

Module 3: Introduction to Java

Module Objectives

- No final deste módulo, os participantes poderão:
 - Identifique e crie código fonte estruturado corretamente;
 - Articular a importância do método "principal";
 - Identifique e crie declarações e expressões apropriadas;
 - Descreva o conceito de variáveis;
 - Identifique as regras para o escopo das variáveis;
 - Descreva os tipos de dados primitivos e seus intervalos;
 - Identifique as regras para atribuir valores a variáveis;
 - Descrever o conceito de matrizes;
 - Identifique operadores Java e explique seus diferentes usos;



Module Objectives (cont.)

- No final deste módulo, os participantes poderão:
 - Explicar o conceito de controle do 'fluxo do programa';
 - Controlar 'fluxo de programa' usando as diferentes instruções de fluxo de controle;
 - Descrever o conceito de métodos;
 - Identifique e crie métodos estruturados corretamente;
 - Descreva o conceito de "chamada de método";

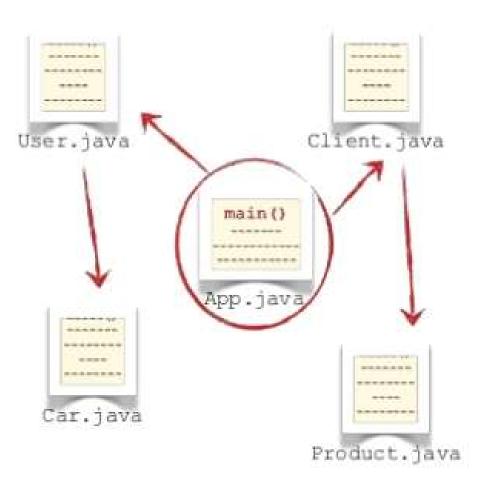


Estrutura do programa fonte

"Estrutura do programa fonte"

 Os aplicativos Java são compostos de arquivos de texto que terminam com um sufixo ".java".

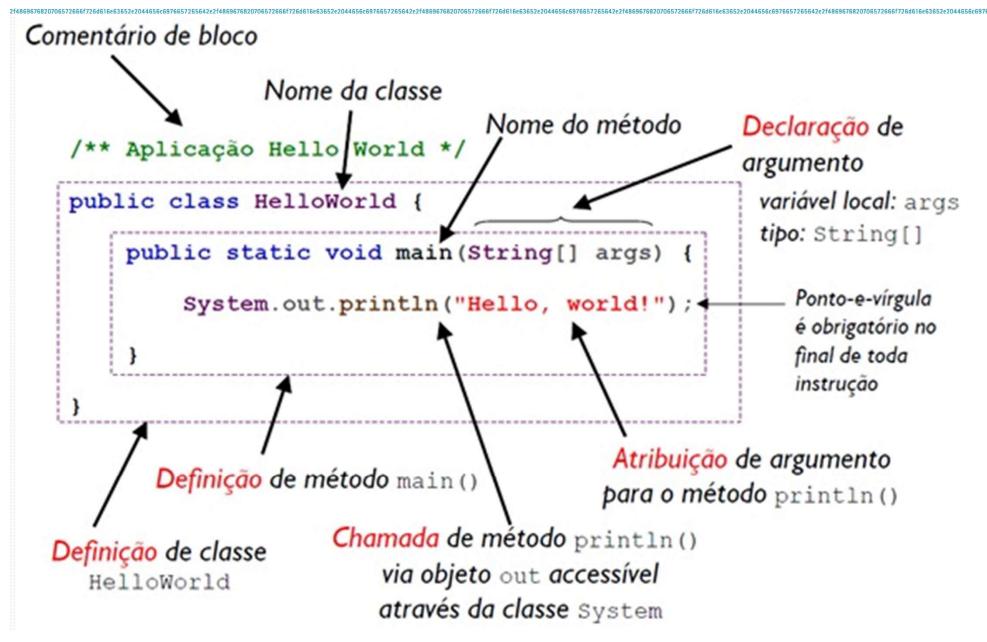
Java



- Os aplicativos Java são compostos de arquivos de texto que terminam com um sufixo ".java".
- Cada arquivo de origem Java consiste em uma 'declaração de classe', que segue uma estrutura específica.
- Os arquivos de origem Java são compilados em arquivos '.class', que são executados pelo interpretador Java.

i

Refer to the MainSample.java sample code



declaration order

1. Package declaration

Usado para organizar uma coleção de classes relacionadas.

2. Import statement

Usado para fazer referência a classes declaradas em outros pacotes.

3. Class declaration

Um arquivo de origem Java pode ter várias classes, mas apenas uma classe pública é permitida.

```
/*
 * Created on Jun 25, 2008
 * First Java Program
package sef.module3.sample;
import java.lang.*;
/**
 * @author SEF
>public class MainSample{
   public static void main(String[] args) {
          // print a message
          System.out.println("Welcome to Java!");
```

Comments /* * Created on Jun 25, 2008 1. Block Comment * First Java Program /* * insert comments here package sef.module3.sample; import java.lang.*; 2. Documentation Comment /** * insert documentation * @author SEF * / public class MainSample{ 3. Single Line Comment // insert comments here public static void main(String[] args) { // print a message The compiler ignores comments. System.out.println("Welcome to Java!");

Comments

1. Block Comment

```
/*
* insert comments here
*/
```

2. Documentation Comment

```
/**
* insert documentation
*/
```

3. Single Line Comment

// insert comments here

The compiler ignores comments.

Whitespaces

Tabulações e espaços são ignorados pelo compilador. Usado para melhorar a legibilidade do código.

```
/*
 * Created on Jun 25, 2008
 * First Java Program
package sef.module3.sample;
import java.lang.*;
/**
 * @author SEF
public class MainSample{
   public static void main(String[] args) {
        // print a message
         System.out.println("Welcome to Java!");
```

Class

- A classe é o componente fundamental de todos os programas Java..
- Todo programa Java inclui pelo menos uma definição de classe pública
- Uma definição de classe contém todas as variáveis e métodos que fazem o programa funcionar. Isso está contido no corpo da classe indicado pelas chaves de abertura e fechamento
- O nome da declaração de classe pública deve ser o mesmo que o nome do arquivo (diferencia maiúsculas de minúsculas).

```
/*
 * Created on Jun 25, 2008
 * First Java Program
package sef.module3.sample;
import java.lang.*;
/**
 * @author SEF
public class MainSample{
   public static void main(String[] args){
         // print a message
         System.out.println("Welcome to Java!");
```

Braces

- Chaves são usadas para agrupar instruções ou bloco de códigos.
- A chave esquerda ({) indica o início de um corpo de classe, que contém variáveis e métodos da classe;
- A chave esquerda também indica o início de um corpo do método;
- Para cada chave esquerda que abre uma classe ou método, você precisa de uma chave direita correspondente (}) para fechar a classe ou o método;
- Uma chave direita sempre fecha sua chave esquerda mais próxima;.

```
/*
 * Created on Jun 25, 2008
 * First Java Program
package sef.module3.sample;
import java.lang.*;
/**
 * @author SEF
public class MainSample{
   public static void main(String[] args) {
         // print a message
         System.out.println("Welcome to Java!");
```

The 'main()' Method

O método "main" é onde a execução de um aplicativo java começa;

```
Syntax: public static void main( String[] args ) {
//Main method implementation goes here
}
```

 Classes que possuem um método main() declarado dentro servem como ponto de partida do aplicativo;

Refer to the MainSample.java sample code.

The 'main()' Method

main() method

Esta linha inicia o método main (). Esta é a linha na qual o programa começará a executar;.

String args[]

Declara um parâmetro chamado args, que é uma matriz de String. Representa argumentos da linha de comando;

```
* Created on Jun 25, 2008
 * Hello World Program
package sef.module3.sample;
   @author SEF
public class MainSample {
   public static void main(String[] args) {
          //Prints out 'Hello World!'
          System.out.println( "Hello World!" );
                 Terminating character
          O ponto e vírgula (;) é o caractere de
          terminação de qualquer instrução
```

java.

Java Keywords

abstract	default	if	package	synchronized
assert	do	implements	private	this
boolean	double	import	protected	throw
break	else	instanceof	public	throws
byte	extends	int	return	transient
case	false	interface	short	true
catch	final	long	static	try
char	finally	native	strictfp	void
class	float	new	super	volatile
continue	for	null	switch	while

goto

const



Perguntas:

Qual método inicia a execução do aplicativo?

MAIN()

Simbolo de delimita o bloco de comandos?

{}

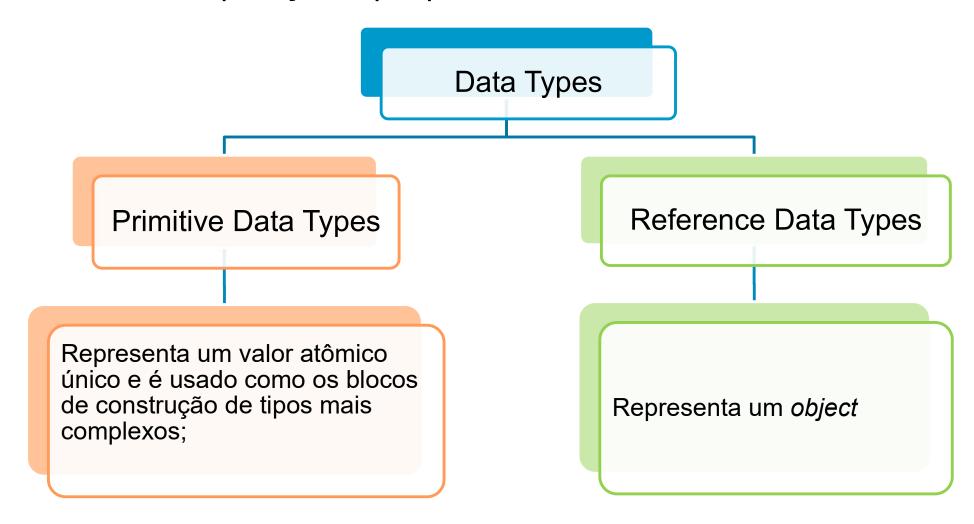
JAVA - M3 - 1

Tipo de Dados

"Tipo de Dados"

Data Types

 Um tipo de dados determina os valores que uma variável pode conter e as operações que podem ser executadas nela;



Data Types

Data Types

Um tipo de dados primitivo com um valor atribuído:

int
$$x = 42$$
;

Um tipo de dados de referência com um valor atribuído:

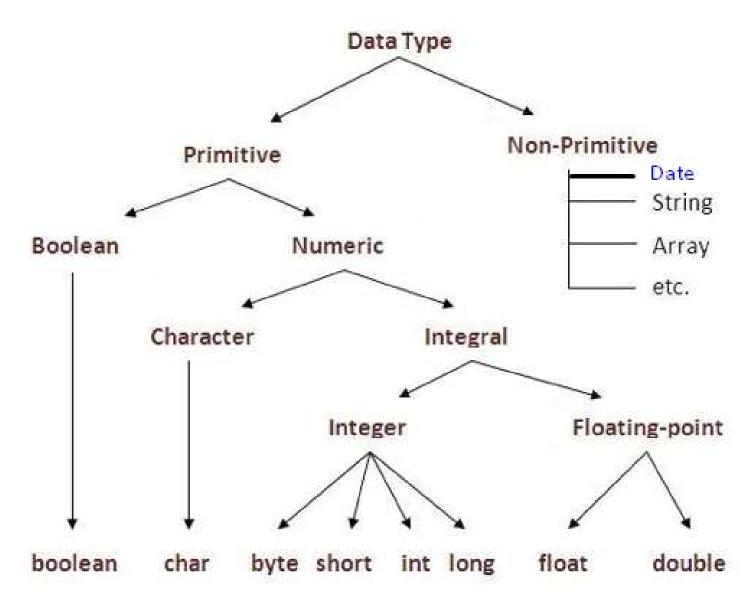
Date today = new Date();

x ----- 42

today -----

Day = 25 Month = June Year = 2011

Data Types



Primitive Data Types

Data Type	Size(bits)	Range		Default
Keyword		Minimum	Maximum	
boolean	1 bit	false	true	false
byte	8 bits	-128	127	0
short	16 bits	-32,768	32,767	0
char	16 bits	'\u0000' (0)	'\uFFFF' (65535)	'\u0000'
int	32 bits	-2,147,483,648	2,147,483,647	0
long	64 bits	-2 ⁶³	2 ⁶³ -1	0
float	32 bits	~1.4e ⁻⁰⁴⁵	32-bit IEEE 754 Floating Point ~3.4e ⁺⁰³⁸	0.0
double	64 bits	~4.9e ⁻³²⁴	64-bit IEEE 754 Floating Point ~1.8e+308	0.0



Perguntas:

O que são dados tipo primitivo?

Dados que contém valores atômicos

O que são dados do tipo de referência?

Dados que referenciam um objeto na memória

JAVA - M3 - 2

"Variáveis"

Variável é uma região de memória do computador que contém um valor e conhecida por um nome especificado pelo usuário.



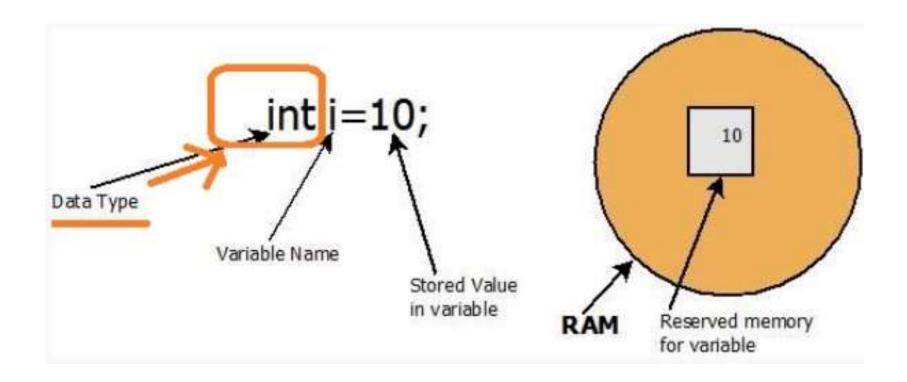
- A declaração especifica as propriedades de uma variável, como seu tipo de dados e o nome com o qual ela seria identificada;
- Declaração básica de variável em Java:

Syntax: <data type> <identifier_name>;

Examples: int myInteger;

String myFirstName;

Date the Date Today;



Variable Assignment

 As variáveis recebem um valor usando o operador de atribuição igual (=)

```
Syntax: <variable_name> = <the_value>;
```

Example: myInteger = 0;

 O tipo de dado do valor que está sendo atribuído deve ser compatível com o tipo de dado da variável que recebe o valor;



Identificador

 Nome dado aos objetos utilizados no programa (variáveis, constantes, funções, etc.)

Atribuição

 Comando que define ou redefine o valor armazenado em uma variável

Expressão

 Pode ser um valor ou um conjunto de comandos que resulta em um valor

Variables: Initialization

Inicializando variáveis com tipo de dados primitivo

```
Syntax: <identifier_name> = <initial_value>;
```

Example: myInteger = 0;

Inicializando variáveis com tipo de dados de referência

```
Syntax: <identifier_name> = <initial_value>;
```

Examples: myFirstName = "Jason";

theDateToday = new Date();

- Não use espaços.
- Letras maiúsculas e minúsculas são diferentes.

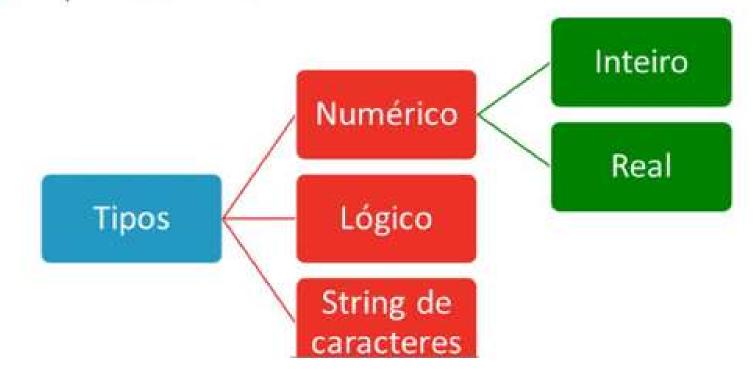
□ Variáveis Area e area são distintas.



Se você declarar uma variável já existente, o conteúdo anterior será perdido.

- O computador executa um script linha por linha.
- Quando alteramos o valor de uma variável, o valor anterior é substituído pelo novo.
- O valor antigo fica perdido para sempre.

- O tipo define a natureza dos dados que a variável armazena.
- Tipos mais comuns



Inteiros (int)

- São números sem a parte fracionária.
- Exemplos: 1 | 0 | -5 | 567

Reais (float)

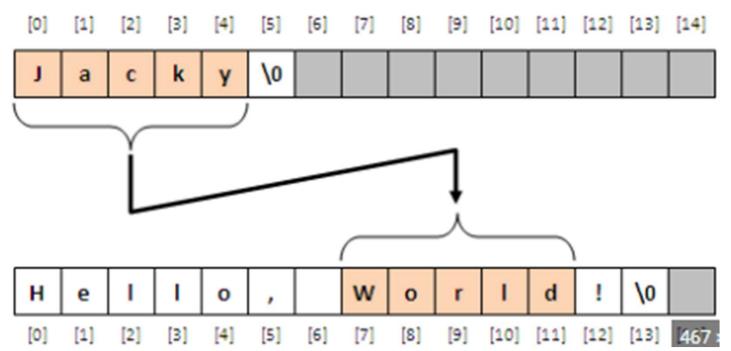
- São números com parte fracionária.
- Também conhecidos como ponto flutuante.
- Exemplos: 1.0 | 3.1415 | 2.7182

- Números inteiros e de ponto flutuante são representados de maneiras distintas na memória do computador.
- na maioria das linguagens de programação,
 utilizamos o ponto e não a vírgula
 como separador entre a parte inteira e a parte fracionária de um número ponto flutuante.

- Variáveis de ponto flutuante são representações da realidade na memória do computador.
- O conjunto dos números reais é infinito, mas o espaço de armazenamento em memória é um recurso finito.
- Logo, somente alguns elementos do conjunto de números reais podem ser representados em um computador.

```
>>> 2 < 3
True
>>> 3 < 2
False
>>> (15/3) < 5
False
>>> (15/3) == 5
True
```

- Uma string (= corda) é uma cadeia de caracteres.
- Uma cadeia de caracteres é um sequência de símbolos, tais como letras, números, sinais de pontuação, etc., que formam textos em geral.



- Uma variável do tipo lógico (ou booleano) armazena um conteúdo que assume apenas um de dois valores possíveis:
 - True (verdadeiro)
 - False (falso)
- Note que as iniciais T e F são escritas em maiúsculas.

O início e o fim de uma string são indicados por aspas ("), de modo a separar o conteúdo da string do restante do texto do programa.

Você pode usar espaços em uma string de caracteres.



Perguntas:

O que é uma variável?

É uma região de memória que contém um valor.

O que é um tipo de dado?

É o tipo de valor que uma variável pode ter. Ex. : Boolean, int, String, etc...

- Códigos devem ser escritos para serem lidos por seres humanos.
- Escreva os comentários no momento em que estiver escrevendo o código.
- Os comentários devem acrescentar informação, e não frasear o comando:

```
# Multiplicacao de b por h:
area = b * h

# Calculo da area do retangulo:
area = b * h
```

Sempre use nomes descritivos e fáceis de lembrar para suas variáveis:

Use sempre letras minúsculas em nomes de variáveis:

```
raio = 1.3
Raio = 4.6
RAIO = 7.9

raio_interno = 1.3
raio_meio = 4.6
raio_externo = 7.9
```

 Não utilize acentos nos nomes das variáveis. Pode funcionar em alguns sistemas, mas em outros, não.



42

- Use espaços em branco para melhorar a legibilidade.
- Utilize parênteses para melhorar a compreensão e evitar erros, mesmo que não alterem a precedência.

$$H = (A**2+B**2)**0.5$$
 $H = ((A ** 2) + (B ** 2)) ** 0.5$

Defina todas as variáveis que você vai utilizar no início de cada script, a fim de tornar mais fácil a manutenção do código.

```
nivel = 0.8  # nivel de combustivel (m)

altura = 2.3  # altura do tanque (m)

raio = 1.5  # raio da secao vertical (m)

volume = 0  # volume de combustivel (m3)
```

Activity

- Abra o arquivo 'VariableAssignmentActivity.java' no package sef.module3.activity
- Faça o seguinte:
 - Declare uma variável do tipo int e atribua à ela o valor padrão;
 - Atualize o valor;
 - Imprimir o valor atualizado no console



"Escopo"

 Refere-se a partes ou seções de um programa em que a variável tem valor e é considerada 'visible';

Tipos de variáveis por escopo:

Class Variables

Compartilhado por todas as instâncias de uma classe. Identificado pela palavra-chave **static**.

Instance Variables

Pertencer a uma instância de uma classe. São exclusivos para cada instância;

Local Variables

São acessíveis apenas dentro de sua localidade e geralmente declarados dentro de um método;

```
4
 5
   package org.businessapptester.monitoring.tcpserver.protocol;
 8= /**
                                                  class definition
     * My documentation.
 9
10
    @SuppressWarnings("unus
    public class MyClass {
12
                                                     instance variables
13
14
       private int intValue;
15
       private String stringValue;
16
17
        // method declaration
       public void doSomething(int intValue, String stringValue){
18E
           this.intValue = intVal
19
20
           this.stringValue = stringVal
21
           // do something with the values
22
23
                                                      instance method
24
```

 O escopo de uma variável informa onde ela pode ser utilizada

Exemplo:

```
1: class VerificaEscopo{
2: int escopoA;
3: public void metodo(int escopoB) {
4: int escopoC;
5: }
6:
7: }
```

Class Variable

É uma variável cujo valor é comum a todos os objetos membros da classe.

Local Variable

Declarados dentro de métodos e / ou subrotinas. *aString* é local no metodo *main*

Instance Variables

Possui um valor diferente para cada objeto instanciado.

```
package sef.module3.sample;
public class VariableScope {
   /**
    * @param args
  public static void main(String[] args) {
   String aString = "This is a local
  variable";
class Employee {
  public static int totalCount = 0;
  private String myFirstName;
  private String myLastName;
  private int myAge;
```

Atividade

O Código abaixo esta correto?

```
package unicesumar.ead.programacao;
public class Pessoa {
    private String nome;
    private int idade;
    private String falar() {
        String frase = "Olá";
        return frase;
    public void andar() {
        System.out.println(nome + frase);
```

Shadowing

Variáveis globais ficam ocultas em trechos do Código que são escopo de outras variáveis com o mesmo nome;

```
public class Parede {
       private String cor;
       public Parede() {
               this.cor = "branca";
       public void colorir(String cor) {
               cor = "verde";
                  O que acontece com o atributo de classe, ao
                      executar o método colorir?
```

"Type Casting"

Type Casting

- Refere-se à conversão do tipo de dados de um valor variável para outro tipo de dados;
- Java é fortemente tipado, o que significa que espera que os valores atribuídos um ao outro sejam compatíveis. A transmissão é uma maneira de expressar essa compatibilidade;

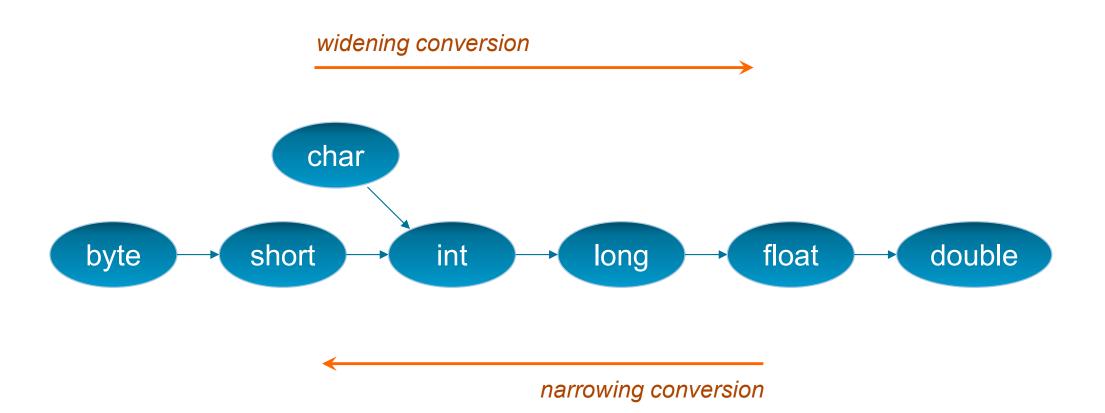
Implicit Casting

Isso é feito automaticamente pelo compilador para garantir que o programa seja executado corretamente

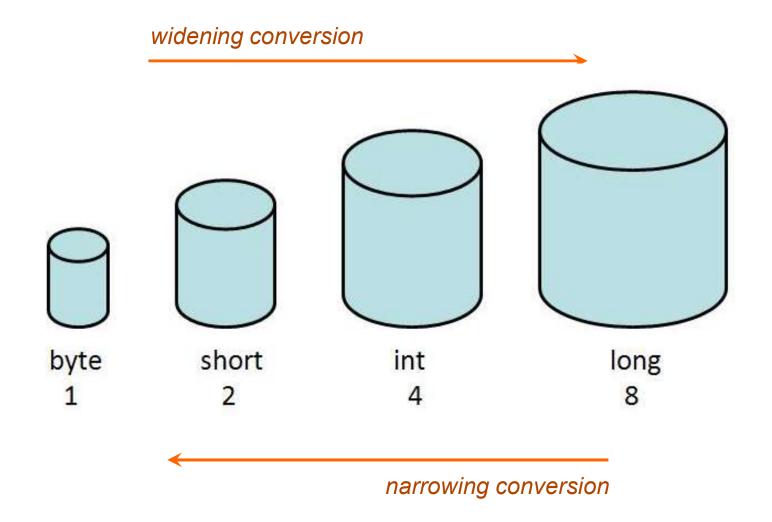
Explicit Casting

Isso é feito
quando o
programa
deseja modificar
o tipo de dados
de uma
expressão

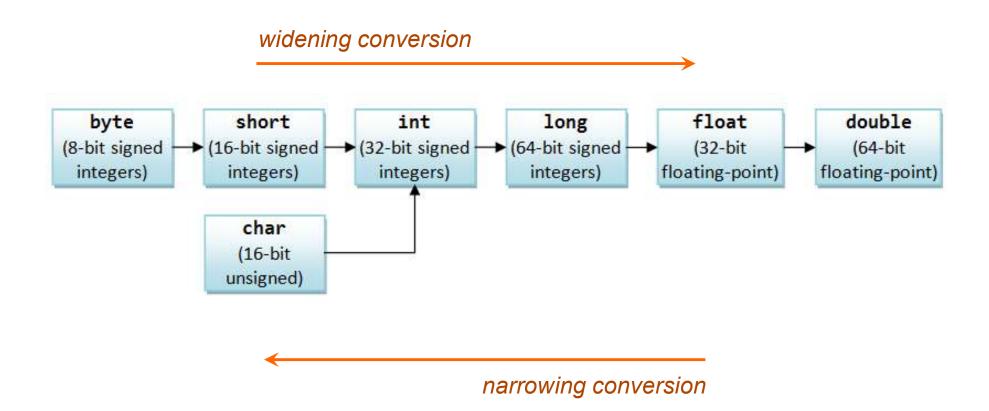
Primitive Data Type Casting Flow



Primitive Data Type Casting Flow



Primitive Data Type Casting Flow



Implicit Casting

- É uma operação implicita;
- Geralmente ocorre ao transmitir de um tipo de dados mais estreito para um tipo de dados mais amplo, isso também pode ser chamado de conversão de ampliação;

```
Example A: int a = 1;
```

double b = a;

Example B: float x = 1.0;

double y = x;

 Como um double é "mais amplo" do que um número inteiro ou um ponto flutuante, ele pode aceitar e converter seus valores implicitamente;

Explicit Casting

- Ocorre ao transmitir de um tipo de dados mais amplo para um tipo de dados mais estreito.
- Isso também pode ser chamado de conversão de restrição;

```
Syntax: <destination variable> = (<destination data type>) <source variable>
```

Example A: double a = 1.5;

int b;

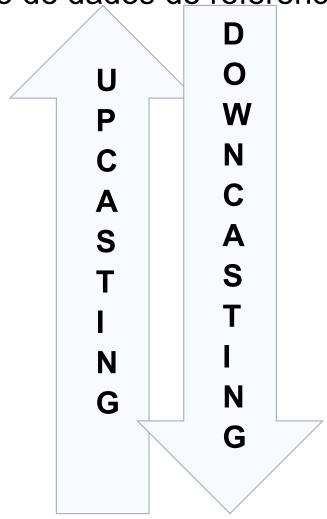
b = (int)a

Explicit Casting

- Como um número inteiro é "mais estreito" do que o double que está sendo atribuído, é necessário afirmar explicitamente que a atribuição é intencional;
- Às vezes, uma conversão restritiva perde parte do valor original durante a conversão. Por exemplo, no Exemplo A, a variável 'b' receberá o valor 1.

Reference Casting

 Refere-se à conversão de um tipo de dados de referência (um objeto) para outro tipo de dados de referência;



"Operadores"

2f4869676820706572666f72bd61be63852e2044556c6976657265642e2f4869676820706572666f72bd61be63852e2044556c6976657265642e2f4869676820706572666f72bd61be63852e2044556c6976657265642e2f4869676820706572666f72bd61be63852e2044556c6976657265642e2f4869676820706572666f72bd61be63852e2044556c6976657265642e2f4869676820706572666f72bd61be63852e2044556c6976657265642e2f4869676820706572666f72bd61be63852e2044556c6976657265642e2f4869676820706572666f72bd61be63852e2044556c6976657265642e2f4869676820706572666f72bd61be63852e2044556c6976657265642e2f4869676820706572666f72bd61be63852e2044556c6976657265642e2f4869676820706572666f72bd61be63852e2044556c69766572666f72bd61be63852e2044556c69766572666f72bd61be63852e2044556c69766572bd61be63852e2044556c69766572bd61be63852e2044556c69766572bd61be63852e2044556c69766572bd61be63852e2044556c69766572bd61be63852e20445666976bd7bd61be63852e204456666976bd7bd61be63852e204456666976bd7bd61be63852e204456666976bd7bd61be63852e20445666976bd7bd61be63852e20445666976bd7bd61be63852e20445666976bd7bd61be63852e20446566976bd7bd61be63852e20446566976bd7bd61be63852e20446566976bd7bd61be63852e20446566976bd7bd61be63852e20446566976bd7bd61be63852e20446566976bd7bd61be63852e20446566976bd7bd61be63852e20446666976bd7bd61be63852e20446566976bd7bd61be63852e20446566976bd7bd61be63852e20446566976bd7bd61be6

 Operadores aritméticos são usados para executar operações matemáticas em valores numéricos;

Operador	Descrição da operação matemática
+	Soma (inteira e ponto flutuante)
*	Subtração (inteira e ponto flutuante)
*	Multiplicação (inteira e ponto flutuante)
1	Divisão (inteira e ponto flutuante)
	Decremento unário (inteiro e ponto flutuante)
++	Incremento unário (inteiro e ponto flutuante)
%	Resto da divisão de inteiros

Examples: 1 + 1 A * (Z + 1)

Operadores unários:

prefix

- Y = ++x; Adiciona 1 a x antes de atribui a y
- *Y = --x*; Subtrai 1 de x antes de atribuir a y

– postfix:

- Y = x++; Atribui x à y antes de somar 1
- Y = x--; Atribui x à y antes subtrair 1

Valor inicial de x	Expressão	Valor final de y	Valor final de x
5	y = x++;	5	6
5	y = ++x;	6	6
5	y = x;	5	4
5	y =x;	4	4

Example

```
int x = 5;
x++; // x is 6
++x; // x is 7
```

- Operadores relacionais são usados para realizar comparações;
- Eles sempre avaliam como verdadeiro ou falso (expressão booleana);

Descrição	Operador	
igual a	== (dois sinais de igual)	
maior que	>	
menor que	<	
maior ou igual a	>=	
menor ou igual a	<=	
diferente de	!=	

Example:

boolean test = x < y;

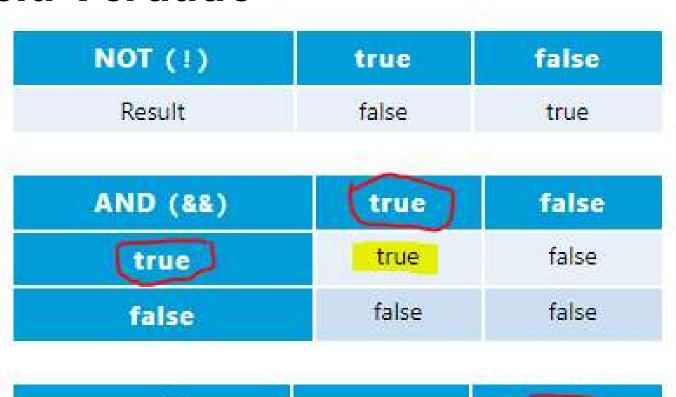
 Operadores lógicos são usados para avaliar expressões booleanas;

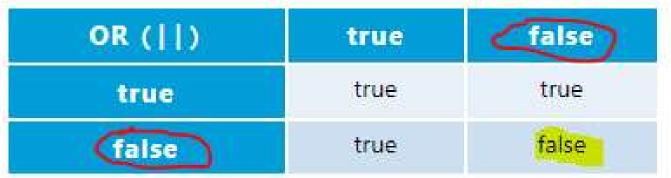
Descrição	Operador
E	&&
OU	(duas barras verticais)
NÃO	! (exclamação)

Example:

boolean test = x >= 1 & x <= 10

Tabela Verdade





• This is a hidden slide.

NOTE: This is a hidden slide. Please refer to the additional content in the notes section.

Activity

- Abra o arquivo 'OperatorActivity.java' no pacote sef.module3.activity.
- Faça o seguinte:
 - Encontre a diferença entre os números informados.
 - Imprima o resultado





Perguntas:

O que é escopo?

O escopo de uma variável, é a parte do programa em que ela pode ser referenciada.

O que é type casting?

Typecast é usado para converter um tipo de dado em outro tipo.

Questions and Comments

 What questions or comments do you have?



