1. Concepto de entorno de desarrollo.

Un entorno integrado de desarrollo (IDE), es un tipo de software compuesto por un conjunto de herramientas de programación. Un IDE se compone de:

- Editor de código
- Compilador
- Intérprete
- Depurador
- Constructor de interfaz gráfico

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.

2. Funciones de un entorno de desarrollo.

Las funciones de los IDE son:

- Editor de código.
- Auto-completado de código, atributos y métodos de clases.
- Identificación automática de código.
- Herramientas para crear y manipular componentes visuales.
- Asistentes y utilidades de gestión y generación de código.
- Archivos fuente en unas carpetas y compilados a otras.
- Compilar proyectos complejos en un solo paso.
- Control de versiones: tener un único almacén de archivos compartido por todos los colaboradores de un proyecto. Ante un error, mecanismo de auto-recuperación a un estado anterior estable.
- Soportar cambios de varios usuarios de manera simultánea.
- Generador de documentación integrado.
- Detectar de errores de sintaxis en tiempo real.

Otras funciones importantes son:

- Ofrece refactorización de código: cambios menores en el código que facilitan su legibilidad sin alterar su funcionalidad (por ejemplo cambiar el nombre a una variable).
- Permite introducir automáticamente tabulaciones y espaciados para aumentar la legibilidad.
- Depuración: seguimiento de variables, puntos de ruptura y mensajes de error del intérprete.
- Aumentar las funcionalidades a través de la gestión de sus módulos y plugins.
- Administrar las interfaces de usuario (menús y barras de herramientas).
- Administrar configuraciones del usuario.

3. Entornos integrados libres y propietarios.

Entornos Integrados Libres

Son aquellos con licencia de uso público. El autor deja libertad a los usuarios, por lo que el programa puede ser usado, copiado, estudiado, modificado y redistribuido libremente sin ser obligatoriamente gratis. Podemos encontrarnos programas bajo esta licencia que son de pago, aunque suelen ser muy económicos.

Tipos de entornos de desarrollo libres más relevantes en la actualidad: NetBeans, Eclipse, Gambas, Anjuta, Geany, GNAT Studio…

Entornos Integrados Propietarios

Son aquellos entornos integrados de desarrollo que necesitan licencia para ser usado, copiado, estudiado, modificado o redistribuido, es lo contrario a software libre. También puede existir gratuito y de pago.

Tipos de entornos de desarrollo propietarios más relevantes en la actualidad: Microsoft Visual Studio, FlashBuilder, C++ Builder, Turbo C++ profesional, Jbuilder, Jcreator, Xcode, C#, ActionScript…

4. Estructura de entornos de desarrollo.

Los entornos de desarrollo, ya sean libres o propietarios, están formados por una serie de componentes software que determinan sus funciones. Estos componentes son:

- Editor de textos: Resalta y colorea la sintaxis, tiene la función de autocompletar código, ayuda y listado de parámetros de funciones y métodos de clase. Inserción automática de paréntesis, corchetes, tabulaciones y espaciados.
- Compilador/intérprete: Detección de errores de sintaxis en tiempo real.
- Depurador: Botón de ejecución y traza, puntos de ruptura y seguimiento de variables.
- Generador automático de herramientas: Para la visualización, creación y manipulación de componentes visuales y todo un arsenal de asistentes y utilidades de gestión y generación código.
- Interfaz gráfica: Nos brinda la oportunidad de programar en varios lenguajes con un mismo IDE. Puede acceder a innumerables bibliotecas y plugins aumentando las opciones de nuestros programas.

5. Configuración y personalización de entornos de desarrollo.

Podemos personalizar la configuración del entorno sólo para el proyecto actual, o bien para todos los proyectos, presentes y futuros.

Parámetros configurables del entorno:

- Carpeta o carpetas donde se alojarán todos 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: compilar al grabar, generar información de depuración.

- Opciones de empaquetado de la aplicación: nombre del archivo empaquetado y momento del empaquetado.
- Opciones de generación de documentación asociada al proyecto.
- Descripción de los proyectos, para una mejor localización de los mismos.
- Opciones globales de formato del editor: número de espaciados en las sangrías, color de errores de sintaxis, color de etiquetas, opción de autocompletado de código, propuestas de insertar automáticamente código.
- Opciones de combinación de teclas en teclado. Etc.

6. Gestión de módulos y plugins.

Un módulo es un componente software que contiene clases de Java que pueden interactuar con las API del entorno de desarrollo y el manifest file, que es un archivo especial que lo identifica como módulo. Se considera que un plugin es normalmente un conjunto de módulos. En NetBeans se encuentran en Tools, Plugins

1. Añadir

Para añadir un nuevo módulo tenemos varias opciones:

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

Una cuarta posibilidad es crear el módulo nosotros mismos, sin embargo, lo más usual es añadir los módulos o plugins que realmente nos interesan desde la web oficial de NetBeans. El plugin se descarga en formato .nbm que es el propio de los módulos en NetBeans. Posteriormente, desde nuestro IDE, cargaremos e instalaremos esos plugins. A esta manera de añadir módulos se le conoce como adición off-line.

La adición on-line requiere tener instalado el plugin Portal Update Center y consiste en instalar complementos desde nuestro mismo IDE sin tener que descargarlos previamente.

2. Eliminar.

Eliminar un módulo requiere seguir los siguientes pasos:

- Encontrar el módulo o plugin dentro de la lista de complementos instalados en el entorno.
- A la hora de eliminarlo, tenemos dos opciones:
  - Desactivarlo: El módulo o plugin sigue instalado, pero en estado inactivo (no aparece en el entorno).
  - Desinstalarlo: El módulo o plugin se elimina físicamente del entorno de forma permanente.

3. Funcionalidades.

Podemos clasificar las distintas categorías de funcionalidades de módulos y plugins en los siguientes grupos:

- Construcción de código: facilitan la labor de programación.
- Bases de datos: ofrecen nuevas funcionalidades para el mantenimiento de las aplicaciones.
- Depuradores: hacen más eficiente la depuración de programas.
- Aplicaciones: añaden nuevas aplicaciones que nos pueden ser útiles.
- Edición: hacen que los editores sean más precisos y más cómodos para el programador.
- Documentación de aplicaciones: para generar documentación de los proyectos en la manera deseada.
- Interfaz gráfica de usuario: para mejorar la forma de presentación de diversos aspectos del entorno al usuario.
- Lenguajes de programación y bibliotecas: para poder programar bajo un Lenguaje de Programación que, en principio, no soporte la plataforma.
- Refactorización: hacer pequeños cambios en el código para aumentar su legibilidad, sin alterar su función.
- Aplicaciones web: para introducir aplicaciones web integradas en el entorno.
- Prueba: para incorporar utilidades de pruebas al software.

4. Herramientas concretas.

- Importador de Proyectos de NetBeans: permite trabajar en lenguajes como Jbuilder.
- Servidor de aplicaciones GlassFish: Proporciona una plataforma completa para aplicaciones de tipo empresarial.
- Soporte para Java Enterprise Edition: Cumplimiento de estándares, facilidad de uso y la mejora de rendimiento hacen de NetBeans la mejor herramienta para crear aplicaciones de tipo empresarial de forma ágil y rápida.
- Facilidad de uso a lo largo de todas las etapas del ciclo de vida del software.
- NetBeans Swing GUI builder: simplifica mucho la creación de interfaces gráficos de usuarios en aplicaciones cliente y permite al usuario manejar diferentes aplicaciones sin salir del IDE.
- NetBeans Profiler: Permite ver de forma inmediata ver cómo de eficiente trabajará un trozo de software para los usuarios finales.
- El editor WSDL facilita a los programadores trabajar en servicios Web basados en XML.
- El editor XML Schema Editor permite refinar aspectos de los documentos XML de la misma manera que el editor WSDL revisa los servicios Web.
- Aseguramiento de la seguridad de los datos mediante el Sun Java System Access Manager.
- Soporte beta de UML que cubre actividades como las clases, el comportamiento, la interacción y las secuencias.

- Soporte bidireccional, que permite sincronizar con rapidez los modelos de desarrollo con los cambios en el código conforme avanzamos por las etapas del ciclo de vida de la aplicación. Etc.

7. Uso básico de entornos de desarrollo.

Cada proyecto comprende una serie de archivos y bibliotecas que lo componen.

- El principal archivo del proyecto Java es el llamado Main.java.
- Ventana derecha: espacio de escritura de los códigos de los proyectos.
- Barra de herramientas: Desde aquí podremos acceder a todas las opciones del IDE.
- Mientras escribimos en el editor de textos nos percatamos de varias características de NetBeans que ya hemos señalado en páginas anteriores:
  - Autocompletado de código.
  - Coloración de comandos.
  - Subrayado en rojo cuando hay algún error y posibilidad de depuración y corrección de forma visual, mediante un pequeño icono que aparece a la izquierda de la línea defectuosa.
- Una vez tenemos el código plasmado en la ventana de comandos y libre de errores de sintaxis, los siguientes pasos son:
  - Compilación
  - Depuración
  - Ejecución.