Formabase
Aprenda passo a passo

Java
Eclipse

Metodologia

- Interprete o documento calmamente e com atenção.
- Acompanhe a execução do exercício no seu computador.
- Não hesite em consultar o formador para o esclarecimento de qualquer questão.
- Não prossiga para o ponto seguinte sem ter compreendido totalmente o ponto anterior.
- Caso seja necessário, execute várias vezes o exercício até ter compreendido totalmente o processo.

Conteúdo programático

1. Editor IDE Eclipse

# 1. Editor IDE Eclipse

Como já foi referido, ao longo deste curso iremos utilizar o **Eclipse** como ferramenta de desenvolvimento. O Eclipse é um dos melhores editores que traz já consigo um elevado conjunto de ferramentas destinadas a apoiar o desenvolvimento de programas em várias linguagens de programação, entre elas o Java.

O termo **IDE** significa ***Integrated Development Environment***. Uma das grandes vantagens de editores IDE em geral, e do Eclipse em particular, é o facto de não ser necessário utilizar a linha de comando para compilar ou correr programas em Java, uma vez que estas ferramentas estão já incluídas no editor como veremos nos exercícios seguintes. Com esta funcionalidade, o Eclipse alerta-nos para os erros no código enquanto estamos a programar, e não apenas quando compilamos ou executamos o programa.

Outra vantagem significativa na utilização de IDE's, é a facilidade de edição de código. Editores como o Eclipse ajudam o programador na utilização de funções ou métodos, através de funcionalidades de preenchimento automático de código (*auto-complete*) ou da abertura de janelas de ajuda, deixando de ser necessário aos programadores saber de antemão o nome de métodos ou funções que pretendem utilizar.

Por fim, mas não menos importante, é a possibilidade de efetuar depuração (*debug*) visual sobre o código com a utilização de pontos de paragem (*breakpoints*) e análise do conteúdo das variáveis enquanto o programa está a correr (*run-time*), como veremos em exercícios mais adiante.

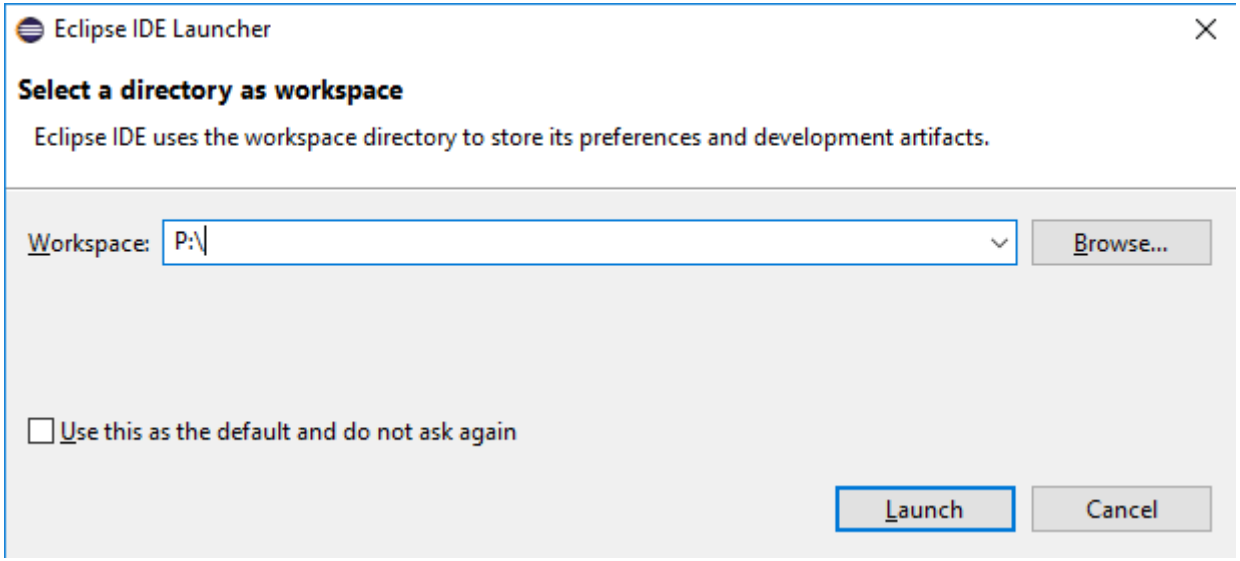
O Eclipse é um editor "orientado a projetos", i. e. considera que o código Java que desenvolvemos está contido num conjunto de classes, dispersas por vários ficheiros que compõem um projeto, e não em apenas num ficheiro onde se introduz todo o código, como se faz com outras linguagens de programação.

Para criar um projeto Java, execute os seguintes passos:

- **Clique** no **ícone do Eclipse que se encontra na barra de tarefas** 1


  - É-nos mostrada a janela "Workspace Launcher" que serve para podermos definir a pasta que o Eclipse irá usar para o nosso trabalho.

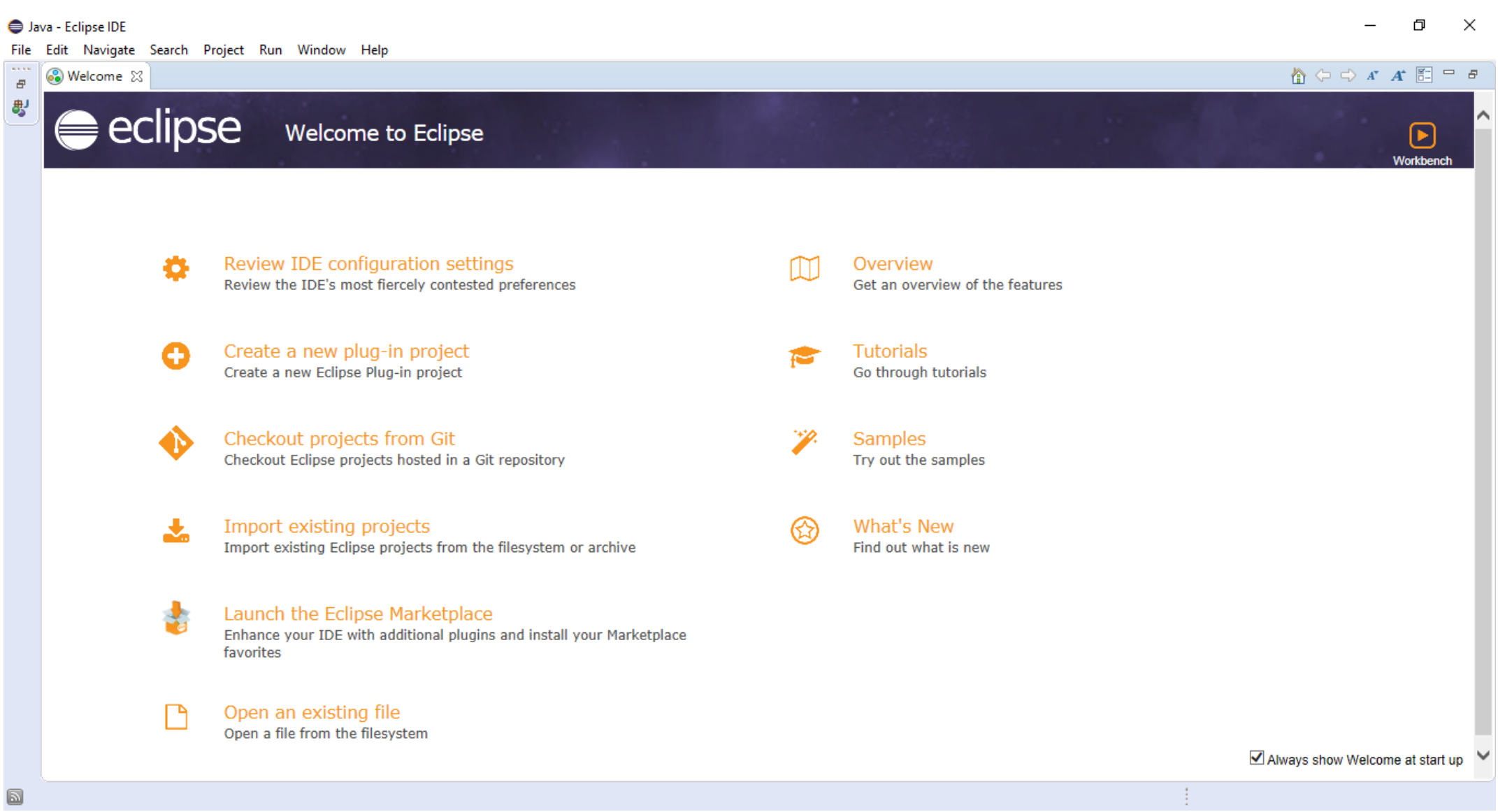
- **Escolha** a sua pasta de aluno **P:** e **prima Ok**




Caso seja a primeira vez que corre o programa, será mostrado o ecrã de boas vindas. Neste momento todas as definições de trabalho do Eclipse serão armazenadas na pasta referida anteriormente.

Se mais tarde necessitar de alterar a pasta de trabalho, pode fazê-lo através da opção **Switch workspace** que se encontra no menu **File**.

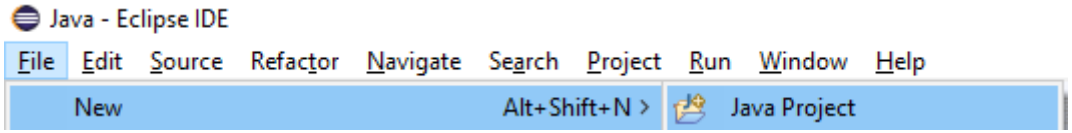
`Fechar ecrã de boas vindas`




- **Feche** o ecrã de boas vindas de forma a conseguir ver o seu espaço de trabalho

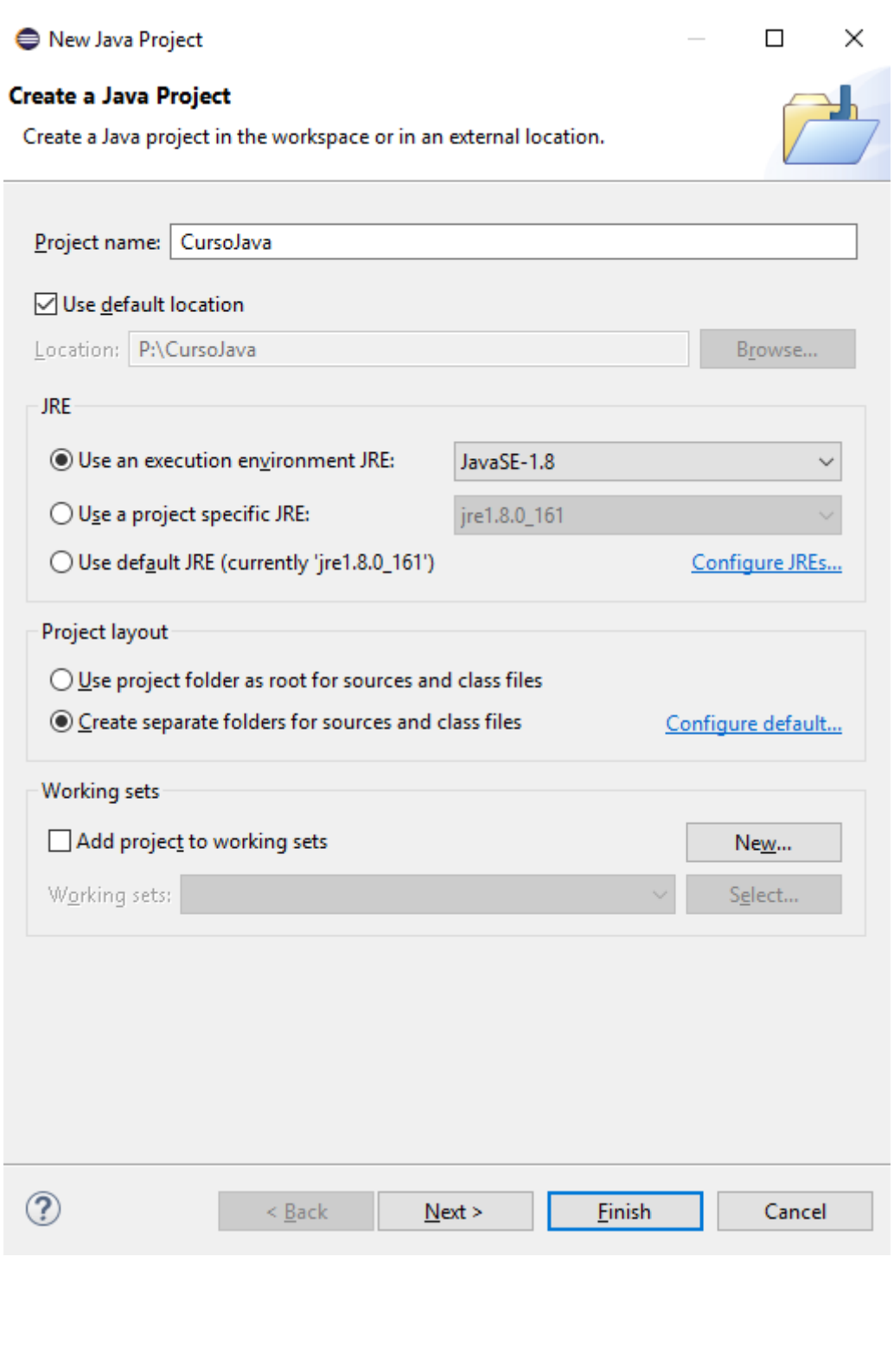
- **Clique** no menu **File**

- **Escolha** a opção **New > Java Project**



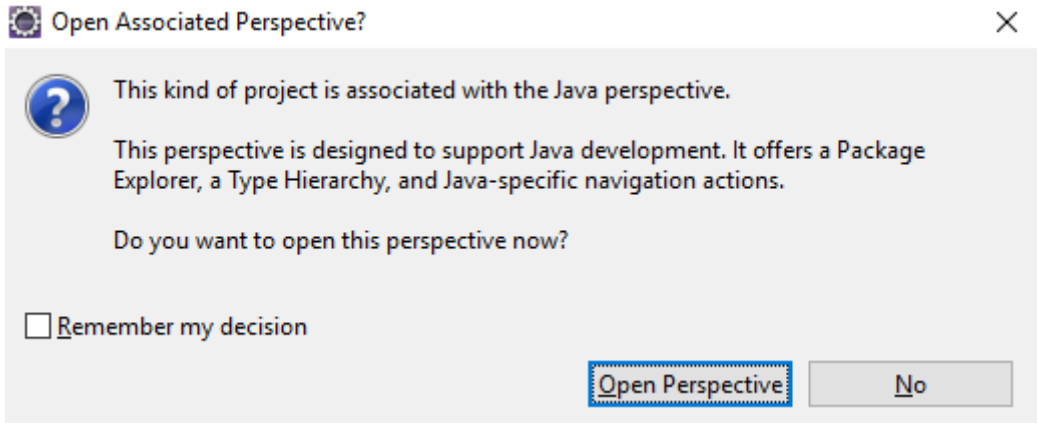

- **Introduza** `CursoJava` em **Project Name**

- **Escolha** as restantes opções de acordo com a imagem seguinte:


New Java Project
Create a Java Project
Create a Java project in the workspace or in an external location.
Project name: CursoJava
Use default location
Location: P:\CursoJava
Browse...
JRE
Use an execution environment JRE: JavaSE-1.8
Use a project specific JRE: jre1.8.0_161
Use default JRE (currently 'jre1.8.0_161')
Configure JREs...
Project layout
Use project folder as root for sources and class files
Create separate folders for sources and class files
Configure default...
Working sets
Add project to working sets
New...
Working sets:
Select...
< Back
Next >
Finish
Cancel

- Depois de introduzir os dados, **clique** no botão **Finish**

Caso seja o primeiro projeto de Java que cria no Eclipse, irá surgir uma janela que pergunta se pretendemos mudar para a perspetiva de Java. A mudança de perspetiva tem impacto nas janelas disponíveis, assim como no tamanho e arranjo das mesmas. Esta funcionalidade existe porque o Eclipse permite-nos programar em diversas linguagens cujas funcionalidades e janelas necessárias podem variar.

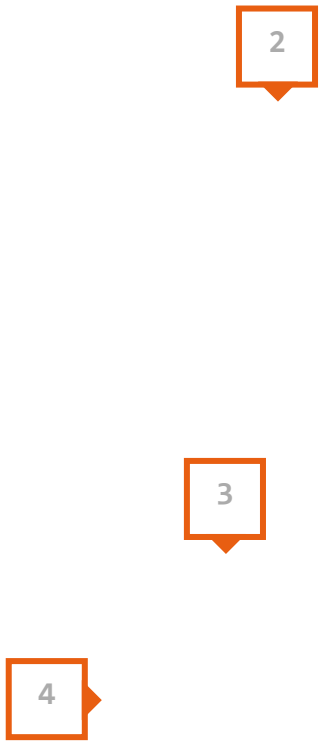


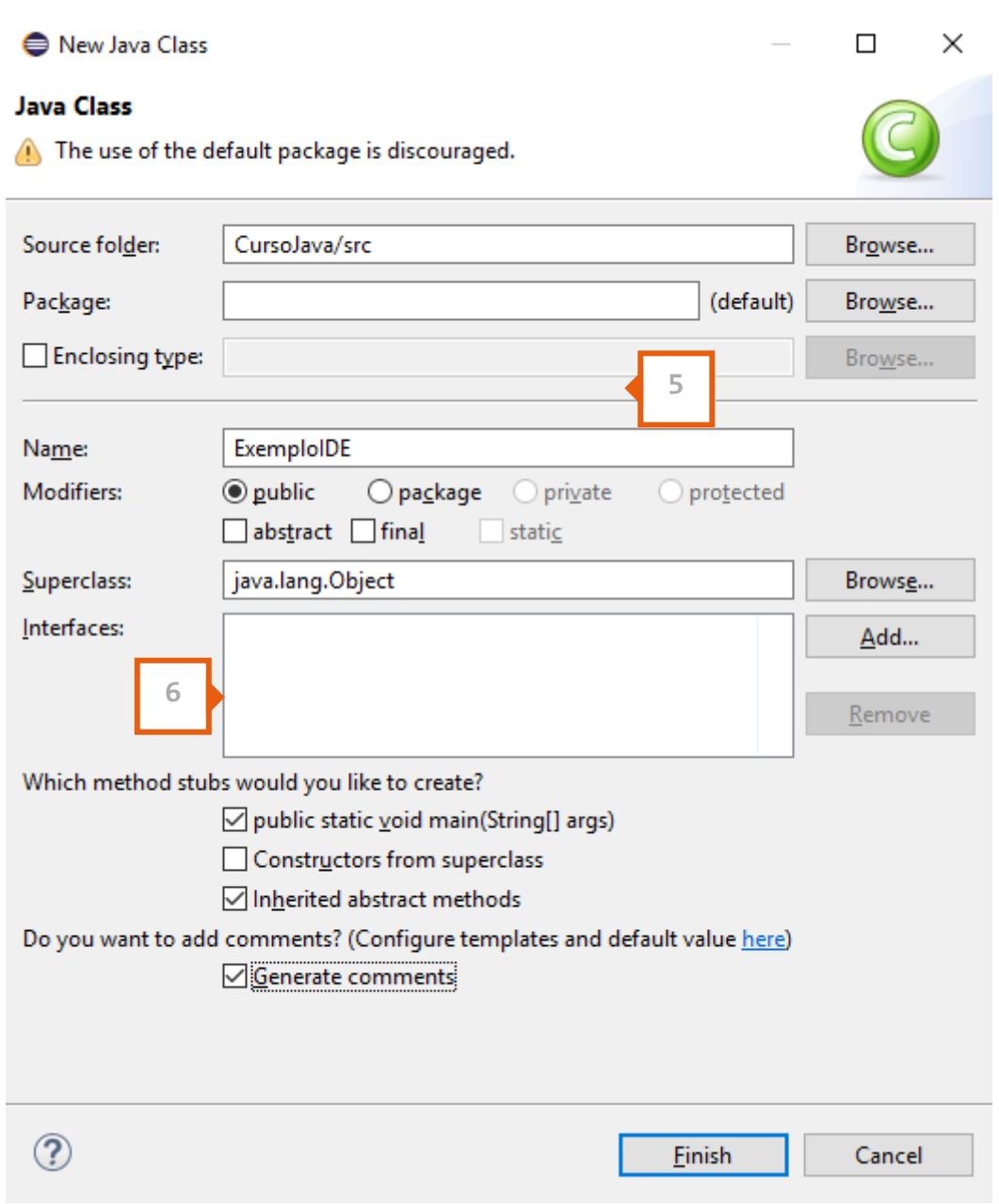

- **Clique** no botão **Yes**

- **Selecione** agora a opção **File > New > Class**

Nesta altura o Eclipse abre uma janela para definição da nova classe a adicionar ao projeto `CursoJava`. Esta janela tem todo um conjunto de definições de forma a automatizar as construções típicas de classes em Java. Este conceito vai ser aprofundado com o decorrer do curso.

- **Introduza** os seguintes **dados** para definição de uma nova classe:




New Java Class
Java Class
The use of the default package is discouraged.
Source folder:
CursoJava/src
Browse...
Package:
(default)
Browse...
Enclosing type:
Browse...
5
Name:
ExemploIDE
Modifiers:
public
package
private
protected
abstract
final
static
Superclass:
java.lang.Object
Browse...
Interfaces:
Add...
6
Remove
Which method stubs would you like to create?
public static void main(String[] args)
Constructors from superclass
Inherited abstract methods
Do you want to add comments? (Configure templates and default value here)
Generate comments
Finish
Cancel

Os pontos assinalados na figura tem as seguintes descrições:

- **[2]** Pasta onde guarda a classe
- **[3]** Nome da classe a criar
- **[4]** Para adicionar o termo "public" na declaração da classe
- **[5]** Indicação para a criação automática do método "main"
- **[6]-** Definição de geração automática de comentários

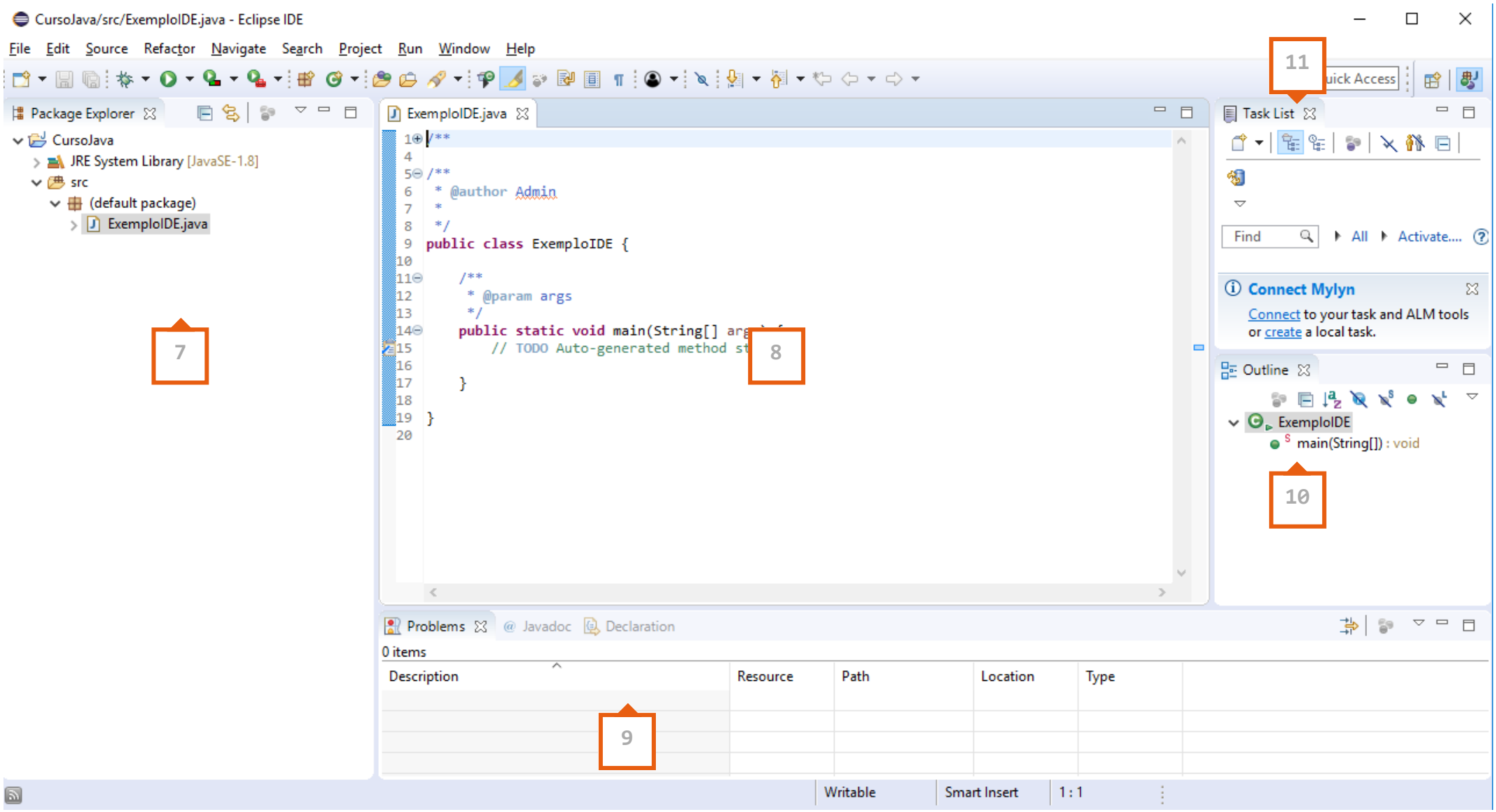
Após introduzirmos todos os dados necessários e clicarmos no botão **Finish**, o Eclipse gera automaticamente o código Java de uma nova classe. É a partir de agora, depois de termos criado já programas em exercícios anteriores através do **Notepad++**, que vamos valorizar a utilização de uma ferramenta tão avançada como esta que faz uma grande parte do trabalho por nós.

Nesta fase o Eclipse apresenta o código gerado automaticamente, da classe `ExemploIDE.java` na secção central do Eclipse, que podemos editar.

› **Analise** agora o conteúdo da pasta **P:**

A pasta .metadata contém todas as definições do ambiente de trabalho do Eclipse.

A imagem seguinte descreve todo o ambiente de trabalho do Eclipse. **Analise-a** com atenção:


CursoJava/src/ExemploIDE.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
CursoJava
JRE System Library [JavaSE-1.8]
src
(default package)
ExemploIDE.java
7
ExemploIDE.java
1/**
4
5/**
6 * @author Admin
7 *
8 */
9public class ExemploIDE {
10
11 /**
12 * @param args
13 */
14 public static void main(String[] arg
15 // TODO Auto-generated method st
16
17 }
18
19}
20
8
Task List
Find All Activate....
Connect Mylyn
Connect to your task and ALM tools or create a local task.
Outline
ExemploIDE
main(String[]) : void
10
11
Problems Javadoc Declaration
0 items
Description Resource Path Location Type
9
Writable Smart Insert 1 : 1

- **[7]** Explorador de classes que compõem o seu projeto
- **[8]** Área central de edição de código
- **[9]** Área de visualização de erros/problemas e de mensagens da consola
- **[10]** Área de visualização de métodos e atributos que compõem a classe aberta na área central
- **[11]** Área de visualização de janelas de utilitários
- **[12]** Barras de menus e atalhos
- **[13]** Perspetiva ativa

Vamos agora introduzir código:

› Dentro do método `main` da classe `ExemploIDE` (na área central de edição de código), **comece** por **escrever** as seguintes instruções:

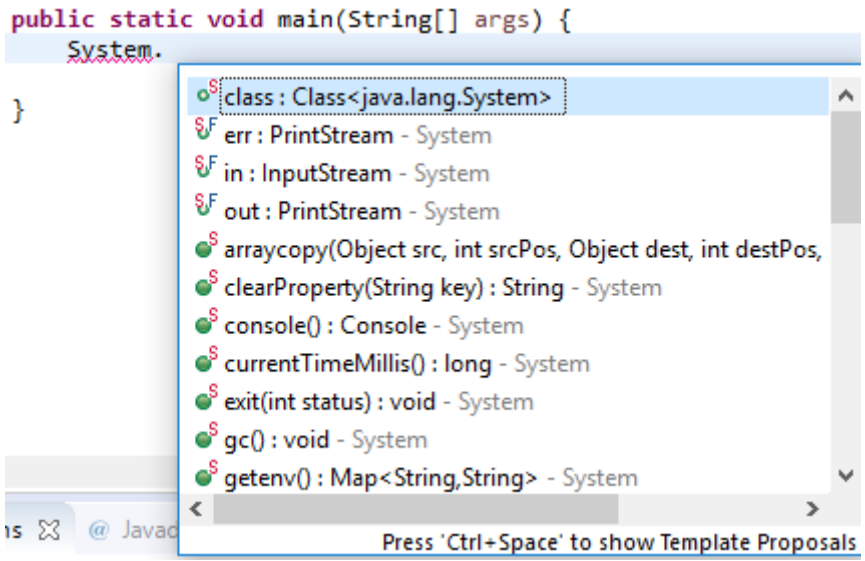
```
String msg = "hello world";
msg = "hello world!!"
```

Repare que na última linha de código foi sublinhada a expressão "hello world!!". Isto porque o Eclipse detetou um erro nesta linha. Se pousarmos o rato por cima deste sublinhado a vermelho é-nos apresentado um descritivo deste erro, bastante útil para tentarmos perceber o que não está correto. Neste caso em concreto falta o "`;`" para terminar a segunda instrução.

› **Corrija** a segunda linha de código e, de seguida, **acrescente** a instrução:

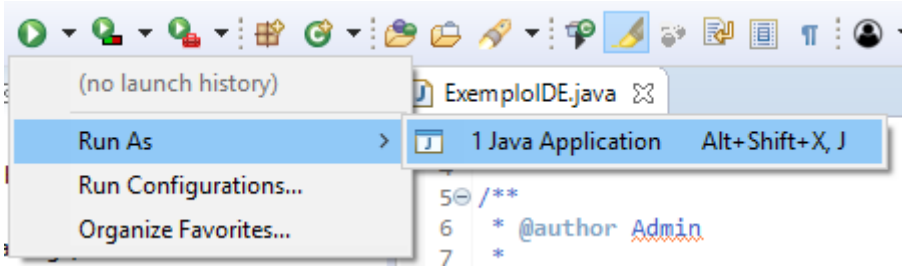
```
System.out.println(msg);
```

Note que quando escreve os pontos, o Eclipse mostra-lhe as opções que tem disponíveis, o que facilita bastante o desenvolvimento do código.




Podemos percorrer a lista de métodos e atributos com as teclas do cursor ou com o rato. À medida que vamos avançando, o Eclipse vai-nos mostrando numa janela adicional informação relativa ao método ou atributo selecionado.

- **Clique** na **seta do botão** 14 e **escolha** a opção **Run As > Java Application**




- **Analise** o resultado na janela **Console** que aparece na **zona** 9

- **Analise** agora o conteúdo da sua pasta **P:**

  - A pasta **P:\CursoJava\bin** é a pasta para onde serão compiladas todas as classes Java.

  - Confirme que a opção **Build automatically** está ativa no menu **Project**.