



**Folha de estilos**

Padrão de Codificação em Java

Versão 1.5

01/09/2014

Grupo:

Emilie Trindade de Morais

Ítalo Paiva Batista

Leonardo Cambraia Corrêa

Tiago Ribeiro de Assunção

Disciplina:

Técnicas de Programação

Ministrada pelo Professor Maurício Serrano

**CONTEÚDO**

[1. ARQUIVOS](#h.gjdgxs)

[1.1 Pacotes (package statement)](#h.30j0zll)

[1.2 Importações (import statement)](#h.1fob9te)

[1.3 Declaração de Classes](#h.3znysh7)

[1.3.1 Ordem dos membros da classe](#h.2et92p0)

[2. VÁRIAVEIS](#h.tyjcwt)

[3. MÉTODOS](#h.3dy6vkm)

[3.1 Variáveis Locais](#h.1t3h5sf)

[4. COMANDOS](#h.2s8eyo1)

[4.1 Estruturas Condicionais](#h.17dp8vu)

[4.2 Estruturas de repetição](#h.3rdcrjn)

[4.3 Tratamento de exceções](#h.26in1rg)

[5. ESPAÇAMENTOS E IDENTAÇÃO](#h.lnxbz9)

[5.1 Regras Gerais](#h.35nkun2)

[5.2 Linhas em branco](#h.1ksv4uv)

[5.3 Espaços em branco](#h.44sinio)

5.4 Quebras de linha

[6. COMENTÁRIOS](#h.2jxsxqh)

1. **ARQUIVOS**

* O nome do arquivo deve ser o mesmo nome da classe pública que o arquivo abriga.
* O arquivo deve conter, em ordem:
  1. Declaração dos pacotes;
  2. Declaração das importações;
  3. Somente uma classe pública;
* Os diferentes grupos de conteúdo acima listados devem ser separados por apenas uma linha em branco.
  1. **Pacotes (package statement)**
* O limite de caracteres por linha não se aplica à declaração de pacotes.
* Os pacotes são declarados após qualquer comentário inicial em um arquivo fonte.
* Pacotes do tipo “java.<nome do pacote>” são listados primeiro, seguidos das classes das extensões de Java (pacotes do tipo “javax.<nome do pacote>”) e, por último, os pacotes das classes específicas do sistema e outras API’s utilizadadas.
  1. **Importações (import statement)**
* O limite de caracteres por linha não se aplica à declaração de importações.
* São declaradas após os pacotes.
  1. **Declaração de Classes**
* Os nomes das classes devem começar com letra maiúscula, devem estar no singular e não podem ser abreviados.
  + Para nomes com mais de uma palavra, cada palavra adicional também deve começar com letra maiúscula;
  + Não deve se usar preposições, artigos e qualquer tipo de caractere específico de uma língua (como o ‘ç’, por exemplo).
    1. **Ordem dos membros da classe**
* Após a declaração do nome da classe e de seus supertipos, segue-se a ordem a seguir de declarações:
  1. Comentários de implementação, se houver;
  2. Constantes;
  3. Variáveis de classe (modificador *static*);
  4. Variáveis de instância;
  5. Construtores;
  6. Métodos;
* Quanto aos modificadores de acesso, segue-se a seguinte ordem:
  1. Públicas (*public*);
  2. Protegidas (*protected*);
  3. Sem modificadores;
  4. Privadas (*private*);

**Exemplo 1:**

**//** Nome do arquivo = FirstClass.java

package mvc.model;

import java.util.Scanner;

public class FirstClass

{

private String nomeCliente;

public void cadastrarCliente ()

{

// Corpo do método

}

}

1. **VÁRIAVEIS**

* Deve ser feita apenas uma declaração por linha.

**Para variáveis de instância e de classe:**

* Todas as variáveis devem ter nomes significativos, sem abreviações (exceto em casos de componentes visuais).
* Todas as variáveis formadas por uma única palavra devem ser escritas somente com letras minúsculas. As que são formadas por mais de uma palavra são escritas com a primeira letra em maiúscula de cada palavra adicional.
  + Não se usa caracteres específicos de uma única língua.
  + Os nomes devem seguir o padrão *lowerCamelCase*.

**Para constantes:**

* Os nomes de constantes não devem ser abreviados, possibilitando o maior entendimento possível.
* As palavras que compõem o nome da constante devem possuir todas as letras maiúsculas. No caso de nomes compostos é utilizado o *underline* para separar as palavras.

**Exemplo 2:**

private int diaDoMes;

private int mesDoAno;

1. **MÉTODOS**

* Os nomes dos métodos devem ser um verbo ou uma frase verbal que indique a utilidade do método.
* A escrita dos nomes deve ser em letra minúscula, e cada palavra adicional deve começar com letra maiúscula.
  + Não se deve abreviar os nomes de métodos.
  + Não se usa caracteres específicos de uma única língua.
  + Os nomes devem seguir o padrão *lowerCamelCase*.
  1. **Variáveis Locais**
* Os nomes de variáveis locais segue o padrão da nomenclatura das variáveis de instância e de classe, porém são mais flexíveis a abreviações.
* Nomes compostos por apenas um caractere devem ser evitados, exceto para variáveis temporárias e de *looping*.
* Também só é permitida uma declaração por linha, de preferência mais perto possível do uso da variável, de modo a reduzir o escopo.

**Exemplo 3:**

public class FirstClass

{

private String nomeCliente;

private int idade;

public FirstClass ()

{

// Construtor

}

public void cadastrarCliente ()

{

//Corpo do método

}

public void calcularIdade ()

{

// Corpo do método

}

}

1. **COMANDOS**
   1. **Estruturas Condicionais**

* Estruturas de *if/else* devem possuir chaves para indicar seus blocos independente da quantidade de linhas.
  + Sempre que utilizar do recurso *if*, necessariamente, utilize um *else* e escreva um comentário, mesmo que este pareça ser redundante.
* Na estrutura *switch*, o comando *break* deve estar na última linha do campo *case*.
  + Todos os campos *case* devem conter chaves delimitando seus respectivos blocos.
  + Toda estrutura *switch* deve conter uma cláusula *default* que, se vazia, deve haver um comentário.
  1. **Estruturas de repetição**
* No laço de repetição *for*, a atribuição e comparação não necessitam de espaço entre os termos.
* No laço de repetição *while* e *do-while*, a comparação não necessita de espaço entre os termos.

**Exemplo 4:**

for (i=0; i > tamanhoTurma; i++)

{

// Comando do for

}

while (i < TamanhoTurma)

{

// Comando do while

}

do

{

// Comando do do while

}while(i < tamanhoTurma)

* 1. **Tratamento de exceções**

**Exemplo 5:**

try

{

// Comando do try

}

catch (Exception e)

{

//Comando do catch

}

finally

{

// Comando do finally

}

1. **ESPAÇAMENTOS E IDENTAÇÃO**
   1. **Regras Gerais**

* O código deve conter todas as chaves começando abaixo das suas linhas de comando.
* Serão utilizados como unidade de identação quatro (4) espaços em branco.
* Em métodos que recebem outros métodos como parâmetros, os métodos passados devem ser aplicados em uma variável antes de ser utilizados como parâmetros.
  1. **Linhas em branco**
* Nos blocos que possuem chaves, deve haver uma única linha em branco entre as chaves e seu respectivo conteúdo.
* A declaração de variáveis deve ser separada de qualquer outro código por uma linha em branco.
* Antes de um comentário deve se inserir uma linha em branco.
* Entre as sentenças *package* e *import* deve se inserir uma linha em branco.
* Entre blocos lógicos dentro de um método deve se inserir uma linha em branco.
* Entre declarações de métodos deve se inserir uma linha em branco.
* Entre a declaração da classe e o bloco das variáveis de instância deve se inserir uma linha em branco.
  1. **Espaços em branco**
* Para atribuições é necessário um espaço entre a variável e o conteúdo.
* Sempre que houver uso de vírgula (como em parâmetros de métodos), deve haver um espaço entre a vírgula e o termo subsequente.
* Deve existir espaço entre o nome do método e os parênteses que contém os seus parâmetros na declaração do escopo do método.
* Para comparações é necessário um espaço entre as variáveis ou valores em questão e o operador.
* Se houver mais de um parêntese em uma sentença, é necessário um espaço entre o começo e o fim do conteúdo dos parênteses mais externos.

5.4 **Quebras de linha**

• A linha é quebrada depois de uma vírgula.

• A linha é quebrada antes de um operador.

• A nova linha é alinhada com o começo da expressão do mesmo nível da linha anterior.

**Exemplo 6:**

public class FirstClass

{

private String nomeCliente;

private int opcao;

if( nomeCliente.isEmpty() )

{

// Bloco do if

}

else

{

// Bloco do else

}

switch(opcao)

{

case 1:

{

// Conteúdo case 1

break;

}

default:

{

// Conteúdo default break;

}

}

public void calcular(int numeroUm, int numeroDois,

int numeroTres)

{

// Conteúdo método;

}

}

1. **COMENTÁRIOS**

* O conteúdo dos comentários em uma única linha deve ser separado por um único espaço das barras duplas (‘//’), não sendo necessário o uso de ponto final para indicar o fim do comentário.
* Já os comentários que precisem de mais de uma linha, os símbolos ‘/\*’ e ‘\*/’ devem estar cada um em uma linha diferente e o conteúdo do comentário em linhas diferentes das que se encontram os símbolos. No caso em que o contexto permitir, é necessário o uso de ponto final.
  + Ao gerar comentários em blocos, que precisam mais de uma linha, cada linha de comentário deve começar com "\* " antes de seu conteúdo.
* Todos os comentários devem começar com letra maiúscula.
* Os comentários de documentação seguirão o padrão Java, utilizando os símbolos ‘/\*\*’ e ‘\*/’ como delimitações.

**Exemplo 7:**

// Comentários de uma linha para o programa

/\*

\* Comentários com mais de uma linha para o programa.

\* Sempre com 1 espaço e letra maiúscula.

\*/