**Guía Sobrecarga de métodos**

|  |
| --- |
| //Nombre archivo: Main.java  public class Main {      public static void main(String[] args) {          Persona p1 = new Persona();          Persona p2 = new Persona("Alex");          Persona p3 = new Persona(20);          Persona p4 = new Persona("Alex", 20);            System.out.println("Nombre: " + p1.nombre + ", edad: " + p1.edad);          System.out.println("Nombre: " + p2.nombre + ", edad: " + p2.edad);          System.out.println("Nombre: " + p3.nombre + ", edad: " + p3.edad);          System.out.println("Nombre: " + p4.nombre + ", edad: " + p4.edad);            p1.unMetodo(15);          p1.unMetodo(15.5f);      }  } |

[?](http://ayudasprogramacionweb.blogspot.cl/2012/12/sobrecarga-metodos-constructores-java.html)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24 | //Nombre archivo: Persona.java  public class Persona {      public String nombre;      public int edad;        public Persona() {}      public Persona(String nombre) {          this.nombre = nombre;      }      public Persona(int edad) {          this.edad = edad;      }        public Persona(String nombre, int edad) {          this.nombre = nombre;          this.edad = edad;      }      public void unMetodo(int i) {          System.out.println("Un metodo con argumento entero");      }      public void unMetodo(float i) {          System.out.println("Un metodo con argumento real");      }  } |

public class Sobrecarga {

public static int quadrado(int num){

int quadrado;

quadrado = num \* num;

return quadrado;

}

public static double quadrado(double num){

double quadrado;

quadrado = num \* num;

return quadrado;

}

public static void main(String[] args){

System.out.println("Inteiro 2 ao quadrado: " + quadrado(2));

System.out.println("Double PI ao quadrado: " + quadrado( Math.PI ));

}

}

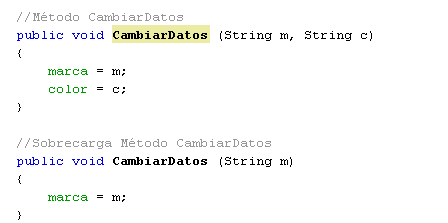
[Sobrecargar métodos y constructores](http://www.aulafacil.com/cursos/l13573/informatica/programacion/java-basico/sobrecargar-metodos-y-constructores)

Un método sobrecargado se utiliza para reutilizar el nombre de un método pero con diferentes argumentos. Las reglas para sobrecargar un método son las siguientes:

* Los métodos sobrecargados deben de cambiar la lista de argumentos.
* Pueden cambiar el tipo de retorno.
* Un método puede ser sobrecargado en la misma clase o en una subclase.

Vamos a verlo con ejemplos. Hagamos un método **sobrecargado** en la Clase Coche.

En el archivo Coche.java, debajo del último método que programamos, ObtenerColor(), introducimos lo siguiente:

**

Como se puede ver, hemos creado el método CambiarDatos.

Este método lo que hará es cambiar el color y marca del coche. Pero claro, ¿qué ocurre si solo queremos cambiar la marca, por ejemplo?

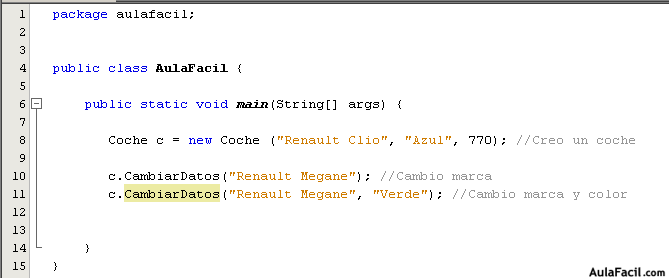
No nos serviría entonces dicho método...Para eso están las sobrecargas.

Lo que hemos hecho es el mismo método, con el mismo nombre, pero cambiando sus parámetros, por si solo se quiere cambiar la marca.

Y también se podría crear otras sobrecarga más, para cambiarle solo el color, por ejemplo.

Las sobrecargas pueden ser cuántas queramos.

Ahora vamos al archivo principal AulaFacil.java, y comprobamos como funciona:

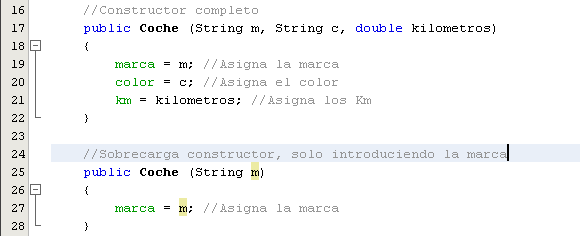
**

Como vemos, se puede utilizar el método de 2 formas diferentes, se le puede pasar un parámetro, o dos. Depende qué queramos conseguir con ello.

La sobrecarga de los métodos son bastante útiles.

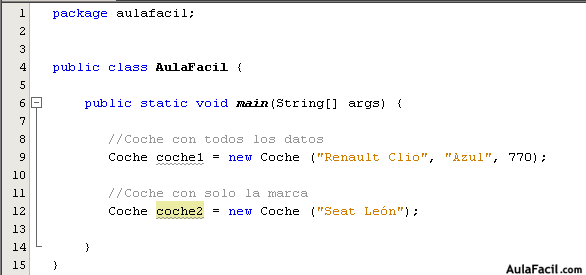
El tema de la sobrecarga también se puede usar en los constructores de la Clase. Como ya sabéis, el constructor es lo que se ejecuta cuando creamos el Objeto de dicha Clase.

Vamos a sobrecargar el constructor de la Clase Coche...Funciona igual que los métodos:

**

De esta forma, a la hora de crear un Coche, podemos hacerlo o bien dándole todos los datos del Coche, o bien dándole solo la marca.

Vamos a crear un Coche con cada constructor:

**

El uso es exactamente el mismo que para los métodos, pero en este caso con los constructores.

Esto cuando se trate de programar problemas más complejos puede ser realmente útil. Hay que tener en cuenta siempre el tema de la sobrecarga.