

1. INSTALAÇÃO DO SOFTWARE

Nas aulas práticas, será utilizado o software **IntelliJ IDEA Ultimate**, cuja licença é gratuita para os estudantes do Instituto Politécnico de Leiria.

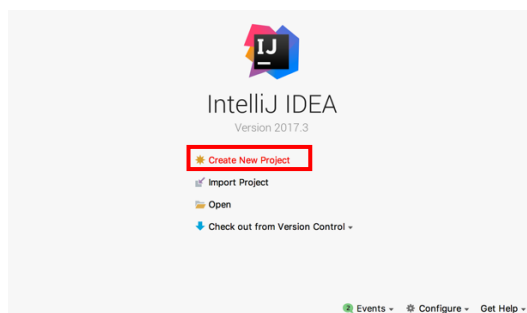
Como tal, deve seguir os passos descritos no documento “Passos para a instalação JDK + IntelliJ IDEA Ultimate”, disponível no Moodle, para a correta instalação e ativação.

2. PRIMEIROS PASSOS NO IntelliJ

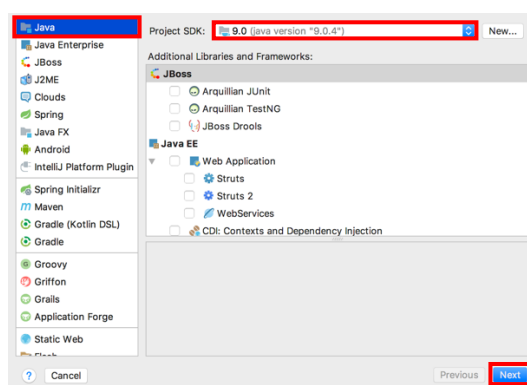
Neste exercício é explicado, passo a passo, como criar e executar um projeto.

2.1. CRIAÇÃO DE UM PROJETO

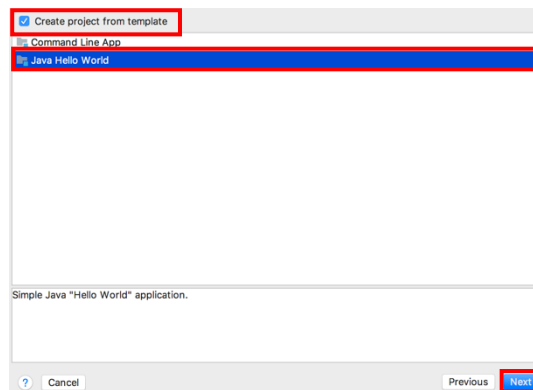
- a) Escolha a opção “Create New Project”



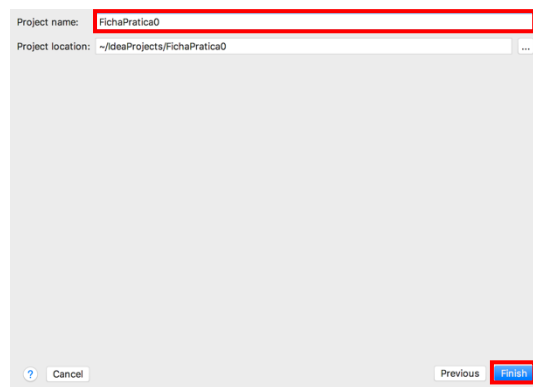
- b) Escolha a opção “Java”, confirme o Project SDK (9.0) e clique em Next



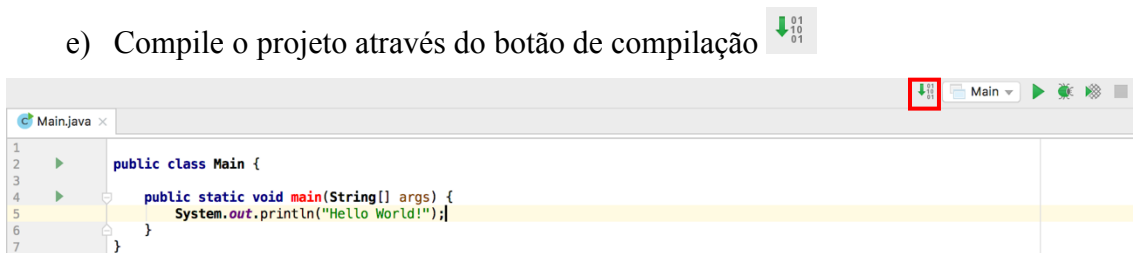
- c) Clique em **Create project from template**, selecione **Java Hello World** e de seguida clique em **Next**



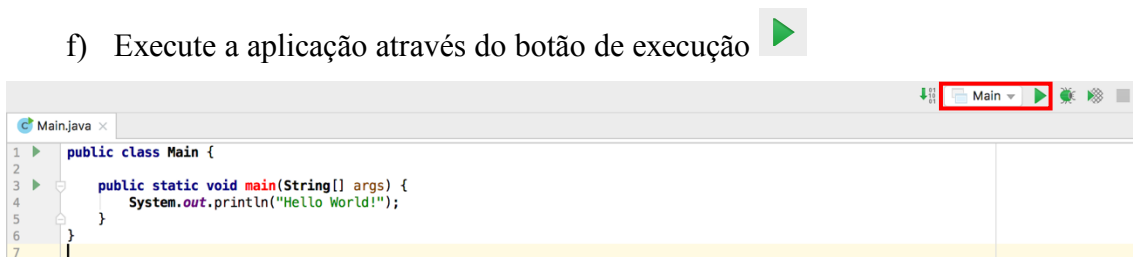
- d) Insira o nome do projeto (**FichaPratica0**), defina a localização do projeto e clique em **Finish**



- e) Compile o projeto através do botão de compilação



- f) Execute a aplicação através do botão de execução



- g) Visualize o resultado da execução na consola



3. CONCEITOS BÁSICOS DE JAVA

3.1. TIPOS DE DADOS

- Primitivos
 - Inteiros: int, long
 - Reais: double, float
 - Caracteres: char
 - Booleanos: boolean (true/false)
- Referências (classes e interfaces)

3.2. OPERADORES

- Unários

Operador	Função	Operador	Função
-	Inversão do sinal	!	Negação
++ (prefixo)	++i equivalente a $i = i + 1$	(sufixo) ++	i++ equivalente a $i = i + 1$ e retorno do valor $i - 1$
-- (prefixo)	--i equivalente a $i = i - 1$	(sufixo) --	i-- equivalente a $i = i - 1$ e retorno do valor $i + 1$

- Binários

Operador	Função	Operador	Função
&&	E lógico (lógica booleana)		Ou lógico (lógica booleana)
==	Igualdade	!=	Diferença
+	Soma	-	Subtração
*	Multiplicação	/	Divisão
>	Maior do que	<	Menor do que
>=	Maior ou igual do que	<=	Menor ou igual do que
+=	Soma e atribuição	-=	Subtração e atribuição
*=	Multiplicação e atribuição	/=	Divisão e atribuição
%	Resto da divisão inteira	%=	Resto da divisão inteira e atribuição

- Ternário

Operador	Função
Condição ? valor se verdade : valor se falso	Devolve o 1º valor, caso a condição se verifique, ou o 2º valor caso contrário

3.3. ESTRUTURAS DE CONTROLO

- Decisão

a) if ... else (alternativa)

```
if (expressão de valor boolean) {
    instruções
} else if (expressão de valor boolean) {
    instruções
} else {
    instruções
}
```

b) switch (seleção)

```
switch (variável) {
    case valor1:
        instruções
        break;
    case valor2:
        instruções
        break;
    default:
        instruções
}
```

Nota: Na estrutura switch, a variável tem que ser de um tipo que permita um conjunto finito de valores. Por exemplo, não é possível utilizar double. A classe String é a única exceção a esta regra, podendo também ser utilizada.

- Repetição

a) while

```
while (expressão de valor boolean) {
    instruções
}
```

b) do ... while

```
do {
    instruções
} while (expressão de valor boolean);
```

c) for

```
for (inicializações; expressão de valor boolean; pós-instruções) {
    instruções
}
```

- Saltos incondicionais

Salto	Aplicável em	Função
break	Estruturas de seleção e estruturas de repetição	Sair da respectiva estrutura de seleção ou repetição
continue	Estruturas de repetição	Seguir para a próxima iteração da estrutura de repetição

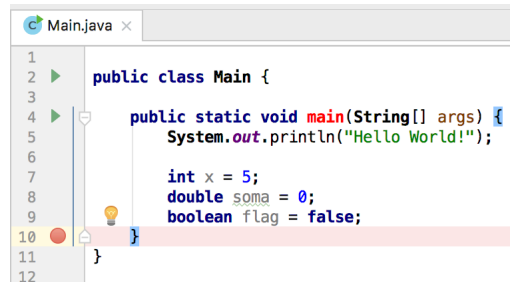
4. APLICAÇÃO DOS CONCEITOS BÁSICOS DE JAVA


4.1. VARIÁVEIS

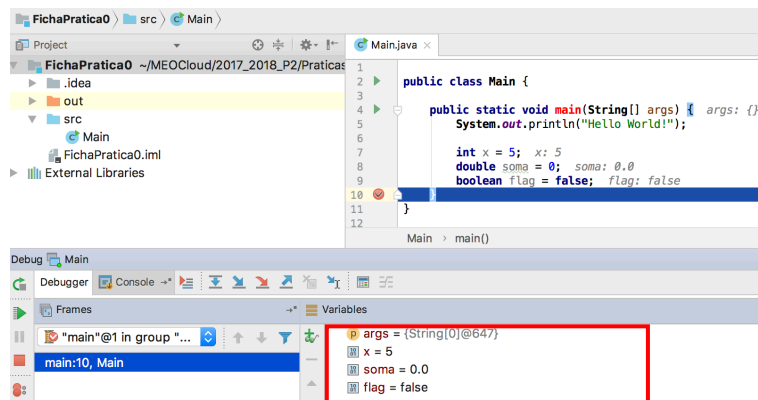
- Crie, no método **main**, uma variável do tipo int com o nome **x** e inicialize-a com o valor 5
- Crie, no método **main**, uma variável do tipo double com o nome **soma** e inicialize-a com o valor 0
- Crie, no método **main**, uma variável do tipo boolean com o nome **flag** e inicialize-a com o valor false

- **Depuração de código no IntelliJ**

1. Crie um *breakpoint* (ponto de paragem), na linha de término da função **main**, clicando no lado direito do número da linha




2. Execute a aplicação **Main** através do menu **Run**, opção **Debug 'Main'** ou do botão de depuração 
3. A execução do programa fica suspensa na linha assinalada a azul, e é possível verificar o valor que as variáveis **x**, **soma** e **flag** possuem no momento

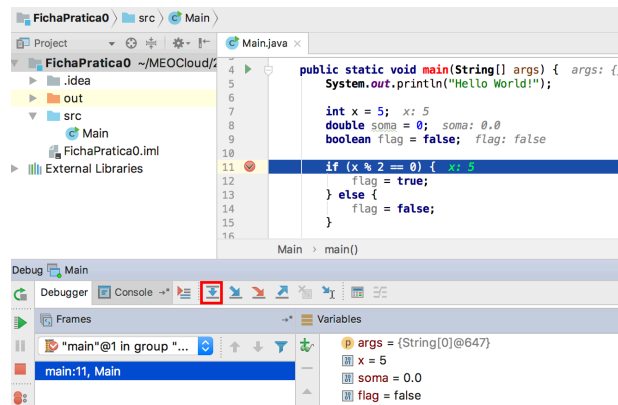



4.2. ESTRUTURAS

- a) Verifique, com recurso a uma estrutura de controlo alternativa **if**, se o valor de **x** é par. Caso seja verdade, modifique o valor da variável **flag** para true. Caso contrário, atribua à variável **flag** o valor false

- **Depuração de código no IntelliJ**


1. Remova o *breakpoint* existente
2. Crie um *breakpoint* na linha da estrutura de controlo alternativa if
3. Execute a aplicação **Main** através do menu **Run**, opção **Debug ‘Main’** ou do botão de depuração 
4. A execução do programa fica suspensa na linha assinalada a azul

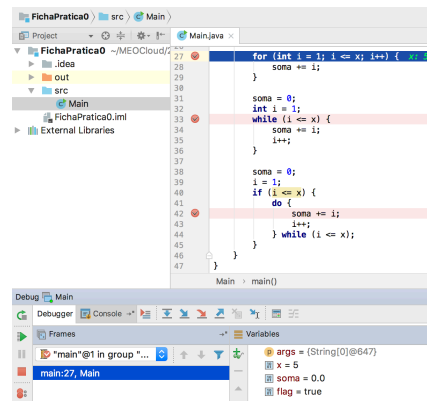


5. Clique no botão  para avançar a execução para a **próxima instrução** e suspendê-la novamente. Verifique que, durante a execução, o valor da variável **flag** mantém false, porque a condição do if é falsa


- b) Repita o exercício anterior, substituindo o if pela estrutura de seleção switch
- c) Repita o exercício anterior, substituindo o switch pelo operador ternário
- d) Repita o mesmo exercício, recorrendo apenas aos operadores de atribuição e de comparação
- e) Efetue a soma dos números inteiros compreendidos entre 1 e **x**, recorrendo à variável **soma** e à estrutura de repetição for
- f) Reinicie o valor da variável **soma** a 0
- g) Efetue a soma dos números inteiros compreendidos entre 1 e **x**, recorrendo à variável **soma** e à estrutura de repetição while
- h) Reinicie o valor da variável **soma** a 0
- i) Efetue a soma dos números inteiros compreendidos entre 1 e **x**, recorrendo à variável **soma** e à estrutura de repetição do ... while

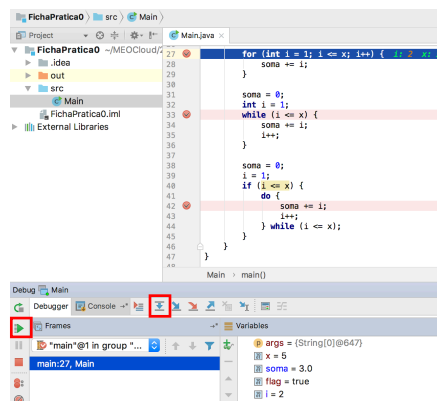
- **Depuração de código no IntelliJ**


1. Remova o *breakpoint* existente
2. Crie um *breakpoint*, na primeira linha de cada ciclo, clicando no lado direito do número da linha, de acordo com a figura abaixo
3. Execute a aplicação **Main** através do menu **Run**, opção **Debug 'Main'** ou do botão de depuração 

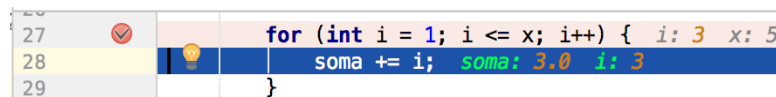


Nota: Os *breakpoints* não devem ser colocados em linhas sem código executável (ex.: linha 41), à exceção das linhas de término de funções (ex.: depuração do exercício 4.1).

4. Clique no botão  para avançar para o **próximo breakpoint**. Verifique que a execução se mantém na mesma linha, embora o valor das variáveis **i** e **soma** tenha sido atualizado



5. Clique no botão , sucessivamente, para avançar para a **próxima instrução** e analise o valor de cada variável

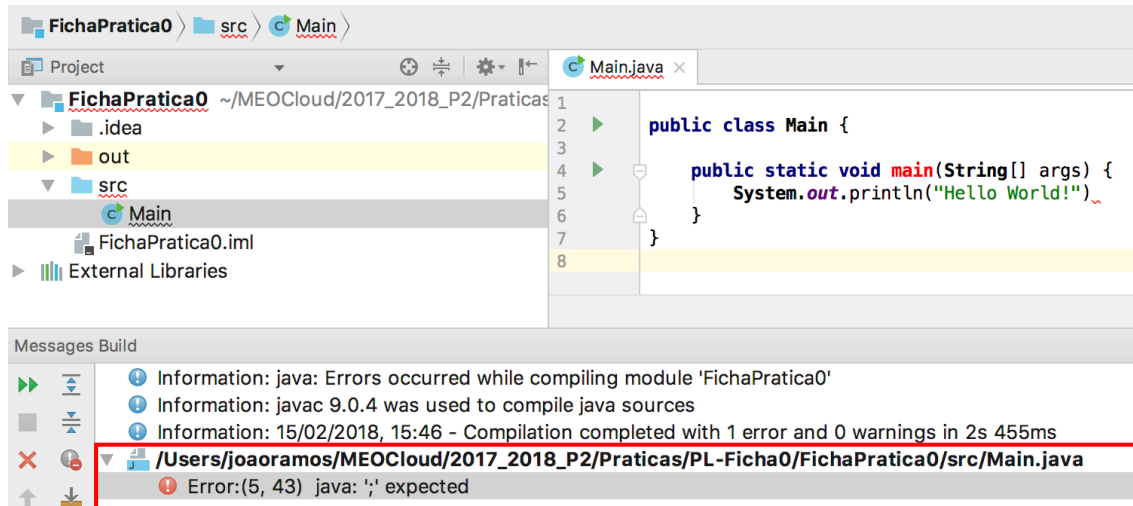


- j) Utilizando o salto incondicional **continue**, altere o ciclo **for** utilizado para somar os números inteiros compreendidos entre 1 e **x** de modo a que o valor 3 não seja somado

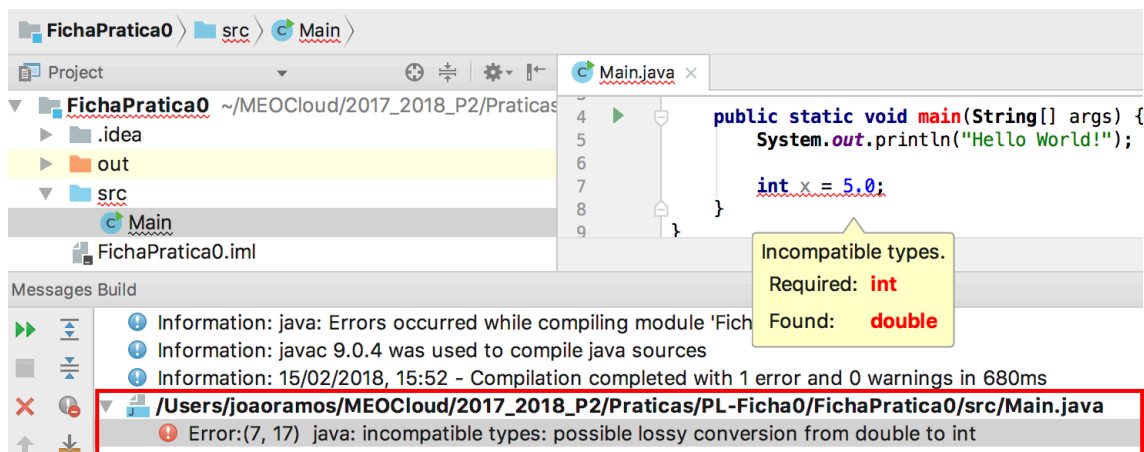
5. IDENTIFICAÇÃO DE ERROS

5.1. EXEMPLOS DE ERROS DE COMPILAÇÃO

- Remova o ';' da linha onde é impresso na consola "Hello World!"
- Tente compilar e verifique o erro de compilação (é esperado um ';' na linha 5, posição 43, da classe **Main**)



- Corrija o erro, adicionando um ';' no final da linha com o problema
- Atribua à variável **x** o valor 5.0
- Tente compilar e verifique o erro de compilação (tipos incompatíveis na linha 7, posição 17, da classe **Main**)



- Corrija o erro, atribuindo à variável **x** o valor 5

Considerando que o valor de **flag** é false, o valor da **soma** é 15 e o valor de **x** é 5, qual o novo valor de cada uma das variáveis após a execução da seguinte linha de código?

```
flag = (x = (int) (soma -= 0.5)) > 10;
```

flag	soma	x