Instalación y uso de entornos de desarrollo .

Entornos de desarrollo

ÍNDICE DE CONTENIDOS

- Definición de IDE.
- Evolución histórica.
- Funciones de un entorno de desarrollo.
- Entornos propietarios y libres.
- Estructura de Entornos de Desarrollo
- Instalación de Entornos Integrados de Desarrollo
- Configuración y personalización de entornos de desarrollo.
- Actualización y mantenimiento de entornos de desarrollo

Definición de IDE.

Un Entorno de Desarrollo (IDE, en inglés, Integrated Development Environment) es un software compuesto por una serie de herramientas que ayudan al desarrollo de software mediante, las cuales están enfocadas a dicho fin.

Este conjunto de herramientas pueden estar enfocadas a dar soporte a un único lenguaje de programación o a varios.

Las principales herramientas que constituyen un IDE son:

- ✓ Editor
- ✓ Compilador o Intérprete
- ✓ Depurador
- ✓ Constructor de interfaz gráfico (GUI)

Definición de IDE.

Un Entorno de Desarrollo (IDE, en inglés, Integrated Development Environment) es un software compuesto por una serie de herramientas que ayudan al desarrollo de software mediante, las cuales están enfocadas a dicho fin.

Este conjunto de herramientas pueden estar enfocadas a dar sopor a un único lenguaje de programación o a varios.

Las principales herramientas que constituyen un IDE son:

- ✓ **Editor**: Se utilizan editores que colorean la sintaxis para ayudar al programador a comprender mejor el programa y facilitando la detección de errores.
- ✓ Compilador o Intérprete
- ✓ Depurador
- ✓ Constructor de interfaz gráfico (GUI)

Definición de IDE.

Un Entorno de Desarrollo (IDE, en inglés, Integrated Development Environment) es un software compuesto por una serie de herramientas que ayudan al desarrollo de software mediante el uso de una serie de herramientas enfocadas a dicho fin.

Este conjunto de herramientas pueden estar enfocadas a dar sopor a un único lenguaje de programación o a varios.

Las principales herramientas que constituyen un IDE son:

- ✓ Editor
- ✓ **Compilador o Intérprete**: Vinculado al lenguaje utilizado, será de un tipo u otro.
- ✓ Depurador
- ✓ Constructor de interfaz gráfico (GUI)

Definición de IDE.

Un Entorno de Desarrollo (IDE, en inglés, Integrated Development Environment) es un software compuesto por una serie de herramientas que ayudan al desarrollo de software mediante el uso de una serie de herramientas enfocadas a dicho fin.

Este conjunto de herramientas pueden estar enfocadas a dar sopor a un único lenguaje de programación o a varios.

Las principales herramientas que constituyen un IDE son:

- ✓ Editor
- ✓ Compilador o Intérprete
- ✓ **Depurador**: Permite la ejecución de la aplicación paso a paso para poder inspeccionar, por ejemplo, valores de variables, etc. Un depurador necesita de un intérprete.
- ✓ Constructor de interfaz gráfico (GUI)

Definición de IDE.

Un Entorno de Desarrollo (IDE, en inglés, Integrated Development Environment) es un software compuesto por una serie de herramientas que ayudan al desarrollo de software mediante el uso de una serie de herramientas enfocadas a dicho fin.

Este conjunto de herramientas pueden estar enfocadas a dar sopor a un único lenguaje de programación o a varios.

Las principales herramientas que constituyen un IDE son:

- ✓ Editor
- ✓ Compilador o Intérprete
- ✓ Depurador
- ✓ **Constructor de interfaz gráfico (GUI)**: Mediante el uso de esta herramienta, el programador, podrá crear componentes gráficos (ventanas, botones, pestañas,..)

Definición de IDE.

Además de las herramientas anteriores, cada vez es más frecuente encontrar las siguientes herramientas dentro de un IDE:

- ✓ Control de versiones
- ✓ Importación/Exportación de programas

Definición de IDE.

Además de las herramientas anteriores, cada vez es más frecuente encontrar las siguientes herramientas dentro de un IDE:

- ✓ **Control de versiones**: Permite almacenar los cambios en nuestro código en repositorios remotos y compartirlo con otros programadores.
- ✓ Importación/Exportación de programas

Definición de IDE.

Además de las herramientas anteriores, cada vez es más frecuente encontrar las siguientes herramientas dentro de un IDE:

- ✓ Control de versiones
- ✓ **Importación/Exportación de programas**: Permite la portabilidad de nuestro programa entre equipos.

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 utilizó un IDE fue el BASIC, Visual 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’s 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	Plataforma
Xcode	Swift, C/C++, Java, Python, Ruby, …	De uso público.	Mac OS X
NetBeans	C/C++, Java, JavaScript, PHP, Python	De uso público.	Multiplataforma
Eclipse	Ada, C/C++, Java, JavaScript, PHP	De uso público.	Multiplataforma
Visual Studio Code	Basic, C/C++, C#	De uso público.	Multiplataforma
JDeveloper	C/C++.	Propietario	Multiplataforma
JBuilder	Java	Propietario	Multiplataforma

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

Framework

Todas las aplicaciones informáticas tienen una estructura que facilita su gestión.

Esta estructura esta constituida por distintos tipos de ficheros: código fuente, ficheros configuración, ficheros binarios, librerías, ejecutables, …

Un **framework** está formado por un conjunto de librerías, herramientas y/o módulos que nos permiten desarrollar aplicaciones en un contexto determinado.

Generalmente no están vinculadas a un lenguaje de programación específico, a mayor nivel de detalle, mayor será la obligatoriedad de vincularse con un lenguaje determinado. Ejemplos:

- Modelo-Vista-Controlador (MVC)
- Laravel

Framework

Todas las aplicaciones informáticas tienen una estructura que para facilita su gestión.

Esta estructura esta constituida por distintos tipos de ficheros: código fuente, ficheros configuración, ficheros binarios, librerías, ejecutables, …

Un **framework** está formado por un conjunto de librerías, herramientas y/o módulos que nos permiten desarrollar aplicaciones en un contexto determinado.

Generalmente no están vinculadas a un lenguaje de programación específico, a mayor nivel de detalle, mayor será la obligatoriedad de vincularse con un lenguaje determinado. Ejemplos:

- **Modelo-Vista-Controlador (MVC)**: Es un sencillo patrón de diseño que nos permite separa las funciones de nuestra aplicación en capas: presentación (vista), datos (modelos) y operaciones (controlador)
- Laravel

Framework

Todas las aplicaciones informáticas tienen una estructura que para facilita su gestión.

Esta estructura esta constituida por distintos tipos de ficheros: código fuente, ficheros configuración, ficheros binarios, librerías, ejecutables, …

Un **framework** está formado por un conjunto de librerías, herramientas y/o módulos que nos permiten desarrollar aplicaciones en un contexto determinado.

Generalmente no están vinculadas a un lenguaje de programación específico, a mayor nivel de detalle, mayor será la obligatoriedad de vincularse con un lenguaje determinado. Ejemplos:

- Modelo-Vista-Controlador (MVC)
- **Laravel**: Es un framework específico para el lenguaje de programación PHP. Este facilita la creación de sitios webs.

Pluging

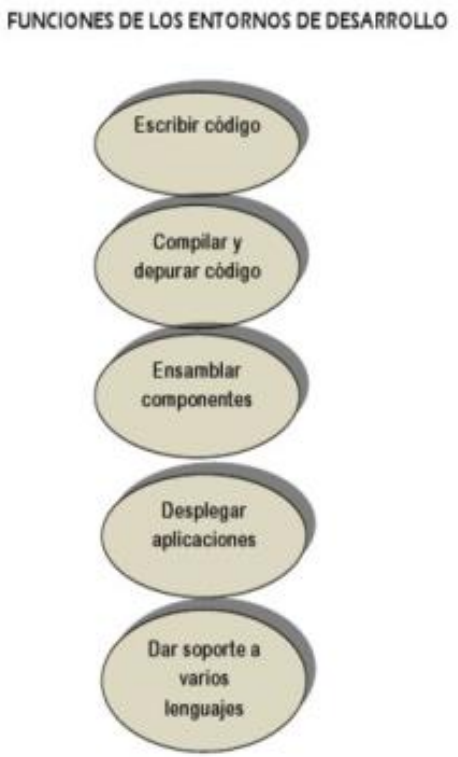
Pluging o plug-in es un programa informático que mejora o amplía la funcionalidad del programa o aplicación principal.

Los plugings pueden ser desarrollados tanto por empresas como por terceros. La principal diferencia radica en que este último no dispone de certificado de seguridad.

Existen gran variedad de aplicaciones cotidianas o famosas que suelen incluirnos:

- Navegadores web.
- Sistemas gestores de contenidos (CMS).
- Reproductores de audio y video

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.

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.

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.