# ESTÁNDAR DE CÓDIGO

El conjunto de prácticas de codificación aceptadas se muestra a continuación, dichas prácticas aplican para el lenguaje de programación Java.

## General

Cada clase debe tener su propio archivo .java, en caso de que se tenga una clase interna será contada como una sentencia y no como una clase.

Al declarar una clase no es válido tener más de un espacio en blanco entre las palabras “class” y el nombre de la clase.

**No valido**

public class Persona {

}

**Valido**

public class Persona {

}

Una línea puede empezar con una palabra reservada solo sí es el inicio de una declaración o sentencia.

**No valido**

Persona persona =

new Persona (123,’Johann’);

**Valido**

Persona persona = new Persona (123,’Johann’);

## Cabecera

La cabecera para las clases que componen el programa debe estar delimitado por el comentario tipo bloque de código ( /\* … \*/ ) y se incluye antes de iniciar la clase.

## Comentarios

Los comentarios diferentes tipos de comentarios (“/\*\*…\*/”,“/\*…\*/”,”//”) deben iniciar y finalizar en líneas a parte del código que no corresponde a comentarios.

Ejemplo:

**No Valido**

Int AGENTES\_PERMITIDOS = 5 /\*Corresponde a la cantidad de agentes permitidos por campaña\*/

**Valido**

/\*Corresponde a la cantidad de agentes permitidos por campaña\*/

Int AGENTES\_PERMITIDOS = 5

## Declaración de variables

Se debe declarar una variable por línea.

Ejemplo:

**No valido**

Int mes,dia;

**Valido**

Int mes;

Int dia;

## Declaración de clases y métodos

Los símbolos que delimitan el cuerpo de una clase o de un método son “{“ y “}”. El corchete de apertura debe estar en la misma línea que la declaración y separado de la misma con un solo espacio en blanco y el de cierre debe estar en una línea a parte sin ninguna otra instrucción. Si el método recibe muchos parámetros o la declaración de la clase es muy larga entonces se acepta que su declaración ocupe más de una línea, pero las líneas siguientes que pertenecen a la misma declaración no pueden iniciar con una palabra reservada.

Al declarar un constructor se debe incluir un modificador de acceso.

Ejemplo:

**No valido**

public class CalculoVista

extends JFrame

{

/\* Cuerpo \*/

}

**Valido**

public class CalculoVista extends JFrame {

/\* Cuerpo \*/

}

Public class PanelBotonesRegistrarCurso extends

JPanel implements ActionListener{

}

## Instanciar una clase

Se debe realizar una instancia por línea. Si el constructor de la clase es muy largo entonces se puede separar por líneas, pero no es válido que la palabra reservada “new” sea la primera.

**No valido**

Persona usuario =

new Persona(123,’Johann’);

**Valido**

Persona usuario = new Persona(123,’Johann’);

## Labels

Para un labeled statement se debe separar el label hasta el símbolo “:” y la sentencia en una línea aparte.

Ejemplo:

**No valido**

Loop : for (int i = 0 ; i < 5 ; i++){

//Sentencias

}

**Valido**

Loop :

for (int i = 0 ; i < 5 ; i++){

//Sentencias

}

## Sentencias

Cada sentencia debe tener su propia línea o líneas independientes de otra sentencia.

Ejemplo:

NO valido

edad = actual + 1; bloques = 0;

Valido

edad = actual + 1;

bloques = 0;

## Estructuras IF o IF-ELSE, FOR, WHILE

La sentencia que se ejecuta debe estar en una línea a parte de la declaración de las estructuras. Y el corchete de cierre debe estar en una línea aparte. La palabra reservada (if,else,for, while, etc) debe estar separado de los demás caracteres por un espacio. Los símbolos que delimitan la estructura “{” y “}” debe estar en una línea a parte. Debe tener un espacio entre la palabra reservada (if,else,for,while ,etc) y el siguiente carácter si lo tiene.

Ejemplo :

No valido

If(condición) Sentencia;

Valido

If (condición){

Sentencia;

}

## Estructura SWITCH

Debe estar delimitada por (“{“ , “}”). Debe tener la siguiente estructura.

switch (Condición) {

case 1 :

Sentencias;

break;

case 2 :

Sentencias;

break;

default :

Sentencias;

break;

}

## TRY – CATCH

Debe estar delimitado por (“{“ , “}”) el corchete de apertura debe estar en la misma línea que el try separador por un espacio, el de cierre debe estar en una línea aparte así mismo para el catch.

**No valido**

try {

// Sentencias

} catch(Exception e) {

//Sentencias

}

**Valido**

try {

// Sentencias

}

catch (Exception e) {

//Sentencias

}

## RETURN

Para la instrucción de retorno en el caso de retornar un valor este debe estar en la misma línea de la instrucción return.

Ejemplo:

**No valido**

return

mes;

**Valido**

return mes ;