



DECLARACIÓN E IMPLEMENTACIÓN DE CLASES Y OBJETOS.

Febrero / 2020

PROGRAMACIÓN ORIENTADA A OBJETOS
INGENIERÍA INFORMÁTICA

Objetivos

Declaración e implementación de una clase en Java a partir de su diseño UML.

Creación y uso de instancias de clases.

Sumario

- ◇ Declaración e implementación de clases.
- ◇ Creación y usos de objetos o instancia de clases.

Bibliografía

- ◇ *Como programar en Java.*
- ◇ *Aprenda Java como si estuviera en primero.*

Motivación

Como podemos llevar el diseño UML de una clase a una implementación en Java ?

Motivación

Punto2D
-m_coordX : double -m_coordY : double
+Punto2D() +Punto2D(_x : double, _y : double) +getCoordX() : double +setCoordX(_coordX : double) : void +getCoordY() : double +setCoordY(_coordY : double) : void

Motivación

Circunferencia

-m_centro : Punto2D

-m_radio : double

+Circunferencia()

+Circunferencia(_radio : double, _x : double, _y : double)

+Circunferencia(_radio : double, _centro : Punto2D)

+getCentro() : Punto2D

+setCentro(_centro : Punto2D) : void

+getRadio() : double

+setRadio(_radio : double) : void

Declaración

La definición de una clase se realiza en la siguiente forma:

```
<visibilidad> class <NombreClase> {  
// definicion de atributos, constructores y metodos  
}
```


Declaración

Donde:

<**visibilidad**>: Hace referencia a la visibilidad de la clase que por lo general es **public**, pero puede ser también **private**, si no se pone, la clase tiene la visibilidad por defecto, esto es, sólo es visible para las demás clases del **package**.

Declaración

Donde:

<**NombreClase**>: Hace referencia al nombre de la clase.

Declaración

```
public class Punto2D {  
    // definicion de atributos, constructores y metodos  
}
```

Declaración

Atributos

Los atributos de una clase como buena práctica es lo primero que se define de ella.

Para su declaración primero se define su visibilidad que por lo general es **private** como buena práctica luego su tipo de dato y por último el nombre de dicho atributo.

Declaración

Atributos

```
public class Punto2D {  
    private double m_coordX;  
    private double m_coordY;  
    // definicion de constructores y metodos  
}
```

Declaración

Constructores

Un constructor es un método que se llama automáticamente cada vez que se crea un objeto de una clase. La principal misión del constructor es reservar memoria e inicializar las variables miembro de la clase.

Declaración

Constructores

Los constructores no tienen valor de retorno (ni siquiera **void**) y su **nombre es el mismo que el de la clase**. Su argumento implícito es el objeto que se está creando.

Declaración

Constructores

De ordinario una clase tiene varios constructores, que se diferencian por el tipo y número de sus argumentos (son un ejemplo típico de **métodos sobrecargados**). Se llama **constructor por defecto** al constructor que no tiene argumentos.

Declaración

Atributos y Constructores

```
public class Punto2D {  
    private double m_coordX;  
    private double m_coordY;  
  
    public Punto2D(){  
        this.m_coordX=0;  
        this.m_coordY=0;  
    }  
    public Punto2D(double _x, double _y){  
        this.m_coordX=_x;  
        this.m_coordY=_y;  
    }  
    // definicion de metodos  
}
```

Declaración

Métodos

Los métodos son funciones definidas dentro de una clase. Salvo los métodos **static** o de clase, se aplican siempre a un objeto de la clase por medio del operador punto (.). Dicho objeto es su argumento implícito.

Declaración

Métodos

Los métodos pueden además tener otros **argumentos explícitos** que van entre paréntesis, a continuación del nombre del método.

Declaración

Métodos

La primera línea de la definición de un método se llama **declaración** o **header**; el código comprendido entre las llaves ... es el **cuerpo** o **body**.

Declaración

Métodos

```
public class Punto2D {  
    // definicion de atributos y constructores  
    public double getCoordX(){  
        return this.m_coordX;  
    }  
    public void setCoordX(double _coordX){  
        this.m_coordX=_coordX;  
    }  
    public double getCoordY(){  
        return this.m_coordY;  
    }  
    public void setCoordY(double _coordY){  
        this.m_coordY=_coordY;  
    }  
}
```

Declaración

Objetos

Un objeto (en inglés, **instance**) es un ejemplar concreto de una clase. Las **clases** son como tipos de variables, mientras que los **objetos** son como variables concretas de un tipo determinado.

Declaración

Objetos

Para declarar un objeto de una clase es bastante parecido a como se declara una variable. Pero en este caso en vez del tipo de dato especificamos el nombre de la clase y luego el nombre que le daremos al objeto.

```
Punto2D p1;  
Punto2D p2,p3;
```

Instanciar Objetos

Cada vez que declaramos un objeto de una clase el mismo tiene valor **null**.

Esto provoca que cada vez que declaremos un objeto tengamos que reservar memoria en la computadora con el uso del operador **new** y uno de los constructores de la clase.

Instanciar Objetos

```
//coordenadas por defecto en el (0;0)
p1 = new Punto2D();
//coordenadas del punto son (9,3)
p2 = new Punto2D(9, 3);
//coordenadas del punto son (-1.5,4)
p3 = new Punto2D(-1.5, 4);
```

De igual forma se puede declarar e inicializar un objeto al mismo tiempo.

```
Punto2D p4 = new Punto2D();
Punto2D p5 = new Punto2D(-15,10);
```

Resumen

Objetos

El proceso de creación de objetos de una clase es el siguiente:

- ◇ Al crear el primer objeto de la clase o al utilizar el primer método o variable **static** se localiza la clase y se carga en memoria.
- ◇ Se ejecutan los **inicializadores static** (sólo una vez).

Resumen

Objetos

- ◇ Cada vez que se quiere crear un **nuevo objeto**:
 1. Se comienza reservando la memoria necesaria.
 2. Se da valor por defecto a las variables miembro de los tipos primitivos.
 3. Se ejecutan los inicializadores de objeto.
 4. Se ejecutan los constructores.

Estudio independiente

1. Diseñe el diagrama de clase en UML Triángulo a partir de sus 3 vértices. Determinar perímetro y área.
2. Implemente la clase Triángulo a partir del UML obtenido anteriormente.
3. Obtener aplicación de consola para crear un triángulo e imprimir su perímetro y área.

Análisis

Que se imprimirá por consola el siguiente código

```
Punto2D p5 = new Punto2D(-15,10);
Punto2D p6 = p5;

p5.setCoordX(100);
p6.setCoordY(100);

System.out.println("Las coordenadas de p5 son: (" + p5
    .getCoordX() + "; " + p5.getCoordY() + ")");

System.out.println("Las coordenadas de p6 son: (" + p6
    .getCoordX() + "; " + p6.getCoordY() + ")");
```

UNIVERSIDAD DE MATANZAS

cosechando el saber

FIN