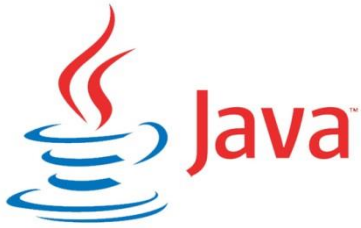
Lógica de Programação

Unidade 4 – Ambiente de desenvolvimento Java



QI ESCOLAS E FACULDADES
Curso Técnico em Informática

SUMÁRIO

A LINGUAGEM JAVA

Java é uma tecnologia. É uma linguagem de programação poderosa e flexível. Utilizamos Java para desenvolver softwares. Pertence à empresa *Oracle* e é gratuita.

- Pode ser utilizada para desenvolver: *websites*, jogos para celular, sistemas comerciais, etc.
- É multiplataforma: um *software* em Java roda em *Windows*, *Linux*, *MacOSX*, celular, etc.
- É orientada a objetos (paradigma de programação mais utilizado no mundo).
- É segura.
- É híbrida: compilada e interpretada.

JVM, JRE, JDK

JVM (*Java Virtual Machine*): Máquina virtual responsável por interpretar e executar o código Java compilado (***bytecode***). Uma JVM é desenvolvida para uma plataforma específica.

JRE (*Java Runtime Environment*): É composto pela JVM e pelas API's Java. É necessária para rodar aplicações Java.

> ***API*** *é o acrônimo de "Application Programming Interface" ou, em português, Interface de Programação de Aplicativos. Esta interface é o conjunto de padrões de programação que permite a construção de aplicativos e a sua utilização de maneira não tão evidente para os usuários. (CIRIACO, 2009)*

JDK (*Java Development Kit*): Conjunto de ferramentas necessárias para realizar o desenvolvimento de aplicações Java. Inclui a JRE e ferramentas como: *javac*: compilador; *jar*: empacotador; *javadoc*: documentação. É necessária para criar aplicações Java.

Bytecode

Uma classe editável em Java possui a extensão **".java"**. Quando esta é compilada, gera um arquivo com a extensão **".class"**. Este arquivo é chamado de ***bytecode***, um código traduzido para a JVM. A JVM interpreta o *bytecode* e o traduz para o código nativo da máquina onde a aplicação está rodando.

PREPARANDO O AMBIENTE DE DESENVOLVIMENTO

Para desenvolver um programa em Java, precisamos inicialmente instalar o kit de desenvolvimento Java, conhecido como **JDK**, e posteriormente a IDE que será utilizada.

> ***IDE*** *significa ambiente integrado de desenvolvimento, um software preparado para oferecer vários recursos que auxiliam no processo de desenvolvimento de um programa.*

Instalando o JDK

O JDK é um *kit* de desenvolvimento Java fornecido livremente pela *Oracle*. Constitui um conjunto de programas que engloba compilador, interpretador e utilitários, fornecendo um pacote de ferramentas básicas para o desenvolvimento de aplicações Java.

1. Para instalar o JDK é preciso acessar o site:
   a. http://www.oracle.com/technetwork/java/javase/downloads/index.html
2. Clicamos em *Java Platform (JDK) 7u7*
3. Aceite o contrato de licença e faça o *download* gratuito do JDK, aproximadamente 70MB.
4. Escolhemos a plataforma conforme nosso computador (Windows x86, Windows x64, Linux, Mac OS X).
5. Execute o instalador.

Instalando a IDE BlueJ

O *BlueJ* é um ambiente Java projetado especificamente para o ensino introdutório. É uma IDE desenvolvida especificamente para os programadores iniciantes, pois facilita a visualização e interação do código auxiliando no entendimento aos conceitos da orientação a objetos.

O *BlueJ* é um ambiente Java projetado especificamente para o ensino introdutório. *BlueJ* foi desenvolvido em uma universidade especificamente para a finalidade de ensinar orientação a objetos com o uso da linguagem de programação JAVA. Sua primeira versão foi lançada em 1999.

Para instalar o BlueJ é preciso acessar o site: http://www.bluej.org/download/download.html

Escolhemos a plataforma, baixamos o programa e executamos o instalador. As telas a seguir mostram o processo de instalação.

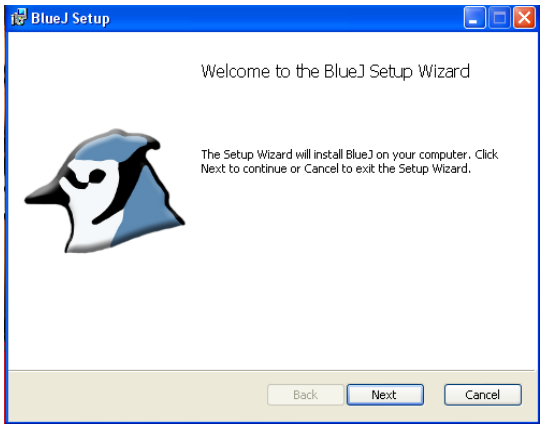


Figura 1 – Tela inicial do instalador do BlueJ – clique em Next

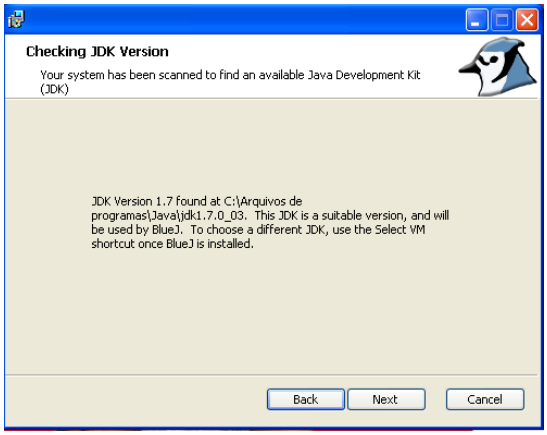


Figura 2 – Tela de verificação do JDK – clique em Next

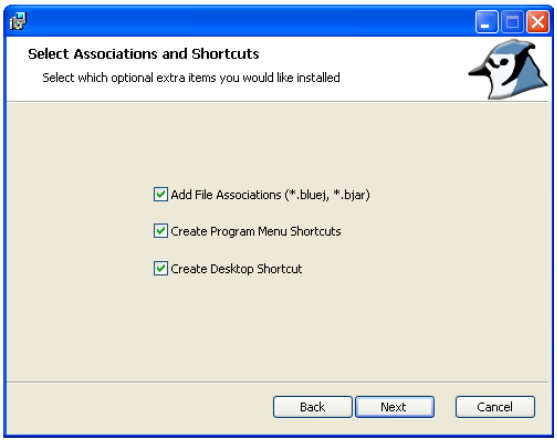


Figura 3 – Associações e atalhos – Marque as opções e clique em Next

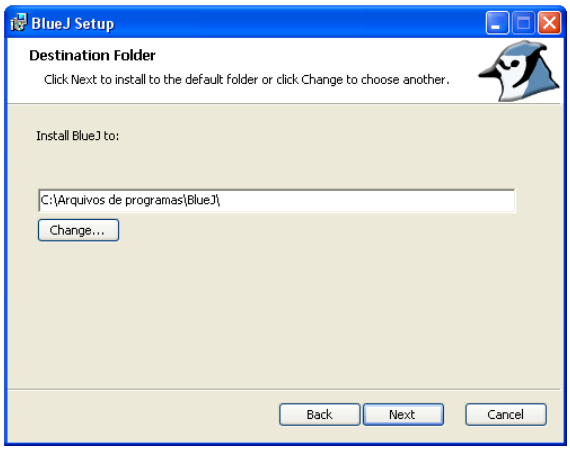


Figura 4 – Local de instalação. O próprio instalador já seleciona. Clique em Next

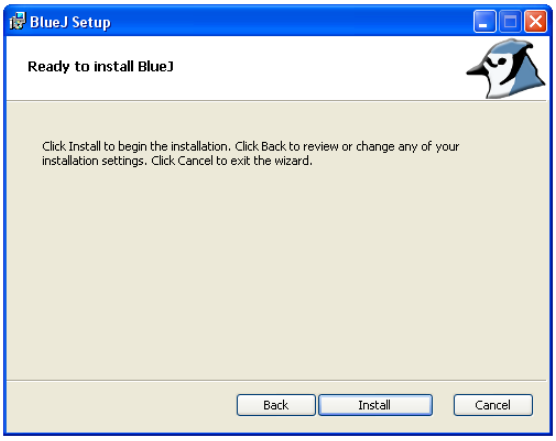


Figura 5 – Pronto para instalação – clique em Install

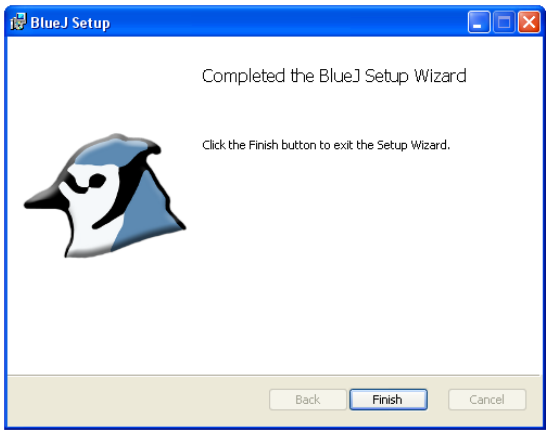


Figura 6 – Clique em Finish para concluir a instalação

CRIANDO UM PROJETO JAVA NO BLUEJ

Para começar a fazer um exercício devemos criar um novo projeto no BLUEJ. Quando criamos um novo projeto automaticamente será criada uma pasta com o nome do projeto. Para criar um novo projeto vamos seguir os passos abaixo:

1º clique na opção **"Project > New Project..."**

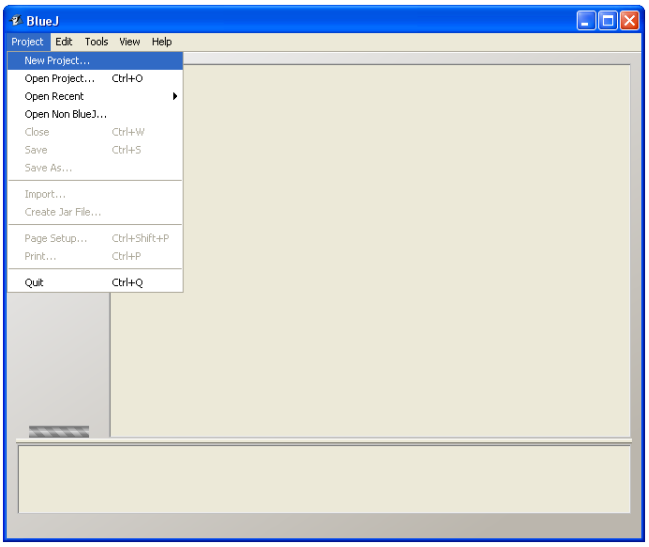


Figura 7 – Criando um novo projeto no BlueJ

2º Em seguida aparecerá uma caixa de diálogo. Nessa caixa de diálogo apague o caminho que aparecerá no "Nome da pasta" e digite o nome que deseja para o projeto, neste caso Exercicio1.

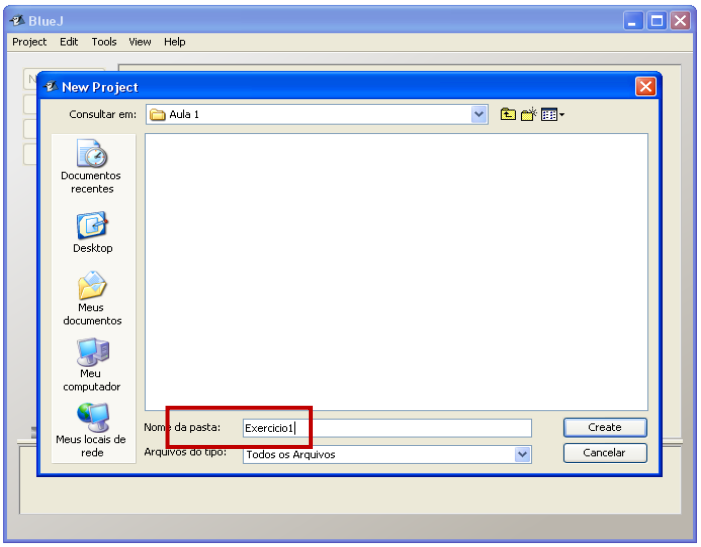


Figura 8 – Nome do projeto

3º Após clique na opção "**Create**".

4º Na tela do projeto clicamos em "**New Class**" para criar uma nova classe.

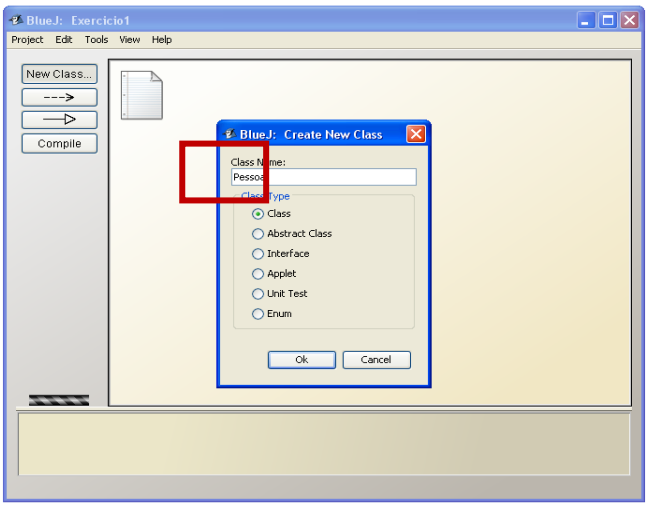


Figura 9 – Criando uma classe. Digite o nome da classe (no exemplo foi Pessoa) e clique em OK

Após a Classe ser criada aparecerá um retângulo conforme figura logo abaixo:

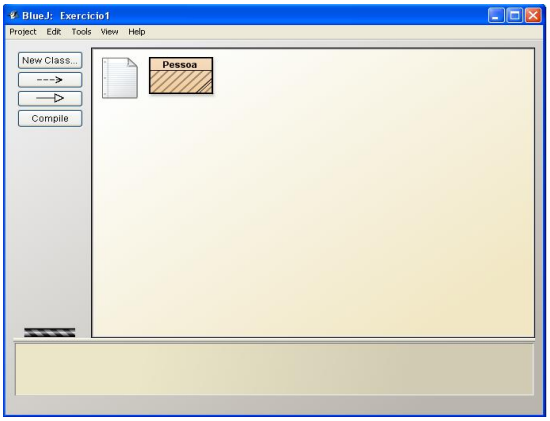


Figura 10 - Classe criada

5º Para começarmos a "programar" devemos dar um duplo clique em cima da Classe Pessoa. O editor de código aparecerá, contendo um exemplo de código.

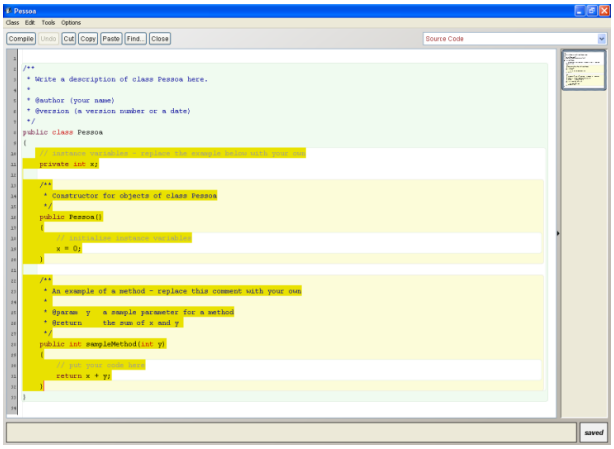


Figura 11

6º O código que aparece é um "código exemplo". Podemos apagar o código e deixar conforme a figura logo abaixo:

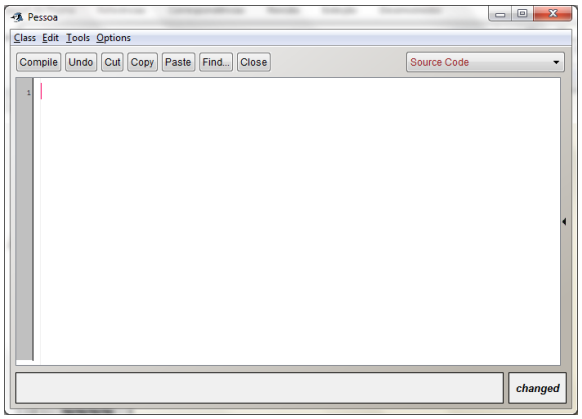


Figura 12

Agora basta digitar o código da nossa classe.

Desenvolvendo a Sintaxe da Classe Pessoa

Diagrama UML

Pessoa
+nome:String +idade:int +salario:double
+falar(texto:String):void +andar(passos:int):void

Observando a classe Pessoa notamos que os atributos e métodos possuem um sinal de + na frente, isso identifica a visibilidade do elemento da classe. Inicialmente utilizaremos a visibilidade pública, simbolizada pelo sinal de +, e mais tarde aprenderemos mais detalhes sobre visibilidade.

Sintaxe JAVA

Para criar uma classe iniciamos o código com a estrutura abaixo:

```
public class NomeClasse{
}
```

As **chaves** funcionam como delimitadores de bloco. Elas determinam o início e o fim da classe. Isto significa que tudo que pertence à classe deve ser escrito dentro destas chaves.

Para escrever o código da classe, seguimos a ordem do diagrama: primeiramente os atributos e em seguida os métodos. Como o diagrama é uma linguagem universal, ao

escrevermos a classe na linguagem Java precisamos traduzir e ajustar conforme as regras de sintaxe Java.

Atributos

Em Java, o sinal de + na frente do atributo ou método se transforma na palavra "*public*", o tipo de dado fica na frente do nome do atributo separados por espaço, e no final da linha o ";" que determina o fim do comando. Observe abaixo os exemplos:

No diagrama	**Na linguagem Java**
+nome:String	public String nome;
+idade:int	public int idade;
+salario:double	public double salario;

Vamos observar como fica a classe com os atributos declarados:

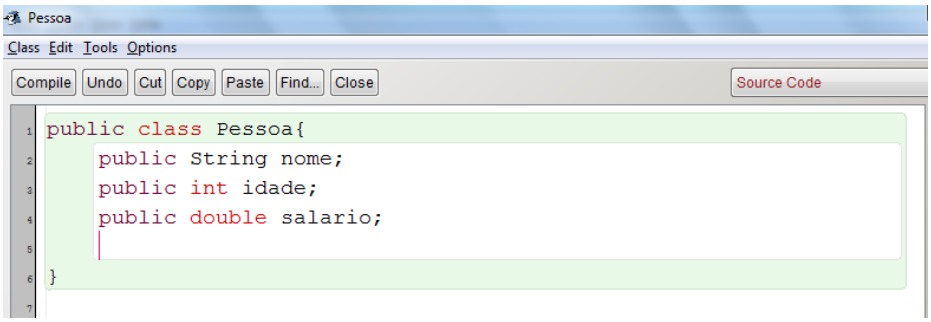


Figura 13

Métodos

Como um método é um bloco de código, ele deve conter seu próprio jogo de chaves, que funcionará como delimitador de início de fim do código do método.

No diagrama	**Na linguagem Java**

+falar(texto:String):void	public void falar(String texto){
	//comandos
	}

+andar(passos:int):void	public void andar(int passos){
	//comandos
	}

Vamos observar como ficou o código da nossa classe:

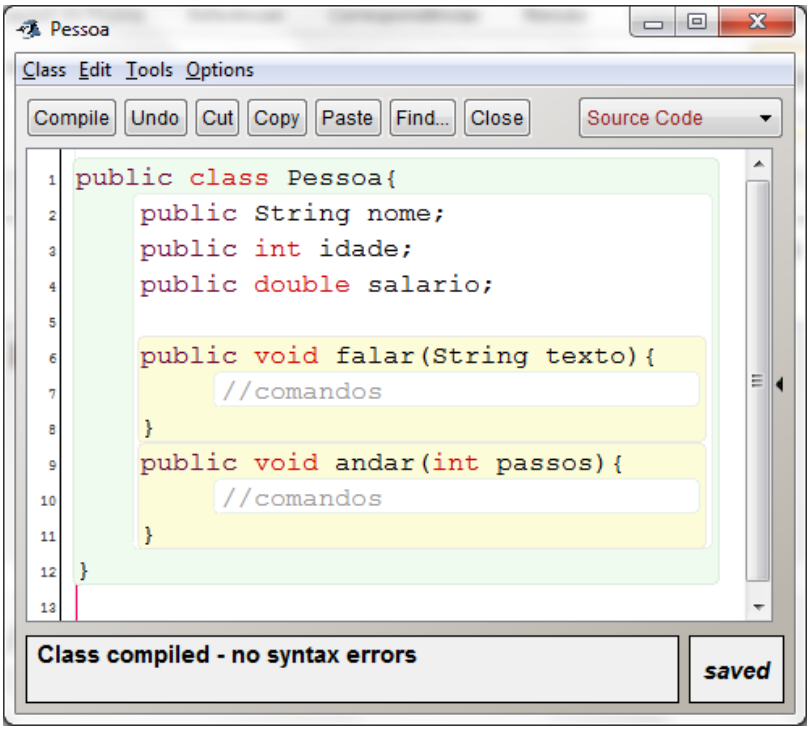


Figura 14

Depois que você digitou o código da classe, clique no botão "**Compile**" ou utilize o atalho CTRL+K para que a IDE verifique se não há nenhum erro na sintaxe. Caso algum erro apareça, o *BlueJ* irá destacar a linha que está com problema. Ele não compila até que toda a classe esteja correta.

> ***Observe*** *que nossos métodos ainda não tem comandos, portanto nossa classe ainda não está completa!*

Tipos de erros

Quando escrevemos um código numa linguagem que estamos recém aprendendo, é natural que se cometa alguns erros, que se esqueça de algumas regras e desta forma, desrespeitando as regras de escrita da linguagem, ela não consegue compreender o comando. Ao desenvolver um programa, podemos cometer dois tipos de erros: os erros de sintaxe e os erros de lógica.

Erros de sintaxe

São os erros que cometemos ao escrever um comando errado, trocar alguma letra minúscula por maiúscula e vice-versa, esquecer um ponto e vírgula, uma chaves, um parêntese, e assim por diante. O compilador acusa o erro e não consegue efetuar a compilação até o erro ser corrigido.

Erros de lógica

Este tipo de erro o compilador não consegue identificar, pois pode ser uma fórmula errada, um comando fora de ordem ou até mesmo a falta de algum comando. Neste caso o compilador consegue entender o código, mas o resultado do programa não é o esperado. Neste caso, é necessário avaliar o código com calma para identificar o erro e gerar uma nova versão.

ENDENTAÇÃO – IDENTAÇÃO

O código apresentado nesta apostila possui alguns espaços entre a margem e o início das declarações de "atributos" e "métodos". Isso é o que chamamos de IDENTAÇÃO - ENDENTAÇÃO e tem como principal objetivo a organização do nosso código para obtermos uma melhor visualização.

Atualmente a programação é feita em conjunto, um dos principais objetivos do paradigma Orientado a Objetos, portanto, nada melhor que organizar o código para outro programador entender melhor.

Resumindo: Sua principal função é facilitar a leitura do código fonte. Para qualquer programador, deve ser um critério a ter em conta, principalmente quando pretendemos partilhar o seu código com outros. A identação facilita também a modificação, seja para correção ou aprimoramento do código fonte. Existem vários estilos de identação, mas consiste basicamente na adição de tabulação no início de cada linha, na quantidade equivalente ao número de bloco em cada linha contida. Ou seja, cada chave aberta, a próxima linha deve ter um recuo, cada chave fechada, esse recuo é retirado.

COMENTÁRIOS NO CÓDIGO

Vamos pensar na realidade das empresas de software, um sistema normalmente é desenvolvido por mais de um programador. Ele então é dividido em etapas e cada programador vai desenvolvendo cada uma. Pensando nisto, temos que lembrar que o nosso código deve ser o mais legível possível, ou seja, deve seguir todas as regras, convenções, identação e muitas vezes até deve conter comentários para explicar o que o método faz, o que deverá ser executado, e assim por diante.

Tipos de Comentários

Temos duas formas de comentar um código, podemos comentar apenas uma linha ou escrever várias linhas de comentários (um bloco).

```
//comentário de linha

/**
*Comentando em um bloco
*inteiro
*/
```

REFERÊNCIAS

CIRIACO, Douglas. **O que é API?** TecMundo, 2009. Disponível em: http://www.tecmundo.com.br/programacao/1807-o-que-e-api-.htm. Acesso em: 25 Out. 2012.