Working with objects

Mar. 28, 2017

Constructors

Initialising Objects

- Variables of a reference type have a special value before they are initialised
  - A “nothing” value called `null`
- Attempting to manipulate an object before its initialised will cause an error
  - A `NullPointerException`
- To properly initialise a reference type, we need to assign it a value by creating an object
  - Objects are created with the `new` operator

```
String someString = new String(“my String”);
```

Constructors

- `new` causes a *constructor* to be invoked
  - Constructor is a special method, used to initialise an object
  - Class often specifies several constructors (for flexibility)
  - `new` operator chooses right constructor based on parameters (*overloading*)
- Constructors can only be invoked by the `new` operator

Constructors – Example 1

```
public class MyClass
{
  private int x;
  public MyClass(int a)
  {
    x = a;
  }
}
```

We can then create an instance of MyClass as follows:

```
MyClass object = new MyClass(5);  //constructor is
   called
```

What are constructors for?

- Why do we use them?
  - Give us chance to ensure our objects are properly initialised
  - Can supply default values to member variables
  - Can accept parameters to allow an object to be customised
  - Can validate this data to ensure that the object is created correctly.

What are constructors for?

- A class *always* has at least one constructor
  - …even if you don't define it, the compiler will
  - This is the *default constructor*

Constructors – Example 2

```
public class EmailAddress
{
  public EmailAddress(String address)
  {
    if (address.indexOf(“@”) == -1)
    {
      //not a valid address, signal an error?
    }
    //must be valid…
  }

  //methods
}
```

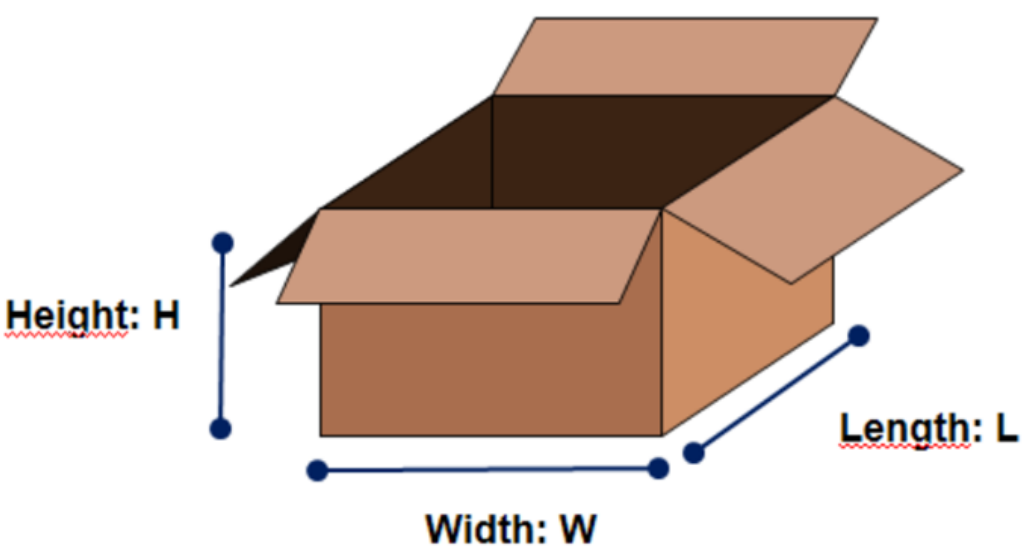
Several Constructors

- Constructors can call one another
  - Allows initialisation code to be kept in one place (not duplicated between constructors)
  - Use the `this` keyword to call another constructor in the same class
  - *Must* be first line in the constructor

```
public class Bookmark {
  public Bookmark(String url) {
     this( new URL(url) );
  }
  public Bookmark(URL url) {
    //other initialisation code
 }
}
```

LAB #4-2 Constructor

- Create a constructor in the following class




```
public class Box {
      private int width;
      private int length;
      private int height;
      private int volume;

      Box(??) {
              ??
       }
}
```

Destroying Objects

- No way to explicitly destroy an object
- Objects destroyed by the Garbage Collector
  - Once they go out of scope (I.e. no longer referenced by any variable)

Destroying Objects

- No way to reclaim memory, entirely under control of JVM
  - There is a `finalize` method, but its not guaranteed to be called (so pretty useless!)
  - Can request that the Garbage Collector can run, buts its free to ignore you