Coleção Livro de Bolso

Programação básica Java para Web (JSP)

Rogério Laureano Martins

Linguagem Java 7
História do Java 8
Features Java 9

Preparação para programar em Java 12
1) Como definir o caminho temporário do JDK no Windows 13
2) Como definir o caminho permanente do JDK no Windows 14
Configurando o caminho Java no Linux OS 19

JSP programação Web 20

Exibição de mensagem e textos 21
Tipos de tags 21
Tag de scriptlet 22
Tag expression 24
Tag declaration 29

Variáveis 30
Operações Numéricas 41
Operações com texto 87
Operações com data 112
Transformações de número para caracter .. 152
Transformações de caracter para número .. 164
Transformações de data para número 171
Transformações de número para data 175
Transformação de String para data 177

Transformação de Data para String *180*
Transformações de número para número *181*
Transformação de String para boolean *193*
Transformação de boolean para String *194*
Transformação de String para char *195*
Transformação de char para String *196*
Transformação de Date para Timestamp *197*
Transformação de Timestamp para Date *198*
Transformações de String para Objeto *199*
Transformações de Objeto para String *200*
Teste de variáveis e tomadas de decisões *201*
Ciclos *221*
Array de variáveis *227*
Enumeration 230
Vector 232
ArrayList 235
LinkedList 237
LinkedHashSet 240
TreeMap 241
TreeSet 243
BitSet 245
Stack 247
HashMap 250
HashSet 252
Hashtable 254

Properties 256
Queue 258

Gravação e Leitura de arquivos 262
Gravação e leitura de arquivos texto 262

Banco de dados 270
Gravação de registros de dados 272
Conexão com outros bancos de dados 273
Métodos da interface Connection 277
Métodos da classe Statement 278
Métodos da classe ResultSet 278
RowSet 280
Métodos da classe PreparedStatement 281
ResultSetMetaData 284
DataBaseMetaData 285
CallableStatement 287
Controle de transações no banco 290
Processamento batch 291

Objetos implícitos do JSP 292
Objeto out 293
Objeto request 293
Objeto response 296
Objeto config 307
Objeto application 309
Objeto session 311

Objeto pageContext *314*
Objeto page *317*
Objeto exception *317*

Diretivas do JSP **318**
Diretiva page import *318*
Diretiva page contentType *319*
Diretiva page info *319*
Diretiva page buffer *320*
Diretiva page errorPage *320*
Diretiva page isErrorPage *321*
Diretiva include *321*

Elementos de ação **322**
jsp:forward *322*
jsp:include *323*
jsp:param *324*
jsp:useBean *326*
jsp:plugin *327*

Conclusão **328**

Minha gratidão a Deus por me permitir compartilhar o que tenho aprendido. Minha gratidão à minha esposa Beatriz e ao meu filho João por terem paciência. Minha gratidão aos professores e youtubers que de alguma forma procura passar o conhecimento pessoal às pessoas.

Rogério Laureano Martins

Linguagem Java

A linguagem Java é uma poderosa ferramenta de programação. Neste livro temos temos diversos conceitos desta linguagem.

Com o Java, o desenvolvimento de 4 tipos de programas são bastante populares.

Standalone → Programas populares para rodar em desktop. Semelhantes a Media Player, antivírus etc.

Web → Páginas HTML dinâmicas

Aplicativos corporativos → Aplicações para empresas como bancos, varejo, indústria etc.

Mobile → Aplicativos para equipamentos mobile como telefone celular, tablets etc.

Para tanto, forma criadas 4 plataformas de desenvolvimento para o java:

Java SE → Plataforma de desenvolvimento java que inclui todos os recursos de IO, Expressões regulares etc.

Java EE → Plataforma para desenvolvimentos corporativos e Web. É um pedaço importante do Java SE.

Java ME → Plataforma "micro" do Java. Utilizada principalmente para Mobile.

Java FX → Plataforma especial utilizada principalmente para aplicativos Web mais ricos.

História do Java

Em 1991 o projeto Java se iniciava com uma equipe formada por James Gosling, Mike Sheridan e Patrick Naughton. O time de desenvolvimento se denominou Green Team.

Por isso o primeiro nome foi Greentalk. Posteriormente a linguagem mudou de nome para Oak, que era uma árvore típica dos Estados Unidos, França, Romênia etc. Como já existia uma empresa chamada Oak, mudou-se o nome para Java. Este nome foi dado em homenagem a ilha de Java, que tem uma história com a produção de café. Por isso também que o símbolo do java é uma chícara de café.

Features Java

O projeto Java, foi feito com o intuito de criar uma linguagem portável, segura e simples. Abaixo a lista dos principais features:

- Programa Simples → Os desenvolvedores pretenderam tirar a dificuldade da linguagem. É muito semelhante a linguagem C. Porém, sem as complicações de ponteiros. Este, por exemplo, podem ser usados. Porém, são internos no Java.

- Orientação por objeto → O Java é uma linguagem orientada por objetos. Possibilita a criação de Objetos, Classes, Heranças, Polimorfismo e todos os conceitos de orientação por objetos.

- Portável → O bytecode gerado é o mesmo em qualquer interpretador Java

- Multiplataforma → Por rodar em uma máquina virtual, "JVM" é portátil e roda em diversos sistemas operacionais.

- Segura → Por estar na JVM, tem mais uma camada de segurança independente do sistema operacional

- Robusta → Ferramente robusta no desenvolvimento

- Archetecture neutral → Por exemplo, os tamanhos dos tipos primitivos são reservados sempre o mesmo tamanho de bits.

- Linguagem interpretada → Ao rodar o aplicativo criado em Java, é necessário que se cria um bytecode. Ou seja, é feito uma "compilação" no Java para criar este bytecode. Após isso o Java interpreta o bytecode e executa o programa.

- Multithreaded → O Java permite programas com diversas threads em paralelo

- Distribuída → Facilmente distribuido

- Dinâmica → Linguagem bastante dinâmica e versátil

Preparação para programar em Java

Para executar qualquer programa java, você precisa

- Instale o JDK se você não o tiver instalado, faça o download do JDK e instale-o.
- Defina o caminho do diretório jdk
- Crie o programa java
- Compile e execute o programa java

É necessário definir o caminho para o uso de ferramentas como javac, java, etc.

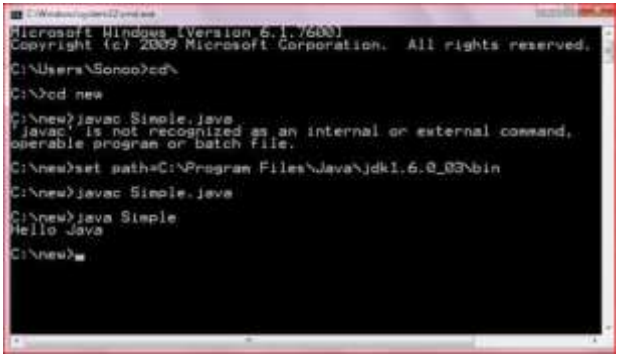
Se você estiver salvando o arquivo de origem Java dentro do diretório JDK / bin, não será necessário definir o caminho, pois todas as ferramentas estarão disponíveis no diretório atual.

No entanto, se você tiver seu arquivo Java fora da pasta JDK / bin, será necessário definir o caminho do JDK.

Existem duas maneiras de definir o caminho em Java:
1 – Temporário
2 – Permanente

1) Como definir o caminho temporário do JDK no Windows

Para definir o caminho temporário do JDK, você precisa seguir as seguintes etapas:
Abra o prompt de comando
Copie o caminho do diretório JDK / bin
Escreva no prompt de comando: defina path = copied_path
Por exemplo:
definir caminho = C: \ Arquivos de Programas \ Java \ jdk1.6.0_23 \ bin
Vamos ver na figura abaixo:



2) Como definir o caminho permanente do JDK no Windows

Para definir o caminho permanente do JDK, siga estas etapas:

Vá para Propriedades do MyComputer -> guia avançado -> variáveis de ambiente -> nova guia da variável do usuário -> caminho de gravação no nome da variável -> caminho de gravação da pasta bin no valor da variável -> ok -> ok -> ok

Por exemplo:

1) Vá para propriedades do MyComputer



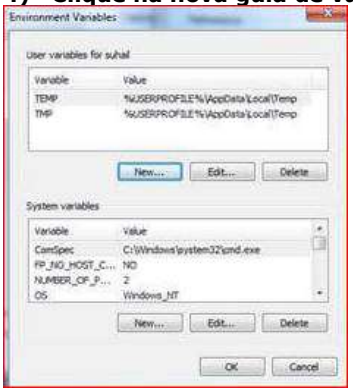
2) Clique na aba avançada



3) Clique nas variáveis de ambiente



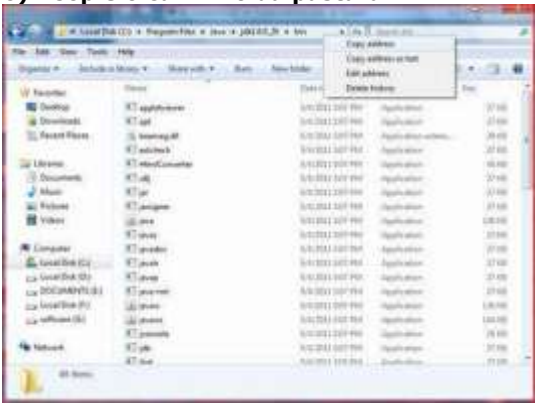
4) Clique na nova guia de variáveis do usuário



5) Escreva o caminho no nome da variável



6) Copie o caminho da pasta bin



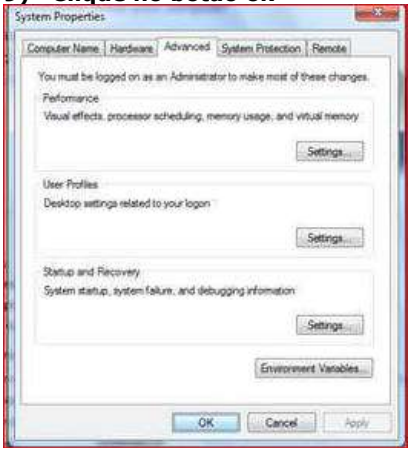
7) Cole o caminho da pasta bin no valor da variável



8) Clique no botão ok



9) Clique no botão ok



Agora seu caminho permanente está definido. Agora você pode executar qualquer programa de java a partir de qualquer unidade.

Configurando o caminho Java no Linux OS

Definir caminho no sistema operacional Linux é o mesmo que definir o caminho no sistema

operacional Windows. Mas, aqui usamos a ferramenta de exportação em vez de definir. Vamos ver como definir o caminho no sistema operacional Linux:

```
  exportar PATH = $ PATH:
/home/jdk1.6.01/bin/
```

Aqui, instalamos o JDK no diretório inicial em Raiz (/ home).

JSP programação Web

Somado com o TomCat, o Java permite a criação de programas Web (intranet ou internet) através do JSP.

O pré-requisito, é que tenha o TomCat instalado.
Em meu exemplo, foi instalado em C:\apache-tomcat-9.0.30

Assim, todas os arquivos com extensão .jar, ficarão no diretório:
C:\apache-tomcat-9.0.30\lib

E todos os arquivos com extensão .jsp ficarão no diretório:
C:\apache-tomcat-9.0.30\webapps\ROOT

Exibição de mensagem e textos

O JSP é organizado por tags, diretivas e elementos de ação. Também possui alguns objetos implícitos. Segue abaixo a explicação de cada um destes elementos.

Tipos de tags

Existem diversos tipos de tags no JSP. As tags de elementos de script são divididas três subdivisões: tag scriptlet, tag expression e tag declaration. Além destas, também temos as tags de diretivas e as tags de elementos de ação

Tag de scriptlet

A tag scriptlet, premite acoplar os comandos em Java dentro dela. A tag é "<% [comandos] %>". A interação com o HTML é feita através de objetos implícitos do JSP. No exemplo abaixo o objeto implício é o "out":

```
<html>
<body>
<%
  out.print("Ola mundo");
%>
</body>
</html>
```



Exemplo 2 com interação do objeto request:

Arquivo (testeScriptlet.html):

```
<html>
<body>
<form action="testeScriptlet.jsp">
Informe seu nome <input type="text"
name="nome"><br/>
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

Arquivo (testeScriptlet.jsp)

```
<html>
<body>
<%
String
nome=request.getParameter("nome");
out.print("Ola " + nome);
%>
</form>
</body>
</html>
```



Ao clicar em "Enviar" o sistema exibe o seguinte:



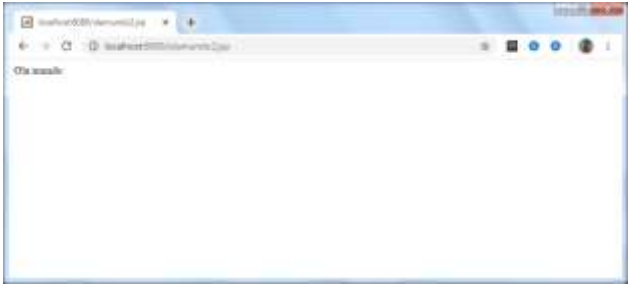
Tag expression

A tag de expressão, permite interagir diretamenente com o HTML com comando em

Java, não utilizando os objetos implícitos do JSP que são de saída para o HTML. A tag é a seguinte
"<%= [comandos] %>".

Exemplo de utilização:

```
<html>
<body>
<%= "Ola mundo" %>
</body>
</html>
```



Note que não foi utilizado o objeto "out" ou o objeto "response". Apenas foi colocada a string de saída.

Outro exemplo de tag de expressão utilizando um comando do Java:

```
<html>
<body>
Hoje <%=
java.util.Calendar.getInstance().getTime()
%>
</body>
</html>
```

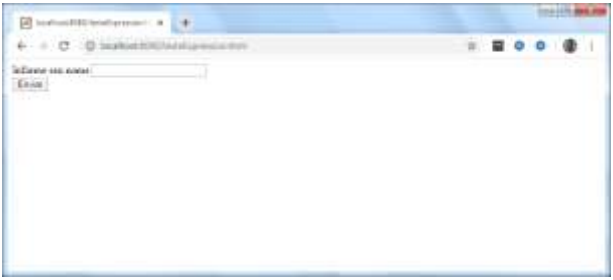


Exemplo de tag expression com interação de objeto implícito do JSP:
Arquivo (testeExpression.html):

```
<html>
<body>
<form action="testeExpression.jsp">
Informe seu nome <input type="text"
name="nome"><br/>
<input type="submit" value="Enviar">
</form>
</body>
</html>
```

Arquivo (testeExpression.jsp)

```
<html>
<body>
<%=
"Ola " + request.getParameter("nome")
%>
</form>
</body>
</html>
```



Ao clicar no botão "Enviar" o sistema exibe a seguinte página:



Tag declaration

A tag de declaração, permite declarar variáveis ou funções que retornam valores dentro de um arquivo jsp. A tag é a seguinte:
<%! [declaração] %>

Exemplo:

```
<html>
<body>
<%! int valor=1; %>
<%= "Valor da variavel: " + valor %>
</body>
</html>
```



Exemplo 2:

```
<html>
<body>
<%!
int dobro (int n){
  return n * 2;
}
%>
<%= "Dobro de 3: " + dobro(3) %>
</body>
</html>
```



Variáveis

Quanto ao tipo de informação primitivos que as variáveis suportam, destaco:

byte → Número inteiro de -128 à 127
short → Número inteiro de -32.768 à 32.767
int → Número inteiro de -2.147.483.648 até 2.147.483.647
long → Número inteiro de -9.223.372.036.854.770.000 à -9.223.372.036.854.770.000
float → Número real (com valores fracionados) de $3,4^{-38}$ à $3,4^{38}$
double → Número real de $1,7^{-308}$ à $1,7^{308}$
char → Caracter que vai de 0 à 65535
boolean → Valor verdadeiro e falso.

O java é case sensitive. Ou seja, faz diferença entre letras maiúsculas e minúsculas no momento da codificação. Abaixo temos a lista de classes básicas do java e a referência com as variáveis primitivas:

Byte → Classe que contem regras, transformações dos valores primitivos do tipo byte.
Short → Classe que contem regras, transformações dos valores primitivos do tipo short.
Integer → Classe que contem regras, transformações dos valores primitivos do tipo int.
Long → Classe que contem regras, transformações dos valores primitivos do tipo long.
Float → Classe que contem regras, transformações dos valores primitivos do tipo float.
Double → Classe que contem regras, transformações dos valores primitivos do tipo double.
String → Classe que contem regras, transformações dos valores primitivos do tipo char.