



METODOLOGIA E LINGUAGEM DE PROGRAMAÇÃO ORIENTADA A OBJETOS

Dr^a. Alana Moraes
alanamm.prof@gmail.com

Ainda não instalou o java!!

- Baixe o JDK (8 ou superior) – Atual Java 12
- Instale
- Have fun!!



Como saber que está instalado
corretamente?



Roteiro

- Estrutura de uma aplicação em Java
- Gerar uma aplicação simples em Java
- Como rodar uma aplicação em Java
 - Eclipse
 - NetBeans
- Diferenças entre C e Java
- Variáveis
- Operadores
- Laços de repetição em Java



Estrutura de uma aplicação em Java:

- Uma aplicação é estruturada por códigos-fontes;
- Um arquivo em **código-fonte** contém uma definição de classe;
- A classe representa uma parte do programa;



Classe

```
public class Cachorro {  
  
}
```

- Uma classe é composta por **métodos** e **atributos**;
- Essas classes podem estar em **um** ou em **vários** arquivos **.java** no programa;



Método

```
public class Cachorro {  
    metodo( ){  
        instrução1;  
    }  
}
```

- Cada método é composto por uma série de instruções;
- **Por enquanto**, pode-se pensar no método como um procedimento ou função;



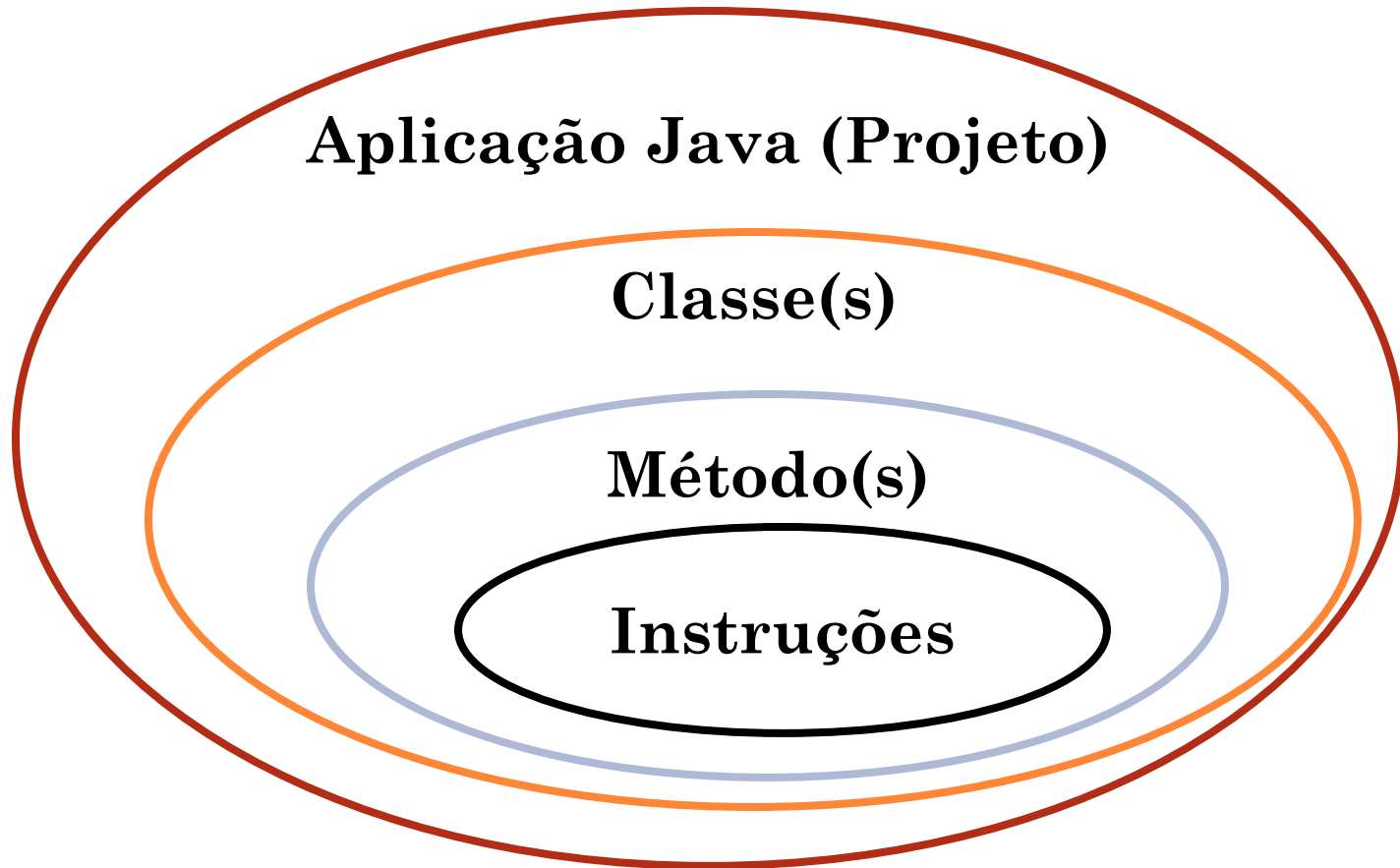
Método

```
public class Cachorro {  
    metodo( ){  
        instrução1;  
    }  
}
```

- E se eu quisesse adicionar outro grupo de instruções? Como ficaria?



Recapitulando ...



Hello World!

- IDE
 - Projeto
 - Classe (pelo menos um deve ter o método main)
 - Programar as classes
 - Testar
- Verifique as classes `.class` geradas



Verifique os seguintes pontos

- Java sempre termina uma linha de ação com ;
- Java precisa de { } para delimitar o começo e final dos métodos (semelhantes às funções) ou de laços e condicionais.
- Comentários tem uma sintaxe diferente
 - //
 - /* */
- Java é uma linguagem Case Sensitive (testem)



Execução de uma aplicação em Java

- O JVM procura por um método chamado:

```
public static void main (String [] args)  
{  
  
  
  
  
  
}
```



Recapitulando ...

- Todo aplicativo Java, precisa ter:
 - Uma classe;
 - Um método main (dentro da classe);

```
public class NomedaClasse {  
    public static void main (String [ ] args) {  
  
    }  
}
```



Em um aplicativo Java só pode haver
UM método main;



Classe principal

- É a classe que contem o método main;
- O que pode inserir no método main?
 - Seu código pode instruir a JVM a:
 - - fazer algo;
 - - fazer algo repetidamente;
 - - fazer algo sob essa condição.



Seu código pode instruir a JVM a ...

... fazer algo:

```
public class ClassePrincipal{  
    public static void main (String [ ] args){  
  
        String nome = “Aula Java”;  
        System.out.println (nome);  
    }  
}
```



Seu código pode instruir a JVM a ...

... fazer algo repetidamente:

```
public class ClassePrincipal{  
    public static void main (String [ ] args){  
        int x = 1;  
  
        while (x<12){  
            x = x + 1;  
            System.out.println(x);  
        }  
        System.out.println("último valor: "+ x);  
    }  
}
```



Seu código pode instruir a JVM a ...

... fazer algo sob essa condição:

```
public class ClassePrincipal{  
    public static void main (String [ ] args){  
        int x = 1;  
        if (x == 1){  
            System.out.print("numero um");  
        } else {  
            System.out.print("não é numero um");  
        }  
    }  
}
```

Exercício

- Faça um programa a média de dois números no Eclipse.
 - `float n1=10, n2=5;`
 - `float media;`
- O usuário não precisa digitar valores (por enquanto).



Python	C	Java
Orientada a Objetos Classes e Objetos (programada estruturada)	Estruturada Funções Structs, pilhas, etc.	Orientada a Objetos Classes e Obj.
Interpretador	Compilador	JVM
Linguagem alto nível	Linguagem médio nível	Linguagem alto nível
Portável	Portabilidade só se consegue com disciplina na programação	Portável
c = classe()	malloc	New
Dinamicamente tipada	Estaticamente tipada	Estaticamente tipada
import	include	import
Tratamento de exceções	Ponteiros	Tratamento de exceções
Sem marcador de fim de linha	Fim de linha ;	Fim de linha ;
Indentação	{ }	{ }

Variáveis

- Uma variável é simplesmente um espaço vago, reservado e rotulado para armazenar dados;
- Toda variável tem um nome de identificação;
- Ao longo da execução essa variável pode receber um valor;



Sintaxe de uma variável

- Na declaração:

int x ;

ou

int x = 2 ;

- Na atribuição:

x = x + 2 ;

ou

x = 2 / (x*3) ;



Identificadores

- Dar nome para variáveis, métodos, classes *etc.* Precisa seguir uma série de regras:
 1. O primeiro caractere de um identificador deve ser uma letra. Os demais caracteres podem ser quaisquer sequências de numerais e letras;
 2. Não apenas os numerais e letras latinas podem ser empregadas, como também letras de quaisquer outro alfabeto;



Identificadores

3. O underscore "_" e o sinal de dólar "\$" são considerados letras e podem ser usados nos identificadores;
4. Os identificadores distinguem o tipo das letras, isto é, as maiúsculas são consideradas distintas das minúsculas;
5. Os identificadores não podem ser palavras reservadas, como: **class**, **for**, **while**, **public**, **etc.**



Identificadores

6. A primeira letra do nome de uma classe é maiúscula. Exemplo: `public class Nome;`
7. Se este nome é composto por várias palavras, elas são escritas juntas (sem usar algum separador) e a primeira letra de cada palavra é maiúscula. Exemplo: `class AloPessoal;`
8. Para o restante: métodos, atributos e referências a objetos, o estilo é similar ao da classe, porém sendo a primeira letra do identificador minúscula. Exemplo:
`boolean luzAcesa;`
`int qtdeParafusos.`



Tipos de dados

- Uma aplicação Java consiste essencialmente em manipulação de dados.
 - Simples tarefas (como escrever mensagens na tela);
 - Até as mais complexas (como resolver equações ou desenhar imagens tridimensionais em animação);



Tipo de dados

Tipo	Descrição
boolean	Tipo lógico que pode assumir o valor true ou o valor false.
char	Caractere em notação Unicode de 16 bits. Serve para a armazenagem de dados alfanuméricos.
byte	Inteiro de 8 bits em notação de complemento de dois. Variáveis deste tipo podem assumir valores entre $-2^7 = -128$ e $2^7 - 1 = 127$.
short	Inteiro de 16 bits em notação de complemento de dois. Os valores possíveis cobrem a faixa de $-2^{15} = -32.768$ a $2^{15} - 1 = 32.767$.
int	Inteiro de 32 bits em notação de complemento de dois. Pode assumir valores entre $-2^{31} = -2.147.483.648$ e $2^{31} - 1 = 2.147.483.647$.
long	Inteiro de 64 bits em notação de complemento de dois. Pode assumir valores entre -2^{63} e $2^{63} - 1$.
float	Representa números em notação de ponto flutuante normalizada em precisão simples de 32 bits em conformidade com a norma IEEE 754-1985. O menor valor positivo que pode ser representado por esse tipo é $1.40239846e-46$ e o maior é $3.40282347e+38$.
double	Representa números em notação de ponto flutuante normalizada em precisão dupla de 64 bits em conformidade com a norma IEEE 754-1985. O menor valor positivo que pode ser representado é $4.94065645841246544e-324$ e o maior valor positivo é $1.7976931348623157e+308$.



Tipo de dados lógicos

- Variável do tipo **boolean**:
 - Pode assumir dois valores : **true** ou **false** ;
 - Compõem as operações lógicas. Ex: **x > w** , **y<=i** ;
- Operações booleanas:

!	Operador lógico de negação
==, !=	Operadores de igualdade e diferença
&&,	Operadores lógicos E e OU .
&=, =, ^=	Operadores de atribuição com operação lógica E, OU e OU-exclusivo

Tipos de dados inteiros

- Refere-se aos tipos **byte**, **int**, **short** e **long** ;

Operação	Descrição
=	Operador de atribuição
==, !=	Operadores de igualdade e diferença
<, <=, >, >=	Operadores de desigualdade (relacionais)
+, -	Operadores unários
+, -, *, /, %	Adição, subtração, multiplicação, divisão e módulo
+=, -=, *=, /=, %=	Operadores de atribuição com adição, subtração, multiplicação, divisão e módulo
++, --	Incremento e decremento
<<, >>, >>>	Operadores de deslocamento de bits
<<=, >>=, >>>=	Operadores de atribuição com deslocamento de bits

Tipo de dado caracter

- Diz respeito a variável do tipo **char**;
- Armazena um caracter Unicode;

Tabela ASCII (códigos de caracteres 0 - 127)

000		016 ▶	032	048 0	064 @	080 P	096 `	112 p
001 ☺		017 ◀	033 !	049 1	065 A	081 Q	097 a	113 q
002 ☹		018 †	034 "	050 2	066 B	082 R	098 b	114 r
003 ♥		019 !!	035 #	051 3	067 C	083 S	099 c	115 s
004 ♦		020 ¶	036 \$	052 4	068 D	084 T	100 d	116 t
005 ♣		021 §	037 %	053 5	069 E	085 U	101 e	117 u
006 ♠		022 ■	038 &	054 6	070 F	086 V	102 f	118 v
007		023 ‡	039 '	055 7	071 G	087 W	103 g	119 w
008		024 ↑	040 (056 8	072 H	088 X	104 h	120 x
009		025 ↓	041)	057 9	073 I	089 Y	105 i	121 y
010		026 →	042 *	058 :	074 J	090 Z	106 j	122 z
011 ♂		027 ←	043 +	059 ;	075 K	091 [107 k	123 {
012 ♀		028 L	044 ,	060 <	076 L	092 \	108 l	124
013		029 ↔	045 -	061 =	077 M	093]	109 m	125 }
014 ♪		030 ▲	046 .	062 >	078 N	094 ^	110 n	126 ~
015 ☼		031 ▼	047 /	063 ?	079 O	095	111 o	127 △

Tipo de dado flutuante

- Refere-se a variável **float** e **double**;
- Além dos possíveis valores numéricos que uma variável de ponto flutuante pode assumir há também os seguintes:
 - menos infinito;
 - mais infinito ;
 - Zero;
 - NAN - not a number.



Tipo de dado flutuante

Operação	Descrição
=	Operador de atribuição
==, !=	Operadores de igualdade e diferença
<, <=, >, >=	Operadores de desigualdade
+, -	Sinais unários
+, -, *, /	Adição, subtração, multiplicação e divisão
+=, -=, *=, /=	Operadores de atribuição com adição, subtração, multiplicação e divisão
++, --	Operadores unários de incremento e decremento



Precedências

Operador	Descrição
. [] () (tipo)	Máxima precedência: separador, indexação, parâmetros, conversão de tipo
+ - ~ ! ++ --	Operadores unários: positivo, negativo, negação (inversão bit a bit), não (lógico), incremento, decremento
* / %	Multiplicação, divisão e módulo (inteiros)
+ -	Adição, subtração
<< >> >>>	Translação (bit a bit) à esquerda, direita sinalizada, e direita não sinalizada (o bit de sinal será 0)
< <= >= >	Operador relacional: menor, menor ou igual, maior ou igual, maior
== !=	Igualdade: igual, diferente
&	Operador lógico e bit a bit
^	Operador lógico ou exclusivo (xor) bit a bit
	Operador lógico ou bit a bit
&&	Operador lógico e condicional
	Operador lógico ou condicional
?:	Condicional: if-then-else compacto
= += -= *= /= %=	Atribuição



Laços de repetição em Java

- Estruturas de condição
 - if ... else
 - switch...case
- Estruturas de repetição
 - for
 - while
 - do ... while



Estruturas condicionais em Java

- `if ... else :`

```
if (condição){  
    instruções;  
    // Entra nesse bloco se a condição for true  
} else if (condição){  
    instruções;  
    // Entra nesse bloco se a condição for true  
} else {  
    instruções;  
    // Entra nesse bloco se a condição for false  
}
```



Estruturas condicionais em Java

- if ... else compacto:

[expressão condicional]?[expressão 1] : [expressão 2]



Exemplo

Expressão:

$$y = (x < 1)? x * x : 2 - x;$$

Equivale a:

```
if (x < 1){  
    y = x * x;  
} else {  
    y = 2 - x;  
}
```



Estruturas condicionais em Java

- switch ... case :
switch (expressao){
 case (constante 1) :
 instruções;
 break;
 case (constante 2) :
 instruções;
 break;
 default:
 instruções;
 break;
}



Estruturas de repetição em Java

- for :

```
for (expressão 1 ; condição ; expressão 2) {  
    instruções;  
}
```



Estruturas de repetição em Java

- while:

```
while (condição) {  
    instruções;  
}
```



Estruturas de repetição em Java

- do ... while :

```
do {  
    instruções  
} while (condição);
```



Vamos pensar em
um exemplo real
Hummm E se eu
precisasse calcular a
área de um
retângulo?



Agora é sua vez!! Faça o mesmo com o cálculo da área de um círculo.



TED

- Faça um programa em Java, que resolva a equação de 2º grau. O usuário ainda não deve digitar os valores, eles devem vir declarados no programa.



Dúvidas?

