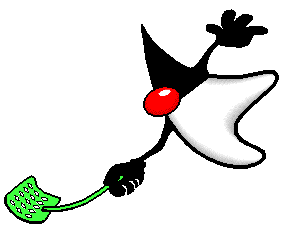


**Ambiente Java/Eclipse para MoP**

C:\workClasses\MoP 1213SV\materials\images\eclipse java.jpg

C:\workClasses\MoP 1213SV\materials\images\eclipse java.jpg

C:\workClasses\MoP 1213SV\materials\images\i1\New Picture (3).jpgC:\workClasses\MoP 1213SV\materials\images\i1\New Picture (5).jpgC:\workClasses\MoP 1213SV\materials\images\i1\New Picture (3).jpg

C:\workClasses\MoP 1213SV\materials\images\i1\Copy (2) of New Picture (3).jpg

C:\workClasses\MoP 1213SV\materials\images\i1\Copy of New Picture (4).jpg



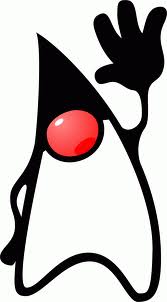
C:\workClasses\MoP 1213SV\materials\images\i1\Copy (2) of New Picture (4).jpg

C:\workClasses\MoP 1213SV\materials\images\i1\Copy of New Picture (3).jpg

C:\workClasses\MoP 1213SV\materials\images\i1\New Picture (3).jpg

C:\workClasses\MoP 1213SV\materials\images\i1\Copy (3) of New Picture (3).jpgC:\workClasses\MoP 1213SV\materials\images\i1\New Picture (3).jpgC:\workClasses\MoP 1213SV\materials\images\i1\New Picture (3).jpgC:\workClasses\MoP 1213SV\materials\images\i1\New Picture (3).jpgC:\workClasses\MoP 1213SV\materials\images\eclipse java.jpg

C:\workClasses\MoP 1213SV\materials\images\i1\New Picture (3).jpgC:\workClasses\MoP 1213SV\materials\images\i1\New Picture (3).jpg

****

**Índex**

[1. Instalações necessárias 1](#_Toc444511207)

[2. Donwload e instalação do JDK/JRE 1](#_Toc444511208)

[3. Download e instalação do eclipse 1](#_Toc444511209)

[4. Acesso à informação online sobre o java 2](#_Toc444511210)

[5. Organização da directoria de MoP e arranque do eclipse 2](#_Toc444511211)

[6. Configuração do eclipse 3](#_Toc444511212)

[7. Criar um projecto java no eclipse 3](#_Toc444511213)

[8. Criar packages e classes java: estruturação packages para MoP 4](#_Toc444511214)

[9. Visualizar os packages em formato plano ou hierárquico 4](#_Toc444511215)

[10. Configuração de Views no eclipse 4](#_Toc444511216)

[11. Criar uma classe Java, content assist, formatação automática e templates 5](#_Toc444511217)

[12. Comentários e task tags em java 6](#_Toc444511218)

[13. Execução de programas no eclipse 7](#_Toc444511219)

[14. Importar código de um package para dentro do eclipse 8](#_Toc444511220)

[15. Importar um projecto eclipse para o eclipse 8](#_Toc444511221)

[16. Perspectivas 8](#_Toc444511222)

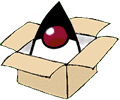
[17. Activar a visualização do número de linha e folding 9](#_Toc444511223)

[18. Execução em Debug 10](#_Toc444511224)

[19. Erros e warnings do eclipse 11](#_Toc444511225)

[20. Help do java 11](#_Toc444511226)

[21. Help do eclipse 12](#_Toc444511227)



# Instalações necessárias

Para MoP é necessário **primeiro** instalar o JDK e **depois** o eclipse. O JDK (Java Development Kit), consiste no ambiente de execução Java, denominado do JRE, ou seja, Java Runtime Environment, e em mais algumas classes e ferramentas extra para desenvolvimento. O eclipse é o IDE (Integrated Development Editor), ou seja o ambiente de desenvolvimento integrado, que vamos utilizar para editar, para compilar e executar as aplicações java. O eclipse utiliza o JRE para a compilação e execução das aplicações java. Com estas duas instalações temos então a possibilidade de editar, compilar e executar aplicações java.

Sugere-se que estas instalações sejam efectuadas para um local único no vosso sistema de ficheiros. Sugere‑se que seja a diretoria: “**C:\Program Files\java**” na versão do Sistema Operativo em inglês ou “**C:\programas\java**” na versão em português.

# Donwload e instalação do JDK/JRE

Sugere-se a instalação do JDK 8 mais recente (consultar slides “MoP 00 – Apresentação.pdf”). Aqui neste documento, vamos considerar que a *release* 202 (JDK 8.0.202). A página da Oracle onde podem fazer o download do JDK é:

<https://www.oracle.com/java/technologies/javase-downloads.html>

Nessa página, em Java SE Donwloads, devem escolher a última versão do java 8 e clicar no seu link de “JDK Download”. Na nova página web de download devem escolher a versão apropriada para o vosso sistema operativo.

Vamos instalar tudo o que seja referente a **java** em **C:\Program Files\Java**. Na instalação deixe as opções por omissão, indicando que o JDK deverá ser instalado em **C:\Program Files\Java\jdk-8.xxxxxx.** Quando aparecer uma janela a indicar: “Destination folder” (que se refere à instalação do JRE público), certifiquem-se que o valor em “Install to:” é a vossa diretoria **java** e como tal deverá aparecer **C:\Program Files\Java\jre-8.xxxxx** e se não for esse o caso primam em “Change” e alterem para localização desejada.

# Download e instalação do eclipse

Sugere-se o download e instalação da versão mais recente do eclipse (consultar slides “MoP 00 – Apresentação.pdf”). A página para download do eclipse é <https://www.eclipse.org/downloads/packages/> e nela devem escolher o package: **Eclipse IDE for Java Developers** tal como se mostra na figura 1. Devem escolher a versão consoante o vosso sistema operativo. Fazem download do ficheiro zip para o desktop. Extraem o conteúdo do zip, ficando com uma diretoria “eclipse”. Sugere-se que alterem o nome dessa diretoria para “**eclipse\_2019\_12”** (se a versão for 2019-12) de forma a poderem reconhecer facilmente a versão instalada. Copiem essa diretoria para dentro da vossa diretoria **java** (referida no ponto 1, **C:\Program Files\Java**) e onde deverá estar instalado o JDK e o JRE público.



Figura 1 - Versão recomendada do eclipse para MoP

|  |
| --- |
| Figura 2 - Conteúdo da diretoria Java (com o eclipse Oxygen 2 e jdk/jre 9.0.4) |

Sugere-se a colocação de um atalho para o Eclipse na Barra de iniciação rápida, que é a barra de programas que está localizada ao lado do símbolo do Windows de Start. Para tal selecionar o executável do eclipse, e escolher “Create shortcut”, depois alterar o nome do shortcut para eclipse\_2019\_12 (ou equivalente), arrastar o shortcut para a barra de iniciação rápida e depois apagar esse shortcut.

Neste momento devem ter a diretoria java com o JDK, o respetivo JRE e o eclipse. Na figura 2 encontra-se o conteúdo esperado dessa diretoria.

# Acesso à informação online sobre o java

A documentação online sobre java para MoP consiste em: descrição da API (Application Programmming Interface) java localizada em <https://docs.oracle.com/javase/8/docs/api/index.html>; e o tutorial java localizado em <http://docs.oracle.com/javase/tutorial>.

Sugere-se a colocação de um icon de acesso rápido para estas duas páginas no vosso browser. Para tal devem adicionar um “favorite” ou um “bookmark” ou algo assim.

# Organização da directoria de MoP e arranque do eclipse

|  |
| --- |
| Figura 3 - Conteúdo da diretoria MoP |

Recomenda-se que o curso e as várias UCs estejam organizadas tal que MoP se situe em: C:\LEIM\2Sem\MoP, ou seja, um *path* pequeno, sem espaços, nem caracteres fora do ASCII de 7-bits.

Sugere-se fortemente que MoP (tal como as outas UCs) contenha as seguintes subdiretorias (tal como indicado na figura 3):

* code,
* exames,
* materiais,
* slides e
* trabalhos Práticos.

Onde cada diretoria deverá conter os respetivos conteúdos. **Devem criar estas diretorias de imediato**.

O eclipse permite trabalhar com várias áreas de trabalho, mas uma de cada vez. As áreas de trabalho são denominadas de workspaces. Assim no arranque de uma execução do eclipse é necessário indicar qual o qual o workspace a carregar. Sugere-se que seja criado um workspace só para MoP, que será localizado na diretoria de nome **code**, dentro da diretoria de MoP

Abra então eclipse e indique como localização (do **workspace)** a directoria **code** de MoP, com a localização: C:LEIM\2Sem\MoP\code. Selecione a checkbox de “Use this as the default and do not ask again”.

Após abrir o eclipse pela primeira vez, abre-se uma página central de *Welcome*, pode fechar essa página.

O eclipse suporta vários projetos, eventualmente cada um numa linguagem de programação diferente. Mas aqui em MoP só estaremos interessados em projetos Java. Cada projeto Java pode ter vários packages, que são entidades Java semelhantes a diretorias, ou seja, que servem para estruturar onde está cada elemento. Cada package pode conter várias classes ou packages Java ou outros elementos.

Em Mop vamos criar um só projeto Java para toda a disciplina. Tal será executado no ponto 7. No ponto 8 iremos criar os packages bookcode, classcode, tps, testes e exames que irão conter respetivamente o código do livro do Savitch, o código das aulas, o vosso código dos trabalhos, o vosso código de testes e o vosso código da resolução de exames. Esperem até ao ponto 8 para concretizar a criação dos packages

# Configuração do eclipse

Neste ponto pretende-se configurar o JRE (Java Runtime Environment) que irá executar as aplicações java dentro do eclipse. O JRE é o ambiente de execução java. Pretendemos que seja utilizado o JRE dentro do JDK que instalámos no ponto 2 pois contém o respetivo código fonte.

Assim no eclipse em window→preferences→Java→”Installed JREs” se já se encontrar o JRE desejado (jre-8.xxxx) passar ao ponto seguinte. Mas antes certifiquem-se que essa instalação cumpre as indicações seguintes, fazendo Edit a esse JRE.

Caso o JRE que consta no eclipse não seja o desejado então temos que adicionar um JRE novo, com Add, Standard VM, em “JRE home” localizar o JDK e nele o seu JRE, algo como: C:\Program Files\Java\jre-8.xxxx e fazer OK. Verificar que o nome da instalação do JRE será “jre-8.xxxx”. Verificar que selecionando a linha de jrt-fs.jar e que em javadoc location que referencia o site do javadoc (java documentation) na oracle (http://download.java.net/java/jdk8/docs/api/). Sob a mesma library (jrt-fs.jar) selecionar Source attachment e nela colocar o ficheiro (src.zip) com as sources (código fonte) do Java dentro do JDK em C:\Program Files\Java\jdk-8.xxxxx\src.zip. Para tal escolher External location, External File e escolher o ficheiro já indicado. Terminar a instalação e regressar à janela de Installed JREs. Nela selecionar o jre-8.xxx como o *Default JRE* e eliminar qualquer outro que haja. Fazer Apply and Close.

# Criar um projecto java no eclipse

Para criar um projeto Java ir ao menu de **File** escolher o menuItem de **New** e escolher a opção de **Java Project** (File→New→Project). Na janela do novo projeto colocar como name “MoP”, verificar que está selecionada a opção em “Project layout” de “Create separate folders for sources and classes files” e faça Finish. Na figura 3 pode observar o conteúdo do Package Explorer já com o novo projeto.

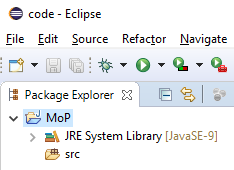


Figura 3 - Novo projecto java

Na própria janela do Package Explorer com um click do botão direito do rato (CLICKRIGHTBUTTON - CRB) também se pode aceder à mesma janela onde selecionar um novo projeto java.

# Criar packages e classes java: estruturação packages para MoP

Os packages java permitem estruturar onde as classes java e outros ficheiros vão ficar.

Para criar um package, sobre o diretório de src do projeto fazer CRB sobre src, escolher New→Packag e colocar o nome do novo package com o nome pretendido.

Tal como já indicado, sugere-se a criação dos packages: **bookcode**, **classcode**, **tps, testes** e **exames**. Assim, em src, crie os packages mencionados.

Agora crie o package p01IntroJava dentro do package classcode. Para tal selecione com o rato o package classcode e crie novo package colocando no nome “classcode.p01IntroJava”. Um package tem como nome o seu pathname completo. Tome em nome que no eclipse é sempre necessário escrever o pathname completo dos packages. Crie também o package classcode.p02Basic. Observe no sistema de ficheiros que foi criada uma diretoria para cada package.

# Visualizar os packages em formato plano ou hierárquico

Os packages podem ser visualizados em formato plano ou hierárquico. O formato plano é o formato apresentado na figura do lado esquerdo da tabela 2 e nesse formato os packages são visualizados cada um numa linha e com o seu nome completo. O formato hierárquico apresenta os packages num formato tipo árvore mostrando para cada package o seu conteúdo, de packages e ficheiros, em que de cada package é mostrado apenas o seu nome.

Para selecionar entre estes dois tipos de visualização selecionar o ícone de “View Menu”, que é o ícone triangular sinalizado com um círculo a vermelho na parte esquerda da tabela 2 e dentro dele selecionar *Package Presentation* e selecionar *Hierarchical*.

|  |  |
| --- | --- |
| C:\workClasses\MoP 1213SV\materials\images\New Picture (4).jpg | C:\workClasses\MoP 1213SV\materials\images\New Picture (5).jpg |

Tabela 1 - Visualização de packages em formato plano ou hierárquico

# Configuração de Views no eclipse

No eclipse há janelas que contêm editores e outras que contêm visualizadores. Na figura 4 pode-se observar três zonas de janelas: a parte esquerda que contém a *view* do *Package Explorer*; a parte direita superior que contém a janela do editor Java com vários ficheiros abertos; e a parte direita inferior que contém as *views* de *Problems*, *Javadoc*, *Declaration* e *Console*. Todos estes elementos podem ser dispostos na área de trabalho do eclipse da forma que se pretenda. Se o seu eclipse contém abertas as *view*s de *Task* *List* e *Outline* feche-as. Pode sempre mais tarde ativá-las caso deseje.

Cada *view* mostra, e em alguns casos permite algum tipo de edição, alguma parte dos artefactos existentes no workspace aberto.

A *view* de *Package Explorer* permite visualizar todos os projetos, packages, classes, atributos e métodos das classes, e ficheiros existentes no workspaces. Esta é a *view* mais importante. A *view* de *Console* permite visualizar o que é escrito na consola. A *view* de *Problems* mostra os erros (*errors*) e os avisos (*warnings*) existentes sobretudo nos ficheiros de código. A *view* de *Javadoc* mostra a informação existente acerca da classe, método ou atributo java selecionado no corrente ficheiro java no editor java. A *view* de *Declaration* permite visualizar o código java associado à declaração da classe, método ou atributo java correntemente selecionado no editor java. Sugere-se a utilização destas *view*s mencionadas.

Pode-se saber quais as *views* existentes no eclipse, indo ao menu de *Window* e selecionando a opção de “*Show view*”. Nesta opção pode-se ver um conjunto de *views*, mas para ver de facto todas deve-se selecionar a última linha que deverá conter o texto de “*Other*…”. Na janela que se abrirá poderão explorar e selecionar as vistas que pretendam. Sugere-se a exploração, ao longo do tempo, das vistas existentes, mas para início devem ativar a vista de Console além das vistas já instaladas de *Problem*, *Javadoc* e *Declaration*.

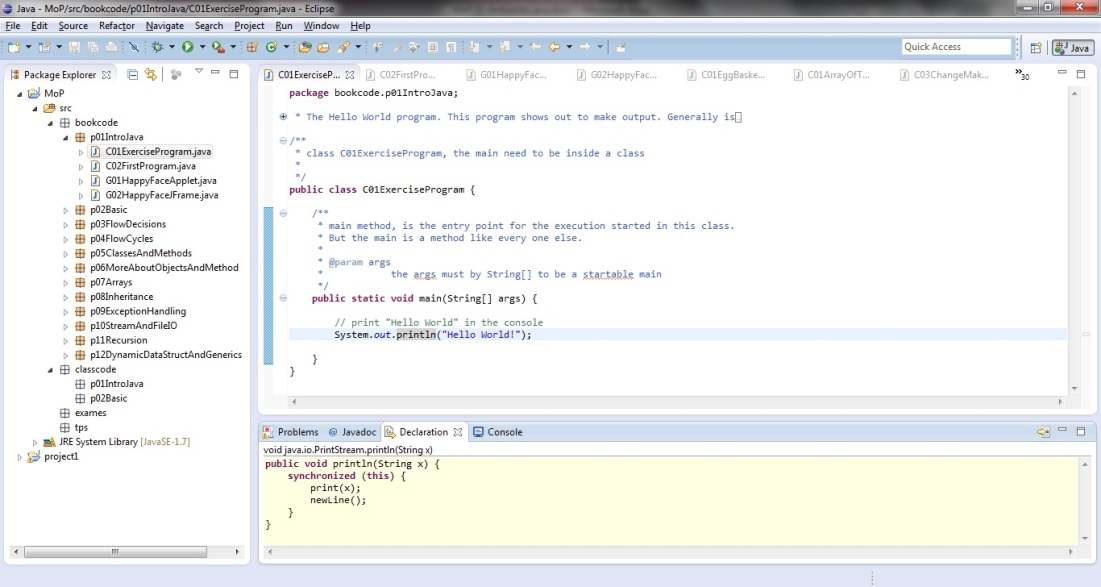


Figura 4 - Views no eclipse

# Criar uma classe Java, content assist, formatação automática e templates

Uma classe java deve preferencialmente ficar dentro de um package. Em MoP iremos colocar as classes sempre dentro de packages.

Para criar uma classe deve-se: escolher o package no qual a classe deve ficar, sobre ele escolher a opção de New→Class; colocar o nome da nova classe, que por convenção deve começar com uma letra maiúscula; selecionar as opções pretendidas; e indicar Finish.

Para já vamos criar a nossa primeira classe de nome C01MyFirstProgram. Sugere-se a colocação de um número de série nas classes, em tempo de aprendizagem, para que se consiga identificar a sequência com que foram criadas. A nova classe deverá ser criada dentro do package classcode.p01IntroJava. Assim deve‑se: começar por selecionar o respetivo package; sobre o package selecionar n New→Class; colocar como nome da classe C01MyFirstProgram; ativar a opção de criar o método main (primeira opção dentro de “Which method stubs would you like to create?”; e fazer Finish. Deverá aparecer no editor um novo ficheiro com o nome de C01MyFirstProgram.java e também aparecer no Package Explorer esse mesmo ficheiro no package correcto.

Dentro do novo ficheiro pode-se observar: que se inica com a descrição do package onde ele se situa; que contém a definição de uma classe com o mesmo nome que o ficheiro; que dentro da classe existe o método main, tal como: public static void main(String[] args) { … }.

Dentro do main: elimine a linha que contém a indicação de TODO; e adicione dentro do método main a linha System.out.println(“Olá pessoal”); em que as aspas em volta de “Olá pessoal” devem ser corrigidas para aspas normais. Para fazer isso sugere-se que escreva somente “sy” depois faça CTRL+SPACE para activar o assistente de edição (**Content assist** - CA). Na janela de CA ao selecionar um elemento abre-se uma outra janela mais à direita com a documentação associada a esse elemento. Recomenda-se a leitura dessa informação para se adquirir um conhecimento concreto do que se utiliza e do que se poderia utilizar. Na janela do CA selecione a linha de System - java.lang e prima Enter ou duplo click. Adicione um ponto e verá que se abre uma outra janela de CA mostrando o conteúdo possível de ser utilizado dentro da classe System. Nessa janela selecione “out” que é um PrintStream. Escreva outro ponto e nova janela de Ca se abrirá para mostrar o que se pode fazer com o System.out. Nessa janela seleccione o método println(String), colocando depois na String o texto “Olá pessoal”. Coloque o ponto e vírgula como **terminador**. O código deve ficar tal como está na figura 5. Faça **CTRL+SHIFT-F** para **formatar automaticamente** o código e assim alinhá-lo de forma automática e depois faça **CTRL+S** para guardar as alterações em disco.

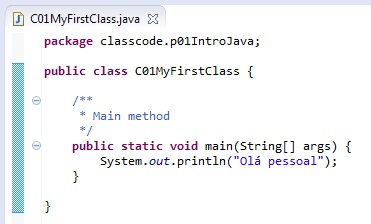


Figura 5 - Conteúdo de C01MyFirstProgram.java

Também se poderia escrever “System.out.println();” de uma forma expedita, escrevendo somente sysout e premindo o CTRL+SPACE. Tal é devido porque existe um **template** que associa sysout a “System.out.println();”. Esses templates podem ser simples como este ou complexos e podem ser consultados ou definidos pelo menu em Window→Preferences→Java→Editor→Templates.

# Comentários e task tags em java

No java, como em muitas linguagens, existem os comentários de linha e os comentários de bloco.

Os comentários de linha iniciam-se com “//” e englobam todo o texto desde esses caracteres até ao final dessa linha. Esses comentários são visualizados no eclipse a verde.

Os comentários de bloco iniciam-se com “/\*”, terminam com “\*/” e podem ocupar qualquer número de linhas. Estes comentários são mostrados pelo eclipse também a verde. No entanto se o comentário começar por “/\*\*” isso representa que o bloco é um bloco de documentação, ou seja, que descreve formalmente a funcionalidade do símbolo em questão. Nesse caso o bloco é mostrado a cor azul.

Os comentários podem ter a palavra de TODO (em maiúsculas) e tal indica que existe uma tarefa para ser realizada, daí TODO se denominar de ser uma “Task tag”, ou seja uma etiqueta de tarefa. O eclipse quando cria um método de forma automática coloca uma linha com essa palavra/tag a indicar que algo deve feito nesse lugar.

Existem as seguintes task tag definidas por omissão no eclipse: TODO, FIXME e XXX. As task tags são mostradas a azul, por omissão e podem ser geridas, ou seja, adicionadas alteradas ou eliminadas, projeto a projeto na opção de menu Project→Properties→Java Compiler→Task Tags. As task tags (marcadas a vermelho na parte central da figura 6) permitem então identificar os locais onde se deixou algo por fazer, e pode-se identificar essas tags na view de Tasks (linhas a vermelho na mesma figura) e nas colunas laterais do editor (elipses a vermelho).



Figura 6 - Task tags

# Execução de programas no eclipse

Para executar uma classe é necessário que essa classe tenha o método **main** com a seguinte definição: public static void main(String args). O método main é um método como outro qualquer e pode ser chamado como outro qualquer, mas tem a particularidade de se iniciar que a execução começa pela classe em questão, então esse será o método de ponto de início da execução.

Para se lançar a execução de classe, ou seja, do seu main, pode-se selecionar a classe no Package Explorer e sobre ela escolher a opção de “Run As”→“Java Application” ou fazer CTRL+F11. Também se pode ir pelo menu de RUN e escolher a opção de Run. Caso se esteja na parte de edição também se pode fazer diretamente CTRL+F11.

Se executar um programa que envie texto para a consola poderá verificar que aparecerá esse texto na view de Console.

Quando executar um programa o eclipse cria uma Run Configuration, ou seja, uma configuração de execução, na qual indica qual a classe, de que package, quais os argumentos, quais as opções da máquina virtual java, etc, etc. Assim após uma primeira execução, já se pode utilizar o atalho de execução através do ícone de execução (círculo verde com uma seta a branco), que está assinalado com um círculo mais reforçado na figura 7. Também é possível aceder a esse histórico de execuções pelo menu de Run escolhendo o item de Run History.

Pelo ícone de run, ou pelo menu de run, é possível aceder à gestão das configurações, para por exemplo alterar os parâmetros de chamada do main, ou fazer duplicados de uma configuração com diferentes parâmetros.

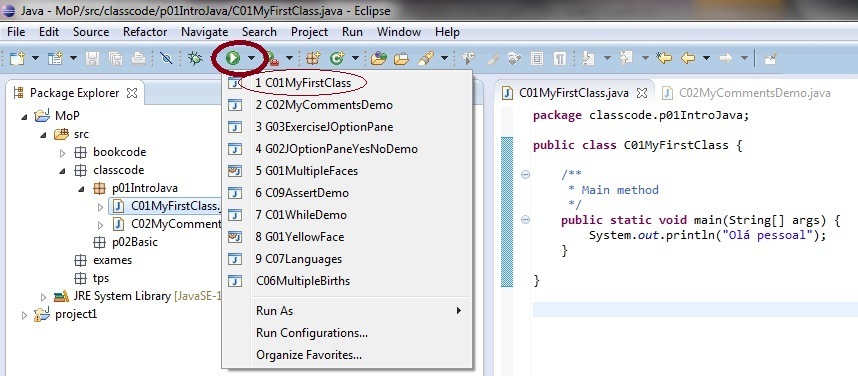


Figura 7 - Execução de um programa no eclipse

# Importar código de um package para dentro do eclipse

Caso receba uma diretoria que corresponde a um package e dentro dela suas classes ou outras diretorias basta fazer copy dessa diretoria e paste sobre local onde esse package deverá ficar.

Por exemplo o ficheiro zip que contém o código do livro do Savitch (bookcode.zip) disponibilizado no moodle, contém a directoria bookcode e dentro dela existem várias diretórias tal como p01Introjava. Assim basta copiar a diretoria bookcode para cima do icon de src, que os packages ficam disponíveis no projeto e sem qualquer erro de localização.

# Importar um projecto eclipse para o eclipse

Pode acontecer ter que importar um projeto inteiro já existente para dentro do seu workspace. Neste caso convém reconhecer que o que recebe é um projecto, e para tal deve verificar que dentro da diretoria existem os ficheiros “.classpath”, “project” e as directorias de src e “.settings”.

Para importar: no Package Explorer, com o rato, escolha a opção “Import…”; depois escolha General e “Import Existing Projects into Workspace”; faça Next; selecione a diretoria raiz onde o projeto está; selecione o projeto em questão na parte de Projects; selecione a opção “Copy projects into workspace”; e faça Finish.

# Perspetivas

Uma perspetiva é somente um conjunto predefinido de janelas (views) e editores com um determinado objectivo. Cada uma delas é configurável no sentido que se pode alterar as vistas activadas. Para MoP interessa-nos a perspetiva de Java, que é a perspetiva por omissão, e a perspetiva de Debug. A perspetiva de Java é dedicada a conter as janelas (views) que facilitam a edição de java. Enquanto que a perspetiva de Debug destina-se a conter as janelas (views) que facilitam a execução em debug em java. A parte de debug será apresentada em seguida.

Em qualquer uma das perspetivas podemos gerir as janelas/views existentes da forma que nos for mais conveniente, basta para isso adicionar, remover e posicionar janelas (views).

Tal como já referido no ponto 10 a perspectiva de Java contém o editor Java. Aconselha-se a que nela tenham as seguintes views: Package Explorer, Console, Declaration, Problems, Tasks, Javadoc e também as views de **Call Hierarchy** e eventualmente Outline.

A perspetiva de Debug contém o editor de java e aconselha-se a se ter as seguintes views: **Variables**, **Breakpoints**, **Debug**, **Declaration** e **Console**. Aconselha-se a posicionar as janelas tal como estão na figura 8.

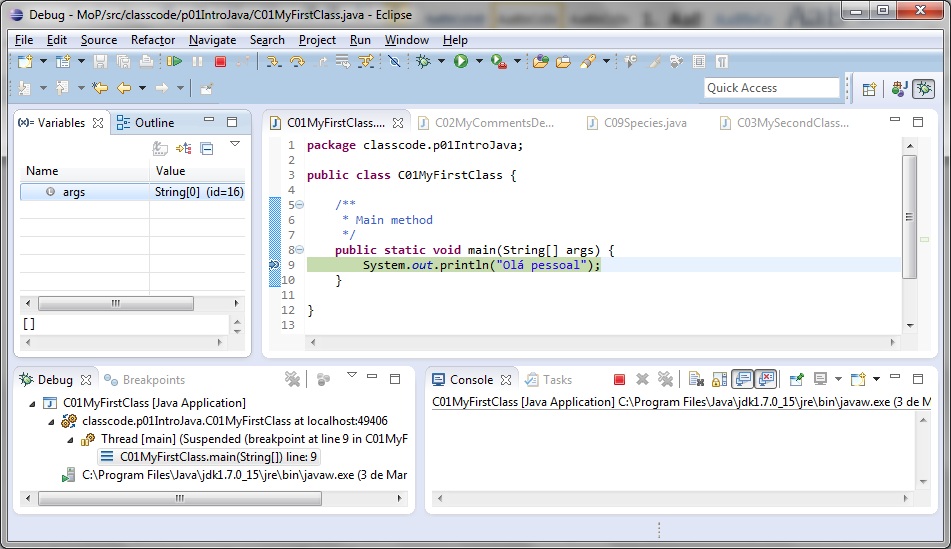


Figura 8 - Perspetiva de Debug

Associadas ao java também existem as perspetivas de Java Browsing e Java Type Hierarchy. Contudo para MoP estas perspectivas não se justificam.

As perspetivas podem ser geridas no menu em Windows→”Open Perpective”→”Other…” e na parte superior direita do eclipse (ver figura 4, onde na qual se pode ver o icon e texto da perspectiva de Java) onde as perspetivas activadas são mostradas com um símbolo e eventualmente com o seu nome e onde está um icon com um sinal mais que permite aceder à janela de adição de perspetivas. Para remover uma perspetiva basta no seu icon escolher a opção de Close.

# Activar a visualização do número de linha e folding

Crie a classe C03 tal como se mostra na figura 9. Tal como se pode observar a visualização do número de linha providencia uma informação muito útil. Para ativar ou desativar o número de linha deve-se premir o botão direito do rato na coluna à esquerda do editor (onde está a elipse mais à esquerda na figura 6) e ativar ou desativar “Show Line Numbers”.

Quando o código é extenso torna-se útil fazer o *folding* dos métodos, que significa reduzir a visualização do método à sua primeira linha de cabeçalho. Desta maneira numa única página localiza-se facilmente mais de uma dezena de métodos permitindo o seu acesso de forma imediata. Pode-se observar na figura 8 e noutras figuras com código, que nos métodos e comentários de bloco que a sua linha inicial está precedia de um símbolo redondo com um sinal de -. Carregando nesse símbolo fechamos a visualização e o símbolo fica com um sinal de mais, a indicar que se pressionado mostrará mais informação. Sobre essa coluna á esquerda, que de facto contém várias colunas, pode-se controlar o *folding*, ativando-o ou desativando-o.

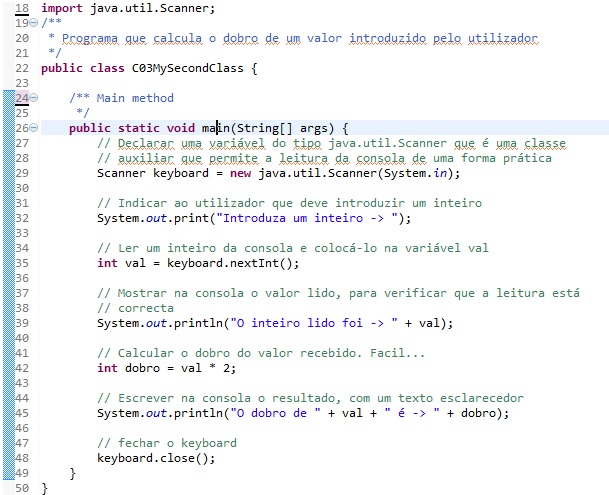


Figura 9 - Classe C03MySecondClass

# Execução em Debug

A execução em *debug* é uma forma de executar um programa que permite a execução linha a linha do seu código e a observação do conteúdo das variáveis existentes. É portanto, uma ferramenta indispensável para a observação do funcionamento dos programas, pois permite verificar se o programa está a comportar-se em linha com o esperado e constatar qual a primeira linha em que ele se desvia do esperado. Dessa forma a sua correção fica facilitada.

Para exercitar esta execução sugere-se que em C03 coloque um *breakpoint*, ou seja um ponto de paragem da execução, na linha com “System.out.println("O inteiro lido foi -> " + val);”, o que é conseguido fazendo duplo *click* na coluna de *breakpoints* na referida linha. Na figura 10 pode-se observar a sinalização da existência de um *breakpoint* o qual está assinalado com um círculo a vermelho.

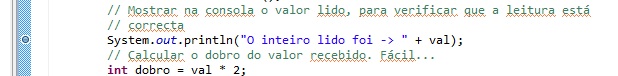


Figura 10 - Colocação de breakpoints

Para ativar o modo de *debug* tem de executar a classe em modo *debug*. Para tal, sobre a classe, com o rato seleccione: “Debug As”→“Java Application” numa primeira execução. Depois dessa primeira execução já foi criada um “Run Configuration” e então já se pode lançar a execução em *Debug* pelo *ícone* de *Debug* que é o icon com um escaravelho ao lado do ícone de *Run*.

Iniciada a execução em *debug* vai aparecer uma janela a indicar se pretende mudar para a perspetiva de *Debug*, indique que sim e para lembrar da decisão. Verifique que a execução está parada à espera de input e ainda não chegou à linha do ponto de paragem (*breakpoint*), introduza o valor de 20 e prima *enter*. Constate que a execução está agora parada no *breakpoint*. A linha a verde que indica a execução está sobre essa linha. Pode observar que na janela de *Debug* a execução (Thread) do *Main* está parada na linha 43 (a linha do System.out.println). Agora pode-se observar o estado das variáveis na *view* de *Variables*. Verifique que “val” tem o valor introduzido de 20. Execute a linha corrente com F6 (Step over). Constate que apareceu na consola: “O inteiro lido foi -> 20”. Faça novamente F6 e veja que foi criada a nova variável “dobro” com o valor de 40. Prima em F8 (ou no Menu de Run prima em Resume) e observe que a execução terminou e que foi mostrado na consola o resultado da operação pretendida

Quando pretender terminar uma execução pode fazê-lo clicando no icon com o quadrado vermelho, que é o ícone de *Terminate*. Se pretender voltar à edição de código mude para a perspetiva java.

Salienta-se os seguintes comandos em *debug* (ver entrada de *Run* do menu principal): F5 – *Step into*, executa entrando dentro do código; F6 – *Step Over*, executa, saltando a próxima linha de código; F7 – *Step return*, executa até fazer *return*; CTRL R – *Runt to Line*, executa até à linha onde se encontra o rato; *Terminate*, termina a execução corrente; F8 – *Resume*, coloca em execução, só vai parar no próximo *breakpoint*.

Depois de terminar uma execução é necessário limpar o estado dessa sessão de *debug*, para isso na janela de *Consola* selecionar o ícone com uma cruz que deve estar a negro e que indica “*Remove Launch*”.

# Erros e warnings do eclipse

Na linha a seguir a “int dobro = val \* 2;” escreva: “int triplo;” e salve/grave o ficheiro. Verifique que ocorreu um aviso a indicar que a variável não é utilizada: a palavra triplo ficou sublinhada a amarelo; apareceu um símbolo de lâmpada na “coluna dos *breakpoints*” com triângulo com uma exclamação a amarelo; apareceu um pequeno quadrado a amarelo na coluna do lado direito. Verifique no Package Explorer que a classe, o package, src e o projeto todos ficaram com um triângulo com uma exclamação a amarelo.

Caso tivesse editado um erro, como por exemplo altere o “int triplo;” para “intttt triplo;” e verá que aparece um quadrado com uma cruz a vermelho nesses locais todos.

Abra a vista de *Problems* para visualizar os erros e *warnings* existentes.

Os erros impedem a execução por isso têm ser resolvidos. Mas os *warnings*, que são avisos, permitem que a execução se faça na mesma. **Contudo como regra de boa programação exige-se que os programas em MoP não tenham warnings.**

Verifique que quando tem um *warning* ou um erro que por cima do símbolo da lâmpada que assinala o *warning* ou o erro pode obter a descrição e sugestões de resolução automática. A resolução automática só poderá sugerir o que se poderá fazer de forma automática e não poderá oferecer a melhor solução para o problema se ele não for de solução direta.

# Help do java

Para se obter o *help*, ou seja a documentação de um elemento java, basta colocar o rato sobre esse elemento que se abrirá uma janela de documentação com a informação existente sobre esse elemento.

Pode-se abrir uma página com um browser com o *javadoc* (da Oracle) do símbolo selecionado premindo SHIFT+F2 sobre o símbolo.

Tal como já descrito o *help* do java pode ser obtido na API e no *tutorial*. Esses dois elementos podem ser consultados *online* em:

<http://docs.oracle.com/javase/8/docs/api/index.html>

<http://docs.oracle.com/javase/tutorial/index.html>

# Help do eclipse

O *help* do eclipse é vocacionado para ajudar na utilização do eclipse e no java em geral. Não sendo, portanto, uma ajuda à linguagem Java em si.