Introdução a Linguagem

Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro
Prof. Edwar Saliba Júnior
Dezembro de 2019

Conteúdo

- Histórico de Java
- Máquina Virtual (JVM)
- *Case Sensitive*
- Tipos Primitivos
- Tipo *String*
- Empacotadoras (*Wrappers*)
- Operadores
- Pacotes (*Packages*)
- Utilizando Outras Classes
- Organização das Pastas
- Modificador de Acess
- *Interface*

Java – Histórico

- Em 1991, na Sun Microsystems, foi iniciado o Green Project, o berço do Java. Os mentores do projeto eram: Patrick Naughton, Mike Sheridan, e James Gosling;
- O objetivo do projeto não era a criação de uma nova linguagem de programação, mas antecipar e planejar a “próxima onda” do mundo digital;
- Eles acreditavam que, em algum tempo, haveria uma convergência dos computadores com os equipamentos e eletrodomésticos comumente usados pelas pessoas no seu dia a dia;
- Para provar a viabilidade desta ideia, 13 pessoas trabalharam arduamente durante 18 meses;
- No verão de 1992 eles fizeram uma demonstração funcional da ideia inicial. O protótipo se chamava *7 (lê-se “*Star Seven*”), um controle remoto com uma interface gráfica *touch screen*. Para o *7, foi criado um mascote o Duke.

Java – Histórico

- O próximo passo era encontrar um mercado para o *7. A equipe achava que uma boa ideia, seria controlar televisões e vídeo por demanda com o equipamento;
- Eles construíram uma demonstração chamada de MovieWood, mas infelizmente era muito cedo para que o vídeo por demanda bem como as empresas de TV a cabo pudessem viabilizar o negócio;
- Permitir ao telespectador interagir com a emissora e com a programação em uma grande rede de cabos era algo muito visionário e estava muito longe do que as empresas de TV a cabo tinham capacidade de entender e comprar. A ideia certa, na época errada!
- O *7 evoluiu e foi ganhou o nome de Oak;
- O estouro da internet aconteceu e rapidamente uma grande rede interativa estava se estabelecendo. Gosling foi incumbido de adaptar o Oak para a internet, e em janeiro 1995 foi lançada uma nova versão do Oak que foi rebatizada para Java.

Java – Histórico

- A velocidade dos acontecimentos seguintes foi assustadora, o número de usuários cresceu rapidamente, grandes fornecedores de tecnologia, como a IBM anunciaram suporte para a tecnologia Java;
- Desde seu lançamento, em maio de 1995, a plataforma Java foi adotada mais rapidamente do que qualquer outra linguagem de programação na história da computação;
- Em 2004 Java atingiu a marca de 3 milhões de desenvolvedores em todo mundo;
- Java tornou-se popular pelo seu uso na internet e hoje possui seu ambiente de execução presente em navegadores, mainframes, sistemas operacionais, celulares, pda's, cartões inteligentes e etc.

Linguagem Java

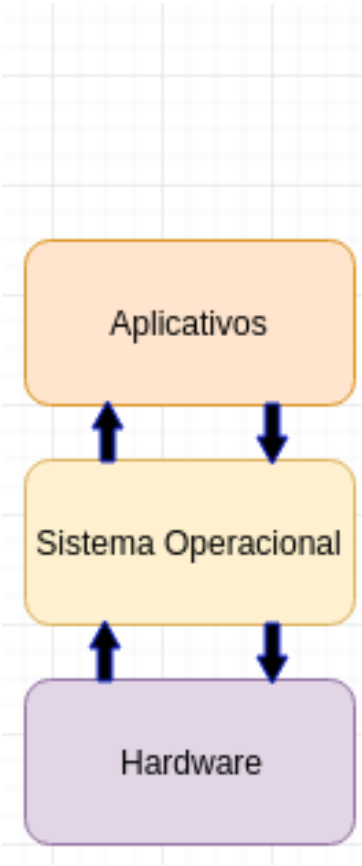
- É uma linguagem de programação orientada a objeto;
- Foi desenvolvida na década de 90 por James Gosling e sua equipe, na empresa Sun Microsystems;
- Diferente das linguagens convencionais:
  - para executar um programa feito em Java, é necessário que se tenha instatalado no Sistema Operacional (SO) uma JVM (*Java Virtual Machine*) e
  - uma vez compilado, um programa pode ser executado em qualquer SO que tenha a JVM instalada.

JVM

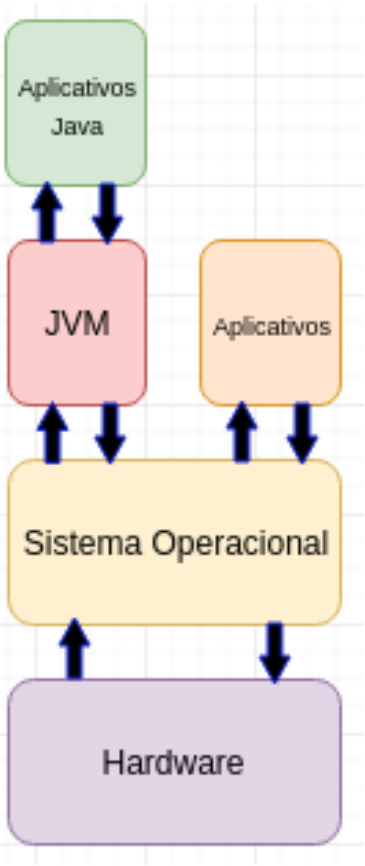
- A compilação de um programa feito em Java não gera um arquivo executável, mas sim um *bytecode* a ser executado pela JVM (*Java Virtual Machine*);
- Um programa fonte em java deve possuir a extensão "**.java**";
- Um programa Java compilado deve possuir a extensão "**.class**".

JVM

Sistemas Convencionais



Sistemas Java



Case Sensitive

- Java, como diversas outras linguagens de programação, é sensível a caixa. Ou seja, faz diferença você escrever:

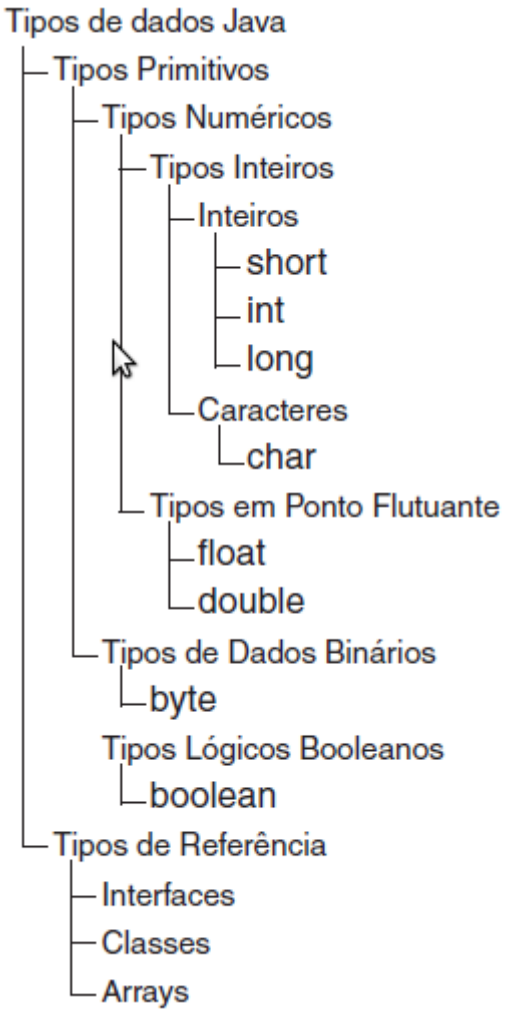
```
Int quantidade;
```

Errado!

- e

```
int quantidade;
```

Tipos de Dados



Tipos Primitivos

- Java possui os seguintes tipos básicos de dados:
  - **boolean**: valores booleanos *true* e *false*;
  - **byte**: inteiro de 8 bits;
  - **short**: inteiro de 16 bits;
  - **int**: inteiro de 32 bits:
    - números inteiros que começam com “0” são octais. Ex.: 077;
    - números inteiros que começam com “0x” são hexadecimais: Ex.: 0xA34;
  - **long**: inteiro de 64 bits;
  - **float**: real de 32 bits;
    - Para indicar que uma constante é float deve-se colocar f ou F no final dela. Ex.: 35.5f;
  - **double**: real de 64 bits;
  - **char**: caracteres.

Tipo `String`

- Em Java, `String` é uma classe pré-definida;
- Cada `String` utilizada no programa é um objeto do tipo `String`;
- Alguns métodos da classe `String`:
  - `charAt(int index)`: devolve o caractere que se encontra na posição `index`;
  - `length()`: retorna o tamanho da `String` e
  - etc.

Empacotadoras (*Wrappers*)

- Para cada tipo primitivo em Java, existe um *Wrapper*, ou seja, uma "Classe Empacotadora" do tipo:

Tipo Primitivo	Classe Empacotadora
boolean	Boolean
byte	Byte
short	Short
char	Character
int	Integer
long	Long
float	Float
double	Double

Métodos das Classes Empacotadoras

- As classes empacotadoras possuem diversos métodos, que podem auxiliar o programador em diversos momentos. Por exemplo:
  - Na conversão de dados:

```
String ss = "123";
int total;

// Converte o valor de "ss" para inteiro.
total = 100 + Integer.parseInt(ss);
```

Operadores

- Aritméticos:

  + - / * % ++ --

- Lógicos:

  ! && || > < >= <= == !=

- Ternário:

  ? : (Exemplo: `A > B ? A += 5 : B -= 3;`)

Pacotes (*Packages*)

- Em Java as classes são organizadas em pacotes;
- Em um pacote deve-se colocar um conjunto de classes relacionadas;
- A palavra reservada **package** indica o pacote ao qual a classe pertence.

Pacotes (*Packages*)

- Exemplo:

```
package rh;
public class Funcionario {
   // Corpo da classe funcionário.
}
```

- A classe **`Funcionario`** está dentro de um pacote chamado **`rh`**. Um pacote corresponde a uma pasta/diretório no SO, onde ficam armazenadas as suas classes.

Pacotes (*Packages*)

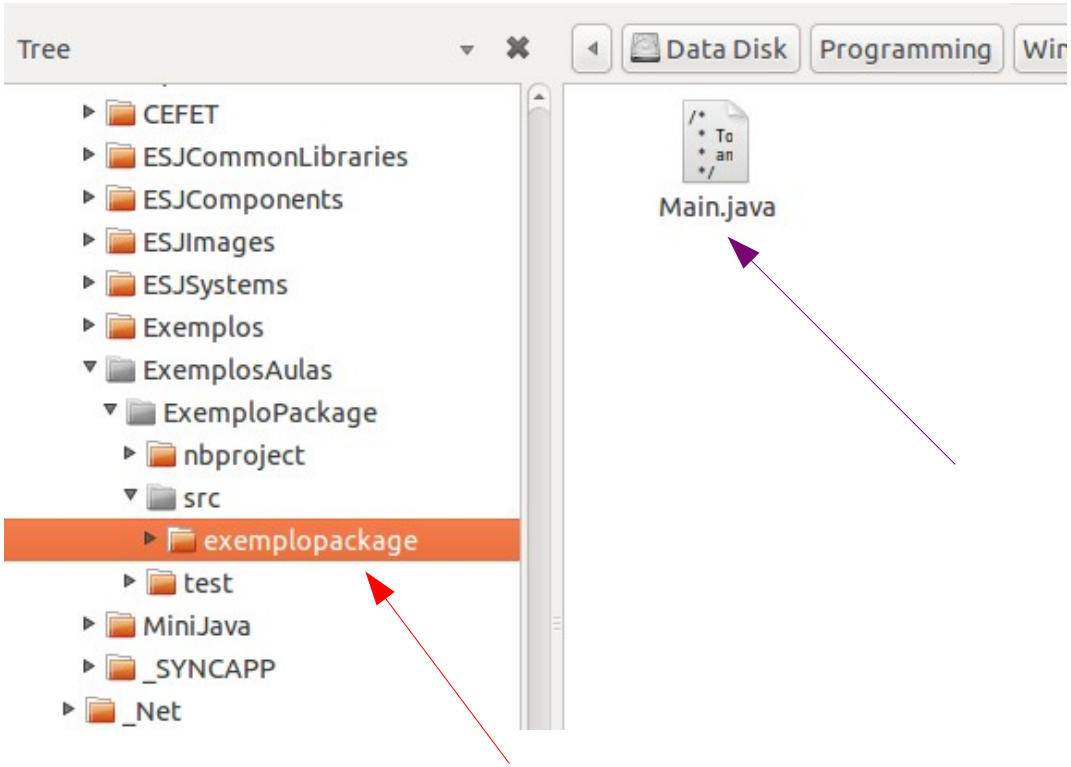
- Exemplo:

```
package exemplopackage;

/**
 *
 * @author Edwar Saliba Júnior
 */
public class Main {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
    }

}
```



Utilizando Outras Classes

- Quando uma classe necessita utilizar uma outra classe, que não esteja em seu pacote, é necessário importar o pacote da classe a ser utilizada;
- Isso é feito incluindo um comando **`import`** no início do código do arquivo “.java”;
- **Exemplo:** se quisermos utilizar a classe **`Date`** da API de Java, temos que importar o pacote onde a classe se encontra:

```
import java.util.*;
```

ou então apenas a classe desejada:

```
import java.util.Date;
```

Organização das Pastas

- Cada IDE tem uma estrutura particular para armazenar os arquivos de um projeto;
- Por exemplo, o NetBeans organiza os arquivos de acordo com a estrutura a seguir:
  - Pasta **build**: contém os arquivos bytecodes compilados (.class) organizados em pacotes;
  - Pasta **dist**: contém o arquivo “.jar” gerado;
  - Pasta **nbproject**: contém arquivos de configuração gerados pelo NetBeans;
  - Pasta **src**: onde ficam os arquivos fontes (.java) organizados em pacotes.

Modificador de Acesso

- Para métodos:
  - ***abstract***: método abstrato, sem corpo;
  - ***final***: método não pode ser redefinido, a partir deste ponto;
  - ***public***: método pode ser acessado por outras classes;
  - ***private***: método só pode ser acessado pela própria classe;
  - ***protected***: método pode ser acessado por classes dentro do mesmo pacote ou pelas subclasses;
  - ***static***: método compartilhado por todos os objetos da classe, com acesso a apenas campos estáticos.

Modificador de Acesso

- Para atributos:
  - ***final***: atributo é uma constante;
  - ***public***: atributo pode ser acessado por outras classes;
  - ***private***: atributo só pode ser acessado pela própria classe;
  - ***protected***: atributo pode ser acessado por classes dentro do mesmo pacote, ou pelas subclasses;
  - ***static***: atributo compartilhado por todos os objetos da classe.

Modificador de Acesso

- A omissão do modificador de acesso implica em um atributo ou método:
  - **público** para as classes que estiverem no mesmo pacote e
  - **privado** para as demais.

Interface do Objeto

- Uma classe é conhecida externamente por sua interface, que descreve os serviços que ela fornece e como eles podem ser utilizados, ocultando a sua implementação;
- Os membros públicos de uma classe constituem a sua interface. Ou seja, somente os membros públicos serão vistos e poderão ser utilizados pelo sistema;
- Informações que fazem parte da interface da classe:
  - nome da classe;
  - assinatura dos construtores;
  - métodos públicos e
  - atributos públicos da classe.

É bom saber!

- Erro de Programação:
  - Declarar mais de uma classe **`public`** no mesmo arquivo, é um erro de compilação.

Exemplo de Código Java

```java
public class OlaMundo {
   /**
   * Método que executa o programa
   * public = É visto em qualquer lugar da aplicação
   * static = é iniciado automaticamente pela JVM, sem precisar de uma instância
   * void = Método sem retorno (retorno vazio)
   * main = Nome do método, que é obrigatorio ser este. Recebe como parâmetro um array de String.
   * String[] args = Array de argumentos que podem ser repassados na chamada do programa.
   */
   public static void main(String[] args) {
      System.out.println("Olá, Mundo!"); //Imprime na tela a frase
   }
}
```

Bibliografia

- DEITEL, H. M.; DEITEL, P. J. **Java Como Programar**; tradução Edson Furmankiewicz; revisão técnica Fábio Lucchini. 6a. ed., São Paulo: Pearson, 2005.
- FERREIRA, Kecia Aline Marques. *Slides* da disciplina de Programação de Computadores II. CEFET-MG, 2009.
- Java. Wikipedia – a enciclopédia livre. Disponível em: <http://pt.wikipedia.org/wiki/Java_%28linguagem_de_programa%C3%A7%C3%A3o%29> Acesso em: 23 jan. 2011.