APRENDERAPROGRAMAR.COM

¿QUÉ ES UNA CLASE JAVA? ATRIBUTOS (PROPIEDADES O CAMPOS), CONSTRUCTOR Y MÉTODOS. (CU00623B)

Sección: Cursos
Categoría: Curso “Aprender programación Java desde cero”

Fecha revisión: 2029

Resumen: Entrega nº23 curso Aprender programación Java desde cero.

Autor: Alex Rodríguez

¿QUÉ ES UNA CLASE JAVA? ATRIBUTOS, CONSTRUCTOR Y MÉTODOS

Hasta ahora hemos visto pequeños fragmentos de código de ejemplo. Vamos a tratar de escribir un código más ajustado a la realidad de la programación Java. Para ello vamos a definir de qué partes consta normalmente una clase Java. Las partes habituales las identificamos en este esquema:

Clase Taxi { ***--- > EL NOMBRE DE LA CLASE***

Propiedades: ***--- >También denominadas atributos o campos (fields)***
Matrícula identificativa
Distrito en el que opera
Tipo de motor diesel o gasolina

Constructor de la clase ***--- > Definición de qué ocurre cuando se crea un objeto del tipo definido por la clase***

Operaciones disponibles: ***--- > Métodos de la clase***
Asignar una matrícula
Asignar un distrito
Asignar un tipo de motor

}

Esto vamos a transformarlo en código usando un ejemplo. Para ello abre un nuevo proyecto en BlueJ y crea en él una clase denominada Taxi. Escribe en ella este código, aunque no entiendas algunas partes de él.

```
//Esta clase representa un taxi. -- > Comentario general que puede incluir: cometido, autor, versión, etc...
public class Taxi { //El nombre de la clase

   private String ciudad; //Ciudad de cada objeto taxi
   private String matricula; //Matrícula de cada objeto taxi
   private String distrito; //Distrito asignado a cada objeto taxi
   private int tipoMotor; //tipo de motor asignado a cada objeto taxi. 0=desconocido, 1 = gasolina, 2 = diesel

   //Constructor: cuando se cree un objeto taxi se ejecutará el código que incluyamos en el constructor
   public Taxi () {
      ciudad = “México D.F.”;
      matricula = "";
      distrito = "Desconocido";
      tipoMotor = 0;
   } //Cierre del constructor ... el código continúa ...
```

```
    //Método para establecer la matrícula de un taxi
    public void setMatricula (String valorMatricula) {
       matricula = valorMatricula; //La matrícula del objeto taxi adopta el valor que contenga valorMatricula
    } //Cierre del método

    //Método para establecer el distrito de un taxi
    public void setDistrito (String valorDistrito) {
       distrito = "Distrito " + valorDistrito; //El distrito del objeto taxi adopta el valor indicado
    } //Cierre del método

    public void setTipoMotor (int valorTipoMotor) {
       tipoMotor = valorTipoMotor; //El tipoMotor del objeto taxi adopta el valor que contenga valorTipoMotor
    } //Cierre del método

    //Método para obtener la matrícula del objeto taxi
    public String getMatricula () { return matricula; } //Cierre del método

    //Método para obtener el distrito del objeto taxi
    public String getDistrito () { return distrito; } //Cierre del método

    //Método para obtener el tipo de motor del objeto taxi
    public int getTipoMotor () { return tipoMotor; } //Cierre del método

} //Cierre de la clase
```

Pulsa el botón Compile y comprueba que no haya ningún error.

Repasemos lo que hemos hecho: hemos creado una clase denominada Taxi. El espacio comprendido entre la apertura de la clase y su cierre, es decir, el espacio entre los símbolos { y } de la clase, se denomina cuerpo de la clase.

Hemos dicho que todo objeto de tipo Taxi tendrá los mismos atributos: una matrícula (cadena de caracteres), un distrito (cadena de caracteres) y un tipo de motor (valor entero 0, 1 o 2 representando *desconocido*, *gasolina* o *diesel*). Los atributos los definiremos normalmente después de la apertura de la clase, fuera de los constructores o métodos que puedan existir.

Hemos definido que cualquier objeto Taxi que se cree tendrá, inicialmente, estos atributos: como matrícula una cadena vacía; como distrito “Desconocido”; y como tipo de motor 0, que es el equivalente numérico de *desconocido*. La sintaxis que hemos utilizado para el constructor es *public nombreDeLaClase { ... }*

Por otro lado, hemos establecido que todo objeto Taxi podrá realizar estas operaciones: recibir un valor de matrícula y quedar con esa matrícula asignada (setMatricula); recibir un valor de distrito y quedar con ese distrito asignado (setDistrito); recibir un valor de tipo de motor y quedar con ese valor asignado (setTipoMotor). Devolver su matrícula cuando se le pida (getMatricula); devolver su distrito cuando se le pida (getDistrito); devolver su tipo de motor cuando se le pida (getTipoMotor).

Para crear objetos Taxi pinchamos sobre el icono Taxi de la clase y con botón derecho elegimos new Taxi(). Nos aparece una ventana que nos pide el nombre del objeto. Crea 5 objetos Taxi denominados taxi1, taxi2, taxi3, taxi4 y taxi5. Cada objeto Taxi tiene tres atributos: matricula, distrito y tipoMotor. En total tendremos 5 taxis x 3 atributos = 15 atributos.

Hemos dicho que un objeto es una instancia de una clase: por eso a los atributos que hemos definido se les denomina "variables de instancia", porque cada instancia es "portadora" de esos atributos. También es frecuente utilizar el término "campos de la clase" como equivalente. Cada clase tendrá sus campos específicos. Por ejemplo, si una clase representa una moneda sus campos pueden ser *pais*, *nombreMoneda*, *valor*, *diametro*, *grosor*. Si una clase representa una persona sus campos pueden ser *nombre*, *apellidos*, *dni*, *peso* y *altura*.

¿Para qué nos sirve la clase? Para crear objetos de tipo Taxi. Por eso se dice que en Java una clase define un tipo. Recordamos ahora la definición de clase que habíamos dado previamente: "**Clase**: abstracción que define un tipo de objeto especificando qué propiedades y operaciones disponibles va a tener."

¿Por qué la clase, el constructor y los métodos se declaran *public* y los atributos *private*? Esto lo discutiremos más adelante. De momento, nos basta con saber que declararemos las clases, constructores y métodos precedidos de la palabra clave *public*, y que esta palabra afecta a en qué partes del programa o por parte de quién se va a poder acceder a ellos (igual que en el edificio con personas trabajando decíamos que una impresora podía tener un uso restringido a el personal de un departamento).

¿El orden campos → constructor → métodos es obligatorio? No, pero a la hora de programar hemos de ser metódicos y evitar el desorden. Muchos programadores utilizamos este orden a la hora de escribir clases, así que no está mal acostumbrarnos a seguir este orden.

¿Por qué en unos casos un método ocupa una línea y en otros varias líneas? Simple cuestión de espacio. Puedes escribirlo como quieras, siempre que quede bien presentado y legible. Hemos de tener claro que un método consta de dos partes: un encabezado o línea inicial y un cuerpo o contenido dentro de las llaves { }. En este curso muchas veces escribiremos métodos en una sola línea, o varias instrucciones en una sola línea, para ahorrar espacio. Sin embargo, en el trabajo como programadores el ahorro de espacio es poco relevante frente a la claridad. Lo importante es que el código sea claro.

¿Por qué establecemos el tipo de motor con un entero en vez de con un texto tipo String? A veces podemos definir las variables de diferentes maneras. En este caso nos resultaría también válido usar un String en vez de un int. Pero ten en cuenta una cosa: a los ordenadores les resulta más fácil analizar y manejar números que palabras. Si tenemos cien taxis en realidad no va a resultar demasiado importante elegir un texto o un número. Pero si tenemos cien mil sí puede ser relevante elegir un tipo numérico porque va a acelerar el procesado de datos.

EJERCICIO

Considera estás desarrollando un programa Java donde necesitas trabajar con objetos de tipo Persona. Define una clase Persona análoga a la que hemos visto para taxis, pero en este caso considerando los siguientes atributos de clase: nombre (String), apellidos (String), edad (int), casado (boolean), numeroDocumentoIdentidad (String). Define un constructor y los métodos para poder establecer y obtener los valores de los atributos. Compila el código para comprobar que no presenta errores. Para comprobar la corrección de tu solución puedes consultar en los foros aprenderaprogramar.com.

Próxima entrega: CU00624B

Acceso al curso completo en aprenderaprogramar.com -- > Cursos, o en la dirección siguiente:
http://www.aprenderaprogramar.com/index.php?option=com_content&view=category&id=68&Itemid=188