



# Fundamentos



# Agenda

- Tipos Primitivos
- Palavras-Chave e Palavras Reservadas
- Conversão entre Tipos
- Constantes
- Operadores
- Exceções



# Tipos Primitivos

- Fortemente tipada
- Oito (8) tipos primitivos
  - ♦ Seis (6) tipos primitivos numéricos
    - ♦ Quatro (4) tipos primitivos inteiros
    - ♦ Dois (2) tipos primitivos de ponto flutuante
  - ♦ Um (1) tipo primitivo caracter
  - ♦ Um (1) tipo primitivo booleano



# Tipos Primitivos Numéricos

## Tipos primitivos Inteiros

<b>Tipo</b>	<b>Tamanho</b>	<b>Val. mínimo</b>	<b>Val. máximo</b>
byte	1 byte	-128	127
short	2 bytes	-32.768	32.767
int	4 bytes	-2.147.483.648	2.147.483.647
long	8 bytes	-9.223.372.036.854.775.808	9.223.372.036.854.775.807



# Literais Inteiros

- Podem ser expressos em formato decimal, octal ou hexadecimal
- O formato padrão é decimal

Ex.: **28**

- Para octal, o literal deve ser precedido por 0 (zero)

Ex.: **034**

- Para hexadecimal, o literal deve ser precedido por 0x ou 0X

Ex.: **0x1c** , **0x1C** , **0X1c** , **0X1C**

- Para binário, o literal deve ser precedido de 0b

Ex.: **0b0111\_0001** , **0b010101**



# Tipos Primitivos Numéricos

## Tipos Primitivos de Ponto Flutuante

Tipo Tamanho		Intervalo
float	4 bytes	Aproximadamente +/- 3.40282347E+38F (6-7 dígitos significativos)
double	8 bytes	Aproximadamente +/- 1.79769313486231570E+308 (15 dígitos significativos)



# Literais Ponto-Flutuante

- Para que um literal numérico seja interpretado como um valor em ponto-flutuante, este deve encaixar-se em um dos seguintes casos:

- Conter um ponto decimal

Ex.: **1.414**

- Conter a letra **E**, indicando notação científica

Ex.: **4.23E+21**

- Conter o sufixo **F** ou **f**, indicando um literal float

Ex.: **1.828f**

- Conter o sufixo **D** ou **d**, indicando um literal double

Ex.: **1234d**

- Um literal ponto-flutuante sem prefixo **F** ou **D** é interpretado como um literal double



# Conversões

As conversões de tipos podem ser divididas em 2 categorias:

- ♦ Conversões Implícitas (Conversão)
- ♦ Conversões Explícitas (Cast ou Coerção)

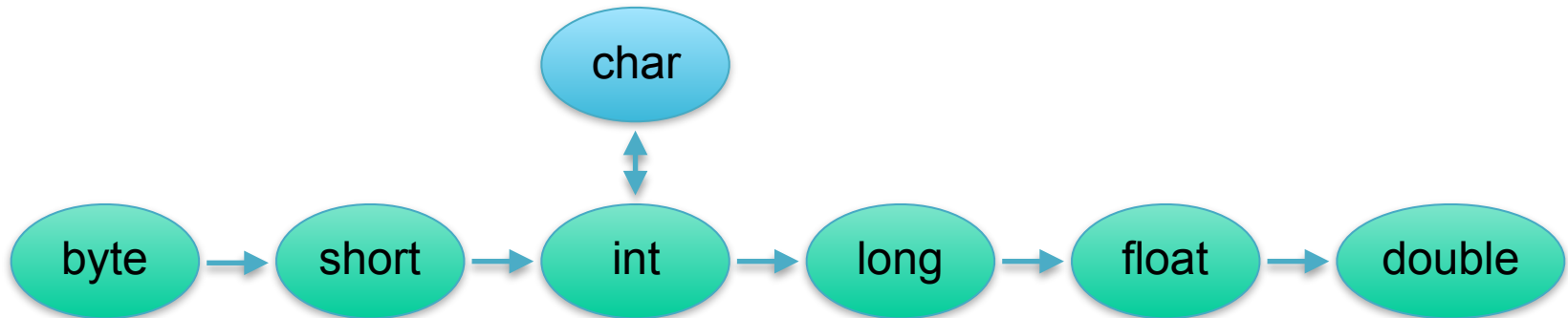




# Conversões

As regras gerais para conversão em atribuição de primitivas são:

- Um boolean não pode ser convertido para qualquer outro tipo.
- Um tipo não booleano pode ser convertido para outro tipo não booleano conforme o gráfico a seguir:





# Conversões

Tipo a ser convertido	Converter em	Forma de conversão
<code>int x = 10;</code>	float	<code>float y = (float)x;</code>
<code>int x = 10;</code>	double	<code>double y = (double)x;</code>
<code>float x = 10.5f;</code>	int	<code>int y = (int)x</code>
<code>String x = "10";</code>	int	<code>int y = Integer.parseInt(x);</code>
<code>String x = "20.54";</code>	float	<code>float y = Float.parseFloat(x);</code>
<code>String x = "20.54";</code>	double	<code>double y = Double.parseDouble(x);</code>
<code>String x = "Java";</code>	Array de bytes	<code>byte[] y = x.getBytes();</code>
<code>int x = 10;</code>	String	<code>String y = String.valueOf(x);</code>
<code>float x = 10.5f;</code>	String	<code>String y = String.valueOf(x);</code>
<code>double x = 10.45;</code>	String	<code>String y = String.valueOf(x);</code>
<code>byte[] x = {'J', 'a', 'v', 'a'};</code>	String	<code>String y = new String(x);</code>



# Wrappers

Tipo Primitivo

Classe Wrapper

boolean

java.lang.Boolean

char

java.lang.Character

byte

java.lang.Byte

short

java.lang.Short

int

java.lang.Integer

long

java.lang.Long

float

java.lang.Float

double

java.lang.Double



# Exceções

## try, catch e finally

- O termo **try** é utilizado para demarcar um bloco de código que pode gerar algum tipo de exceção
- O termo **catch** oferece um caminho alternativo a ser percorrido no case de ocorrer efetivamente uma exceção
- O termo **finally** delimita um bloco de código que será executado em quaisquer circunstâncias (ocorrendo ou não uma exceção)



# Exceções

## throw e throws

- A instrução **throw** é utilizada para gerar intencionalmente uma exceção
- O termo **throws** é utilizado para declarar um método que pode gerar uma exceção com a qual não consegue lidar



# Exceções

A sintaxe geral de tratamento de exceções

```
try {  
    // bloco  
} catch(Exception1 var) {  
    // bloco  
    throw new Exception();  
}  
// ...  
catch (Exception2 var) {  
    // bloco  
    throw new Exception();  
} finally {  
    // bloco  
}
```



# Exceções

```
public static void main(String[] args) {  
    int num = 0;  
    while (num == 0) {  
        try {  
            String temp = JOptionPane.showInputDialog("Informe um N°");  
            num = Integer.parseInt(temp);  
            System.out.println(num);  
        } catch (NumberFormatException ex) {  
            JOptionPane.showMessageDialog(null, "O n° informado é inválido!");  
        }  
    }  
    System.out.println("Fim");  
}
```

ao lançar  
a exceção,  
é executado  
o catch

