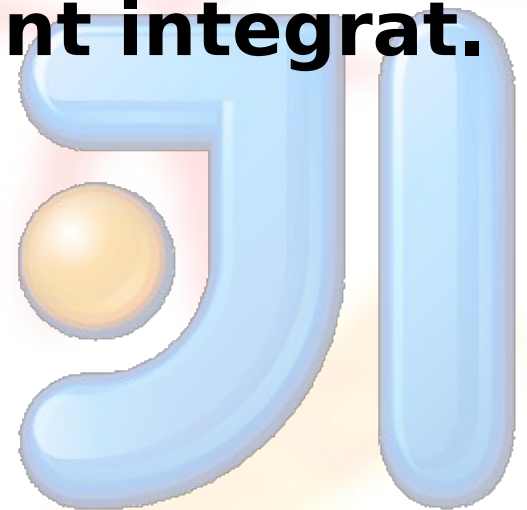
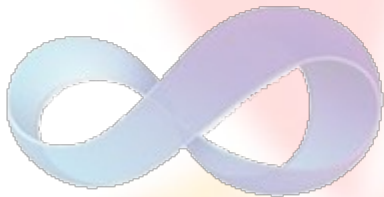
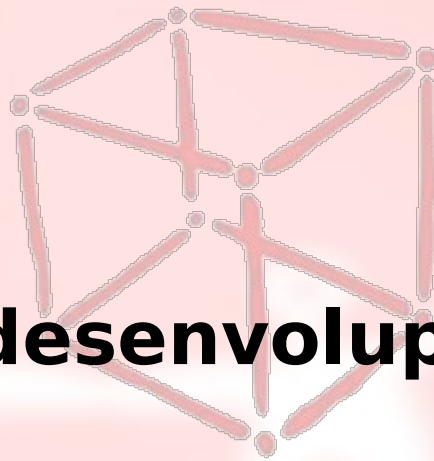


Entorns de desenvolupament.



Tema 2:

Entorns de desenvolupament integrat.



Entorn de desenvolupament

Ja hem vist com és el procés de desenvolupament de codi: el programador escriu el codi font, corregeix les errades, el compila, l'enllaça a les llibreries i obté el codi executable.

Totes les eines que utilitzi per a realitzar aquestes tasques formen l'entorn de desenvolupament utilitzat per aquest desenvolupament.

Eines que poden formar part d'aquest entorn:

- Un editor de text: notepad, notepad++, gedit, ...
- El compilador
- L'enllaçador
- Un debugger
- Màquines virtuals
- Intèrprets
- ...

Per exemple, per desenvolupar una aplicació java necessitam només un editor de text, el compilador *javac* i el JRE, amb *java* per executar el programa i fer proves.

Cada vegada que volem comprovar uns canvis que haguem al codi haurem de

- Guardar el codi font.
- Anar a la finestra de comandes o terminal i allà
- Compilar el codi font amb *javac*
- Si hi ha errors,
 - mirar a quina línia ens diu el compilador que es troba l'error
 - Tornar a l'editor de text
 - Cercar la línia
 - Corregir-lo
- Si no hi ha errors,
 - executar-lo

Entorn de desenvolupament integrat

Un entorn de desenvolupament integrat, a partir d'ara IDE, és un programari que inclou totes les eines necessàries per a desenvolupar codi en un llenguatge determinat o inclús en diversos llenguatges.

La idea de reunir en una sola aplicació totes les eines necessàries per al desenvolupament de programari cerca facilitar la tasca del programador, millorar el seu rendiment i augmentar la qualitat del codi elaborat.

Per exemple en un IDE, com ara eclipse, compilar i executar un programa és tan fàcil com pitjar un botó que fins i tot ens guarda el codi font si hi havia modificacions abans de compilar-lo i executar el resultat final.

L'elecció d'un IDE adequat al projecte que estem desenvolupant i la seva utilització eficient faciliten molt la tasca dels desenvolupadors, augmenta la seva productivitat i ajuda a augmentar la qualitat del codi.

Funcions d'un IDE

- **GUI: Interfície gràfica d'usuari.** Normalment un IDE disposa d'una interfície gràfica amb botons, finestres, ... que reuneix en un sol lloc totes les possibilitats de l'entorn.
- **Editor de text:** Es impensable un entorn dedicat a picar codi sense un editor de text que ens ho permeti. Els editors solen incorporar funcions avançades de les que no disposa el bloc de notes, com per exemple:
 - **Resaltat de la sintaxi:** Mentre s'escrivim, en reconèixer una paraula clau del llenguatge, o un literal els escriu amb una tipografia diferent per facilitar la lectura del codi.
 - **Detecció d'errades i advertències:** Mentre escrivim, analitza el codi i si detecta errades de compilació marca la línia on es troben.
 - **Compleció codi:** Ens ajuda a completar el codi. En escriure una el nom d'una classe o d'un objecte ens mostra els seus membres, ...
 - **Regions desplegable:** Permet definir apartats de codi que es poden amagar o desplegar a voluntat per facilitar la visualització del que realment ens interessa.
 - ...

- **Compilador:** Si l'IDE està pensat per treballar amb un llenguatge compilat normalment incorporarà un compilador per a generar el codi executable. Normalment localitzarà les errades i ens dirà on són. Si no n'hi ha generarà el codi executable.
- **Intèrpret:** Si l'IDE està pensat per treballar amb un llenguatge interpretat normalment incorporarà un interpret o qualche mecanisme per llençar un intèrpret i provar el codi.
- **Depurador:** En anglès debugger. És un programa que permet provar i trobar errades al nostre codi en temps d'execució. Algunes de les seves funcions típiques són:
 - Punts d'interrupció: Aturar l'execució del codi en una certa instrucció del programa.
 - Control de variables: Veure en tot moment el valor que van prenent les variables, normalment és útil en combinació amb els punts d'interrupció.
 - Execució pas a pas del codi: Per veure per on va passant, quines instruccions s'executen i quins valor prenen les variables.

- **Accés a bases de dades:** Opcionalment poden incloure eines d'accés a bases de dades per controlar la interacció amb elles del nostre programa.
- **Gestió d'arxius:** Normalment incorporen un gestor d'arxius que ens permet moure fitxers dins el projecte, copiar-los, eliminar-los...
- **Control de versions:** Solen incorporar qualche sistema de control de versions, molt útil sobre tot en treballar en equip.
- **Refactorització:** És una tècnica, o millor dit, tota una sèrie de tècniques, per canviar l'estructura del codi sense alterar la seva funcionalitat però millorant la seva eficiència. Els IDEs solen incorporar eines per a fer-ho.
- **Documentació:** També solen incorporar eines que faciliten la documentació de l'aplicació, per exemple JavaDoc.
- **Ajuda:** La gran majoria incorpora diferents eines d'ajuda, no només sobre la utilització de l'IDE, sinó també sobre la sintaxi del llenguatge que estiguem utilitzant.

Instal·lació d'un IDE

La instal·lació d'un entorn de desenvolupament integrat, com podeu suposar no segueix un patró general, cada un té les seves peculiaritats. Si volem treballar en Java el primer que haurem de fer és instal·lar el JDK. El que ve després ja depèn de l'IDE en qüestió.

Eclipse

La instal·lació d'eclipse es realment senzilla.

- Baixam de la seva pàgina la versió que ens interressi tenint en compte perquè el volem utilitzar(quines utilitats inclourà) i l'arquitectura del sistema en el que l'utilitzarem.
- Descomprimim l'arxiu en un directori sobre el que tinguem permisos d'execució. Intentau no fer-ho a l'escriptori. Es crearà una carpeta *eclipse*.
- Executem l'eclipse de dins aquesta carpeta.
- Ens demanarà quin workspace volem utilitzar. Podriem dir que el workspace és el magatzem on tendrem les nostres aplicacions. Escollim una carpeta i si marcam la casella no ens tornarà a fer aquesta pregunta.

Utilització d'un IDE

Projecte

Els IDE's de Java normalment utilitzen el concepte de projecte. Un **projecte** és tot el conjunt de fitxers, tant de codi com auxiliars (imatges, text, ...), paquets, ... que formen part d'una aplicació.

Normalment els projectes només tenen una classe executable, una classe que tengui el mètode *main*. Noltros, per començar ens botarem aquesta limitació i farem que cada exercici tengui la seva pròpia classe dins el *main*.

El primer que farem serà crear un projecte Java. Anam al menú *File->New->Java Project*. Si no el trobau, anau a *File->New->Others->Java Project*.

Al diàleg que s'obrirà posarem el nom del projecte i pitjarem el botó *Finish*. Pot ser que eclipse vos demani si voleu obrir una perspectiva. Marcau la casella per que recordi la vostra decisió i contestau afirmativament.

Una **perspectiva** no és més que un conjunt de vistes o plafons que obrirà l'eclipse per facilitar una determinada tasca.

Java - Proves2/src/exercicis/Tercera.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access

Java EE Debug Java

Package Explorer

- Proves
- Proves2
 - src
 - exercicis
 - Segona.java
 - Tercera.java
- Tema2
 - src
 - JRE System Library [JavaSE-1.7]

Primera.java Segona.java *Tercera.java

```
1 package exercicis;
2
3 public class Tercera {
4
5     public static void main(String[] args) {
6         System.out.println("Segona");
7         System.out.
8     }
9 }
10
11 }
12
```

Task List

Find All Activ...

Connect Mylyn

Connect to your task and ALM tools or create a local task.

Outline

- exercicis
- Tercera
 - main(String[]) : void

Problems Javadoc Declaration Console

<terminated> Primera [Java Application] /usr/lib/jvm/java-7-oracle/bin/java (17/10/2013 11.28.37)

Primera

Writable Smart Insert 8 : 1

A la imatge anterior podeu veure l'aspecte de la perspectiva de java de l'eclipse:

- A l'esquerra teniu l'explorador de projectes. Hi teniu tots els projectes del vostre *workspace*. Si desplegau un projecte apareix una carpeta *src* on tendreu tot el vostre codi font, tots els vostres arxius *.java*

Des d'aquesta finestra podeu copiar, renombrar(refactor), eliminar arxius. És aconsellable fer-ho sempre des d'aquí i no directament des del sistema operatiu. Per exemple, si copiau un arxiu Java vos demanarà el nom de la còpia i posarà el mateix nom a la classe dins el codi. Recordau que la classe i el fitxer han de tenir exactament el mateix nom.

- Al centre hi trobau l'editor de text. Podeu veure que pinta les paraules reservades Java d'un color, les cadenes de text d'un altre, ...

També ens marca a quina línia hi ha errades de sintaxi, amb una creu vermella a l'esquerra (si posam el cursor al damunt ens diu quin és l'error, i si el pitjam ens proposa una solució) i amb una línia vermella a l'esquerra que és útil en programes llargs ja que clicant-hi a sobre ens porta a la línia en qüestió.

Si vos apareix una icona d'una bombilla groga i una exclamació significa que en aquella línia hi ha un warning (una alerta), no és una errada però s'hauria de revisar

Per mostrar els nombres de línia(molt recomanable) heu de pitjar el botó dret del ratolí sobre la barra on apareix la creu vermella i marcar *Show line numbers*.

- A la dreta hi trobareu
 - Una finestra de tasques, que es pot utilitzar per posar recordatoris de coses pendents
 - Una finestra que posa outline i que en programes complexes és molt útil perquè permet accedir directament a mètodes o atributs de la classe que estam editant.
- Finalment a la part inferior teniu una finestra amb una sèrie de pestanyes:
 - Consola: simula la “finestra negra” del windows. Apareixen els missatges que volguem mostrar i hi escriurem per teclat el que ens demani el programa.
 - Problems: Mostra les errades i les alertes del projecte actual. Si clicam a sobre d'un d'ells ens porta a la línia que provoca l'errada.
 - Javadoc: Ens mostra l'ajuda de les classes de l'API de Java.

A part d'aquestes finestres, la perspectiva es pot modificar des del menú *Windows*:

- *Customize perspective* permet modificar els botons de la barra d'eines, comandes dels menús, ...
- *Show view*: Mostrar plafons de la perspectiva, molt útil si sense voler en tancam qualche un.
- *Preferences*: Ens permet, entre moltes altres opcions, canvar el tipus de lletra i el color de l'editor, o com es resalten les paraules clau, els comentaris, ...
- ...

Una particularitat molt útil és que eclipse ens permet veure el codi de més d'un fitxer a la vegada. Per fer-ho hem d'arrossegar la pestanya del fitxer en qüestió i arrosegar-la per la pantalla. Apareixeran unes línies que ens indiquen com quedarà configurada. En tenir-ho com volem, amollam el fitxer i quedarà configurada d'aquesta manera. Fins i tot es pot treure un fitxer de la finestra de windows, per exemple per posar-lo a la pantalla auxiliar.

Java - Proves/src/Primera.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access Java EE Debug Java

Pack

- Proves
- Proves2
- Tema2

Primera.java Exercici01.java

```
1
2 public class Primera {
3
4     /**
5      * @param args
6      */
7     public static void main(String[] args) {
8         System.out.println("Primera");
9     }
10
11 }
12
```

Segona.java

```
1 package exercicis;
2
3 public class Segona {
4
5     public static void main(String[] args) {
6         System.out.println("Segona");
7     }
8
9 }
10
11
```

Tercera.java

```
1 package exercicis;
2
3 public class Tercera {
4
5     public static void main(String[] args) {
6         //TODO: Completar el main
7         System.out.println("Segona");
8         int a=3;
9     }
10
11 }
```

Problems @ Javadoc Declaration Console Call Hierarchy Tasks

No consoles to display at this time.

Writable Smart Insert 8:39

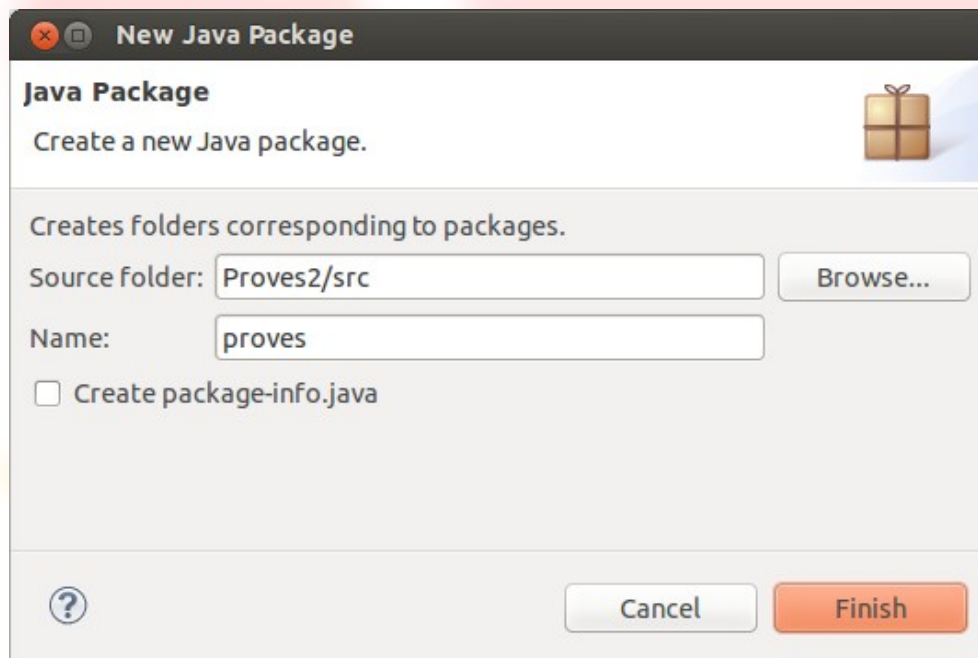


Package

Els paquets, en projectes grossos, permeten ordenar les distintes classes que formen el projecte. Segons l'estàndard de Java totes les classes haurien d'estar dins un paquet. Per això en crearem un dins el nostre projecte.

Ens situam a sobre del projecte, o de la carpeta *src*, a l'explorador i pitjam el botó dret *New->Package*.

Posam el nom del paquet, en minúscules, i pitjam *Finish*.



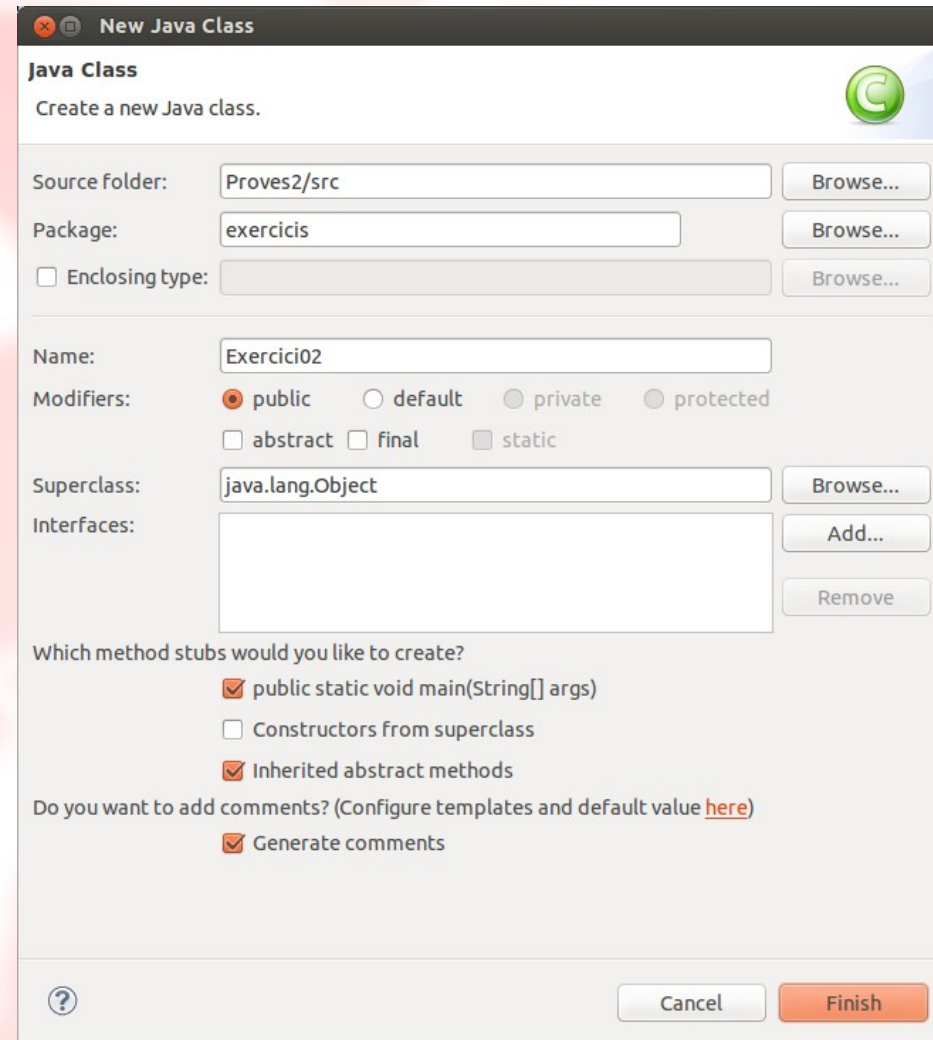
Classe

Qualsevol codi que volguem programar en Java haurà d'estar dins una classe. Per crear una classe ens situam a sobre del package que la contindrà i pitjam el botó dret *New->Class*.

El nom de la classe sempre ha de començar amb majúscula. Vegeu l'inici del capítol de sintaxi.

Si volem que la classe es pugui executar haurà de tenir el mètode *main*. De moment totes les nostres classes el tendran. Marcarem la casella de l'assistent.

Podem posar que generi els comentaris i aprofitar-los per posar el nostre nom i l'enunciat de l'exercici. Pitjam el botó *Finish* i tendrem la classe creada.



New Java Class

Create a new Java class.

Source folder: Proves2/src Browse...

Package: exercicis Browse...

☐ Enclosing type: Browse...

Name: Exercici02

Modifiers: ☒ public ☐ default ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass: java.lang.Object Browse...

Interfaces: Add... 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

? Cancel Finish

Execució

Amb eclipse no fa falta compilar explícitament el codi font. Cada vegada que es guarda el fitxer es compila.

Per executar una classe només ens hem de situar dins el codi de la classe i pitjar el botó verd amb una fletxa blanca dins de la barra d'eines d'eclipse. Això compilarà i executarà la classe.

Eclipse permet definir configuracions d'execució, per poder executar el projecte en distintes condicions. De moment no ens farà falta utilitzar-les. Algunes de les parametritzacions que ens permet:

- Elegir la classe que s'executarà en iniciar el projecte.
- Amb quina màquina virtual s'executarà si n'hi ha més d'una instal·lada
- Quins arguments es passen a l'aplicació des de la línia de comandes,
- ...

Connectors

Per si no ens basta tot el que duu eclipse es pot ampliar la seva funcionalitat afegint connectors, en anglès *pluggins*. Podem instal·lar els que volguem, però hem de tenir en compte que la instal·lació de connectors penalitza el rendiment de l'aplicació, sobre tot augmenta el temps d'arrencada.

Per exemple, una de les coses que ens poden fer falta és un dissenyador visual de pantalles:

Al menú *Help* trobarem l'opció *Eclipse marketplace*. Es tracta d'un lloc web des d'on cercar els connectors que ens interessin. Cercam Windows i recorrem la llista fins trobar el *WindowBuilder Pro Designer*.

Picam el *learn more* i ens mostrarà el lloc web del connector. Si clicam a sobre la fletxa verda de descàrregues ens mostrarà una url. La copiam sencera.

Anam al menú *Help->Install new software*. Enganxam l'url i pitjam el botó *Add*. Posam un nom que ens indiqui que representa aquesta url i pitjam OK.

Seleccionam *Window Builder Engine* i *Swing designer* i pitjam *Next*. Tornam a pitjar *Next*, acceptam les llicències i *Finish*. Una vegada reiniciat eclipse ja el podem utilitzar.