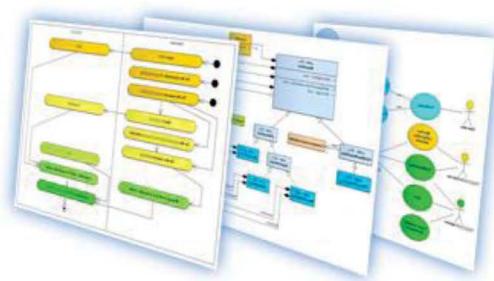


Tema 2: Entornos de Desarrollo

Tema 2: Entornos de Desarrollo
Entornos de Desarrollo

Alejandro Cardo Grau



Entornos de Desarrollo

T2: Entornos de Desarrollo

ÍNDICE

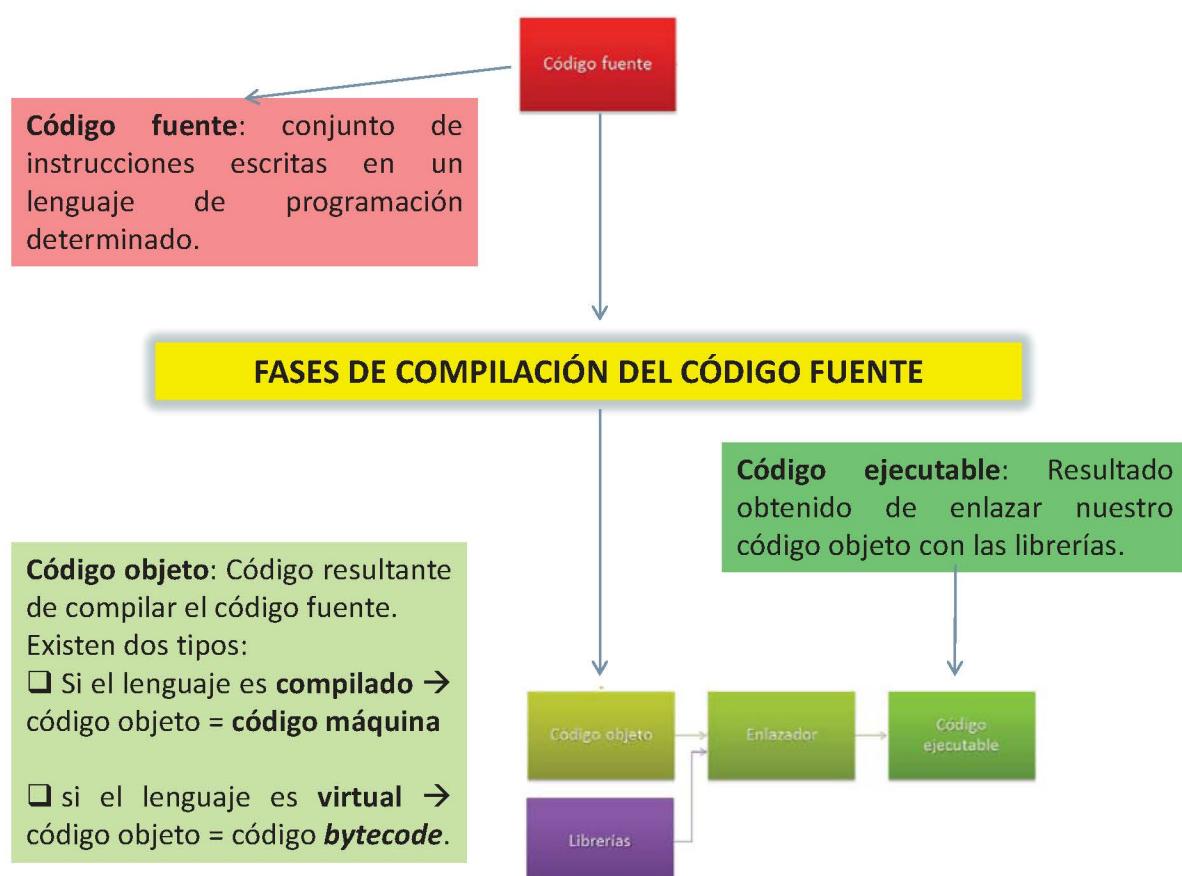
CRONOLOGÍA 1º TRIMESTRE

		Capítulos del Libro	
BLOQUE I: INTRODUCCIÓN A LOS ENTORNOS DE DESARROLLO	T1: El Programa Informático	CAPÍTULO 1. DESARROLLO DE SOFTWARE	Pendiente para el 2º trimestre: 1.6 ARQUITECTURA DE SOFTWARE 1.6.1 Patrones de desarrollo 1.6.2 Desarrollo en tres capas
BLOQUE II: ANÁLISIS Y DISEÑO DE PROYECTOS	T2: Entornos de Desarrollo	CAPÍTULO 2. INSTALACIÓN Y USO DE ENTORNOS DE DESARROLLO	
	T3: Ingeniería del Software	CAPÍTULO 1. DESARROLLO DE SOFTWARE	
	T4: Modelado del Proceso	CAPÍTULOS 5 y 6. DISEÑO ORIENTADO A OBJETOS. DIAGRAMAS DE COMPORTAMIENTO	5.1 INTRODUCCIÓN A UML 6.1 Tipos y campo de aplicación 6.2 Diagramas de actividad
	T5: Modelado de Análisis de Requisitos	CAPÍTULO 6. DISEÑO ORIENTADO A OBJETOS. DIAGRAMAS DE COMPORTAMIENTO	6.3 Diagramas de casos de uso

Alejandro Cardo Grau

Entornos de Desarrollo

1. Funciones de un Entorno de Desarrollo. Licencias.
2. Componentes y Tipos de Entornos de Desarrollo
3. Instalación y Configuración de un Entorno de Desarrollo
4. Uso Básico del Entorno de Desarrollo Netbeans y Eclipse
 1. Manejo básico del IDE. Tipos de proyectos.
 2. Vistas, paneles de navegación, editores y perspectivas. Personalización.
 3. Herramientas para la compilación, ejecución y generación.
 4. Componentes de la aplicación, creación y distribución del empaquetado.
5. Manejo de Herramientas para la Depuración
6. Integración de Plugins y Herramientas CASE en el Desarrollo



T2: Entornos de Desarrollo

REPASO

- **Herramientas CASE para el modelado UML:**

- **Aplicación:**

- Las herramientas de ingeniería de software asistida por computadora (CASE), ayudan al proceso del desarrollo del proyecto software durante todos los pasos del ciclo de vida.

- **Herramientas CASE UML:**

- **Integradas al IDE:**

- Eclipse:
 - ArgoEclipse
 - UML2 Tools SDK
 - Más plugins: <http://marketplace.eclipse.org/search/site/uml>
- Microsoft Visual Studio .NET
- NetBeans IDE UML

- **Independientes:**

- Visual Paradigm
- Enterprise Architect
- Microsoft Visio
- ArgoUML
- UMLPad

- **Generadores Online de Diagramas (no categorizadas como CASE):**

- [YUML](#)
- [WebSequenceDiagram](#) (Diagramas de secuencia)

Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

REPASO

Features [edit]

Name	UML 2	MDA	XMI	Templates	Languages generated	Reverse engineered languages <small>[clarification needed]</small>	Can be integrated with	Details
AgileJ StructureViews	No	No	Custom reverse-engineered class-diagrams — Java/Eclipse/Agile.	Unknown	Unknown	Java	Eclipse	Batch production of diagrams, Emphasis on filtering, Diagram tailoring while viewing in a browser
Altova UModel	Yes	Yes	Yes	Yes	Java, C#, Visual Basic	Java, C#, Visual Basic	Eclipse, Visual Studio	Also supports business process modeling, SysML, and database modeling
ArgoUML	No	Unknown	Yes	Unknown	C++, C#, Java, PHP4, PHP5, Python, Ruby	Java (other languages with plugins)	Unknown	Closely follows the UML standard
Artisan Studio	Yes	Yes	Yes	Yes	Ada, C, C++, C#, Java, IDL, SQL, VB	Mathworks Simulink, DOORS, Microsoft Word/Excel		Runs live on a highly scalable, multi-user database. UML, SysML & UPDM modeling. Diagram template driven code synchronization.
astah*	Yes	Unknown	Yes	Unknown	Java, C++, C#	Java, C++, C#		Mind Mapping, ER Diagram, DFD, Flowchart, CRUD, Traceability Map, Requirement Diagram and Requirement table. Provides API and Plugins, RTF, HTML Export.
ATL	Yes	No	Yes	No	Unknown	Unknown	Available from the Eclipse M2M project (Model to Model).	Can transform UML & EMF models into other models. It has a repository of transformations called ZOO about a large set of common industrial concerns and educational labs.
Borland Together	Yes	Yes	No	Yes	Java 6, C++, CORBA	Unknown	Eclipse and MS VS.NET 2005	
BOUML	Yes	Yes	Yes	Yes	C++, Java, PHP, IDL, Python	C++, Java, PHP	Unknown	Solid code roundtrip, fast
CaseComplete	Unknown	Unknown	Export	Unknown	Unknown	Unknown	Unknown	Provides management and editing of use cases, their flow of events, and referenced requirements. Supports use case and activity diagrams.
Creately for UML	Unknown	Unknown	Unknown	Unknown	Unknown	Unknown	Unknown	

Comparativa de Herramientas de Modelado UML (2013)

http://en.wikipedia.org/wiki/List_of_UML_tools

Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

1. Funciones de un Entorno de Desarrollo. Licencias.

- **Entorno de Desarrollo (*Integrated Development Environment*):**

- Aplicación informática compuesta por un conjunto de herramientas de programación que facilitan la tarea al programador y obtienen mayor rapidez en el desarrollo.
- Tiene el objetivo de asistir al programador en la difícil tarea de diseñar y codificar un software mediante la inclusión de múltiples herramientas destinadas para dichas tareas.
- Puede ser pensada para el uso de un lenguaje de programación o incluso la integración de varias tecnologías que se manejen.

- **Los IDE's producen artefactos:**

- Elemento que el proyecto produce o usa mientras se trabaja en busca del producto final.
- Muchas de las actividades realizadas en el desarrollo de proyectos en los IDE's están orientadas a:
 - Obtener códigos fuentes, ejecutables, enlazadores de los códigos objetos, manuales técnicos del código fuente, pruebas, etc.

Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

1. Funciones de un Entorno de Desarrollo. Licencias.

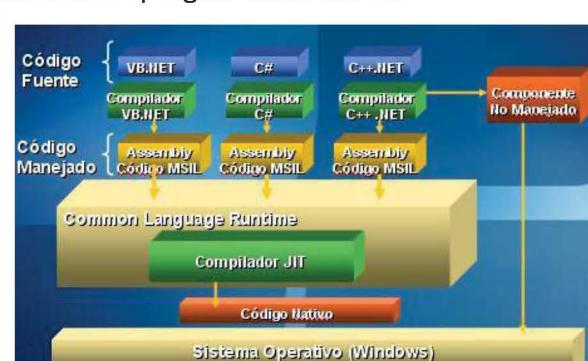
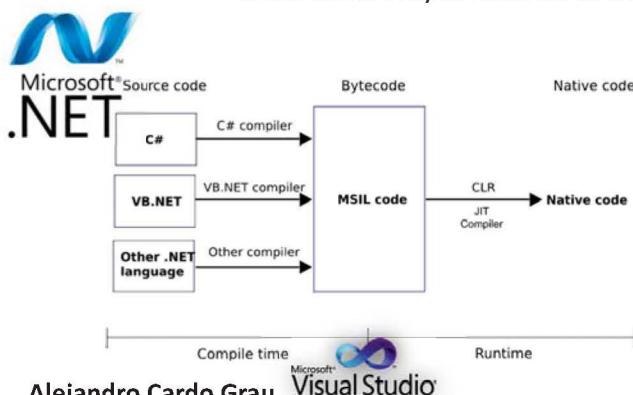
- **Los IDE's usan frameworks:**

Módulo o parte funcional de un sistema
Ejemplo: Plugins IDE

- Es un conjunto de **componentes** que cooperan entre sí y permiten reutilizarse para desarrollar software.
- Parten de un conjunto diseñado específicamente para ser extendido y no como software final.
 - Normalmente se componen de interfaces de comunicación (API).

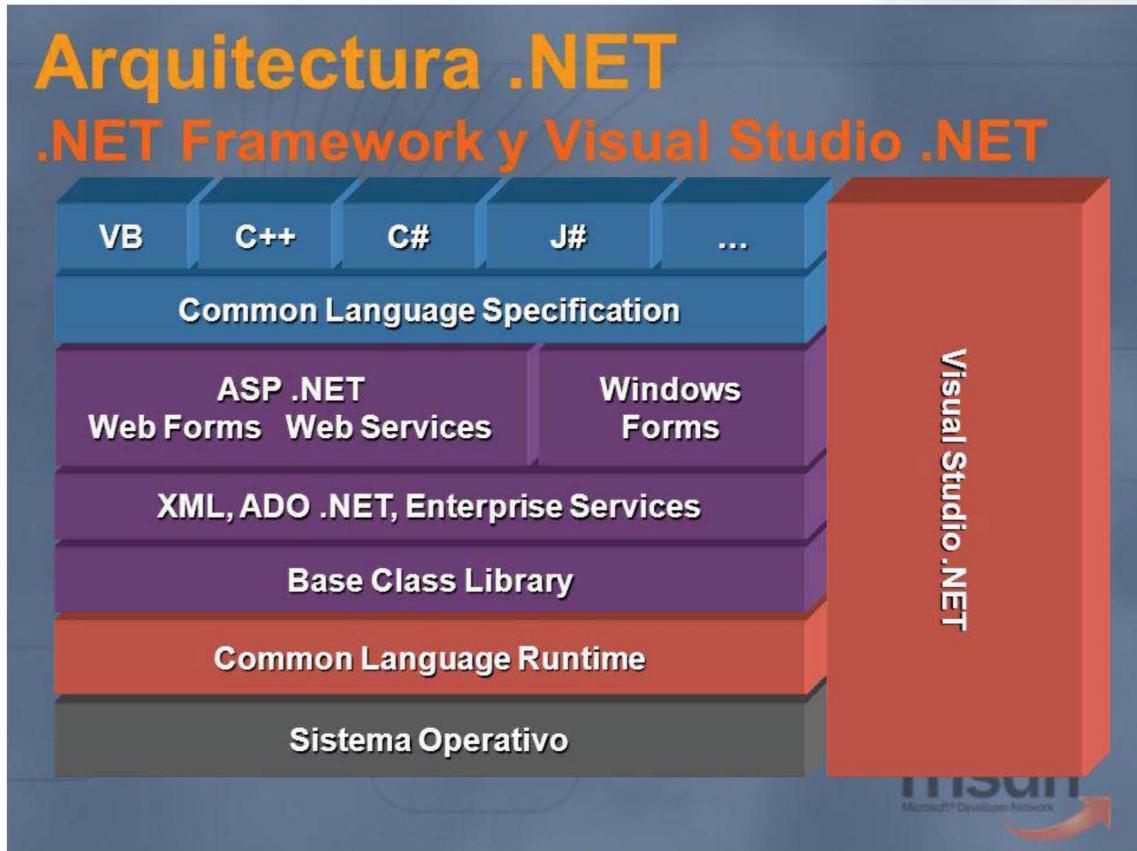
- **Los IDE's usan lenguajes de programación:**

- Están dedicados a un determinado lenguaje de programación, pero puede manejar incluso varios:
 - Las últimas versiones de los IDEs tienden a ser compatibles con varios lenguajes de programación (por ejemplo, Eclipse, NetBeans, Microsoft Visual Studio...) mediante la instalación de plugins adicionales.



Alejandro Cardo Grau

Entornos de Desarrollo

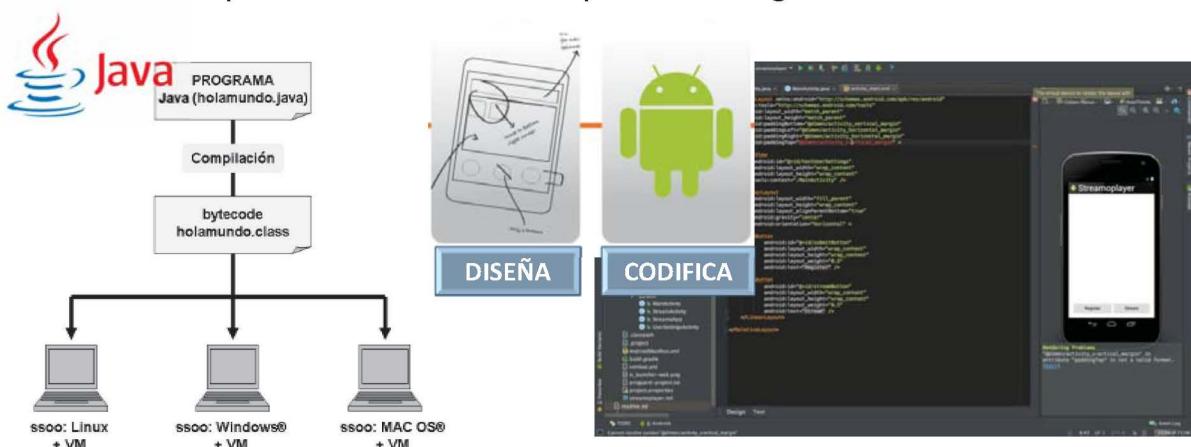


Alejandro Cardo Grau

Entornos de Desarrollo

- **Codificación:**

- El programador recibe las especificaciones del diseño y las transforma en un conjunto de instrucciones escritas (código fuente) usando para la implementación del proyecto:
 - Los lenguajes de programación.
 - Los entornos de desarrollo necesarios y sus compiladores asociados.
 - Los *plugins*, librerías, frameworks y módulos externos requeridos.
 - Las plataformas o sistemas operativos dirigidos en el desarrollo.



Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

1. Funciones de un Entorno de Desarrollo. Licencias.

- **Pruebas:** Buscan comprobar la calidad y estabilidad del código realizado:
 - Confirmando que la codificación ha sido exitosa y que no hay errores.
 - Comprobando que el software hace lo que debe hacer.
- **Documentación:** Debe mostrar una información completa y de calidad para ejecutar y manejar la aplicación final.
 - Los IDE's facilitan la generación de la:
 - **Documentación dirigida al equipo técnico:**
 - Documentación técnica de desarrollo para su codificación e integración.
 - Generador automatizado de documentación: Herramienta que genera la documentación de un software en un formato específico (generalmente en HTML) a partir de los comentarios embebidos del código, el IDE facilita:
 - 1) Analiza el código fuente.
 - 2) Extrae los comentarios.
 - 3) Les da el formato deseado.
 - 4) Genera una salida que pone a disposición del usuario.

Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

1. Funciones de un Entorno de Desarrollo. Licencias.

- **Historia de los IDE's:**
 - Los primeros entornos de desarrollo integrados nacen a principios de los 70 y se popularizan en la década de los 90.
 - Tienen el objetivo de ganar fiabilidad y tiempo en los proyectos de software.
 - Proporcionan al programador una serie de componentes con la misma interfaz gráfica, con la consiguiente comodidad, aumento de eficiencia y reducción de tiempo de codificación.
 - El primer lenguaje de programación que utiliza un IDE fue el BASIC en equipos informáticos de los años 70 y 80.
 - Éste primer IDE estaba basado en **terminal por consola de comandos** exclusivamente:
 - Con la aparición de sistemas operativos con GUI aparecen mejoras en los IDE con mejoras visuales incrementando su mejor gestión de archivos, compilación, depuración, etc.

Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

1. Funciones de un Entorno de Desarrollo. Licencias.

```

Amstrad Microcomputer (v4)
©1985 Amstrad plc
and Locomotive Software Ltd.
PARADOS V1.1. ©1997 QUANTUM Solutions.
PROTEXT word processor ©1985 Arnor Ltd.
MAXAM 1½ assembler ©1988 Arnor Ltd.
PhrozenC V1.0 ICCHELP for help
BASIC 1.1

Ready
lcc example1.c
EXAMPLE1.ASM
EXAMPLE1.C
PRINT C
EXAMPLE1.C (starting line 9)
0 errors.
Ready

```

IDE por terminal de comandos para el control del CPC-464 Amstrad
usando el lenguaje de programación Basic y ensamblador

Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

1. Funciones de un Entorno de Desarrollo. Licencias.



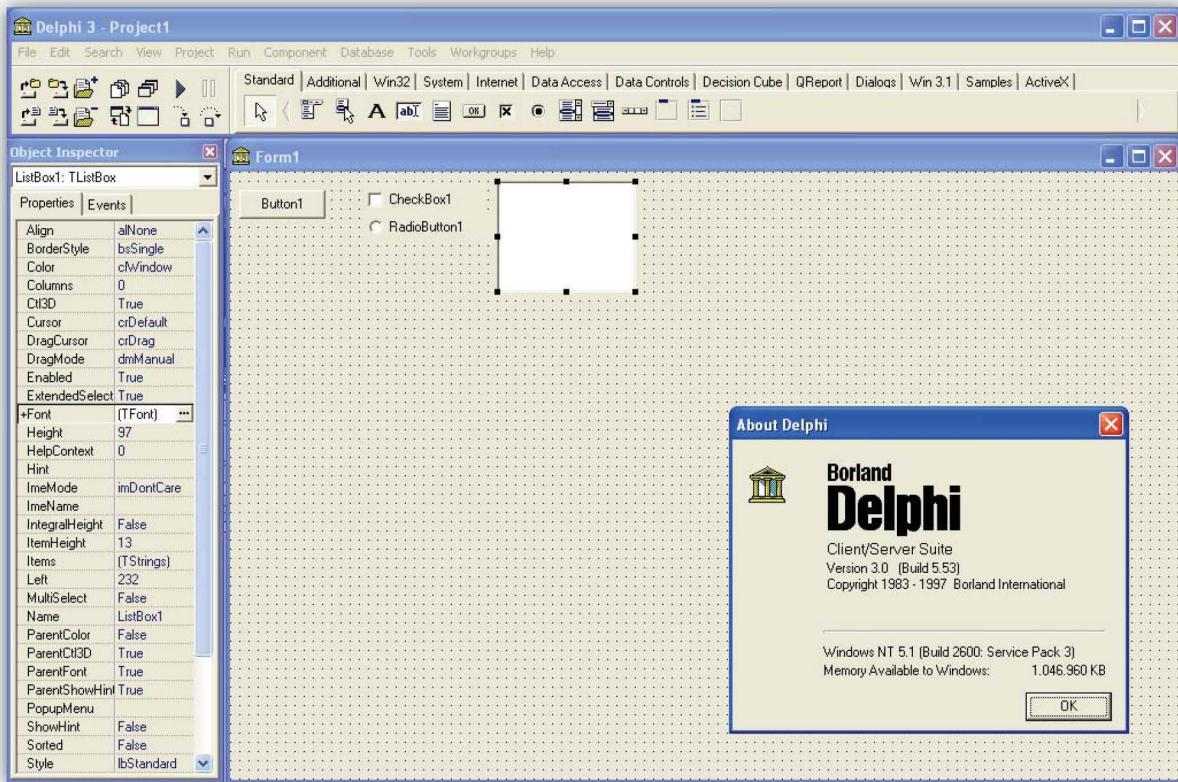
Borland Turbo C (IDE de los 80/90 diseñado para los lenguajes de programación C/C++)

Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

1. Funciones de un Entorno de Desarrollo. Licencias.



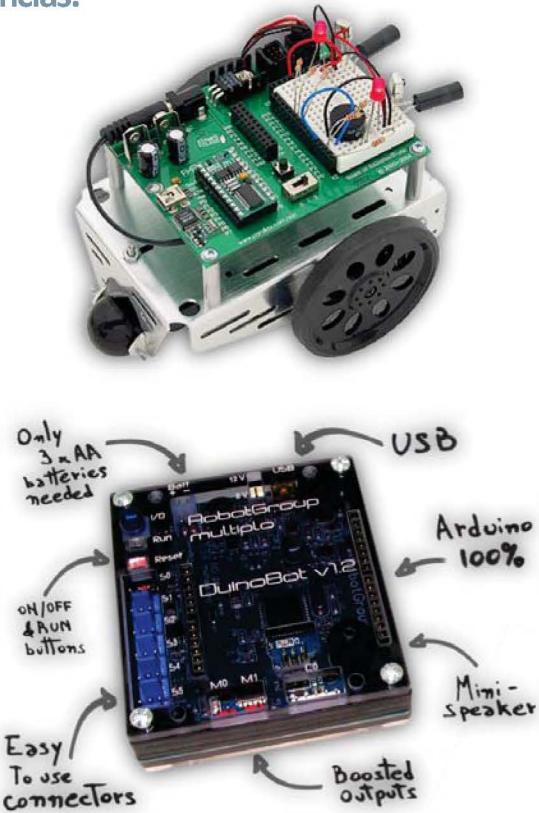
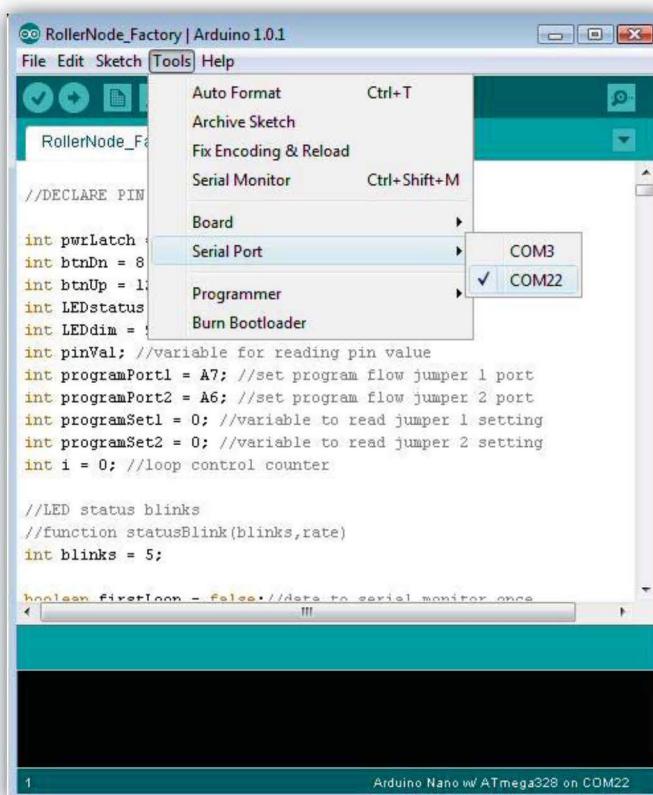
Borland Delphi (IDE con GUI de los 90 diseñado para el lenguaje Delphi, derivado de Pascal)

Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

1. Funciones de un Entorno de Desarrollo. Licencias.



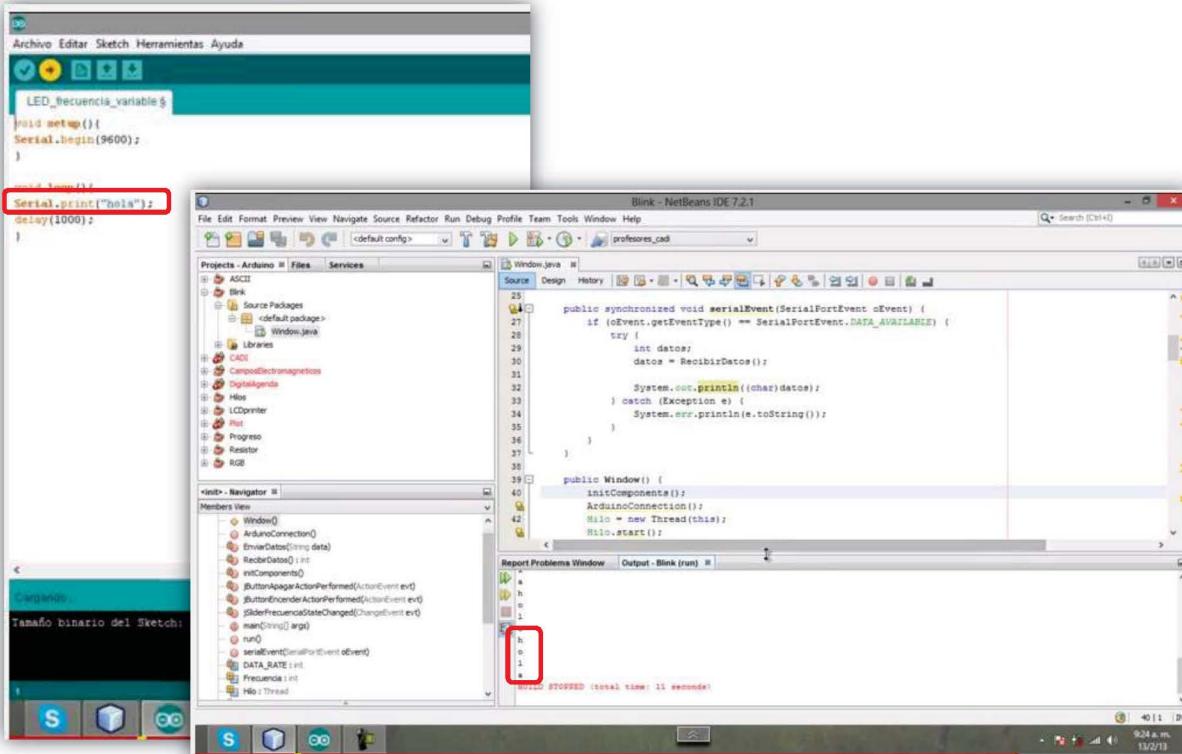
IDE para equipos empotrados de hardware libre Arduino usando lenguaje C

Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

1. Funciones de un Entorno de Desarrollo. Licencias.



Comunicación por puerto serie entre diferentes IDEs usando diferentes lenguajes de programación (C en Arduino IDE y Java en Netbeans):
https://www.youtube.com/watch?v=K_ifGhgt09k

Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

1. Funciones de un Entorno de Desarrollo. Licencias.

- **Funciones principales de un IDE:**
 - Edición del código (creación y modificación).
 - Interpretación directa del código en un lenguaje.
 - Mejoras del proceso de compilación y ejecución.
 - Autocompilación del código objeto a código máquina.
 - Enlazado-ensamblado de componentes.
 - Ejecución y despliegue aplicaciones.
 - Ejecutar el modo depuración del código.

Edición del código

Compilación y depuración del código

Ensamblado de componentes

Despliegue de aplicaciones

Soporte a varios lenguajes

Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

1. Funciones de un Entorno de Desarrollo. Licencias.

- **Otras funciones:**

- Detección de errores de sintaxis en tiempo real.
- Compilación de proyectos complejos en un solo paso.
- Analizar consistencia y calidad del código.
- Ejecución automática de pruebas.
- Control de versiones del código.
- Generar documentación del código.
- Refactorizar código.
- Dar soporte a varios lenguajes y *frameworks*.
- Dar soporte funcional mejorado con *plugins*.

Edición del código

Compilación y depuración del código

Ensamblado de componentes

Despliegue de aplicaciones

Soporte a varios lenguajes

T2: Entornos de Desarrollo

1. Funciones de un Entorno de Desarrollo. Licencias.



- **Licencias**

- La licencia del IDE que se elija para el desarrollo de un proyecto es una cuestión de vital importancia.
- En su elección prevalecerá la decisión de los supervisores del proyecto, de la dirección de la empresa y de las tecnologías que se usarán en el desarrollo.

- **Licencias del software libre en IDE's multiplataforma para Java:**

- Windows, Linux, MacOS y para arquitecturas de CPU de 32 y 64 bits
- Incluyen la Java Development Kit (JDK) y versiones con plugins

- **Eclipse**

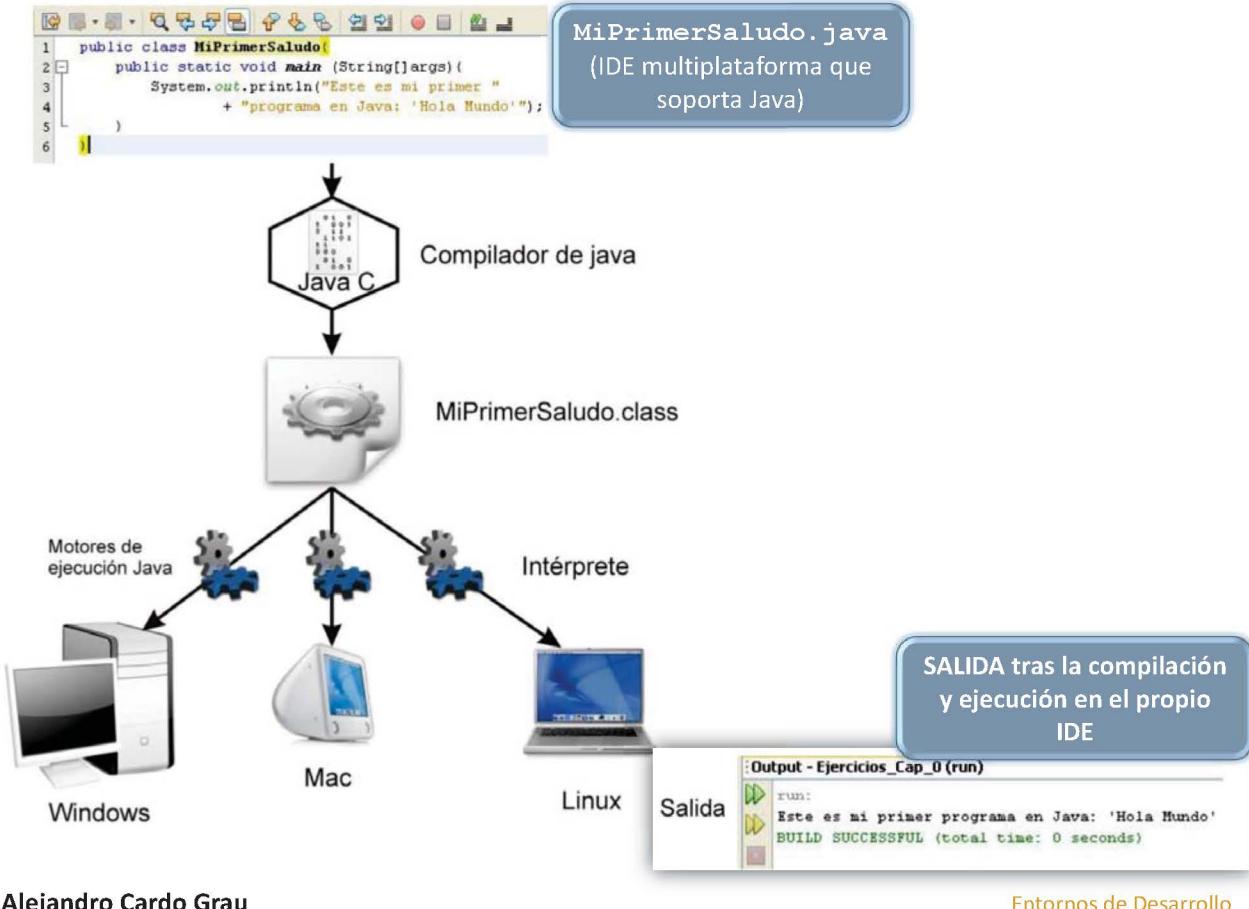
- Licencia del software libre: **Licencia Pública Eclipse (EPL)**
 - Licencia de software de código abierto utilizado por la Fundación Eclipse.
 - Enlace: <https://www.eclipse.org>

- **Netbeans:**

- Licencia del software libre: **Licencia CDDL + Licencia GNU GPL**
 - Enlace: <https://netbeans.org>

T2: Entornos de Desarrollo

1. Funciones de un Entorno de Desarrollo. Licencias.



T2: Entornos de Desarrollo

1. Funciones de un Entorno de Desarrollo. Licencias.

C and C++	Anjuta · Code::Blocks · CodeLite · Dev-C++ · Eclipse · GNAT Programming Studio · KDevelop · Kuzya · MonoDevelop · NetBeans · QDevelop · Qt Creator · wxDev-C++ · Ultimate++ · Pelles C · Sun Studio · Xcode · C++Builder · CodeWarrior · IBM VisualAge · Visual Studio (Express)
Java	Android Studio · BlueJ · Eclipse · JetBrains IntelliJ IDEA · Greenfoot · KDevelop · JBuilder · JCreator · JDeveloper · jGRASP · MyEclipse · NetBeans · IBM Rational Application Developer for WebSphere
.NET	MonoDevelop · SharpDevelop · Visual Studio (Express)

Comparativa de software IDE's del mercado (2014) (no están todos)

http://en.wikipedia.org/wiki/Integrated_development_environment

- **PREGUNTA 1 (TEST):**

- **Eclipse se define como:**

- 1) Entorno de desarrollo integrado (IDE) exclusivo para programar únicamente en Java.
- 2) Entorno de desarrollo no integrado (EID) para programación exclusiva en Java.
- 3) Informatización de desarrollo especializado (IDE) para Java.
- 4) Ninguna de las otras respuestas es correcta.

- **EJERCICIO 2 (FORO):**

- **Elabora un listado de IDE's usados para el desarrollo en otras plataformas ó sistemas operativos (p.e. MacOS, Linux, etc.):**

- Nombre del IDE
- Tipo de licencia del software
- Lenguajes de programación soportados (ya sea por plugins, etc.)
- Plataformas o sistemas operativos escogidos (puede ser multiplataforma)
- Web de descarga del IDE
- Captura de pantalla del IDE

- **EJERCICIO 3 (FORO):**

- 1) ¿Qué tipo de IDE es usado para generar apps para dispositivos móviles Android? ¿Y para dispositivos móviles iOS?
- 2) Busca los lenguajes de programación que son soportados por el framework .net usado en el IDE Visual Studio.
- 3) ¿Cómo es posible gestionar diferentes versiones del código en un mismo equipo de programadores usando el mismo IDE?
- 4) ¿Es posible desarrollar videojuegos para la consola XBox usando un IDE? ¿Qué tecnologías y/o lenguajes de programación se usan en su desarrollo?
- 5) Busca en Internet el significado de WIDE y detalla la relación de IDE's que poseen estas características. ¿Es necesario su instalación en un equipo informático?
- 6) Para la consola Playstation 4, detalla qué lenguajes de programación, IDE's y sistemas operativos son soportados para el desarrollo de videojuegos.

T2: Entornos de Desarrollo

2. Componentes y Tipos de Entornos de Desarrollo

- Componentes de los Entornos de Desarrollo:**

- Editor de textos:**

- Interpretación directa del código en un lenguaje.
 - Se resalta y colorea la sintaxis (palabras reservadas del lenguaje).
 - Función de autocompletado del código.
 - Ayuda en parámetros de métodos de la documentación de la API.
 - Inserción automática de instrucciones, paréntesis, corchetes, tabulaciones y espaciados.
 - Inserción e indentación automática del código en las estructuras del lenguaje (selectivas, repetitivas, etc.)

- Los IDE's ligeros suelen poseer únicamente el componente de Editor de Textos y el Generador Automático de Herramientas.**

Editor de Textos

Compilador
Intérprete

Depurador

Generador
Automático de
Herramientas

Interfaz Gráfica

Control de
Versiones

Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

2. Componentes y Tipos de Entornos de Desarrollo

- Componentes de los Entornos de Desarrollo:**

- Traductor: Compilador/intérprete:**

- Autocompilación del código objeto a código máquina.
 - Enlazado-ensamblado (*linker*) de componentes, paquetes y códigos objeto de la aplicación.
 - Detección de errores de sintaxis en tiempo real.
 - Analizar consistencia y calidad del código.
 - Características de refactorización del código.
 - Compilación de proyectos complejos en un solo paso.
 - Ejecución y despliegue aplicaciones.

- Los IDE's multiplataforma que soportan Java dependen del intérprete de la JVM como resultado.**

Editor de Textos

Compilador
Intérprete

Depurador

Generador
Automático de
Herramientas

Interfaz Gráfica

Control de
Versiones

Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

2. Componentes y Tipos de Entornos de Desarrollo

- **Componentes de los Entornos de Desarrollo:**
 - **Depurador (debugger):**
 - Ejecutar el **modo depuración** del código.
 - identificar y corregir errores de programación.
 - Examinar el código en tiempo de ejecución:
 - Paso a paso
 - instrucción a instrucción
 - Juegos de ensayo y casos de prueba
 - Botón de ejecución y traza, puntos de ruptura en instrucción (breakpoint) y seguimiento de variables
 - Opción de depurar por hilos de ejecución y en servidores de aplicaciones remotos.
 - **Generador automático de herramientas:**
 - Conjunto de asistentes y utilidades para la gestión, generación y mejora de los proyectos.
 - Ejecución automática de pruebas.
 - Generación de la documentación del código.
 - Complementado por otros plugins del IDE dependiendo del tipo de desarrollo aplicado.
 - Motor y despliegue de paquetes necesarios para la construcción de proyectos: Ant, Maven, etc.



Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

2. Componentes y Tipos de Entornos de Desarrollo

- **Componentes de los Entornos de Desarrollo:**
 - **Interfaz gráfica:**
 - Nos brinda la oportunidad de programar en varios lenguajes con un mismo IDE a partir de una interfaz visual.
 - Posee una GUI que puede acceder:
 - Control de bibliotecas y plugins
 - Opciones de configuración del IDE y de proyectos.
 - Modos de depuración del código
 - Paneles y vistas complementarias que informan de:
 - El estado de salida de la aplicación
 - Los componentes necesarios del proyecto: paquetes, librerías externas, etc.
 - Los controladores de bases de datos existentes.
 - La jerarquía de ficheros y directorios del proyecto.
 - Etc.



Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

2. Componentes y Tipos de Entornos de Desarrollo

- **Componentes de los Entornos de Desarrollo:**

- **Control de versiones:**

- Disponer de un **almacén de archivos compartido** por todos los colaboradores de los proyectos del desarrollo.
- Ante un error existen mecanismos de auto-recuperación a un estado anterior del código:
 - Versionado del código fuente y en el desarrollo
- Se establecen ramas o **branches** que determinan las líneas de evolución del código.
- Suelen ir integrados como *plugins* en los IDE.
- Ejemplos:
 - GIT
 - SVN
 - Github

Editor de Textos

Compilador
Intérprete

Depurador

Generador
Automático de
Herramientas

Interfaz Gráfica

Control de
Versiones

Entornos de Desarrollo

Alejandro Cardo Grau

T2: Entornos de Desarrollo

REPASO

- **PREGUNTA 1 (TEST):**

- **Un IDE está compuesto por:**

- 1) Editor de código y un tipo de traductor.
- 2) Interfaz gráfica, editor de código y un depurador.
- 3) Editor de código, traductor, depurador e interfaz gráfica.
- 4) Depurador e interfaz gráfica.

- **PREGUNTA 2 (TEST):**

- **¿Qué componente del IDE es necesario para unir archivos en la generación del ejecutable?:**

- 1) Únicamente el compilador.
- 2) Linker
- 3) Ensamblador
- 4) Intérprete

Alejandro Cardo Grau

Entornos de Desarrollo

2. Componentes y Tipos de Entornos de Desarrollo

- **Criterios de Selección en Tipos de Entornos de Desarrollo:**

- **Sistema operativo**

- (In)dependencia de la plataforma del IDE en el proyecto escogido:
 - Si estamos desarrollando aplicaciones para Linux, resultaría bastante inusual desarrollar la aplicación en Windows para ello.
 - Esto realmente no se debe a una restricción inherente al IDE, sino al compilador que tiene el IDE integrado (dependencia de plataforma).

- **Lenguaje de programación**

- Un IDE puede soportar uno o varios lenguajes de programación dependiendo del *framework* escogido:
 - Es necesario saber en qué lenguaje de programación vamos a codificar nuestro software para escoger un IDE u otro.

- **Herramientas y disponibilidad**

- Seguramente nos encontraremos con varios IDE que cumplen los requisitos de lenguaje y sistema operativo pero:
 - No todos tienen las mismas funciones, por lo que saber cuáles son esas herramientas y/o plugins que pueden mejorar el desarrollo de las aplicaciones.

INDICE

- 1. Funciones de un Entorno de Desarrollo. Licencias.**
- 2. Componentes y Tipos de Entornos de Desarrollo**
- 3. Instalación y Configuración de un Entorno de Desarrollo**
- 4. Uso Básico del Entorno de Desarrollo Netbeans y Eclipse**
 - 1. Manejo básico del IDE. Tipos de proyectos.**
 - 2. Vistas, paneles de navegación, editores y perspectivas. Personalización.**
 - 3. Herramientas para la compilación, ejecución y generación.**
 - 4. Componentes de la aplicación, creación y distribución del empaquetado.**
- 5. Manejo de Herramientas para la Depuración**
- 6. Integración de Plugins y Herramientas CASE en el Desarrollo**

INDICE

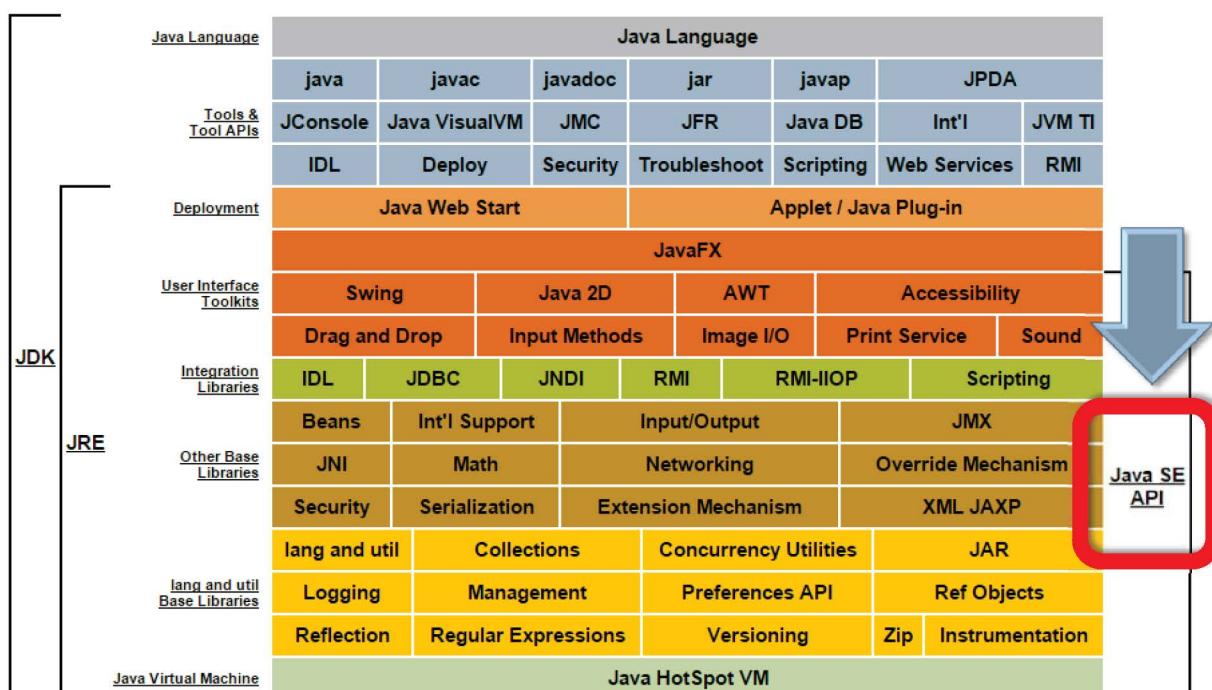
- 1. Funciones de un Entorno de Desarrollo. Licencias.**
- 2. Componentes y Tipos de Entornos de Desarrollo**
- 3. Instalación y Configuración de un Entorno de Desarrollo**
- 4. Uso Básico del Entorno de Desarrollo Netbeans y Eclipse**
 - 1. Manejo básico del IDE. Tipos de proyectos.**
 - 2. Vistas, paneles de navegación, editores y perspectivas. Personalización.**
 - 3. Herramientas para la compilación, ejecución y generación.**
 - 4. Componentes de la aplicación, creación y distribución del empaquetado.**
- 5. Manejo de Herramientas para la Depuración**
- 6. Integración de Plugins y Herramientas CASE en el Desarrollo**

Alejandro Cardo Grau

Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo**REPASO**

Java SE Development Kit Documentation:


<https://docs.oracle.com/javase/7>

Alejandro Cardo Grau

Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

REPASO

Java SE Development Kit Documentation (descarga offline) :

The screenshot shows the Oracle Java SE Development Kit Documentation page. At the top, there's a navigation bar with links for Account, Sign Out, Help, Country, Communities, I am a..., I want to..., Search, and a magnifying glass icon. Below that is another navigation bar with Products, Solutions, Downloads, Store, Support, Training, Partners, About, and OTN. A breadcrumb trail at the top indicates the path: Oracle Technology Network > Java > Java SE > Documentation. On the left, there's a sidebar with links for Java SE, Java EE, Java ME, Java SE Support, Java SE Advanced & Suite, Java Embedded, Java DB, Web Tier, Java Card, Java TV, New to Java, Community, and Java Magazine. The main content area has tabs for Overview, Downloads, Documentation (which is selected), Community, Technologies, and Training. The title "Java SE Development Kit 7 Documentation" is displayed. Below it, a box titled "Java SE Development Kit 7u71 Documentation" contains a message: "You must accept the Java SE Development Kit 7 Documentation License Agreement to download this software." It includes two radio buttons: "Accept License Agreement" (selected) and "Decline License Agreement". At the bottom of this box is a table with three columns: Product / File Description, File Size, and Download. The first row shows "Documentation" with a file size of "58.29 MB" and a download link labeled "jdk-7u71-docs-all.zip". To the right of the main content, there are two sections: "Java SDKs and Tools" and "Java Resources".

<http://www.oracle.com/technetwork/es/java/javase/documentation/java-se-7-doc-download-435117.html>

Alejandro Cardo Grau

Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.1. Manejo básico del IDE. Tipos de proyectos

- **IDE Netbeans:**
 - IDE OpenSource escrito en código Java y mantenido por Sun MicroSystems.
 - Licencia del software libre: **Licencia CDDL + Licencia GNU GPL**
 - Lanzado en junio de 2000, continúa siendo el patrocinador principal de los proyectos Oracle Sun Java.
 - Enlace: <https://netbeans.org>
 - Windows, Linux, MacOS y para arquitecturas de CPU de 32 y 64 bits
 - Incluyen la Java Development Kit (JDK) y versiones con plugins.
 - El IDE Eclipse comparte la **cuota de mercado multiplataforma** para el desarrollo Java.
 - Elegido para la asignatura por dos razones principales para las asignaturas del ciclo formativo:
 - Al estar mantenido por Sun los cambios en el lenguaje Java tienen su reflejo más inmediato en una actualización de versión de NetBeans.
 - Su estructura compacta (aunque permite el uso de plug-ins) la hace mas sencilla de utilizar para el programador que el IDE Eclipse.

Alejandro Cardo Grau

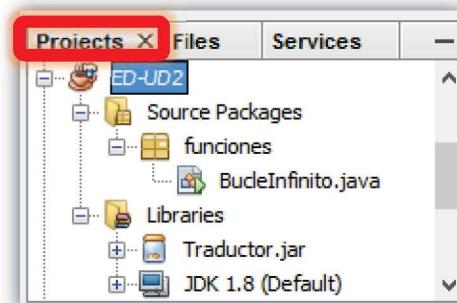
Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.1. Manejo básico del IDE. Tipos de proyectos

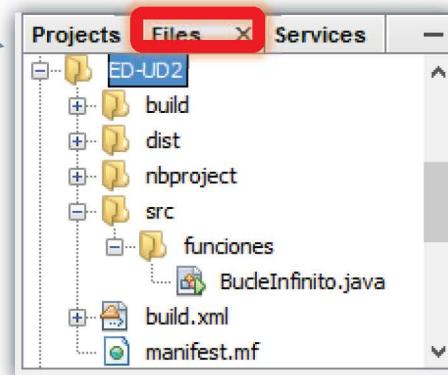
- **Tipos de Proyectos:**

- El IDE Netbeans, al igual que el IDE Eclipse, no trabaja a nivel de archivo sino a nivel de proyecto.



- Un proyecto incluye todos los **recursos necesarios** para construir la aplicación:

- Archivos generadores del proyecto
- Archivos con el código fuente
- Bibliotecas/Librerías externas
- Otros recursos: imágenes, sonidos y otros artefactos necesarios.



Alejandro Cardo Grau

Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

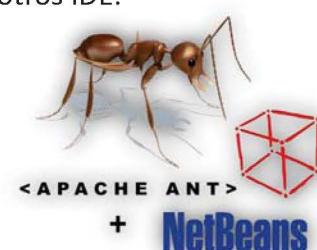
2.1. Manejo básico del IDE. Tipos de proyectos

- **IDE Netbeans:**

- El IDE provee de asistentes para empezar proyectos desde cero y/o para crear un proyecto con códigos fuentes ya existentes.

- El IDE NetBeans trabaja a nivel de proyecto:

- No se puede compilar el código fuente Java si no está integrado dentro de un proyecto Netbeans.
- El IDE Netbeans automatiza el uso y generación de los artefactos necesarios del proyecto de codificación basándose en una herramienta automatizada denominada **Ant** de Apache.
 - Es una herramienta usada en programación para la realización de tareas mecánicas y repetitivas, normalmente durante la fase de compilación, enlazado y construcción (build), usadas también en otros IDE.



Alejandro Cardo Grau

Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.1. Manejo básico del IDE. Tipos de proyectos

- **IDE Netbeans:**

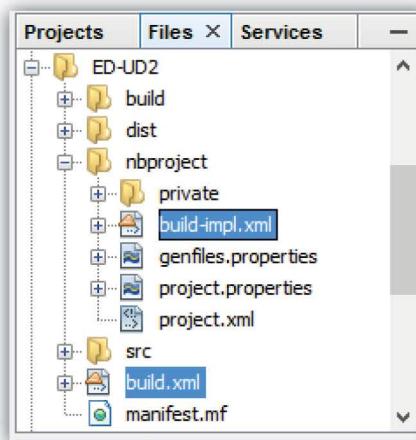
- NetBeans genera **dos archivos de Apache Ant**:

- **build.xml**

- Es el artefacto que se invoca cada vez que compila, enlaza, construye y ejecuta el proyecto de la aplicación Java desde el IDE Netbeans.
- Importa todas las tareas definidas en el archivo XML: build-impl.xml

- **nbproject/build-impl.xml**

- Donde se implementan todas las tareas automatizadas del proyecto (podemos incorporar también tareas complementarias a nuestro proyecto).



```
<?xml version="1.0" encoding="UTF-8"?>

<project name="ED-UD2" default="default" basedir=".">
    <description>Builds, tests, and runs the project ED-UD2.</description>
    <import file="nbproject/build-impl.xml"/>
```

build.xml

Alejandro Cardo Grau

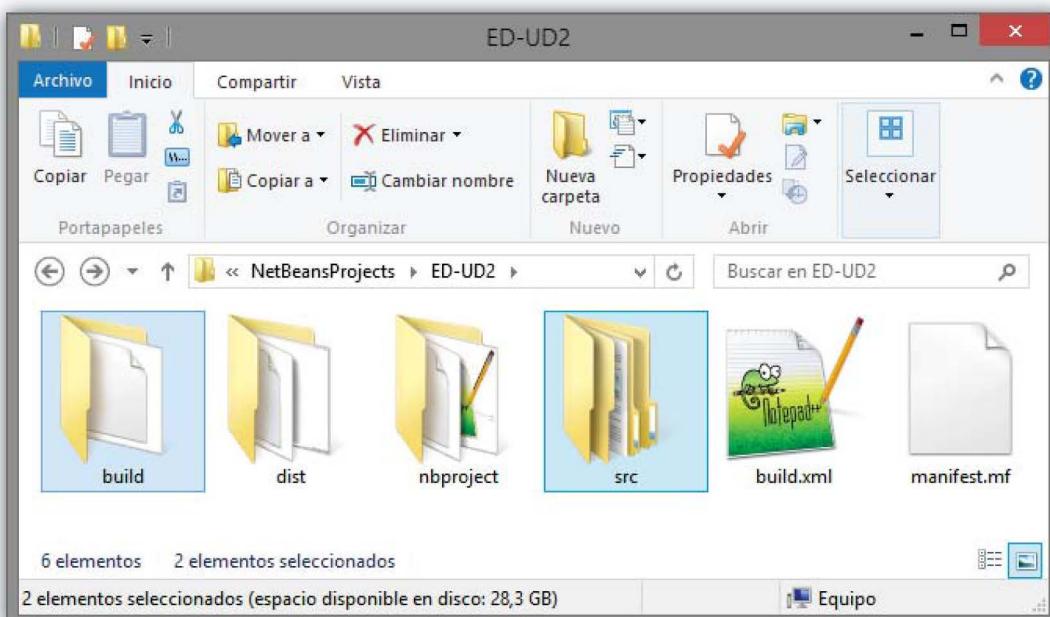
Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.1. Manejo básico del IDE. Tipos de proyectos

- **Estructura de subdirectorios de un proyecto Netbeans:**

- **src**: donde se incluyen los códigos fuentes (**ficheros .java**) (<dirs>)
- **build**: donde se sitúan los **ficheros .class** compilados. (<dirclass>)



Alejandro Cardo Grau

Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

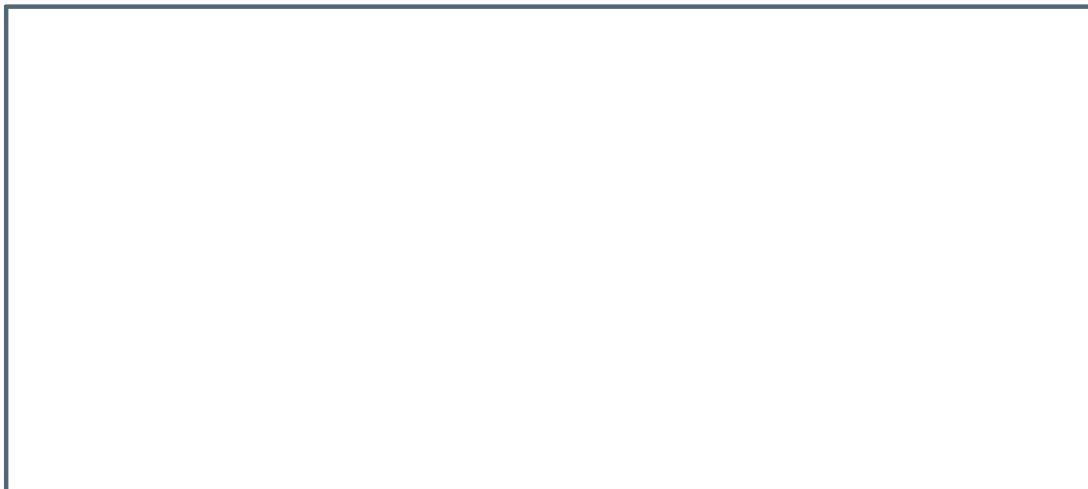
2.1. Manejo básico del IDE. Tipos de proyectos

- Procedimientos básicos para el IDE Netbeans:

- 1) Abrir con el editor de texto del IDE un fichero .java



- 2) Crear un nuevo proyecto Java



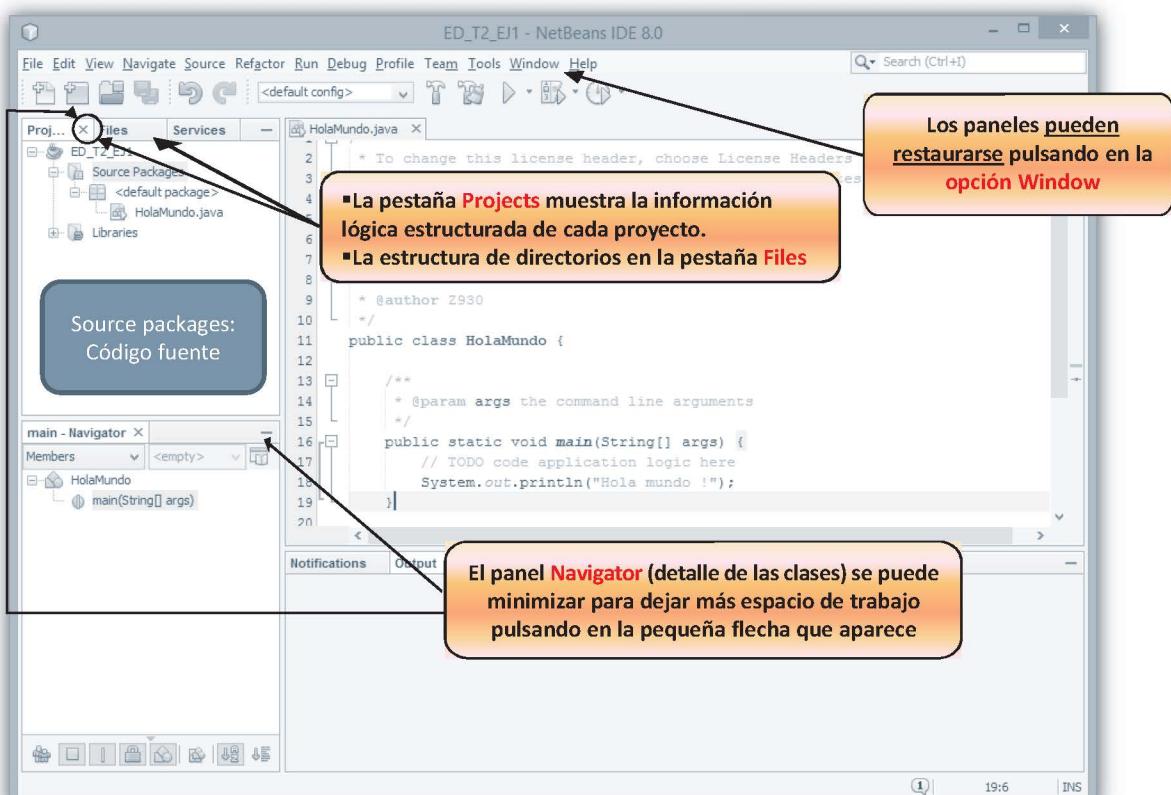
Alejandro Cardo Grau

Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.2. Vistas, paneles de navegación, editores y perspectivas. Personalización.

- Vistas y paneles de navegación:



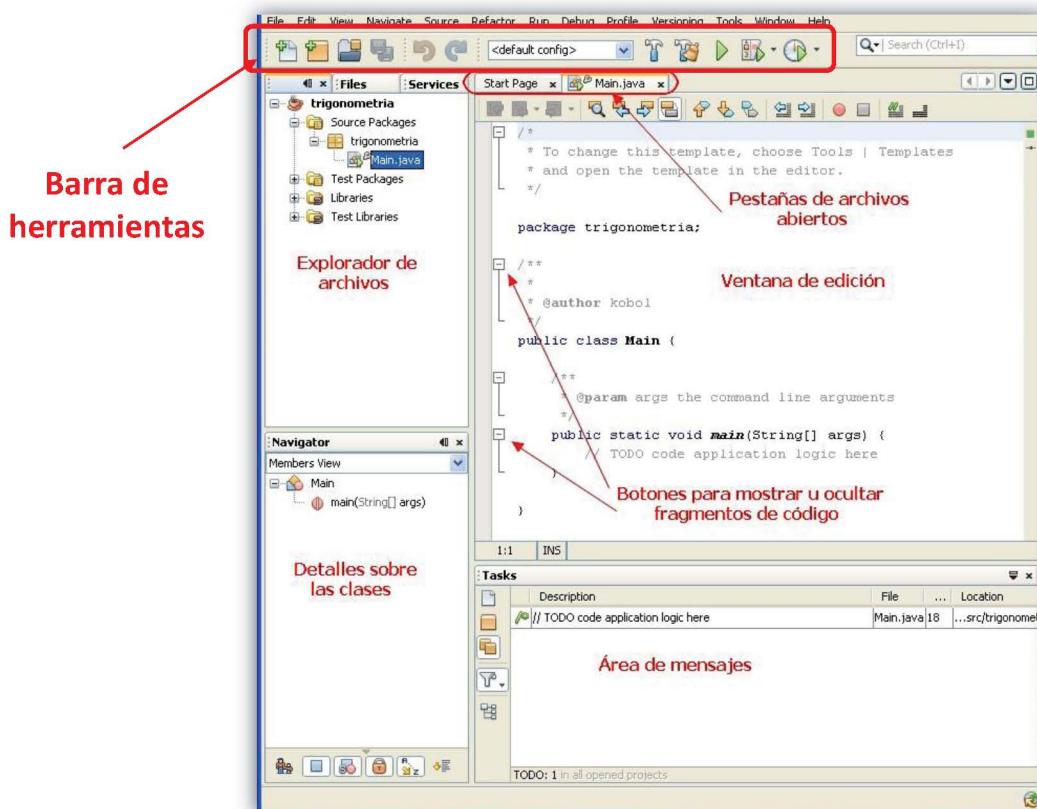
Alejandro Cardo Grau

Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.2. Vistas, paneles de navegación, editores y perspectivas. Personalización.

- **Vistas y paneles de navegación:**



Alejandro Cardo Grau

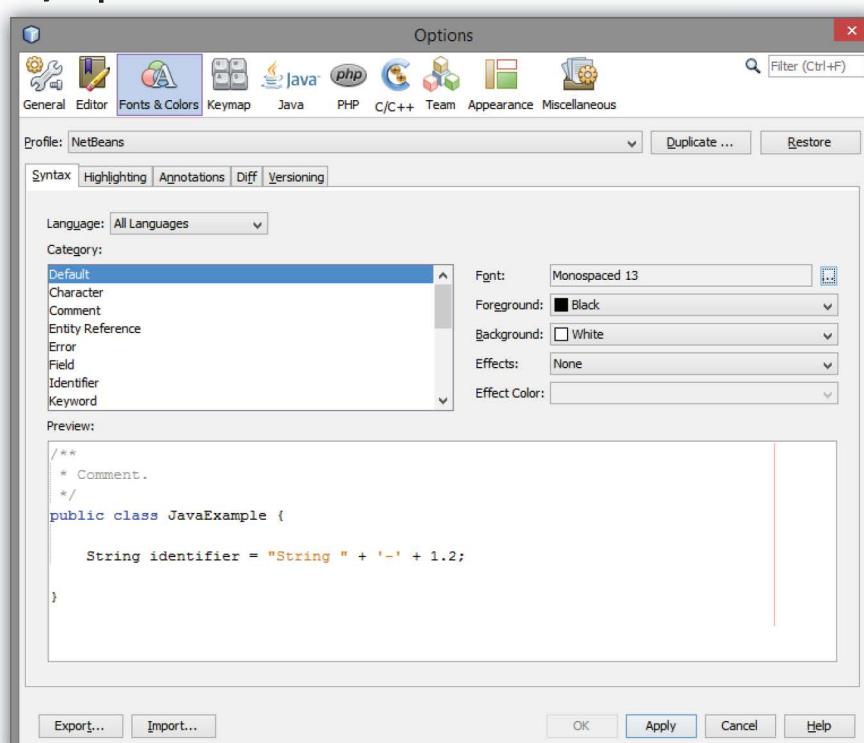
Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.2. Vistas, paneles de navegación, editores y perspectivas. Personalización.

- **Personalización del editor de código:**

- Tools / Options



Alejandro Cardo Grau

Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.2. Vistas, paneles de navegación, editores y perspectivas. Personalización.

- **Procedimientos básicos para el IDE Netbeans:**

- 3) Crear una nueva clase dentro del proyecto “ED_T2_EJ1”

- 1) Seleccionar File / New File
- 2) Seleccionar “Java Main Class”
- 3) Teclear “CalculaNumeros”
- 4) Pulsar “Finish”

- 4) Generar el siguiente código en CalculaNumeros . java

```

9  /*
10 */
11 public class CalculaNumeros {
12     /**
13      * @param args the command line arguments
14     */
15     public static void main(String[] args) {
16         // TODO code application logic here
17         int num1 = 12, num2 = 2;
18         System.out.println("SUMA: " + (num1 + num2));
19         System.out.println("RESTA: " + (num1 - num2));
20         System.out.println("MULTIPLICACIÓN: " + (num1 * num2));
21         System.out.println("DIVISIÓN: " + (num1 / num2));
22         System.out.println("RESTO: " + (num1 % num2));
23     }
24 }

```

Alejandro Cardo Grau

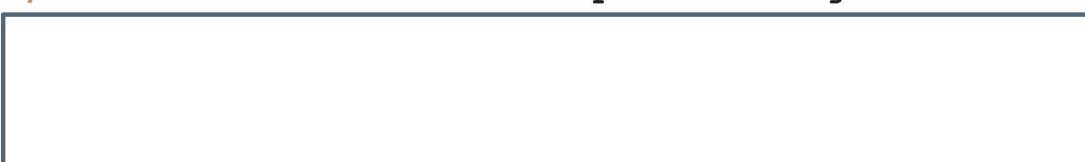
Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

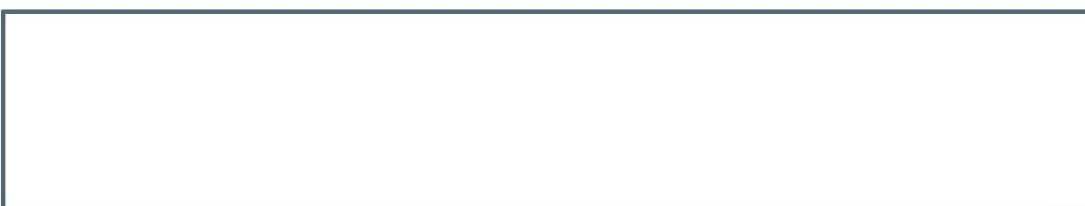
2.2. Vistas, paneles de navegación, editores y perspectivas. Personalización.

- **Procedimientos básicos para el IDE Netbeans:**

- 5) Modificar el nombre a: CalculaOperaciones . java



- 6) Copiar CalculaOperaciones . java a CalculaSumaResta . java



Projects X Files Services

ED_T2_EJ1

Source Packages <default package>

CalcularOperaciones.java CalcularSumaResta.java HolaMundo.java

Start Page X CalculaOperaciones.java X HolaMundo.java X CalculaSumaResta.java X

```

public class CalculaSumaResta {
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        int num1 = 12, num2 = 2;
        System.out.println("SUMA: " + (num1 + num2));
        System.out.println("RESTA: " + (num1 - num2));
    }
}

```

Alejandro Cardo Grau

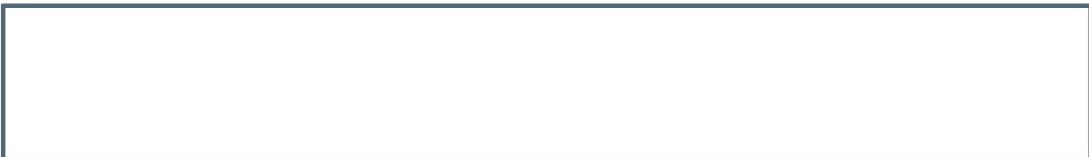
Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

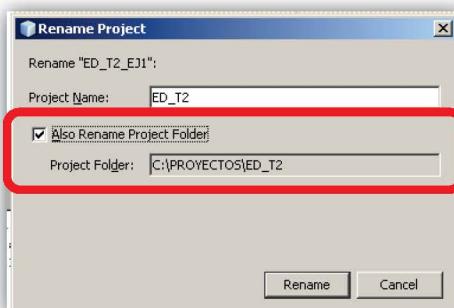
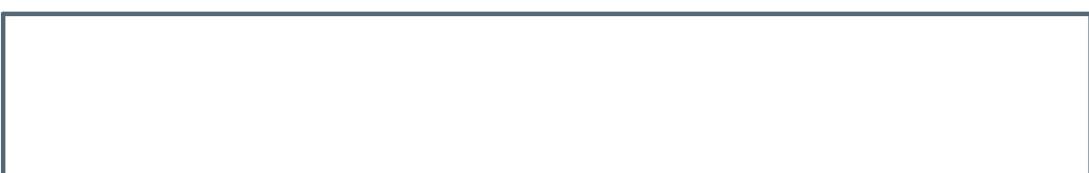
2.2. Vistas, paneles de navegación, editores y perspectivas. Personalización.

- Procedimientos básicos para el IDE Netbeans:

7) Cierra y abre el proyecto



8) Cambiar el nombre del proyecto a “ED_T2” y luego vuelvo a renombrarlo a “ED_T2_EJ1” (incluido directorios).

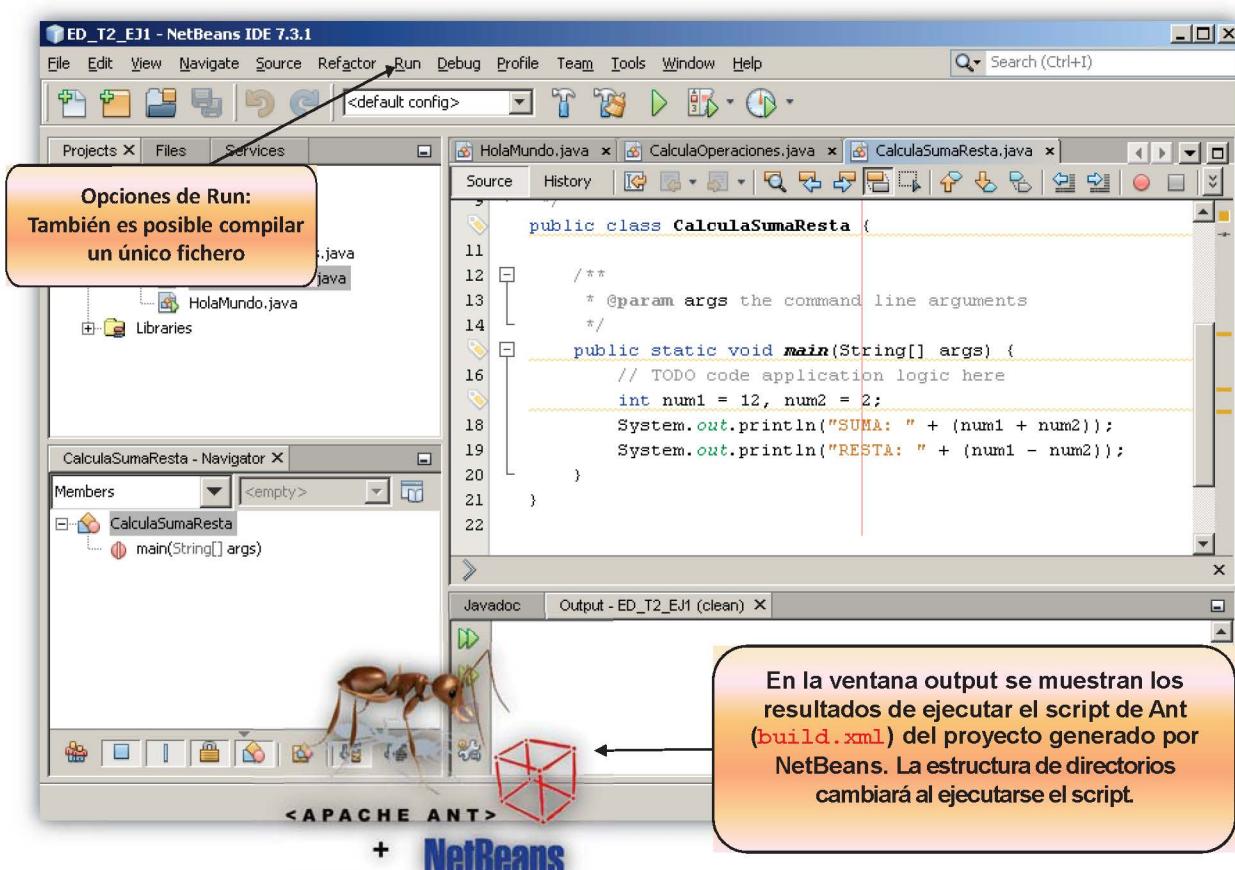


Alejandro Cardo Grau

Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.3. Herramientas para la compilación, ejecución y generación.



Alejandro Cardo Grau

Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

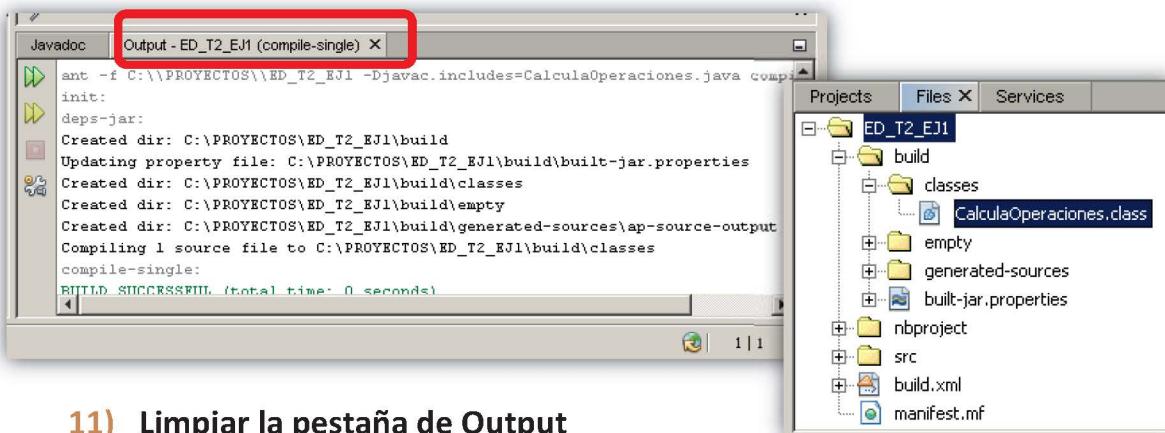
2.3. Herramientas para la compilación, ejecución y generación.

- Procedimientos de compilación sencilla de un único fichero:

9) Compila el fichero: `CalculaOperaciones.java` y comprueba en la pestaña de “Output” (*compile-single*) lo realizado:



10) Comprobar en la pestaña pestaña “Files” que se ha generado la compilación de: `CalculaOperaciones.java`



11) Limpiar la pestaña de Output

Alejandro Cardo Grau

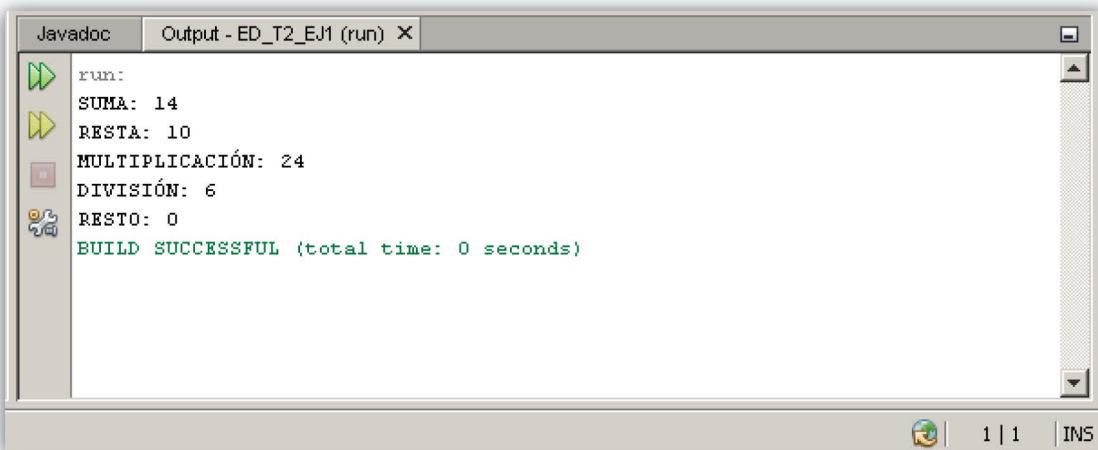
Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.3. Herramientas para la compilación, ejecución y generación.

- Procedimientos de ejecución sencilla de un único fichero:

12) Ejecuta el binario compilado generado:



Alejandro Cardo Grau

Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

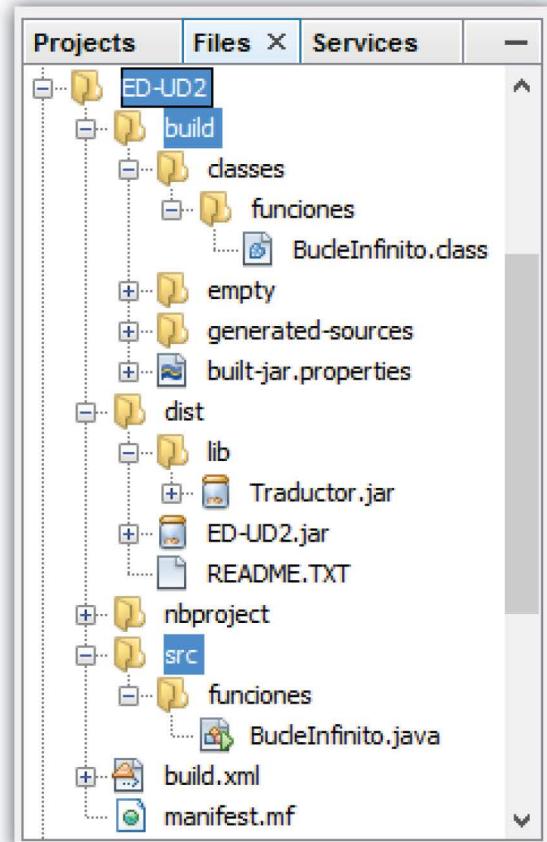
2.3. Herramientas para la compilación, ejecución y generación.

- Estructura de subdirectorios de un proyecto Netbeans:**

- **src:** donde se incluyen los códigos fuentes (**ficheros .java**) (<dirsrc>)
- **build:** donde se sitúan los **ficheros .class** compilados. (<dirclass>)

- Otros directorios:**

- **nbproject:** incluye la información y administración.
- **dist:** donde se sitúan el empaquetado .jar de la aplicación a desplegar y librerías/bibliotecas necesarias para la ejecución de la aplicación.
- **test:** donde se incluyen los códigos de los tests para pruebas de unidad.



Alejandro Cardo Grau

Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.3. Herramientas para la compilación, ejecución y generación.

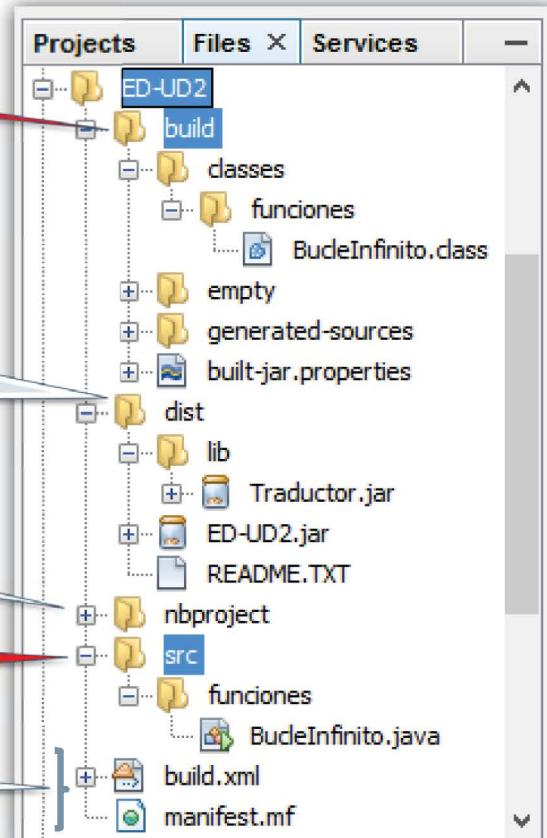
Ficheros compilados .class

Distribuciones del empaquetado (.jar) y librerías/bibliotecas necesarias para la ejecución de la aplicación Java

Uso interno para el IDE Netbeans

Código fuente de la aplicación (estructurados por paquetes)

Otros archivos auxiliares



Alejandro Cardo Grau

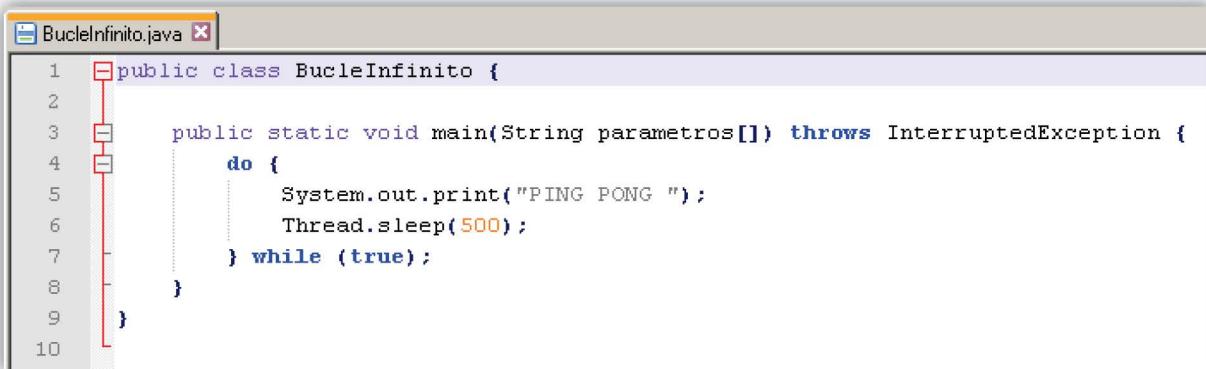
Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.3. Herramientas para la compilación, ejecución y generación.

- Procedimientos de ejecución sencilla de un único fichero:

13) Desarrolla en Notepad++ la siguiente clase, agregándola al proyecto:



```

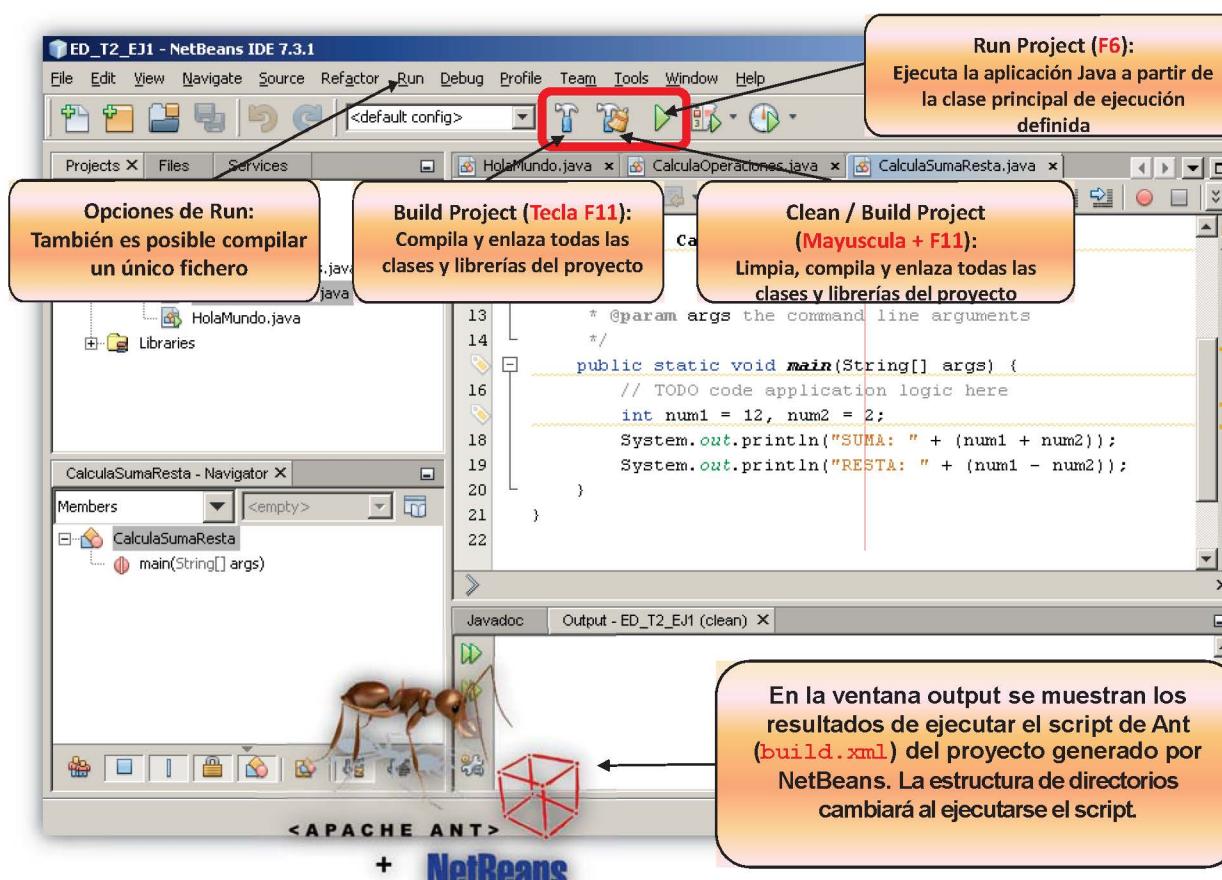
1 public class BucleInfinito {
2
3     public static void main(String parametros[]) throws InterruptedException {
4         do {
5             System.out.print("PING PONG ");
6             Thread.sleep(500);
7         } while (true);
8     }
9 }

```

14) Compila, ejecuta la clase desde el IDE y finaliza su ejecución.

2. Uso Básico del Entorno de Desarrollo

2.3. Herramientas para la compilación, ejecución y generación.



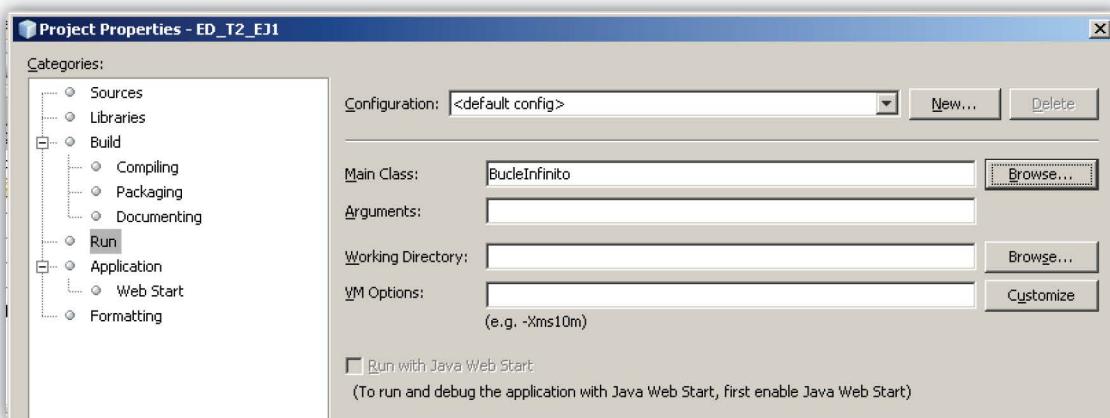
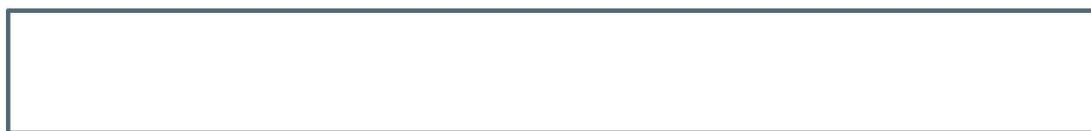
2. Uso Básico del Entorno de Desarrollo

2.3. Herramientas para la compilación, ejecución y generación.

- **Compilación, ejecución y generación del proyecto:**

- **Definir el main-class (clase principal de ejecución) del proyecto:**

- Usando las herramientas de generación del empaquetado del JAR Ejecutable en los IDE Eclipse o Netbeans, determinando la clase principal a ejecutar en la configuración del proyecto:



Alejandro Cardo Grau

Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.3. Herramientas para la compilación, ejecución y generación.

- **Compilación, ejecución y generación del proyecto:**

15) Seleccionar el main-class de `BucleInfinito.java`:

16) Realiza una operación de limpieza del proyecto y observa la estructura de subdirectorios que se han eliminado (pestaña “Files”).



17) Compila, ejecuta el proyecto y finaliza su ejecución.



Alejandro Cardo Grau

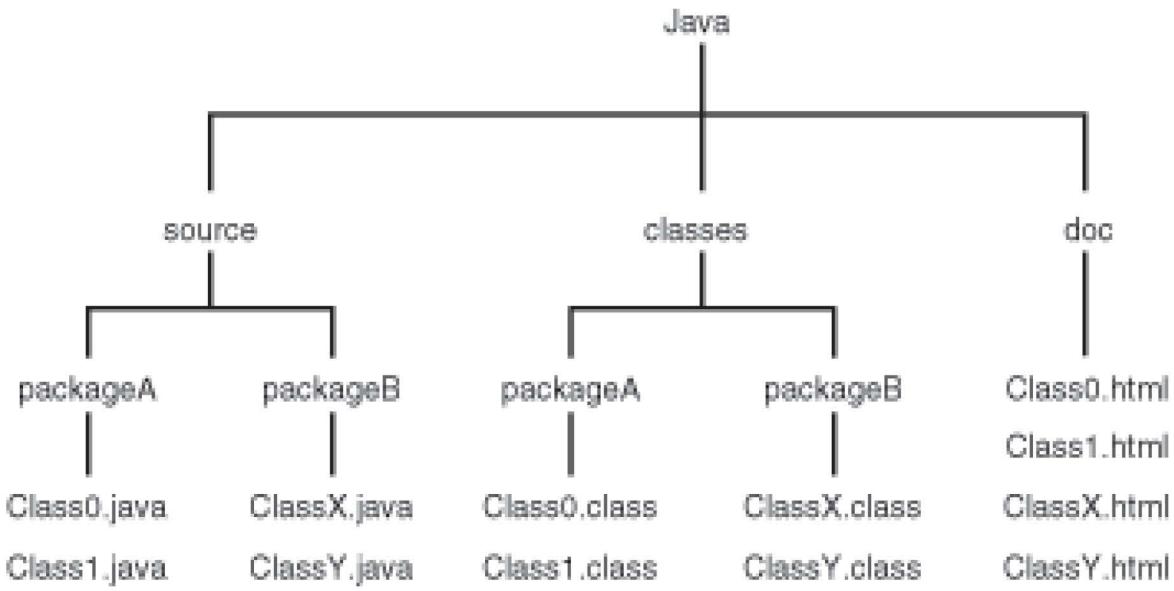
Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- **Componentes de la aplicación:**

- En Java se usa una organización jerárquica para gestionar los ficheros de código fuente y de clases compiladas.



2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- **Componentes de la aplicación:**



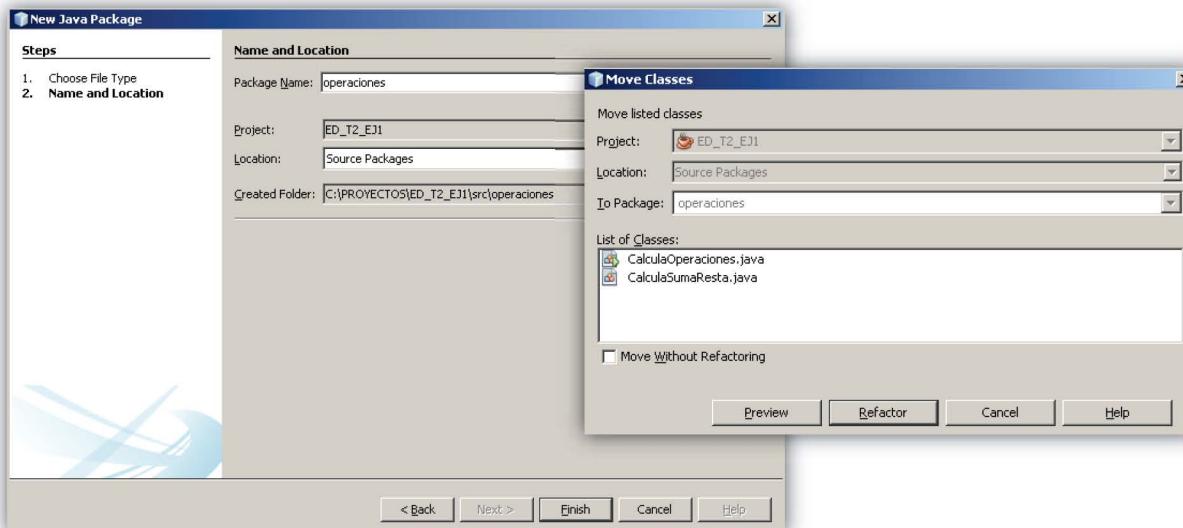
- Un paquete o *package* de Java es un conjunto de clases e interfaces relacionadas entre sí.
- Las clases e interfaces que forman parte de un package se **estructuran y organizan por funciones o tareas**.
- Cualquier programador puede introducir sus clases e interfaces en paquetes de la aplicación para facilitar:
 - El uso del desarrollo y reutilización en las aplicaciones desarrolladas.
- Por convención Java, el nombre de los paquetes debe estar totalmente en minúsculas.
- Para incluir una clase en un paquete debe incluirse la siguiente sentencia al principio del archivo fuente que contiene la clase:

```
package nombrepaquete;
```

2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- Creación de paquetes:



Alejandro Cardo Grau

Entornos de Desarrollo

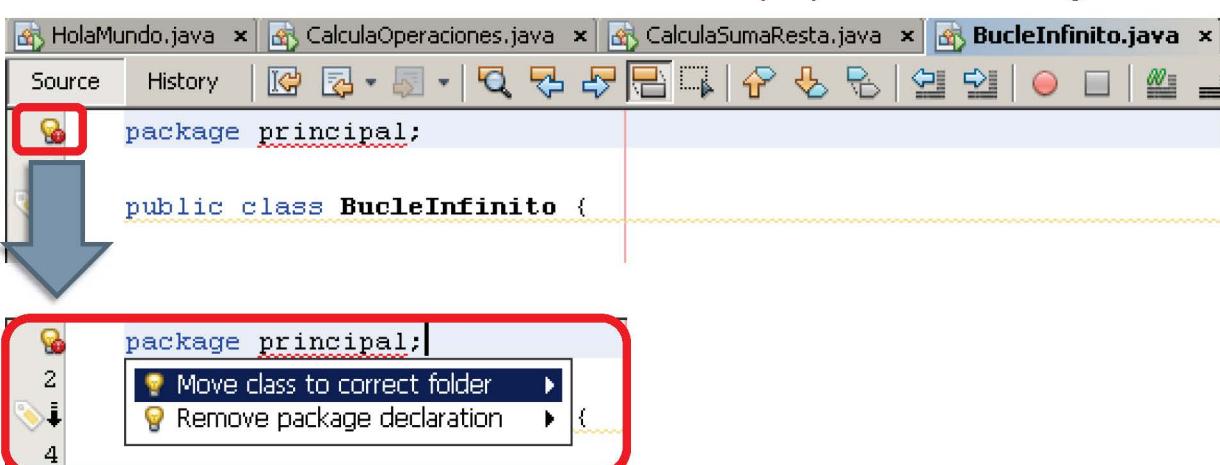
2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- Creación de paquetes:



18) Ubicar las clases `BucleInfinito.java` y `HolaMundo.java` en el paquete “principal” escribiendo “`package principal;`” y moviéndola adecuadamente usando el propio editor de código.



Alejandro Cardo Grau

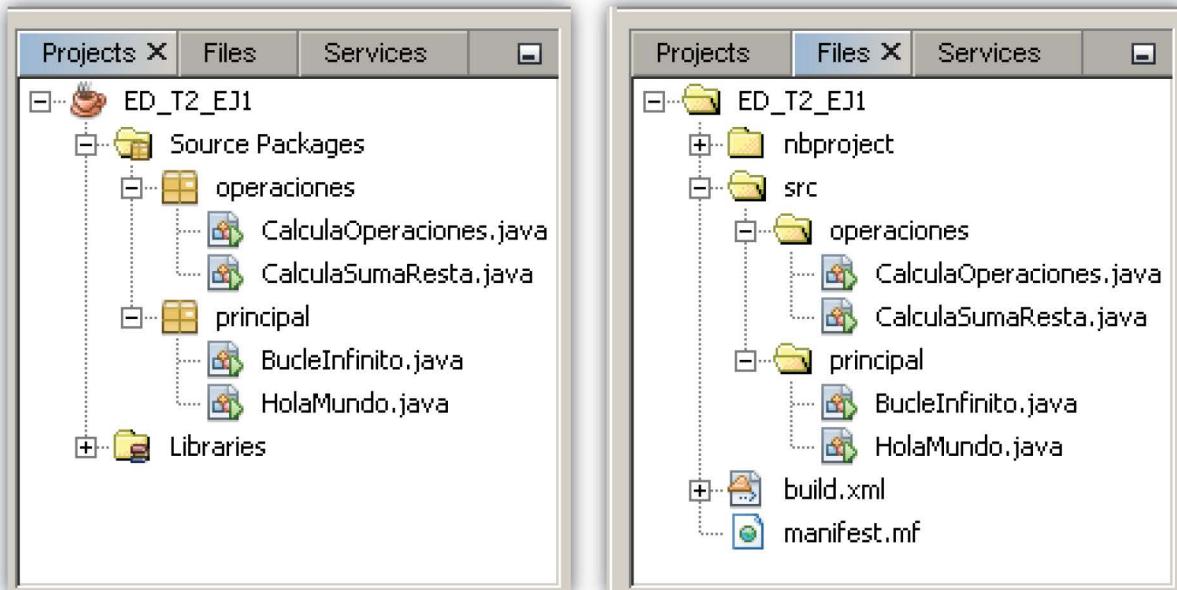
Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- **Creación de paquetes:**

- 19) Limpiar el proyecto y observar la estructura final de subdirectorios y ficheros creada.**



Alejandro Cardo Grau

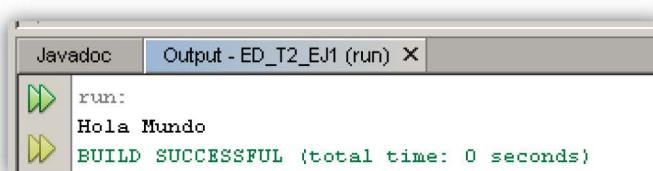
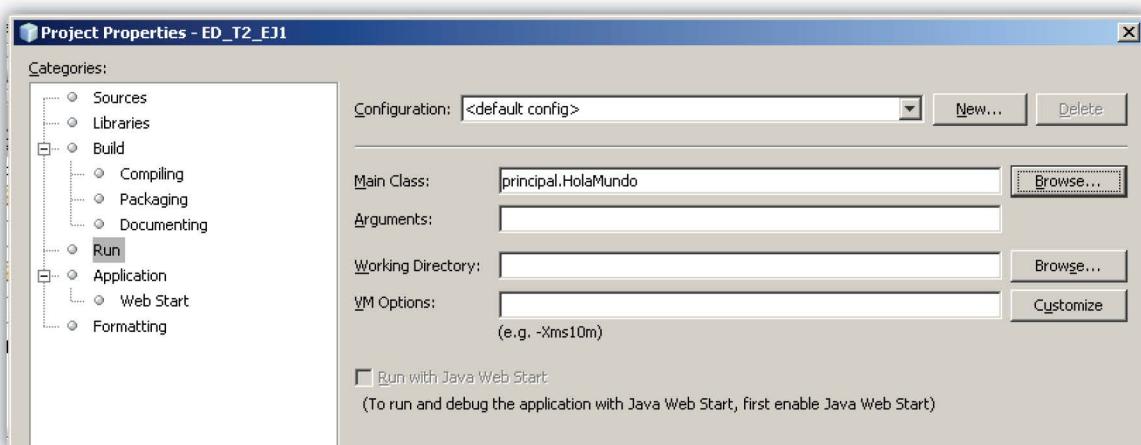
Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- **Creación de paquetes:**

- 20) Definir el main-class del proyecto en "HolaMundo.java", limpiar, generar y ejecutar el proyecto.**



Alejandro Cardo Grau

Entornos de Desarrollo



2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- El empaquetado de una aplicación Java en **JAR (Java Archive)** permite agrupar clases Java, librerías, recursos y un manifiesto de la aplicación.
- **Ventajas:**
 - Mejora la modularidad de las aplicaciones Java
 - Elimina dependencias en Classpath
 - Protege quién accede a qué
 - Soporte de versiones
- Adicionalmente JAR proporciona varios beneficios:
 - **Seguridad:**
 - Se puede firmar digitalmente el contenido del fichero JAR.
 - **Compresión:**
 - JAR comprime los ficheros usando el propio IDE.
 - **Portabilidad:**
 - Los ficheros JAR son un estándar del API de Java.
 - **Empaquetado**
 - Para extensiones, sellado y versionado de aplicaciones.
 - **Recursos:**
 - Al descargarse todos los recursos necesarios de una aplicación empaquetados en un único artefacto generado durante el desarrollo.

Alejandro Cardo Grau

Entornos de Desarrollo

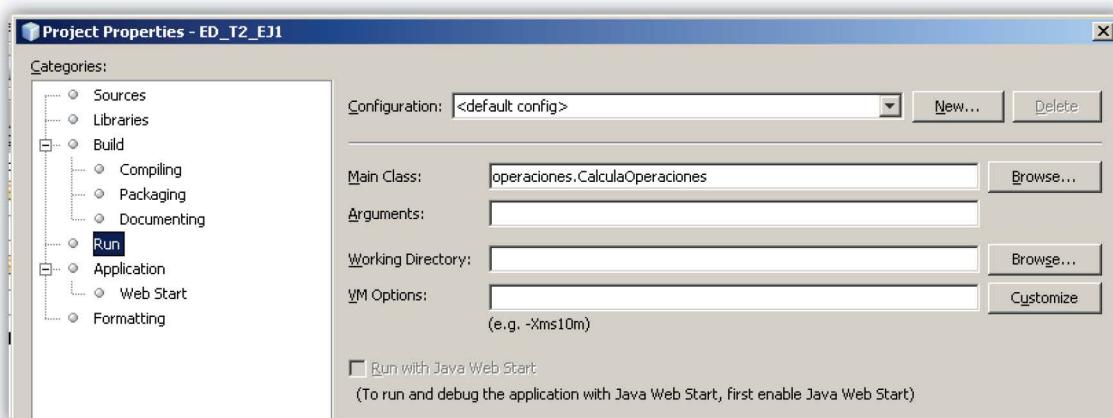


2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- **Creación del empaquetado JAR de la aplicación:**

21) Definir el main-class del proyecto en "CalculaOperaciones.java", limpiar y generar el proyecto (SIN EJECUTARLO).



Alejandro Cardo Grau

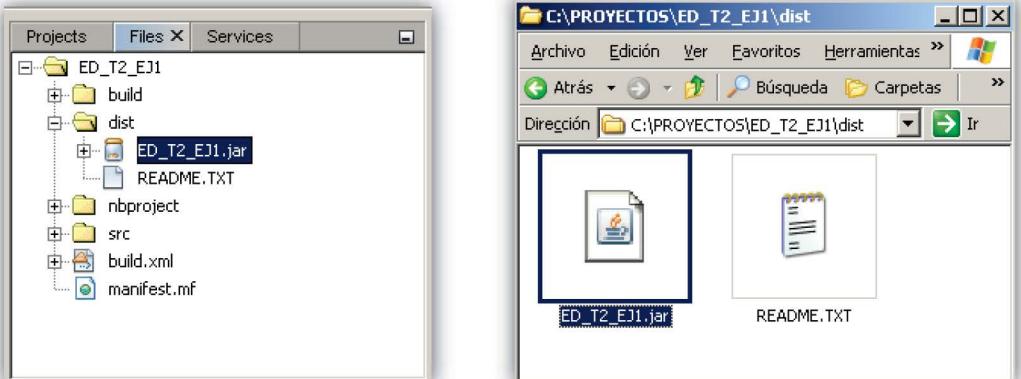
Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- **Creación del empaquetado JAR de la aplicación:**

- 22)** Localizar el empaquetado de la aplicación JAR generado en el directorio dist.



- 23)** Copiar el archivo .jar generado a otra ruta (por ejemplo C:\) y ejecuta la opción –jar de JVM desde terminal de línea de comandos (verifica antes si el PATH del sistema tiene JDK/bin correctamente enlazado):

```
C:\>java -jar ED_T2_EJ1.jar
SUMA: 14
RESTA: 10
MULTIPLICACIÓN: 24
DIVISIÓN: 6
RESTO: 0
```

Alejandro Cardo Grau

Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- **Uso de librerías y otros componentes para la aplicación:**

- Una librería se puede definir como un conjunto de clases que nos ayudan a desarrollar ciertas funciones dentro de nuestros proyectos Java.
- Vamos a ver como añadir una librería, empaquetada en JAR, a un determinado proyecto desde el entorno de desarrollo NetBeans.
 - Vamos a añadir un componente visual que nos permite manejar la fecha de una manera muy sencilla en nuestra aplicación:
 - Las clases se encuentran empaquetadas en una librería JAR denominada jcalendar-1.3.2.jar
- Para ubicarlas correctamente nuestra librería la ubicaremos en el subdirectorio /lib de nuestro proyecto que debemos crear.

Alejandro Cardo Grau

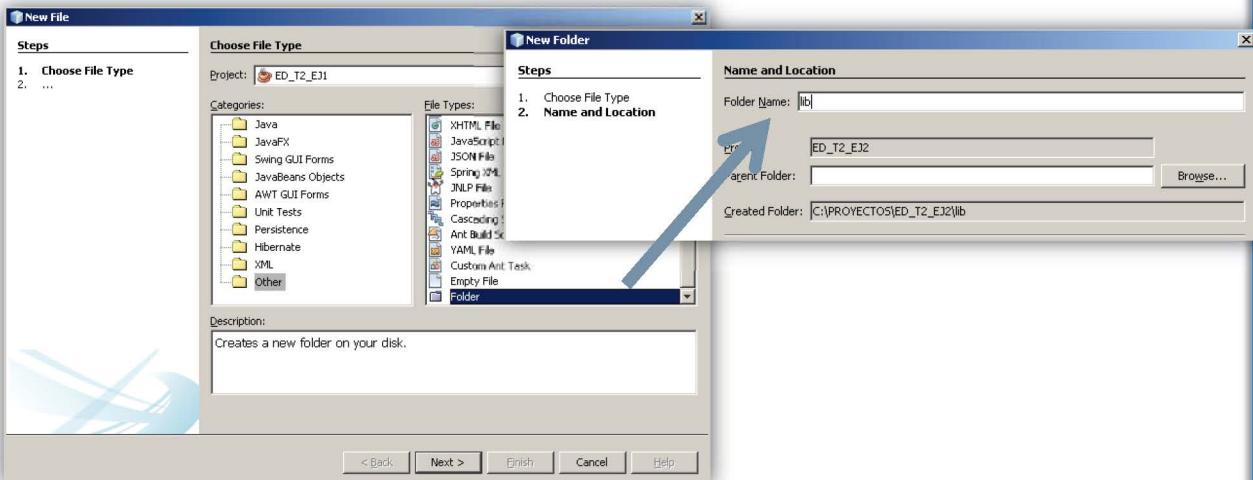
Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- **Uso de librerías y otros componentes para la aplicación:**

- 1) Descargar la librería `jcalendar-1.3.2.jar` aportada.
- 2) Crear el proyecto “ED_T2_EJ2” en el directorio de PROYECTOS sin ninguna Main-Class.
- 3) Crear una nueva carpeta en el Proyecto pulsando en “New / New File” / Other / New Folder” denominada “lib” y luego en “Finish”.

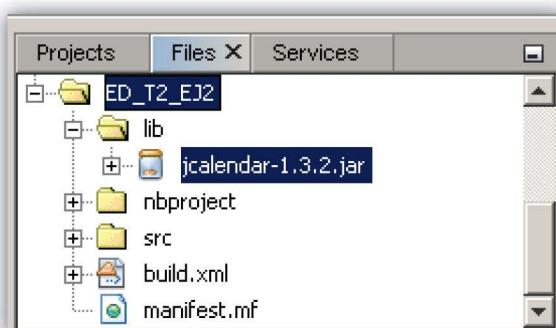


2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- **Uso de librerías y otros componentes para la aplicación:**

- 4) Mover la librería `jcalendar-1.3.2.jar` al directorio /lib del proyecto usando el Drag & Drop y la pestaña “Files”.

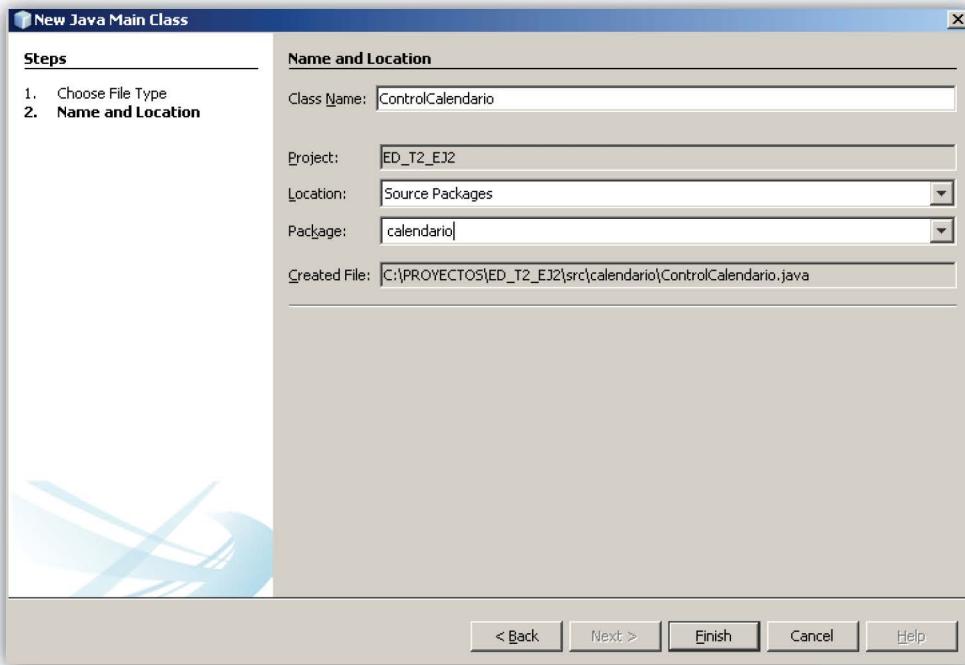


2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- Uso de librerías y otros componentes para la aplicación:**

- 5) Crear un clase “Java Main Class” denominada “ControlCalendario” y ubicada en el paquete “calendario”.



Alejandro Cardo Grau

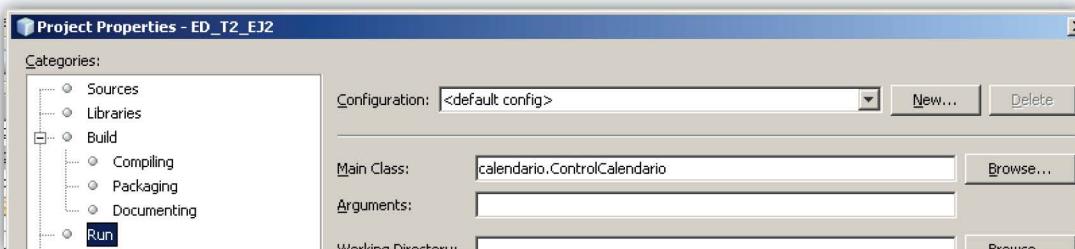
Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

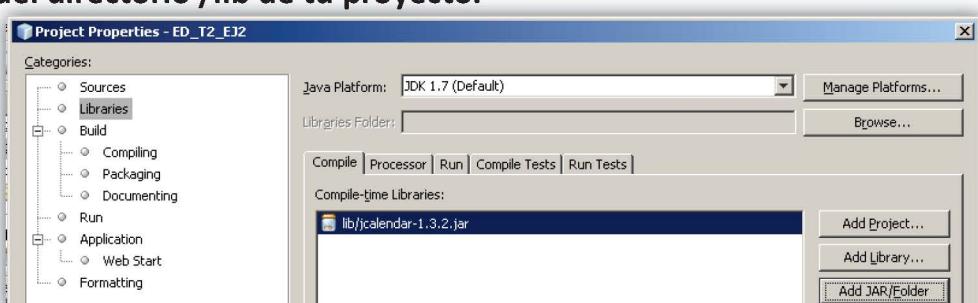
2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- Uso de librerías y otros componentes para la aplicación:**

- 6) Determinar en la configuración del proyecto como Main-Class la clase “ControlCalendario”.



- 7) Pulsar en “Libraries” de la misma ventana de propiedades, luego en “Add JAR / Folder” y localizar la librería: jcalendar-1.3.2.jar del directorio /lib de tu proyecto.



Alejandro Cardo Grau

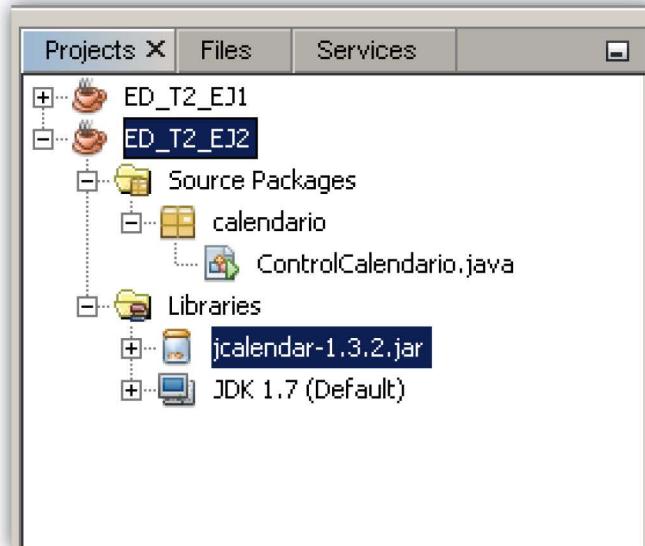
Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- Uso de librerías y otros componentes para la aplicación:

8) Comprobar que la librería se encuentra enlazada al proyecto:

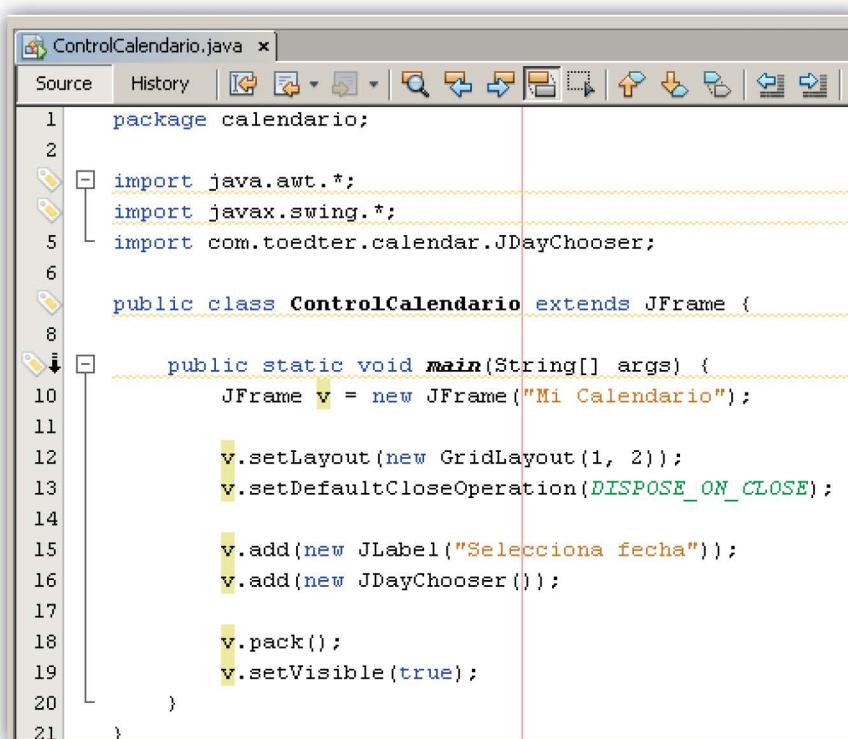


2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- **Uso de librerías y otros componentes para la aplicación:**

9) Escribir el siguiente código en “ControlCalendario.java”:

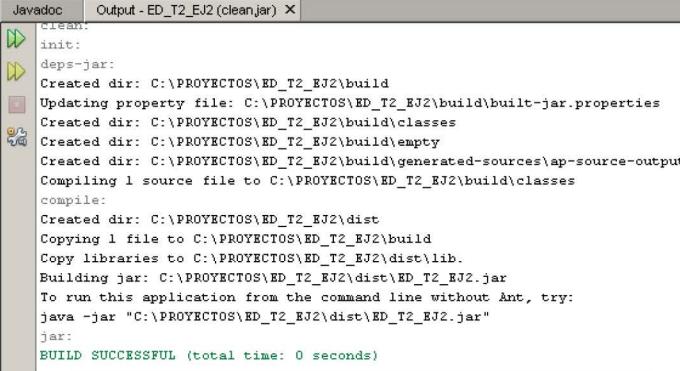


2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- **Uso de librerías y otros componentes para la aplicación:**

10) Construir y ejecutar el proyecto, comprobando la estructura de subdirectorios generada:



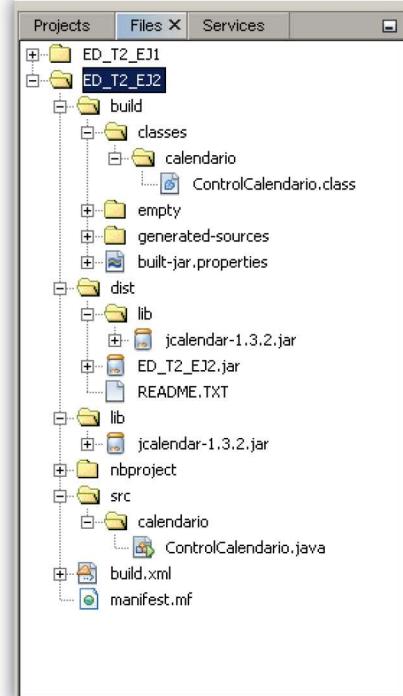
```

Javadoc Output - ED_T2_EJ2 (cleanjar) X
clean:
init:
deps-jar:
Created dir: C:\PROYECTOS\ED_T2_EJ2\build
Updating property file: C:\PROYECTOS\ED_T2_EJ2\build\built-jar.properties
Created dir: C:\PROYECTOS\ED_T2_EJ2\build\classes
Created dir: C:\PROYECTOS\ED_T2_EJ2\build\empty
Created dir: C:\PROYECTOS\ED_T2_EJ2\build\generated-sources\ap-source-output
Compiling 1 source file to C:\PROYECTOS\ED_T2_EJ2\build\classes
compile:
Created dir: C:\PROYECTOS\ED_T2_EJ2\dist
Copying 1 file to C:\PROYECTOS\ED_T2_EJ2\build
Copy libraries to C:\PROYECTOS\ED_T2_EJ2\dist\lib.
Building jar: C:\PROYECTOS\ED_T2_EJ2\dist\ED_T2_EJ2.jar
To run this application from the command line without Ant, try:
java -jar "C:\PROYECTOS\ED_T2_EJ2\dist\ED_T2_EJ2.jar"
jar:
BUILD SUCCESSFUL (total time: 0 seconds)

```



Alejandro Cardo Grau



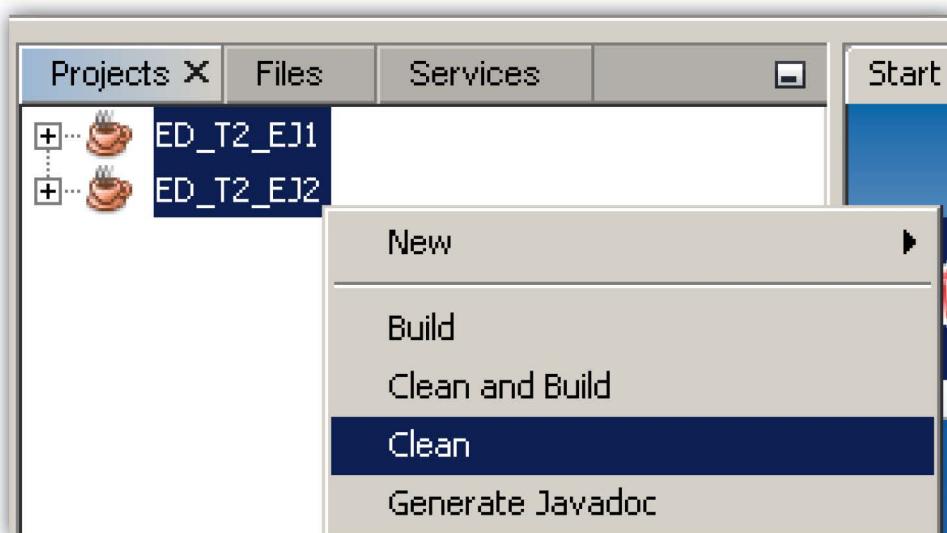
Entornos de Desarrollo

2. Uso Básico del Entorno de Desarrollo

2.4. Componentes de la aplicación, creación y distribución del empaquetado.

- **Uso de librerías y otros componentes para la aplicación:**

11) Limpiar los dos proyectos generados (selección múltiple):



Alejandro Cardo Grau

Entornos de Desarrollo

INDICE

- 1. Funciones de un Entorno de Desarrollo. Licencias.**
- 2. Componentes y Tipos de Entornos de Desarrollo**
- 3. Instalación y Configuración de un Entorno de Desarrollo**
- 4. Uso Básico del Entorno de Desarrollo Netbeans y Eclipse**

- 1. Manejo básico del IDE. Tipos de proyectos.**
- 2. Vistas, paneles de navegación, editores y perspectivas. Personalización.**
- 3. Herramientas para la compilación, ejecución y generación.**
- 4. Componentes de la aplicación, creación y distribución del empaquetado.**

5. Manejo de Herramientas para la Depuración**6. Integración de Plugins y Herramientas CASE en el Desarrollo****5. Manejo de Herramientas para la Depuración.**

- **Codificación:**
 - El programador recibe las especificaciones del diseño y las transforma en un conjunto de instrucciones escritas (código fuente).
- **Pruebas: Buscan comprobar la calidad y estabilidad del código realizado:**
 - Confirmando que la codificación ha sido exitosa y que no hay errores.
 - Comprobando que el software hace lo que debe hacer.
 - Se evalúan los resultados de la ejecución y fruto de esa evaluación se comprueba si existe correspondencia entre los:
 - Resultados esperados
 - Resultados obtenidos

T2: Entornos de Desarrollo

5. Manejo de Herramientas para la Depuración.

- **Tipos de errores en la codificación/programación:**

- **Errores sintácticos:**

- Fáciles de detectar y corregir desde el Editor de Textos del IDE.
- Localiza el error y presenta opciones para solucionarlo.
- Detectados en **tiempo de compilación**.
- **Ejemplos:**
 - Uso de una variable no declarada.
 - Omitir la separación de instrucciones del lenguaje (;)

- **Errores lógicos:**

- El programa se puede compilar con éxito.
- Su ejecución puede devolver resultados inesperados y erróneos.
- Más difíciles de detectar.
- Detectados en **tiempo de ejecución**.
- **Ejemplos:**
 - Bucle infinito
 - Uso de una variable no inicializada
 - Overflow (desbordamiento) del rango de datos permitido por una variable
 - Errores de E/S (Entrada-Salida)
 - División por cero:
 - velocidad = espacio / tiempo
 - Si la variable tiempo posee valor 0, se produce un error en tiempo de ejecución en la operación de división.

Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

5. Manejo de Herramientas para la Depuración.

- **Depuración:**

- Es el proceso de identificar y corregir errores tras la codificación.
- Una vez identificado los errores en la fase de pruebas, se procede a corregirlos.
- Cuando se lanza el proceso de depuración del código en el IDE, siempre se realiza una compilación y construcción completa del código del proyecto.
- También denominado como *debugging*, ya que se asemeja a la eliminación de bichos (**bugs**):
 - Manera en que se conoce informalmente a los errores de codificación/programación.

- **Resultados de la depuración:**

- Se encuentra la causa del error, se corrige y se elimina.
- No se encuentra la causa del error:
 - En la fase de pruebas del desarrollo se deben diseñar los casos de prueba.
 - Repetir las pruebas necesarias hasta identificar los errores y poder corregirlos.

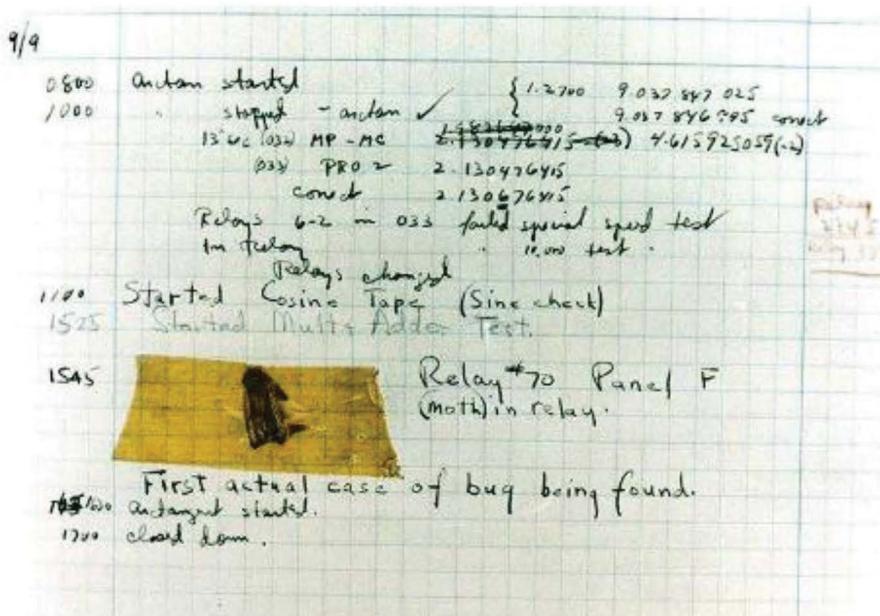
Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

5. Manejo de Herramientas para la Depuración.

El 9 de Septiembre de 1945, Grace Murray Hopper, la madre del Cobol, estaba trabajando con uno de los primeros ordenadores, un Mark II Aiken Relay Calculator cuando el equipo comenzó a dar problemas, al revisarlo un técnico observó que había un polilla entre los puntos de los relés



El bicho y la bitácora se encuentran en el Museo Nacional de Historia Americana, Smithsonian, en Washington, D.C.

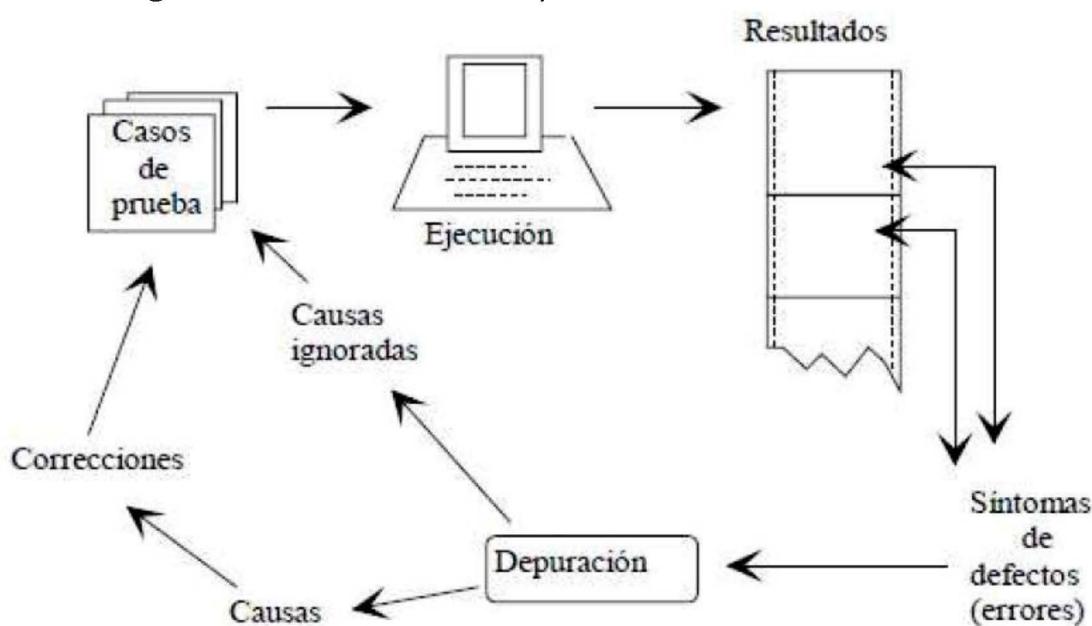
Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

5. Manejo de Herramientas para la Depuración.

- **Un caso de prueba es un conjunto de entradas, condiciones de ejecución y resultados esperados, desarrollado para:**
 - Conseguir un objetivo.
 - Conseguir una condición en la prueba.



Alejandro Cardo Grau

Entornos de Desarrollo

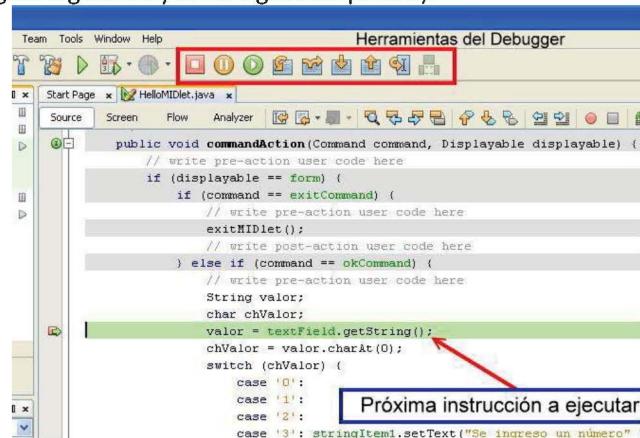
T2: Entornos de Desarrollo

5. Manejo de Herramientas para la Depuración.

- **IDE Netbeans:**

- **1º OPCIÓN: Inicio del modo depuración ([ir a la instrucción](#)):**

- 1) Es necesario definir un **proyecto Netbeans** con una **Main-Class**.
- 2) Pulsar clic sobre la instrucción que se quiere alcanzar.
- 3) Seleccionar **Debug / Run to Cursor** ó **tecla F4**
- 4) Aparece una línea verde ahora en la instrucción marcada:
 - Esa línea verde indica que la ejecución del programa se detuvo y espera la orden del usuario para continuar la ejecución
 - Las ordenes vienen de la barra de herramientas tal y como se observa en la figura siguiente y en el siguiente paso 5).



Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

5. Manejo de Herramientas para la Depuración.

- **IDE Netbeans:**

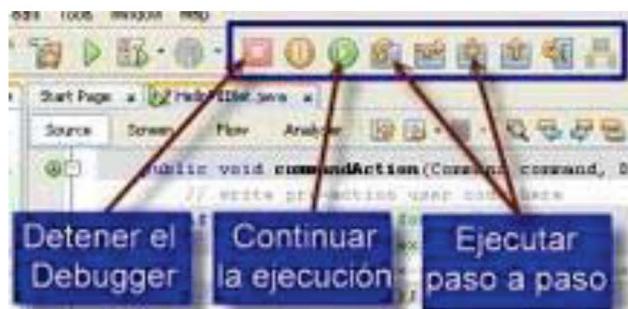
- **1º OPCIÓN: Inicio del modo depuración ([ir a la instrucción](#)):**

- 5) Usar las herramientas del debugger:

- **Detener el debugger**
 - Seleccionar **Debug / Finish Debugger Session** ó teclas **SHIFT + F5**
 - Cancelar el proceso



- **Ejecutar paso a paso cada instrucción, incluso de métodos (tecla F7)**
- **Ejecutar paso a paso, pero si encuentra un método saltarlo (tecla F8)**
 - Permite ejecutar el programa instrucción por instrucción, saltando sus métodos, esto para ir analizando principalmente lo que va pasando en el programa. Se ejecuta la instrucción que está en la línea verde.



Línea Verde:

La línea verde indica la próxima instrucción a ejecutarse.

Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

5. Manejo de Herramientas para la Depuración.

- **IDE Netbeans:**

- **Inspección/visores de variables (watches):**

- **1º FORMA:**

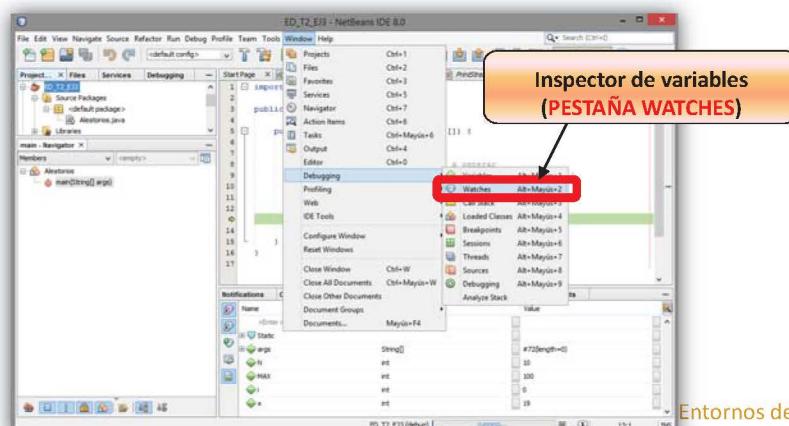
- Con solo colocar el ratón encima de la variable.

```
for (i = 0; i < N; i++) {
    x = new Random().nextInt(24);
}
System.out.println(x);
```

Diferenciar entre PESTAÑA
VARIABLES Y WATCHES

- **2º FORMA: Pestaña Variables (Window/Debugging/Watches):**

- Permite agregar y mostrar el contenido y tipo de la variable.
- Permite modificar incluso el valor de la variable en tiempo de ejecución.



Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

5. Manejo de Herramientas para la Depuración.

- **IDE Netbeans:**

- **2º OPCIÓN: Inicio del modo depuración (breakpoints):**

- El debugger se detiene en el punto de ruptura marcado en la instrucción y nos permite inspeccionar los valores de algunas variables.

- 1) Es necesario definir un **proyecto Netbeans** con una **Main-Class**.
- 2) Se crea un punto de ruptura (*break point*) justo en la instrucción o instrucciones donde se desea que se analice el código con un simple clic del mouse (1) en el lateral del Editor de Textos del IDE.
- 3) Seleccionar **Debug / Debug Project** ó teclas **CTRL + F5**



Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

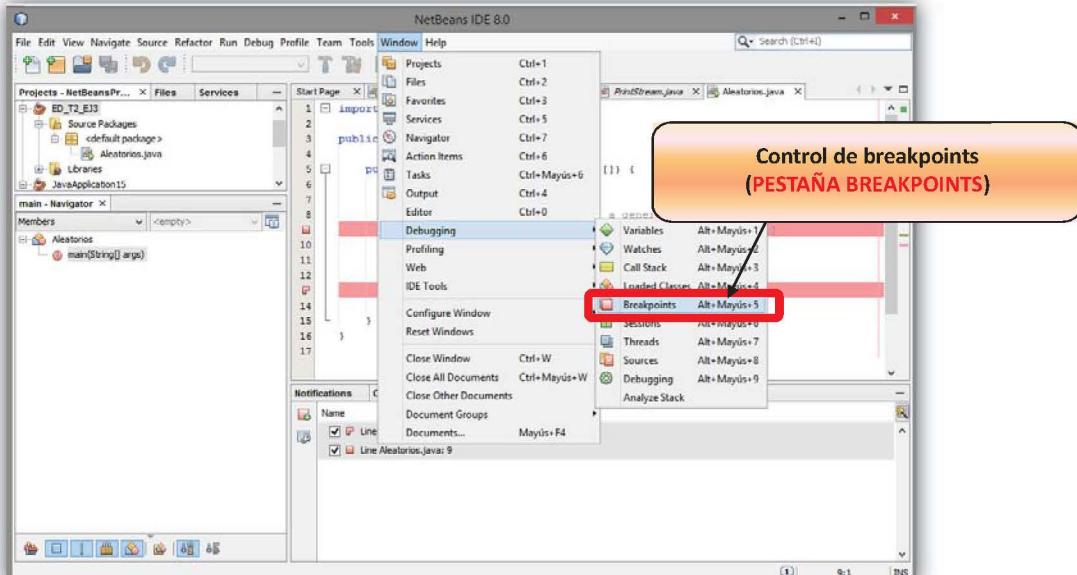
5. Manejo de Herramientas para la Depuración.

Debug Project (CTRL + F5):
Aunque se escoja la 1º OPCIÓN (Run to Cursor) SERÁ MAS PRIORITARIO PARA
EL MODO DEPURACIÓN LOS
BREAKPOINTS ANTERIORES DEFINIDOS

- **IDE Netbeans:**

- **Control de breakpoints:**

- Permite agregar nuevos breakpoints con propiedades avanzadas.
- Permite mostrar los números de línea y clases donde se encuentran
- Permite desactivar y eliminar breakpoint definidos (pulsa doble clic)



Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

5. Manejo de Herramientas para la Depuración.

- **IDE Netbeans:**

- **Secuencia de control y parada sólo entre breakpoints:**

- 1) Definir los breakpoints necesarios:

```

9     N = 10; // N números aleatorios a generar
10    MAX = 100; // Número aleatorio calculado entre [0, MAX-1]
11
12    for ( i = 0; i < N; i++ ) {
13        x = new Random().nextInt(MAX);
14        System.out.println(x);
15    }

```

- 2) Seleccionar **Debug / Debug Project** ó teclas **CTRL + F5**

- 3) Se detiene en el primer breakpoint definido:

```

9     N = 10; // N números aleatorios a generar
10    MAX = 100; // Número aleatorio calculado entre [0, MAX-1]
11
12    for ( i = 0; i < N; i++ ) {
13        x = new Random().nextInt(MAX);
14        System.out.println(x);
15    }

```

- 4) Para saltar al siguiente breakpoint: seleccionar **Debug / Continue** ó **tecla F5** ó botón

- 5) Se detiene en el siguiente breakpoint definido:

```

9     N = 10; // N números aleatorios a generar
10    MAX = 100; // Número aleatorio calculado entre [0, MAX-1]
11
12    for ( i = 0; i < N; i++ ) {
13        x = new Random().nextInt(MAX);
14        System.out.println(x);
15    }

```

Alejandro Cardo Grau

Entornos de Desarrollo

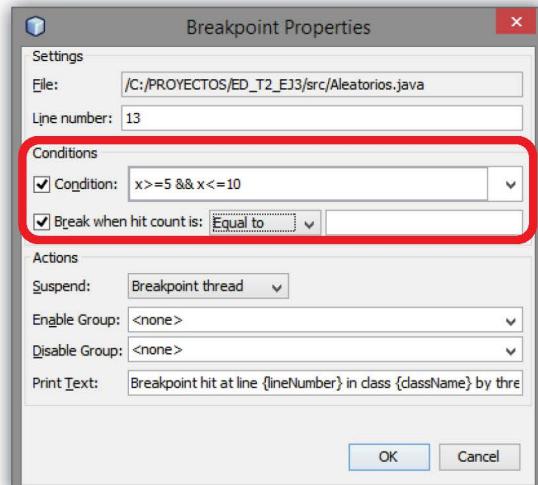
T2: Entornos de Desarrollo

5. Manejo de Herramientas para la Depuración.

- IDE Netbeans:

- Breakpoint Properties

- Es el control para determinar propiedades concretas a un breakpoint, determinando un caso de prueba para establecer:
 - Condiciones de parada en el número de iteraciones calculadas.
 - Condiciones en las expresiones lógicas determinadas, incluso entre variables declaradas en el propio código fuente generado.
 - Es posible continuar el proceso de depuración con:
 - Debug / Continue
 - Tecla F5



Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

5. Manejo de Herramientas para la Depuración.

- IDE Netbeans:

Antes

Durante

Tras
ejecutar

Alejandro Cardo Grau

Entornos de Desarrollo

5. Manejo de Herramientas para la Depuración

EJERCICIO 1 (Casos de Prueba): Aleatorios.java (ED_T3_EJ3)

- EJERCICIO 1: Describe los pasos generados en el IDE Netbeans para:**

- 1) Calcular el valor de **i**, **x**, **N** y **MAX** tras ejecutar la 5º iteración.
- 2) Determinar el número de iteraciones posibles (valor de variable **i**) para obtener un número aleatorio mayor o igual que 10.
- 3) Modificar el valor del aleatorio **x**, generado en la 6º iteración, para imprimir obligatoriamente un número igual a 10.

ESCRIBIR VALORES
2 CAPTURAS
3 CAPTURAS

```
import java.util.Random;

public class Aleatorios {

    public static void main(String args[]) {
        int i, x, N, MAX;

        N = 10; // N números aleatorios a generar
        MAX = 100; // Número aleatorio calculado entre [0, MAX-1]

        for (i = 0; i < N; i++) {
            x = new Random().nextInt(MAX);
            System.out.println(x);
        }
    }
}
```

Alejandro Cardo Grau

Entornos de Desarrollo

5. Manejo de Herramientas para la Depuración

EJERCICIO 2 (Casos de Prueba): NumeroMagico1.java (ED_T3_EJ3)

- EJERCICIO 2: Describe los pasos generados en el IDE Netbeans para:**

- 1) Calcular el valor de las variables tras ejecutar la 4º iteración.
- 2) Determinar el número de iteraciones posibles (valor de variable **n**) para obtener un primer número **num** entre 10 y 20.
- 3) Calcular el valor de las variables tras finalizar el programa.

ESCRIBIR VALORES
2 CAPTURAS
3 CAPTURAS

```
public class NumeroMagico1 {
    public static void main(String args[]) {
        int num, n;

        n = 0;
        num = 6;

        while (num != 1 && n < 100) {
            if (num % 2 == 0) {
                num = num / 2;
            } else {
                num = 3 * num + 1;
            }
            n++;
        }
    }
}
```

Alejandro Cardo Grau

Entornos de Desarrollo

5. Manejo de Herramientas para la Depuración

EJERCICIO 3 (Casos de Prueba): NumeroMagico2.java (ED_T3_EJ3)

- EJERCICIO 3: Describe los pasos generados en el IDE Netbeans para:**

- 1) Calcular el valor de las variables tras ejecutar la 2º iteración del 2º bucle.
- 2) Determinar el valor de las variables para obtener un primer número **num** que sea impar.
- 3) Calcular el valor de las variables tras finalizar el programa.

ESCRIBIR VALORES

2 CAPTURAS

3 CAPTURAS

```
public class NumeroMagico2 {
    public static void main(String args[]) {
        int i, j, num;

        num = 0;

        for (i = 1; i <= 5; i = i + 2) {
            j = 0;

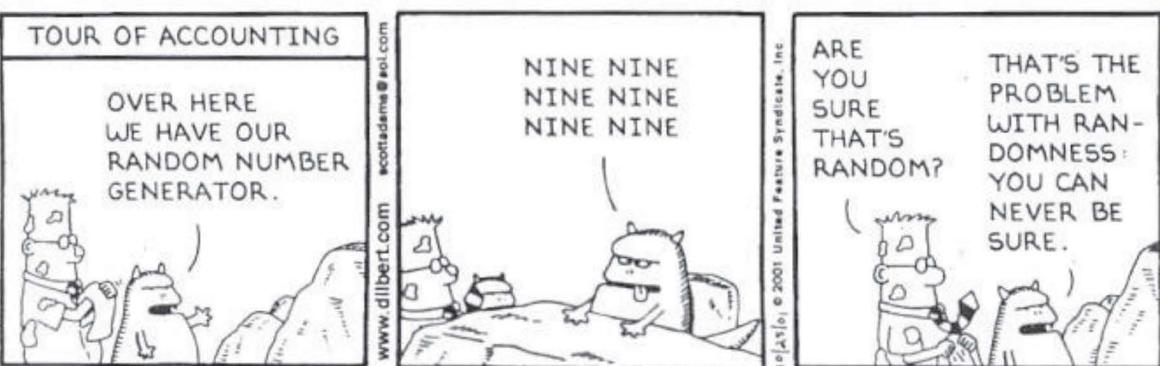
            while (j < 3 && num < 10) {
                j++;
                num = (num * num) + j;
            }
        }
    }
}
```

Alejandro Cardo Grau

Entornos de Desarrollo

T2: Entornos de Desarrollo

7. REFERENCIAS



Alejandro Cardo Grau

Entornos de Desarrollo