Programação Estruturada

Aula 01 - Introdução à linguagem Java e à Programação Estruturada

Apresentação

Caro aluno, neste momento, você inicia seus estudos da disciplina Programação Estruturada. Essa disciplina pode ser vista como a continuação da disciplina de Lógica de Programação, na qual você estudou conceitos básicos de algoritmo. Através dela, você aprenderá novos conceitos e uma nova linguagem de programação que lhe permitirá resolver problemas mais interessantes, como desenvolver jogos e outros tipos de sistemas comerciais.

O termo programação estruturada é usualmente associado ao paradigma (estilo) de programação imperativo já trabalhado na disciplina de Lógica de Programação. Relembrando, este paradigma é baseado na entrada, armazenamento, processamento e saída de dados. Por ter uma estrutura similar à arquitetura de Von Neumann, o paradigma imperativo é bastante eficiente em termos de tempo computacional para resolução de problemas. Existem diversas linguagens que seguem o paradigma imperativo.

Nessa disciplina, você aprenderá a utilizar a linguagem Java, uma linguagem de programação bastante conhecida e utilizada tanto no mercado como na academia (universidade). A linguagem Java, porém, segue o paradigma orientado a objetos, assunto este ainda a ser visto na disciplina Programação Orientada a Objetos (POO). Entretanto, o paradigma orientado a objetos pode ser visto como uma extensão do paradigma imperativo. Dessa forma, a linguagem Java também pode ser utilizada para ensinar programação estruturada.

Em resumo, você aprenderá a sintaxe (vocabulário da linguagem) Java, bem como comandos e funções básicas disponibilizadas pela linguagem. Recursos da linguagem relacionados ao paradigma orientado a objetos, porém, só serão vistos na disciplina de POO.

Por estarem relacionadas a conteúdos que requerem muita prática de sua parte, nossas aulas possuem diversos exercícios, incluindo o desenvolvimento de um jogo. Não deixe de praticar, isso será essencial para um bom aprendizado e fixação do conhecimento de programação.

Nesta nossa primeira aula, você irá aprender sobre os conceitos básicos da programação estruturada e da linguagem de programação Java. Em primeiro lugar, você irá entender a origem e a importância do estudo dessa forma de programação modular e da linguagem de programação Java, que é caracterizada como uma linguagem portável (detalharemos isso mais adiante).

Nas aulas posteriores, você irá aprender conceitos, comandos e técnicas de programação em Java. O conhecimento adquirido nestas aulas será utilizado para o desenvolvimento de vários pequenos programas e, como já falamos, também de um jogo.

Temos certeza que este curso será bastante proveitoso para você. Bons estudos!

Vídeo 01 - Apresentação

Objetivos

Após a leitura desta aula, você será capaz de:

- Identificar as vantagens de se utilizar a linguagem de programação Java.
- Identificar as características da linguagem Java.
- Identificar as características de ambientes de programação em Java; possibilitando seu uso no desenvolvimento de programas Java.

1. Programação estruturada em Java

Bem, nesta primeira aula, será feita uma apresentação do que é a programação e quais as principais linguagens de programação do mercado, dando ênfase à linguagem Java. Esta aula é importante porque conheceremos um pouco da história da linguagem de programação que iremos aprender nesta disciplina, similar a uma apresentação pessoal, com informações básicas para que você possa conhecer o máximo dessa linguagem tão poderosa. Também serão introduzidos conceitos utilizados na área de programação, visando a nossa preparação para nos lançarmos nesse mundo intrigante da criação de programas computacionais. Então, a partir de agora, nos lançaremos em um maravilhoso e curioso mundo de desafios e descobertas sobre a programação.

2. Introdução à linguagem Java

Você já conhece a linguagem Java? Já pesquisou sobre isso? Bem, a linguagem Java é atualmente uma das mais conhecidas no mercado e na academia, sendo utilizada por boa parte dos programadores do mundo inteiro. Sua popularidade se deve principalmente às seguintes características:

1. Portabilidade – capacidade de escrever código capaz de ser executado em várias plataformas e sistemas operacionais;
2. suporte a várias tecnologias (redes, acesso a banco de dados etc.);
3. disponibilidade de ambientes de execução em vários dispositivos móveis, como celular, palmtop e PDA's.

Outra característica fundamental é a de ser uma linguagem relativamente aberta e que evolui de acordo com os desejos da sua comunidade de software (principais empresas internacionais do mercado), além de possuir várias ferramentas gratuitas disponíveis para os desenvolvedores.

Veja agora um pouco sobre o histórico dessa linguagem.

3. Histórico de Java

A linguagem Java começou com o nome Oak, desenvolvida na década de 1990 por uma equipe de programadores da empresa Sun Microsystem, que foi comprada pela empresa Oracle em janeiro de 2010. A ideia dessa nova linguagem era se antecipar com uma tendência prevista de integração entre os computadores e os eletrodomésticos, como a integração de computadores e aparelhos de TV, ou de uma geladeira que avisasse que alimentos estariam para estragar-se.

Apesar de essa previsão estar correta, já que essa integração já é uma realidade em vários equipamentos, como no caso das TVs, ainda estava muito cedo para se desenvolver essa linguagem. Porém, nessa época, estava surgindo a Internet. Então, em 1995, uma nova versão da antiga linguagem Oak foi lançada com o nome Java, agora focada no desenvolvimento Web. Ela introduziu na época os Applets, programas que rodavam na máquina do cliente através dos navegadores Web. Com a "explosão" da nova sensação chamada internet, a linguagem Java pegou carona e conseguiu ser conhecida e se estabelecer no mercado.

Algumas outras estratégias foram utilizadas, como a semelhança de sua sintaxe com a sintaxe da linguagem C, a qual era fortemente utilizada na época em questão (e que ainda é bastante utilizada até hoje no mercado). Isso facilitou a migração de programadores C para Java.

Além disso, existem alguns outros pontos positivos que ajudaram a consolidar e popularizar a linguagem de programação Java, tornando-a de grande interesse para os mais diversos segmentos da computação:

1. linguagem orientada a objetos (recursos que facilitam a programação e que serão vistos na próxima disciplina de programação);
2. linguagem possuidora de diversos recursos para a programação de sistemas distribuídos, concorrentes (execução "paralela"), robustas a erros, seguras, entre outras coisas;
3. portabilidade, pois uma vez escrito um código em Java, a ideia é que ele possa ser executado em qualquer ambiente que possua uma máquina de execução Java, podendo ser um computador, uma geladeira, uma televisão, etc. Essa característica estava clara no slogan utilizado pela Sun na época: "escreva uma vez, execute em qualquer lugar" (em inglês, "write once, run anywhere").

Atividade 01

1. Qual o nome da linguagem que originou Java? Qual o seu intuito original?
2. A sintaxe da linguagem Java foi desenvolvida baseada em alguma outra linguagem de mercado? Qual? E por qual motivo?
3. Cite características da linguagem de programação Java.

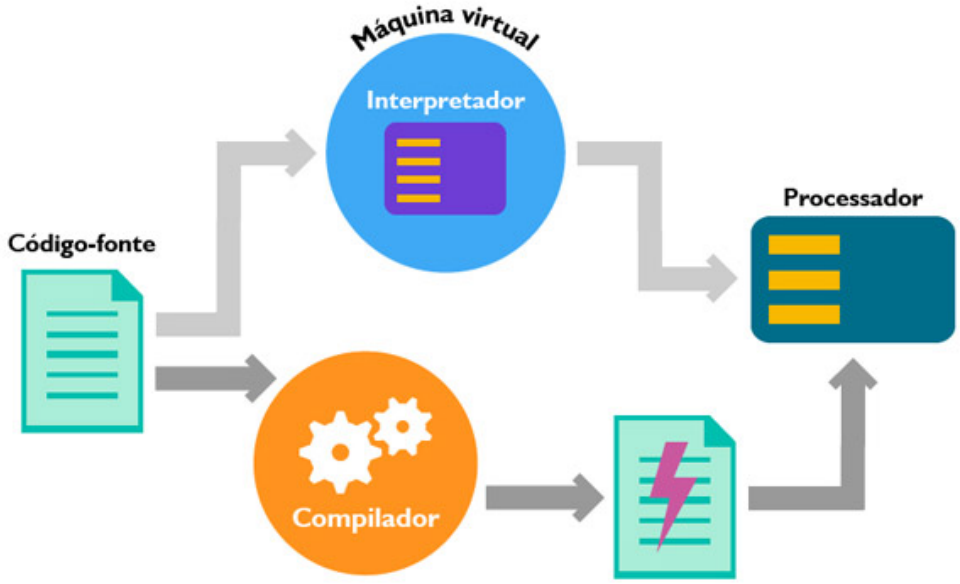
4. Processos de compilação ou de interpretação

A característica de portabilidade de Java é realmente marcante, por isso vamos explorá-la mais um pouco. Um programa é portável se ele pode ser executado em diferentes plataformas de software e hardware. Exemplos de plataformas de software são os ambientes Windows e Linux. Você já percebeu que, ao fazer o

download de alguns programas, você é perguntado se sua máquina é Windows, Linux, etc.? E se sua máquina é um computador de 32 ou de 64 bits (plataforma de hardware)?

Vamos entender isso melhor. Ao escrever um programa, dizemos que o programador escreveu o **código-fonte** do programa. Para rodar esse programa, precisamos compilá-lo ou interpretá-lo. Interpretar é entrar com o código-fonte em um programa chamado de interpretador. Isso foi o que você fez na disciplina de Lógica de Programação e Algoritmos. Vocês usaram o VisualG ou uma ferramenta equivalente, não foi? Relembrando, o VisualG é uma ferramenta que lê código escrito em português e executa-o, comando por comando. Veja isso na **Figura 1**.

Figura 01 - Processo de compilação e de interpretação



Fonte: Adaptado de <http://www.pasteur.fr/formation/infobio/python/ch05s02.html>. Acesso em: 7 dez. 2011.

Note o uso da terminologia máquina virtual (*virtual machine*) no caso do interpretador. Essa máquina virtual simula uma máquina física que entende a linguagem escrita pelo ser humano. Lembre-se de que as máquinas reais, os computadores, só entendem sinais elétricos, representados pelos bits 0 e 1 (linguagem de máquina). Por isso, a máquina virtual tem a função de traduzir cada comando, um por vez, em comandos entendidos pela máquina, ou seja, ela traduz os comandos para a linguagem de máquina.

Já no caso do compilador, essa ferramenta lê o código-fonte por completo, realiza várias verificações e, por fim, gera um código-fonte de saída equivalente ao programa de entrada, porém escrito na linguagem que a máquina compreende.

Qual o melhor processo a ser utilizado? Compilação ou interpretação? Cada um deles tem suas vantagens e desvantagens. Na compilação, temos a vantagem de realizar uma enorme quantidade de verificações de erro antes de gerarmos o código que será executável na máquina. Em caso de existir algum erro no programa, o código executável não será gerado! Entretanto, ao gerarmos o programa executável, ele acaba sendo específico para uma determinada plataforma de software e hardware.

Por esse motivo é que muitas vezes são geradas (compiladas) várias versões de um mesmo programa fonte, uma para cada possível ambiente de execução (Linux ou Windows, 32 ou 64 bits, etc.). É muito chato tanto para o desenvolvedor do software como para o usuário, que tem que ficar escolhendo, entre várias versões, qual a correta para ele baixar, de acordo com a máquina que ele vai utilizar o software.

Na interpretação, não temos as diversas verificações que são feitas pelo compilador. Isso quer dizer que podemos começar a executar um programa e ele poderá parar pela metade porque o programador esqueceu de colocar o símbolo "+" em uma operação de soma, por exemplo.

Entretanto, a interpretação nos traz a possibilidade de se obter portabilidade. Isso porque, para rodar um mesmo programa em ambientes diferentes, é só fazer uso de máquinas virtuais específicas para cada ambiente. Isso quer dizer que, ao instalarmos uma máquina virtual (interpretador da linguagem) na nossa máquina, precisaremos fazer o download da versão correta, de acordo com nossa plataforma (computador/sistema operacional).

Entretanto, qualquer programa escrito na linguagem pode, em teoria, ser executado em qualquer máquina virtual, esteja ela instalada em um computador, com processador de 32 ou de 64 bits, com Linux ou Windows, ou em aparelho de TV ou geladeira, etc.

Um programa é portável se ele pode ser executado em diferentes plataformas de software e hardware.

Atividade 02

1. Como funciona o processo de compilação? Quais suas vantagens e desvantagens?
2. Como funciona o processo de interpretação? Quais suas vantagens e desvantagens?

5. Portabilidade de Java

Muito bem, vejamos agora como um código-fonte Java pode ser trabalhado para poder ser executado em uma determinada máquina. Em Java, o código-fonte dos programas é escrito em arquivos com extensão ".java", como mostra a **Figura 2**.

Em uma primeira etapa, os arquivos Java são passados para um compilador Java, o qual é responsável por analisar o código-fonte em busca de erros. Caso existam, eles serão apresentados para o programador. Caso contrário, um programa equivalente (código-objeto) será gerado, agora em arquivos com mesmo nome, mas com extensão ".class".

Figura 02 - Processo de compilação e interpretação de Java

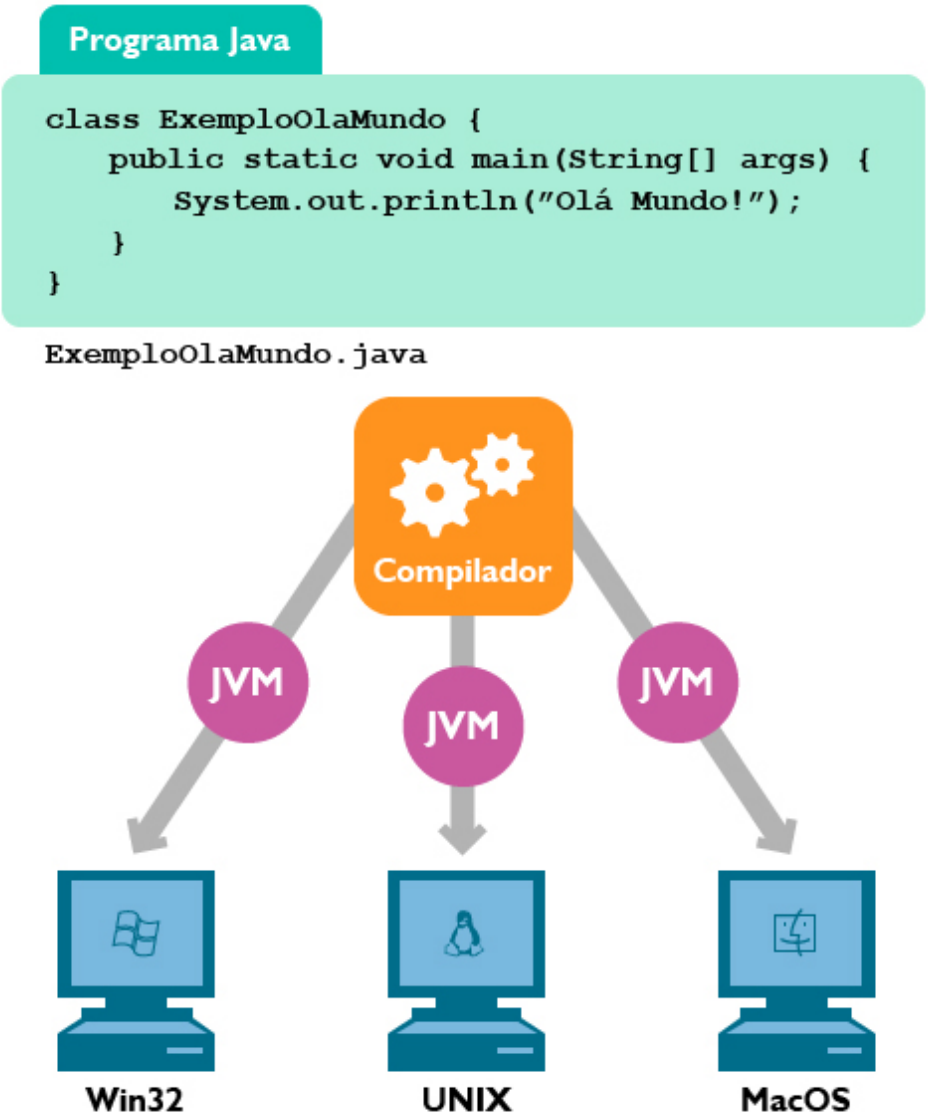
Fonte: Adaptado de <http://download.oracle.com/javase/tutorial/getStarted/intro/definition.html>. Acesso em: 7 dez. 2011.

Os arquivos ".class" são escritos pelo compilador em uma linguagem que não é de máquina, mas que também não é legível pelos seres humanos. Essa linguagem é chamada de bytecode e não é dependente de nenhuma plataforma de hardware ou de software. Sendo assim, consideramos teoricamente portáveis os programas escritos em bytecode.

Em uma segunda etapa, quando queremos efetivamente rodar um programa Java, pegamos o código objeto (arquivos ".class") e passamos para o interpretador Java, mais conhecido como máquina virtual Java (JVM – Java Virtual Machine). Como já estudamos, a JVM irá ler cada comando do programa escrito em bytecode e o executará, traduzindo o mesmo para comandos entendíveis pela máquina, de acordo com a plataforma utilizada (hardware e software).

Dessa forma, podemos dizer que Java é tanto compilado (etapa 1) como interpretado (etapa 2). A portabilidade de Java se dá então como o apresentado pela **Figura 3**. Temos o programa Java (código-fonte), um único compilador e várias JVMs, uma para cada plataforma de execução diferente.

Figura 03 - Portabilidade de Java



Fonte: Adaptado de <http://download.oracle.com/javase/tutorial/getStarted/intro/definition.html>. Acesso em: 7 dez. 2011.

Atividade 03

1. Java é uma linguagem compilada ou interpretada? Explique.

6. Estrutura de um programa Java

Na seção anterior, foi discutido o processo Java necessário para sairmos do código fonte para sua execução. No decorrer dessa seção, será mostrada a estrutura básica de um programa em Java. A partir de agora, para que o assunto seja bem absorvido, assim como ocorreu nas aulas do módulo de Lógica de Programação, serão sugeridas atividades práticas durante o desenvolvimento da aula e ao final dela, tudo isso para que você possa absorver melhor o assunto.

Mas, vamos em frente! Para entender a estrutura de programas Java, precisamos ter em mente que um programa pode ser constituído por um ou mais arquivos com extensão ".java". O nome desse arquivo será referenciado dentro do código-fonte. Por exemplo, o arquivo "ExemploOlaMundo.java" requer que exista o seguinte código Java dentro dele:

```
1 public class ExemploOlaMundo{
2
3 }
```

As palavras **public class** são obrigatórias e serão explicadas na disciplina de POO. O que importa no momento é você saber que o programa começa com **public class**, depois vem o nome do programa (que é igual ao do arquivo), e em seguida tem a abertura e fechamento de chaves, as quais delimitam o local onde podemos escrever nosso código.

Apesar de compilar, esse programa não faz nada e não pode ser executado. Para poder ser executado, criamos o que chamamos de método **main**. O termo método é utilizado em linguagens orientadas a objetos. Como vamos estar trabalhando com Java, mas no paradigma imperativo, vamos chamar métodos de procedimentos ou de funções, ok? Na disciplina de Programação Orientada a Objetos, vocês serão introduzidos à terminologia da orientação a objetos e conhecerão termos como "classes", "métodos" e "atributos". Veja como fica agora o código com o procedimento main incluído nele:

```
1 public class ExemploOlaMundo{
2    public static void main(String[] args){
3
4    }
5
6 }
```

No momento, considere que todo esse comando **public static void main(String[] args)** é utilizado para delimitar, junto com a abertura e fechamento de chaves, o início e fim da área que podemos usar para escrever o início do nosso programa. O programa **ExemploOlaMundo** agora pode ser compilado e executado, porém o procedimento **main** não faz nada, pois está vazio. A seguinte versão faz uso do comando **System.out.println** para imprimir a mensagem Olá Mundo!, delimitada entre aspas duplas:

```
1 public class ExemploOlaMundo{
2    public static void main(String[] args){
3       System.out.println("Olá Mundo!");
4    }
5
6 }
```

Além disso, como no desenvolvimento de sistemas em Java é comum termos vários arquivos, podemos organizá-los em pacotes. Os pacotes reúnem arquivos com objetivos relacionados e refletem a estrutura de diretórios onde os arquivos se encontram. Por exemplo, o pacote "exemplo" requer a existência de uma pasta "exemplo" no computador. Podemos colocar o programa mostrado

```
1 package exemplo;
2
3 public class ExemploOlaMundo{
4    public static void main(String[] args){
5       System.out.println("Olá Mundo!");
6    }
7
8 }
```

A palavra chave package, quando utilizada, deve ser o primeiro comando do arquivo, indicando o nome do pacote, o qual está obrigatoriamente relacionado ao nome da pasta que contém o arquivo Java. Por fim, você pode colocar diversos

comentários no código, tendo os mesmos que estarem entre /* e */, ou depois de //, sendo que nesta última opção, o comentário tem que estar em uma única linha. Veja agora como fica o código com comentários:

```
1  package exemplo;
2
3  /*
4     Este é um comentário que envolve
5     mais de uma linha.
6  */
7
8  public class ExemploOlaMundo{
9     public static void main(String[] args){
10       System.out.println("Olá Mundo!");
11    }
12
13 }
```

Vídeo 02 - Instalação JDK.

Vídeo 03 - Instalação Netbeans.

Vídeo 04 - Exemplo Netbeans.

Atividade 04

1. Qual a estrutura básica de programas Java?

2. Execute o roteiro encontrado em <http://netbeans.org/kb/docs/java/quickstart_pt_BR.html>, de forma a rodar o programa Java apresentado. Para usar o roteiro, você precisará das ferramentas Java e NetBeans instaladas em seu computador. Siga os guias de instalação encontrados na seção de referência, se necessário.

Conclusão

Bem, chegamos ao fim de nossa primeira aula. A partir desta aula, esperamos que você tenha absorvido o conhecimento necessário para iniciar o desenvolvimento de programas básicos em Java. É importante destacar que o processo de aprendizado também necessita que você mostre curiosidade para buscar complementar os seus estudos através da leitura de outros materiais, sejam eles livros ou artigos na Internet. Sugerimos que, sempre que possível, você procure códigos disponíveis na internet, analise-os e execute-os em seu ambiente de programação.

Abraços e até a próxima aula, quando entraremos em mais detalhes sobre a linguagem de programação Java!

7. Leitura Complementar

O ambiente NetBeans de desenvolvimento de programas em Java possui diversos materiais disponíveis para leitura (sistema de ajuda, tutoriais, etc.), tanto na instalação local, como na Internet.

Procure e leia a documentação relativa a essa ferramenta na Internet (site: http://netbeans.org/), que é a que você irá utilizar durante seus estudos nessa disciplina.

8. Resumo

Nesta aula, você estudou uma definição básica sobre a linguagem Java, suas principais características, a estrutura de um programa básico em Java, citando ambientes de programação amplamente utilizados, tanto no meio acadêmico, quanto no meio organizacional. Você viu o conceito de compilação e interpretação, assim como a aplicação dos mesmos na linguagem Java. Além disso, aprendeu a estrutura básica de programas em Java.

9. Autoavaliação

1. Descreva o funcionamento geral das linguagens compiladas, interpretadas e do que acontece no caso da linguagem Java.
2. Descreva a estrutura básica de programas em Java.

10. Referências

ASCENCIO, Ana Fernanda Gomes; CAMPOS, Edilene Aparecida Veneruchi de. **Fundamentos da programação de computadores**: algoritmos, Pascal, C/C++ e Java. São Paulo: Editora Pearson, 2008.

JAVA PLATFORM (JDK). **Guia de instalação no Windows**. Disponível em: <http://gsd.ime.usp.br/~kon/MAC110/instalaDrJava/windows.html>. Acesso em: 7 dez. 2011.

_____. **e Linux**. Disponível em: <http://gsd.ime.usp.br/~kon/MAC110/instalaDrJava/linux.html>

NETBEANS. **Guia de instalação**. Disponível em: <http://netbeans.org/community/releases/70/install_pt_BR.html>. Acesso em: 7 dez. 2011.

THE JAVA tutorials. Disponível em: <http://download.oracle.com/javase/tutorial/>. Acesso em: 6 dez. 2011.