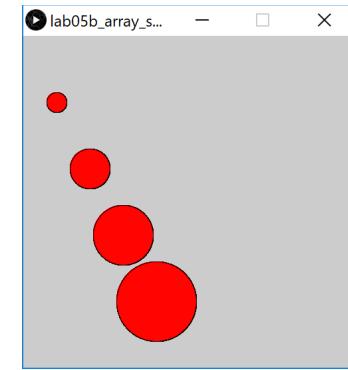
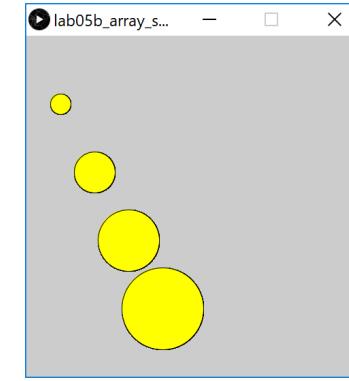
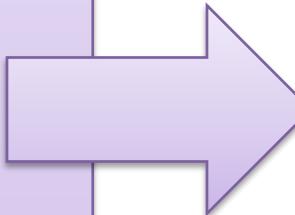
```
spots[3] = new Spot(50,40,30);
```

spots



Example using a **Spot** object array

```
Spot[] spots; // create a spots array of type class Spot  
  
void setup(){  
    size(500,500);  
    spots = new Spot[4]; // make the array size 4  
  
    for(int i = 1; i <= spots.length; i++){  
        spots[i-1] = new Spot(i*50, i*100, i*30);  
            // create a new instance of a Spot  
            // in each element of the array  
    }  
}  
  
void draw(){  
    for (int i=0; i < spots.length; i++){  
        spots[i].display(); // display each Spot  
        spots[i].colour(mouseX, mouseY, 0);  
            // colour using mouse position  
    }  
}
```



Questions?

