Universidad Andrés Bello®
Conectar · Innovar · Liderar

FUNDAMENTOS DE PROGRAMACIÓN EN JAVA

PROGRAMACIÓN BÁSICA EN JAVA

El entorno Java para la Programación

Estándares, convenciones y estilos de codificación

Como todo lenguaje, Java tiene su propio estilo de codificación. Los estilos de codificación(cómo tabular, dónde ubicar las llaves, cómo nombrar las variables, etc) ayudan a crear código que se vea consistente y sea fácil de interpretar por distintos desarrolladores. Un estilo de codificación no es una cuestión de gustos, sino una convención que ayuda a los desarrolladores.

En programación, seguir buenas prácticas, estándares y/o convenciones beneficia de granmanera la lectura, modificación y mantenimiento de los sistemas. Alrededor de un 80% del coste del código de un programa va a su mantenimiento, además de que casi ningún software lo mantiene toda su vida el autor original, por lo que se hace de vital importancia mantener un código ordenado y comentado, permitiendo entender código nuevo mucho más rápidamente y más a fondo.

Para que funcionen estas buenas prácticas, cada persona que escribe software deben seguirlas al pie de la letra.

Sangría

Como norma general se establecen 4 caracteres como unidad de sangría. Los entornos de desarrollo integrado (IDE) más populares, tales como Eclipse o NetBeans, incluyen facilidades para formatear código Java.

En el caso particular de Eclipse, para formatear o indentar el código, debe seleccionar la sección que desea formatear, y presionar la combinación de teclas Control + Shift + F, o bien ir al menú principal y seleccionar la opción “Source ⍰ Format”.

Longitud de línea

La longitud de línea no debe superar los 80 caracteres por motivos de visualización e impresión.

División de líneas

Cuando una expresión ocupe más de una línea, esta se podrá romper o dividir en funciónde los siguientes criterios:

- Tras una coma.
- Antes de un operador.
- Se recomienda las rupturas de nivel superior a las de nivel inferior.

- Alinear la nueva línea con el inicio de la expresión al mismo nivel que la línea anterior.
- Si las reglas anteriores generan código poco comprensible, entonces se deben establecer tabulaciones de 8 espacios.

Comentarios de implementación

Estos comentarios se utilizan para describir el código, y en ellos se incluye información relacionada con la implementación, tales como descripción de la función de variables locales, fases lógicas de ejecución de un método, captura de excepciones, etc. Se distinguen tres tipos de comentarios de implementación:

- **Comentarios de bloque:** Permiten la descripción de ficheros, clases, bloques, estructuras de datos y algoritmos.

```
/* * Esto es un comentario
* de bloque */
```

- **Comentarios de línea:** Son comentarios cortos localizados en una sola línea y tabulados al mismo nivel que el código que describen. Si ocupa más de una línea se utilizará un comentario de bloque. Deben estar precedidos por una línea en blanco.

```
/* Esto es un comentario de línea */
// Esto es otro comentario de línea
```

- **Comentario a final de la línea:** Comentario situado al final de una sentencia de código y en la misma línea.

```
int contador = 4 + 10; // Inicialización del contador
contador++; /* Incrementamos el contador */
```

Inicialización

Toda variable local tendrá que ser inicializada en el momento de su declaración, salvo quesu valor inicial dependa de algún valor que tenga que ser calculado previamente.

```
int idUnidad = 1;
String[] funciones = new String[]{ "Administración",
"Intervención", "Gestión" };
```

Declaración de clases

Durante la creación de clases en Java se deben seguir las siguientes reglas de formateo:

- No incluir ningún espacio entre el nombre del método y el paréntesis inicial del listado de parámetros.
- El carácter inicio de bloque "{" debe aparecer al final de la línea que contiene la sentencia de declaración.
- El carácter fin de bloque "}" se sitúa en una nueva línea tabulada al mismo nivel que su correspondiente sentencia de inicio de bloque, excepto cuando la sentencia sea nula, en tal caso se situará detrás de "{".
- Los métodos se separarán entre sí mediante una línea en blanco.

Espacios en blanco

Las líneas y espacios en blanco mejoran la legibilidad del código permitiendo identificar las secciones de código relacionadas lógicamente. Se utilizarán espacios en blanco en lossiguientes casos:

- Entre una palabra clave y un paréntesis. Esto permite que se distingan las llamadas a métodos de las palabras clave.

```
while (true) {
... }
```

- Tras cada coma en un listado de argumentos.

```
objeto.unMetodo(a, b, c);
```

- Para separar un operador binario de sus operandos, excepto en el caso del operador ("."). Nunca se utilizarán espacios entre los operadores unarios (p.e., "++" o "--") y sus operandos.

```
a += b + c;
a = (a + b) / (c + d);
contador++;
```

- Para separar las expresiones incluidas en la sentencia "for".

```
for (expresion1; expresion2; expresion3)
```

- Al realizar el moldeo o "casting" de clases.

```
Unidad unidad = (Unidad) objeto;
```

Clases e interfaces

En la creación de clases e interfaces, el programador debe tener presente los siguientespuntos:

- Los nombres de clases deben ser sustantivos, y deben tener la primera letra en mayúsculas.
- Si el nombre es compuesto, cada palabra componente deberá comenzar con maýusculas.
- Los nombres serán simples y descriptivos.
- Debe evitarse el uso de acrónimos o abreviaturas, salvo en aquellos casos en los que dicha abreviatura sea más utilizada que la palabra que representa (URL, HTTP,etc.).

UNIVERSIDAD ACREDITADA
Gestión Institucional
Docencia de Pregrado
Investigacion
Vinculación con el Medio
Por 5 años hasta diciembre 2022
CHE MSA Middle States Commission on Higher Education 3624 Market Street Philadelphia, PA 19104-2680
UNIVERSIDAD ACREDITADA 2015 - 2020
CHEA International Quality Group
UNAB forma parte de la Red LAUREATE INTERNATIONAL UNIVERSITIES
www.unab.cl

- Las interfaces se nombran siguiendo los mismos criterios que los indicados para las clases.
- Como norma general toda interfaz se nombrará con el prefijo "I" para diferenciarla de la clase que la implementa (que tendrá el mismo nombre sin el prefijo "I").

```
class Persona
class OrganigramaDAO
class AgendaService
```

Métodos

Los métodos deben ser verbos escritos en minúsculas. Cuando el método esté compuestopor varias palabras cada una de ellas tendrá la primera letra en mayúsculas.

```
public void insertaUnidad(Unidad unidad);
public void eliminaAgenda(Agenda agenda);
public void actualizaTramite(Tramite tramite);
```

Variables

Al igual que en los demás casos, sobre las variables igualmente aplican condiciones en sudefinición y uso. Estas reglas son las siguientes:

- Las variables se escribirán siempre en minúsculas.
- Las variables compuestas tendrán la primera letra de cada palabra componente en mayúsculas.
- Las variables nunca podrán comenzar con el carácter "_" o "$".
- Los nombres de variables deben ser cortos y sus significados tienen que expresar con suficiente claridad la función que desempeñan en el código.
- Debe ser un nemotécnico, es decir, diseñado para indicar al observador casual la intención de su uso.
- Se deben evitar los nombres de variable de un carácter, excepto para variables temporales. Los nombres comunes para las variables temporales son: i, j, k, m y n para enteros; c, d y e para los caracteres.

Contantes

Todos los nombres de constantes tendrán que escribirse en mayúsculas. Cuando losnombres de constantes sean compuestos las palabras se separarán entre sí mediante el carácter de subrayado "_".

```
int LONGITUD_MAXIMA;
int LONGITUD_MINIMA;
```

Convenciones

Clases e interfaces en Java

Los nombres de las clases deben ser sustantivos, en mayúsculas y minúsculas, con la primera letra de cada palabra interna en mayúscula. El nombre de las interfaces también debe estar en mayúscula (la primera letra) al igual que los nombres de las clases. Use palabras completas y debe evitar acrónimos y abreviaturas.
Generar la costumbre de escribir ciertas palabras en Inglés, es más fácil encontrar ayudaen foros o sitios en la Web. Además, mejorará el dominio con ese idioma.

Métodos en Java

Los métodos deben ser verbos, en mayúsculas y minúsculas, con la primera letra de cadapalabra interna (a partir de la segunda) en mayúscula.

Paquetes

Por defecto todos los paquetes se deben escribir en minúsculas, y sin utilizar caracteres especiales. El paquete base queda definido como cl.institucion, en donde el primer término es el código del país, y el segundo el nombre o sigla de la institución que desarrolla el proyecto.

Se tendrá, así mismo, otro nivel extra dentro del paquete definido como el nombre del proyecto o del módulo (Ej. cl.institucion.xxxx). Si existiera una parte común avarios de estos módulos, el nombre de los paquetes comenzarán por cl.institucion.comunes.

La estructura en árbol de los paquetes siguientes a partir de este último se define como sigue: (presuponiendo que se trata de una aplicación web multicapa).

- negocio

Anexo: Referencias

1.- Convenciones de nomenclatura en Java
Referencia: https://javadesdecero.es/fundamentos/convenciones-nomenclatura-java/

2.- Ejercicios de ejemplo con arreglos en Java
Referencia: http://puntocomnoesunlenguaje.blogspot.com/2012/07/arreglos-en-java-calcular-media.html