

# Programação I

# Classe Scanner

Prof. Me. Fábio Perfetto

## Classe Scanner

Uma das funções mais utilizadas em um programa é a capacidade de entrada de dados, ou seja, do usuário interagir com o programa inserindo informações no momento em que isto lhe é requisitado.

A forma que usaremos para implementar esta funcionalidade é através da classe Scanner.

## Declaração da classe Scanner

A declaração da classe scanner é até bem simples :

```
Scanner entrada = new Scanner(System.in);
```

Entrada é o nome da variável

New Scanner significa que entrada é do tipo Scanner e herda todos os seus atributos.

System.in indica que este é um objeto de entrada de dados

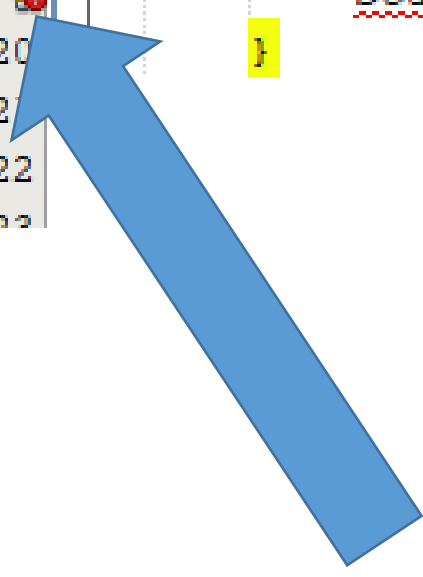
## Biblioteca

Quando fazemos a declaração da classe Scanner o compilador do Java percebe que esta classe não pertence ao `Java.lang`, que é a biblioteca básica do java.

É necessário fazer a importação da biblioteca `java.util` para a classe Scanner

O NetBeans pode fazer isto por você, quando você faz a declaração da classe Scanner ele indica, do lado esquerdo com uma lâmpada que a uma ação a ser realizada, que é a importação da biblioteca

```
12 public class JavaApplication17 {  
13  
14     /**  
15      * @param args the command line arguments  
16      */  
17     public static void main(String[] args) {  
18         // TODO code application logic here  
19         Scanner entrada = new Scanner(System.in)  
20     }  
21  
22  
23
```



Aqui o NetBeans faz a importação da biblioteca para você.

```
14  /**
15   * @param args the command line arguments
16   */
17  public static void main(String[] args) {
18      // TODO code application logic here
19      Scanner entrada = new Scanner(System.in)
```

- Adicionar importação para java.util.Scanner
- Adicionar importação para jdk.nashorn.internal.parser.Scanner
- Adicionar importação para com.sun.java\_cup.internal.runtime.Scanner
- Criar class "Scanner" com o construtor "Scanner(java.io.InputStream)" no pacote javaapplication17 (Pacotes de Códigos-fonte)
- Criar class "Scanner" no pacote javaapplication17 (Pacotes de Códigos-fonte)
- Criar class "Scanner" em javaapplication17.JavaApplication17
- Criar class "Scanner" em javaapplication17.JavaApplication17

Linha de comando de importação da biblioteca que foi adicionada

```
5  L  */
6  package javaapplication17;
7
8  [- import java.util.Scanner;
9
10 [- /**
11    *
12    * @author PERFETTO
13    */
14  public class JavaApplication17 {
15
16  [-    /**
17        * @param args the command line arguments
18        */
19  [-    public static void main(String[] args) {
20        // TODO code application logic here
21        Scanner entrada = new Scanner(System.in)
22    }
23
24 }
```



## Parâmetros de leitura

A leitura pode ser feita para uma string, neste caso utilizamos `entrada.nextLine()`

```
7
8  import java.util.Scanner;
9
10 /**
11  *
12  * @author PERFETTO
13  */
14 public class JavaApplication17 {
15
16     /**
17     * @param args the command line arguments
18     */
19     public static void main(String[] args) {
20         // TODO code application logic here
21         String nome;
22         Scanner entrada = new Scanner(System.in);
23         System.out.println("Digite seu nome ");
24         nome = entrada.nextLine();
25     }
26
27 }
28
```

A leitura pode ser para um valor inteiro, neste caso utilizamos `entrada.nextInt()`

```
8  import java.util.Scanner;
9
10 /**
11  *
12  * @author PERFETTO
13  */
14 public class JavaApplication17 {
15
16     /**
17     * @param args the command line arguments
18     */
19     public static void main(String[] args) {
20         // TODO code application logic here
21         int nome;
22         Scanner entrada = new Scanner(System.in);
23         System.out.println("Digite seu nome ");
24         nome = entrada.nextInt();
25
26     }
27
```

Ou a leitura pode ser ainda para um numero real, neste caso utilizamos `entrada.nextFloat()`;

```
1
8  import java.util.Scanner;
9
10 /**
11  *
12  * @author PERFETTO
13  */
14 public class JavaApplication17 {
15
16     /**
17     * @param args the command line arguments
18     */
19     public static void main(String[] args) {
20         // TODO code application logic here
21         float nome;
22         Scanner entrada = new Scanner(System.in);
23         System.out.println("Digite seu nome ");
24         nome = entrada.nextFloat();
25
26
```

# Tabela com todos os parâmetros de leitura

Lendo um valor inteiro:

```
int n;
```

```
System.out.printf("Informe um número para a tabuada: ");  
n = ler.nextInt();
```

Lendo um valor real:

```
float preco;
```

```
System.out.printf("Informe o preço da mercadoria = R$ ");  
preco = ler.nextFloat();
```

Lendo um valor real:

```
double salario;
```

```
System.out.printf("Informe o salário do Funcionário = R$ ");  
salario = ler.nextDouble();
```

Lendo uma String, usado na leitura de palavras simples que não usam o caractere de espaço (ou barra de espaço):

```
String s;
```

```
System.out.printf("Informe uma palavra simples:\n");  
s = ler.next();
```

Lendo uma String, usado na leitura de palavras compostas, por exemplo, Pato Branco:

```
String s;
```

```
System.out.printf("Informe uma cadeia de caracteres:\n");  
s = ler.nextLine();
```

# Cuidado

Na leitura consecutiva de valores numéricos e String deve-se esvaziar o buffer do teclado antes da leitura do valor String, por exemplo:

```
int n;
```

```
String s;
```

```
System.out.printf("Informe um Número Inteiro: ");
```

```
n = ler.nextInt();
```

```
ler.nextLine(); // esvazia o buffer do teclado
```

```
System.out.printf("Informe uma cadeia de caracteres:\n");
```

```
s = ler.nextLine();
```

# Primeiro Programa

```
public static void main(String[] args) {  
    System.out.println("Neste programa vamos fazer a inclusão de alguns valores utilizando");  
    System.out.println("a classe Scanner e faremos ainda a conversão implícita de tipos");  
    System.out.println("e a conversão explícita utilizando métodos da classe Integer");  
    int inteiro;  
    float real;  
    String letra;  
    Scanner ler = new Scanner(System.in);  
    /*System.out.println("Digite um valor do tipo real");  
    real = ler.nextFloat();  
    System.out.println("Agora vamos atribuir o valor real para a variável do tipo inteiro");  
    inteiro = real;  
    System.out.println("o valor real é: " + real);  
    System.out.println("o valor inteiro ficou assim: " + inteiro);*/  
    System.out.println("Digite um valor inteiro");  
    inteiro = ler.nextInt();  
    System.out.println("Vamos agora atribuir este valor inteiro para uma variável real");  
    real = inteiro;  
    System.out.println("O valor real agora é : " + real);  
    System.out.println("Agora digite um valor que será atribuído a uma variável do tipo character");  
    letra = ler.next();  
    System.out.println("vamos agora atribuir este valor convertendo ele para inteiro");  
    inteiro = Integer.parseInt(letra);  
    System.out.println("o valor do inteiro agora é : " + inteiro);  
}
```

## Exemplo esvaziando o buffer - erro

```
public static void main(String[] args) {  
    Scanner ler = new Scanner(System.in)  
    // instanciando e criando um objeto Scanner  
    int idade;  
    String nome;  
    System.out.println("Informe a sua idade:\n");  
    idade = ler.nextInt(); // entrada de dados (lendo um valor inteiro)  
    ler.nextLine(); // esvazia o buffer do teclado  
    System.out.println("Informe o seu nome:");  
    nome = ler.nextLine(); // entrada de dados (lendo uma String)  
    System.out.println("\nResultado:\n");  
    System.out.println( nome + " tem " + idade + " anos."); }  
  
}
```

