Algunas reglas de estilo para la programación en Java

LOS PROGRAMAS INFORMÁTICOS deben escribirse de forma que puedan ser leídos, comprendidos y modificados por lectores humanos. El estilo de programación es importante, y la mayoría de los programas se evalúan tanto por su estilo como por su corrección. Este texto enumera algunas reglas básicas de estilo de programación que debe tener en cuenta al escribir sus programas.

La regla principal 0.

Un programa debe ser legible. Todas las demás reglas de esta página pueden ser anuladas por esta regla principal. Tus programas deben mostrar buen gusto, que generalmente sólo puedes adquirir prestando atención a las prácticas de otros programadores expertos.

Comentarios

1. Cada clase (excepto las clases internas anónimas) debe tener un comentario Javadoc que especifique el propósito de la clase y describa su interfaz pública en términos generales. El comentario principal de una clase debe incluir su nombre.
2. Cada método, a menos que se declare privado, debe tener un comentario Javadoc que explique lo que hace. El comentario debe incluir una declaración de las condiciones que deben cumplirse cuando se llama al método, como restricciones sobre los valores aceptables de sus parámetros. (El propósito de cada parámetro debe estar claramente documentado. Si existe un valor de retorno, debe documentarse su significado. Si el método puede lanzar alguna excepción, es una buena idea documentarla también. Se recomienda utilizar las etiquetas @param, @return y @throws para la documentación pertinente.
3. Cada variable que tenga un papel no trivial en el programa debe tener un comentario que explique su propósito. Esto incluye tanto las variables miembro como las variables locales. Para las variables miembro no privadas, los comentarios deben estar en formato Javadoc. Las variables de bucle for y otras variables locales de utilidad no tienen, en general, que ser comentadas.
4. Los comentarios pueden incluirse en el cuerpo de un método cuando sean necesarios para explicar la lógica del código. En general, el código bien escrito necesita pocos comentarios.
5. Los comentarios nunca deben utilizarse para explicar el lenguaje Java. Un comentario como "declare una variable int llamada x" o "incremente la variable ct" es peor que inútil. Asuma que su lector conoce Java. (Tenga en cuenta que este tipo de comentarios se utilizan a veces en los libros de texto de programación o en la pizarra, pero nunca en programas reales).

Formato

6. Utilice la sangría para mostrar la estructura de su programa. El cuerpo de la definición de una clase debe ir sangrado. El cuerpo de la definición de un método debe estar sangrado. Cuando una sentencia está anidada dentro de otra sentencia, debe sangrarse un nivel adicional. (Tenga en cuenta que los entornos de desarrollo pueden corregir la sangría de cualquier segmento de código: Basta con resaltar el segmento de código y pulsar una combinación de teclas).

7. Un "}" final debe estar en una línea por sí mismo. El "{" de apertura puede estar al principio de una línea, o puede colocarse en una línea por sí mismo de modo que se alinee con el "}" correspondiente. Ambas opciones son válidas, pero sea cual sea la que utilices, sé coherente.
8. No pongas más de una sentencia en una línea.
9. Evita las líneas muy largas. En general, las líneas no deben superar los 80 caracteres. Las declaraciones más largas deben dividirse en varias líneas.
10. Evite el anidamiento muy profundo de sentencias. Si utiliza más de dos o tres niveles de anidamiento, piense en definir subrutinas para dividir el código en partes más manejables.
11. Los espacios en blanco y las líneas en blanco pueden facilitar la lectura de un programa. En general, debería poner algunas líneas en blanco entre las definiciones de métodos. Deje espacios en blanco alrededor de operadores como =, ==, !=, etc.

Nombres

12. Utilice nombres significativos para sus variables, métodos y clases.
13. Siga la convención habitual de Java para el uso de mayúsculas: Los nombres de variables, métodos y paquetes comienzan con minúsculas. Los nombres de clases comienzan con mayúsculas. Cuando un nombre contenga más de una palabra, escriba en mayúsculas las palabras adicionales, como en "interestRate".
14. Utilice "final static" para declarar constantes con nombre que representen datos constantes. Considere el uso de "enum" para crear varias constantes relacionadas. Las constantes suelen tener nombres completamente en mayúsculas, con palabras separadas por caracteres de subrayado.

Métodos

15. Un método debe tener una tarea clara, única e identificable.
16. La definición de un método individual no debe ser demasiado larga. En general, una función no debe ser más larga que una página impresa (y eso es exagerar).
17. Los métodos de instancia pueden acceder a las variables de instancia que representan el estado de un objeto, pero debes evitar usar variables de instancia simplemente para pasar información de un método a otro - para eso, debes usar parámetros y valores de retorno.

Clases

18. Una clase debe representar un concepto claro, único e identificable.
19. Utilice los modificadores public, protected y private para controlar el acceso a las variables y métodos de una clase.
20. En general, las variables miembro deben declararse privadas. Se pueden proporcionar métodos getter y setter para acceder y manipular las variables miembro privadas.