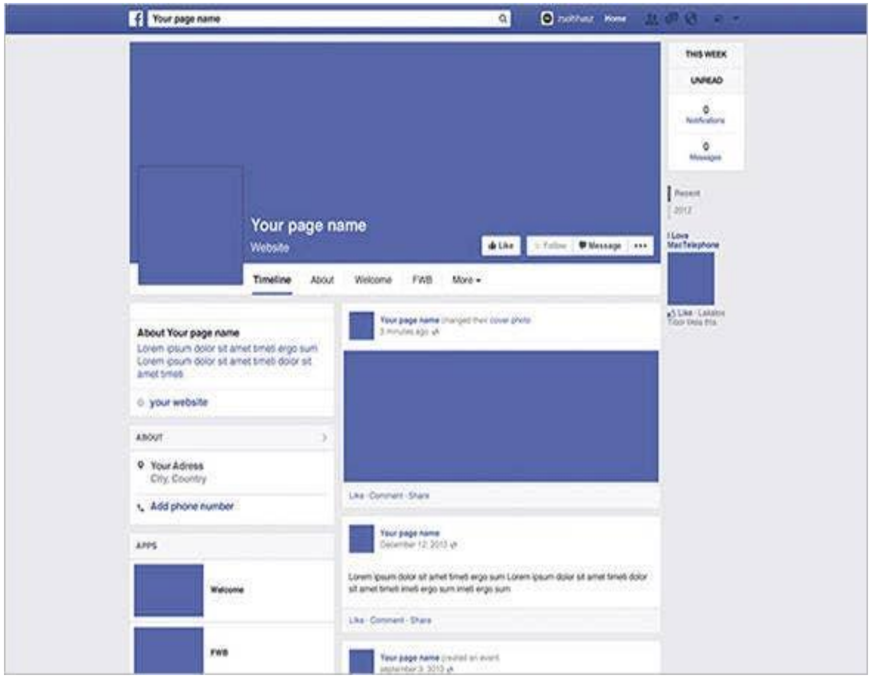
Object Oriented Programming

Chapter 6

Ayesha Khatun

Lecturer, Green University of Bangladesh

Class VS Object



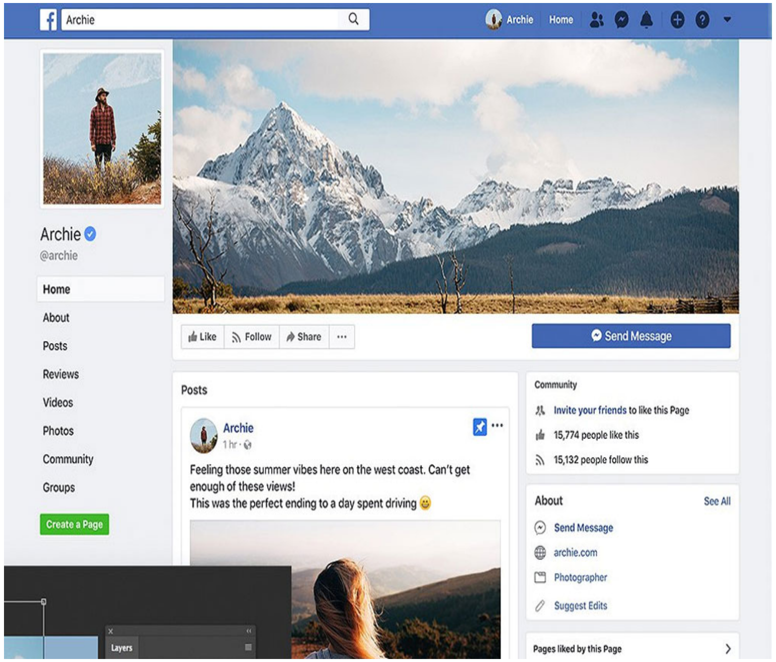
Lorem ipsum dolor sit amet timeti ergo sum Lorem ipsum dolor sit amet timeti dolor sit amet timeti

Your page name changed their cover photo
3 minutes ago

Your page name
December 12, 2013

Lorem ipsum dolor sit amet timeti ergo sum Lorem ipsum dolor sit amet timeti dolor sit amet timeti imeti ergo sum imeti ergo sum

Your page name created an event
september 3, 2013



Archie
1 hr ·

Feeling those summer vibes here on the west coast. Can't get enough of these views!
This was the perfect ending to a day spent driving 😀

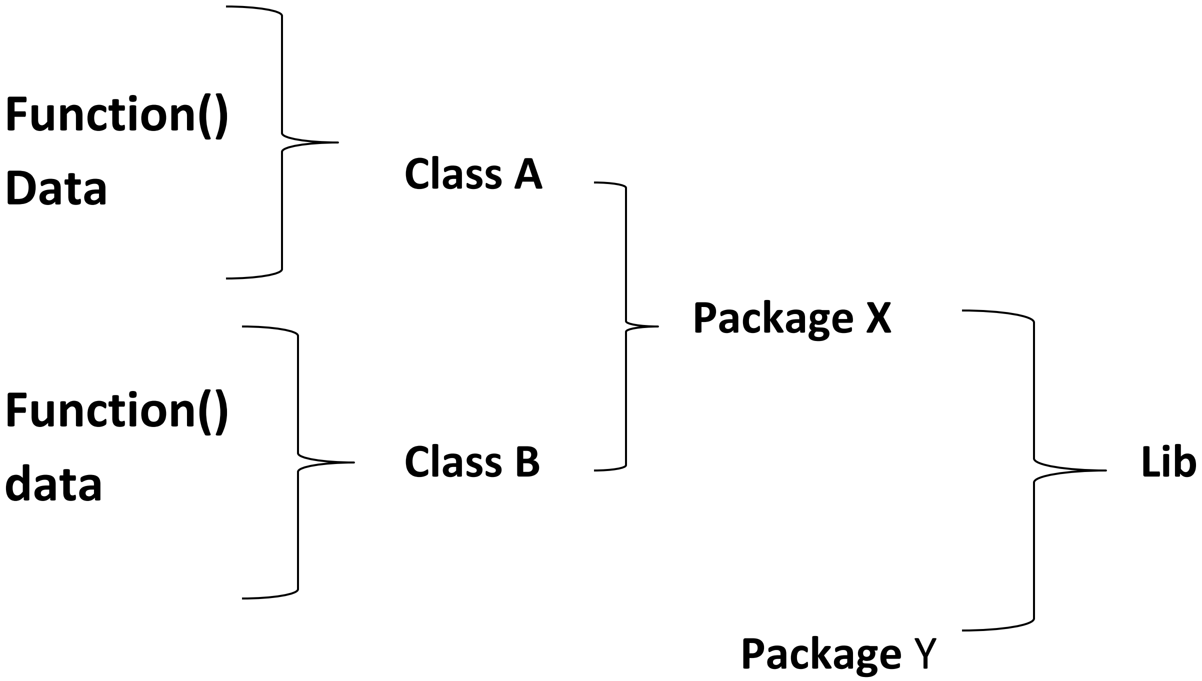
Invite your friends to like this Page
15,774 people like this
15,132 people follow this

Class is like an empty template

When fill up the empty class, is like an object

A class can have millions of objects.

Basic Structure of OOP



Function()
Data

Class A

Package X

Function()
data

Class B

Lib

Package Y

The General Form of a Class

- class Box {
- double width;
- double height;
- double depth;
- }

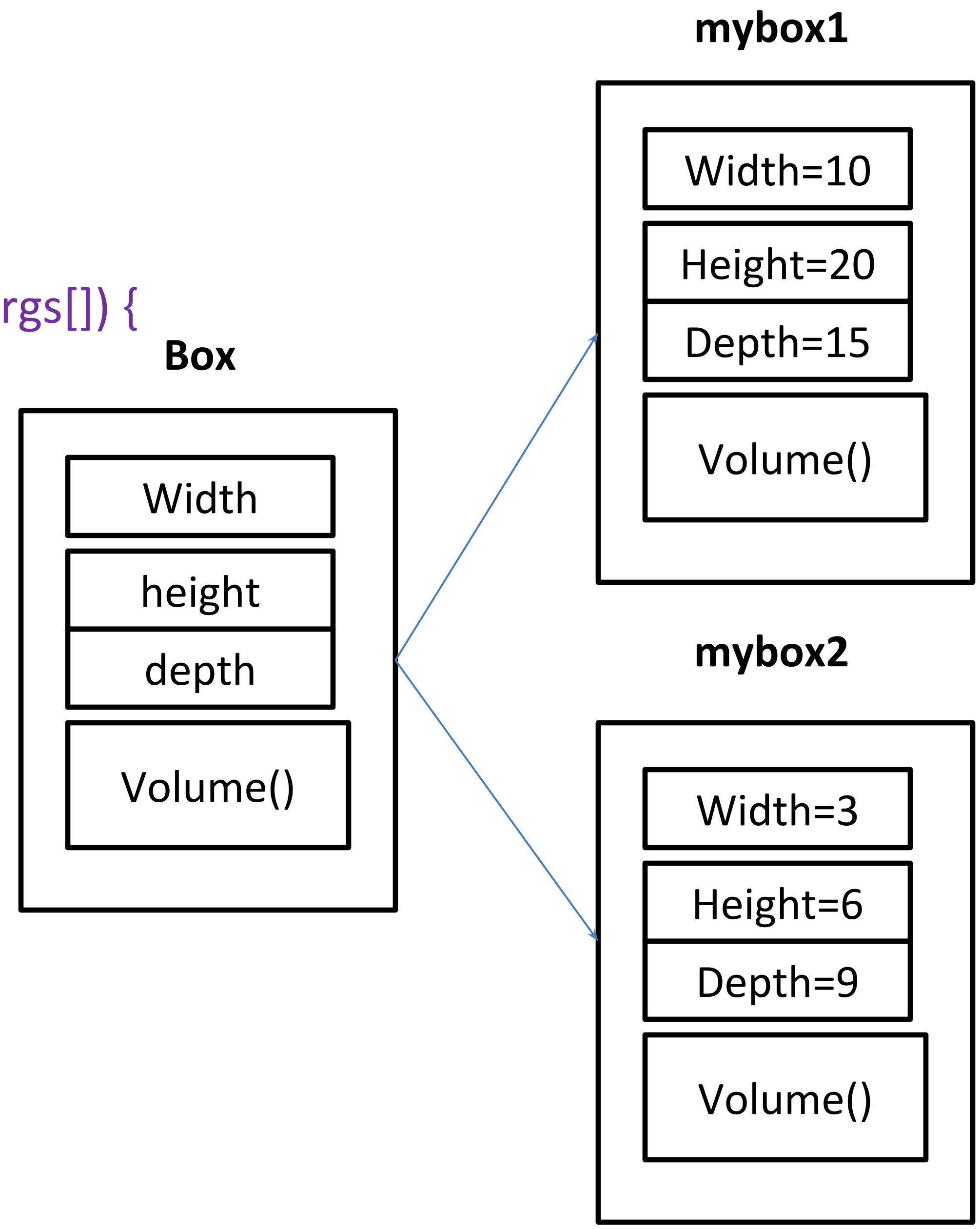
Object

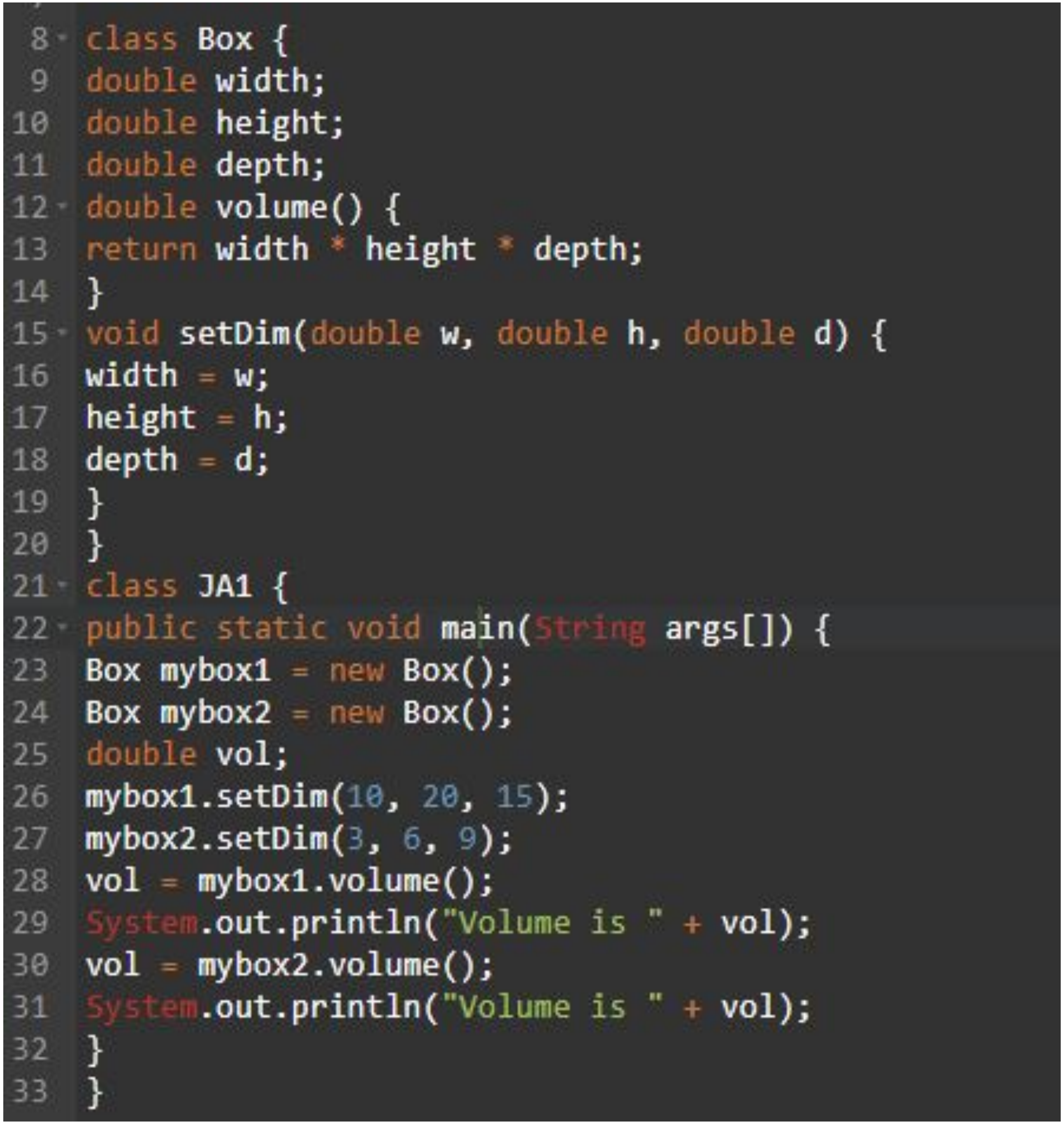
```
class Box {
double width;
double height;
double depth;   }
class BoxDemo {
public static void main(String args[]) {
Box mybox = new Box();
double vol;
mybox.width = 10;
mybox.height = 20;
mybox.depth = 15;

vol = mybox.width * mybox.height * mybox.depth;
System.out.println("Volume is " + vol); } }
```

Adding a Method to the Box Class

- class Box {
- double width;
- double height;
- double depth;
- // display volume of a box
- void volume() {
- System.out.print("Volume is ");
- System.out.println(width * height * depth);
- }
- }

```
class BoxDemo3 {
public static void main(String args[]) {
Box mybox1 = new Box();
Box mybox2 = new Box();
mybox1.width = 10;
mybox1.height = 20;
mybox1.depth = 15;
mybox2.width = 3;
mybox2.height = 6;
mybox2.depth = 9;
mybox1.volume();
mybox2.volume();
}
```





```java
class Box {
double width;
double height;
double depth;
double volume() {
return width * height * depth;
}
void setDim(double w, double h, double d) {
width = w;
height = h;
depth = d;
}
}
class JA1 {
public static void main(String args[]) {
Box mybox1 = new Box();
Box mybox2 = new Box();
double vol;
mybox1.setDim(10, 20, 15);
mybox2.setDim(3, 6, 9);
vol = mybox1.volume();
System.out.println("Volume is " + vol);
vol = mybox2.volume();
System.out.println("Volume is " + vol);
}
}
```

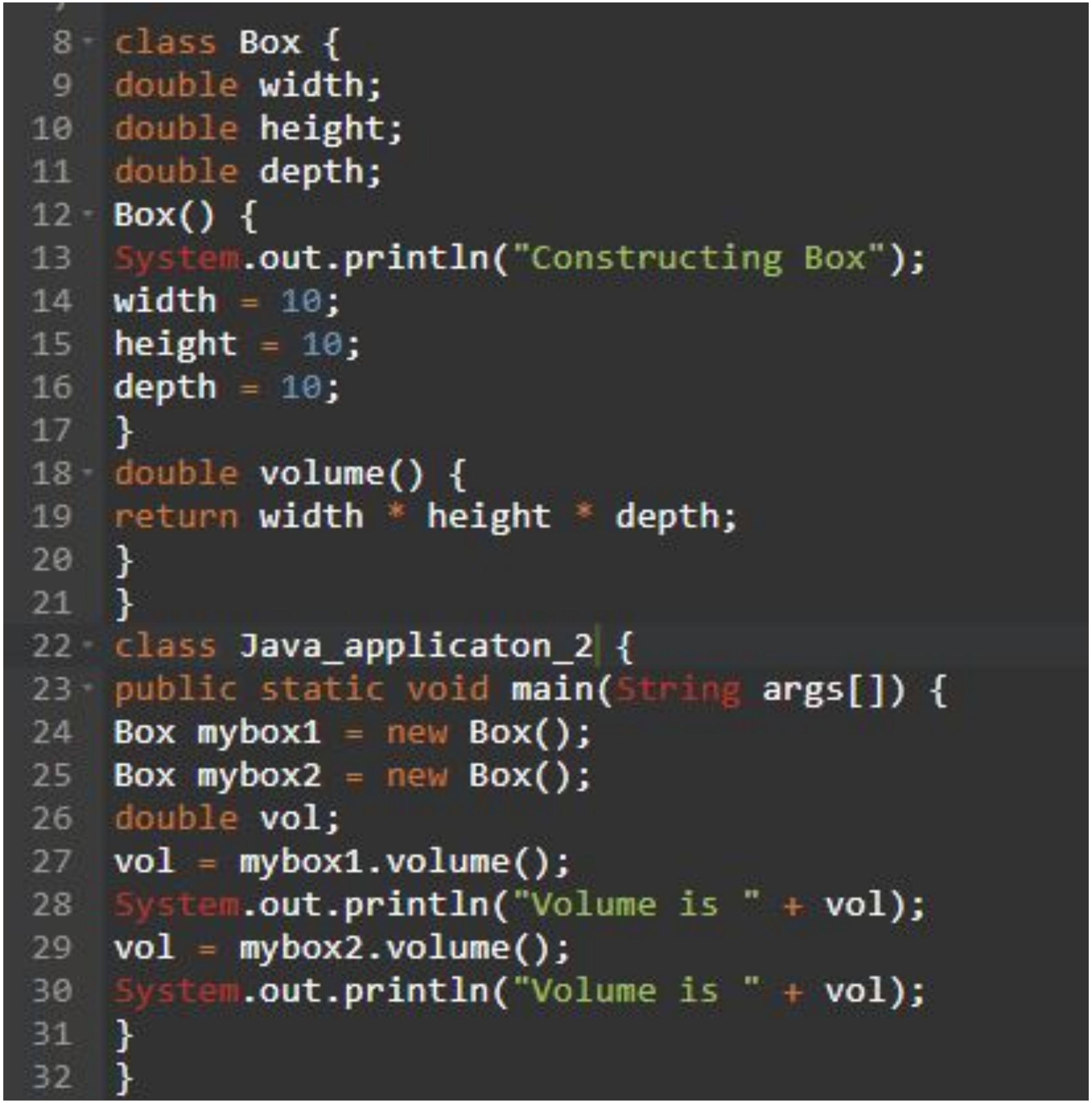
Constructors

- A *constructor initializes an object immediately upon creation.*

  1. class name=function name

  2. No return type

  3. value Initialization



```
class Box {
double width;
double height;
double depth;
Box() {
System.out.println("Constructing Box");
width = 10;
height = 10;
depth = 10;
}
double volume() {
return width * height * depth;
}
}
class Java_applicaton_2 {
public static void main(String args[]) {
Box mybox1 = new Box();
Box mybox2 = new Box();
double vol;
vol = mybox1.volume();
System.out.println("Volume is " + vol);
vol = mybox2.volume();
System.out.println("Volume is " + vol);
}
}
```

Parameterized Constructors

```
class Box {
double width;
double height;
double depth;
Box(double w, double h, double d)
{
width = w;
height = h;
depth = d;
}
double volume() {
return width * height * depth;
}
}
```

```
class BoxDemo7 {
public static void main(String args[])
{

Box mybox1 = new Box(10, 20, 15);
Box mybox2 = new Box(3, 6, 9);

double vol;
vol = mybox1.volume();
System.out.println("Volume is " +
vol);
vol = mybox2.volume();
System.out.println("Volume is " +
vol);
}
}
```

“this” Keyword

- this can be used inside any method to refer to the current object.

```
Box() {
this.width = w;
this.height = h;
this.depth = d;
}
```

Garbage Collection

- objects are dynamically allocated by using the new operator, you might be wondering how such objects are destroyed and their memory released for later reallocation.

- Java handles deallocation automatically. The technique that accomplishes this is called *garbage collection.*