Sistema de Biblioteca (SIB)

CONVENÇÃO DE FORMATAÇÃO

Convenções de Código Java

As convenções de código são uma boa prática de programação. Ajudam por exemplo na manutenção de um sistema. A violação dessas “regras” prejudica em muito o entendimento de uma API.

As convenções de código são uma boa prática de programação. Ajudam por exemplo na manutenção de um sistema. A violação dessas “regras” prejudica em muito o entendimento de uma API, por exemplo. Além do mais, com as convenções, se é capaz de produzir um código legível, “limpo”, que demonstra qualidade e de maneira profissional.
Abaixo descrevemos algumas das principais convenções Java:

Nomes de Arquivo

- Extensão de Arquivos

.java - Java source
.class - Java bytecode

- Organização dos arquivos

Um arquivo consiste de seções que devem ser separadas por linhas em branco e opcionalmente de
um comentário identificando cada seção.

Evite arquivos de código com mais de 20.000 linhas pois ficam pesados.

- Arquivos de código Java

Cada arquivo de código Java contem uma única classe publica ou interface. Quando classes privadas e
interfaces são associadas com uma classe publica, você pode colocá-los no mesmo no mesmo arquivo de
código da classe publica. A classe pública deve ser a primeira classe ou interface no arquivo.

Arquivos de código Java devem seguir a ordem:
- Comentário Inicial;
- Declaração do Package e Imports;
- Declaração de classes e interfaces;

Pacotes:

Deve obedecer a certa hierarquia, separadas por pontos. O uso de letras minúsculas é bem vindo, e raramente se faz usos de números. Se o pacote a ser utilizado é de fora da companhia ou organização, este deve começar com o domínio de internet redigido de forma contrária. Por exemplo, o pacote com.sun. Essas “partes” devem ser curtas, com até oito caracteres.
Em relação a bibliotecas padrão e pacotes opcionais, cujos nomes começam com java e javax, são exceções a esta regra e não são permitidos esses nomes para uso do desenvolvedor.
São permitidas abreviações, como por exemplo, o pacote util ao invés de utilities. O uso de acrônimos também é visto com bons olhos.
A primeira linha não comentada da maioria dos arquivos de códigos Java é a declaração do Package. Depois disso, é seguido da declaração dos imports. Por exemplo:

```
1.  package java.awt;
2.  import java.awt.peer.CanvasPeer;
```

Classes:
Quando escrevermos classes e interfaces Java, as seguintes regras devem ser seguidas:

- Sem espaço entre o nome do método e os parênteses que inicia a sua lista de parâmetros
- Abrir colchetes "{" aparece no final da mesma linha que o comando de declaração.
- Fecha colchetes "{" inicia em uma linha própria recuando para combinar com a abertura de colchetes, da
correspondente declaração, exceto quando a declaração é nula neste caso o "}" deve seguir imediatamente
depois do "}".

```
1.  class Sample extends Object {
2.     int ivar1;
3.     int ivar2;
4.
5.     Sample(int i, int j) {
6.        ivar1 = i;
7.        ivar2 = j;
8.     }
9.
10.         int emptyMethod() {}
11.         ...
12. }
```

Métodos e Atributos:

Devem seguir as mesmas convenções tipográficas de classes e interfaces, exceto que a primeira letra deve ser minúscula. Os métodos normalmente utilizam verbos ou frases verbais. Métodos que retornem um tipo boleano devem usar prefixo "is" seguido de um substantivo ou adjetivo, por exemplo, isEmployee.

Já os que retornam o resultado de uma função ou um atributo do objeto são nomeados com substantivos com ou sem o prefixo "get".

Em relação a JavaBeans, o uso do get e set (método para atribuir o valor a um atributo do objeto) são obrigatórios.

As variáveis locais seguem as mesmas convenções de métodos e campos, mas abreviações são permitidas.

Comentários:

A seguinte tipografia é uma das mais usadas:

/*
* *Classname*
*
* *Version info*
*
* *Copyright notice*
*/

Mas lembre-se de consultar a documentação Sun para a geração via comentários do popular e já tradicional Javadoc.

Obs: A maioria dos IDES do mercado tem configuração seguindo as convenções Java e notificam ao programador de eventuais “erros” durante a atividade de desenvolvimento.

Práticas de Programação

- Fornecer acesso a variáveis de instancia e de classes;
  Não faça qualquer instancia ou variável de classe public, sem uma boa razão.

- Referindo-se a variáveis e métodos de classe;
  Evite usar um objeto para acessar uma classe (static) ou método. Use um nome de classe em vez disso. Por exemplo:

```
1. classMethod();          //OK
2. AClass.classMethod();       //OK
3. anObject.classMethod();          //EVITAR!
```

REFERENCIAS

Java Code Convertions, Sun http://java.sun.com/docs/codeconv/CodeConventions.pdf