

MP0488.

**Desarrollo de interfaces de usuario
UF2. Herramientas de Edición**

**2.1. Interfaces de
usuario multiplataforma**

Índice

☰	Objetivos	3
☰	El desarrollo de interfaces multiplataforma	4
☰	IDE's y MDA	7
☰	UI multiplataforma: Propietarias y Libres	11
☰	IDE: Entorno de Desarrollo Integrado	16
☰	Resumen	24

Objetivos

En esta lección perseguimos los siguientes objetivos.

- 1 Conocer los posibles entornos para desarrollo de interfaces multiplataforma, tanto propietarios como libres.
- 2 Ajustar la configuración lógica del sistema, o entorno de desarrollo, analizando las necesidades y criterios establecidos para configurar y explotar sistemas informáticos.
- 3 Seleccionar y emplear técnicas, motores y entornos de desarrollo, evaluando sus posibilidades, para participar en el desarrollo de aplicaciones, desarrollo de websites y aplicaciones en teléfonos, PDA y otros dispositivos móviles.
- 4 Seleccionar y emplear lenguajes, herramientas y librerías, interpretando las especificaciones, para desarrollar aplicaciones multiplataforma con acceso a bases de datos.

¡Ánimo y adelante!

El desarrollo de interfaces multiplataforma

Las interfaces multiplataforma tienen una calidad superior a los sites tradicionales en diseño, capacidad, rendimiento y funcionalidades. Además, pueden utilizarse en cualquier dispositivo, independientemente del sistema operativo.

Una **plataforma** consiste en una combinación de hardware y software que se usa para ejecutar aplicaciones de software.

Así, una **multiplataforma** es un atributo que solemos utilizar para denominar a los programas, métodos y conceptos de cómputo que son implementados y operan entre sí en múltiples plataformas informáticas.

El software multiplataforma puede dividirse en dos tipos:

1

Requiere una compilación individual para cada plataforma que le da soporte, como es el caso de .NET para los ordenadores y iOS para los dispositivos móviles.

2

Se puede ejecutar directamente en cualquier plataforma sin preparación especial, como ocurre con PHP en los ordenadores y Android en los dispositivos móviles.

Si tomamos, por ejemplo, una aplicación multiplataforma de última generación podemos ejecutarla en Microsoft Windows en la arquitectura x64, Linux en la arquitectura x64 y Mac OS X basados en x64, también en Android y iOS, sin ningún tipo de problema.

La principal ventaja de las interfaces multiplataforma es que funcionan en los principales sistemas operativos en los ordenadores de sobremesa, servers, laptops, smartphones y tablets, eliminando la necesidad de crear una versión distinta para cada dispositivo.

Basándonos en los conceptos de polivalencia y versatilidad, este tipo de interfaces ofrecen una serie de ventajas:

- Gran ahorro de tiempo y costes económicos.
- Posibilidad de desarrollar interfaces con la misma calidad y prestaciones que las ya desarrolladas.
- Máxima integración con el hardware y el software específicos de cada dispositivo.

Pero, antes de entrar en materia, ¿sabemos qué claves existen en el desarrollo de interfaces multiplataforma?

Las claves principales en el desarrollo de interfaces multiplataforma son:

Formarse adecuadamente

Con las nuevas plataformas no es necesario ser un experto programador, lo que no significa que no se deba dominar aspectos como el diseño, el funcionamiento de los distintos módulos para añadir funciones adicionales o las técnicas de integración e interacción con otras interfaces de aplicaciones o con el propio hardware del dispositivo.

Buscar la máxima interconexión con el dispositivo

Possibilidad de interactuar tanto con el software del dispositivo (Facebook, Twitter y otras redes sociales o programas), como con ciertos componentes del hardware (micro, cámara o GPS).

Aplicar los últimos avances tecnológicos

El futuro de Internet es móvil y 100% accesible. Los cálculos (actuales) de porcentaje de penetración de navegación a través de móviles inteligentes y tablets han alcanzado el 70% en ciertos sectores, lo que ha permitido crear un nuevo perfil de usuario con unas expectativas y necesidades nuevas, y una alta exigencia y demanda de funcionalidades muy avanzadas tecnológicamente y totalmente conectadas con los dispositivos (geolocalización o realidad aumentada, por ejemplo).

Elegir la plataforma adecuada

Para empezar a crear interfaces válidas para cualquier sistema móvil (iOs, Android, Blackberry, Windows Phone o Linux) o fijo (Windows, Linux o MacOS) sin tener que crear versiones diferentes, es necesario utilizar alguna plataforma o programa específico para el desarrollo de interfaces multiplataforma que se amolde a nuestros conocimientos previos, objetivos y actitudes.



Si hablamos de **entornos de desarrollo multiplataforma**, nos referimos a los que posibilitan crear aplicaciones de escritorio en cualquier plataforma y en un lenguaje de programación cuyos resultados no lleguen a despistar al usuario.

El control en un entorno puede tener una funcionalidad y una forma de interactuar con él, mientras que en el otro entorno esa manera de trabajar puede no ser la habitual.

IDE's y MDA

IDE's: un concepto, una realidad.

Los denominados IDE's para programadores (Entornos de Desarrollo Integrados), son un tipo de software que facilita la escritura de código para el desarrollo de interfaces o aplicaciones, y permite utilizar un lenguaje o programación interactivos para que todo el proceso de desarrollo del software tenga la mayor productividad.

Los IDE's no deben confundirse con un editor de texto de código fuente.

En la actualidad existen muchos IDE's. Estos entornos deben disponer básicamente de un conjunto de herramientas como:

- Gestión avanzada de proyectos y soporte para múltiples lenguajes.
- Editor de código fuente con autocompletador inteligente y resaltador de sintaxis.
- Compilador o intérprete, ejecutador y depurador (con analizador de errores).
- Constructor de GUI, navegador de clases u objetos y métodos.
- Frameworks y otros.



Arquitecturas basadas en modelos (MDA)

A lo largo del tiempo, la creciente complejidad de las UI para el manejo de unas aplicaciones a su vez más y más complejas ha obligado a la utilización de herramientas cada vez con un mayor nivel de abstracción que puedan generar el mismo diseño para distintas plataformas de forma automática o semiautomática.

Este desarrollo ha evolucionado desde la programación de UI, usando lenguajes de propósito general, hasta las actuales aproximaciones basadas en **modelos**, siguiendo las líneas definidas dentro del campo de las **arquitecturas guiadas por modelos**, es decir, técnicas o métodos denominados **MDA (Model Driven Architecture)**, que abordan el modelado de la interacción de sistemas software, incluyendo la interacción de aplicaciones web, dispositivos móviles, sistemas domóticos, etc.

Para evitar estos problemas y sus consecuencias -principalmente económicas y de tiempo de desarrollo- se han introducido una serie de **modelos (conceptuales)** que especifican todos los **aspectos estáticos, dinámicos y de implementación**. Son los siguientes:

1

OO-Method. Método de producción que permite la generación automática de sistemas software, orientado a objetos y con un marco para la construcción de sistemas de información organizada. Cubre todas las fases del proceso de desarrollo de software.

2

OOWS (Object Oriented Web Solution). Metodología basada en el paradigma MDA para modelar la interacción con el usuario para entornos web. Extiende a OO-Method introduciendo la noción de navegación y de modelo de presentación más adecuado para entornos web, cuya estrategia de generación de código está basada en modelos, evitando problemas con las primitivas de los entornos web.

En vez de definir un nuevo modelo desde 0, se parte de experiencias previas que ya han sido contrastadas, cuyo modelo de interacción es totalmente independiente de la plataforma destino, para generar una UI adecuada.

Aproximaciones basadas en modelos.

A lo largo de la última década, se han presentado **distintas aproximaciones para el diseño de UI basadas en modelos**. La mayoría de estas herramientas proceden del ámbito académico y, en general, sólo presentan prototipos de las [herramientas CASE](#) para soportar los métodos de diseño y especificación. Aunque ya se pueden encontrar algunos ejemplos de herramientas comerciales basadas en modelos como **WebRatio²**, **VisualWADE³** u **Olivanova⁴**, **TRIDENT**, **OVID**, **UIDE**, **JANUS**, **CTT**, **MOBI-D**, **Wisdom**, **IDEAS**, **Just-UI**, **OO-H**, **UMLi** y **UWE**.

Las últimas versiones de estas herramientas, como puede ser **Olivanova⁹**, permiten automáticamente la generación de aplicaciones basadas en formularios.

¿Interfaz nativa o diseño propio?

En el desarrollo de aplicaciones hay dos tendencias claras: utilizar las interfaces nativas que cada sistema operativo ofrece o hacer uso de un desarrollo propio que lo haga independiente del sistema operativo.

¿Qué opción es mejor? Teniendo en cuenta que las interfaces nativas se basan en elementos preestablecidos (botones, listas, encabezados...), cada plataforma o sistema operativo, ya definidos en cuanto a las características básicas, puede ajustarse en mayor o menor medida para que se corresponda con la estética buscada.

El inconveniente de las interfaces nativas se encuentra en que limitan la personalidad del diseño y en algunos casos es necesario ir un paso más allá. Para desarrollar una interfaz personalizada hay que planearlo de antemano, porque representa una mayor complejidad y tiempo de desarrollo. **En la mayoría de los casos no se trata de elegir entre una u otra, sino de alcanzar el balance adecuado combinando ambas.**

Además del objetivo o del tipo de aplicación, las interfaces nativas tienen un punto a favor porque **están constituidas por elementos que el usuario ya conoce y a los que está habituado**, de modo que no representan un nuevo aprendizaje.

UI multiplataforma: Propietarias y Libres

Las ventajas de algunos entornos pueden ser desventajas en otros.

RAD (Rapid Application Development)

En la actualidad disponemos de un extenso abanico de herramientas para realizar prototipos de UI, ya que a lo largo de los últimos años se han multiplicado. En estos momentos prácticamente nadie se aventura a diseñar UI o GUI sin realizar una tarea tan básica como es iterar en base a prototipos, evaluados y refinados gracias a las distintas posibilidades. Estas herramientas de diseño de UI (incluidas las GUI) se encuadran dentro de las denominadas RAD (Rapid Application Development).

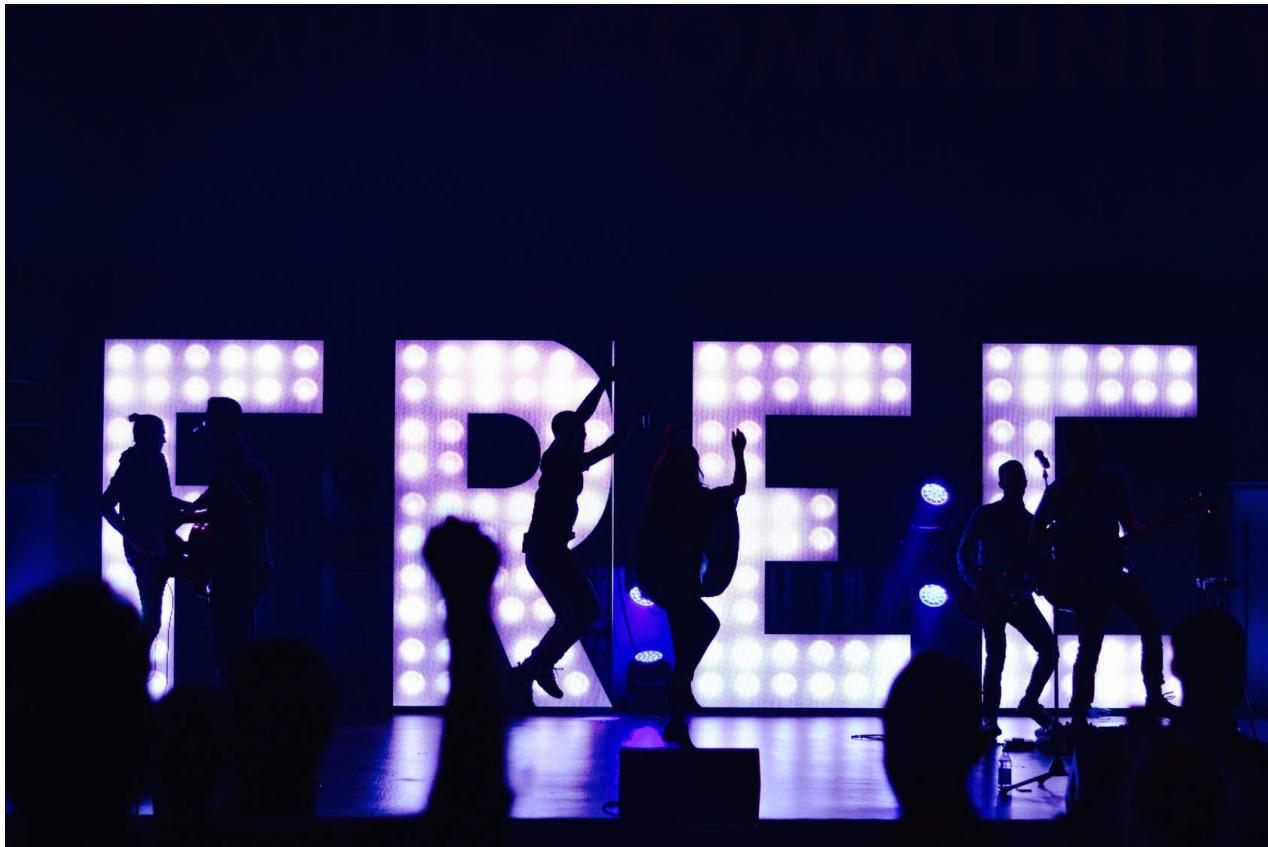
Como ya sabemos, disponemos de una gran variedad de opciones en cuanto a software se refiere, por lo que podemos emplear programas comerciales que nos facilitan realizar tareas. Sin embargo, **¿qué ocurre cuando queremos compartir ese mismo software con alguien o modificarlo para adaptarlo a nuestras necesidades?** Simplemente no es posible porque no tenemos acceso al código fuente. Además, si distribuimos dicho software sin permiso del autor o autores estaremos incurriendo en un delito. En este punto podríamos comenzar a hablar del software libre cuya distribución, uso y modificación es perfectamente legal.

Software Libre

Su dueño renuncia a la posibilidad de obtener pago alguno por las licencias, patentes o cualquier forma que adopte su derecho de propiedad sobre él, pudiendo ser usado, copiado, estudiado, modificado y redistribuido libremente.

También está disponible gratuitamente o a precio de costo del medio en el que se distribuya.

En algunas ocasiones (no en todas) incluye el código fuente para su posible modificación.



[Free Software Foundation](#) (FSF): El proyecto más ambiciosamente libre.

A finales de 1.940, los ordenadores eran enormes en dimensiones y costes, y entre los usuarios altamente experimentados y especializados -no existían los usuarios convencionales- era muy común el intercambio de programas y códigos para disponer de las mejoras que se iban haciendo (conocidas como **hacks**).

Por aquel entonces tampoco existían las licencias de software, hasta que años más tarde las restricciones derivadas de las licencias de uso, la mayor parte implementadas por desarrolladores y grandes compañías, plantearon la necesidad de dividir el software entre libre y propietario. Tiempo después, el software libre se empezó a ver como una gran colección de pequeños fragmentos de código, susceptible de ser modificado y adaptado.

El software libre no ha dejado de crecer y multiplicarse a lo largo de estos años.

Características principales:

- Se puede encontrar disponible el código fuente, por lo que puede modificarse sin ningún límite.
- Ofrece libertad de uso para estudiarlo y adaptarlo, distribuir copias, mejorarlo y publicar los cambios.

VENTAJAS

- El usuario no comete delito por tenerlo o usarlo.
- Amplísima gama y variedad de herramientas libres, además de actualizaciones periódicas y una gran comunidad de apoyo y soporte.
- 100% libre de virus y alto nivel de estabilidad.
- Diversifica las soluciones informáticas, abarata los costes y da flexibilidad por su independencia tecnológica.

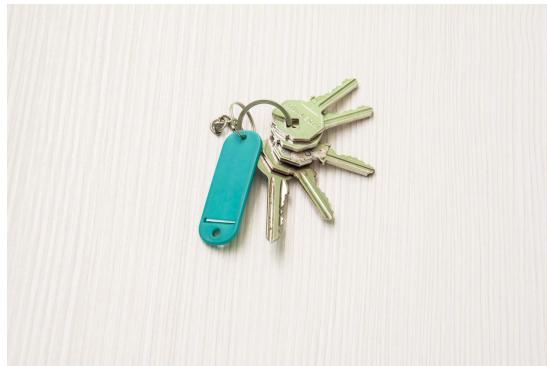
DESVENTAJAS

- Carece de una estructura de marketing, además de la falta de algunas aplicaciones específicas en el mercado actual.
- Requiere profesionales debidamente cualificados para la administración del sistema.
- Falta de garantía por parte del autor.
- Dificultad en el intercambio de archivos, instalación o configuración de los sistemas, con poca estabilidad y flexibilidad en el campo de multimedia y juegos, y menor compatibilidad con el hardware, que debe ser de calidad y disponer de estándares abiertos.
- Dispone de GUI o UI menos amigables, incluyendo el uso de las CLI.

Software Propietario

Es cualquier software en el que el usuario tiene limitaciones para usarlo, modificarlo o redistribuirlo.

También se llama **código cerrado**, concepto que se aplica a cualquier software no libre o semi-libre, ya sea porque tiene prohibido su uso, redistribución o modificación o porque requiere permiso expreso del titular del software.



En los años 60, los laboratorio Bell proporcionaron el código fuente de su sistema operativo UNIX y poco tiempo después comenzó a existir lo que se conoce como **software de código cerrado**. Ya en la década de los 70, sobre 1.972, el gobierno de EE.UU. obligó a IBM a distinguir entre software y hardware, dando lugar a los primeros intentos de cerrar el código de los programas y comenzar a comercializar el software para derecho de uso. En 1.975, [Bill Gates](#) y [Paul Allen](#) fundaron Microsoft, principal impulsor del software propietario. A partir del año 1.991, (nacimiento de Linux, creado por [Linus Torvalds](#)), IBM y Microsoft dejaron de cooperar en el desarrollo de sistemas operativos al desarrollar OS/2 y Windows.

Características principales:

- No se puede realizar ningún tipo de modificación del código fuente, ni distribuir el software sin el permiso del propietario.
- El usuario debe realizar cursos para el manejo del sistema como tal debido a su alta capacidad de uso, además de disponer de accesos para que un usuario implemente otro tipo de sistema en él.

VENTAJAS

- Propiedad y decisión de uso del software por parte de la empresa.
- Soporte para todo tipo de hardware, así como mayor compatibilidad en el terreno de multimedia y juegos; también una mayor compatibilidad con el hardware.
- Mejor acabado de la mayoría de aplicaciones y menor necesidad de técnicos especializados, con mejor protección de las obras con copyright y unificación de productos.
- Facilidad de adquisición, además de la existencia de programas diseñados específicamente para desarrollar tareas muy específicas.
- Interfaces gráficas mejor diseñadas, eliminando las CLI y mejorando enormemente las GUI.

DESVENTAJAS

- No existen aplicaciones para todas las plataformas (Windows, Linux y MacOS).
- Imposibilidad de copia, redistribución y modificación, así como restricciones en el uso marcadas por la licencia que se está pagando.
- Por lo general, suelen ser menos seguras.
- El coste de las aplicaciones es mayor. Disponen de un soporte exclusivo (casi siempre de pago) de la marca propietaria y depende de la empresa a un 100%.

IDE: Entorno de Desarrollo Integrado

Los IDE's (Entornos de Desarrollo Integrados) son un tipo de software que facilita la escritura de código para el desarrollo de interfaces o aplicaciones.

Permiten utilizar un **lenguaje o programación interactiva** para que todo el proceso de desarrollo del software tenga la mayor productividad.

(i) ¡Ojo! Los IDE no deben confundirse con un editor de texto de código fuente.

Realmente, los IDE proveen un marco de trabajo *amigable* para la mayoría de los lenguajes de programación (C++, PHP, Python, Java, C#, Delphi, etc.), los cuales disponen de herramientas integradas como editor de texto, compilador, intérprete, depurador, cliente, posibilidad de ofrecer un sistema de control de versiones y facilidad para ayuda en la construcción de GUIs.

Además, ofrecen ventajas como:

- **Multiplataforma.**
- Soporte para diversos lenguajes de programación de última generación y POO.
- **Integración con Sistemas de Control de Versiones** y recuperación de información.
- **Reconocimiento de sintaxis. Depurador y Extensiones/Componentes añadidos.**
- **Integración con Framework populares**, importación y exportación de proyectos.
- **Múltiples idiomas y manual de Usuarios y Ayuda.**

Muchos usuarios expertos y no tan avanzados opinan que el uso de este tipo de entornos de desarrollo llegan a *ensuciar* el código programado, ya que redundan en muchas líneas durante la implementación de la aplicación, disponen de formateo para el código, funciones específicas para renombrar variables y a otras funciones, warnings y errores de sintaxis en pantalla, herramientas de extracción de código a métodos o funciones nuevas, creación de proyectos (visualización gráfica), navegador para testeo de lo programado (no muy recomendable ya que tiene multitud de fallos), etc.

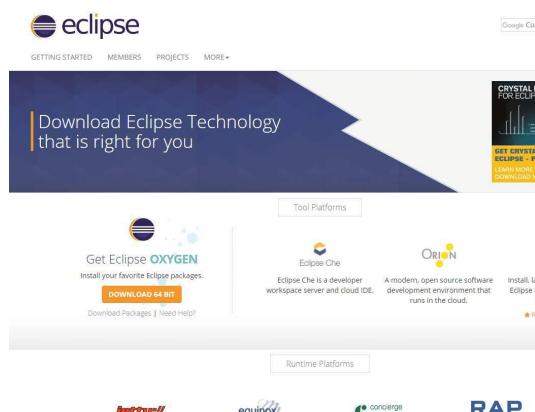
Veamos algunos de los entornos de desarrollo de GUIs y sus posibilidades de uso.

Eclipse

Es el más conocido y utilizado por los programadores, porque se trata de un entorno de código abierto y multiplataforma soportado por una comunidad de usuarios que brindan multitud de plugins.

Sirve para casi cualquier lenguaje de programación: JAVA, C++, PHP, Perl, etc. También nos permite realizar aplicaciones de escritorio y aplicaciones web, así que se caracteriza por una gran versatilidad.

Puedes descargarlo pulsando [aquí](#).

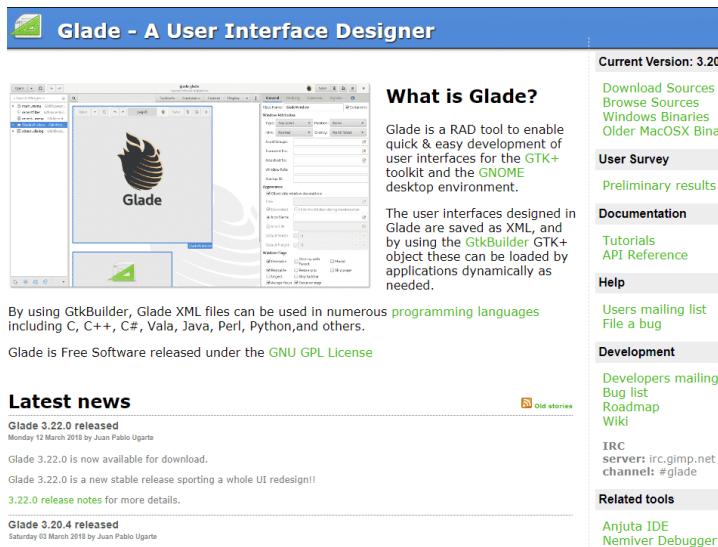


Glade

También conocido por **Glade Interface Designer**. Es una herramienta de desarrollo visual de GUIs mediante GTK/GNOME, independiente del lenguaje de programación.

No genera código fuente de forma predeterminada, sino un archivo XML. Por tanto, le procura versatilidad y potencia para convertirse en uno de los entornos más potentes del mercado.

Puedes descargarlo pulsando [aquí](#).



The screenshot shows the official website for Glade. At the top, there's a navigation bar with links like "Home", "About", "Downloads", "Documentation", "Help", and "Community". Below the navigation, there's a large image of the Glade application interface, which shows a window titled "glade-win" containing a graphical user interface with various buttons and labels. To the right of the image, there's a sidebar with the following sections:

- Current Version: 3.20**
- What is Glade?**: A brief description of Glade as a RAD tool for developing user interfaces using GTK+ and GNOME.
- Download Sources**, **Browse Sources**, **Windows Binaries**, **Older Mac OSX Binaries**
- User Survey**
- Preliminary results**
- Documentation**
- Tutorials**, **API Reference**
- Help**
- Users mailing list**, **File a bug**
- Development**
- Developers mailing list**, **Bug list**, **Roadmap**, **Wiki**
- IRC server: irc.gimp.net channel: #glade**
- Related tools**
- Anjuta IDE**, **Nemiver Debugger**

At the bottom left, there's a section for "Latest news" with links to "Glade 3.22.0 released" (Monday 12 March 2018) and "Glade 3.22.0 is now available for download.", as well as a link to "3.22.0 release notes". At the bottom right, there's a "Old stories" link.

NetBeans

Como el anterior, es muy utilizado debido a que se trata de un entorno multilenguaje y multiplataforma. Incluye el desarrollo de aplicaciones web y de escritorio, y también cuenta con plugins para trabajar en Android.

El lenguaje que mejor soporta es JAVA ya que fue creado por Oracle como IDE específicamente para él. Soporta JavaScript, HTML5, PHP, C/C++ etc.

Puedes descargarlo pulsando [aquí](#).



The screenshot shows the official website for NetBeans IDE. At the top, there's a navigation bar with links for "NetBeans IDE", "NetBeans Platform", "Enterprise", "Plugins", "Docs & Support", and "Community". Below the navigation, there's a large banner for "NetBeans IDE 8.2" with a "NEW!" badge, a "Learn More" button, and a "Download" button. The banner also features the tagline "Fits the Pieces Together" and the text "Quickly and easily develop desktop, mobile and web applications with Java, JavaScript, HTML5, PHP, C/C++ and more". Below the banner, there's a section for "Featured News:" with three items: "Support for Multiple Languages", "Rich Set of Community Provided Plugins", and "Best Support for Latest Java Technologies". Each news item has a small thumbnail image and a "More" link. At the bottom, there's a footer with links for "SiteMap", "About Us", "Contact", "Legal & Licences", and social media icons for Facebook, YouTube, Google+, and Twitter.

JetBrain

No es un entorno concreto, si no una compañía que desarrolla entornos de programación libre y crean entornos para multitud de lenguajes, como JAVA, Ruby, Python, PHP, SQL, Objective-C, C++ y JavaScript.

También están desarrollando IDE's para C# y GO.

Puedes descargarlo pulsando [aquí](#).

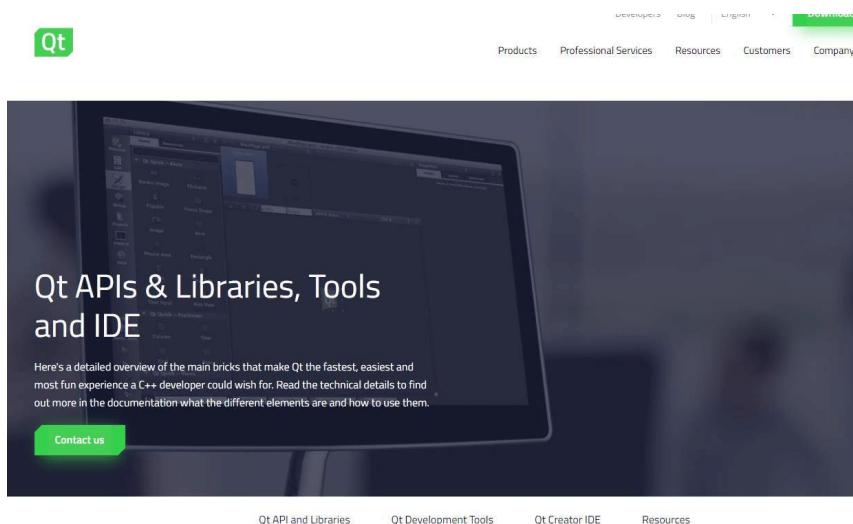


QtCreator

Es un entorno de programación multiplataforma para C++ principalmente, con un entorno amigable desarrollado en C++, JavaScript y QML.

Está diseñado específicamente para utilizar el framework de QT, lo que facilita aplicaciones multiplataforma de una manera sencilla y rápida.

Puedes descargarlo pulsando [aquí](#).

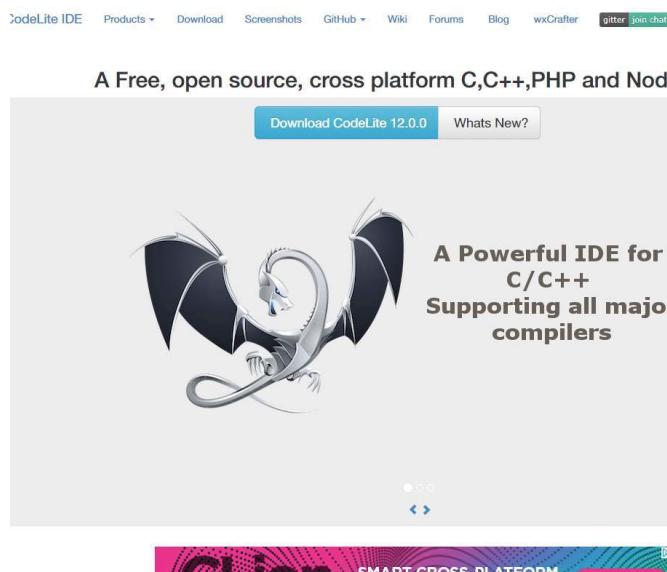


CodeLite

Es un IDE de código abierto y libre bajo la licencia GPU (*General Public License*) para diversos sistemas operativos. Utiliza wxWidgets para su GUI (herramienta libre).

Soporta los lenguajes C/C++, PHP y Node.js.

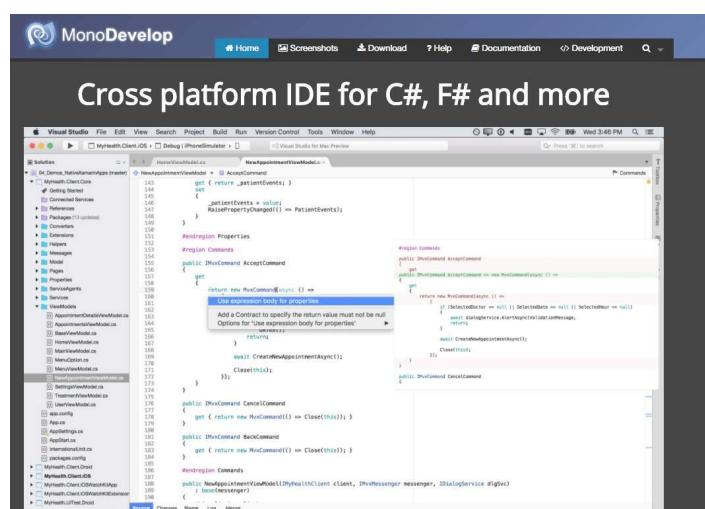
Puedes descargarlo pulsando [aquí](#).



Monodevelop

Diseñado para Linux, Windows y MacOS X, soporta lenguajes de programación como C#, Visual Basic.Net, C / C++, .Net y Vala.

Puedes descargarlo pulsando [aquí](#).



BlueJ

Es un entorno desarrollado como software libre y dirigido al aprendizaje de JAVA, en un entorno académico y sin uso a nivel profesional.

Es muy sencillo, porque incluye algunas funcionalidades dirigidas a las personas que están aprendiendo con el objetivo de que tengan mayor facilidad para comprender aspectos clave de la programación orientada a objetos.

Puedes descargarlo pulsando [aquí](#).

BlueJ

A free Java Development Environment designed for beginners, used by millions worldwide. Find out more...

"One of my favourite IDEs out there is BlueJ."
— James Gosling, creator of Java.

Created by



University of Kent

```
/**  
 * Add a student to this LabClass.  
 */  
public void enrollStudent(Student newStudent)  
{  
    if(students.size() == capacity) {  
        System.out.println("The class is  
    } else {  
        students.add(newStudent);  
    }  
}
```

○ ○ ●

Supported by ORACLE®

Download and Install

Version 4.1.2, released 9 November 2017 (adds an interactive tutorial and fixes compilation and printing bugs)

Windows



Requires Windows 7 or later.
[Download](#)

Mac OS X



Requires OS X 10.7.3.

Ubuntu/Debian



Please read the [Installation](#)

Other



Please read the [Installation](#)

A continuación vamos a ver algunos **Entornos de Desarrollo Integrados (IDEs)** propietarios.

Visual Studio

Fue diseñado por Microsoft como uno de los mejores entornos de programación para sus lenguajes de programación, como .Net, por ejemplo. Actualmente, Microsoft quiere pasarse al software libre, por lo que ha creado **Visual Studio Community**, muy parecido al Visual Studio de pago, pero soportado por la comunidad. Este entorno permite realizar aplicaciones web y de escritorio, y ayuda mucho al programador.

The screenshot shows the Microsoft Visual Studio download page. At the top, there's a navigation bar with links to Microsoft 365, Azure, Office 365, Dynamics 365, SQL, Windows 10, and 'Más'. A search bar is on the right. Below the navigation is a header for 'Descargas de Visual Studio' with 'Windows' and 'macOS' buttons. Four download cards are displayed:

- Visual Studio Community 2017**: IDE gratuito con todas las características para estudiantes, desarrolladores de código abierto y desarrolladores individuales. Includes a 'Descarga gratuita' button.
- Visual Studio Professional 2017**: Herramientas de desarrollo profesionales, servicios y ventajas para suscripción para equipos pequeños. Includes an 'Evaluación gratuita' button.
- Visual Studio Enterprise 2017**: Solución completa para satisfacer las exigentes necesidades de calidad y escala de equipos de todos los tamaños. Includes an 'Evaluación gratuita' button.
- Visual Studio Code**: Edición de código redefinida. Gratuito y de código abierto, que se ejecuta en cualquier parte. Includes a 'Descarga gratuita' button.

At the bottom of the page are three buttons: 'Descargar Visual Studio Preview' (with a download icon), 'Comparar las ediciones de Visual Studio' (with a compare icon), and 'Cómo instalar sin conexión' (with a connection icon).

Puedes descargarlo pulsando [aquí](#)

JBuilder IDE

Es un entorno preparado para el lenguaje de programación JAVA, de Borland, y CodeGear. Es un software comercial que permite desarrollos gráficos.

Posibilita el desarrollo RAD con JAVA, ya que está construido sobre Eclipse, e integra todos los beneficios de las soluciones OpenSource con el soporte de productividad y escalabilidad para sus aplicaciones de negocios corporativas con JAVA.



Build once and deploy modern apps for every platform - FAST



Puedes descargarlo pulsando [aquí](#)

Wingware Python IDE

Uno de los mejores IDE para el lenguaje de programación Python, cuyo principal problema podría ser el coste de la licencia: aproximadamente 200\$.



Puedes descargarlo pulsando [aquí](#)

JCreator

Es un software comercial, aunque se pueden obtener versiones de prueba o versiones simplificadas gratuitas de este IDE desarrollado en C++.

Omite herramientas para desarrollos gráficos, lo que lo hace más rápido y eficiente que otros IDEs.



Puedes descargarlo pulsando [aquí](#)

No hay un editor o IDE mejor que otro, cada uno tiene sus características, sus pros y contras. Además, en la mayoría de los casos hacen lo mismo.

Resumen

Has terminado la lección, repasemos los puntos más importantes.

- Las interfaces multiplataforma **tienen una calidad superior a los sites tradicionales** en diseño, capacidad, rendimiento y funcionalidades. Además, pueden utilizarse en cualquier dispositivo, independientemente de su sistema operativo.
- Los denominados **IDE** para programadores (**Entornos de Desarrollo Integrados**) consisten en un tipo de software que facilita la escritura de código para el desarrollo de interfaces o aplicaciones, y permite utilizar un lenguaje o programación interactiva para que todo el proceso de desarrollo del software tenga la mayor productividad. Los **IDE** **no deben confundirse con un editor de texto de código fuente**.
- A lo largo del tiempo, la creciente complejidad de las UI para el manejo de unas aplicaciones a su vez más y más complejas, ha obligado a la utilización de herramientas basadas en modelos, siguiendo las líneas definidas dentro del campo de las arquitecturas guiadas por modelos, técnicas o métodos denominadas **MDA (Model Driven Architecture)**.
- En la actualidad disponemos de un extenso abanico de herramientas para realizar prototipos de UI (incluidas las GUI), que se encuadran dentro de las denominadas **RAD (Rapid Application Development)**.
- **Software Libre** es aquel en que su dueño renuncia a la posibilidad de obtener pago alguno por las licencias, patentes o cualquier forma que adopte su derecho de propiedad sobre él.
- Software Propietario es cualquier software en el que el usuario tiene limitaciones para usarlo, modificarlo o redistribuirlo (también llamado código cerrado), concepto que se aplica a cualquier software no libre o semilibre.



PROEDUCA