AS NORMAS DE ESCRITA EM JAVA

António Fábio Da Piedade Pedro Cuhema

Faculdade Integradas De Camões

Tecnologia em analise e desenvolvimento de sistemas

Caixa Postal-80530-290-Curitiba-Paraná-Brasil

RESUMO

A adoção de Normas e convecções de escrita em Java é um fator fundamental para o desenvolvimento de softwares que atendam requisitos de segurança. Práticas gerais de programação segura têm sido divulgadas, porém sua aplicação específica à linguagem Java nem sempre é pertinente. Por outro lado, as informações disponíveis sobre normas na linguagem Java são dispersas e não sugerem a aplicação prática de técnicas de programação segura. Este artigo tem como objetivo compilar e classificar um conjunto de normas de programação em Java, que permita a sua aplicação prática imediata tanto num contexto educacional e de treinamento, quanto num contexto profissional.

NORMAS E CONVENÇÕES DE ESCRITA EM JAVA:

As normas e convenções para escrita possibilitam a criação de um código padronizado de melhor entendimento por parte dos programadores que o utilizem.

Comentários:

Java possui 2 tipos de comentários:

Os comentários da implementação (implementation comments) que são delimitados por /*.....*/ e // e os destinados a documentação (doccomments) existentes apenas na linguagem Java e são delimitados por /**.....*/.
Os comentários da implementação devem conter informações sobre a implementação da classe em particular.
Os comentários para documentação (javadoc) deve descrever as especificações do código que deverá ser lida por outros programadores que não necessitam ter o arquivo fonte em mãos.

NORMAS GERAIS DE ESCRITA DO ARQUIVO FONTE JAVA

ORDEM GERAL:

- Comentários iniciais.
- Pacotes e importações (imports)
- Declaração das classes e interfaces.

Comentários Iniciais:

Recomenda-se iniciar o arquivo com um comentário contendo o nome da classe, informações sobre a versão, data e direitos autorais.

Pacotes e Importações:

O código fonte começa de verdade com as declarações dos pacotes (packages) seguidas das declarações das importações (imports).

Declaração das classes e interfaces:

- Comentários para a documentação da classe/interface desejada (javadoc)
- Declaração da classe/interface
- Comentários da implementação da classe/interface (deve conter qualquer informação que não é apropriada para a documentação)
- Declaração das variáveis static da classe, primeiramente as publicas, as protegidas (ptotected), as do nível dos pacotes e então as privadas (private)
- Declaração das vaiáveis de instância, na mesma ordem que as variáveis static da classe (citado acima)
- Determinação do construtor
- Definições dos métodos, que devem ser agrupados de acordo com a funcionalidade e nível de acesso.

Inicialização

Tente inicializar as variáveis no mesmo local em que elas são declaradas, não ser que seu valor dependa de algum cálculo posterior.
Ponha as declarações das variáveis apenas no inicio dos blocos, estruturas delimitadas pelas chaves { }, não espere para declarar apenas quando for utilizá-las.

DECLARAÇÕES:

Número por linhas:

É recomendável uma declaração de variável por linha, mesmo elas sendo do mesmo tipo.
Nunca defina diferentes tipos de variáveis numa mesma linha.

Declaração de classes e interfaces:

Tem as seguintes regras:

- Nenhum espaço entre o nome do método e o parênteses
- O braço de abertura "{"é escrito na mesma linha da declaração
- O braço de fechamento "}" tem uma própria linha e é indentado na mesma coluna que o inicio do método correspondente ao seu bloco.Exceto quando o bloco esta vazio, tendo que aparecer logo em seguida ao braço de abertura "{}".

Exemplo:

```
Class exemplo extends Object
{
      Int var1;

      Exemplo (int i)
   {
            var1 = i;
    }

      int metodoVazio () {}
}
```

NORMAS DE ESTILO:

INDENTAMENTO:

4 espaços equivale a uma unidade do indentamento.

Tamanho da Linha:

As linhas devem ter em torno de 80 caracteres sendo que as linhas de comentário destinadas a documentação não devem possuir mais de 70 caracteres.

Quebra de Linha:

Se necessário quebrar uma linha:

- Quebre antes de um comando
- Quebre depois de um operador
- Alinhar a nova linha com o inicio da expressão, no mesmo nível, da linha anterior
- Se as regras acima levar a um código confuso ou muito próximo da margem direita deve-se indentar 8 espaços.

As linhas quebradas para na declaração das condições do if deve utilizar a regra do 8 espaços para indentação. Exemplo:

```
if ((condition1 && condition2)
        || (condition3 && condition4)
        || (condition5 && condition6)) {
   dosomething;
}
```

LINHAS EM BRANCO

Deve-se deixar uma linha em branco entre:

- Métodos
- Variáveis locais de um método e sua primeira declaração
- Depois de um bloco ou uma linha de comentário.

Deve-se deixar 2 linhas em branco entre:

- Definições das classes e interfaces
- Entre secções do source file.

ESPAÇOS EM BRANCO

Deve ser usado:

- Entre uma palavra chave seguido de parênteses
- Em todos os operadores envolvendo operações com dois ou mais termos, o que não se aplica as operações envolvendo um só termo. Exemplo: incremento, decremento
- Entre as declarações em um for

- Entre os operadores de coerção (casts) e o termo que esta sendo modelado.

ESTRUTURAS DE CONTROLE:

Declaração do if, if-else, if else-if else:

Tem a seguinte forma:

```
if (condição)
 {
   Declaração;
}

if  (condição)
{
   Declaração;
}
else
 {
   Declaração;
 }

if  (condição)
{
   Declaração;
} else if (condição)
 {
   Declaração;
}
else
{
   Declaração;
}
```

Declaração do for:

Tem a seguinte forma:

```
for (inicio; condição; incremento/decremento)
{
   Declaração;
}
```

Caso o bloco do for seja vazio tem a seguinte forma:

for (inicio; condição; incremento/decremento);

Declaração do while:

Tem a seguinte forma:

```
while (condição)
 {
   Declaração;
}
```

Caso o bloco do while seja vazio tem a seguinte forma:

while (condição);

Declaração do do-while:

Tem a seguinte forma:

```
Do
 {
   Declaração;
} while (condição);
```

Declaração do switch:

Tem a seguinte forma:

```
switch (condição)
 {
case a:
    Declaração;
    break;
case b:
    Declaração;
    break;
}
```

Declaração try-catch:

Tem a seguinte forma:

```
try
{
    Declaração;
} catch (Exceptionclass e)
 {
    Declaração;
}
```

Pode-se utilizar também o finally:

```
try {
    Declaração;
}
 catch (Exceptionclass e)
 {
    Declaração;
}
finally
{
    Declaração;
}
```

CONVENÇÕES DE NOMENCLATURA:

Todo nome composto pela união de dois ou mais nomes a primeira letra do primeiro nome pode ou não ser maiúscula mais a primeira letra dos demais nomes sempre será maiúscula.
Recomenda-se não utilizar acentuação.

TIPOS	REGRAS	EXEMPLOS
Classes	Devera ser um nome, de preferência simples. Deve começar com letra maiúscula	class Raster; class ImageSprite;
Interfaces	Possui a primeira letra maiúscula assim como a classe	interface Storing;
Métodos	Deveram ser verbos com a primeira letra sempre minúscula	run(); runFast();
Variáveis	Devem começar com letra minúscula, sublinhado ou cifrão, devem ser nomes curtos indicando a um observador sua utilidade. Nomes com apenas um caracter deve ser evitado a menos que represente uma variável descartável.	int i; float myWidth;
Constantes	Deve ser escrita sempre em maiúsculo com o sublinhado separando os nomes	final int MIN_WIDTH = 4;

Referencias

Devi-Media: **Convenções de nomenclatura**
https://www.devmedia.com.br/convencoes-de-codigo-java/23871
Devi-Media: **Normas de escrita de arquivos em java**
https://www.devmedia.com.br/leitura-e-escrita-de-arquivos-de-texto-em-java/25529
Nação Livre: **Estrutura de controle**
http://www.nacaolivre.com.br/java/estruturas-de-controle-em-java/
Devi-Media: **Declaração de variaveis**
https://www.devmedia.com.br/java-declaracao-e-utilizacao-de-classes/38374