Clases y Objetos en Java

Emmanuel Cruz Hernández
emmanuel_cruzh@ciencias.unam.mx

15 de octubre de 2020

Contenido

Clases

Una clase es una especificación genérica para un número arbitrario de objetos similares.
Las clases permiten construir una taxonomía de un nivel abstracto.

Si dos objetos presentan el mismo comportamiento, decimos que pertenecen a la misma clase.

Definición de clases

La declaración de una clase introduce un nuevo tipo de referencia del mismo nombre.
En una clase se definen los atributos y métodos que va a tener la clase.

Los **atributos** son los valores que debe recordar cada objeto, además describe el estado del objeto.

Los **métodos** son algoritmos paso a paso que se ejecutan como respuesta a un mensaje cuyo nombre es el mismo que el del método.

Declaración de clases

Una clase se introduce por la palabra reservada **class**, la cual se sigue de un identificador o nombre de la clase.
Por convención, el nombre de una clase siempre comienza con mayúscula seguido por minúsculas. Si se desea hacer una clase con más de una palabra, las palabras siguientes deben ir con mayúscula la primera palabra seguido de minúsculas.
El cuerpo de la clase se delimita entre llaves, y puede contener una serie de declaraciones de variables, métodos o clases.[1]

La forma más simple de declarar una clase en Java es la siguiente:

<M. de Acceso> **class** *<NombreDeLaClase>* {

    ...

}

Atributos de una clase

Las variables de instancia o **atributos** deben ser escogidos de tal forma que representen el estado de los objetos de una clase.
Pueden ser de cualquier tipo, ya sea primitivo o de referencia.
Por convención, una variable de instancia comienza con una letra minúscula.

Variables estáticas

Una variable estática pertenece a una clase y no es única de una sola instancia. Una variable estática es compartida por todas las instancias de una misma clase.
Una variable estática puede ser accesada por cualquier método o cualquier método estático del mismo entorno de clase.
Para declarar una variable estática o de clase se usa la palabra reservada **static**.

Por ejemplo:

```
class Clase1 {
    static int variableEstatica = 5;
}
```

Variables finales

Al definir un dato como constante o final le podremos asignar un valor por primera vez y luego de eso no será posible modificarlo.
Una vez inicializado el dato, este no podrá cambiar su valor.
Por convención sus nombres se escriben con letras mayúsculas y cada palabra es separada por un guión bajo.
Para declarar una variable constante se usa la palabra reservada **final**.

Por ejemplo:

```
class Clase2 {
    final int VARIABLE_CONSTANTE = 10;
}
```

Declaración de un Método

La declaración básica de un método es la siguiente:

<M. de Acceso> <Tipo de regreso> <Nombre> (*<Parámetros>*) {

    ...

}

Ejemplo de un método simple

```
public int suma ( int a, int b ) {
    return a + b;
}
```

- El modificador de acceso es público, representado por la palabra reservada **public**.
- El tipo de regreso es un entero representado por la palabra reservada **int**.
- El nombre del método es *suma*. Observe que esta no es una palabra reservada. De hecho, el nombre de los métodos no pueden ser palabras reservadas.
- En este caso, la lista de parámetros está conformado por dos elementos: a y b, ambos de tipo **int**.

Tipos de regreso

¿Puede un método regresar nada?

En Java existe una forma de especificar en un método que no se regresa un elemento con la palabra reservada **void**.
Si un método tiene como tipo de regreso **void**, no se debe poner la palabra reservada **return** dentro del método en cuestión para regresar un resultado.

En cambio, siempre que un método tenga en su firma un tipo de regreso, se debe usar la palabra reservada **return** para poder regresar un resultado (objeto) del mismo tipo que se especifica en la firma del método.

Tipo de regreso void

```
public void imprime ( ) {
    System.out.println( "Hola mundo");
}
```

Nótese que en este caso, la palabra reservada **return** no está presente en el cuerpo del método *imprime*.

Java también permite tener métodos sin parámetros.

Métodos Constructores

Un método constructor sirve para crear un nuevo objeto de un tipo específico.
La declaración de un método constructor no tiene tipo de retorno, y su nombre debe coincidir con el nombre de la clase dentro de la cual es declarado.

<M. de Acceso> <Nombre de la Clase> (*<Parámetros>*) {

    ...

}

Métodos de Acceso

Sirven para proporcionar información respecto al estado del objeto.

El nombre de estos métodos es get<*Nombre de atributo*>.

Regresan un valor del tipo del atributo que deseamos observar.

<*M. de Acceso*> <*Tipo del Atributo*> *get*<*Atributo*> ( ) {
    return atributo;
}

Métodos de manipulación o mutantes

Se usan para modificar el estado del objeto.

El nombre de estos métodos es set<*Nombre de atributo*>.

El tipo de regreso es **void**.

Además recibe un parámetro del tipo del atributo que se quiere modificar.

```
<M. de Acceso> void  set<Atributo> (<Tipo del atributo> atributo2) {
    this.atributo = atributo2;
}
```

Métodos de implementación

Sirven para implementar los servicios o tareas que ofrece un ejemplar de una clase.

El tipo de regreso y los parámetros que recibe dependen del servicio o problema que se quiera resolver.

Métodos auxiliares

Son métodos que sirven para resolver una sub-tarea. Es un auxiliar para resolver una parte de un problema general.

El tipo de regreso y los parámetros que recibe dependen del servicio o problema que se quiera resolver.

El tipo de acceso de estos métodos es privado. Representado con la palabra reservada **private**.

Modificadores de Acceso

El uso de los modificadores de acceso es opcional, y puede ser omitido. Las palabras reservadas que sirven como modificadores de acceso tienen el siguiente significado:

- **public** hace que una declaración sea accesible por cualquier clase.
- **protected** hace una declaración accesible por cualquier subclase de la clase que se declara, o a cualquier clase dentro del mismo paquete.
- **private** hace una declaración accesible sólo dentro de la clase en que se declara.

Objetos

Para crear un ejemplar u objeto en Java, se usa la palabra reservada **new** seguido de un método constructor.

Las instancias corresponden a los ejemplares que podemos exhibir de una clase dada.

Un mismo objeto puede pasar por distintos estados, esto depende del valor de los atributos que formen parte de un objeto.[2]

¿Cómo usar los métodos de un objeto?

Para mandar a llamar un método o servicio de una clase e utiliza el operador punto (.) seguido del nombre del método y entre paréntesis se pasan los parámetros, llamados argumentos.

El uso de los métodos dependerá del modificador de acceso que define el método.

<Nombre de la instancia de clase>.*<Nombre del método>* (*<Parámetros>*);

Ejemplo: Clase Mascota

Puedes consultar un ejemplo en el siguiente enlace:

`https://codenowprogramming.000webhostapp.com/entradas/Java/7-ejemploClase.php`

Método Main

Cuando una clase declara un método *main*, éste provee de un punto de inicio para la ejecución de un programa utilizando esa clase.

Un método *main* toma un argumento de tipo arreglo de String, llamado *args*, y contiene cualquier argumento del comando de línea.

El método main se ve como sigue:

public static void *main* (**String**[] *args*) {

    ...

}

Bibliografía

Jorge L. Ortega Arjona.
Notas de Introducción al Lenguaje de Programación Java.
2004.

Elisa Viso G. Canek Peláez V.
Introducción a las Ciencias de la Computación con Java.
Segunda edition, 2012.