

DESARROLLO DE INTERFACES

UD1. DISEÑO INTERFACES. CONFECCIÓN DE INTERFACES DE USUARIO. XML

2CFGS

DAM

DISEÑO INTERFACES. CONFECCIÓN DE INTERFACES DE USUARIO. XML



Pau Gallego Garcia

CFGS DISEÑO APLICACIONES MULTIPLATAFORMA

Módulo: 0488 – Desarrollo de Interfaces

UD1: Diseño interfaces. Confección de interfaces de usuario. XML

INTRODUCCIÓN

Uno de los campos de la informática que se encuentra en constante evolución y experimentación es el del diseño y desarrollo de interfaces de usuario. Y esto no sólo se debe a la evolución de herramientas de desarrollo o aparición de nuevos estándares, sino también al desarrollo de interfaces hardware que dan la vuelta a la forma de interacción habitual con las máquinas.

En esta primera unidad didáctica haremos una introducción a las interfaces de usuario y su historia y los elementos que habitualmente se utilizan dentro de una interfaz para conseguir una buena interacción hombre-máquina. También haremos un repaso a algunas de las herramientas que podemos utilizar para desarrollar los diferentes tipos de interfaz, tanto de software libre como privativo.

Dentro de este proceso de desarrollo juega un papel importante el lenguaje XML, que nos permitirá exportar, modificar e importar nuestros diseños entre diferentes herramientas de diseño. Veremos con ejemplos cómo utilizar este lenguaje.

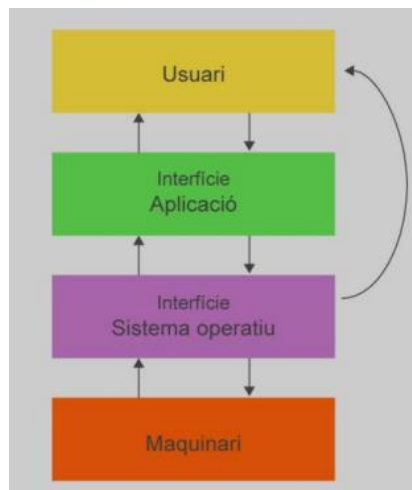
1. CONFECCIÓN DE INTERFACES DE USUARIO

Actualmente, tiene alrededor multitud de ejemplos de aplicaciones informáticas, con las interfaces de usuario correspondientes, que ayudan a la vida cotidiana de los seres humanos. Naturalmente, ante un ordenador, se utilizarán muchas aplicaciones, pero ante un cajero automático en el que se quiere sacar dinero o realizar cualquier otra operación, habrá que interactuar con otro tipo de interfaz de usuario (que será interactiva, en este caso).

Para cada situación podemos encontrar distintos ejemplos de interfaces, adaptadas en todo momento para el usuario final y la usabilidad de cada producto. Las interfaces de usuario están en continua evolución y deben conocerse una serie de herramientas que permitan su desarrollo.

1.1 INTRODUCCIÓN A LAS INTERFACES DE USUARIO

Empezaremos recordando el funcionamiento de un sistema informático. ¿Cómo consigue un usuario manejar como a él le interesa un ordenador? El conjunto de piezas que componen el hardware de un ordenador deberá llegar a funcionar según los deseos del usuario. Pero la relación entre uno y otro tendrá algunos actores intermedios, como puede verse en la figura:



En la imagen podemos observar los diferentes agentes que intervienen en un sistema informático:

- **Sistema Operativo.** Encargado de la gestión y coordinación de las tareas que llevan a cabo un intercambio de información entre los diferentes recursos. Se compone por un sistema de entrada/salida, de la gestión de procesos, de la gestión de la memoria principal o del sistema de archivos. Actúa de interfaz entre el hardware y las aplicaciones utilizadas.
- **Aplicaciones.** Se encuentran por encima del sistema operativo y por debajo de los usuarios. Las aplicaciones tendrán unas funcionalidades concretas, y ayudan a los usuarios a conseguir sus objetivos determinados.

¿Y las interfaces?

*Las **interfaces** forman parte de las aplicaciones. Son la parte de las aplicaciones con las que se relacionaran los usuarios.*

A lo largo de los siguientes apartados observaremos aquellas interfaces pertenecientes a utilidades creadas para facilitar las tareas o automatizar los procedimientos tanto a escala empresarial como a escala masiva de usuarios.

1.1.1 ¿QUE SON LAS INTERFACES DE USUARIO?

*Una aplicación informática tendrá varias interfaces de usuario. Una **interfaz de usuario** es un conjunto de elementos (que pueden pertenecer al software o al hardware) que ofrecen una información al usuario, y permiten, además, la interacción (física o lógica) entre el usuario y el ordenador, por medio de un dispositivo periférico o un enlace de comunicaciones.*

En nuestro caso nos fijaremos en las interfaces de usuario gráficas o GUI (Graphical User Interface). Estas interfaces permiten al usuario interactuar con el sistema informático de más formas que las alternativas tradicionales: teclear instrucciones complejas y difíciles de entender para el usuario no experto.

*Las **interfaces gráficas de usuario (GUI)** son aquellas que utilizan elementos gráficos, como pueden ser menús, ventanas o diálogos, además del uso de otros recursos del sistema informático (periféricos como el teclado, el ratón o el sonido) para permitir al usuario interactuar con el ordenador de forma muy sencilla e intuitiva.*

Actualmente podemos encontrar un uso generalizado de las interfaces gráficas de usuario. La gran mayoría de los sistemas operativos tienen este tipo de interfaz, de modo que integrar una aplicación en el sistema operativo sea muy sencillo, y sólo sea necesario adaptar algunos parámetros. Incluso existen dispositivos que han desarrollado su sistema operativo propio, como los dispositivos móviles, con sus propias interfaces.

Algunos ejemplos de interfaces gráficas de usuario los podemos encontrar en los siguientes sistemas:

- **Windows:** la interfaz de usuario de Windows ha ido evolucionando junto con la evolución del sistema operativo. Todas tienen algunos denominadores comunes, como ofrecer un menú de inicio, una barra de tareas, un escritorio y algunas características como la ejecución de varias ventanas de forma simultánea o la configuración de muchas de sus funcionalidades.
- **MAC:** los sistemas informáticos de los Mac llevan un sistema operativo Mac OS X. El nombre de la interfaz gráfica de usuario que incorpora su sistema operativo se conoce como Aqua (desde el año 2000). Aporta un Dock, una barra de accesos directos a las aplicaciones y carpetas de documentos, que facilita la navegación por el sistema. Incorpora también un menú siempre en la misma ubicación, que irá cambiando en función de la aplicación que esté activa.
- **Linux:** el sistema UNIX/Linux lleva una interfaz primaria o básica de tipo texto, y era el único sistema para poder interactuar con ellos hasta no hace mucho tiempo. Actualmente existen varias interfaces gráficas:
 - **X Window:** incorpora un modelo cliente-servidor que permite el uso directo por parte de las aplicaciones de los periféricos de entrada/salida.
 - **KDE (SUSE Linux):** entorno de escritorio para varios sistemas operativos. Permite la ejecución automática de nuevos dispositivos de datos, incorpora un inicio rápido.
 - **GNOME:** entorno de escritorio en el proyecto GNU. Nació como alternativa a KDE. Mejora su usabilidad y aporta elementos útiles para discapacitados.
 - **Open Look:** permite ver y ejecutar varias aplicaciones de forma simultánea en diferentes ventanas.

Cada interfaz tendrá su razón de ser y sus funcionalidades bien definidas, pero podemos establecer algunas **funciones principales** que pueden desempeñar las **interfaces gráficas de usuario**. Podemos encontrar:

- + Configuración de las interfaces gráficas de usuario y del entorno de trabajo.
- + Control de acceso a una aplicación informática.
- + Sistemas de ayuda interactivos.
- + Gestión y manipulación de directorios y archivos de un sistema operativo.
- + Arranque y cierre de un sistema informático mediante un sistema.
- + Intercambio de datos entre distintas aplicaciones. operativo.
- + Comunicación entre sistemas informáticos.
- + Ayuda al desarrollo de aplicaciones informáticas.
- + Ayuda al diseño y desarrollo de interfaces gráficas de usuario.
- + Gestión y manipulación de las funcionalidades que se puedan configurar en los sistemas informáticos.

Estas funciones ayudan para el **objetivo principal de una interfaz de usuario, que consiste en comunicarse de manera sencilla y agradable con el usuario por medio de un dispositivo de entrada/ salida.**

Pero no todas las interfaces han sido, ni son, sencillas y agradables de utilizar. Las interfaces de líneas de instrucciones **(CLI, command line interface), son un tipo de interfaz más arcaico, pero con algunas funcionalidades mucho más prácticas para sus usuarios, normalmente usuarios más expertos.**

Uno de los puntos críticos será el diseño de las interfaces de usuario. Será necesario que cumpla todos los requisitos y que sea sencilla de utilizar. Si en la fase de diseño no se consiguen estas premisas, difícilmente se obtendrá una interfaz adecuada para ser utilizada de forma agradable y sencilla por sus usuarios.

1.1.2 EVOLUCIÓN DE LAS INTERFACES DE USUARIO

A lo largo de los años, las interfaces gráficas han estado ligadas de las manos con los sistemas operativos. A medida que éstos iban evolucionando, las interfaces de las aplicaciones que se ejecutaban sobre aquellos sistemas operativos habían evolucionado en la misma medida, o más, si esto era posible.

Los sistemas operativos gráficos acercaron la informática a muchos más usuarios de los que hasta ese momento habían utilizado los ordenadores. Ya no era necesario ser conocedor de las órdenes a introducir por la línea de instrucciones o tener conocimientos de BASIC.

A medida que el hardware evolucionaba en prestaciones y descendía en coste había una lucha por conseguir el negocio de la informática de gran consumo a escala de software, es decir, en cuanto a sistemas operativos. Todo esto ha llevado a una lucha entre diferentes empresas por desarrollar los entornos más agradables y sencillos de utilizar, lo que ha provocado una continua evolución de las interfaces a lo largo de los últimos años.

A lo largo del tiempo se puede dividir la evolución de las interfaces de usuario en las siguientes fases:

- **Interfaces de usuario basadas en la línea de instrucciones (hasta los años setenta).**
- **Nacimiento de las interfaces de usuario, hacia 1970.**
- **Evolución hacia las interfaces gráficas de usuario (1980-1995).**
- **Interfaces gráficas de usuario no basadas en instrucciones (1996-2001).**
- **Interfaces gráficas de usuario interactivas (2002-ahora).**

INTERFACES DE USUARIO BASADAS EN LA LÍNEA DE INSTRUCCIONES (hasta 1970)

En los inicios de los sistemas informáticos no existía ninguna interfaz de usuario. Si se quería modificar la programación **se accedía directamente al hardware** para realizar las adecuaciones pertinentes. Posteriormente, con la programación por lotes tampoco existía una interfaz que permitiera la interacción. El usuario debía indicar un conjunto de instrucciones en el sistema que, una vez procesadas, ofrecería unos resultados.

A partir de los años sesenta, con los lenguajes de órdenes, se llegó a la interfaz de línea de instrucciones. Esta interfaz permitía al usuario interactuar con el ordenador con una línea de texto, que serviría como texto que llevaría incluido una orden.

NACIMIENTO DE LAS INTERFACES DE USUARIO (1970)

En la década de los setenta comienza a evolucionarse hacia el uso de un ordenador por parte de un único usuario. Estos primeros ordenadores personales permitirán a los usuarios almacenar información, editarla y compartirla de forma sencilla. Las aplicaciones que se desarrollan siguen estos objetivos e intentan ofrecer unas interfaces de usuario adecuadas para conseguirlo. Se suele utilizar un diseño de interfaces de dos dimensiones con menús jerárquicos de pantalla completa. Se utilizan las teclas de función como una forma de interactuar con las aplicaciones y de completar los menús.

EVOLUCIÓN HACIA LAS INTERFACES GRÁFICAS DE USUARIO GUI (1980-1995)

En la década de los ochenta los ordenadores personales se hicieron más habituales para el gran público, aunque las interfaces gráficas de usuario tardarían un poco en llegar. En esta fase se empiezan a utilizar las interfaces llamadas WIMP (Windows, Icons, Menus and Pointer), es decir, interfaces que disponen del movimiento de un puntero por la pantalla y que tendrán ventanas, iconos y menús. Se puede considerar que hasta el momento se trabajaba en dos dimensiones con las interfaces con líneas de instrucciones y, a partir de la aparición de las interfaces WIMP, se puede considerar que se utiliza una tercera dimensión que permite trabajar con más de una ventana (y aplicación) a la vez, superponiendo una encima de la otra.

Las GUI permiten a los usuarios trabajar directamente en aquellas funcionalidades que les interesan, y deben desplazarse hasta otras interfaces para localizar y poder utilizar otras funcionalidades. Ofrecen una interacción con los usuarios que se basa en la manipulación directa, ofrece los elementos importantes para el usuario en cada interfaz y permite su manipulación.

INTERFACES GRÁFICAS DE USUARIO NO BASADAS EN INSTRUCCIONES (1996-2001)

A partir del año 1995 existe un cambio muy importante en el uso de las interfaces de usuario. La aparición de sistemas operativos que interactúan con los usuarios de forma única y completa con GUI convirtió las interfaces en un elemento más de consumo y generalizó más aún el uso de los ordenadores personales dentro de las casas y sistemas informáticos en las organizaciones.

Una de las novedades más importantes es la aparición de elementos multimedia que se pueden añadir a las interfaces, como sonidos. Estos elementos pueden considerarse una “dimensión” más para añadir a las dos dimensiones que tenían las interfaces de estilo WIMP, aunque no se puede hablar propiamente de una tercera “dimensión”.

INTERFACES GRÁFICAS DE USUARIO INTERACTIVAS (2002-ACTUALIDAD)

Las interfaces gráficas que nos encontramos actualmente van encaminadas a utilizar aún más dimensiones y otras evoluciones.

La primera nueva “dimensión” que se incorporó fue la de utilizar elementos multimedia. En algunos casos por necesidad y en otros para investigar hacia una evolución que haga aún más

sencillas y útiles las interfaces. Por esta razón se podría considerar una tercera "dimensión" el concepto de tiempo o incluso una cuarta "dimensión" con el concepto de comunicación verbal.

Pero tampoco se dejará de lado en el futuro el aprovechamiento de las nuevas posibilidades que puede ofrecer añadir una tercera dimensión real a las interfaces, como pueden ser las pantallas y sus funcionalidades (tecnología LED, 3D....)

1.1.3 TIPOS DE INTERFACES DE USUARIO

Las interfaces gráficas pueden clasificarse en diferentes tipos en función de las características y funcionalidades que tengamos en cuenta.

Si dividimos las interfaces en función de la **forma de interaccionar con los usuarios**, podremos encontrar **tres** tipos de interfaces:

- Las **interfaces de línea de instrucciones (CLI)**. Las alfanuméricas, con las que habrá que trabajar con instrucciones.
- Las **interfaces gráficas de usuario (GUI)**. Interacción más rápida, sencilla e intuitiva gracias al hecho de representar de forma gráfica los elementos de las interfaces.
- Las **interfaces desarrolladas para pantallas táctiles**. Estas interfaces deben adaptarse a las necesidades de un dispositivo de entrada y de salida a la vez, que será una pantalla táctil, ya sus características específicas.
- Las **interfaces desarrolladas para dispositivos móviles**. Dispositivos como teléfonos móviles, **PDA** (personal digital assistant) u otras herramientas similares disponen de sistemas operativos y tipos de interfaces adecuadas a sus necesidades.

Si nos fijamos en **cómo se ha desarrollado la interfaz**, los elementos con los que se ha construido, podemos encontrar tres tipos:

- De **hardware**: interfaces que permiten la interacción entre el usuario y el sistema informático mediante elementos de hardware, dispositivos como **pulsadores** o **lectores de barras** u otros reguladores o instrumentos.
- De **software**: interfaces desarrolladas para ordenadores que basan su creación en desarrollo de software, mediante el cual se llevará a cabo toda la **interacción con los usuarios**.
- De **software-hardware**: son interfaces que compartirán los dos tipos explicados anteriormente.

Otra forma de clasificar las interfaces es **según la arquitectura de las aplicaciones y su forma de trabajar**:

- ♦ **Aplicaciones locales. Interfaz WINFORMS**. Se desarrollan para trabajar en **una única máquina un único usuario** al mismo tiempo. Se podrá desarrollar en diferentes lenguajes de programación y se podrá haber vinculado a diferentes tipos de bases de datos, pero se encontrarán instaladas en una máquina que será el único lugar desde donde se podrá utilizar.
- ♦ **Aplicaciones cliente-servidor**. Este tipo de aplicaciones están basadas en el trabajo en **más de una máquina**, en la que una hará de servidor y el resto harán de clientes que

acceden remotamente al servidor para acceder a los datos o funcionalidades. Es necesario realizar una instalación en la máquina servidor y otra instalación en cada una de las máquinas cliente.

- ♦ **Aplicaciones Web.** Son un tipo específico de las aplicaciones cliente/servidor, permiten acceder a datos y funcionalidad a través de las redes telemáticas, y el acceso más habitual es el que se realiza por medio de un navegador de Internet, que accede a un servidor web, donde se localizarán los datos y la implementación de las funcionalidades.

Una última forma de clasificar las interfaces es a partir del **software donde se desarrollarán:**

- **Plataformas RichClient:** Son herramientas de desarrollo de software que **contendrán todos los elementos necesarios para diseñar y desarrollar** sin otra necesidad una interfaz gráfica de usuario. Además, se podrán aprovechar de las bibliotecas y elementos existentes.
- **Plataformas ThinClient:** Son herramientas de desarrollo de software que **necesitan otras herramientas para poder diseñar y desarrollar** interfaces gráficas de usuario y partes de software.

1.1.4 ELEMENTOS DE LAS GUI. PROPIEDADES Y CARACTERÍSTICAS.

Podemos observar cómo el denominador común ha sido llegar a ofrecer a los usuarios unas interfaces con una usabilidad óptima. Para ello se ha establecido que las interfaces deben cumplir una serie de **características** que, entre otras, serán:

- **Accesible e intuitiva.** Mostrar con claridad las funcionalidades que ofrece y facilitar llegar de manera sencilla y clara.
- **Uso de metáforas.** Las interfaces deben utilizar metáforas que vinculen sencillamente icono o **imagen con funcionalidad** u objetivo.
- **Aprendizaje y uso fácil.** Fáciles de usar y de aprender por parte de los nuevos usuarios.
- **Consistencia.** Seguir un mismo diseño y estructura entre ellas y con otras interfaces análogas. Consistentes en diferentes entornos.
- **Ofrecer el control de las interfaces.**
- **Anticipación.** Prever los posibles errores que pueda cometer un usuario y ofrecer soluciones.
- **Legibilidad.** Fácilmente interpretables.
- **Autonomía.** Un usuario no debe necesitar más ayuda de la que la interfaz ofrece.
- **Reducir carga de memoria.**
- **Internacionalización de la interfaz.** Debe ser entendida y utilizable por usuarios de diferentes culturas e idiomas.
- **Valores iniciales.** Conocidos como valores por defecto, valores que aparecerán inicialmente seleccionados.
- **Ley de Fitts.** Conseguir optimizar la ley de Fitts, hablando de ergonomía.

Las interfaces gráficas de usuario (GUI a partir de ahora) disponen de una serie de elementos propios, comunes a muchas de las GUI desarrolladas, que dispondrán de unas características y propiedades. Los elementos que se pueden encontrar de forma habitual son: ventanas,

cuadros de diálogo, asistentes, menús, pestañas, barras de herramientas, iconos, entorno de trabajo, entorno de escritorio, controles y tipografía.

VENTANAS

Uno de los elementos más importantes. Las ventanas son, normalmente, **bidimensionales y de forma rectangular, y se ubicarán en el escritorio del sistema operativo**. Las ventanas nos ofrecen la posibilidad de trabajar a la vez con distintos entornos, que podrán ser de la misma aplicación o de diferentes aplicaciones, nos permitirán ofrecer información al usuario de forma organizada y navegar por las funcionalidades de una aplicación visualizando la información de forma jerarquizada. Esto permite gestionar y manipular la información de forma bastante sencilla. Cada ventana puede ser manipulada como al usuario le interese, maximizándola o minimizándola, moviéndolas o cerrándolas, ...

A lo largo de los años las ventanas han ido evolucionando, llegando hoy a lo que se podría llamar un estándar de diseño de ventanas, con unos elementos básicos, de los que harán uso todas las ventanas, como son: el marco, la cabecera de la ventana, el espacio del contenido, la barra de desplazamiento y el pie de la ventana.

CUADROS DE DIÁLOGO

Los cuadros de diálogo son un tipo de ventanas, también consideradas como ventanas secundarias. Los cuadros de diálogo aparecen por encima del resto de ventanas pidiendo una interacción concreta al usuario, que deberá contestarle. Esta interacción podrá ser sólo informativa o un requisito concreto de algún dato o de alguna acción que habrá que contestar de manera concreta para poder continuar utilizando la aplicación.

ASISTENTES

Los asistentes son **ventanas que van apareciendo una detrás de otra hasta que se alcanza un determinado objetivo**. Como su nombre indica, asisten al usuario hasta completar una funcionalidad, preguntando paso a paso todas las informaciones necesarias hasta completarla correctamente.

Los asistentes son habituales en la instalación de software o como ayuda en la utilización de una funcionalidad muy concreta.

MENÚS

Los menús serán los **encargados de mostrar todas las posibilidades de interactuar con la aplicación informática**, desplegarán todo el conjunto de funcionalidades que tendrá el usuario a su alcance. Cada opción de un menú dispondrá de un subconjunto de opciones, lo que convierte a los menús en un árbol jerarquizado con un número de niveles que habrán decidido los desarrolladores.

La forma más habitual de seleccionar una opción de menú será utilizando el ratón, sin embargo, las aplicaciones deben poder ofrecer menús accesibles para usuarios que no dispongan de ratón, con la posibilidad de moverse por los menús con un teclado utilizando la combinación de la tecla Alt con la letra subrayada para llegar a una opción determinada.

Existen muchos **tipos** de menús, dependiendo de su entorno de ejecución o de la tipología de aplicación a la que pertenece la interfaz donde se encontrará el menú:

- **Menús jerárquicos:** menús representados en forma de árbol, con un número de niveles horizontal (opciones del menú) y un número determinado de niveles verticales (hasta dónde podrá llegar el usuario navegando por el menú).
- **Menús contextuales:** no se encontrarán visibles en la interfaz hasta que el usuario no provoque su activación. Son menús que se abrirán en una nueva ventana flotante, que variarán en función de la ubicación del ratón en el momento de llamarlos.
- **Menús de navegación (scrolls):** se puede considerar una evolución de los menús jerárquicos. Se trata de un tipo de menú que muestra sólo las opciones más utilizadas de los menús, ocultando el resto, pero dejando visible un pequeño icono que, al ser seleccionado, muestra todo el resto de las opciones del menú.

PESTAÑAS

Las pestañas de propiedades son un elemento que permite mostrar un conjunto de datos o funcionalidades relacionadas entre sí de forma agrupada. En una misma ventana se podrán mostrar los contenidos de tantas ventanas como se desee, separándolas por pestañas, que tendrán que tener un título y contener informaciones relacionadas.

BARRAS DE HERRAMIENTAS

Las barras de herramientas facilitan el acceso a algunas funcionalidades mediante un icono. Todas las funcionalidades ofrecidas en las barras de herramientas se podrán encontrar desarrolladas en los menús, pero no todas las opciones que ofrecen los menús se podrán encontrar en las barras de herramientas.

Una vez seleccionadas aquellas funcionalidades que se utilizarán más habitualmente se identificará cada una con un símbolo que represente aquella funcionalidad.

ICONOS

Los iconos son imágenes que representan funcionalidades o acciones que se podrán realizar haciendo un clic encima. Entre otros sitios, en las interfaces gráficas de usuario, son elementos que se utilizan en las barras de herramientas.

Es necesario que los iconos sean muy fácilmente identificables por parte de los usuarios, o, en su defecto, que estén muy bien informados de sus funcionalidades, puesto que hará más rápido, atractivo y sencillo el uso de las aplicaciones.

ENTORNO DE TRABAJO

Se trata de la ubicación principal de las informaciones con las que se interactúa con el usuario, donde se mostrará el texto, imágenes o datos que el usuario habrá solicitado mediante los menús.

TIPOGRAFÍA

Se conocen como tipografía digital los tipos de letra desarrollados exclusivamente para pantallas, para usarlos en interfaces gráficas de aplicaciones informáticas. La cuidadosa elección de estas tipografías también ofrecerá un mensaje específico al usuario y es un elemento más a tener en cuenta en el diseño y desarrollo de las interfaces.

CONTROLES

Éstos proporcionan funciones a la interfaz de usuario que permitirán muchas más posibilidades en las interacciones entre interfaz y usuario. Un tipo de control son los **botones**, objetos de control que dan la opción de introducir un dato de confirmación en el sistema. Existen diferentes tipos de botones:

- **Botones en forma de radio (*radio buttons*)**: son elementos que se utilizan en formularios o en menús para dar a seleccionar al usuario **una opción** entre una lista. Estos botones en forma de radio son botones redondos que podrán ser escogidos por medio de acciones de usuario.
- **Botones de confirmación (*check box*)**: botones de forma cuadrada también para poder seleccionar opciones de una lista de opciones. La diferencia con los botones en forma de radio será que en los botones de confirmación **se podrá seleccionar más de una opción entre las mostradas**, y en cambio en los botones en forma de radio sólo se puede seleccionar una.
- **Botones de relieve**: este tipo de botones imita un botón de un dispositivo físico que **simula un volumen**, dando así la posibilidad de tener **varios estados (activado o no, seleccionado o no)**, además de poder incluir texto con la definición de la funcionalidad que va a representar.

1.1.5 HERRAMIENTAS DE PROPIEDAD Y HERRAMIENTAS LIBRES DE EDICIÓN DE INTERFACES.

El desarrollo de una interfaz gráfica de usuario desprenderá mucho del lenguaje de programación que se utilice y de su entorno de desarrollo.

Las herramientas de creación y edición de interfaces son un tipo de software que ayudará al programador a desarrollar interfaces que sigan las indicaciones establecidas en la fase de diseño a partir del análisis de las necesidades. Deberán tener en cuenta todas las fases de desarrollo de un proyecto informático.

Para conseguir desarrollar de forma sencilla las interfaces, las herramientas incorporan varios elementos preprogramados que facilitan mucho la creación de interfaces gráficas complejas de una forma sencilla.

A continuación, se enumeran algunas de las muchas **herramientas de desarrollo** de interfaces existentes:

- **Microsoft Visual Studio**. Se trata de un entorno de desarrollo integrado desarrollado por Microsoft para sistemas operativos Windows. Es de pago. Soporta los siguientes lenguajes de programación: Visual C++, Visual C#, Visual J#, ASP.NET y Visual Basic .NET.
- **Eclipse**. Desarrollado inicialmente por IBM. Actualmente desarrollado por una fundación sin ánimo de lucro, lo que hace la herramienta gratuita y de código abierto. Soporta, entre otros, los lenguajes Java, C++, Perl y PHP. Es multiplataforma.
- **IDE NetBeans**. Desarrollado inicialmente por Sun, que actualmente patrocina los proyectos de desarrollo. Herramienta de código abierto y gratuita. Soporta, entre otros, los lenguajes Java, JSP, C++ y PHP. Es multiplataforma.

- **MonoDevelop.** Entorno de desarrollo gratuito y de código abierto. Actualmente patrocinado por Novell. Diseñado para C#, también soporta Java o Boo. Es multiplataforma.
- **SharpDevelop.** Entorno de desarrollo de código abierto. Para los lenguajes de programación C#, Visual Basic .NET y Boo. Para sistemas operativos Windows y desarrolladores que no quieren o no pueden utilizar Microsoft Visual Studio.

1.2 HERRAMIENTAS DE DISEÑO DE INTERFACES

Las interfaces gráficas están llenas de elementos que ofrecen muchas posibilidades para interactuar con los usuarios. Pero hay que echarle un vistazo desde el punto de vista de los desarrolladores de interfaces gráficas. ¿Cómo podrá un desarrollador elegir cuáles son los iconos que querrá utilizar para identificar diferentes acciones? ¿Cómo seleccionará la forma más adecuada de interactuar en cada momento? ¿Cuáles son las herramientas que podrán servir para conseguir esos objetivos?

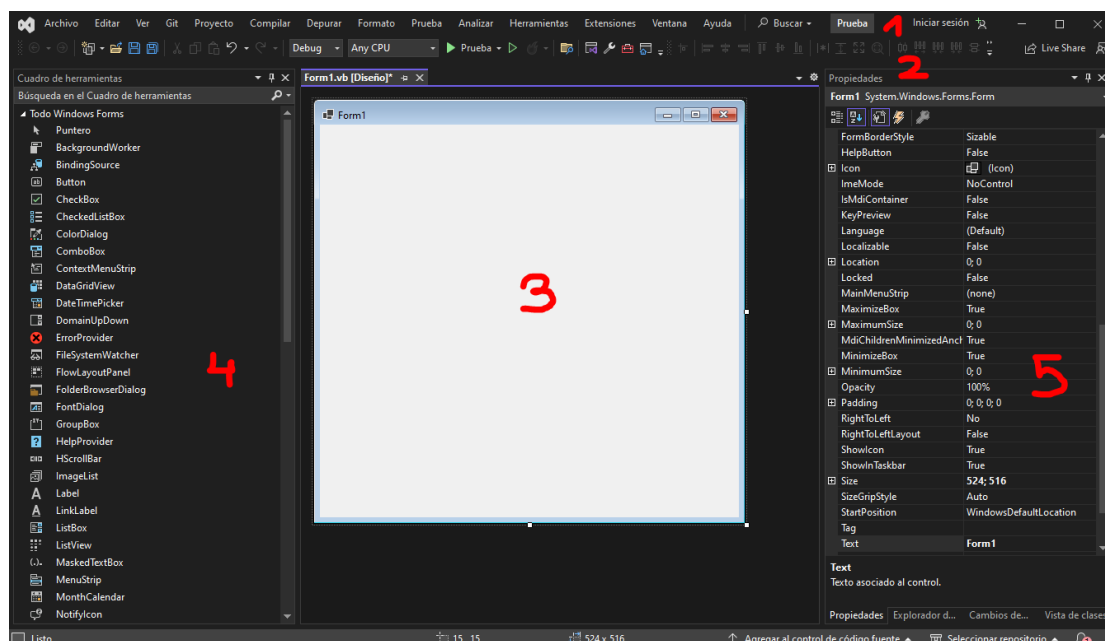
Todavía existe la posibilidad de programar como se hacía hace años, utilizando la programación por código, de forma parcial o completa. Es muy difícil encontrar, hoy en día, gente que aún desarrolle interfaces gráficas de usuario a partir de un lenguaje de programación. Sin embargo, si es más habitual utilizar herramientas de diseño de interfaces y, a partir de ubicar manualmente los elementos en las ventanas, manipular posteriormente el código que configurará y gestionará las acciones de estos elementos. La programación visual consiste en diseñar las interfaces colocando los elementos en los formularios y configurando sus propiedades y acciones sin tener que implementar una línea de código.

Muchas de estas herramientas de diseño y desarrollo de interfaces de usuario tienen algunos elementos comunes (o muy similares), como puede ser disponer de un área de diseño, una paleta de componentes, editores de propiedades o diferentes contenedores. Para explicar algunos de estos elementos que son similares en sus características y en su uso se utilizará como herramienta para los ejemplos la versión 2022 del software Visual Studio, de Microsoft.

1.2.1 ELEMENTOS DE LAS HERRAMIENTAS DE DESARROLLO DE INTERFACES

Al iniciar el trabajo con una herramienta de diseño y desarrollo de interfaces gráficas es necesario identificar y entender muy bien el entorno de trabajo, es decir, bajo qué sistema operativo y condiciones se ejecutará.

En la siguiente imagen puede verse el entorno de trabajo de **Visual Studio 2022** al iniciarlo con un proyecto en **Visual Basic** para una aplicación desarrollada con **Windows Forms**.



Se pueden identificar los siguientes elementos en el entorno de trabajo de una aplicación de diseño o desarrollo de interfaces de usuario:

1. **Menús.** Es donde se localizarán todas las funcionalidades que ofrece el software. Se pueden definir acciones para un conjunto de teclas y configurar qué funcionalidad estarán más accesibles.
2. **Menú de barra de herramientas.** Conjunto de iconos, en la parte superior de la interfaz, normalmente justo debajo del menú. Cada icono representa una acción a desarrollar, a la que también se podría llegar por medio del menú en formato de texto.
3. **Área de diseño.** Ocupa la parte central. Contendrá los paneles en los que se irán ubicando los elementos que se quieren añadir a la interfaz. Dispondrá de pestañas para poder trabajar con más de un panel o interfaz al mismo tiempo.
4. **Cuadro de herramientas.** Contiene todos los elementos que podremos añadir a los paneles que se estén desarrollando en el área de diseño. Seleccionando y arrastrando cada elemento en el área de diseño se podrá diseñar la interfaz que queremos.
5. **Hoja de propiedades.** Es una ventana en la parte derecha del área de diseño que muestra las propiedades o características del elemento seleccionado. Estos valores de las propiedades se podrán modificar según interese para conseguir el cumplimiento de los objetivos del diseño de las interfaces.

De hecho, todos estos elementos que componen una aplicación de desarrollo de interfaces son los mismos que se pueden utilizar para crear interfaces para las aplicaciones a desarrollar.

1.2.2 COMPONENTES: CARACTERÍSTICAS Y CAMPOS DE APLICACIÓN

Cuando se habla de interfaces de usuario normalmente se utiliza un vocabulario que incluye palabras como contenedores, controles o componentes.

¿Qué es un componente? Se trata de un objeto que podrá reutilizarse, que permite interactuar con otros objetos y proporciona el control sobre determinados recursos externos en tiempo de diseño. Los componentes se podrán diseñar y configurar, lo que hace que se pueda utilizar en una herramienta tipo IDE. Un componente podrá añadirse al cuadro de

herramientas, se podrá seleccionar y arrastrar hacia un panel o formulario y se podrán modificar sus características.






Algunos ejemplos de componentes simples: botones, listas desplegables, barras de progreso, etiquetas, botones de radio... Algunos ejemplos de componentes complejos: menús, tablas, árboles, cuadros de diálogo...

¿Qué es un control? Algunos autores consideran un control como un tipo de componente. Otros indican que son objetos similares a los componentes, al poder diseñarse y configurarse. Lo que sí está claro es que un control es un objeto que proporciona una interfaz al usuario, y permite la interacción entre usuario y aplicación para intercambiar información. Algunos ejemplos de controles pueden ser los botones, los cuadros de listas, los cuadros de edición...





¿Qué es un panel? Es un control que servirá para contener otros controles. Es una herramienta adecuada para ocultar o mostrar un determinado tipo de controles. Este tipo de control podrá tener barras de desplazamiento del control, además de poder personalizar sus características y propiedades de presentación. Un tipo de contenedor será una barra de herramientas.

¿Qué es un formulario? Es un tipo de componente que será como una ventana que el desarrollador, en tiempo de diseño, podrá utilizar para ubicar otros componentes o controles. Un formulario puede existir por sí solo o puede formar parte de un conjunto de formularios. Como ejemplos de formulario se tienen las ventanas con o sin marco, los cuadros de diálogo...

A continuación, podremos observar los principales elementos (componentes y controles) que podremos encontrar iguales o muy similares en diversas herramientas de diseño de interfaces o entornos integrados de desarrollo de software.

COMPONENTE	NOMBRE	DESCRIPCIÓN
 Label	Label	Ofrece la posibilidad de ubicar en el formulario una etiqueta con un texto determinado, que no podrá ser manipulado por el usuario de la aplicación.
 Button	Button	Son botones que ejecutarán una funcionalidad cuando se haga clic o doble clic sobre ella. Contendrán un texto indicativo de la acción que representan.
 PictureBox	PictureBox	Permite seleccionar una imagen o gráfico de un archivo previamente existente.
 ProgressBar	ProgressBar	Ofrece la progresión en la ejecución de una funcionalidad determinada que se está ejecutando.
 RadioButton	RadioButton	Son un conjunto de opciones agrupadas que ofrecen diferentes alternativas al usuario para poder escoger una. Las

opciones que se ofrecen serán mutuamente excluyentes.

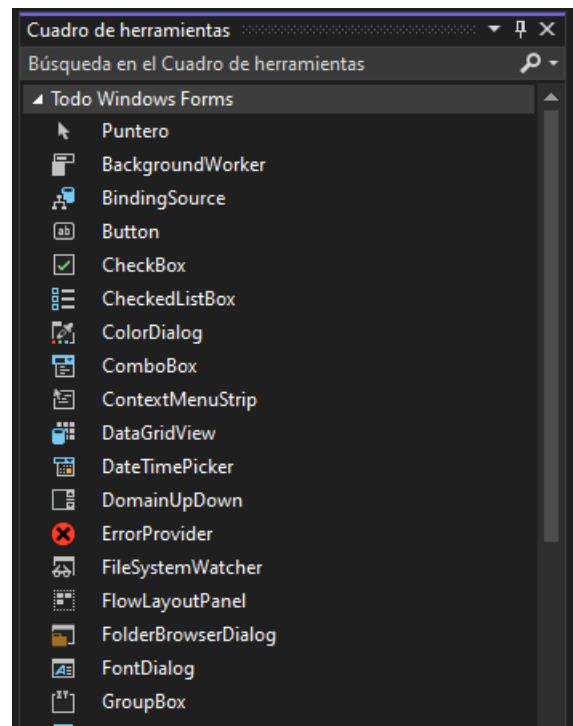
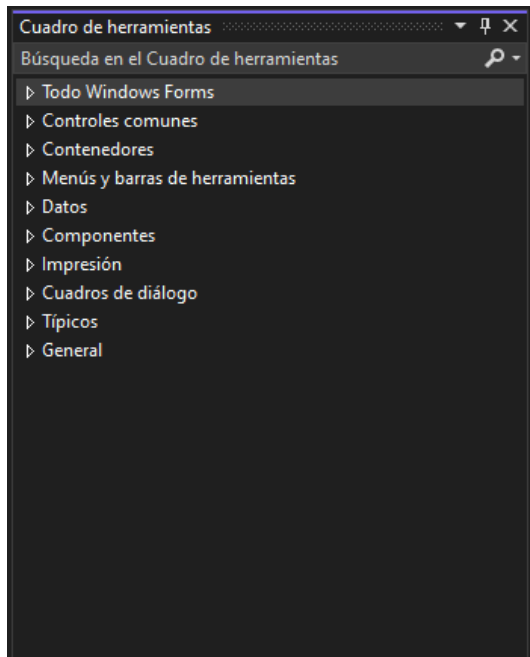
 CheckBox	CheckBox	Conjunto de opciones con un espacio en forma de cuadrado a la derecha para poder seleccionarl
 ListBox	ListBox	Este control muestra las opciones en formato de lista. El usuario podrá seleccionar una o más de las opciones.
 TextBox	TextBox	Puede mostrar información en formato texto escrita en tiempo de diseño y recoger información introducida por el usuario en tiempo de ejecución.
 ComboBox	ComboBox	Ofrece una lista de opciones, de las que se podrá seleccionar una o introducir un texto en el cuadro de edición.

1.2.3 VENTANA DE CUADRO DE COMPONENTES

La ventana del cuadro de herramientas contiene los componentes y otros elementos que se pueden utilizar en las interfaces que se desarrollan en el área de diseño. Para añadir un componente es necesario seleccionarlo con el ratón y arrastrarlo hasta el área de diseño dentro de un formulario.

Hay varios grupos de controles disponibles; el número depende del tipo de diseñador activo en el editor. Cuando se está diseñando un formulario tipo Windows Forms, aparecen las herramientas necesarias para trabajar con formularios Windows, que serán diferentes de los componentes para trabajar con formularios Web y que, a su vez, están agrupados por categorías de controles.

Esta ventana del cuadro de herramientas será posible personalizarla según las necesidades del desarrollador. Es posible crear nuevos grupos o añadir componentes adicionales a un grupo existente.



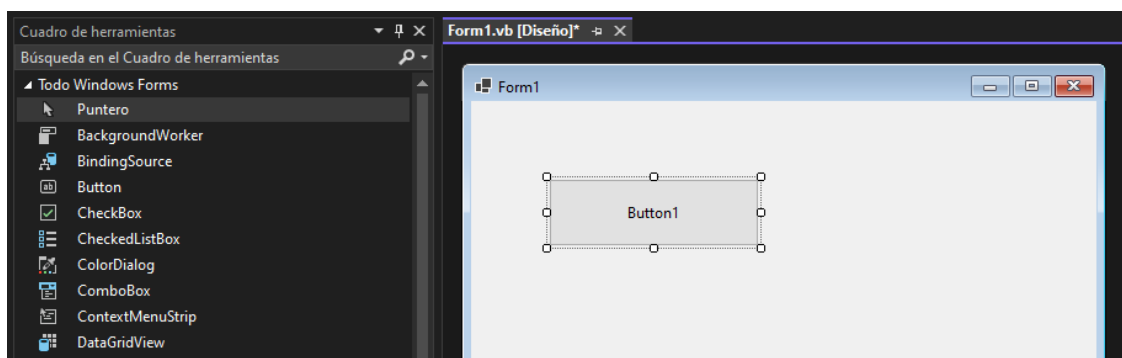
1.2.4 AÑADIR/ELIMINAR COMPONENTES A LA INTERFAZ DE USUARIO

El cuadro de herramientas permite observar cuáles son los componentes que se podrán utilizar para añadirlos al formulario que se desarrollará. En el caso de programación de Windows Forms, la plantilla de un proyecto inicial crea por defecto un formulario vacío sobre el que dibujaremos los controles.

La forma de añadir un nuevo control a un formulario es:

Tener abierto un formulario donde se desea ubicar el control en el área de diseño.

- Abrir la ventana con el cuadro de herramientas.
- Seleccionar el grupo y el control o componente que queremos.
- Hacer clic sobre el formulario o arrastrarlo hasta el formulario en el que se quiera introducir para que forme parte de esa interfaz.
- Le podremos dar las medidas que queramos.



Para eliminar un control o componente ya ubicado en un formulario basta con seleccionarlo y utilizar la tecla *Supr* para suprimirlo, de esta forma se pierde el diseño y la configuración del objeto.

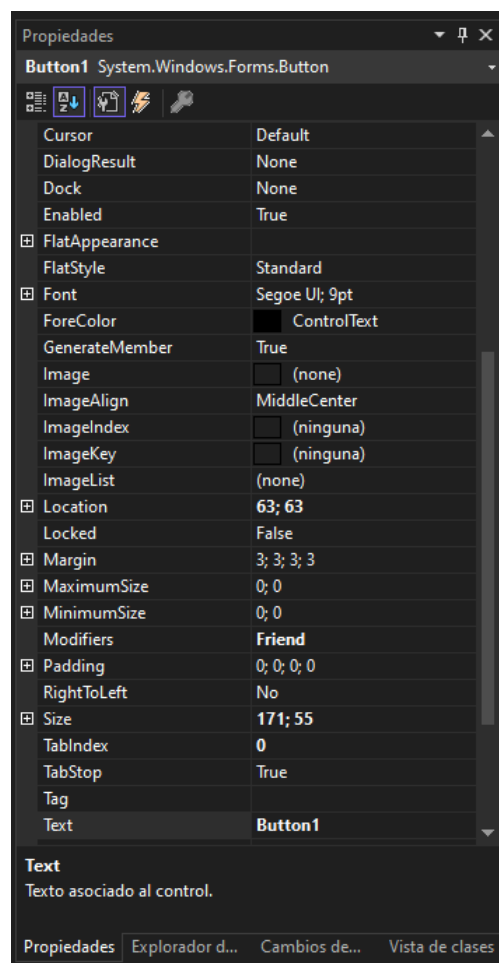
1.2.5 UBICACIÓN Y ALINEACIÓN DE COMPONENTES. MODIFICACIÓN DE PROPIEDADES

Una vez colocados los componentes en los formularios es necesario ubicarlos de forma que el usuario pueda entender las funcionalidades que ofrecen y disponerlos en una distribución agradable y sencilla. El entorno de desarrollo dispone de herramientas que permiten dar mayor precisión a la colocación de los controles mediante las características de alineación, que aparecen en la opción “Alinear” del menú “Formato” o en la barra de herramientas de “Diseño”.

La mayor parte de estas opciones de alineación sólo se pueden utilizar cuando existen varios controles seleccionados. En estos casos, la operación se realizará sobre el último control seleccionado. El primer control seleccionado será el que actúe como control primario y los demás controles se alinearán a partir del primero.

Para seleccionar varios controles se pueden ir marcando mientras se mantiene pulsada la tecla Ctrl o Mayús.

La modificación de las propiedades de los componentes, formularios o controles, por ejemplo, es una tarea necesaria e importante para poder configurar estos objetos según los objetivos del diseñador de las interfaces. Estos atributos o propiedades pueden modificarse mediante código (en tiempo de ejecución) o mediante la ventana “Propiedades” (en tiempo de diseño). La ventana Propiedades proporciona un acceso a las propiedades accesibles en tiempo de diseño.



Para modificar las propiedades de un control en tiempo de diseño se utiliza la ventana “Propiedades”. Primero habría que seleccionar el control de la lista que aparece en la parte superior de la ventana o en el propio formulario, de modo que aparecerían en la ventana todas las propiedades de lectura y escritura modificables en tiempo de diseño (hay otras propiedades que sólo se pueden modificar en tiempo de ejecución).

También es posible modificar las propiedades de varios controles, seleccionándolos en el formulario, de forma que en la ventana “Propiedades” aparezcan las propiedades comunes a todos.

1.3 CARACTERÍSTICAS ESPECÍFICAS DE LAS HERRAMIENTAS DE DISEÑO DE INTERFACES

1.3.1 CONTROLES: CLASES, PROPIEDADES Y MÉTODOS

Los controles son objetos que permiten un desarrollo de las interfaces más sencillo gracias a su posibilidad de modularidad y reutilización. Los controles tienen propiedades y métodos, y se podrán asociar a eventos.

Los controles ofrecen un abanico de posibilidades muy grande a los desarrolladores, al posibilitar dotar de unas mismas características a un conjunto de controles que se utilizarán para interfaces directamente relacionadas o poder utilizar un mismo control en varios proyectos de desarrollo de software.

Los controles disponen de tres características importantes:

- Propiedades
- Métodos (procedimientos y funciones)
- Eventos

Las **propiedades** son las características que identifican a cada control. Éstos son lo que son a partir de las partes que constituyen el objeto. Por ejemplo, una persona podría decirse que presenta propiedades como podrían ser sus ojos, las orejas, los labios, la altura, el peso... Estas propiedades pueden tener unos valores (por ejemplo, Ojos = Verdes, Orejas = Grandes, Labios = Fines, etc.) que harán que esa persona sea identificable respecto de otra por aquellas características. Del mismo modo ocurre con los controles. Las propiedades del control pueden ser modificadas en tiempo de diseño, pero también pueden ser modificadas en tiempo de ejecución, colocando su nombre y el nombre de la propiedad.



```
Button1.Text = "Button1"
```

Los controles siguen el principio de la encapsulación; esto significa que se puede tener un control 50 veces en un formulario, y si le cambiamos una propiedad a un control sólo se cambia en éste y no en los 49 controles restantes, es decir, cada control mantiene encapsuladas sus propiedades.

Los **métodos** son los procedimientos o funciones asociados a un control, que permiten realizar ciertas operaciones útiles sobre este control y obtener una respuesta. Los

procedimientos y funciones son muy similares; la diferencia es que los procedimientos no devuelven ningún valor, mientras que las funciones siempre devuelven un solo valor.

Por último, un **evento** es una acción determinada que se puede hacer vinculada con un control. Por ejemplo, si sobre un control tipo Button se hace un clic con el ratón se ejecutará una determinada acción. Se puede decir que el evento “clic sobre el control Button” activa cierta funcionalidad.

1.3.2 DIÁLOGOS MODALES Y NO MODALES

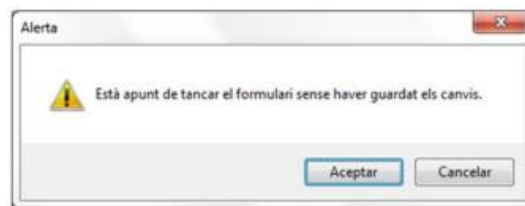
Un **cuadro de diálogo** es una ventana que aparecerá ofreciendo una información a los usuarios en un momento determinado de la ejecución de una interfaz.

Un cuadro de diálogo es *modal* cuando la aplicación misma o el formulario que le ha llamado no puede recibir ningún evento hasta que se haya cerrado el cuadro de diálogo.

Un cuadro de diálogo es *no modal* de lo contrario, cuando la aparición del cuadro de diálogo no limita al usuario a continuar interactuando con el resto de ventanas.

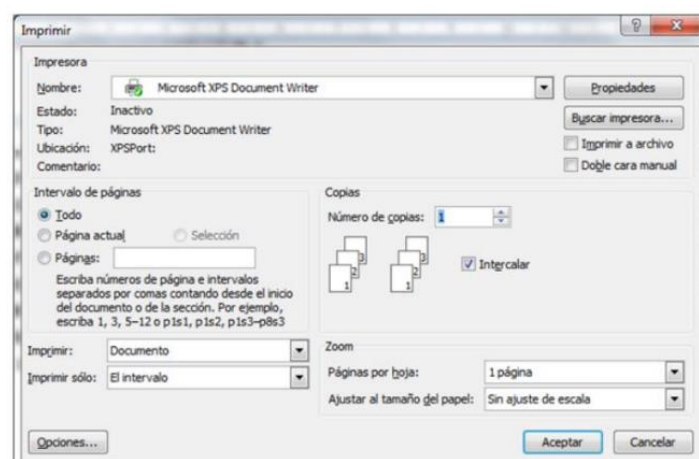
A continuación, se muestran tres ejemplos de cuadros de diálogos modales.

Un cuadro de mensaje es un cuadro de diálogo modal que puede utilizarse para mostrar información textual y permitir que los usuarios tomen decisiones con botones. Para crear un cuadro de mensaje, se utiliza la clase *MessageBox*.



Otro tipo de cuadros de diálogo son los cuadros de diálogo comunes, que pueden ser tan modales como no modales. El entorno Windows implementa varios cuadros de diálogo reutilizables que son comunes para todas las aplicaciones, incluyendo cuadros de diálogo para abrir archivos, guardar archivos e imprimir.

Dado que estos cuadros de diálogo están disponibles para todas las aplicaciones y que ayudan a proporcionar una experiencia de usuario coherente, se conocen como cuadros de diálogo comunes. Un ejemplo de cuadro de diálogo sería el creado cuando se intenta imprimir un documento.



1.3.3 ENLACE DE COMPONENTES A ORÍGENES DE DATOS. DATAGRID.

El enlace de datos es el mecanismo proporcionado por algunas plataformas que permiten, en aplicaciones con interfaz Windows o en aplicaciones web, enlazar objetos contenedores de datos con los controles de formulario, para poder realizar operaciones de navegación y edición.

Existen varios tipos de fecha binding o enlaces a datos, como los simple data binding o los complejo data binding.

SIMPLE DATA BINDING

Consiste en una asociación entre un control, que puede mostrar un único dato, y el objeto que actúa como contenedor de datos, como el enlace entre un control TextBox o una etiqueta y un objeto DataColumn de una DataTable de un DataSet. Por ejemplo, puede vincular el campo NombreColumn de la tabla NombreTabla a un control de tipo TextBox. Cuando existe una modificación en la propiedad de destino, el cambio también se realiza para el objeto origen.

COMPLEX DATA BINDING

Este tipo de enlace es posible entre un control que puede actuar como interfaz o visor de datos, ya que dispone de la capacidad de mostrar varios o todos los datos, normalmente más de un registro, del objeto contenedor de datos. Esto se utiliza normalmente en controles DataGridView (parrilla), ListBox o ErrorProvider. Un ejemplo clásico es el vínculo entre una tabla de datos y un objeto DataTable de un DataSet.

1.3.4 LOS EVENTOS: CONCEPTO, ASOCIACIÓN DE ACCIONES, EVENTOS ESCUCHADORES.

Una aplicación no puede saber en qué momento el usuario querrá realizar una determinada funcionalidad. Esta interacción será asíncrona durante la ejecución del programa. Los eventos o eventos proporcionan un mecanismo adecuado para tratar este tipo de situaciones que, normalmente, son producidas desde el exterior de la aplicación, por ejemplo, cuando el usuario presiona una tecla.

Se muestran a continuación algunos de los eventos más comunes que se pueden producir en la ejecución de una aplicación por medio de sus interfaces gráficas de usuario:

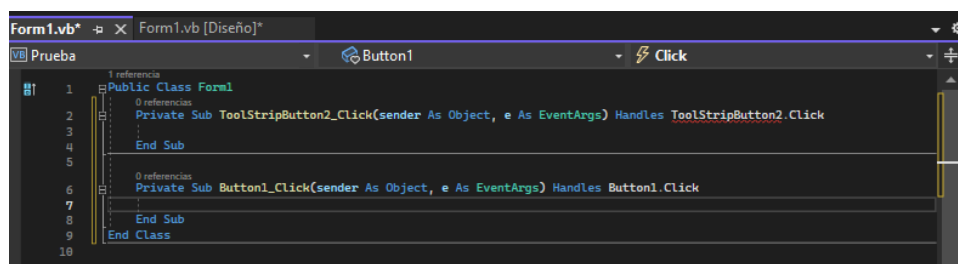
- *MouseMove*: al mover el ratón por encima de un control.
- *MouseDown*: al pulsar cualquier botón del ratón
- *Change*: al cambiar el contenido del control
- *Click*: al hacer clic con el botón izquierdo del ratón sobre el control
- *GetFocus*: este evento se activa cuando el control recibe el enfoque, es decir, cuando se activa el control en tiempo de ejecución para introducir datos o realizar alguna operación.
- *LostFocus*: es lo contrario del evento anterior, se activa cuando el control pierde el enfoque, es decir, se pasa a otro control para seguir introduciendo datos.

1.3.5 EDICIÓN DEL CÓDIGO GENERADO POR LAS HERRAMIENTAS DE DISEÑO.

Los entornos integrados de desarrollo y otras herramientas de diseño y desarrollo de interfaces son aplicaciones que permiten diseñar interfaces gráficas de usuario sin necesidad de programar una sola línea de código. Pero en muy pocos casos un desarrollador obtendrá un producto final sin necesidad de modificar algún detalle al que no habrá llegado la aplicación de desarrollo.

Por esta razón será necesario tener acceso al código generado por las herramientas de diseño, para poder modificarlo y adecuarlo a nuestras necesidades. Hay aplicaciones que integran un editor de código, lo que permite acceder sin tener que salir del entorno de desarrollo, además de poder ver los resultados de las modificaciones hechas de forma inmediata.

Es necesario que el código generado sea un código modular y sencillo de entender. Si un programador debe dedicar más tiempo a entender y hacer suyo el código, puede que no le haya sido de mucha ayuda la herramienta de desarrollo. Se puede ver un ejemplo de edición de código con Microsoft Visual Studio en la siguiente imagen:



Prácticamente todos los IDE existentes hoy en día permiten esta edición de código. Algunos ejemplos de estas herramientas pueden ser Eclipse, MonoDevelop, IDE NetBeans o Glade. Algunas de estas herramientas incluso generan código automáticamente a partir de las interfaces en desarrollo en un entorno gráfico, lo que facilita mucho el trabajo de los programadores.