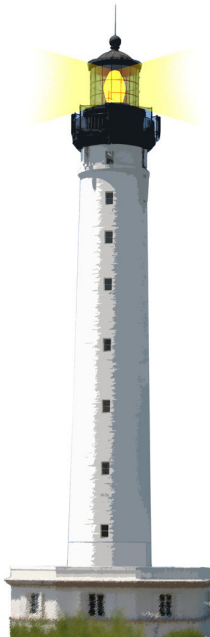
About Constructors

S. Ducasse



Pharo

Goal

- Constructors
-
-

Simple Constructor

```
public JButton() {
   this(null, null);
}
```

Constructor (I)

- Constructors may invoke each other.

```
public JButton(String text) {
   this(text, null);
```

```
public JButton(String text, Icon icon) {
   setModel(new DefaultButtonModel());
   init(text, icon);
}
```

- The keyword this is used for that purpose.

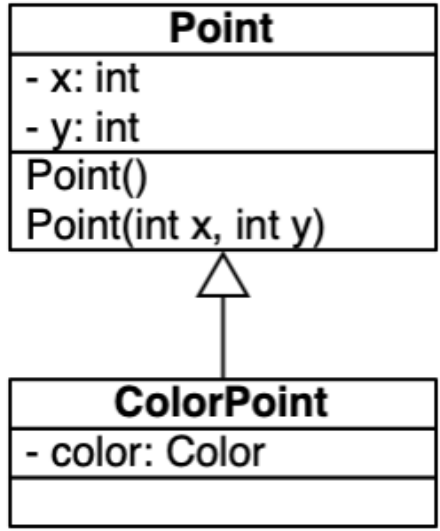
Note that this this, used in to invoke constructor, has nothing to do with the this pseudo variable used in method.

Constructors are not inherited

```
new Point()
new Point(2, 3)
=> Okay
```

```
new ColorPoint()
=> Okay (because of the
default constructor)

new ColorPoint(2, 3)
=> Does not compile
```

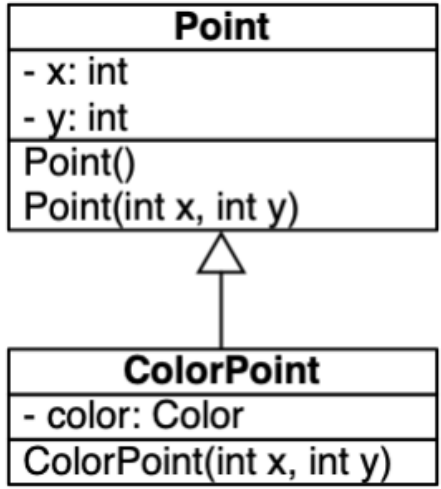


Missing default constructor

new Point()
new Point(2, 3)
=> Okay

new ColorPoint()
=> Does not compile
(because there is no
default constructor)

new ColorPoint(2, 3)
=> Okay



Constructors

```
class Rectangle {
    protected int length; protected int width;
    public Rectangle(){
      length = 0;
      width = 0; }
    public Rectangle(int length, int width) {
        this.length = length;
        this.width = width; }
```

```
class Box extends Rectangle {
    protected int height;
    public Box() {
       super();
       height = 0;
    }
    public Box(int length, int width, int height) {
        super(length, width);
        this.height = height;
}
```

About implicit constructor invocation

```
class Super {
   String s;
   public Super(){ System.out.println("Super"); }
}

public class Sub extends Super {
   public Sub(){
    super (); // implicitly added if not present
    System.out.println("Sub"); }
   public static void main(String[] args){
    Sub s = new Sub(); }
}

>>> Super
>>> Sub
```

About implicit constructor invocation

- If a constructor does not explicitly invoke a superclass constructor, the Java compiler automatically inserts a call to the no-argument constructor of the superclass.
- If the super class does not have a no-argument constructor, you will get a compile-time error.
- Object does have such a constructor, so if Object is the only superclass, there is no problem.
- The rules is: subclass constructor has to invoke super class constructor, either explicitly by programmer or implicitly by compiler.
- For either way, the invoked super constructor has to be defined.

Example of missing super constructor

- Implicit super constructor is undefined for default constructor.
- You must define an explicit constructor.

```
class Super {
   String s;
   public Super(String s) { this.s = s; }
}

public class Sub extends Super {
   public Sub(){
    System.out.println("Sub"); }

   public static void main(String[] args){
    Sub s = new Sub(); }
}
```

A course by

S. Ducasse, G. Polito, and Pablo Tesone