



2. Instalación y uso de entornos de desarrollo

Autor	ⓧ Xerach Casanova
Clase	Entornos de desarrollo
Fecha	@Nov 24, 2020 10:22 AM

1. Concepto de entorno de desarrollo. Evolución histórica
 - 1.1. Evolución histórica
2. Funciones de un entorno de desarrollo
3. Entornos integrados libres y propietarios
4. Estructura de entornos de desarrollo.
5. Instalación de entornos integrados de desarrollo
 - 5.1. Instalación de JDK
 - Configuración de las variables de entorno.
6. Configuración y personalización de entornos de desarrollo
7. Gestión de módulos (Netbeans)
 - 7.1. Añadir módulos
 - 7.2. Eliminar módulos
 - 7.3. Funcionalidades
 - 7.4. Herramientas concretas
8. Uso básico de entornos de desarrollo.
 - 8.2. Generación de ejecutables
9. Actualización y mantenimiento de entornos de desarrollo.

1. Concepto de entorno de desarrollo. Evolución histórica

Un entorno integrado de desarrollo (IDE), es un tipo de software compuesto por un conjunto de herramientas de programación. De las cuales podemos destacar:

- Editor de código de programación

- Accesos al compilador desde botones u opciones de menú.
- Acceso a la ejecución del programa desde botones u opciones de menú.
- Depurador.
- Constructor de interfaz gráfico.

El objetivo de los IDE es ganar fiabilidad y tiempo en proyectos de software garantizando comodidad, aumento de eficiencia y reducción de tiempo de codificación.

Suelen estar dedicados a un determinado lenguaje, pero últimamente tienden a ser compatibles varios lenguajes con la instalación de plugins.

1.1. Evolución histórica

El primer lenguaje en utilizar un IDE fue BASIC, el primero en abandonar las tarjetas perforadas. Estaba basado solamente en una consola de comandos, pero la gestión de archivos, compilación y depuración es semejante a los IDE actuales.

El primer IDE popular se llama Maestro, nació en los 70 y fue utilizado por unos 22000 programadores en todo el mundo.

Estos son los IDE más relevantes en la actualidad:

Entorno de desarrollo	Lenguajes que soporta	Tipo de licencia
NetBeans.	C/C++, Java, JavaScript, PHP, Python.	De uso público.
Eclipse.	Ada, C/C++, Java, JavaScript, PHP.	De uso público.
Microsoft Visual Studio.	Basic, C/C++, C#.	Propietario.
C++ Builder.	C/C++.	Propietario.
JBuilder.	Java.	Propietario.

La elección del IDE más adecuado depende del lenguaje y el tipo de licencia con la que queramos trabajar.

2. Funciones de un entorno de desarrollo

Las funciones de los IDE son:

- Editor de código: coloración de la sintaxis.
- Auto-compleado de código, atributos y métodos.
- Identificación automática de código.
- Herramientas de concepción visual para crear y manipular componentes visuales.
- Asistentes y utilidades de gestión y generación de código.
- Organización de archivos fuente en carpetas y compilados en otras.
- Compilación de proyectos complejos en un solo paso.
- Control de versiones (almacenaje de archivos compartido por todos los colaboradores del proyecto y auto-recuperación).
- Soporta cambios de varios usuarios simultáneamente.
- Generador de documentación integrado.
- Detección de errores de sintaxis en tiempo real.

Otras funciones son:

- Refactorización de código: cambios menores que facilitan su legibilidad sin alterar funcionalidad (cambiar el nombre a una variable).
- Tabulaciones y espaciados.
- Depuración: seguimiento de variables, puntos de ruptura, mensajes de error del intérprete.
- Aumento de funcionalidades a través de módulos y plugins.
- Administración de interfaz de usuario.
- Administración de configuración de usuario.
- Empaquetar software para su posterior despliegue e instalación en entorno de ejecución.

3. Entornos integrados libres y propietarios

Entornos de desarrollo libres más relevantes:

Entornos de desarrollo libres más relevantes en la actualidad		
IDE	Algunos lenguajes que soporta	URL
Eclipse	Ada, C/C++, Java, JavaScript, PHP	https://www.eclipse.org/
NetBeans	C/C++, Java, JavaScript, PHP, HTML5 ...	https://netbeans.org/
CodeLite	C/C++, PHP, Node.js	https://codelite.org
JDeveloper	Java, HTML, XML, SQL, PL/SQL, Javascript, PHP, UML ...	http://www.oracle.com/technetwork/developer-tools/jdev/overview/index.html
IntelliJ	Java, Groovy, Perl, Scala,XML/XSL,Python ...	https://www.jetbrains.com/idea/
Microsoft Visual Studio	C#, Visual Basic, F#, C++, HTML, JavaScript, TypeScript, Python, ...	https://visualstudio.microsoft.com/es/vs/community/ https://code.visualstudio.com/

Entornos integrados propietarios:

Entornos de desarrollo propietarios más relevantes en la actualidad		
IDE	Algunos lenguajes que soporta	URL
Microsoft Visual Studio	C++, C#, Visual Basic ...	https://visualstudio.microsoft.com/es/
C++ Builder	C++	https://www.embarcadero.com/es/
IntelliJ	Java, Groovy, Perl, Scala, ML/XSL,Python, Ruby, Sql ...	https://www.jetbrains.com/idea/
QtCreator	C++ con framework QT	https://www.qt.io/

4. Estructura de entornos de desarrollo.

Los entornos de desarrollo están formados por:

- Editor de textos:
 - Resaltado y coloreado de sintaxis.
 - Funciones de auto-completado de código.
 - Inserción automática de paréntesis, corchetes, tabulaciones y espaciadoos.
 - Ayuda y listado de parámetros de funciones y métodos de clases.
- Compilador/intérprete:
 - Detección de errores de sintaxis en tiempo real.

- **Depurador:**
 - Ejecución del programa paso a paso, definición de ruptura y seguimiento de variables. Opción depurar en servidores remotos.
- **Generador automático de herramientas:**
 - Herramientas para la visualización, creación y manipulación de componentes visuales, utilidades de gestión y generación de código.
- **Interfaz gráfica:**
 - Oportunidad de programar en varios lenguajes en el mismo IDE. Acceso a bibliotecas y plugins.

5. Instalación de entornos integrados de desarrollo

5.1. Instalación de JDK

JDK (Java Development Kit) es el entorno de desarrollo necesario para poder ejecutar NetBeans. Trae integrado el JRE y un compilador.

Para poder interpretar los archivos bytecodes (obtenidos tras la compilación del código fuente de Java), se necesita tener instalado el JRE (Java Runtime Environment). El JRE está conformado por la máquina virtual de Java (JVM), las bibliotecas Java y otros componentes necesarios para poder ejecutar una aplicación escrita en Java. JRE es un intermediario entre el S.O. y Java.

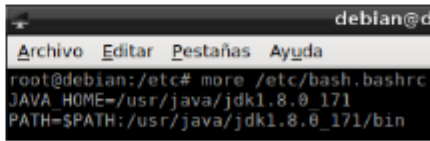
La JVM es el programa que ejecuta los bytecodes y las librerías de clase estándar son las que implementan el API de Java, ambas deben ser consistentes entre sí y se distribuyen en conjunto.

Los programas más importantes que se incluyen en el JDK son:

- **appletviewer:** visor de applets para generar sus vistas previas (carecen de método main y no los puede ejecutar el programa java).
- **javac.exe:** compilador de java
- **java.exe:** intérprete de java.
- **javadoc.exe:** genera la documentación de las clases de un programa en java.

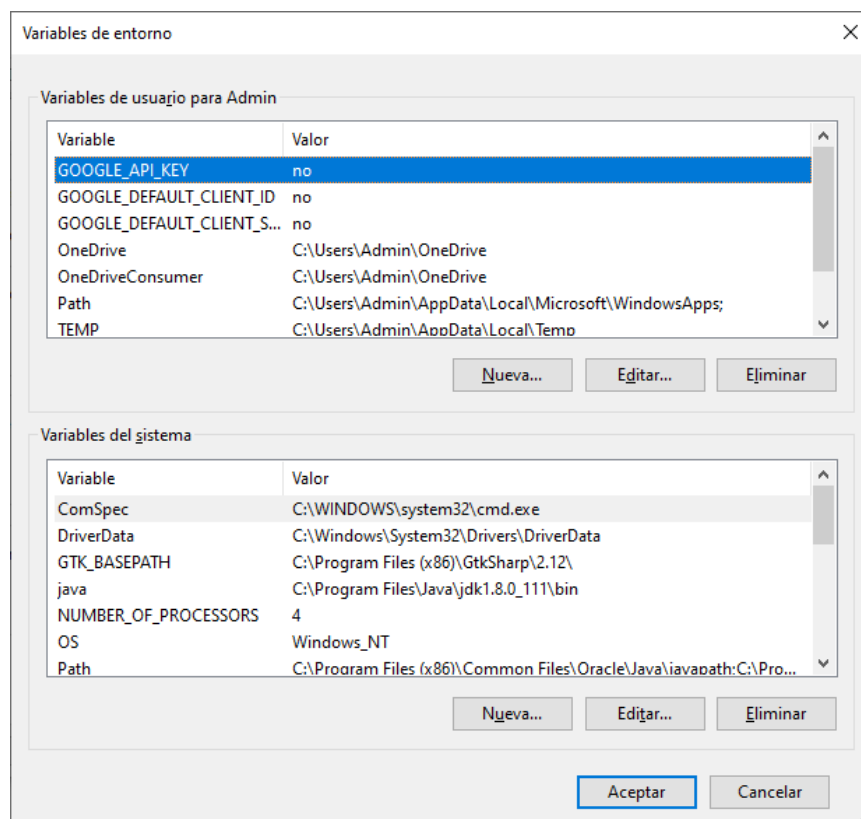
Configuración de las variables de entorno.

A través de Linux (con bash):

Acción	Orden en consola linux
Abrir un terminal Linux.	
Ganar privilegios de administrador.	sudo su + Contraseña
Con un editor de texto (por ejemplo nano), acceder al fichero /etc/bash.bashrc.	nano /etc/bash.bashrc
Introducir las variables de entorno: JAVA_HOME=/usr/java/jdk1.8.0_171 PATH=\$PATH:/usr/java/jdk1.8.0_171/bin En este caso, la ruta del jdk en JAVA_HOME y también añadimos la ruta de los ejecutables en PATH	

A través de Windows 10:

- Ir a variables de entorno en configuración avanzada del sistema.



Después elegimos Nueva (en variables de Sistema) e introducimos los datos de la variable de entorno, su nombre: JAVA_HOME y su valor (la ruta donde está instalado el jdk). En este ejemplo es un jdk 8. Después Aceptar.

Como la variable PATH ya existe, seleccionamos la variable PATH y elegimos Editar. Añadimos la ruta del directorio bin de nuestro jdk. Ponemos un punto y coma y luego %JAVA_HOME%\bin. Para finalizar, Aceptar.

6. Configuración y personalización de entornos de desarrollo

Parámetros y configuraciones del entorno de desarrollo:

- Carpeta o carpetas donde se alojarán los archivos de los proyectos.
- Carpetas de almacenamiento de paquetes fuente y paquetes prueba.
- Administración de la plataforma del entorno de desarrollo.
- Opciones de la compilación de los programas.
- Opciones de empaquetado de la aplicación.
- Opciones de generación de documentación asociada al proyecto.
- Descripción de los proyectos.
- Opciones globales de formato del editor.
- Opciones de combinación de atajos de teclado.

7. Gestión de módulos (Netbeans)

Con plataformas como Netbeans podemos hacer uso de módulos o plugins para desarrollar aplicaciones.

En categoría de lenguajes de programación podremos añadir nuevos lenguajes soportados.

Un módulo es un componente software con clases de java que pueden interactuar con las API y el manifest file (archivo especial que lo identifica como módulo). Se pueden construir y desarrollar de forma independiente.

7.1. Añadir módulos

Tenemos varias opciones:

- Añadir módulos de los que NetBeans instala por defecto.
- Descargar un módulo de algún sitio web permitido y añadirlo.
- Instalarlo on-line en el entorno.
- Crear el módulo nosotros mismos.

El plugin se descarga en formato .nbm y desde el IDE se carga e instala (adición off-line).

También se pueden instalar on-line, sin salir del IDE. Para ello debemos tener instalado el plugin "Portal Update Center"

7.2. Eliminar módulos

Eliminar un módulo requiere seguir los siguientes pasos.

1. Encontrar el módulo en la lista de complementos instalados.
2. Optar por:
 - Desactivarlo.
 - Desinstalarlo.

7.3. Funcionalidades

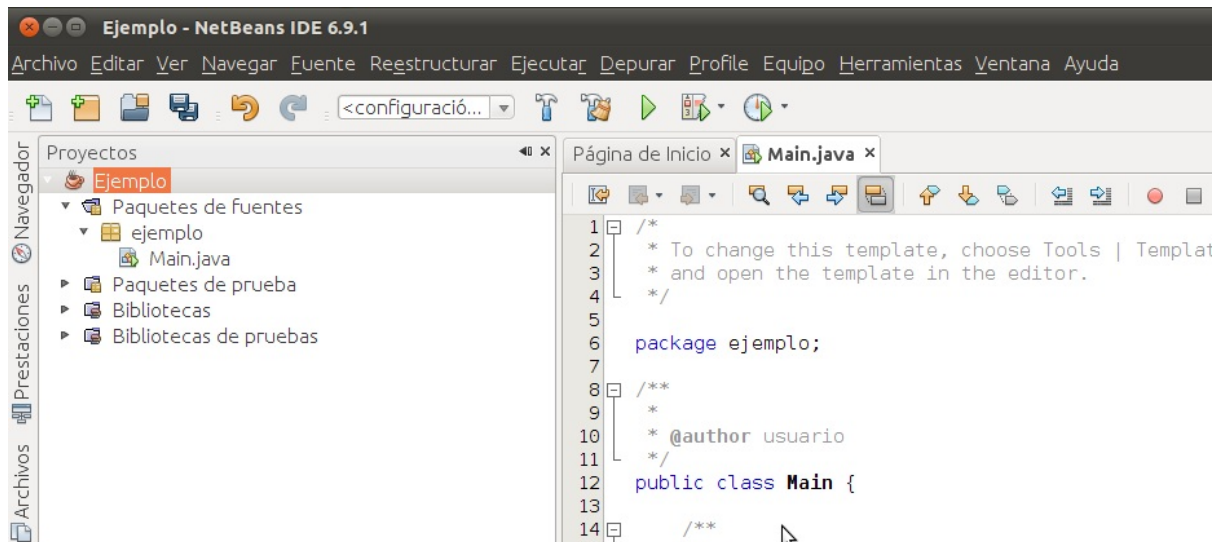
Los módulos y plugins tienen diversas funciones categorizadas por:

- Construcción de código.
- Bases de datos.
- Depuradores.
- Aplicaciones: añaden aplicaciones que pueden ser útiles.
- Edición.
- Documentación de aplicaciones.
- Interfaz gráfica de usuario.
- Refactorización.
- Aplicaciones web.
- Prueba.

7.4. Herramientas concretas

- Importador de proyectos de Netbeans: permite trabajar en lenguajes como JBuilder.
- Servidores de aplicaciones GlassFish: Plataforma completa para aplicaciones de tipo empresarial
- Soporte para JEE: Cumplimiento de estándares, facilidad e uso, mejora de rendimiento.
- NetBeans Swing GUI Builder: simplifica la creación de interfaces gráficos en aplicaciones cliente y permite manejar diferes aplicaciones sin salir del IDE.
- NetBeans Profiler: permite ver de forma inmediata la eficiencia de un trozo de software.
- Editor WSDL: facilita a los programadores trabajar en servicios Web basados en XML
- XML Schema Editor: permite refinar aspectos de los documentos XML.
- Aseguramiento de la seguridad de datos mediante Sun Java System Access Manager.
- Soporte beta de UML que cubre actividades como clases, comportamiento interacción y secuencias.
- Soporte bidireccional: que permite sincronizar con rapidez modelos de desarrollo con los cambios en el código conforme avanzamos por las etapas del ciclo de vida de la app.

8. Uso básico de entornos de desarrollo.



Ventana izquierda: ventana de proyectos.

Donde aparece la relación de proyectos, archivos, módulos o clases.

Cada proyecto comprende una serie de archivos y bibliotecas. El archivo principal es el Main.java, que es donde empieza la ejecución. No tiene que llamarse Main pero es aconsejable.

Ventana derecha: espacio de escritura de los códigos de los proyectos.

El esqueleto propio de un programa escrito en lenguaje java. Tiene las siguientes características:

- Autocompletado de código
- Coloración de comandos.
- Subrayado en rojo cuando hay algún error o posibilidad de deupración y corrección (a través de un pequeño icono en la parte izquierda de la línea defecutosa).

Barra de herramientas.

Para acceder a todas las opciones del IDE.

8.2. Generación de ejecutables

Una vez tenemos el código terminado y libre de errores de sintaxis los siguientes pasos son: compilación, depuración, ejecución. Para ejecutar solo hay que pulsar shift+F6.

9. Actualización y mantenimiento de entornos de desarrollo.

El mantenimiento y las actualizaciones se hacen de forma On-Line, con el complemento Auto Update Services, el cual permite realizar continuas revisiones del entorno y actualizaciones de plugins.

La gestión de las bases de datos asociadas a nuestros proyectos es importante y hay que realizar copias de seguridad periodicas. Tanto para asegurar su restauración en caso de fallos como para mantenerlas actualizadas para su posible portabilidad a nuevas versiones del entorno que utilicemos.