Convenciones de Código Java Cheat Sheet

by MNotoroid via cheatography.com/23208/cs/5072/

Extensiones de los ficheros

Tipo de fichero	Extensión
Fuente Java	.java
Bytecode	.class

Nombres de ficheros comúnes

Nombre de fichero	Uso
GNUmakefile	Usamos gnumake para construir nuestro software.
README	El nombre preferido para el fichero que resume los contenidos de un directorio particular.

Ficheros fuente Java

Cada fichero fuente Java contiene una única clase o interfaces pública. Cuando algunas clases o interfaces privadas están asociadas a una clase pública, pueden ponerse en el mismo fichero que la clase pública.

Comentarios de comienzo

Todos los ficheros fuente deben comenzar con un comentario en el que se lista el nombre de la clase, información de la versión, fecha, y copyright:

```
/*
* Nombre de la clase
*
* Informacion de la version
*
* Fecha
*
* Copyright
*/
```

Declaraciones

Inicia lizaci-ón	Intentar inicializar las variables locales donde se declaran. La única razón para no inicializar una variable donde se declara es si el valor inicial depende de algunos cálculos que deben ocurrir

Declaraciones (cont)

Colocación	Poner las declaraciones solo al principio de los bloques (un bloque es cualquier código encerrado por llaves "{" y "}".)
Declaraciones de class e interfaces	Al codificar clases e interfaces de Java, se siguen las siguientes reglas de formato: · Ningún espacio en blanco entre el nombre de un método y el paréntesis "(" que abre su lista de parámetros · La llave de apertura "{" aparece al final de la misma línea de la sentencia declaracion · La llave de cierre "}" empieza una nueva línea indentada para ajustarse a su sentencia de apertura correspondiente

Declaraciones de clases e interfaces

Partes de la declaración de una clase o interface	Los comentarios de documentación describen clases Java, interfaces, constructores, métodos y atributos.
Sentencia class o interface	En el encabezado se usa la palabra clave interface en lugar de class o abstract class. Por ejemplo public interface NombreDelInterface {…}

Declaraciones de clases e interfaces (cont)

Comentario de implementación de la clase o interface si fuera necesario (/.../)	Este comentario debe contener cualquier información aplicable a toda la clase o interface que no era apropiada para estar en los comentarios de documentación de la clase o interface.
Variables de clase (static)	Primero las variables de clase public, después las protected, después las de nivel de paquete (sin modificador de acceso) , y despúes las private.
Variables de instancia	Primero las public, después las protected, después las de nivel de paquete (sin modificador de acceso), y después las private.
Constructores	Cuando se construye un objeto es necesario inicializar sus variables con valores coherentes, imaginemos un objeto de la clase Persona cuyo atributo color de pelo al nacer sea verde, un estado incorrecto tras construir el objeto persona.

Declaraciones de clases e interfaces (cont)

Métodos	Estos métodos se deben agrupar por funcionalidad más que por visión o accesibilidad. Por ejemplo, un método de clase privado puede estar entre dos métodos públicos de instancia.

Sentencias

Sentencias simples	Cada línea debe contener como mucho una sentencia. Ejemplo: argv++; // Correcto argc--; // Correcto argv++; argc--; // EVITAR!
Sentencias compuestas	Las sentencias compuestas son sentencias que contienen listas de sentencias encerradas entre llaves "{ sentencias }".
Sentencias de retorno	Una sentencia return con un valor no debe usar paréntesis a menos que hagan el valor de retorno más obvio de alguna manera. Ejemplo: return; return miDiscoDuro.size(); return (tamanyo ? tamanyo : tamanyoPorDefecto);

Sentencias (cont)

Sentencias if, if-else, if else-if else	La clase de sentencias if-else debe tener la siguiente forma: if (condicion) { sentencias; } if (condicion) { sentencias; } else { sentencias; } if (condicion) { sentencia; } else if (condicion) { sentencia; } else{ sentencia; }
Sentencias for	Una sentencia for debe tener la siguiente forma: for (inicializacion; condicion; actualizacion) { sentencias; }
Sentencias while	Una sentencia while debe tener la siguiente forma: while (condicion) { sentencias;
Sentencias do-while	Una sentencia do-while debe tener la siguiente forma: do { sentencias; } while (condicion);
Sentencias switch	Una sentencia switch debe tener la siguiente forma: switch (condicion) { case ABC: sentencias; / *este caso se propaga* / case DEF: sentencias; break; case XYZ: sentencias; break; default: sentencias; break; }

Sentencias (cont)

Sentencias try-catch	Una sentencia try-catch debe tener la siguiente forma: try { sentencias; } catch (ExceptionClass e) { sentencias; }

Indentación

Longitud de la línea
Evitar las líneas de más de 80 caracteres, ya que no son manejadas bien por muchas terminales y herramientas.
Rompiendo líneas
Cuando una expresión no entre en una línea, romperla de acuerdo con estos principios:
· Romper después de una coma.
· Romper antes de un operador.
· Preferir roturas de alto nivel (más a la derecha que el "padre") que de bajo nivel (más a la izquierda que el "padre").
· Alinear la nueva linea con el comienzo de la expresión al mismo nivel de la linea anterior.
· Si las reglas anteriores llevan a código confuso o a código que se aglutina en el margen derecho, indentar justo 8 espacios en su lugar.

Comentarios

Los programas Java pueden tener dos tipos de comentarios: comentarios de implementación y comentarios de documentación. Los comentarios de implementación son aquellos que también se encuentran en C++, delimitados por /.../, y //. Los comentarios de documentación (conocidos como "doc comments") existen sólo en Java, y se limitan por /*.../. Los comentarios de documentación se pueden exportar a ficheros HTML con la herramienta javadoc.

Cheatography

Convenciones de Código Java Cheat Sheet

by MNotoroid via cheatography.com/23208/cs/5072/

Formatos de los comentarios de implementación

Comentarios de bloque	Los comentarios de bloque se usan para dar descripciones de ficheros, métodos, estructuras de datos y algoritmos.
Comentarios de una linea	Pueden aparecer comentarios cortos de una única línea al nivel del código que siguen. Si un comentario no se puede escribir en una línea, debe seguir el formato de los comentarios de bloque.
Comentarios de remolque	Pueden aparecer comentarios muy pequeños en la misma línea que describen, pero deben ser movidos lo suficientemente lejos para separarlos de las sentencias.
Comentarios de fin de linea	El delimitador de comentario // puede convertir en comentario una línea completa o una parte de una linea.
Comentarios de documentación	Los comentarios de documentación describen clases Java, interfaces, constructores, métodos y atributos.

Convenciones de nombres

Paquetes	El prefijo del nombre de un paquete se escribe siempre con letras ASCII en minúsculas, y debe ser uno de los nombres de dominio de alto nivel, actualmente com, edu, gov, mil, net, org, o uno de los códigos ingleses de dos letras que identifícan cada país como se especifica en el ISO Standard 3166, 1981.	com.sun.eng com.apple.quicktime.v2 edu.cmu.cs.bovik.cheese

Convenciones de nombres (cont)

Clases	Los nombres de las clases deben ser sustantivos, cuando son compuestos tendrán la primera letra de cada palabra que lo forma en mayúsculas. Intentar mantener los nombres de las clases simples y descriptivos.	class Cliente; class ImagenAnimada;
Interfaces	Los nombres de las interfaces siguen la misma regla que las clases.	interface ObjetoPersistente; interface Almacen;
Métodos	Los métodos deben ser verbos, cuando son compuestos tendrán la primera letra en minúscula, y la primera letra de las siguientes palabras que lo forma en mayúscula.	ejecutar(); ejecutarRapido(); cogerFondo();

Convenciones de nombres (cont)

Variables	Excepto las constantes, todas las instancias y variables de clase o método empezarán con minúscula. Las palabras internas que lo forman (si son compuestas) empiezan con su primera letra en mayúsculas. Los nombres de variables no deben empezar con los caracteres subguión "_" o signo del dolar "$", aunque ambos estan permitidos por el lenguaje.	int i; char c; float miAnchura;
Constantes	Los nombres de las variables declaradas como constantes deben ir totalmente en mayúsculas separando las palabras con un subguión ("_").	static final int ANCHURA_MINIMA = 4; static final int ANCHURA_MAXIMA = 999; static final int COGER_LA_CPU = 1;