01

Tipo inteiro: int

Transcrição

Trabalharemos com sintaxes de variáveis e controles de fluxo - laços e condicionais - pela criação de um novo projeto acessando-se "New > Java Project". Poderíamos fazer tudo isto no mesmo arquivo, mas o intuito aqui é de treinar a codar e perder o medo das janelas e suas diversas opções.

Criaremos o "sintaxe-variaveis-e-fluxo", os dois tópicos que começaremos a ver. O novo projeto contendo o diretório "src" estará visível na *view* de "Package Explorer". No prompt, há um diretório "bin" escondido, pois o programa não quer mostrar o `.class`, e sim o código fonte Java. Reparem que no momento estou usando Mac, o que pouco importa, já que o Eclipse funciona da mesma maneira em todos os sistemas operacionais.

Criaremos nossa classe para começar a trabalhar com variáveis. Clicaremos com o lado direito do mouse em "src" e depois em "New > Class", e a classe se chamará "TestaVariaveis". No Java, um *statement* (ou instrução) não funciona fora dos métodos, portanto precisaremos do ponto inicial, do `public static void main(String[] args)`, após o qual salvaremos:

```
public class TestaVariaveis {

    public static void main(String[] args) {

    }
}
```

Poderíamos rodar a aplicação assim como está, mas não aconteceria nada. Então, digitaremos:

```
public class TestaVariaveis {

    public static void main(String[] args) {
        System.out.println("ola novo teste");
    }
}
```

Salvaremos novamente e rodaremos a aplicação indo à "Run > Run As > Java Application", ou clicando com o lado direito do mouse na classe com `main`, e em "Run As > Java Application". Também há o atalho "Ctrl + S". O Console mostrará o print, e com isto repetimos o mesmo teste do "ola mundo" feito anteriormente.

As palavras que aparecem em roxo no editor são as palavras chave, reservadas, e deverão estar sempre em caixa baixa. Agora, para criarmos uma variável denominada `idade`, que armazenará nossas idades, digitaremos:

```
public class TestaVariaveis {

    public static void main(String[] args) {
        System.out.println("ola novo teste");

        idade = 37;
```

```
    }
}
```

No Java, como o Eclipse já está dando a entender sublinhando `idade` com vermelho, não compila isto, pois trata-se de uma linguagem **estaticamente ou fortemente tipada**, ou seja, que necessita da declaração de todas as variáveis e tipos a serem utilizados. Passando o mouse sobre a palavra sublinhada, lê-se a mensagem de erro *"idade cannot be resolved to a variable"*.

Significa que " `idade` não pode ser entendida como uma variável", pois não foi declarada. O Eclipse inclusive dará algumas opções de "rápido conserto", ou *quick fix*, para a criação local da variável, ou remoção da linha, por exemplo. `idade = 37` é uma **atribuição**, em que `37` se encontra dentro de `idade` .

Precisaremos declará-la informando que ela é do tipo numérico e que guarda um valor inteiro, sem decimais ou pontos flutuantes. `int` vem de *Integer*:

```
public class TestaVariaveis {

    public static void main(String[] args) {
        System.out.println("ola novo teste");

        int idade;
        idade = 37;
    }
}
```

Salvaremos e rodaremos este código. Clicando-se na setinha ao lado do ícone verde que indica *play* na barra de ferramentas superior, vê-se os últimos programas que foram rodados no programa. E clicando no ícone verde, roda-se o último deles.

O valor foi guardado, mas parece que nada aconteceu de fato. Além de atribuirmos uma variável, pode-se usar o valor, mostrando-o na tela. Para isto, utilizaremos o `System.out.println` de novo, desta vez sem as aspas, pois queremos a *evaluation*, o resultado daquela expressão, e não uma cadeia de caracteres, uma *string*:

```
public class TestaVariaveis {

    public static void main(String[] args) {
        System.out.println("ola novo teste");

        int idade;
        idade = 37;

        System.out.println(idade);
    }
}
```

Inclusive, é possível ver que todas as menções à variável `idade` ficam em *highlight*, destacadas para mostrar que tratam-se da mesma variável. Vamos rodar o código acima para imprimirmos o valor de `idade` ! No "Console", obteremos:

```
ola novo teste
37
```

Poderemos trabalhar com os operadores aritméticos junto a estas variáveis, também:

```
idade = 30 + 10;
idade = 7 * 5 + 2;
```

Como na maioria das linguagens, no Java também há precedência, então as operações matemáticas seguem uma determinada ordem de prioridade, mas poderemos usar parênteses, desta forma:

```
idade = (7 * 5) + 2;
```

E assim por diante. Imprimiremos a idade três vezes:

```
int idade;
idade = 37;

System.out.println(idade);

idade = 30 + 10;

System.out.println(idade);

idade = (7 * 5) +  2;

System.out.println(idade);
```

E obteremos o resultado esperado, na aba "Console":

```
37
40
37
```

No código, usamos algumas convenções: ao criarmos a classe `TestaVariaveis`, cuja funcionalidade ainda desconhecemos, usamos a primeira letra em maiúscula e, ao acrescentarmos a segunda palavra, não utilizamos *underscore* ou algo do tipo, e sim a primeira letra em caixa alta de novo. Isto se chama ***Camel Case***, e aparece com frequência no Java e em muitas outras linguagens - é uma **convenção de código**, e seu uso não é obrigatório.

Da mesma forma, a variável iniciando-se com "i" minúsculo é o padrão, bem como não há o costume de se abreviar palavras. No Java, vocês verão nomes gigantescos de variáveis! É legal nos atentarmos a estas práticas para começarmos a nos acostumar com estes hábitos essenciais para quando formos trabalhar com grandes equipes.

Para mostrarmos uma frase antes de um número, basta imprimirmos uma *string*, como "a idade é", juntamente com a variável `idade`, assim:

```
System.out.println("a idade é " + idade);
```

O operador `+`, na maioria das vezes, tem a função de somar variáveis de tipo numérico, sendo a única exceção estes casos em que acompanham *strings*, com os números sendo convertidos em letras e tudo sendo concatenado. Este

operador, portanto, também serve para concatenar algo com uma palavra ou frase (uma *string*).

Salvando e rodando a aplicação, teremos:

```
A idade é 37
```

Pode-se acrescentar mais *string*s após a variável usando-se o operador.

Há outra versão do `System.out.println()`, o `System.out.print()`, sem o **ln**, isto é, sem o *line*, que pula a linha, que poderá ser utilizado de acordo com sua preferência.