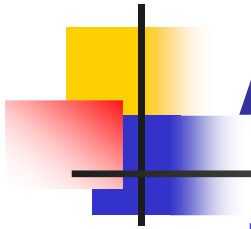
Java Standard Edition (JSE)

Capítulo 02. Tipos primitivos, operadores e controle de fluxo

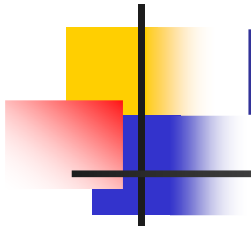
Esp. Márcio Palheta

MSN: marcio.palheta@hotmail.com



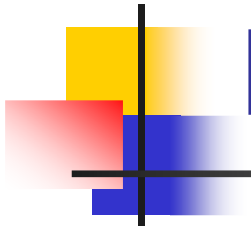
Agenda

- Revisão do aula anterior;
- Motivação – A simplicidade;
- Declaração de variáveis;
- Operadores;
- Tipos primitivos;
- Exercícios: Variáveis e tipos primitivos;
- Casting
- If – Else
- Laços de repetição (While e For)
- Escopo de variáveis;
- Exercícios de fixação



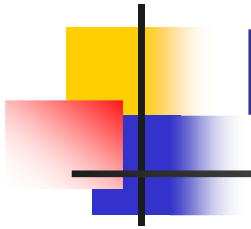
Revisão

- Java – Liguagem OO;
- .java, .class e JVM;
- O nome da classe de ser igual ao do arquivo .java;
- Metodo main();
- Dúvidas ?



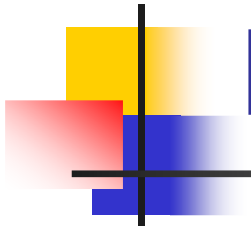
Motivação

- A simplicidade está nos olhos de quem vê: "O doutor e o martelo";
- Seja simples, prático e objetivo. Atinja seu objetivo. Vença - Vídeo "Capoeira Fighter";
- Projetos complexos são formados a partir da reunião de partes simples.
- A cada pequena atividade bem realizada, você se aproxima um pouco mais da vitória: Vídeo "Sinuca e Dominó";



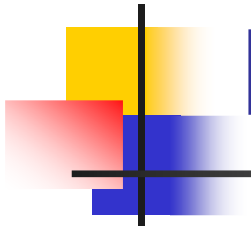
Novos recursos a aprender

- Declaração de variáveis;
- Atribuição de valores;
- Casting e comparação de variáveis;
- Controle de fluxos com if e else;
- Laços de repetição for e while;
- Controle de fluxos com breake e continue;



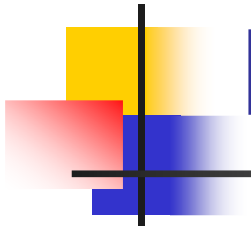
Declaração de variáveis

- Toda variável tem um nome e um tipo que não pode ser alterado após a declaração;
- Sintaxe: **tipoVariavel** nomeVariavel;
- Exemplo: int idade;
- Atribuição de valor: idade = 15;



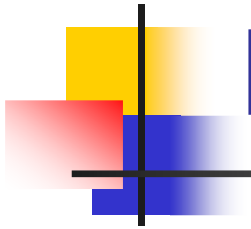
Declaração de variáveis

- Utilizar o valor da variável idade: System.out.println("idade: "+idade);
- Podemos utilizar operadores aritméticos para trabalhar com variáveis numéricas(+, -, *, /)
- O operador % é utilizado para recuperarmos o resto de uma divisão



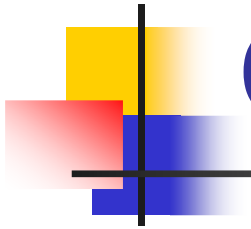
Declaração de variáveis

- int quatro = 2 + 2;
- int tres = 5 – 2;
- int oito = 4 * 2;
- int dezesseis = 64 / 4;
- int um = 5 % 2; // 5 dividido por 2 dá 2 e tem resto 1;
- É considerada apenas a parte inteira do resto da divisão



Exercício 01

- Escreva um programa java que possui uma variável inteira, que receba inicialmente a sua idade;
- Imprima o valor da variável idade;
- A partir da variável criada, imprima a sua idade daqui a 10 anos;



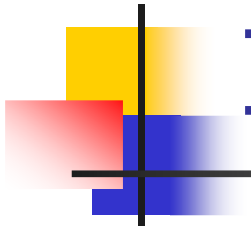
Operadores

- Um operador produz um novo valor a partir de um ou mais argumentos
- Os operadores em Java são praticamente os mesmos encontrados em outras linguagens

  +, -, /, *, =, ==, <, >, >=, && etc.
- A maior parte dos operadores só trabalha com valores de tipos primitivos.
- Exceções:

  + e += são usados na concatenação de strings

  !=, = e == são usados também com objetos (embora não funcionem da mesma forma, quanto aos valores armazenados nos objetos);



Incremento e decremento

- Exemplo

  int a = 10;

  int b = 5;
- Incrementa ou decrementa antes de usar a variável

  int x = ++a; // a contém 11, x contém 11

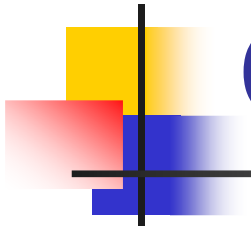
  int y = --b; // b contém 4, y contém 4
- Atribuição feita **DEPOIS**;
- Incrementa ou decrementa depois de usar a variável

  int x = a++; // a contém 11, x contém 10

  int y = b--; // b contém 4, y contém 5
- A atribuição foi feita **ANTES**!

Lista de operadores java

OPERADOR	FUNÇÃO	OPERADOR	FUNÇÃO
+	Adição	~	Complemento
-	Subtração	<<	Deslocamento à esquerda
*	Multiplicação	>>	Deslocamento à direita
/	Divisão	>>>	Desloc. a direita com zeros
%	Resto	=	Atribuição
++	Incremento	+=	Atribuição com adição
--	Decremento	-=	Atribuição com subtração
>	Maior que	*=	Atribuição com multiplicação
>=	Maior ou igual	/=	Atribuição com divisão
<	Menor que	%=	Atribuição com resto
<=	Menor ou igual	&=	Atribuição com AND
==	Igual	|=	Atribuição com OR
!=	Não igual	^=	Atribuição com XOR
!	NÃO lógico	<<=	Atribuição com desl. esquerdo
&&	E lógico	>>=	Atribuição com desloc. direito
||	OU lógico	>>>=	Atrib. C/ desloc. a dir. c/ zeros
&	AND	? :	Operador ternário
^	XOR	(*tipo*)	Conversão de tipos (cast)
|	OR	instanceof	Comparação de tipos

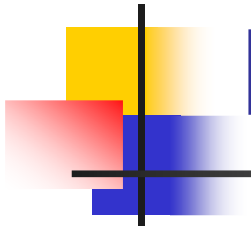


Operador ternário (if-else)

- Retorna um valor ou outro, dependendo do resultado de uma expressão booleana:
  variavel = (expressão) ? valor_true: valor_false;
- Exemplo:

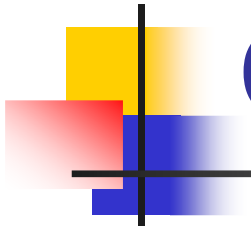
  int idade = 15;
- **String tipo = (idade >= 18) ? "Adulto" : "Adolescente"**
- Aprecie com moderação:

  Pode tornar o código difícil de entender;



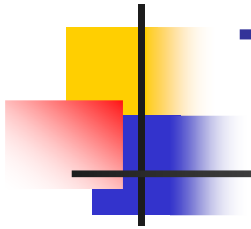
Exercício 02

- Crie um programa JAVA que imprima:
- "Criança" para pessoas até 12 anos;
- "Adolescente" para pessoas até 17 anos;
- "Adulto" para pessoas a partir de 18 anos;



Convenções de código

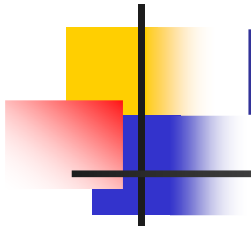
- Nomes de Classes;
- Nomes de métodos;
- Nomes de variáveis;
- Comentários de linha (// linha);
- Comentários em bloco (/* bloco */);



Tipos primitivos

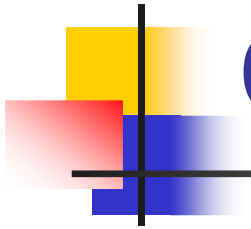
- A linguagem java oferece vários tipos de dados, que podem ser manipulados de acordo com a situação enfrentada;
- Existem os tipos primitivos e tipos de referência, que se referem a arrays, classes e interfaces;
- Os tipos de referência serão discutidos em capítulos futuros;
- A seguir, uma lista com os tipos primitivos

Tipo	Descrição
boolean	Pode ser contido em 1 bit, porém o seu tamanho não é precisamente definido. Assume os valores true ou false.
char	Caractere em notação Unicode de 16 bits. Serve para armazenar dados alfanuméricos. Também pode ser usado como um dado inteiro com valores na faixa entre 0 e 65535.
byte	Inteiro de 8 bits em notação de complemento de dois. Pode 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 representável por esse tipo é 1.40239846e-46 e o maior é 3.40282347e+38. 4 bytes de tamanho e 23 dígitos binários de precisão.
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 representável é 4.9406564584124654e-324 e o maior é 1.79769313486231570e+308. 8 bytes de tamanho e 52 dígitos binários de precisão.



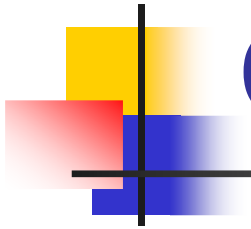
Exercício 03: Tipos primitivos

- Na empresa onde trabalhamos, há tabelas com o quanto foi gasto em cada mês. Para fechar o balanço do primeiro trimestre, precisamos somar o gasto total. Sabendo que, em Janeiro, foram gastos 15000 reais, em Fevereiro, 23000 reais e em Março, 17000 reais.
- Faça um programa que imprima o gasto mensal, além de calcular e imprimir o gasto total no trimestre.



Casting

- Às vezes, precisamos que um número quebrado seja arredondado e armazenado num número inteiro. Para fazer isso sem que haja o erro de compilação, é preciso ordenar que o número quebrado seja **moldado (casted)** como um número inteiro. Esse processo recebe o nome de **casting**.
- Sintaxe: variavel_A = (tipo)variavel_B;
- double d3 = 3.14;
- int i = (int) d3;

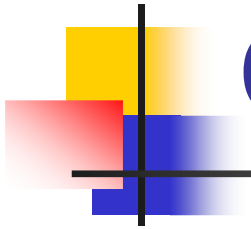


Casting (cont...)

- O casting foi feito para moldar a variável d3 como um int. O valor de i agora é 3.
- O mesmo ocorre entre valores int e long.
  long x = 10000;
  int i = x; *// nao compila, pois pode estar perdendo informação*
- E, se quisermos realmente fazer isso, fazemos o casting:
  long x = 10000;
  int i = (int) x;

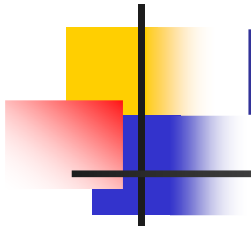
Motivação - Superação

- Você pode superara as expectativas:
  - Vídeo 02.03 Voce Pode Gol;
- Esteja sempre preparado para as adversidades. A final, imprevistos acontecem:
  - Vídeo 02.04 Acupuntura;
- Sempre seja tranquilo:
  - 02.05 crianças



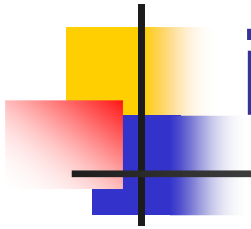
Controle de fluxo

- O controle do fluxo da execução em Java utiliza os mesmos comandos existentes em outras linguagens
- Repetição: for, while, do-while
- Seleção: if-else, switch-case
- Desvios (somente em estruturas de repetição): continue e break



Expressões booleanas

- Todas as expressões condicionais usadas nas estruturas for, if-else, while e do-while são expressões booleanas;
- O resultado das expressões deve ser true ou false;
- Em java, não há conversão automática entre booleanos: (x=5) gera erro;
- Operadores: !=, ==, >, < etc;



if-else

- Sintaxe:
- if (expressão booleana) {

  bloco de instruções

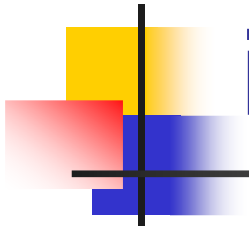
  } else if (expressão booleana) {

  instruções

  } else {

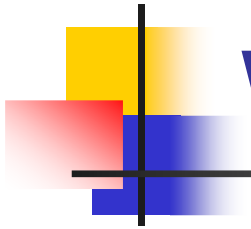
  instruções

  }



if-else: exemplo

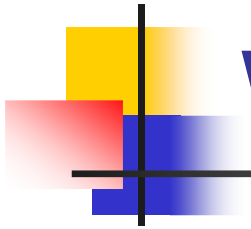
```
if(qtdeLados == 3){
    System.out.println("Triângulo");
}else if(qtdeLados == 5){
    System.out.println("Pentágono");
}else{
    System.out.println("Outra figura");
}
```



while e do-while

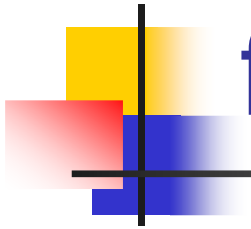
- Sintaxe:
- while (expresão booleana)

```
{
    instruções;
}
do
{
    instruções;
} while (expresão booleana );
```



while e do-while: exemplo

```
int x = 0;
while (x < 10) {
        System.out.println ("item " + x);
        x++;
    }
int y = 0;
do {
        System.out.println ("item " + y);
        y++;
    } while (y < 10);
```

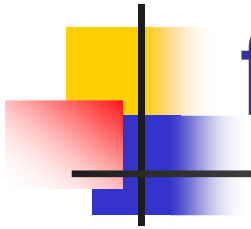


for

Sintaxe:

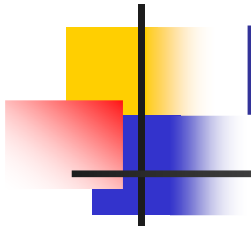
- for (inicialização;
  expressões booleanas;
  passo da repetição)

```
{
    instruções;
}
```



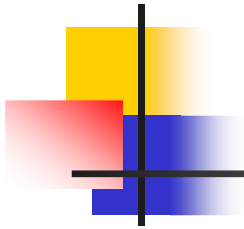
for: exemplo

- for (int x = 0; x < 10; x++) {
  System.out.println ("item " + x);
  }
- for (int x = 0, int y = 25;
  x < 10 && (y % 2 == 0);
  x++, y = y - 1)
  {
  System.out.println (x + y);
  }



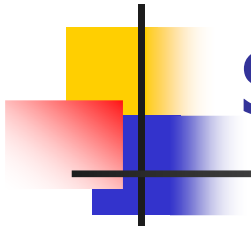
break e continue

- break: interrompe a execução do bloco de repetição.
  - Continua com a próxima instrução, logo após o bloco.
- continue: interrompe a execução da iteração;
  - Testa a condição e reinicia o bloco com a próxima iteração.



break e continue: exemplo

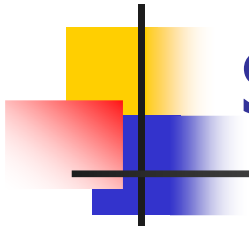
```
while (!terminado) {
        passePagina();
        if (alguemChamou == true) {
                break; // caia fora deste loop
        }
        if (paginaDePropaganda == true) {
                continue; // pule esta iteração
        }
        leia();
}
restoDoPrograma();
```



switch (case)

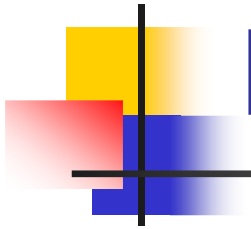
- Sintaxe:
- switch(seletor_inteiro) {

```
        case valor_inteiro_1 :
                instruções;
                break;
        case valor_inteiro_2 :
                instruções;
                break;
        (...)
        default:
                instruções;
}
```



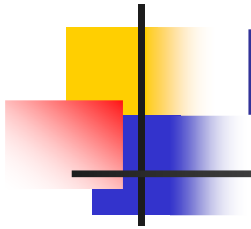
switch (case): exemplo

```
int numero = 2;
switch(numero) {
        case 1 :
                System.out.println ("número UM");
                break;
        case 2 :
                System.out.println ("número DOIS");
                break;
        default:
                 System.out.println ("outro número");
}
```



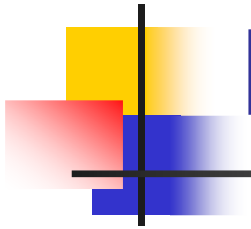
Entrada e Saída de dados

- Entrada do usuário:
  - JOptionPane.showInputDialog(null, String);
- Exibir resultado na tela:
  - JOptionPane.showMessageDialog(null, String);
- Pacote: javax.swing.JOptionPane
- **OBS:** a API gráfica Swing não é foco do nosso curso;



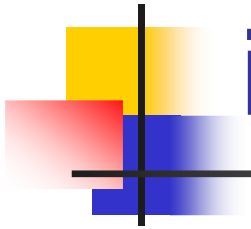
Exemplo:

```
public static void main(String[] args) {
    String nome =
    JOptionPane.showInputDialog("Nome:"
);
    JOptionPane.showMessageDialog(null,
"Nome informado: ");
    System.exit(0);
}
```



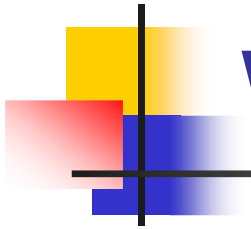
Exercício 04:

- Implemente o trecho de código anterior em uma classe java, execute a aplicação e veja o resultado;
- Utilize a classe JOptionPane para a implementação dos próximos exercícios;



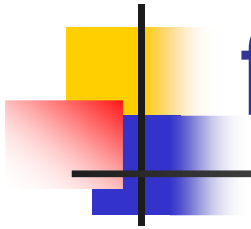
if-else: exercício 05

- O Cine Grande Zé tem apenas uma sala em funcionamento, onde está passando um filme com a classificação inadequada a menores de 14 anos.
- Escreva um programa java que leia as variáveis nome, idade e classificação;
- Imprima nome, idade e se a pessoa pode assistir ao filme em cartaz ou não;



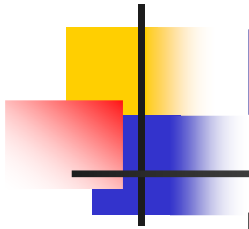
while e do-while: exercício 06

- Escreva um programa java que imprima todos os números pares entre 1 e 15, usando while;
- Escreva um programa java que imprima os 10 maiores números negativos, usando do-while;



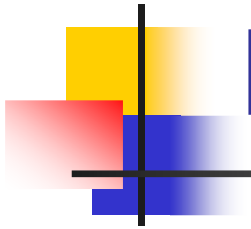
for: exercício 07

- Escreva um programa java que imprima os 20 primeiros números naturais;



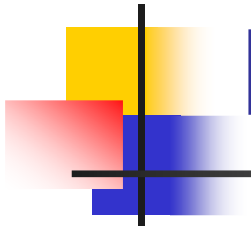
Exercícios de fixação

- Não copie e cole de um exercício já existente! Aproveite para praticar.
- 1) Imprima todos os números de 150 a 300.
- 2) Imprima a soma de 1 até 100.
- 3) Imprima todos os múltiplos de 3, entre 1 e 100.
- 4) Imprima os fatoriais de 1 a 10. O fatorial de um número n é n * n-1 * n-2 ... até n = 1. Lembre-se de utilizar os parênteses.
  - O fatorial de 0 é 1; O fatorial de 1 é (0!) * 1 = 1;
  - O fatorial de 2 é (1!) * 2 = 2 O fatorial de 3 é (2!) * 3 = 6;
  - O fatorial de 4 é (3!) * 4 = 24
- Faça um for que inicie uma variável n (número) como 1 e fatorial (resultado) como 1 e varia n de 1 até 10:
  - for (int n=1, fatorial=1; n <= 10; n++) {...



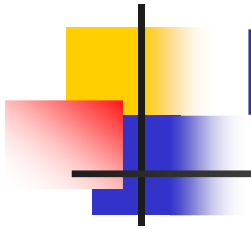
Métodos

- **Sintaxe:**
- public static tiporetorno nomeMetodo(){
  - [bloco de comandos]
- }
- **Exemplo:**
- public static int soma(int a, int b, int c){
  - return a + b + c;
- }



Bibliografia

- Java - Como programar, de Harvey M. Deitel
- Use a cabeça! - Java, de Bert Bates e Kathy Sierra
- (Avançado) Effective Java Programming Language Guide, de Josh Bloch



Referências WEB

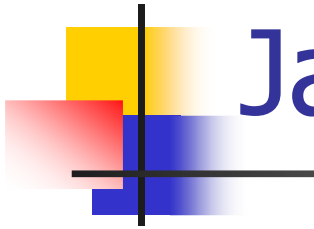
- SUN: www.java.sun.com

Fóruns e listas:

- Javaranch: www.javaranch.com
- GUJ: www.guj.com.br

Apostilas:

- Argonavis: www.argonavis.com.br
- Caelum: www.caelum.com.br



Java Standard Edition (JSE)

Capítulo 02. Tipos primitivos, operadores e controle de fluxo

Esp. Márcio Palheta

MSN: marcio.palheta@hotmail.com