[Variables locales a un método o constructor en Java. Conflictos y sobrecarga de nombres](https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=494:variables-locales-a-un-metodo-o-constructor-en-java-conflictos-y-sobrecarga-de-nombres-cu00638b&catid=68&Itemid=188)

**VARIABLES LOCALES. SOBRECARGA DE NOMBRES.**

Una variable que se declara y se usa dentro de un método (o de un constructor) se dice que es una variable local. Su ámbito es sólo el método o constructor y su tiempo de vida es solo el del método, es decir, son **variables temporales que se crean cuando comienza a ejecutarse el método y se destruyen cuando termina** **de ejecutarse.**

**Ejemplo**

public class Estudiante { //El nombre de la clase

private String nombre; //Campo de los objetos Estudiante

//Constructor: cuando se cree un objeto Estudiante se ejecutará el código que incluyamos en el constructor

public Estudiante () {

nombre = "Pepe";

} //Cierre del constructo

//Método que devuelve true si el nombre del objeto tipo Estudiante es Pepe

public boolean esPepe() {

boolean seLlamaPepe = false;

if (nombre == "Pepe") { seLlamaPepe = true; }

return seLlamaPepe;

} //Cierre del método

} //Cierre de la clase

La variable seLlamaPepe es una variable local booleana. Es habitual inicializar las variables locales cuando se las declara, pero no es estrictamente necesario. Sí es obligatorio inicializar las variables en algún momento ya que no se debe considerar que tengan un valor por defecto. Crea un objeto de tipo Estudiante , invoca el método esPepe().

Ahora crea otro método y trata de establecer en él la variable seLlamaPepe con valor true. El compilador lanzará un mensaje de error del tipo “cannot find symbol – variable seLlamaPepe”. ¿Por qué podemos usar la variable nombre en cualquier método mientras que la variable seLlamaPepe sólo dentro del método esPepe()? Como hemos dicho, el ámbito de una variable declarada en un método es solo ese método. El resto de métodos no conocen la variable. En cambio, un campo de la clase tiene como ámbito toda la clase y lo podemos usar en cualquier lugar del código de la clase. Hay algunas conclusiones interesantes:

1) Podemos usar el mismo nombre de variable local en muchos métodos, puesto que no van a interferir entre ellas.

2) Una declaración de campo siempre la hacemos precedida de public o private. En las variables locales, estas palabras clave no se usan debido a su carácter temporal.

3) En los métodos tipo función con frecuencia en la sentencia return se devuelve como resultado el valor de una variable local que ha sido objeto de cálculo en el método. Tener en cuenta que no se devuelve la variable en sí (que en realidad desaparece cuando termina el método), sino su valor o contenido.

**¿Puede un constructor tener variables locales?** Sí. Un constructor son una serie de instrucciones. A veces muy sencillas, pero otras veces pueden requerir cálculos o procesos complejos. Por tanto, podemos usar variables locales dentro de ellos declarándolas y usándolas como si se tratara de un método. No tenemos restricciones en cuanto al código que se puede incluir en un constructor.

**¿Puede una variable local ser tipo objeto?** Sí. Hemos dicho que las clases definen tipos. Por ejemplo podríamos tener una variable local miTaxi1 declarada como Taxi miTaxi1;.

|  |  |  |
| --- | --- | --- |
|  | AMBITO | DURACIÓN |
| Campos(atributos de objetos) | Toda la clase | Indefinida (vida del objeto) |
| Parámetro de un método o constructor | Local (solo el método o constructor) | Temporal |
| Variables locales de un método o constructor. | Local (solo el método o constructor) | temporal |

Ejercicio

Considera estás desarrollando un programa Java donde necesitas trabajar con objetos de tipo Motor (que representa el motor de una bomba para mover fluidos). Define una clase Motor considerando los siguientes atributos de clase: tipoBomba (int), tipoFluido (String), combustible (String). Define un constructor asignando unos valores de defecto a los atributos y los métodos para poder establecer y obtener los valores de los atributos. Crea un método tipo función que devuelva un booleano (true o false) denominado dimeSiMotorEsParaAgua() donde se cree una variable local booleana motorEsParaAgua de forma que si el tipo de motor tiene la variable tipoBomba 1 y tipo de fluido es líquido tomará valor true y si no lo es tomará valor false. El método debe devolver la la variable local booleana motorEsParaAgua.