UT2. Instalación y uso de entornos de desarrollo .

Entornos de desarrollo

ÍNDICE DE CONTENIDOS

Definición de IDE.

Un Entorno de Desarrollo (IDE, en inglés, Integrated Development Environment) es un software compuesto por una serie de herramientas que utilizan los programadores para desarrollar código.

- Puede dar soporte a un único lenguaje de programación o a varios
- Las herramientas que normalmente componen un IDE son
  - ✓ Editor del código de programación
  - ✓ Compilador
  - ✓ Intérprete
  - ✓ Depurador
  - ✓ Constructor del interfaz gráfico

Evolución histórica de los IDE

Los primeros entornos de desarrollo integrados nacen a principios de los años 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.

Normalmente, un IDE está dedicado a un determinado lenguaje de programación. No obstante, las últimas versiones de los IDEs tienden a ser compatibles con varios lenguajes (por ejemplo, Eclipse, NetBeans, Microsoft Visual Studio...) mediante la instalación de plugins adicionales.

Evolución histórica de los IDE

En las décadas de utilización de la tarjeta perforada como sistema de almacenamiento el concepto de Entorno de Desarrollo Integrado sencillamente no tenía sentido.

Los programas estaban escritos con diagramas de flujo y entraban al sistema a través de las tarjetas perforadas. Posteriormente, eran compilados.

El primer lenguaje de programación que utiliza un IDE fue el BASIC.

Este primer IDE estaba basado en consola de comandos exclusivamente. Sin embargo, el uso que hace de la gestión de archivos, compilación, depuración… es perfectamente compatible con los IDE actuales.

A nivel popular, el primer IDE puede considerarse que fue el IDE llamado Maestro I. Nació a principios de los 70 y fue instalado por unos 22000 programadores en todo el mundo. Lideró el campo durante los años 70 y 80.

El uso de los entornos integrados de desarrollo se ratifica y afianza en los 90 y hoy en día contamos con infinidad de IDE, tanto de licencia libre como no.

Evolución histórica de los IDE

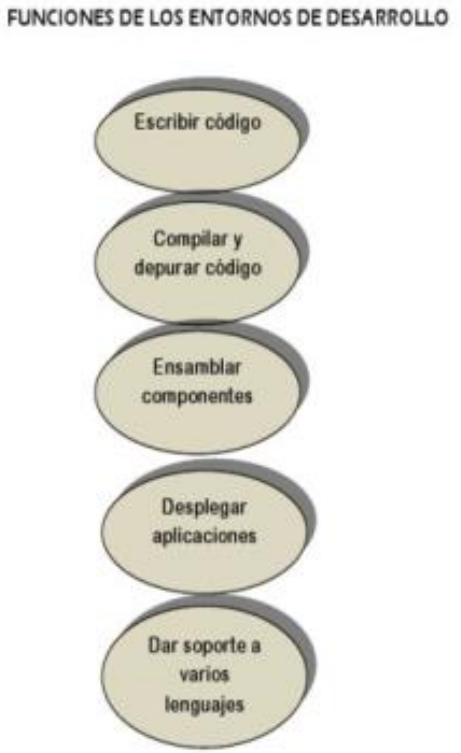
Entorno de desarrollo	Lenguajes que soporta	Tipo de licencia
NetBeans	C/C++, Java, JavaScript, PHP, Python	De uso público.
Eclipse	Ada, C/C++, Java, JavaScript, PHP	De uso público.
Microsoft Visual Studio.	Basic, C/C++, C#	Propietario
C++ Builder	C/C++.	Propietario
JBuilder	Java	Propietario

Funciones de un Entorno de Desarrollo

Como sabemos, los entornos de desarrollo están compuestos por una serie de herramientas software de programación, necesarias para la consecución de sus objetivos. Estas herramientas son:

- ✓ Un editor de código fuente.
- ✓ Un compilador y / o un intérprete.
- ✓ Automatización de generación de herramientas.
- ✓ Un depurador.

Principales funciones de un Entorno de Desarrollo



- ✓Editor de código: coloración de la sintaxis.
- ✓Auto-completado de código, atributos y métodos de clases.
- ✓Identificación automática de código.
- ✓Herramientas de concepción visual para crear y manipular componentes visuales.
- ✓Asistentes y utilidades de gestión y generación de código.
- ✓Archivos fuente en unas carpetas y compilados a otras.
- ✓Compilación de 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.
- ✓Soporta cambios de varios usuarios de manera simultánea.
- ✓Generador de documentación integrado.
- ✓Detección de errores de sintaxis en tiempo real.

Otras funciones de un Entorno de Desarrollo.

- ✓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.
- ✓Aumento de funcionalidades a través de la gestión de sus módulos y plugins.
- ✓Administración de las interfaces de usuario (menús y barras de herramientas).
- ✓Administración de las configuraciones del usuario.

Entornos Integrados Libres y Propietarios

Entornos Integrados Libres

- Son aquellos con licencia de uso público.
- No hay que pagar por ellos, y aunque los más conocidos y utilizados son Eclipse y NetBeans, hay bastantes más.

IDE	Lenguajes que soporta	Sistema Operativo
NetBeans	C/C++, Java, JavaScript, PHP, Python	Windows, Linux, Mac OS X
Eclipse	Ada, C/C++, Java, JavaScript, PHP	Windows, Linux, Mac OS X
Gambas	Basic	Linux
Anjuta	C/C++, Python, Javascript	Linux
Geany	C/C++, Java.	Windows, Linux, Mac OS X
GNAT Studio	Fortran	Windows, Linux, Mac OS X

Entornos Integrados Libres y Propietarios

Entornos Integrados Propietarios

Son aquellos entornos integrados de desarrollo que necesitan licencia. No son free software, hay que pagar por ellos.
El más conocido y utilizado es Microsoft Visual Studio, que usa el framework .NET y es desarrollado por Microsoft.

IDE	Lenguajes que soporta	Sistema Operativo
Microsoft Visual Studio	Basic, C/C++, C#	Windows
FlashBuilder	ActionScript	Windows, Mac OS X
C++ Builder	C/C++	Windows
Turbo C++ profesional	C/C++	Windows
Jbuilder	Java	Windows
JCreator	Java	Windows, Linux, Mac OS X
Xcode	C/C++, Java	Mac OS X

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: Se 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. Características de refactorización.

Depurador: Botón de ejecución y traza, puntos de ruptura y seguimiento de variables. Opción de depurar en servidores remotos.

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. Es una interfaz agradable que puede acceder a innumerables bibliotecas y plugins, aumentando las opciones de nuestros programas.

Configuración y personalización de entornos de desarrollo.

Una vez tenemos instalado nuestro entorno de desarrollo podemos acceder a personalizar su configuración.

Al abrir un proyecto existente, o bien crear un nuevo proyecto, seleccionaremos un desplegable con el nombre de “configuración” desde el que podremos personalizar distintas opciones del proyecto.

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

Configuración y personalización de entornos de desarrollo.

Parámetros configurables del entorno:

- ✓ Carpeta/s donde se alojarán todos los archivos de los proyectos (es importante la determinación de este parámetro, para tener una estructura de archivos ordenada).
- ✓ 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 (con extensión .jar, que es la extensión característica de este tipo de archivos empaquetados) 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.

Gestión de módulos.

Un **módulo** es un componente software que contiene clases de Java que pueden interactuar con las APIs del entorno de desarrollo y el manifest file, que es un archivo especial que lo identifica como módulo.

En la página oficial de NetBeans encontramos una relación de módulos y plugins, divididos en categorías.

Seleccionando la categoría Lenguajes de Programación, encontraremos aquellos módulos y plugins que nos permitan añadir nuevos lenguajes soportados por nuestro IDE.

Los módulos se pueden construir y desarrollar de forma independiente. Esto posibilita su reutilización y que las aplicaciones puedan ser construidas a través de la inserción de módulos con finalidades concretas. Por esta misma razón, una aplicación puede ser extendida mediante la adición de módulos nuevos que aumenten su funcionalidad.

Existen en la actualidad multitud de módulos y plugins disponibles para todas las versiones de los entornos de desarrollo más utilizados.

Gestión de módulos. Funcionalidades

Los módulos y plugins disponibles para los entornos de desarrollo, en sus distintas versiones, tienen muchas y muy variadas funciones.

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.

Actualización y mantenimiento de entornos de desarrollo

El mantenimiento del entorno de desarrollo es una tarea fundamental que requiere tener todos sus componentes periódicamente actualizados.

También es de vital importancia realizar copias de seguridad sobre las bases de datos de nuestros proyectos por si ocurriera algún error o proceso defectuoso poder restaurarlos.

Instalación de Entornos Integrados de Desarrollo

- INSTALACIÓN DEL JDK
- INSTALACIÓN DE NETBEANS