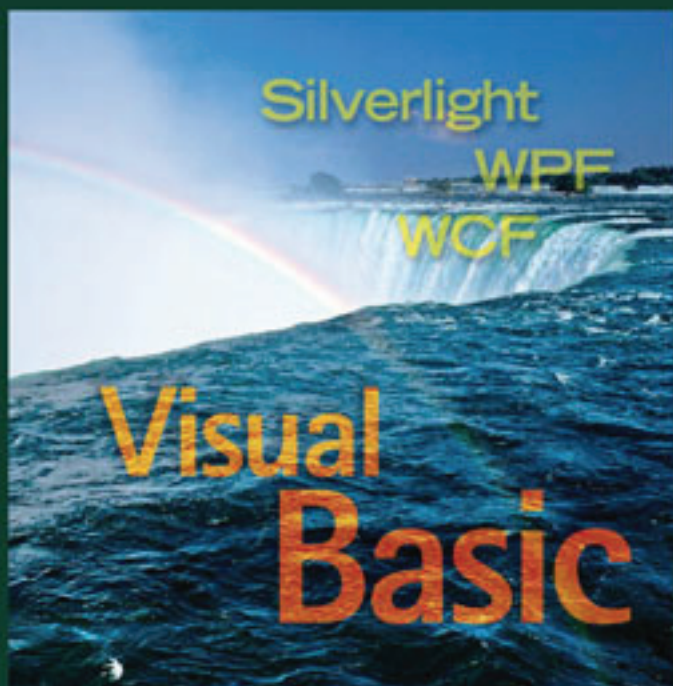


Microsoft® **Visual Basic™**

Interfaces gráficas y aplicaciones para Internet con WPF, WCF y Silverlight

- Aplicación WPF
- Introducción a WPF
- Menús y barras de herramientas
- Controles y cajas de diálogo
- Enlaces de datos en WPF
- Acceso a una base de datos



- LINQ
- Navegación de tipo Web
- Silverlight
- Servicios WCF
- Autenticación y autorización
- Acceso a datos utilizando WCF RIA Services

Fco. Javier Ceballos

WWW



Puede descargarse el CD-ROM con las URL para obtener el software de desarrollo y las aplicaciones contenidas en el libro



Ra-Ma®

Visual BasicTM

Interfaces gráficas y aplicaciones para Internet
con WPF, WCF y Silverlight

Fco. Javier Ceballos Sierra

Profesor titular de la
Escuela Politécnica Superior
Universidad de Alcalá

<http://www.fjceballos.es>





Visual Basic: Interfaces gráficas y aplicaciones para Internet con WPF, WCF y Silverlight.

© Fco. Javier Ceballos Sierra

© De la edición: RA-MA 2012

MARCAS COMERCIALES: las marcas de los productos citados en el contenido de este libro (sean o no marcas registradas) pertenecen a sus respectivos propietarios. RA-MA no está asociada a ningún producto o fabricante mencionado en la obra, los datos y los ejemplos utilizados son ficticios salvo que se indique lo contrario.

RA-MA es una marca comercial registrada.

Se ha puesto el máximo empeño en ofrecer al lector una información completa y precisa. Sin embargo, RA-MA Editorial no asume ninguna responsabilidad derivada de su uso, ni tampoco por cualquier violación de patentes ni otros derechos de terceras partes que pudieran ocurrir. Esta publicación tiene por objeto proporcionar unos conocimientos precisos y acreditados sobre el tema tratado. Su venta no supone para el editor ninguna forma de asistencia legal, administrativa ni de ningún otro tipo. En caso de precisarse asesoría legal u otra forma de ayuda experta, deben buscarse los servicios de un profesional competente.

Reservados todos los derechos de publicación en cualquier idioma.

Según lo dispuesto en el Código Penal vigente ninguna parte de este libro puede ser reproducida, grabada en sistema de almacenamiento o transmitida en forma alguna ni por cualquier procedimiento, ya sea electrónico, mecánico, reprográfico, magnético o cualquier otro, sin autorización previa y por escrito de RA-MA; su contenido está protegido por la Ley vigente que establece penas de prisión y/o multas a quienes intencionadamente, reprodujeren o plagiaran, en todo o en parte, una obra literaria, artística o científica.

Editado por:

RA-MA Editorial

C/ Jarama, 3A, polígono industrial Igarsa

28860 PARACUELLOS DEL JARAMA, Madrid

Teléfono: 91 658 42 80

Telefax: 91 662 81 39

Correo electrónico: editorial@ra-ma.com

Internet: www.ra-ma.es y www.ra-ma.com

ISBN: 978-84-9964-204-8

Depósito legal: M-24114-2012

Autoedición: Fco. Javier Ceballos

Filmación e impresión: Closas-Orcoyen, S.L.

Impreso en España

Primera impresión: julio 2012

*Si algo puede salir mal, saldrá mal.
Ley de Murphy.*

*Dedico esta obra, con mucho cariño,
a mi nieta Isabella.*

CONTENIDO

| | |
|--|--------------|
| PRÓLOGO..... | XXI |
| Para quién es este libro..... | XXIII |
| Cómo está organizado el libro..... | XXIV |
| Qué se necesita para utilizar este libro | XXV |
| Sobre los ejemplos del libro | XXV |
| Agradecimientos | XXV |
| CAPÍTULO 1. APLICACIÓN WPF | 1 |
| PROGRAMANDO EN WINDOWS..... | 3 |
| BIBLIOTECA WPF | 5 |
| ESTRUCTURA DE UNA APLICACIÓN..... | 6 |
| XAML | 8 |
| ¿Por qué XAML? | 10 |
| Código subyacente | 11 |
| INICIO DE LA APLICACIÓN..... | 12 |
| COMPILAR Y EJECUTAR LA APLICACIÓN | 15 |
| DISEÑO DE LA INTERFAZ GRÁFICA..... | 17 |
| Información básica sobre XAML..... | 17 |
| Espacios de nombres XML | 18 |
| Propiedades como atributos | 19 |
| Propiedades como elementos | 20 |
| Propiedades de contenido | 20 |
| Extensiones de marcado | 21 |
| Propiedades asociadas..... | 23 |
| Propiedades de dependencia | 24 |

| | |
|--|----|
| Crear un elemento | 26 |
| Controles más comunes | 26 |
| Añadir una etiqueta y editar sus propiedades..... | 27 |
| Añadir un botón de pulsación y editar sus propiedades..... | 28 |
| Añadir una descripción abreviada a un elemento | 29 |
| Paneles de diseño | 29 |
| Canvas..... | 29 |
| StackPanel..... | 30 |
| WrapPanel..... | 31 |
| DockPanel | 32 |
| Grid | 33 |
| MANEJO DE EVENTOS | 36 |
| Asignar manejadores de eventos a un objeto | 37 |
| EVENTOS ADJUNTOS | 38 |
| INYECTAR CÓDIGO XAML DURANTE LA EJECUCIÓN | 38 |
| CICLO DE VIDA DE UNA VENTANA | 41 |
| PROPIEDADES BÁSICAS DE LA VENTANA | 42 |
| Administración de la duración | 43 |
| Administración de ventanas | 43 |
| Apariencia y comportamiento | 43 |
| CONFIGURACIÓN DE UNA APLICACIÓN..... | 45 |
| RECURSOS DE UNA APLICACIÓN | 47 |
| ATRIBUTOS GLOBALES DE UNA APLICACIÓN..... | 48 |
| CICLO DE VIDA DE UNA APLICACIÓN..... | 49 |
| Permitir una sola instancia de la aplicación | 51 |
| Cómo se genera un evento | 52 |
| Especificar cuándo se cerrará la aplicación..... | 52 |
| Pantalla de presentación..... | 53 |
| Argumentos en la línea de órdenes | 54 |
| Acceso a la aplicación actual | 55 |
| RESUMEN..... | 56 |
| EJERCICIOS PROPUESTOS..... | 56 |

CAPÍTULO 2. INTRODUCCIÓN A WPF 57

| | |
|--|----|
| CLASES WPF..... | 57 |
| ETIQUETAS, CAJAS DE TEXTO Y BOTONES | 61 |
| Desarrollo de la aplicación..... | 63 |
| Objetos | 63 |
| Eventos..... | 63 |
| Pasos a seguir durante el desarrollo | 64 |
| El formulario, los controles y sus propiedades | 64 |

| | |
|---|-----|
| Tecla de acceso | 67 |
| Botón predeterminado..... | 67 |
| Propiedades comunes..... | 67 |
| EVENTOS ENRUTADOS | 69 |
| ¿Cómo se definen? | 71 |
| Responder a los eventos..... | 72 |
| Eventos relacionados con el teclado | 74 |
| Eventos relacionados con el foco | 77 |
| Seleccionar el texto de una caja de texto | 79 |
| Eventos relacionados con el ratón..... | 80 |
| INTERCEPTAR LA TECLA PULSADA | 84 |
| Estado del teclado | 86 |
| VALIDACIÓN DE UN CAMPO DE TEXTO | 87 |
| ENLACE DE DATOS | 89 |
| Enlace de datos sin el motor de WPF..... | 89 |
| Notificar cuándo cambia una propiedad | 93 |
| Enlace de datos con el motor de WPF | 96 |
| La clase Binding | 98 |
| Contexto de datos..... | 99 |
| Crear un enlace | 99 |
| Origen de datos implícito..... | 100 |
| Origen de datos explícito | 103 |
| Enlaces con otros controles..... | 103 |
| Conversores..... | 104 |
| Validación de datos..... | 107 |
| Regla ExceptionValidationRule..... | 108 |
| Regla DataErrorValidationRule..... | 112 |
| Información del enlace..... | 114 |
| Regla de validación personalizada..... | 116 |
| Fuentes relativas..... | 118 |
| ESTILOS Y PLANTILLAS..... | 118 |
| Estilos..... | 119 |
| Vincular controladores de eventos | 120 |
| Desencadenadores | 121 |
| Plantillas..... | 122 |
| Plantillas de control..... | 123 |
| Plantillas de datos | 125 |
| RECURSOS | 126 |
| Recursos creados mediante código | 128 |
| Recursos del sistema | 129 |
| TEMAS Y MÁSCARAS | 130 |
| RESUMEN..... | 134 |
| EJERCICIOS PROPUESTOS..... | 135 |

CAPÍTULO 3. MENÚS Y BARRAS DE HERRAMIENTAS..... 137

| | |
|---|-----|
| ARQUITECTURA DE UNA BARRA DE MENÚS..... | 137 |
| DISEÑO DE UNA BARRA DE MENÚS | 138 |
| Crear una barra de menús..... | 139 |
| Controlador de un elemento de un menú | 142 |
| Aceleradores y nemónicos | 144 |
| ÓRDENES ENRUTADAS | 145 |
| Vincular una orden enrutada con un control | 146 |
| Modelo de una orden enrutada..... | 148 |
| Cómo se ejecuta una orden enrutada..... | 150 |
| Órdenes enrutadas personalizadas..... | 153 |
| Aceleradores de teclado | 155 |
| Información adicional en las órdenes enrutadas | 159 |
| ¿Dónde se aplica la orden?..... | 161 |
| Utilizar parámetros..... | 162 |
| ICommand versus RoutedCommand | 165 |
| DETALLES DE UN ELEMENTO DE UN MENÚ | 174 |
| MENÚS CONTEXTUALES | 175 |
| BARRA DE HERRAMIENTAS..... | 177 |
| Diseño de una barra de herramientas | 177 |
| Contenedor de barras de herramientas | 180 |
| BARRA DE ESTADO | 181 |
| Diseño de una barra de estado..... | 182 |
| DESARROLLO DE UN EDITOR DE TEXTOS | 183 |
| Caja de texto multilínea | 184 |
| Diseño del editor | 185 |
| El portapapeles..... | 186 |
| Clase Clipboard..... | 187 |
| Manipular la selección del texto | 187 |
| Diseño de la barra de menús | 189 |
| Diseño de la barra de herramientas | 191 |
| Asignar a un elemento de la interfaz la tarea a realizar..... | 193 |
| Archivo - Salir..... | 193 |
| Editar - Cortar | 193 |
| Editar - Copiar..... | 194 |
| Editar - Pegar | 195 |
| Opciones - Fuente | 195 |
| Opciones - Tamaño | 197 |
| Ayuda - Acerca de | 198 |
| Eventos comunes a todos los elementos WPF | 199 |
| Habilitar o inhabilitar los elementos de un menú..... | 199 |
| Marcar el elemento seleccionado de un menú | 201 |

| | |
|--|-----|
| Deshacer y rehacer | 202 |
| Recordar las ediciones reversibles | 202 |
| Añadir a la interfaz la orden Deshacer | 203 |
| Añadir a la interfaz la orden Rehacer | 204 |
| Menú contextual | 204 |
| Asociar un icono a la aplicación | 204 |
| MENÚS DINÁMICOS | 204 |
| RESUMEN | 208 |

CAPÍTULO 4. CONTROLES Y CAJAS DE DIÁLOGO 211

| | |
|--|-----|
| CAJAS DE DIÁLOGO MODALES Y NO MODALES | 212 |
| CAJAS DE MENSAJE | 212 |
| CAJAS DE DIÁLOGO PERSONALIZADAS | 215 |
| Crear una caja de diálogo | 217 |
| Mostrar una caja de diálogo | 219 |
| Gestionar los botones Aceptar y Cancelar | 220 |
| Introducción de datos y recuperación de los mismos | 221 |
| DIÁLOGO ACERCA DE | 223 |
| VENTANA PROPIETARIA | 225 |
| OTROS CONTROLES WPF | 226 |
| Casillas de verificación | 227 |
| Botones de opción | 232 |
| GroupBox y Expander | 237 |
| Listas simples | 238 |
| Diseñar la lista | 241 |
| Iniciar la lista | 242 |
| Acceder a los elementos seleccionados | 242 |
| Colección de elementos de una lista | 243 |
| Lista de elementos de tipo CheckBox | 245 |
| Listas desplegables | 246 |
| Diseñar la lista | 247 |
| Iniciar la lista | 249 |
| Acceder al elemento seleccionado | 249 |
| Colección de elementos de una lista desplegable | 249 |
| Controles de rango definido | 251 |
| ScrollBar | 251 |
| Slider | 255 |
| ProgressBar | 256 |
| Visor con barras de desplazamiento | 258 |
| Control con pestañas | 259 |
| Gestión de fechas | 260 |

| | |
|--|-----|
| ListView..... | 262 |
| TreeView..... | 264 |
| Guardar el documento XML..... | 269 |
| Recargar el documento XML..... | 269 |
| Expandir o contraer los nodos..... | 270 |
| DataGrid..... | 271 |
| Columnas del DataGrid..... | 271 |
| Inmovilizar columnas..... | 274 |
| Filas del DataGrid..... | 274 |
| Detalles de las filas..... | 276 |
| CAJAS DE DIÁLOGO ESTÁNDAR..... | 278 |
| Cajas de diálogo Abrir y Guardar..... | 278 |
| Cajas de diálogo Windows Forms estándar..... | 281 |
| Caja de diálogo Imprimir..... | 282 |
| CONTROLES DE DOCUMENTOS WPF..... | 283 |
| Documentos dinámicos..... | 284 |
| Elementos Block..... | 285 |
| Elementos Inline..... | 289 |
| Paragraph y Run..... | 291 |
| Interactuando con los elementos mediante programación..... | 291 |
| Acceso a documentos en un fichero..... | 297 |
| Editar un documento..... | 297 |
| Imprimir un documento..... | 300 |
| TEMPORIZADORES Y MODELO DE SUBPROCESOS..... | 303 |
| Timer..... | 304 |
| Resolución del temporizador..... | 306 |
| DispatcherTimer..... | 308 |
| RESUMEN..... | 309 |
| EJERCICIOS PROPUESTOS..... | 309 |

CAPÍTULO 5. ENLACE DE DATOS EN WPF..... 315

| | |
|--|-----|
| ASPECTOS BÁSICOS..... | 315 |
| ENLACE A COLECCIONES DE OBJETOS..... | 317 |
| Cómo implementar colecciones..... | 317 |
| Vistas de colección..... | 318 |
| PLANTILLAS DE DATOS..... | 320 |
| Definir una plantilla de datos..... | 322 |
| Mejorar la presentación..... | 324 |
| Utilizar desencadenadores para aplicar valores de propiedad..... | 325 |
| XML COMO FUENTE DE DATOS..... | 327 |
| Datos jerárquicos..... | 328 |

| | |
|---|------------|
| Islas de datos | 331 |
| Soporte .Net para trabajar con XML | 331 |
| Obtener la vista | 333 |
| Elemento actual | 333 |
| Navegar | 334 |
| Ordenar | 335 |
| Filtrar | 336 |
| Agrupar | 337 |
| Fuente de datos XML sin el proveedor | 337 |
| Vinculación maestro-detalle | 339 |
| OBJETOS COMO FUENTE DE DATOS | 343 |
| Enlace a una colección de objetos | 345 |
| Vistas de colección de objetos | 347 |
| Obtener la vista | 350 |
| Elemento actual | 350 |
| Navegar | 351 |
| Ordenar | 352 |
| Filtrar | 352 |
| Agrupar | 353 |
| Insertar y borrar elementos de la colección | 353 |
| Vinculación maestro-detalle | 354 |
| Proveedor de datos de objetos | 357 |
| Virtualización | 359 |
| Datos introducidos por el usuario | 360 |
| Solicitar datos al usuario | 361 |
| Validación | 366 |
| Visualización de los errores de validación | 368 |
| Regla de validación personalizada | 370 |
| Permanecer en la caja de diálogo si hay errores | 373 |
| Grupos de enlaces | 374 |
| DataGrid | 376 |
| Columnas del DataGrid | 378 |
| Inmovilizar columnas | 379 |
| Filas del DataGrid | 379 |
| Selección de celdas | 380 |
| Detalles de las filas | 381 |
| Filtrado, agrupación y ordenación | 383 |
| Validación | 383 |
| RESUMEN | 383 |

| | |
|---|------------|
| CAPÍTULO 6. ACCESO A UNA BASE DE DATOS | 385 |
| SQL | 386 |
| Crear una base de datos..... | 386 |
| Crear una tabla | 386 |
| Escribir datos en la tabla | 388 |
| Modificar datos de una tabla | 388 |
| Borrar registros de una tabla | 389 |
| Seleccionar datos de una tabla | 389 |
| Crear una base de datos..... | 391 |
| Base de datos Microsoft Access..... | 391 |
| Base de datos Microsoft SQL Server | 393 |
| ADO.NET | 394 |
| Componentes de ADO.NET..... | 395 |
| Conjunto de datos..... | 396 |
| Proveedor de datos | 398 |
| Objeto conexión | 399 |
| Objeto orden | 401 |
| Objeto lector de datos | 401 |
| Adaptador de datos | 402 |
| Modos de conexión | 404 |
| Probando una conexión..... | 405 |
| Servicio de conexiones..... | 407 |
| ACCESO CONECTADO A BASE DE DATOS..... | 408 |
| ATAQUES DE INYECCIÓN DE CÓDIGO SQL..... | 411 |
| Órdenes parametrizadas | 414 |
| Procedimientos almacenados | 415 |
| TRANSACCIONES..... | 417 |
| Transacción implícita TransactionScope | 417 |
| Transacciones explícitas..... | 421 |
| CONSTRUIR COMPONENTES DE ACCESO A DATOS..... | 424 |
| Capa de presentación | 425 |
| Operaciones contra la base de datos..... | 427 |
| Objetos de negocio..... | 428 |
| Capa de acceso a datos..... | 431 |
| Capa de lógica de negocio | 435 |
| Lógica de interacción con la capa de presentación | 436 |
| Desacoplar la IU del resto de la aplicación | 439 |
| Adaptar la colección de objetos | 440 |
| Capa de lógica de negocio | 443 |
| Lógica de interacción con la capa de presentación | 446 |
| Validación | 447 |
| ACCESO DESCONECTADO A BASE DE DATOS | 451 |

| | |
|---|-----|
| Crear la base de datos..... | 455 |
| Crear un proyecto WPF..... | 456 |
| Conectarse a la base de datos Sql Server | 457 |
| Crear la capa de acceso a datos | 458 |
| Capa de lógica de negocio | 461 |
| Lógica de interacción con la capa de presentación | 463 |
| Actualizaciones | 465 |
| Clase DataView..... | 467 |
| RESUMEN..... | 470 |

CAPÍTULO 7. LINQ..... 471

| | |
|--|-----|
| RECURSOS DEL LENGUAJE COMPATIBLES CON LINQ..... | 471 |
| Declaración implícita de variables locales | 472 |
| Matrices de tipos definidos de forma implícita..... | 472 |
| Tipos anónimos..... | 472 |
| Propiedades auto-implementadas..... | 473 |
| Iniciadores de objetos y colecciones | 473 |
| Métodos extensores..... | 474 |
| Expresiones lambda | 475 |
| El delegado Func(Of T, TResu)..... | 477 |
| Operadores de consulta | 478 |
| Árboles de expresiones lambda..... | 481 |
| EXPRESIONES DE CONSULTA..... | 484 |
| Compilación de una expresión de consulta | 487 |
| Sintaxis de las expresiones de consulta..... | 489 |
| Cláusula Group | 489 |
| Productos cartesianos..... | 490 |
| Cláusula Join..... | 490 |
| Cláusula Into | 491 |
| Cláusula Let | 492 |
| PROVEEDORES DE LINQ | 493 |
| ENTITY FRAMEWORK | 494 |
| MARCO DE ENTIDADES DE ADO.NET | 495 |
| Consultar un modelo de objetos..... | 499 |
| ACCESO A UNA BASE DE DATOS..... | 501 |
| Conectarse a la base de datos | 502 |
| Generar el modelo de entidades | 502 |
| Las clases de entidad y el contexto de objetos | 509 |
| Propiedades de navegación | 511 |
| Mostrar datos en una interfaz gráfica..... | 513 |
| Una aplicación con interfaz gráfica..... | 514 |

| | |
|---|-----|
| Vincular controles con el origen de datos | 516 |
| Filtros | 521 |
| Contextos de corta duración..... | 522 |
| REALIZAR CAMBIOS EN LOS DATOS..... | 522 |
| Modificar filas en la base de datos | 525 |
| Insertar filas en la base de datos..... | 526 |
| Borrar filas en la base de datos | 529 |
| Problemas de concurrencia | 532 |
| El seguimiento de cambios..... | 535 |
| EJERCICIOS RESUELTOS | 538 |
| RESUMEN..... | 543 |
| EJERCICIOS PROPUESTOS..... | 544 |

CAPÍTULO 8. NAVEGACIÓN DE TIPO WEB..... 545

| | |
|---|-----|
| WPF, XBAP y Silverlight..... | 545 |
| NAVEGACIÓN | 546 |
| Crear la base de datos..... | 547 |
| Crear el proyecto..... | 548 |
| NavigationWindow | 549 |
| Page..... | 551 |
| Añadir páginas a la aplicación | 552 |
| Diseño de la interfaz gráfica | 554 |
| Lógica de negocio | 561 |
| Pasar datos entre páginas | 563 |
| Duración y diario de las páginas..... | 564 |
| Hyperlinks..... | 565 |
| Frame | 567 |
| Funciones de página..... | 569 |
| Diseño | 571 |
| Lógica de negocio | 574 |
| APLICACIÓN XBAP | 576 |
| Publicar la aplicación | 577 |
| Seguridad | 581 |
| ACCESO A UNA BASE DE DATOS DESDE UNA XBAP..... | 583 |
| Crear la base de datos..... | 583 |
| Conectarse a la base de datos | 584 |
| Generar el modelo de entidades | 584 |
| Interfaz gráfica | 585 |
| Vincular controles con el origen de datos | 586 |
| Controles de usuario..... | 590 |
| Modificar registros..... | 593 |

| | |
|--------------------------------------|-----|
| Guardar los cambios realizados | 595 |
| Añadir un nuevo registro..... | 596 |
| Borrar un registro | 602 |
| EL CONTROL WEBBROWSER..... | 602 |
| RESUMEN..... | 603 |

CAPÍTULO 9. SILVERLIGHT 605

| | |
|--|-----|
| ARQUITECTURA..... | 606 |
| CREAR UNA APLICACIÓN SILVERLIGHT | 608 |
| Arquitectura de la aplicación Silverlight..... | 609 |
| Compilación de la aplicación Silverlight..... | 612 |
| Página de entrada | 613 |
| DISEÑAR UNA PÁGINA SILVERLIGHT | 615 |
| Controles Silverlight | 615 |
| Redistribuir el espacio de los elementos de un Grid..... | 615 |
| Texto estático | 618 |
| Imágenes | 619 |
| Controles de contenido | 620 |
| Atributos de anotación de datos..... | 621 |
| Diseño de la interfaz | 622 |
| Contexto de datos | 624 |
| TextBox | 624 |
| DescriptionViewer..... | 625 |
| ValidationSummary | 625 |
| Label | 625 |
| Validación de los datos | 625 |
| Origen de los datos | 627 |
| Controles de elementos..... | 629 |
| Controles de texto y elementos de texto | 631 |
| Controles de rango definido..... | 633 |
| Controles para gestionar fechas | 633 |
| Degradados | 634 |
| Ventanas y cajas de diálogo..... | 636 |
| Popup | 636 |
| ChildWindow..... | 637 |
| GRÁFICOS, ANIMACIÓN Y MULTIMEDIA | 642 |
| Gráficos..... | 642 |
| Transformaciones..... | 643 |
| Animaciones..... | 647 |
| Audio y vídeo..... | 654 |
| NAVEGACIÓN | 664 |

| | |
|---|-----|
| Navegación personalizada..... | 665 |
| Navegación de Silverlight..... | 666 |
| Frame | 667 |
| Administrador de identificadores de recursos..... | 671 |
| Navegación externa..... | 674 |
| Extender el sistema de navegación | 675 |
| Compatibilidad de ejecución fuera del explorador | 675 |
| Plantilla aplicación de navegación de Silverlight | 676 |
| ACCESO A DATOS..... | 676 |
| Acceso a los datos de una colección | 677 |
| Crear la base de datos | 681 |
| Crear una aplicación Silverlight..... | 683 |
| Vincular controles con el origen de datos | 684 |
| Paginación controlada | 687 |
| Paginación personalizada | 688 |
| Filtrar los registros de la colección | 693 |
| Trabajar con imágenes | 694 |
| Cargar una nueva imagen..... | 699 |
| Guardar los cambios realizados | 701 |
| Añadir un nuevo registro..... | 701 |
| Borrar un registro | 705 |
| PUBLICAR LA APLICACIÓN..... | 706 |
| RESUMEN..... | 708 |

CAPÍTULO 10. SERVICIOS WCF..... 711

| | |
|---|-----|
| MODELO DE PROGRAMACIÓN DE WCF | 712 |
| Implementar un servicio WCF | 712 |
| Definir un contrato | 714 |
| Implementar un cliente WCF | 720 |
| Configuración del cliente | 724 |
| Obtener acceso al servicio WCF | 725 |
| Comunicación entre dominios..... | 728 |
| Publicar la aplicación | 730 |
| SERVICIOS WCF HABILITADOS PARA SILVERLIGHT | 734 |
| Crear un servicio WCF habilitado para Silverlight..... | 734 |
| Implementar un cliente WCF | 738 |
| Añadir una referencia al servicio | 738 |
| Publicar la aplicación | 740 |
| SERVICIOS WEB Y LINQ..... | 743 |
| Arquitectura de N capas lógicas y N niveles físicos | 744 |
| Crear la base de datos..... | 745 |

| | |
|--|-----|
| Obtener acceso a la base de datos | 746 |
| Crear el servicio WCF | 748 |
| Ciente Silverlight | 756 |
| Llenar la lista | 761 |
| Mensajes para el usuario | 762 |
| Ordenar la lista | 763 |
| Mostrar datos | 764 |
| Actualizar datos | 766 |
| Actualizar la foto | 767 |
| Agregar datos | 767 |
| Borrar datos | 768 |
| Publicar el servicio WCF y la aplicación Silverlight | 769 |
| RESUMEN | 775 |

CAPÍTULO 11. AUTENTICACIÓN Y AUTORIZACIÓN 777

| | |
|---|-----|
| SERVICIOS DE AUTENTICACIÓN | 778 |
| Autenticación de Windows | 779 |
| Autenticación mediante formularios | 779 |
| Clase FormsAuthentication | 781 |
| Autenticación mediante formularios en Silverlight | 782 |
| SERVICIOS DE APLICACIÓN DE ASP.NET | 790 |
| Crear la estructura de la aplicación | 791 |
| Asignar y configurar servicios de aplicación | 793 |
| Crear usuarios | 796 |
| Autenticación | 800 |
| Funciones (roles) | 804 |
| Perfiles | 808 |
| Autorización de ASP.NET | 808 |
| SIMPLIFICAR EL DESARROLLO DE APLICACIONES | 813 |
| Plantilla aplicación de negocios Silverlight | 815 |
| Autenticación, funciones y perfiles | 819 |
| RESUMEN | 823 |

CAPÍTULO 12. ACCESO A DATOS UTILIZANDO RIA SERVICES 825

| | |
|---|-----|
| ACCESO A DATOS | 826 |
| Crear y configurar la solución | 827 |
| Mostrar datos utilizando la clase LoadOperation | 828 |
| Generar el modelo de entidades | 828 |
| Agregar un servicio de dominio | 829 |
| LoadOperation | 832 |

| | |
|--|------------|
| DomainDataSource | 839 |
| Parámetros de consulta | 841 |
| Ordenar, filtrar y agrupar | 842 |
| Paginación..... | 842 |
| Actualizar la base de datos..... | 843 |
| Añadir nuevos registros | 848 |
| Borrar registros | 851 |
| RESUMEN..... | 852 |
| | |
| APÉNDICE A. ENTORNO DE DESARROLLO INTEGRADO | 855 |
| | |
| MICROSOFT VISUAL STUDIO..... | 855 |
| Crear un nuevo proyecto | 857 |
| El formulario | 861 |
| Dibujar los controles | 862 |
| Borrar un control..... | 866 |
| Propiedades de los objetos | 867 |
| Icono de la aplicación | 869 |
| Escribir los controladores de eventos..... | 869 |
| Guardar la aplicación | 872 |
| Verificar la aplicación..... | 872 |
| Propiedades del proyecto | 874 |
| Crear soluciones de varios proyectos..... | 875 |
| Opciones del EDI..... | 876 |
| Personalizar el EDI | 877 |
| SQL SERVER EXPRESS..... | 877 |
| SQL SERVER MANAGEMENT STUDIO EXPRESS..... | 880 |
| EXPLORADOR DE BASES DE DATOS..... | 881 |
| AÑADIR UN DATASET AL PROYECTO | 883 |
| Esquemas XSD | 886 |
| Base de datos XML..... | 887 |
| VISUAL WEB DEVELOPER..... | 891 |
| INSTALACIÓN DE ASP.NET EN WINDOWS..... | 891 |
| Registro manual de ASP.NET en IIS | 891 |
| | |
| APÉNDICE B. CD..... | 893 |
| | |
| ÍNDICE | 895 |

PRÓLOGO

Visual Basic es hoy uno de los lenguajes de programación más populares del mundo. Desde que Microsoft liberó Visual Basic 1.0 en 1991 han tenido lugar muchos cambios. Visual Basic 1.0 revolucionó la forma de desarrollar software para Windows; desmitificó el proceso de desarrollo de aplicaciones con interfaz gráfica de usuario y abrió este tipo de programación a las masas. En sus posteriores versiones, Visual Basic ha continuado proporcionando nuevas características que facilitaron la creación de aplicaciones para Windows cada vez más potentes; por ejemplo, la versión 3.0 introdujo el control de datos para facilitar el acceso a bases de datos, y la versión 4.0 mejoró y potenció este acceso con los objetos DAO. Con la aparición de Windows 95, Microsoft liberó Visual Basic 4.0, que abrió la puerta al desarrollo de aplicaciones de 32 bits y a la creación de DLL. La versión 5.0 mejoró la productividad con la incorporación de la ayuda inteligente y la introducción de los controles ActiveX. Posteriormente la versión 6.0 nos introdujo en la programación de Internet con las aplicaciones DHTML y el objeto *WebClass*. Después dispusimos de Visual Basic .NET, que vino a revolucionar el mundo de las comunicaciones permitiendo escribir aplicaciones escalables para Internet. Siguió Visual Basic 2005, Visual Basic 2008 y ahora Visual Basic 2010, una evolución del lenguaje Visual Basic, que se diseñó para generar aplicaciones con seguridad de tipos y orientadas a objetos de manera productiva. Esta generación de Visual Basic continúa la tradición de ofrecer una manera rápida y fácil de crear aplicaciones basadas en .NET Framework.

Visual Basic .NET, después Visual Basic 2005, Visual Basic 2008 y ahora Visual Basic 2010, cambia la idea de programar de las versiones iniciales. Ahora se requiere una programación orientada a objetos, lo que obligará al desarrollador a programar de forma ordenada, con unas reglas metodológicas de programación análogas a las de otros lenguajes de programación orientados a objetos como C++, C# o Java por citar algunos de los más utilizados.

La palabra “Visual” hace referencia, desde el lado del diseño, al método que se utiliza para crear la interfaz gráfica de usuario si se dispone de la herramienta adecuada (con *Microsoft Visual Studio* se utiliza el ratón para arrastrar y colocar los objetos prefabricados en el lugar deseado dentro de un formulario) y desde el lado de la ejecución, al aspecto gráfico que toman los objetos cuando se ejecuta el código que los crea, objetos que formarán la interfaz gráfica que el usuario de la aplicación utiliza para acceder a los servicios que ésta ofrece. La palabra “Basic” hace referencia al lenguaje BASIC (*Beginners All-Purpose Symbolic Instruction Code*), un lenguaje utilizado por más programadores que ningún otro lenguaje en la historia de la informática. Visual Basic ha evolucionado a partir del lenguaje BASIC original y ahora está soportado por una biblioteca orientada a objetos directamente relacionada con la interfaz gráfica de Windows. Y “NET” hace referencia al ámbito donde operarán nuestras aplicaciones webs (*Network* - red).

De forma resumida, Visual Basic es un lenguaje orientado a objetos seguro y elegante que permite a los desarrolladores construir un amplio rango de aplicaciones seguras y robustas que se ejecutan sobre .NET Framework. Podemos utilizar Visual Basic para crear aplicaciones cliente Windows tradicionales, servicios webs XML, servicios WCF, componentes distribuidos, aplicaciones cliente servidor, aplicaciones para acceso a bases de datos, y muchas otras. *Microsoft Visual Basic 2010* y superiores proporcionan un editor de código avanzado, diseñadores de interfaces de usuario apropiados, depurador integrado, y muchas otras utilidades para facilitar el desarrollo rápido de aplicaciones basadas en el lenguaje Visual Basic y en .NET Framework.

Por otra parte, .NET es una plataforma de desarrollo compuesta básicamente por el CLR (la máquina virtual), la BCL (la biblioteca básica) y un conjunto de lenguajes de programación, entre los que se encuentra Visual Basic. .NET partió con la versión 1.0 y pasó por las 1.1, 2.0, 3.0, 3.5 y 4.0, que es la versión en el momento de escribir esta obra, la cual se incluye con Visual Studio 2010 y con Windows 7 e incluye, entre otras cosas, LINQ, WPF, WCF, WF, ASP.NET 4.0, AJAX, VSTO y Silverlight. ¿Y todo esto para qué? Pues para crear aplicaciones de Internet más ricas. Se trata realmente de nuevas bibliotecas que nos permiten hacer lo que ya hacíamos con .NET 2.0, pero de una forma más vistosa, más rápida y más eficiente. Si miramos hacia atrás (Win32), teníamos la biblioteca MFC. Después vino .NET y ahora vienen otras bibliotecas que tratan de facilitar al desarrollador la realización de aplicaciones más ricas en menos tiempo.

El contenido de este libro se va a centrar básicamente en las tecnologías WPF, WCF y Silverlight.

WPF (*Windows Presentation Foundation*) es una biblioteca para el desarrollo de interfaces gráficas de usuario vectoriales avanzadas. Esta biblioteca de clases no ha sido creada para sustituir a *Windows Forms*, sino que es otra biblioteca que

facilita el desarrollo de aplicaciones de escritorio en las que estén implicados diversos tipos de medios: vídeo, documentos, contenido 3D, secuencias de imágenes animadas, etc. WPF también es idóneo si lo que se necesita es crear una interfaz de usuario con un aspecto personalizado, si hay que establecer vínculos con datos, o si desea crear una aplicación de escritorio con un estilo de navegación similar a una aplicación web. A diferencia de *Windows Forms*, utiliza el lenguaje de marcado XAML para implementar su interfaz gráfica y los lenguajes de programación administrados, como Visual Basic, para escribir el código subyacente que implemente su comportamiento. Esta separación entre la apariencia y el comportamiento permite a los diseñadores implementar la apariencia de una aplicación al mismo tiempo que los programadores implementan su comportamiento.

WCF (*Windows Communication Foundation*) es un marco de trabajo unificado para hacer fácil la comunicación entre aplicaciones diversas en cualquier plataforma (.NET, J2EE, etc.) combinando en una sola tecnología lo que en versiones anteriores de Visual Studio existía en varias tecnologías: servicios webs ASMX, .NET Remoting, Enterprise Services y Message Queue Server. WPF está orientado al servicio. Los servicios son autónomos y comparten esquemas (datos) y contratos (funcionalidad), no clases ni tipos en general y al intercambiar mensajes no tienen que asumir nada acerca de “qué es lo que hay al otro lado del extremo”. Los clientes consumen servicios y los servicios ofrecen soluciones a los clientes intercambiando mensajes, con lo que un servicio puede, a su vez, ser cliente de otro servicio.

Silverlight es una tecnología multiplataforma que se ejecuta en varios exploradores. Al igual que Flash, Silverlight permite crear contenido interactivo que se ejecuta en el cliente, con soporte para gráficos dinámicos, contenido multimedia y animación, que va mucho más allá del HTML ordinario y también, al igual que Flash, Silverlight se implementa en los navegadores mediante un plug-in. Silverlight incluye una versión reducida de .NET Framework y de WPF, lo que permite a los desarrolladores escribir código de cliente utilizando Visual Basic o C#.

Para quién es este libro

Este libro está pensado para aquellas personas que quieran aprender a desarrollar aplicaciones que muestren una interfaz gráfica al usuario, aplicaciones para acceso a bases de datos y para Internet, utilizando básicamente las bibliotecas WPF, WCF, Silverlight y el entorno de desarrollo Visual Studio 2010 o superior. Para ello, ¿qué debe hacer? Pues simplemente leer ordenadamente los capítulos del libro, resolviendo cada uno de los ejemplos que en ellos se detallan.

Evidentemente, el autor asume que el lector conoce el lenguaje *Visual Basic*, ya que este libro no describe este lenguaje debido a que este tema fue expuesto

pormenorizadamente en sus otros libros *Microsoft Visual Basic .NET - Lenguaje y aplicaciones* o *Microsoft Visual Basic .NET - Curso de programación*, ambos editados también por RA-MA, y que tiene conocimientos orientados al desarrollo de aplicaciones utilizando la biblioteca *Windows Forms* y el entorno de desarrollo ASP.NET, temática que tampoco se expone porque fue expuesta en su otro libro *Enciclopedia de Microsoft Visual Basic*.

Cómo está organizado el libro

El libro se ha estructurado en 12 capítulos más algunos apéndices que a continuación se relacionan. El capítulo 1 estudia los conceptos básicos de WPF y nos introduce en el desarrollo de una aplicación WPF. En el capítulo 2 se hace una introducción a la jerarquía de clases de WPF, al uso de los controles y eventos más frecuentes, a la validación de datos y a la personalización de la apariencia de una aplicación. El capítulo 3 nos enseña cómo añadir una barra de menús, de herramientas o de estado a una ventana WPF, cómo añadir un menú contextual y a utilizar las órdenes enrutadas. El capítulo 4 explica cómo utilizar multitud de controles WPF en el diseño de interfaces gráficas y cómo apoyar estos diseños con cajas de diálogo. En el capítulo 5 se estudia el enlace a datos, uno de los pilares de WPF, y las colecciones de objetos, ya que éstas serán los orígenes de los datos que serán proporcionados por los enlaces a los controles de la interfaz gráfica del usuario y viceversa. El capítulo 6 cubre el acceso a bases de datos utilizando ADO.NET y el desarrollo de aplicaciones basado en capas. El capítulo 7 continúa con el acceso a bases de datos, pero utilizando el lenguaje de consultas integrado LINQ, para después centrarnos en el proveedor LINQ to Entities que permite consultar las entidades que definen el modelo conceptual de Entity Framework. El capítulo 8 nos enseña cómo utilizar el modelo de navegación de WPF, un modelo basado en páginas. En el capítulo 9 se estudia la tecnología Silverlight y cómo desarrollar aplicaciones para la web o no, utilizando esta tecnología. En el capítulo 10 se estudian los servicios WCF y se expone cómo desarrollar una aplicación Silverlight de N capas y N niveles, que tiene que acceder a una base de datos a través de servicios WCF. El capítulo 11 estudia cómo implementar la autenticación y la autorización en una aplicación Silverlight. Y el capítulo 12 estudia el acceso a datos utilizando WCF RIA Services.

CAPÍTULO 1. APLICACIÓN WPF

CAPÍTULO 2. INTRODUCCIÓN A WPF

CAPÍTULO 3. MENÚS Y BARRAS DE HERRAMIENTAS

CAPÍTULO 4. CONTROLES Y CAJAS DE DIÁLOGO

CAPÍTULO 5. ENLACES DE DATOS EN WPF

CAPÍTULO 6. ACCESO A UNA BASE DE DATOS

CAPÍTULO 7. LINQ

CAPÍTULO 8. NAVEGACIÓN DE TIPO WEB

CAPÍTULO 9. SILVERLIGHT

CAPÍTULO 10. SERVICIOS WCF

CAPÍTULO 11. AUTENTICACIÓN Y AUTORIZACIÓN

CAPÍTULO 12. ACCESO A DATOS UTILIZANDO WCF RIA SERVICES

APÉNDICE A. ENTORNO DE DESARROLLO INTEGRADO

Qué se necesita para utilizar este libro

Este libro ha sido escrito utilizando el paquete *Microsoft .NET Framework Software Development Kit* (SDK) versión 4.0 incluido en el entorno de desarrollo *Microsoft Visual Studio 2010* (o en su defecto *Visual Basic 2010 Express*, *Visual Web Developer 2010 Express* y *SQL Server 2010 Express*) que incluye todo lo necesario para escribir, construir, verificar y ejecutar aplicaciones .NET. Por lo tanto, basta con que instale en su máquina el software mencionado. Las versiones *Express*, que puede descargar desde <http://www.microsoft.com/express/>, son gratuitas.

Nota: para probar las aplicaciones webs se recomienda instalar el servidor de aplicaciones IIS (*Internet Information Server*) incluido con Windows (*Inicio > Panel de control > Agregar y quitar programas > Windows*). Esto tiene que hacerlo antes de instalar *Microsoft Visual Studio 2010* o *Visual Web Developer 2010 Express*.

Sobre los ejemplos del libro

El código fuente de todos los ejemplos del libro podrá descargarse, según se indica en los apéndices, de la web www.ra-ma.es desde la página web correspondiente al libro.

Agradecimientos

He recibido ayuda de algunas personas durante la preparación de este libro, y por ello les estoy francamente agradecido. También, deseo expresar mi agradecimiento a *Microsoft Ibérica* por poner a mi disposición, en particular, y de todos los lectores, en general, el software que el estudio de esta obra requiere.

Francisco Javier Ceballos Sierra

<http://www.fjceballos.es/>

ENTORNO DE DESARROLLO INTEGRADO

Se puede desarrollar una aplicación que muestre una interfaz gráfica utilizando como herramientas *Microsoft Framework SDK* (proporciona, entre otras cosas, la biblioteca de clases .NET y el compilador de Visual Basic) y un simple editor de texto, o bien utilizando un entorno de desarrollo integrado (EDI). En el primer caso hay que escribir el código fuente línea a línea, para después, desde la línea de órdenes, compilarlo, ejecutarlo y depurarlo. Lógicamente, escribir todo el código necesario para crear la interfaz gráfica de la aplicación es una tarea repetitiva que, de poder mecanizarse, ahorraría mucho tiempo en la implementación de una aplicación y permitiría centrarse más y mejor en resolver los problemas relativos a su lógica y no a su aspecto. Justamente esto es lo nuevo que aporta *Visual Studio* o, en su defecto, las versiones de *Visual Basic Express* y *Visual Web Developer Express*.

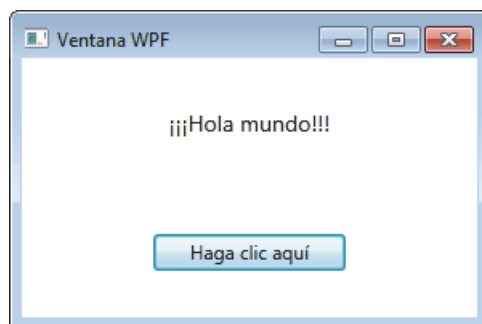
El propósito de este apéndice es mostrar cómo crear una aplicación sencilla de *Windows Presentation Foundation* (WPF) y familiarizarse con el entorno de desarrollo integrado (EDI) de *Visual Studio/Visual Basic Express*. Al igual que las aplicaciones de formularios Windows Forms, las aplicaciones WPF se pueden diseñar arrastrando controles desde la caja de herramientas hasta el panel de diseño.

MICROSOFT VISUAL STUDIO

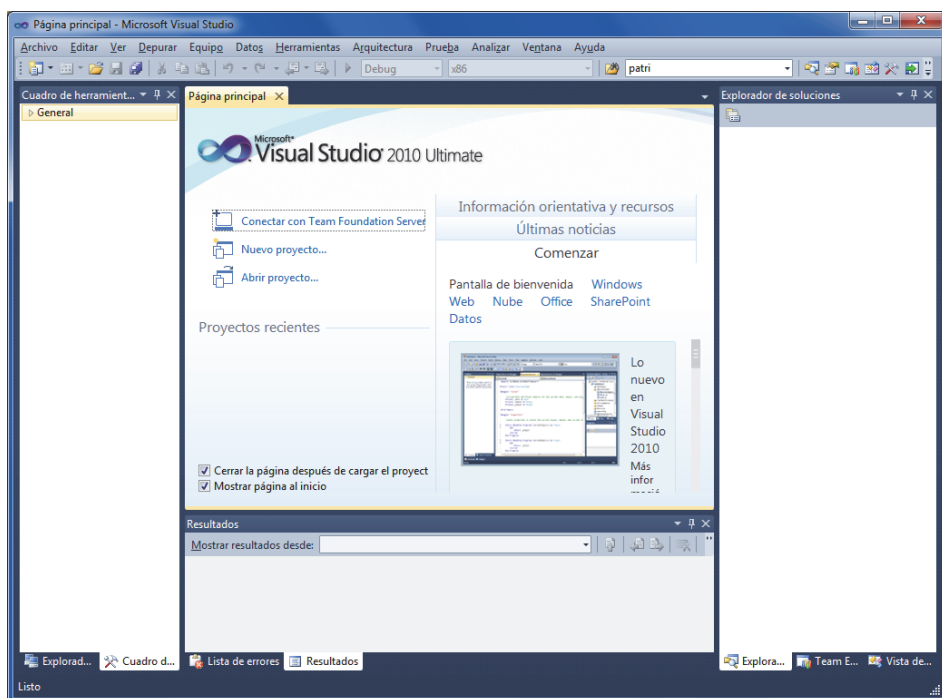
Visual Studio permite diseñar la interfaz gráfica de una aplicación WPF de manera visual, sin más que arrastrar con el ratón los controles que necesitemos sobre la ventana destino de los mismos. Unas líneas de guía o una rejilla mostrada sobre el formulario (o ventana) nos ayudarán a colocar estos controles y a darles el tamaño adecuado, y una página de propiedades nos facilitará la modificación de los valo-

res de las propiedades de cada uno de los controles. Todo lo expuesto lo realizaremos sin tener que escribir ni una sola línea de código. Después, un editor de código inteligente nos ayudará a escribir el código necesario y detectará los errores sintácticos que introduzcamos, y un depurador nos ayudará a poner a punto nuestra aplicación cuando lo necesitemos.

Como ejemplo, vamos a realizar una aplicación Windows denominada *Saludo*, que presente una interfaz al usuario como la de la figura siguiente:



Para empezar, arranque Visual Studio. Se visualizará una ventana similar a la siguiente:

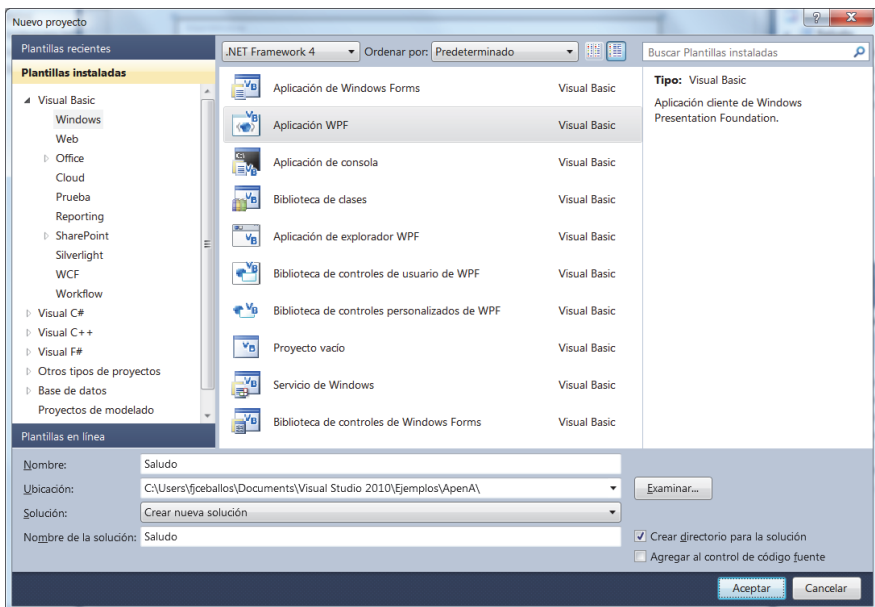


¿Cuáles son los siguientes pasos para desarrollar una aplicación WPF? En general, para construir una aplicación de este tipo con Visual Studio, siga los pasos indicados a continuación:

1. Cree un nuevo proyecto (una nueva aplicación), entendiendo por proyecto un conjunto de ficheros, normalmente distribuidos en carpetas y recursos que pueden ser compilados como una sola unidad. Visual Studio mostrará una página de diseño con un formulario vacío por omisión (una ventana).
2. Dibuje los controles sobre el formulario. Los controles serán tomados de una caja de herramientas.
3. Defina las propiedades del formulario y de los controles.
4. Escriba el código para controlar los eventos que consideremos de cada uno de los objetos.
5. Guarde, compile y ejecute la aplicación.
6. Opcionalmente, utilice un depurador para poner a punto la aplicación.

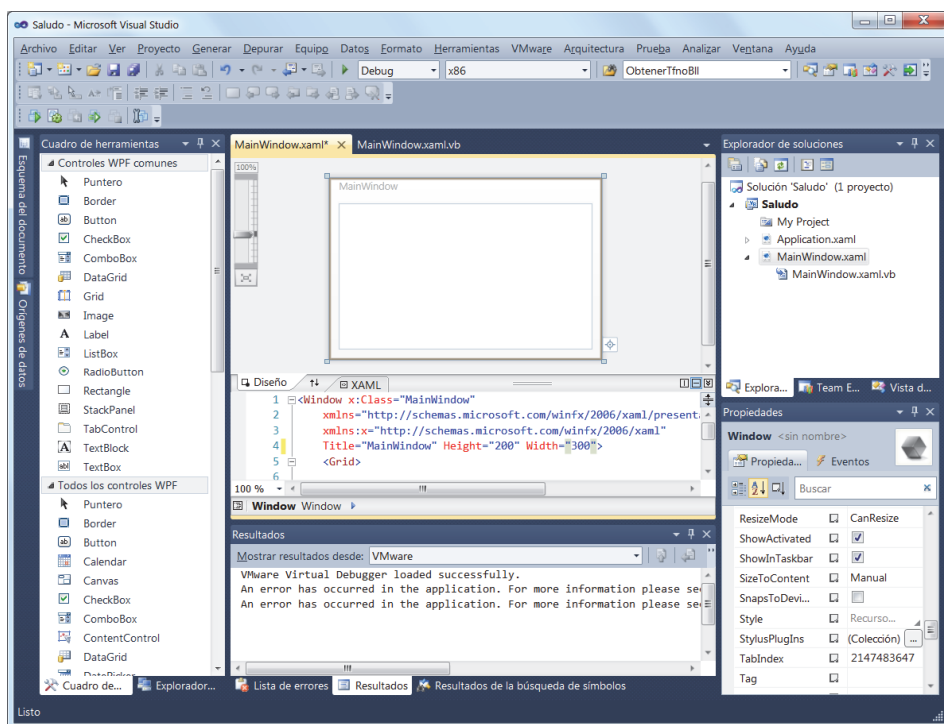
Crear un nuevo proyecto

Para crear un nuevo proyecto, diríjase a la barra de menús y ejecute *Archivo > Nuevo Proyecto*. En el diálogo que se visualiza, seleccione el tipo de proyecto *Visual Basic > Windows > Aplicación WPF* y asígnele el nombre *Saludo*. Observe, en la parte superior de la ventana, que puede elegir la versión de *.NET Framework*. Después, para continuar, haga clic en el botón *Aceptar*:



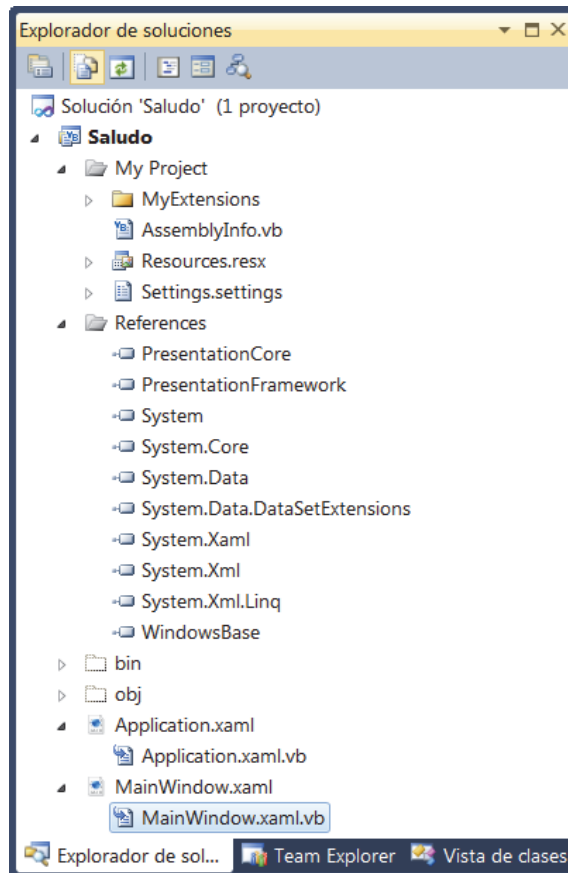
Obsérvese que se ha elegido la carpeta donde se almacenará el proyecto. Esta tarea puede posponerse sin que afecte al desarrollo de la aplicación. Ahora bien, si desea que esta tarea se realice automáticamente en el momento de crear el proyecto, caso del autor, ejecute *Herramientas > Opciones*, seleccione la opción *Proyectos y Soluciones* y marque la casilla *Guardar los proyectos nuevos al crearlos*.

Después de crear una nueva aplicación Windows, el entorno de desarrollo Visual Studio mostrará un formulario, *MainWindow*, en el diseñador. También pondrá a nuestra disposición una caja de herramientas con una gran cantidad de controles WPF listos para ser incluidos en la ventana.



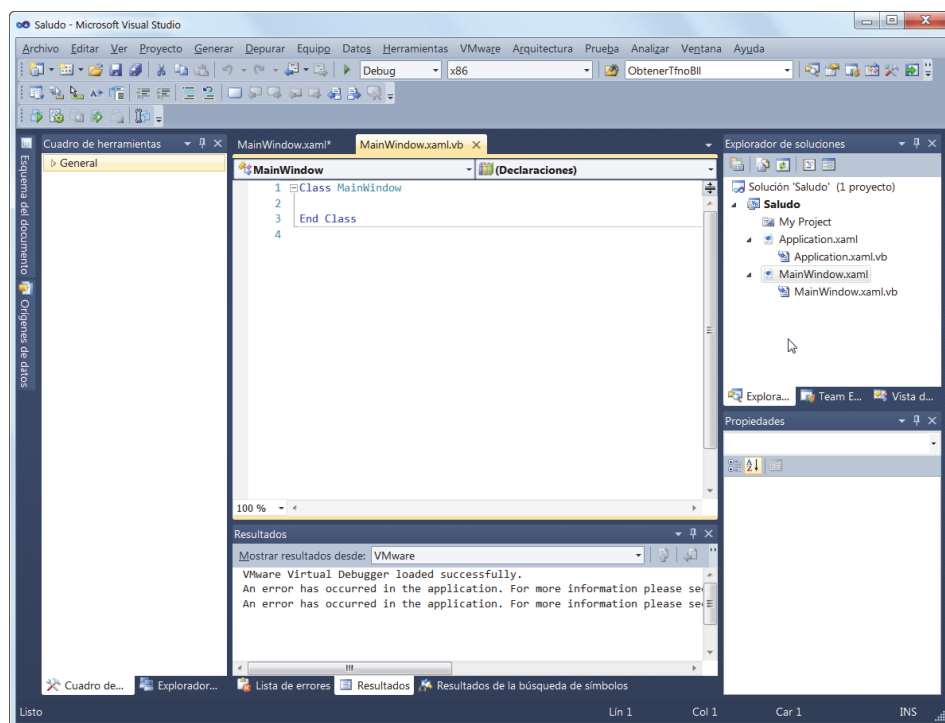
Otra característica interesante de este entorno de desarrollo es la ayuda que facilita. Basta con colocar el punto de inserción sobre una palabra clave y pulsar la tecla *F1* para que le muestre la ayuda correspondiente a la palabra seleccionada.

En la esquina superior derecha también se localiza otra ventana con varias páginas: explorador de soluciones, vista de clases, etc.; en la figura siguiente vemos el *Explorador de soluciones*:



El *Explorador de soluciones* muestra el nombre de la solución (una solución engloba uno o más proyectos), el nombre del proyecto (un proyecto administra los ficheros que componen la aplicación) y el de todos los formularios y demás módulos que intervienen en la aplicación; en nuestro caso, observamos un formulario, denominado *MainWindow*, descrito por los ficheros de código *MainWindow.xaml* y *MainWindow.xaml.vb*; el primero es el utilizado por el diseñador de formularios y el segundo, el utilizado por el programador para escribir el código. También se observa un nodo *References* que agrupa las referencias a las bibliotecas de clases de objetos que utilizará la aplicación en curso; podemos añadir nuevas referencias a otras bibliotecas haciendo clic con el botón secundario del ratón sobre ese nodo o bien eliminarlas.

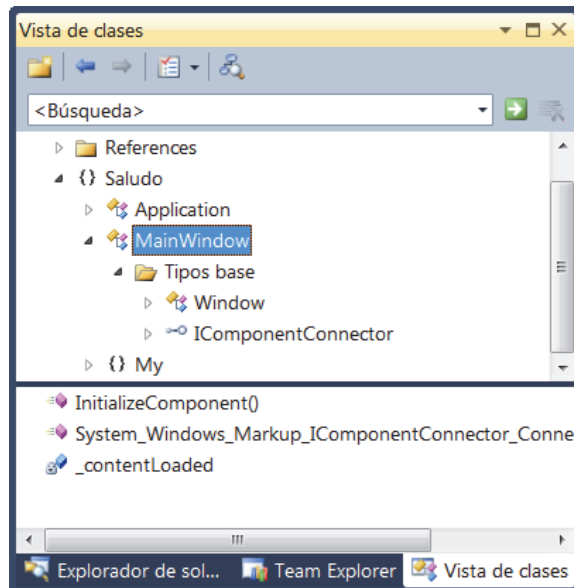
Así mismo, en su parte superior, muestra una barra de botones que permiten ver el *código*, el *diseñador* de formularios, la ventana de *propiedades*, etc. Por ejemplo, si estamos viendo el diseñador de formularios y hacemos clic en el botón *Ver código*, la página de diseño será sustituida por el editor de código, como se puede observar en la figura siguiente:



Una característica digna de resaltar del editor de Visual Studio es la incorporación de bloques de código contraíbles. En la figura superior podemos ver uno de estos bloques; si hacemos clic en un nodo $-$, contraeremos el bloque y ese nodo se convertirá en otro $+$ que permitirá expandir de nuevo el bloque.

Otra característica del editor es la finalización y el formato de código automáticos. Por ejemplo, al escribir un método, el editor mostrará automáticamente la ayuda en línea de la palabra clave (**Public**, **Sub**, **Integer**, etc.) que intenta escribir; si escribimos una sentencia **If**, exactamente igual. Puede personalizar las características del editor ejecutando *Herramientas > Opciones > Editor de texto*.

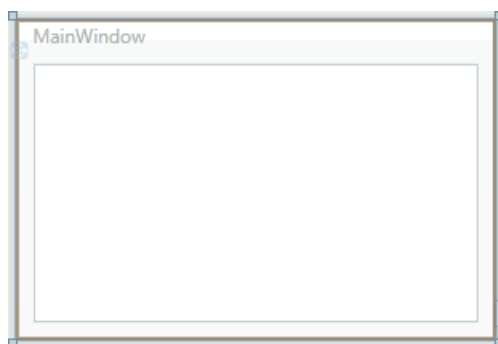
Si cuando se está visualizando el explorador de soluciones desea mostrar la vista de clases de su aplicación, sólo tiene que hacer clic en la pestaña *Vista de clases*. Esta ventana en su parte superior muestra las clases que componen la aplicación y en su parte inferior los métodos pertenecientes a la clase seleccionada.




Expandiendo el nodo del proyecto, vemos, en primer lugar, el espacio de nombres al que pertenecen las clases que definen la aplicación: *Saludo* (un espacio de nombres define un ámbito). Si ahora expandimos este otro nodo, veremos que incluye las clases mencionadas, la que define el objeto aplicación, *Application*, y la que define la ventana principal, *MainWindow*, y si expandimos a su vez este nodo, podremos observar su clase base, **Window**. Si seleccionó el nodo *MainWindow*, en el panel inferior de la figura podemos observar los métodos de la clase como, por ejemplo, el método *InitializeComponent*, entre otros.

El formulario

El formulario, objeto de la clase **Window** (una ventana), es el plano de fondo para los controles. Después de crear un nuevo proyecto, la página de diseño muestra uno como el de la figura siguiente. Lo que ve en la figura es el aspecto gráfico de un objeto de la clase *MainWindow*. Para modificar su tamaño ponga el cursor del ratón sobre alguno de los lados del cuadrado que lo rodea y arrastre en el sentido deseado.



Si ahora ejecutamos esta aplicación, para lo cual podemos pulsar las teclas *Ctrl+F5*, o bien elegir la orden correspondiente del menú *Depurar*, aparecerá sobre la pantalla el formulario (un objeto ventana), con el tamaño asignado, y podremos actuar sobre cualquiera de sus controles, o bien sobre las órdenes del menú de control, para minimizarlo, maximizarlo, moverlo, ajustar su tamaño, etc. Ésta es la parte que el diseñador de Visual Studio realiza por nosotros y para nosotros; pruébelo. Finalmente, para cerrar la ejecución de la aplicación disponemos de varias posibilidades:

1. Hacer clic en el botón  que cierra la ventana.
2. Hacer un doble clic en el icono situado a la izquierda en la barra de título de la ventana.
3. Activar el menú de control de la ventana *MainWindow* y ejecutar *Cerrar*.
4. Pulsar las teclas *Alt+F4*.

Dibujar los controles

En Visual Studio disponemos fundamentalmente de dos tipos de objetos: *ventanas* y *controles*. Las ventanas son los objetos sobre los que se dibujan los controles como cajas de texto, botones o etiquetas, dando lugar a la interfaz gráfica que el usuario tiene que utilizar para comunicarse con la aplicación y que genéricamente denominamos formulario.

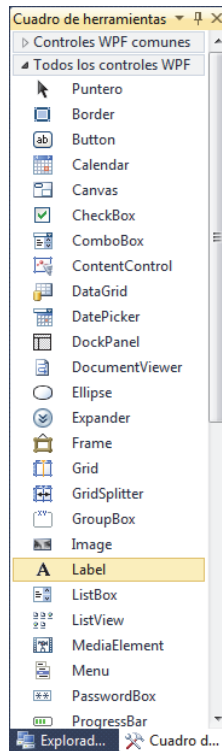
Para añadir un control a un formulario, utilizaremos la caja de herramientas que se muestra en la figura siguiente. Cada herramienta de la caja crea un único control. El significado de algunos de los controles más comunes se expone a continuación.

Puntero. El puntero no es un control. Se utiliza para seleccionar, mover y ajustar el tamaño de los objetos.

Label. Una *etiqueta* permite mostrar un texto de una o más líneas que no puede ser modificado por el usuario. Son útiles para dar instrucciones al usuario.

Button. Un *botón de pulsación* normalmente tendrá asociada una orden con él. Esta orden se ejecutará cuando el usuario haga clic sobre el botón.

TextBox. Una *caja de texto* es un área dentro del formulario en la que el usuario puede escribir o visualizar texto.



Menu. Permite añadir un menú de opciones para que el usuario seleccione una.

CheckBox. Una *casilla de verificación* se utiliza para seleccionar una opción. Utilizando estos controles se pueden elegir varias opciones de un grupo.

RadioButton. El control *botón de opción* se utiliza para seleccionar una opción entre varias. Utilizando estos controles se puede elegir una sola opción de un grupo de ellas.

GroupBox. Un *marco* se utiliza para realzar el aspecto del formulario. También los utilizamos para formar grupos de botones de opción, o bien para agrupar controles relacionados entre sí.

Grid. Control que actúa como contenedor de otros controles.

DataGrid. Proporciona una tabla para visualizar los datos de un origen de datos de una forma personalizada.

ListBox. El control *lista fija* (lista desplegada) contiene una lista de elementos de la que el usuario puede seleccionar uno o varios.

ComboBox. El control *lista desplegable* combina una caja de texto y una lista desplegable. Permite al usuario escribir lo que desea seleccionar o elegir un elemento de la lista.

ListView. El control *vista de lista* muestra una colección de elementos que se pueden visualizar mediante una de varias vistas distintas.

TreeView. Representa un control que muestra datos jerárquicos en una estructura de árbol con nodos que se pueden expandir y contraer.

TabControl. Es un control que agrupa un conjunto relacionado de fichas.

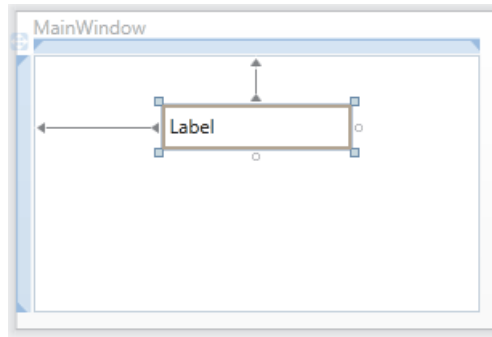
DatePicker. Control que permite seleccionar la fecha y hora.

ScrollBar. Representa una *barra de desplazamiento horizontal* o *vertical* que permite seleccionar un valor dentro de un rango de valores. Estos controles son utilizados independientemente de otros objetos, y no son lo mismo que las barras de desplazamiento de una ventana.

Otros controles de interés son la barra de progreso (*ProgressBar*), la caja de texto enriquecido (*RichTextBox*), la barra de estado (*StatusBar*), etc.

Observe que el formulario *MainWindow* ya contiene un **Grid** que actuará como contenedor de los controles que arrastremos sobre el mismo.

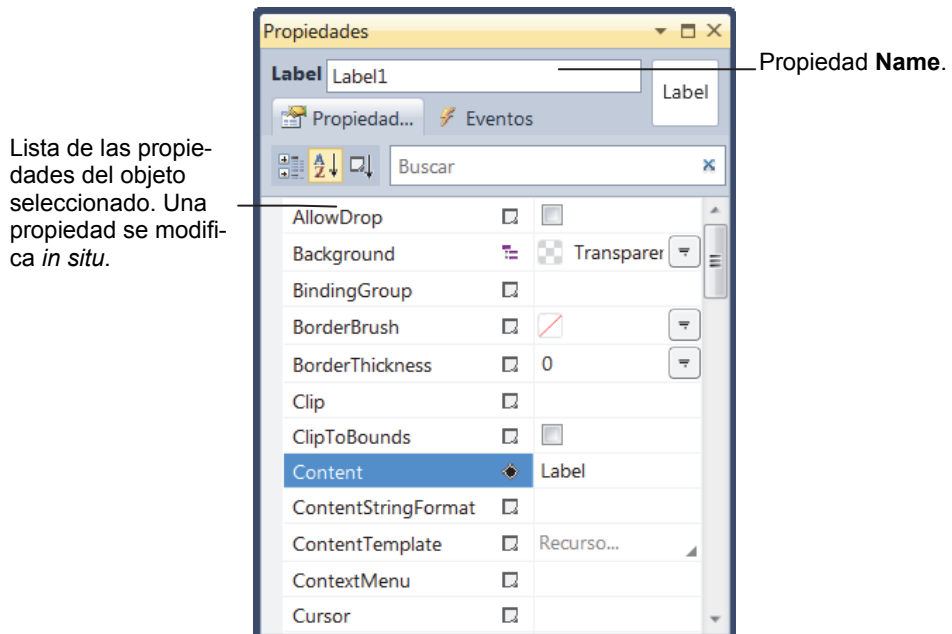
Siguiendo con nuestra aplicación, seleccionamos de la caja de herramientas que acabamos de describir los controles que vamos a utilizar. En primer lugar vamos a añadir al formulario una etiqueta. Para ello, hacemos clic sobre la herramienta etiqueta (*Label*) y, sin soltar el botón del ratón, la arrastramos sobre el formulario. Cuando soltemos el botón del ratón aparecerá una etiqueta de un tamaño predefinido, según se muestra en la figura siguiente:



Observe en la página de propiedades del entorno de desarrollo, mostrada en la figura siguiente, las propiedades **Name**, nombre, y **Content**, contenido. La primera tiene asignado el valor *label1* que es el nombre por defecto dado al control *Label*, y la segunda tiene asignado por defecto el valor “Label”, contenido que mostrará la etiqueta.

Si la ventana propiedades no está visible, ejecute la orden *Ventana de propiedades* en el menú *Ver* o pulse *F4*.

El nombre de un control se utiliza para referirnos a dicho control en el código de la aplicación.



Estando la etiqueta seleccionada se observa sobre la misma un rectángulo con unos cuadrados distribuidos a lo largo de su perímetro, que reciben el nombre de *modificadores de tamaño*, indicando que se puede modificar el tamaño del control que estamos dibujando. Para ello, primero selecciónelo haciendo clic sobre él, después apunte con el ratón a alguno de los lados del rectángulo que lo envuelve, observe que aparece una doble flecha, y, entonces, con el botón izquierdo del ratón pulsado, arrastre en el sentido que desee ajustar el tamaño.

También puede mover el control a un lugar deseado dentro del formulario. Para mover un control, primero selecciónelo haciendo clic sobre él y después apunte con el ratón a alguna zona perteneciente al mismo y, con el botón izquierdo del ratón pulsado, arrastre hasta situarlo en el lugar deseado.

Después de haber arrastrado un control sobre el formulario, Visual Studio crea automáticamente el código XAML que hará que el control se muestre cuando se ejecute el programa. De manera predeterminada, el editor XAML aparece bajo el diseñador. En la parte superior de este editor hay botones que le permitirán ver el marcado XAML en modo pantalla completa, en modo pantalla compartida horizontalmente o verticalmente, en la parte superior, etc., y en la parte inferior izquierda hay otro botón que muestra el esquema del documento que le permitirá navegar por el árbol de elementos de la ventana.



Borrar un control

Para borrar un control, primero se selecciona haciendo clic sobre él y, a continuación, se pulsa la tecla *Supr* (*Del*). Para borrar dos o más controles, primero se seleccionan haciendo clic sobre cada uno de ellos, al mismo tiempo que se mantiene pulsada la tecla *Ctrl*, y después se pulsa *Supr*.

Se pueden seleccionar también dos o más controles contiguos, pulsando el botón izquierdo y arrastrando el ratón hasta rodearlos.

Propiedades de los objetos

Cada clase de objeto tiene predefinido un conjunto de propiedades, como nombre, tamaño, color, etc. Las propiedades de un objeto representan todos los atributos que por definición están asociados con ese objeto.

Cuando se selecciona más de un objeto, la página de propiedades visualiza las propiedades comunes a esos objetos.

Cada propiedad de un objeto tiene un valor por defecto que puede ser modificado *in situ* si se desea. Por ejemplo, la propiedad **Title** del formulario del ejemplo que nos ocupa tiene el valor *MainWindow*.

Para cambiar el valor de una propiedad de un objeto, siga los pasos indicados a continuación:

1. Seleccione el objeto. Para ello, haga clic sobre el objeto o pulse sucesivamente la tecla *Tab* hasta que esté seleccionado (el control seleccionado aparecerá rodeado por un rectángulo modificador de tamaño).
2. Seleccione en la lista de propiedades la propiedad que desea cambiar.
3. Modifique el valor que actualmente tiene la propiedad seleccionada. El valor actual de la propiedad en cuestión aparece escrito a continuación del nombre de la misma. Para cambiar este valor, sobrescriba el valor actual o, si es posible, seleccione uno de la lista que se despliega haciendo clic sobre la misma. Para algunas propiedades, esta lista es sustituida por una caja de diálogo.

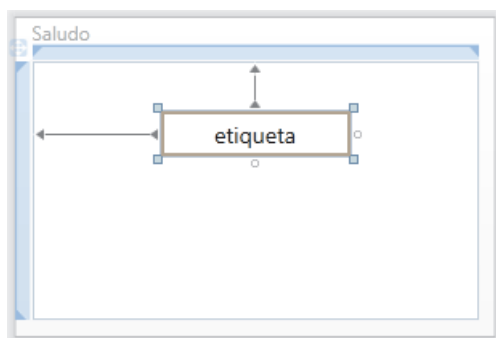
Se puede también modificar una propiedad durante la ejecución de la aplicación. Esto implica añadir el código necesario en el método que deba realizar la modificación.

Para verificar el valor de una misma propiedad en varios objetos, se selecciona ésta en la página de propiedades para uno de ellos, y a continuación se pasa de un objeto al siguiente haciendo clic con el ratón sobre cada uno de ellos, o simplemente pulsando la tecla *Tab*.

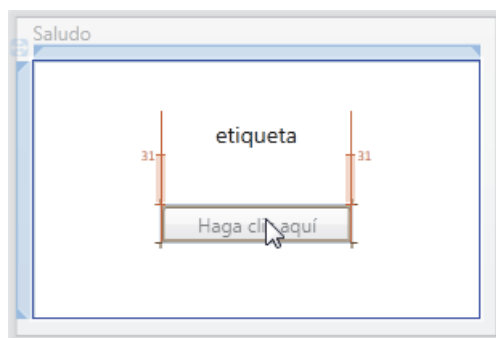
Siguiendo con nuestro ejemplo, vamos a cambiar el título del formulario. Para ello, seleccione el formulario y a continuación la propiedad **Title** en la página de propiedades. Después, sobrescriba el texto “*MainWindow*” con el texto “*Saludo*”.

Veamos ahora las propiedades de la etiqueta. Seleccione la etiqueta y observe la lista de propiedades. Algunas de estas propiedades son **Background** (color del fondo de la etiqueta), **Name** (identificador de la etiqueta para referirnos a ella en

el código) y **Content** (contenido de la etiqueta). Siguiendo los pasos descritos anteriormente, cambie el valor actual de la propiedad **Name** al valor *etSaludo*, el contenido “Label” de la propiedad **Content** a “etiqueta” y alinee este texto para que se muestre centrado tanto horizontal como verticalmente; esto requiere asignar a las propiedades **HorizontalContentAlignment** y **VerticalContentAlignment** el valor *Center*. A continuación, vamos a modificar el tamaño de la letra de la etiqueta; para ello, seleccione la propiedad **FontSize** en la página de propiedades, despliegue la lista de la derecha y elija como tamaño, por ejemplo, 14. El resto de las propiedades las dejamos como están.



El paso siguiente será añadir un botón. Para ello, hacemos clic sobre la herramienta *Button* de la caja de herramientas y arrastramos el botón sobre el formulario. Modificamos sus propiedades y asignamos a **Content** el valor *Haga clic aquí*, y a **Name**, el valor *btSaludo*. Después, movemos el botón y ajustamos su tamaño para conseguir el diseño que observamos en la figura siguiente.



También observamos que al colocar el control aparecen unas líneas indicando la alineación de éste con respecto a otros controles. Es una ayuda para alinear los controles que coloquemos dentro del formulario. Puede elegir entre los modos *SnapLines* (líneas de ayuda), es el modo que estamos utilizando, o *SnapToGrid* (rejilla de ayuda; se visualizan los puntos que dibujan la rejilla). Para elegir el modo de ayuda, ejecute *Herramientas > Opciones*, seleccione la opción *Diseña-*

dor de formularios Windows y asigne a la propiedad **LayoutMode** el modo deseado. Para que las opciones elegidas tengan efecto, tiene que cerrar el diseñador y volverlo a abrir.

Icono de la aplicación

Todos los formularios visualizan un icono en la esquina superior izquierda que generalmente ilustra la finalidad de la aplicación y que también aparece cuando se minimiza el formulario. Por omisión, Visual Studio utiliza un icono genérico.

Para utilizar su propio icono (de 16×16 o de 32×32 píxeles), sólo tiene que asignarlo a la propiedad **Icon** del formulario; esto es, seleccione el formulario, vaya a la página de propiedades, elija la propiedad **Icon**, pulse el botón que se muestra a la derecha y asigne el fichero *.ico* que contiene el icono.

Escribir los controladores de eventos

Sabemos que el nombre de un objeto, propiedad **Name**, nos permite referirnos a él dentro del código de la aplicación; por ejemplo, en las líneas de código siguiente, la primera asigna el valor “¡¡¡Hola mundo!!!” a la propiedad **Content** del objeto *etSaludo* y la siguiente obtiene el valor de la etiqueta y lo almacena en la variable *sTexto*:

```
etSaludo.Content = "¡¡¡Hola mundo!!!"
Dim sTexto As String = etSaludo.Content.ToString()
```

En Visual Basic la forma general de referirse a una propiedad de un determinado objeto es:

Objeto.Propiedad

donde *Objeto* es el nombre del formulario o control y *Propiedad* es el nombre de la propiedad del objeto cuyo valor queremos asignar u obtener.

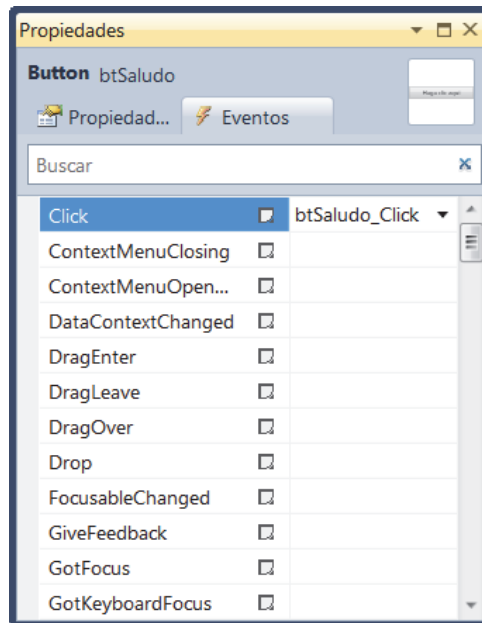
Una vez que hemos creado la interfaz o medio de comunicación entre la aplicación y el usuario, tenemos que escribir los métodos para controlar, de cada uno de los objetos, aquellos eventos que necesitemos manipular.

Hemos dicho que una aplicación en Windows es conducida por *eventos y orientada a objetos*. Esto es, cuando sobre un objeto ocurre un suceso (por ejemplo, el usuario hizo clic sobre un botón) se produce un evento (por ejemplo, el evento **Click**); si nosotros deseamos que nuestra aplicación responda a ese evento, tendremos que escribir un método que incluya el código que debe ejecutarse y vincularlo con el objeto que genera el evento. El método pertenecerá a la interfaz

del objeto o del objeto padre. Por ejemplo, el método que responda al evento **Click** de un botón pertenecerá a la interfaz de su ventana padre, esto es, a su contenedor.

¿Dónde podemos ver la lista de los eventos a los que puede responder un objeto de nuestra aplicación? En la ventana de propiedades.

Por ejemplo, seleccione el botón *btSaludo* en la ventana de diseño, vaya a la ventana de propiedades y muestre la lista de eventos para el control seleccionado, haciendo clic en el botón *Eventos*. Haga doble clic en el evento **Click**, o bien escriba manualmente el nombre del controlador y pulse *Entrar*.



El resultado es que se añade a la clase *MainWindow* un manejador para este evento (archivo *MainWindow.xaml.vb*):

```
Private Sub btSaludo_Click(sender As System.Object,  
    e As System.Windows.RoutedEventArgs) Handles btSaludo.Click  
  
End Sub
```

Alternativamente, podríamos añadir este manejador sin especificar la cláusula **Handles** y especificando en el código XAML (archivo *MainWindow.xaml*) que este método, *btSaludo_Click*, manejará el evento **Click**, según muestra el código escrito a continuación:

```

<Window x:Class="MainWindow"
    xmlns="http://schemas.microsoft.com/.../presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Saludo" Height="200" Width="300">
    <Grid>
        <Label Name="etSaludo" Content="etiqueta" FontSize="14"
            Height="28" Margin="80,31,0,0" Width="118"
            HorizontalAlignment="Left" VerticalAlignment="Top"
            HorizontalContentAlignment="Center"
            VerticalContentAlignment="Center" />
        <Button Name="btSaludo" Content="Haga clic aquí"
            Height="23" Width="118" Margin="80,96,0,0"
            HorizontalAlignment="Left" VerticalAlignment="Top"
            Click="btSaludo_Click" />
    </Grid>
</Window>

```

y el método *btSaludo_Click* que responderá a ese evento **Click** cada vez que se genere (fichero *MainWindow.xaml.vb*):

```

Private Sub btSaludo_Click(sender As Object, e As RoutedEventArgs)
    ' Escriba aquí el código que tiene que ejecutarse para responder
    ' al evento Click que se genera al pulsar el botón
End Sub

```

El primer parámetro del método anterior hace referencia al objeto que generó el evento y el segundo contiene datos relacionados con el evento.

Una vez añadido el controlador para el evento **Click** del botón *btSaludo*, ¿cómo lo completamos? Lo que deseábamos era que la etiqueta mostrara el mensaje “¡¡¡Hola mundo!!!” cuando el usuario hiciera clic en el botón. Según esto, complete este controlador así:

```

private void btSaludo_Click(object sender, RoutedEventArgs e)
{
    etSaludo.Content = "¡¡¡Hola mundo!!!"
}

```

Para añadir el controlador anterior, también podríamos habernos dirigido a la página de diseño y haber hecho doble clic sobre el botón de pulsación.

Además del evento **Click**, hay otros eventos asociados con un botón de pulsación, según se puede observar en la figura anterior.

Un detalle de estilo a la hora de escribir el código. Observe que *Visual Studio* antepone a los nombres de las clases y otras estructuras de datos el nombre del espacio de nombres al que pertenecen (por ejemplo **System.Object** en lugar de es-

cribir solamente el nombre de la clase **Object**), aunque no sería necesario, porque las referencias añadidas en el explorador de soluciones, equivalentes a las sentencias **Imports** que se muestran a continuación, ya los especifican.

```
Imports System
Imports System.Xaml
Imports System.Windows.Controls
```

Análogamente a como las carpetas o directorios ayudan a organizar los ficheros en un disco duro, los espacios de nombres ayudan a organizar las clases en grupos para facilitar el acceso a las mismas y proporcionan una forma de crear tipos globales únicos, evitando conflictos en el caso de clases de igual nombre pero de distintos fabricantes, ya que se diferenciarán en su espacio de nombres.

Guardar la aplicación

Una vez finalizada la aplicación, se debe guardar en el disco para que pueda tener continuidad; por ejemplo, por si más tarde se quiere modificar. Esta operación puede ser que se realice automáticamente cuando se compila o se ejecuta la aplicación y si no, puede requerir guardar la aplicación en cualquier instante ejecutando la orden *Guardar todo* del menú *Archivo*.

Si desplegamos el menú *Archivo*, nos encontraremos, además de con la orden *Guardar todo*, con dos órdenes más: *Guardar nombre-fichero* y *Guardar nombre-fichero como...* La orden *Guardar nombre-fichero* guarda en el disco el fichero actualmente seleccionado y la orden *Guardar nombre-fichero como...* realiza la misma operación, y además nos permite cambiar el nombre, lo cual es útil cuando el fichero ya existe.

No es conveniente que utilice los nombres que Visual Basic asigna por defecto, porque pueden ser fácilmente sobrescritos al guardar aplicaciones posteriores.

Verificar la aplicación

Para ver cómo se ejecuta la aplicación y los resultados que produce, hay que seleccionar la orden *Iniciar sin depurar* del menú *Depurar* o pulsar *Ctrl+F5*.

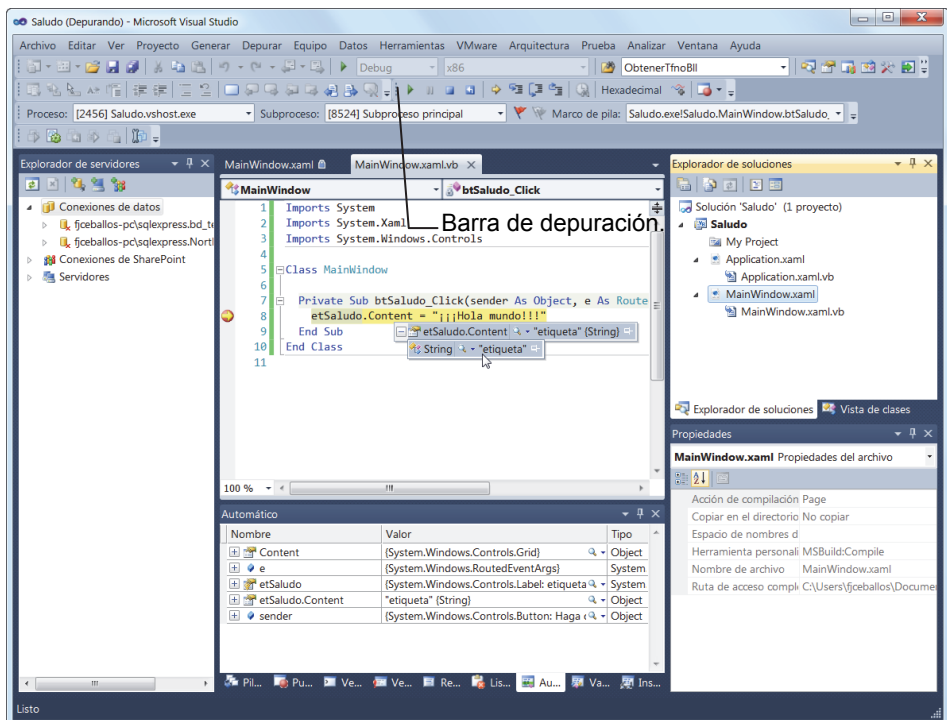
Si durante la ejecución encuentra problemas o la solución no es satisfactoria y no es capaz de solucionarlos por sus propios medios, puede utilizar, fundamentalmente, las órdenes *Paso a paso por instrucciones (F11)*, *Paso a paso por procedimientos (F10)*, *Alternar puntos de interrupción (F9)*, todas ellas del menú *Depurar*, para hacer un seguimiento paso a paso de la aplicación, y las opciones

del menú *Depurar* > *Ventanas* para observar los valores que van tomando las variables y expresiones de la aplicación.

La orden *Paso a paso por instrucciones* permite ejecutar cada sentencia de cada método de la aplicación paso a paso. Esta modalidad se activa y se continúa pulsando *F11*. Cuando una sentencia se corresponde con una llamada a un método y quiere que éste se ejecute en un solo paso, utilice la tecla *F10* (*Paso a paso por procedimientos*). Para detener la depuración pulse las teclas *Mayús+F5*.

La orden *Alternar un punto de interrupción* (*F9*) permite colocar una pausa en cualquier línea. Esto permite ejecutar la aplicación hasta la pausa en un solo paso (*F5*) y ver en la ventana *Automático* los valores que tienen las variables en ese instante. Para poner o quitar una pausa, se coloca el cursor donde se desea que tenga lugar dicha pausa y se pulsa *F9*, o bien se hace clic con el ratón sobre la barra situada a la izquierda del código.

Alternativamente al menú de depuración, puede utilizar la barra de herramientas de depuración. La figura siguiente muestra esta barra dentro de la ventana de Visual Studio en un proceso de depuración. La línea de código sombreada es la siguiente sentencia a ejecutar.



También puede utilizar el ratón para arrastrar el puntero de ejecución (observe la flecha en el margen izquierdo de la ventana anterior) a otro lugar dentro del mismo método con la intención de alterar el flujo normal de ejecución.

Durante el proceso de depuración, puede ver en la ventana *Automático* los valores de las variables y expresiones que desee. Además, en la ventana *Inspección* puede escribir la expresión cuyo resultado desea ver.

También, puede seleccionar en la ventana de código el objeto o propiedad cuyo valor quiere inspeccionar y ejecutar *Inspección rápida...* del menú *Depurar*. Una forma más rápida de hacer esto último es dirigirse al código y situar el puntero del ratón sobre el identificador del objeto o propiedad; le aparecerá una etiqueta con el valor, como se puede observar en la ventana de código anterior.

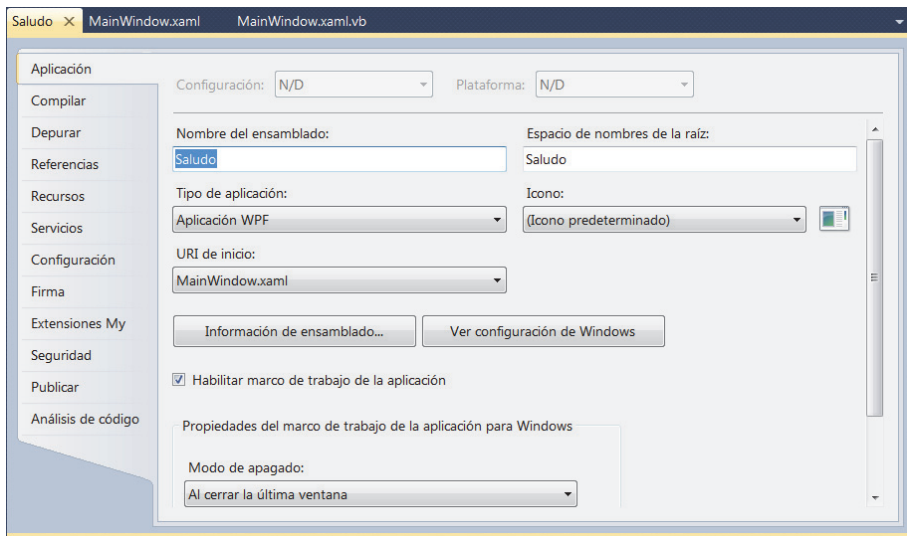
Así mismo, según se observa en la figura siguiente, puede ejecutar en la *Ventana Inmediato* cualquier sentencia de una forma inmediata. Para mostrar u ocultar esta ventana ejecute la orden *Ventanas > Inmediato* del menú *Depurar*. La figura siguiente muestra un ejemplo de cómo se puede inspeccionar el valor de la propiedad **Content** de la etiqueta *etSaludo* (observe el uso del símbolo ?).



Una vez iniciada la ejecución de la aplicación, si se pulsa la tecla *F5*, la ejecución continúa desde la última sentencia ejecutada en un método hasta finalizar ese método o hasta otro punto de parada.

Propiedades del proyecto

Cuando sea necesario establecer determinadas propiedades del proyecto actual o revisar las actuales hay que ejecutar la orden *Proyecto > Propiedades de nombre-proyecto...* Se le mostrará una ventana con varios paneles. Seleccione el deseado y modifique las propiedades que considere.



Crear soluciones de varios proyectos

Una solución agrupa uno o más proyectos. Por omisión, cuando se crea un nuevo proyecto, en la misma carpeta física se crea la solución (fichero con extensión *.sln*) a la que pertenece, con el mismo nombre que el proyecto. Esta solución permite que los ficheros que forman parte del proyecto se almacenen bajo una estructura de directorios que facilite su posterior localización así como las tareas de compartir la solución con otros desarrolladores de un supuesto equipo.

¿Qué tenemos que hacer si necesitamos agrupar varios proyectos bajo una misma solución? Crear una solución vacía y añadir nuevos proyectos a la solución o añadir nuevos proyectos a la solución existente. Asegúrese de que se va a mostrar siempre el nombre de la solución en el explorador de soluciones. Para ello, ejecute *Herramientas > Opciones > Proyectos y Soluciones > Mostrar siempre la solución*.

Para crear una nueva solución vacía:

1. Ejecute la orden *Archivos > Nuevo > Proyecto*.
2. Como tipo de proyecto, seleccione *Otros Tipos de Proyectos*.
3. Y como plantilla, seleccione *Solución en Blanco*.
4. Finalmente, introduzca el nombre que desea dar a la solución. Se creará un fichero *.sln* con el nombre dado, almacenado en una carpeta con el mismo

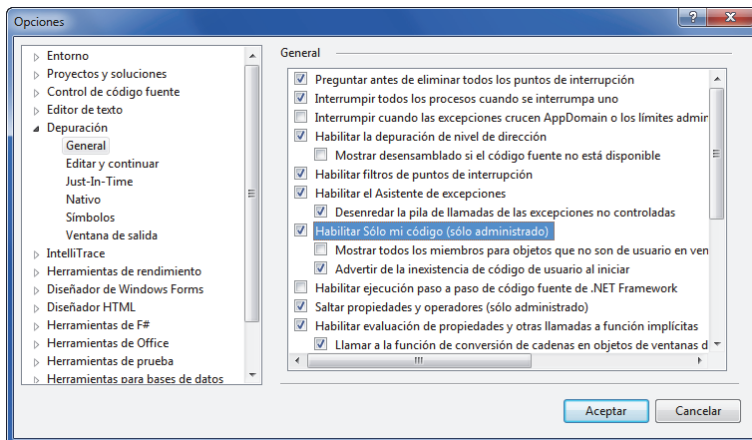
nombre. Puede elegir, si lo desea, la posición que ocupará esa carpeta en el sistema de ficheros de su plataforma.

Para añadir un nuevo proyecto a una solución existente:

1. Diríjase al explorador de soluciones y haga clic sobre el nombre de la solución utilizando el botón secundario del ratón. Del menú contextual que se visualiza, ejecute la orden *Añadir > Nuevo Proyecto...*
2. Seleccione el tipo de proyecto y la plantilla que va a utilizar para crearlo.
3. Para añadir nuevos proyectos repita los pasos anteriores.
4. Para activar el proyecto sobre el que va a trabajar, haga clic sobre el nombre del proyecto utilizando el botón secundario del ratón y del menú contextual que se visualiza, ejecute la orden *Establecer como proyecto de inicio*.

Opciones del EDI

Las opciones del entorno que se pueden establecer en el entorno de desarrollo integrado (EDI) son mostradas por la ventana *Opciones* que observa en la figura siguiente. Para mostrar esta ventana tiene que ejecutar la orden *Herramientas > Opciones...* Se puede observar que desde esta ventana es posible establecer opciones para el entorno de desarrollo, para los proyectos y soluciones, para el diseñador, para el depurador, etc. Por ejemplo, para que el depurador sólo navegue a través del código escrito por el usuario, no sobre el código añadido por los asistentes, tiene que estar activada la opción “habilitar sólo mi código”; *Herramientas > Opciones > Depuración > General > Habilitar sólo mi código*.



Personalizar el EDI

Para personalizar el entorno de desarrollo tiene que ejecutar la orden *Herramientas > Personalizar...* Desde esta ventana podrá añadir o quitar elementos de un menú, añadir o quitar una barra de herramientas, añadir o quitar un botón de una barra de herramientas, etc.

SQL SERVER EXPRESS

SQL Server Express es el motor de base de datos gratuito, potente, pero sencillo, que se integra perfectamente con el resto de productos Express. Se trata de una versión aligerada de la nueva generación de SQL Server.

Este producto tiene el mismo motor de base de datos que toda la familia SQL Server y utiliza el mismo lenguaje SQL, pero tiene ciertas limitaciones como por ejemplo que sólo puede usar una UCP y un máximo de 1GB de RAM, y las bases de datos no pueden sobrepasar los 4GB. Otra característica interesante es la movilidad de las bases de datos de un servidor a otro con **XCOPY**. Con esta utilidad podemos mover un fichero MDF de una máquina a otra a cualquier ubicación dentro de su sistema de ficheros, quedando la base de datos movida lista para trabajar. Para utilizar esta base de datos deberemos hacer uso de la opción **AttachDBFilename** en la cadena de conexión, según se muestra a continuación:

```
connectionString="Data Source=.\sqlexpress;Initial Catalog=;
Integrated Security=True;AttachDBFilename=C:\bd\bd_telefonos.mdf"
```

La entrada *Data Source* especifica el servidor de base de datos que vamos a utilizar y **AttachDBFilename**, la localización del fichero de base de datos. Obsérvese que la entrada *Initial Catalog* está vacía. O también, si la base de datos la copiamos en el directorio *App_Data* de una aplicación web, la cadena de conexión sería similar a esta otra:

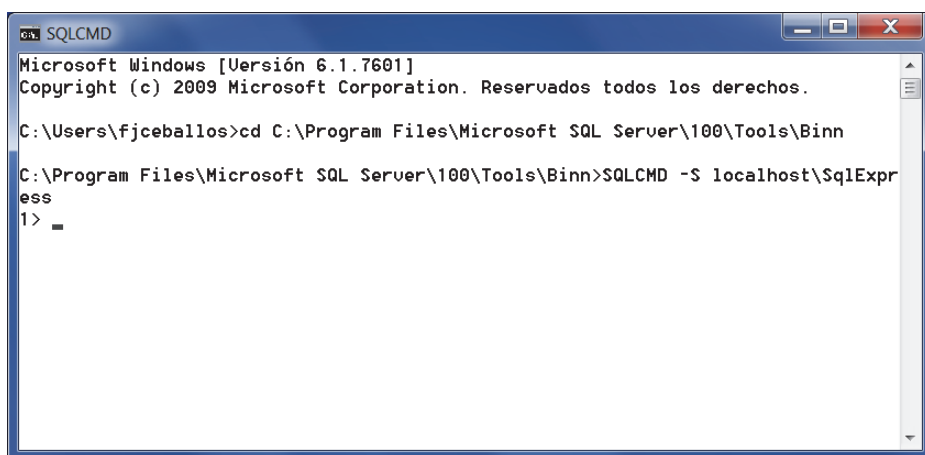
```
connectionString = "Data Source=.\sqlexpress;" +
"Integrated Security=True;" +
"AttachDBFilename=|DataDirectory|\bd_telefonos.mdf;" +
"User Instance=True";
```

El nombre del fichero comienza con *|DataDirectory|*. Esto automáticamente apunta a la carpeta *App_Data* del proyecto web. La entrada *User Instance* especifica que la base de datos es una instancia del usuario porque ésta no está registrada en el catálogo maestro del SQL Server. Esto es, cada base de datos que generamos mediante un *script*, por ejemplo, su nombre es registrado en el catálogo maestro, nombre que después será utilizado en la cadena de conexión por *Initial Catalog*, pero si copiamos una base de datos, ésta no es registrada. En este

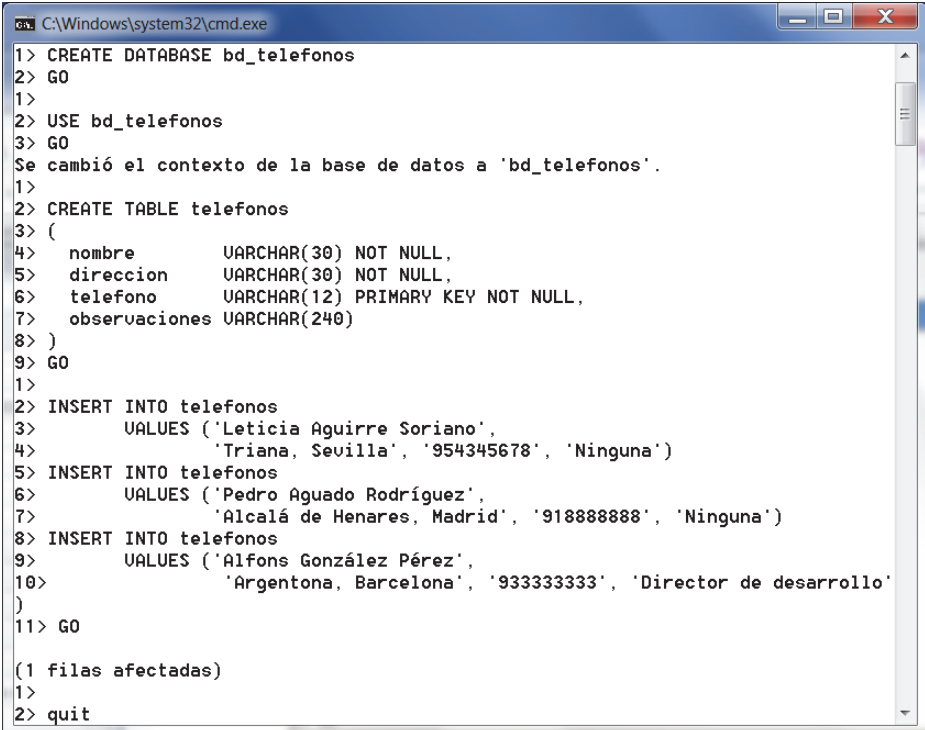
caso Sql Server Express permite acceder a ella tratándola como una instancia del usuario, característica que no está disponible en Sql Server.

Para crear una base de datos utilizando SQL Server Express tiene que hacerlo desde la línea de órdenes (véase también el capítulo titulado *Acceso a una base de datos*). Para iniciar la consola que le permita trabajar contra el motor de base de datos SQL Server, localice en su instalación el fichero SQLCMD.EXE, cambie a ese directorio y ejecute la orden:

```
SQLCMD -S nombre-del-ordenador\Sq1Express
```



Una vez iniciada la consola, puede escribir órdenes SQL a continuación del símbolo ">". Para ejecutar un bloque de sentencias escriba GO. Para salir, escriba QUIT. Por ejemplo, el guión que muestra la figura siguiente crea la base de datos *bd_telefonos* con una tabla *telefonos* y añade tres filas a la tabla:



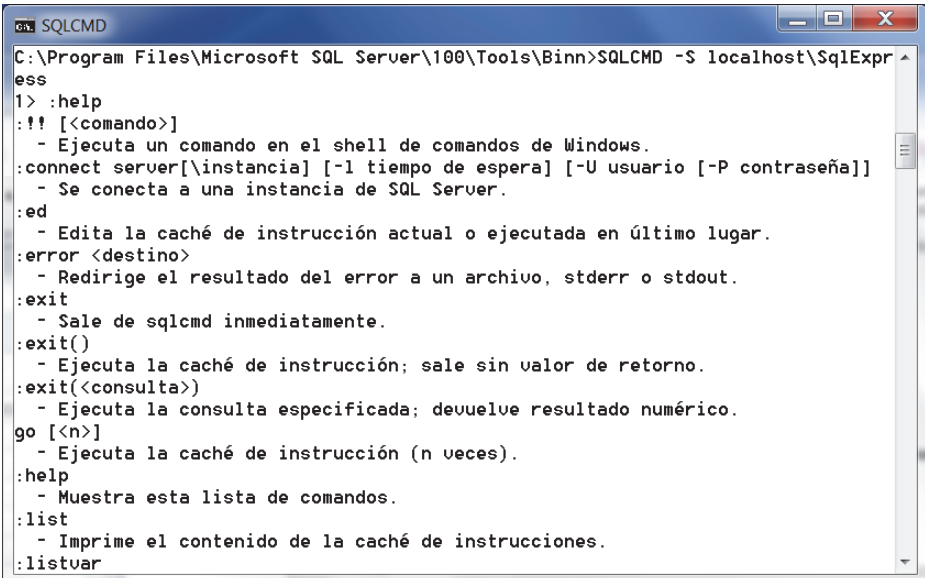
```

C:\Windows\system32\cmd.exe
1> CREATE DATABASE bd_telefonos
2> GO
1>
2> USE bd_telefonos
3> GO
Se cambió el contexto de la base de datos a 'bd_telefonos'.
1>
2> CREATE TABLE telefonos
3> (
4>     nombre          VARCHAR(30) NOT NULL,
5>     direccion       VARCHAR(30) NOT NULL,
6>     telefono        VARCHAR(12) PRIMARY KEY NOT NULL,
7>     observaciones   VARCHAR(240)
8> )
9> GO
1>
2> INSERT INTO telefonos
3>     VALUES ('Leticia Aguirre Soriano',
4>             'Triana, Sevilla', '954345678', 'Ninguna')
5> INSERT INTO telefonos
6>     VALUES ('Pedro Aguado Rodríguez',
7>             'Alcalá de Henares, Madrid', '918888888', 'Ninguna')
8> INSERT INTO telefonos
9>     VALUES ('Alfons González Pérez',
10>             'Argentona, Barcelona', '933333333', 'Director de desarrollo')
11> GO

(1 filas afectadas)
1>
2> quit

```

Para ver la relación de órdenes que puede utilizar a través de la aplicación *SQLCMD* ejecute la orden **help** como se muestra en la figura siguiente. Obsérvese que cada orden va precedida por dos puntos (:).



```

C:\Program Files\Microsoft SQL Server\100\Tools\Binn>SQLCMD -S localhost\Sq1Expr
ess
1> :help
:!! [<comando>]
- Ejecuta un comando en el shell de comandos de Windows.
:connect server[<instancia>] [-l tiempo de espera] [-U usuario [-P contraseña]]
- Se conecta a una instancia de SQL Server.
:ed
- Edita la caché de instrucción actual o ejecutada en último lugar.
:error <destino>
- Redirige el resultado del error a un archivo, stderr o stdout.
:exit
- Sale de sqlcmd inmediatamente.
:exit()
- Ejecuta la caché de instrucción; sale sin valor de retorno.
:exit(<consulta>)
- Ejecuta la consulta especificada; devuelve resultado numérico.
go [<n>]
- Ejecuta la caché de instrucción (n veces).
:help
- Muestra esta lista de comandos.
:list
- Imprime el contenido de la caché de instrucciones.
:listvar

```

SQL SERVER MANAGEMENT STUDIO EXPRESS

Si instaló SQL Server, habrá comprobado que la única herramienta que proporciona al usuario es *SQL Computer Manager* que sirve para gestionar los servicios básicos de SQL Server y para configurar los protocolos de red. Por tal motivo, Microsoft también ha desarrollado una nueva aplicación para gestionar bases de datos que puede obtener de forma gratuita de Internet en la dirección especificada a continuación:

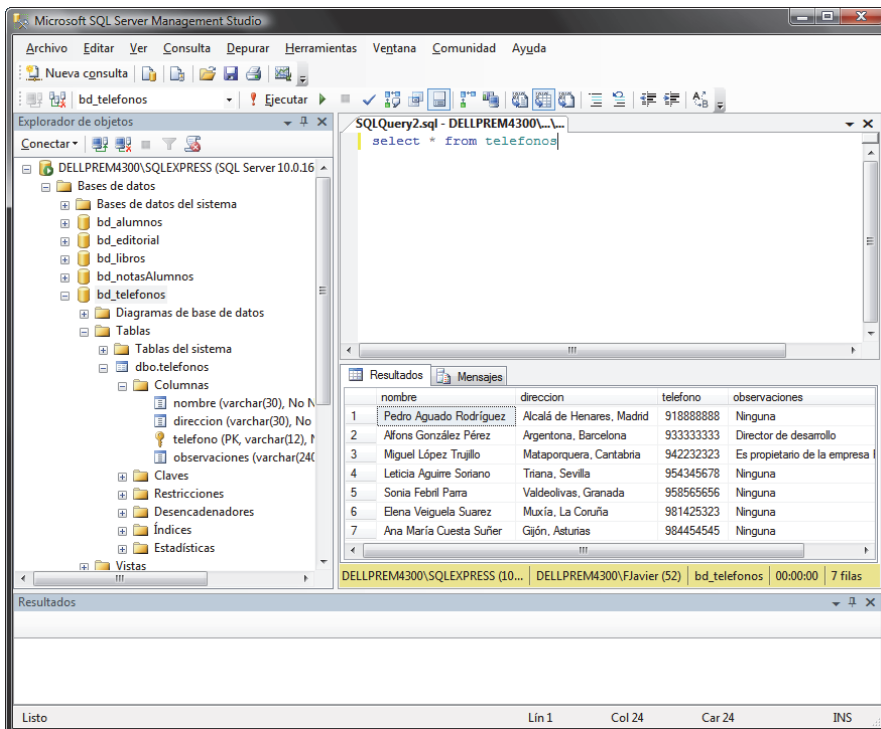
<http://www.microsoft.com/downloads/es-es/>

Esta aplicación presenta una interfaz gráfica, muy sencilla de utilizar, para realizar tareas típicas como crear bases de datos, gestionar las tablas de la base, los procedimientos almacenados, crear usuarios, etc.

Cuando inicie *SQL Server Management Studio Express*, le serán solicitados el nombre del servidor de bases de datos, el tipo de autenticación, y el usuario y la contraseña sólo si eligió autenticación SQL Server:



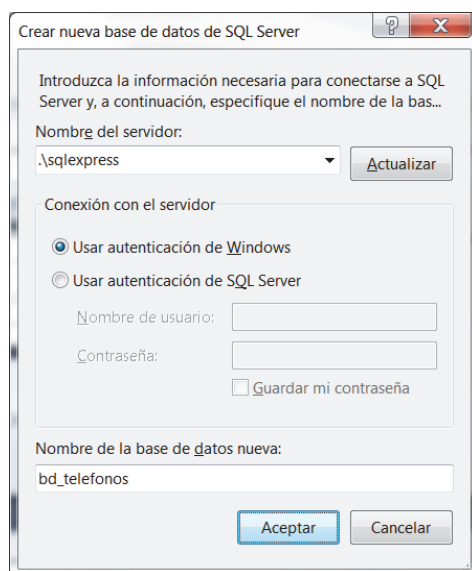
Una vez realizada la conexión con el gestor de bases de datos, le será mostrada la ventana de la figura siguiente. Seleccione en la lista del panel de la izquierda la base de datos con la que desea trabajar, haga clic en el botón *Nueva consulta* de la barra de herramientas y, después, escriba en el mismo las sentencias SQL que desee ejecutar. Para ejecutar una sentencia SQL haga clic en el botón *Ejecutar* de la barra de herramientas.



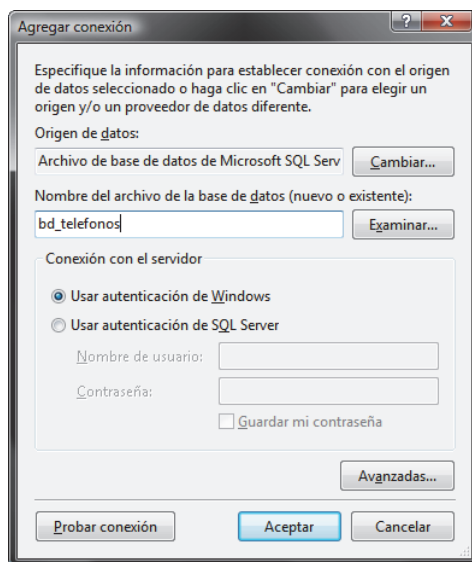
EXPLORADOR DE BASES DE DATOS

Otra forma de crear bases de datos es a través del explorador de bases de datos del EDI de Visual Studio. Si instaló Visual Studio o Visual Basic Express y SQL Server Express, entonces es posible añadir al proyecto elementos nuevos de tipo base de datos utilizando el explorador de bases de datos. Para ello, abra el *Explorador de bases de datos* (*Ver > Explorador de servidores* en Visual Studio o *Ver > Otras ventanas > Explorador de bases de datos* en Visual Basic Express). Haga clic con el botón secundario del ratón sobre el nodo *Conexiones de datos* y seleccione *Crear nueva base de datos SQL Server* (o *Nueva conexión* en Visual Basic Express).

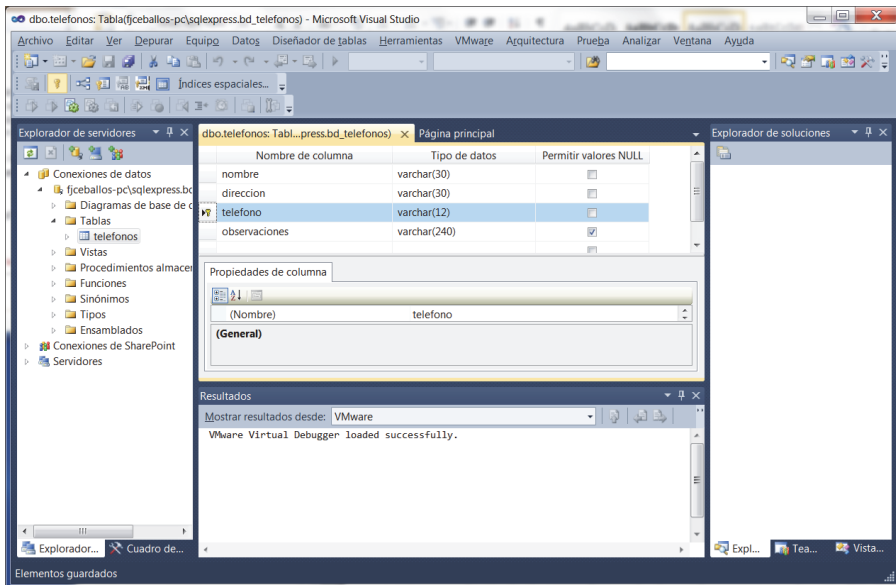
Si está utilizando Visual Studio, se visualizará la ventana siguiente. Elija como nombre del servidor `.\sqlexpress`, escriba el nombre de la base de datos y haga clic en *Aceptar*.



Si está utilizando Visual Basic Express, se visualizará esta otra ventana. Elija como origen de datos *Microsoft SQL Server*, escriba el nombre de la base de datos y haga clic en *Aceptar*.



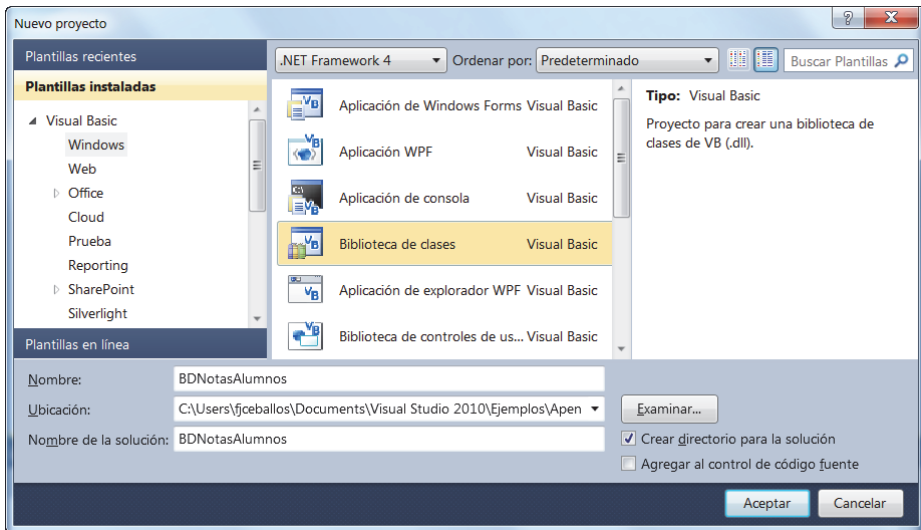
Una vez creada la base de datos, el paso siguiente es añadir las tablas. Despliegue el árbol correspondiente a la nueva conexión, haga clic con el botón secundario del ratón sobre el nodo *Tablas* y agregue una nueva tabla. Después complete el proceso de creación de la misma.



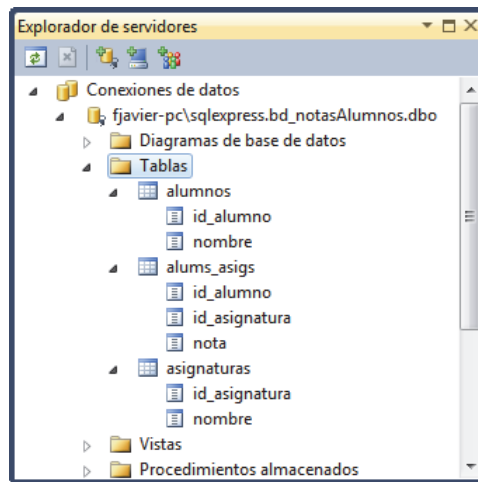
AÑADIR UN DATASET AL PROYECTO

Para acceder a una base de datos podemos escribir código basado en los objetos que proporciona ADO.NET, o bien podemos añadir un **DataSet** (conjunto de datos) al proyecto. Cuando se añade un **DataSet** al proyecto lo que sucede es que el asistente de diseño de Visual Studio crea una capa de acceso (DAL) a las tablas seleccionadas de la base de datos que nos abstrae de SQL. Dicha capa incluye todo un conjunto de clases con la funcionalidad suficiente para modificar, añadir o borrar filas en las tablas de la base de datos.

Como ejemplo, vamos a crear un proyecto *BDNotasAlumnos*, de tipo *Biblioteca de clases*, que nos proporcione una interfaz de clases Visual Basic para acceder a una base de datos formada por tres tablas, *alumnos*, *asignaturas* y *alums_asigs*, para gestionar las notas de las asignaturas en las que se ha matriculado cada uno de los alumnos (puede ver más detalles acerca de esta base de datos en el capítulo 6). Esta forma de proceder nos permitirá utilizar esta biblioteca en cualquier proyecto donde sea necesaria, simplemente añadiendo al proyecto en cuestión una referencia a la misma.



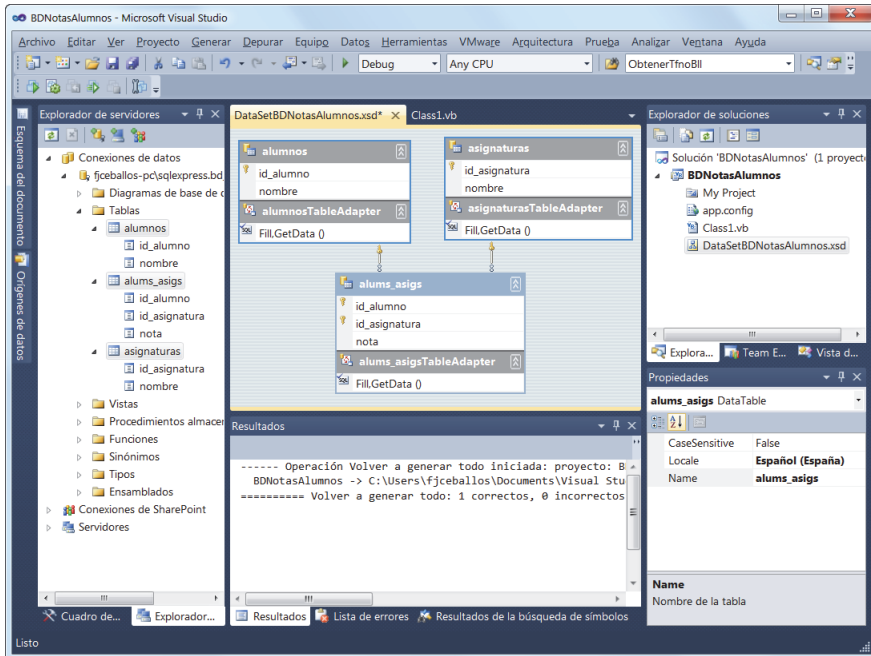
Suponiendo que tenemos creada la base de datos SQL, abrimos el explorador de servidores (explorador de base de datos) y añadimos una conexión a esta base de datos:



Para crear la capa de acceso a la base de datos, añadimos un **DataSet** al proyecto. Para ello, haga clic con el botón secundario del ratón sobre el nombre del proyecto y seleccione *Agregar > Nuevo Elemento > Conjunto de datos*, elija un nombre, por ejemplo *DataSetBDNotasAlumnos.xsd* y haga clic en *Agregar*. Observará en el *Explorador de soluciones* que se ha añadido un esquema XSD inicialmente vacío. También observará que la ventana de edición muestra el diseñador de **DataSet** para crear y editar visualmente conjuntos de datos con tipo.

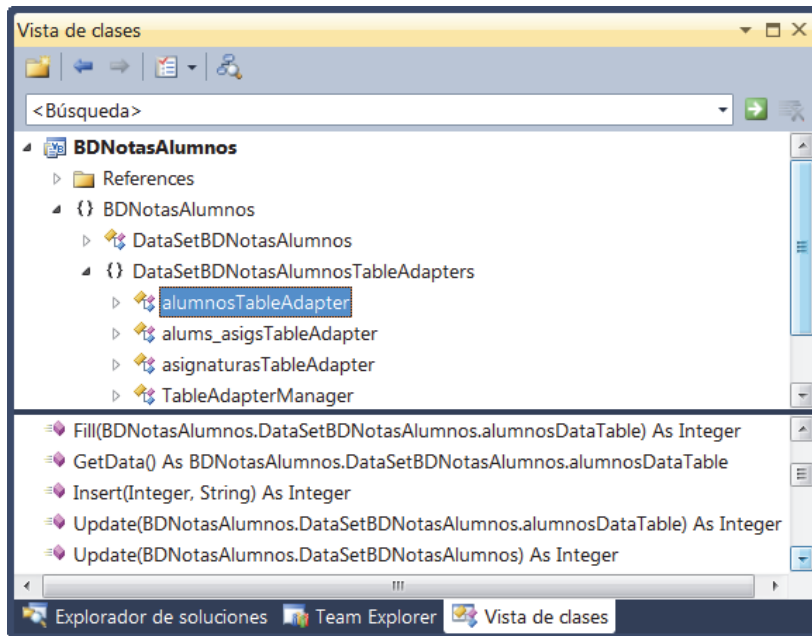
También puede realizar estas operaciones desde el menú *Datos* de Visual Studio.

Como siguiente paso, arrastre desde el *Explorador de servidores* los elementos de la base de datos con los que vaya a trabajar. En nuestro caso seleccionaremos las tres tablas.



También puede agregar nuevos elementos haciendo clic con el botón secundario del ratón sobre la superficie de diseño.

Si ahora muestra la vista de clases, podrá observar la capa de acceso a datos, formada por clases Visual Basic, que le permitirá realizar operaciones sobre los elementos de la base de datos que seleccionó y añadió al diseñador.



Esquemas XSD

Se pueden crear vistas XML de datos relacionales utilizando el lenguaje de definición de esquemas XML (XSD). Estas vistas pueden consultarse después utilizando consultas XPath (*lenguaje de rutas XML*).

Un esquema XML describe la estructura de un documento XML y también describe las distintas restricciones en los datos del documento. Todas las declaraciones de elementos deben incluirse dentro del elemento `<xsd:schema>` o `<xs:schema>`:

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema id="DataSetBDNotasAlumnos"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">

</xs:schema>
```

Un esquema XSD con anotaciones que describan la base de datos y las operaciones sobre la misma, puede utilizarlo para consultar la base de datos y devolver los resultados en forma de documento XML.

Base de datos XML

Para realizar pruebas rápidas en diferentes proyectos puede ser interesante simular una determinada base de datos con el esquema XSD correspondiente a las tablas del conjunto de datos construido, con los datos de las distintas tablas especificados en XML y con el código Visual Basic personalizado necesario para construir, a partir de la información anterior, un **DataSet** que modele la base de datos. Después, la funcionalidad proporcionada por la clase **DataSet** nos permitirá acceder a la base de datos.

Como ejemplo, partiendo de la aplicación anterior, vamos a construir en primer lugar los ficheros XML que definan el **DataSet** y los datos almacenados en las distintas tablas que incluimos en el conjunto de datos. Una forma fácil de hacer esto es construir una nueva solución con un nuevo proyecto de tipo *Aplicación WPF*, denominado, por ejemplo, *EnlaceDeDatosXML*, que incluya una referencia a la biblioteca que creamos anteriormente (para mayor facilidad, añada el proyecto *BDNotasAlumnos* a esta solución), y ejecutar desde el mismo, para cada una de las tablas, un código análogo al siguiente:

```
Imports BDNotasAlumnos
...
Dim dt1 As New DataSetBDNotasAlumnos.alumnosDataTable()
Dim t1 As New
    DataSetBDNotasAlumnos.TableAdapters.alumnosTableAdapter()
t1.Fill(dt1)
dt1.WriteXmlSchema("xsd_alumnos.xml")
dt1.WriteXml("datos_alumnos.xml")
```

Una vez obtenido el esquema XSD de cada una de las tablas, fusionamos todas ellas en un único fichero; por ejemplo en *bd_notasAlumnos.xsd*:

```
<?xml version="1.0" standalone="yes"?>
<xs:schema id="bd_notasAlumnos" xmlns=""
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">
  <xs:element name="bd_notasAlumnos"
    msdata:IsDataSet="true" msdata:UseCurrentLocale="true">
    <xs:complexType>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element name="alumnos">

          </xs:element>
        <xs:element name="asignaturas">

          </xs:element>
        <xs:element name="alums_asigs">
```

```
        </xs:element>
    </xs:choice>
</xs:complexType>

<xs:unique name="Constraint1" msdata:PrimaryKey="true">

</xs:unique>
<xs:unique name="Constraint2" msdata:PrimaryKey="true">

</xs:unique>
<xs:unique name="Constraint3" msdata:PrimaryKey="true">

</xs:unique>
</xs:element>
</xs:schema>
```

Hacemos lo mismo con los datos de cada una de las tablas; por ejemplo fusionamos los datos en *bd_notasAlumnos.xml*. Observe que la etiqueta que envuelve todo el documento, *bd_notasAlumnos*, coincide con la identificación del esquema XSD.

```
<?xml version="1.0" standalone="yes"?>
<bd_notasAlumnos>
  <alumnos>
    <id_alumno>1234567</id_alumno>
    <nombre>Leticia Aguirre Soriano</nombre>
  </alumnos>
  ...
  <asignaturas>
    <id_asignatura>20590</id_asignatura>
    <nombre>PROGRAMACION AVANZADA</nombre>
  </asignaturas>
  ...
  <alums_asigs>
    <id_alumno>1234569</id_alumno>
    <id_asignatura>33693</id_asignatura>
    <nota>9.5</nota>
  </alums_asigs>
</bd_notasAlumnos>
```

Ahora ya podemos eliminar de la biblioteca *BDNotasAlumnos* el **DataSet** que agregamos inicialmente y, en su lugar, añadimos a este proyecto estos dos ficheros que acabamos de crear: *bd_notasAlumnos.xsd* y *bd_notasAlumnos.xml*. En realidad no es necesario añadirlos; lo hacemos simplemente a efectos de modificaciones. Donde sí tienen que estar es en la carpeta desde la que se ejecute la aplicación WPF que haga referencia a esta biblioteca que hemos generado; en nuestro caso en *EnlaceDeDatosXML*.

Para construir un objeto **DataSet** a partir de los ficheros *bd_notasAlumnos.xsd* y *bd_notasAlumnos.xml*, añadimos a la biblioteca *BDNotasAlumnos* una nueva clase, *DataSetBDNotasAlumnos*, que almacenamos en el fichero *bd_notasAlumnosDataSet.cs*:

```
Public Class DataSetBDNotasAlumnos
    ' Construcción del DataSet a partir de
    ' bd_notasAlumnos.xsd y bd_notasAlumnos.xml
    Public Shared Function ObtenerTablaAsignaturas() As DataTable
        Return ReadDataSet().Tables("asignaturas")
    End Function

    Public Shared Function ObtenerAlumsAsigsNotas() As DataSet
        Return ReadDataSet()
    End Function

    Friend Shared Function ReadDataSet() As DataSet
        Try
            Dim ds As New DataSet()
            ds.ReadXmlSchema("../..bd_notasAlumnos.xsd")
            ds.ReadXml("../..bd_notasAlumnos.xml")
            Return ds
        Catch exc As Exception
            System.Diagnostics.Debug.WriteLine(exc.Message)
            Return Nothing
        End Try
    End Function
End Class
```

Y para implementar la lógica de negocio que permita acceder a la base de datos añadimos otra clase más, *bd_notasAlumnos*, que almacenamos en el fichero *bd_notasAlumnos.cs*:

```
Imports System.Collections.ObjectModel

Public Class bd_notasAlumnos
    Public Function ObtenerAsignatura(ID As Integer) As Asignatura
        Try
            Dim tablaAsigs As DataTable =
                DataSetBDNotasAlumnos.ObtenerTablaAsignaturas()
            Dim filaAsignaturas As DataRow =
                tablaAsigs.[Select]("id_asignatura = " & ID.ToString())(0)
            Dim asig As New Asignatura(
                CInt(filaAsignaturas("id_asignatura")),
                DirectCast(filaAsignaturas("nombre"), String))
            Return asig
        Catch exc As Exception
            System.Diagnostics.Debug.WriteLine(exc.Message)
            Return Nothing
        End Try
    End Function
End Class
```

```
End Try
End Function

Public Function ObtenerAsignaturas() As ICollection(Of Asignatura)
    Dim tablaAsigs As DataTable =
        DataSetBDNotasAlumnos.ObtenerTablaAsignaturas()

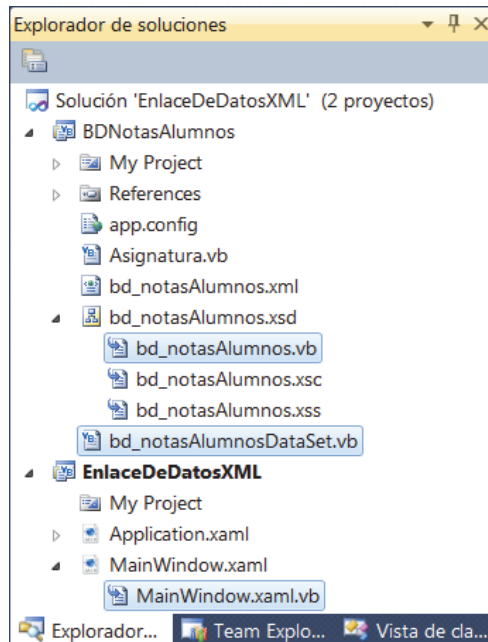
    Dim asigs As New ObservableCollection(Of Asignatura)()

    For Each fila As DataRow In tablaAsigs.Rows
        asigs.Add(New Asignatura(CInt(fila("id_asignatura")),
            DirectCast(fila("nombre"), String)))
    Next
    Return asigs
End Function

, ...
End Class
```

Esta clase requiere añadir también la clase *Asignatura* que implementaremos de la misma forma que lo hicimos en el capítulo *Enlace de datos en WPF*.

En la figura siguiente se pueden observar ambos proyectos, el correspondiente a la biblioteca, *BDNotasAlumnos*, y el correspondiente a la aplicación WPF, *EnlaceDeDatosXML*, que utilizará esa biblioteca.



VISUAL WEB DEVELOPER

Visual Web Developer Express (VWD) es una herramienta enfocada exclusivamente al desarrollo de aplicaciones webs dinámicas con ASP.NET. Para ello, proporciona:

- Diseñadores visuales para crear las páginas webs.
- Un editor de código potente que muestra referencias del lenguaje a medida que se escribe código, sin tener que dejar el editor. Con este editor podrá escribir código en los diferentes lenguajes .NET (Visual VB o C#) y código HTML.
- Desarrollo rápido de aplicaciones webs integradas con bases de datos. Para ello proporciona acceso integrado a SQL Server Express.

Crear un sitio web con VWD y probar su funcionamiento es algo sencillo porque integra un servidor web, por lo que no es necesario desplegar la aplicación en el servidor IIS de Windows (*Internet Information Server*). En realidad, cualquier carpeta con páginas .aspx puede ser considerada virtualmente como un sitio web. Este servidor web integrado será utilizado automáticamente cuando elijamos como tipo de sitio web “sistema de archivos”. Otros tipos de sitios webs son HTTP y FTP, ambos necesitan tener instalado IIS.

La forma de trabajar con esta herramienta es la misma que hemos explicado en los últimos capítulos de este libro para crear sitios y servicios webs con Visual Studio, ya que esta herramienta está integrada en Visual Studio.

INSTALACIÓN DE ASP.NET EN WINDOWS

ASP.NET queda instalado automáticamente cuando instaló Visual Studio o, en su defecto, Visual Web Developer. Ahora bien, para crear y ejecutar aplicaciones para Internet necesitará un servidor de aplicaciones, en el caso de la plataforma Windows éste es IIS (*Internet Information Server*). La instalación de IIS debe ser anterior a la de .NET Framework. Si no se hizo así, ASP.NET no estará habilitado en IIS y no podremos realizar aplicaciones ASP.NET. En este caso, la solución es registrar manualmente ASP.NET en IIS.

Registro manual de ASP.NET en IIS

Para registrar manualmente ASP.NET en IIS, abra una ventana de consola, sitúese en la carpeta `C:\WINDOWS\Microsoft.NET\Framework\vx.x.xxxx` y ejecute:

```
aspnet_regiis.exe -i -enable
```

Esto habilitará IIS para la ejecución de ASP.NET, registrará las extensiones de ASP.NET (*aspx*, *asmx*, *asax*, etc.) y ya podremos empezar a trabajar con ASP.NET.

Si echa una ojeada a todas las opciones disponibles (*aspnet_regiis.exe -help*), podrá observar algunas muy interesantes, como *-c*, que instala las secuencias de órdenes del cliente de esta versión en el subdirectorio *aspnet_client* de todos los directorios de sitios IIS.

Si la operación anterior no solucionó el problema, pruebe a reparar la instalación actual a partir del CD de instalación del producto. Si esto tampoco solucionara el problema, sólo queda desinstalar el producto y volverlo a instalar.

APÉNDICE B

© F.J.Ceballos/RA-MA

CD

La imagen del CD de este libro, con las aplicaciones desarrolladas y el software para reproducirlas, puede descargarlo desde <http://www.fjceballos.es> (sección *Mis publicaciones > Visual Basic > CD*) o desde <http://www.ra-ma.com> (en la página correspondiente al libro). La descarga consiste en un fichero ZIP con una contraseña ddd-dd-dddd-ddd-d que se corresponde con el ISBN de este libro localizado en la página IV del mismo (teclea los dígitos y los guiones).

Cuando descomprima el fichero obtendrá la imagen ISO del CD que puede grabar en un soporte convencional ejecutando la orden *Grabar imagen*, o equivalente, de su programa de grabación de CD/DVD.

ÍNDICE

A

- AcceptsReturn, 184
- AcceptsTab, 184
- acceso conectado a bases de datos, 408
- acceso desconectado a datos, 451
- Access, 400
- AccessText, 67
- aceleradores, 144
- Acerca de, 198, 223
- Activate, 42, 43
- Activated, 41, 49
- actualizar datos, 766
- actualizar la base de datos, 843
- actualizar las filas, 525
- adaptador, 402
- Add, colección, 205
- Added, 397
- AddedItems, 438
- AddHandler, 71, 73, 662
- AddNew, 768
- AddObject, 527
- administración de sitios webs, 798
- ADO.NET, 394
- AdornedElementPlaceholder, 124
- agregar datos, 767
- agrupación de los datos, 842
- agrupar los elementos de la vista, 353
- alineación del texto, 27
- AllowsTransparency, 44
- ámbito de foco, 149
- animación, 326
- animaciones, 647
- animar Opacity, 326
- animar una transformación, 650
- añadir imagen a un elemento de un menú, 175
- añadir nuevos objetos, 701
- añadir texto a una caja de texto, 80
- añadir un nuevo registro, 848
- aplicación
 - crear, 857
 - de negocios de Silverlight, 815
 - depurar, 872
 - desarrollo, 63
 - instanciar sólo una vez, 51
 - mínima, 6
- aplicaciones XBAP, 576
- app.config, 46
- AppendText, 188
- Application, 13, 41, 58
- Application.Current, 55
- ApplicationServices, 790
- árbol con nodos, 864
- árboles de expresiones, 481
- área cliente, 42
- área de trabajo, 3
- argumentos en la línea de órdenes, 54
- asíncronico, modelo, 724
- ASP.NET
 - acceso a los recursos, 746
 - arquitectura, 778
 - instalación, 891
- aspecto de un control, 122
- aspnet_regiis, 892
- Assembly, 49

- atajo, 155
- ataques de inyección SQL, 411
- atributos de anotación de datos, 621
- atributos globales de una aplicación, 48, 224
- Attach, 754
- AttachDBFilename, 877
- audio, 654
- autenticación, 777, 819
 - de Windows, 779, 816
 - mediante formularios, 779, 816
- autenticación y autorización, 675
- Authenticate, 787
- AuthenticationService, 790, 793, 800
- AutoCompleteBox, 631
- AutoPlay, 655
- autorización, 777, 808
- ayuda dinámica, 858

B

- BAML, 16
- barra de desplazamiento, 2, 251, 864
 - eventos, 254
- barra de estado, 181
- barra de herramientas, 177
 - añadir botón, 178
 - diseño, 191
 - orientación, 180
- barra de menús, 1, 139
 - crear, 139
 - diseño, 189
- barra de progreso, 256
- barra de tareas, botón de la ventana, 42
- barra de título, 2
- base de datos, 887
 - crear, 878, 881
 - mover, 877
 - XML, 821
- BasedOn, 120
- BeginEdit, 440
- BeginInvoke, 304
- BeginTime, 650
- binary, 746
- BinaryExpression, 484
- Binding, 21, 23, 97, 98, 316, 678
- BindingExpression, 114
- BindingGroup, 374, 449
- BindingListCollectionView, 320
- Bitmap, 695
- BitmapImage, 695

- Block, 285
- BlockUIContainer, 286
- Boolean?, 220
- Border, 26, 61, 324, 616
- borrar datos, 768
- borrar objetos, 705
- borrar registros en una tabla, 389
- borrar tabla de la base de datos, 389
- borrar un control, 866
- borrar un registro, 851
- botón de opción, 232, 863
- botón de pulsación, 28, 62, 863
- botón predeterminado, 67
- bucle de mensajes, 4, 304
- Button, 26, 60, 62
- ButtonBase, 60
- byte[], 697

C

- caché, 581
- cadena de conexión, 811
- caja de diálogo
 - Abrir, 278
 - Color, 281
 - crear, 217
 - Guardar, 280
 - Imprimir, 282
 - modal o no modal, 212
 - mostrar, 219
 - para mostrar un mensaje, 212
 - recuperar datos, 221
- caja de herramientas, 862
- caja de texto, 863
 - multilínea, 184
 - seleccionar el contenido, 79
- cajas de diálogo estándar, 278
- cajas de diálogo personalizadas, 215
- calculadora, 135
- Calendar, 261
- cambios en los datos, 522
- cambios, seguimiento, 535
- campos, 385
- CancelEdit, 440
- CanExecute, 151, 174
- CanExecuteChanged, 152, 173
- CanUndo, 188, 202
- Canvas, 29, 61
- capa de acceso a datos, 431
- capa de lógica de negocio, 435

- capa de presentación, 425
 - desacoplar, 439
 - lógica, 436
- CaptureSource, 664
- carpeta virtual, 578
- cascada, operaciones en, 531
- casilla de verificación, 227, 863
- CenterOwner, 217
- cerrar, 2
- cerrar la aplicación, 52
- CheckBox, 26, 60, 227, 245
- Checked, 228
- CheckStateChanged, 232
- Children, 296
- ChildWindow, 637
- ciclo de vida de una aplicación, 49
- ciclo de vida de una ventana, 41
- cifrado, texto, 217
- cifrar, 783
- clase
 - Application, 13
 - Binding, 98, 316
 - Border, 26
 - Button, 26
 - CheckBox, 26, 227
 - Clipboard, 187
 - ColorDialog, 281
 - ComboBox, 26, 246
 - ContentElement, 57
 - Control, 26
 - Convert, 236
 - DataRow, 397
 - DataTable, 397
 - de entidad, 509
 - de un objeto, determinar, 294
 - Debug, 144
 - Dispatcher, 304
 - FrameworkContentElement, 57
 - FrameworkElement, 27, 57
 - GroupBox, 232
 - Label, 26
 - ListBox, 26, 238
 - Menu, 139
 - MenuItem, 139
 - Mouse, 81
 - NavigationWindow, 549
 - ObjectContext, 510
 - OpenFileDialog, 279
 - Page, 551
 - ProgressBar, 256
 - RadioButton, 26, 232
 - RoutedCommand, 146
 - SaveFileDialog, 280
 - ScrollBar, 26, 251
 - Separator, 139
 - Slider, 255
 - TextBlock, 26
 - TextBox, 26, 188
 - TextBoxBase, 188
 - TextRange, 298
 - TimeSpan, 307
 - UIElement, 57
 - Validation, 109
 - Window, 9
- clases, vista, 860
- Clear, 188, 243
- Click, 4, 76
- ClientAccessPolicy.xml, 729
- cliente Silverlight, 756
- cliente WCF, 720
- Clipboard, 187
- Close, 43, 143, 220, 402
- Closed, 42
- Closing, 42
- clr-namespace, 18, 102
- código subyacente, 11
- código Visual Basic, insertar en XAML, 12
- colección
 - añadir, 243
 - borrar todo, 243
 - borrar, 243
 - ConstraintCollection, 397
 - DataColumnCollection, 397
 - DataRelationCollection, 397
 - DataRowCollection, 397
 - DataTableCollection, 397
 - de elementos, 205, 864
 - insertar, 243
- colecciones, 317
- CollectionChanged, 354, 444
- CollectionView, 319, 332
- CollectionViewSource, 319, 686
- colocar el control, 802
- color, 20, 281
- Color, propiedad, 282
- ColorAnimation, 648
- ColorDialog, 281
- Column, 33
- columnas, 385
- ColumnSpan, 33

- ComboBox, 26, 60, 246
 - ComboBoxItem, 60, 246
 - Command, 148, 398, 401
 - CommandBinding, 151
 - CommandParameter, 163
 - commands, 145
 - CommandTarget, 149
 - CommitNew, 768
 - compilador de marcado de WPF, 11
 - Completed, 839
 - componentes de acceso a datos, 424
 - comunicación entre dominios, 728
 - concurrency, 532, 744, 826
 - ConcurrencyMode, 532
 - conexión abierta/cerrada, 511
 - conexión con la base de datos, 404
 - conexión, probar, 405
 - configuración de la aplicación, 46
 - configuración del cliente, 724
 - configuración servicio WCF, 741
 - ConfigurationManager, 811
 - confirmar, 214
 - conjunto de datos, 396
 - Connection, 398
 - ConstantExpression, 484
 - Constraint, 397
 - ConstraintCollection, 397
 - contenedor, 798
 - Content, 27, 550
 - ContentControl, 60, 620
 - ContentElement, 59
 - ContentLoader, 675
 - ContentRendered, 41
 - ContextMenu, 60, 176
 - contexto de datos, 99
 - contexto de dominio, 831
 - Load, 832
 - contexto de objetos, 510
 - contextos de corta duración, 522
 - contrato, 713
 - contrato de datos, 717
 - contrato de servicio, 714
 - Control, 26, 59
 - control adornado, 124
 - control de usuario, 590
 - control PasswordBox, 217
 - control, borrar, 866
 - control, mover, 866
 - controlador de eventos, 37, 72, 74
 - añadir, 75
 - controles, 3
 - controles de contenido, 620
 - controles de rango definido, 251
 - controles WPF, 226
 - ControlTemplate, 123
 - conversor, 104, 695
 - convertidores intrínsecos, 104
 - Convert, 77, 104, 236, 695
 - ConvertBack, 104, 695
 - ConverterCulture, 101
 - convertir un entero en una cadena de caracteres en una base, 237
 - convertir una cadena de caracteres en una base a un entero, 236
 - Copy, 188
 - Count, 479
 - crear una aplicación, 857
 - crear una base de datos, 386, 391, 393
 - crear una caja de diálogo, 217
 - crear una tabla, 386
 - CREATE DATABASE, 386
 - CREATE TABLE, 386
 - credentials, 782
 - CSDL, 496
 - cuadro combinado, 247
 - Current, 55
 - CurrentChanged, 692
 - CurrentItem, 333, 350, 692
 - CurrentSource, 550
 - Cut, 188
- D**
- Data Mapper, 501
 - Data Transfer Object, 425
 - DataAdapter, 398, 453
 - DataColumn, 397
 - DataColumnCollection, 397
 - DataContext, 99
 - DataContract, 718
 - DataErrorValidationRule, 108
 - DataForm, 843
 - DataGrid, 271, 426, 436, 685
 - colección de objetos, 376
 - columnas, 378
 - detalles de las filas, 381
 - eventos, 381
 - filas, 379
 - selección de celdas, 380
 - validar, 447

DataGridView, 274, 379
 DataMember, 718
 DataPager, 687, 842
 DataReader, 398, 402
 DataRelationCollection, 397
 DataRow, 397, 459
 DataRowCollection, 397, 464
 DataSet, 452
 añadir al proyecto, 883
 métodos, 452
 DataSourceProvider, 333
 DataTable, 397, 459
 DataTableCollection, 397
 DataTemplate, 125, 264, 322
 DataTrigger, 122
 DataView, 462, 467
 DatePicker, 261
 datos jerárquicos, 328
 DbProviderFactory, 399
 Deactivated, 41, 49
 Debug, 144
 Decorator, 61
 deep link, 670
 deep zoom, 664
 degradados, 634
 delegado, 37, 693
 delegado Func, 477
 delegados, 475
 DELETE, 389
 DeleteCommand, 403
 Deleted, 397
 DeleteObject, 529
 DependencyObject, 40, 58
 depuración, 144
 depurar una aplicación, 872
 descripción abreviada, 29
 DescriptionViewer, 625
 desencadenadores, 121, 325
 deshacer, 188, 202
 deslizador, 255
 Detach, 745
 diálogo Acerca de, 223
 diálogo, introducir datos, 361
 diálogo, permanecer o cancelar, 373
 DialogResult, 220
 diccionario de recursos, 127, 130
 dirección, 713
 directorio virtual, 578
 diseño, 29
 diseño de tipo maestro-detalle, 339

Dispatcher, 304
 DispatcherObject, 58, 304
 DispatcherTimer, 304, 308
 DispatcherUnhandledException, 49
 DisplayMemberPath, 420
 Dispose, 409
 DockPanel, 32, 61
 documento dinámico, 284
 en un fichero, 297
 modificar, 293
 navegar, 293
 documentos dinámicos, 283
 documentos fijos, 283
 DocumentViewer, 283
 DomainContext, 832
 DomainDataSource, 840
 DomainService, 829
 DoubleAnimation, 650
 DragMove, 44
 DROP TABLE, 389
 duración y diario de las páginas, 564
 DynamicResource, 22, 127

E

EDI, 855
 opciones, 876
 personalizar, 877
 EditingElementStyle, 449
 editor de textos, 183
 ejecutar, 16
 ejecutar aplicación, 862
 Elapsed, evento, 304, 305
 ElementName, 98, 103, 316
 elemento actual, 350
 elemento adornado, 124
 elemento de un menú, marcar, 202
 eliminar filas, 529
 Ellipse, 61
 EndEdit, 441
 enfocar un componente, 77
 enlace, 713
 a colecciones, 317
 a una colección, 345
 con otros controles, 103
 de datos de WPF, 89
 de datos WPF, 96
 de datos, 315
 de datos, crear, 99
 información, 114

- profundo, 670
- TwoWay, 349
- Entities, 835
- Entity Client, 497
- Entity Framework, 495
- Entity SQL, 496
- EntityCollection, 511
- EntityKey, 536
- EntityQuery, 835
- EntityReference, 512
- EntitySet, 536
- entorno de desarrollo integrado, 855
- Entrar, Aceptar, 218
- enumeración HorizontalAlignment, 28
- error durante el diseño, 587
- ErrorContent, 111
- Errors, 109
- ErrorTemplate, 109, 124
- Esc, Cancelar, 218
- escribir datos en una tabla, 388
- espacios de nombres, 15, 872
 - declarar en XAML, 102
 - XAML, 18
- eSQL, 496
- esquema del documento, 866
- esquema XSD, 886
- esquemas XML, 886
- estilo, 119
- estilos, 556
- etiqueta, 27, 763
- evento, 4
 - adjunto, 246
 - Checked, 228
 - Click, 76
 - controlado, 70
 - de propagación, directo y de túnel, 70
 - Initialized, 199
 - Loaded, 77, 196, 199
 - LoadingRow, 274
 - NavigationFailed, 567
 - PropertyChanged, 93
 - RowEditEnding, 439
 - SelectionChanged, 437
 - TextChanged, 92
 - Unchecked, 228
 - Unloaded, 199
 - Validation.Error, 110
- eventos, 36, 69, 869
 - adjuntos, 38
 - de entrada, 70

- de la aplicación, 49
- de WPF, 71
- del teclado, 74
- enrutados, 69
- lista, 870
- suscribirse a, 72
- EventSetter, 120
- excepciones, 359
- ExceptionValidationRule, 107, 367
- Execute, 174
- Executed, 151
- ExecuteNonQuery, 401
- ExecuteReader, 401
- Exit, 49
- explorador de bases de datos, 881
- explorador de soluciones, 858
- expresión de consulta, 480, 484
 - compilación, 487
- expresiones lambda, 475
- Expression, 481, 484
- Extension, 474, 488
- extensiones de marcado, 21
- extremo, 713, 795

F

- fecha y hora, 864
- fechas, 260
- fichas, 864
- ficheros y documentos dinámicos, 297
- Figure, 289
- filas, 385
- FileInfo, 700
- FileName, 280
- FileStream, 700
- Fill, 404
- Filter, 280, 693
- FilterDescriptors, 842
- FilterIndex, 280
- filtrar datos, 693, 842
- filtrar los elementos de una colección, 352
- filtro de nombres de fichero, 280
- filtros, 468
- finalizar la ejecución, 16, 862
- Finally, 407
- FindLogicalNode, 40
- FindName, 40
- FindResource, 102, 128
- FixedDocument, 283
- Floater, 289

FlowDocument, 285
 FlowDocumentPageViewer, 284
 FlowDocumentReader, 283
 FlowDocumentScrollViewer, 284
 foco, 61, 77, 218
 Focus, 77, 86, 236
 Focusable, 78
 FocusedElement, 86
 FocusManager, 78, 149
 Font..., 27
 FontFamily, 196
 FontSize, 197
 Format, 77
 FormatException, 77
 FormsAuthentication, 781, 787
 formularios, 3
 Frame, 547, 567, 667
 FrameworkContentElement, 59
 FrameworkElement, 27, 40, 59
 From, 478, 487
 FromByteArray, 698
 FromResource, 699
 fuentes, 195
 fuentes relativas, 118
 Func, 477
 función de un usuario autenticado, 805
 funciones de página, 569
 funciones de usuario, 820

G

GetCustomAttributes, 49
 GetDefaultView, 333
 GetExecutingAssembly, 49
 GetHasError, 373
 GetNavigationService, 551
 GetObject, 47
 GetRolesForCurrentUser, 804
 GetString, 47
 GetWindow, 43
 GoBack, 550
 GoForward, 550
 GotFocus, 78
 GotKeyboardFocus, 78
 gráficos, 642
 Grid, 33, 61, 615
 Grid.Column, 66
 Grid.Row, 66
 GridSplitter, 35, 615
 GridView, 262

Group, 489
 Group ... By, 479
 GroupBox, 232
 GroupDescriptors, 842
 GroupName, 232
 grupos de aplicaciones, 771
 guardar una aplicación, 872

H

habilitar o inhabilitar los elementos de un menú, 199
 Handled, 70, 75, 83
 Handles, 870
 HasError, 109, 373, 376
 HashPasswordForStoringInConfigFile, 783
 HeaderedItemsControl, 60, 328
 Height, 67
 herramientas, caja, 862
 Hide, 43
 HierarchicalDataTemplate, 264, 328
 historial de navegación, 667
 HorizontalAlignment, 28, 68
 HorizontalContentAlignment, 28, 68
 hospedar aplicaciones WPF, 545
 Hyperlink, 551, 565
 HyperlinkButton, 669, 674

I

ICommand, 146, 165
 ICommandSource, 149
 Icon, 44, 175, 204, 869
 icono de la aplicación, 869
 icono, añadir a la aplicación, 204
 IDataErrorInfo, 108, 112
 identidad utilizada por el grupo de aplicaciones, 771
 IDENTITY, 746
 IDisposable, 409
 IEditableObject, 440
 IEnumerable, 479, 493
 Image, 175, 178, 191
 ImageBrush, 126
 imagen, añadir a un botón, 191
 imagen, cargar, 699
 imágenes, 619, 694
 ImageSource, 695
 imprimir, 282
 imprimir un documento dinámico, 300

- INavigationContentLoader, 675
- información sobre un enlace, 114
- iniciadores de colecciones, 474
- iniciadores de objetos, 473
- inicio de sesión, 818
- InitialDirectory, 280
- Initialized, 199
- Inline, 289
- InlineUIContainer, 290
- INotifyCollectionChanged, 354
- INotifyPropertyChanged, 93, 317, 349, 678
- InputBinding, 156, 158
- InputGesture, 156, 158
- InputGestureText, 145, 155
- Insert, 243
- INSERT, 388
- insertar filas, 526
- insertar o borrar elementos en una vista, 353
- InsertCommand, 403
- interceptar la tecla pulsada, 84
- interfaces de usuario dinámicas, 41
- interfaces gráficas, 5
- interfaz, 3
 - ICommand, 146, 165
 - IEditableObject, 440
- Into, 491
- introducir datos, 360
- Invoke, 304
- inyección de código SQL, 411
- inyectar código XAML, 38
- IQueryable, 479, 493
- IsActive, 41, 43
- IsCancel, 218
- IsCheckable, 175
- IsChecked, 175, 228
- IsCurrentUserInRole, 804
- IsDefault, 67, 218
- IsEditable, 247
- IsEnabled, 176, 200
- IsFocusScope, 150
- IsKeyDown, 86
- IsKeyToggled, 86
- IsKeyUp, 86
- isla de datos XML, 331
- IsLoggedIn, 800
- IsMouseDirectlyOver, 81
- IsMouseOver, 81
- IsMuted, 655
- IsReadOnly, 247
- IsThreeState, 228

- IsVisible, 78
- it, 500
- Items, 205
 - propiedad, 240
- ItemsControl, 60
- ItemsSource, 272, 321
- ItemsSource, propiedad, 240
- IValueConverter, 104

J

- Join, 479, 490
- Journal, 565
- JournalOwnership, 671

K

- KeepAlive, 551, 564, 601
- Key, 85, 156
- KeyBinding, 155
- Keyboard, 78, 86
- KeyDown, 70, 74
- KeyGesture, 156, 158
- KeyUp, 74

L

- Label, 26, 60, 61, 625
- LargeChange, 251
- LayoutMode, 869
- Lazy Loading, 501
- lector de datos, 401
- Left, 44
- Let, 492
- Line, 61
- LinearGradientBrush, 635
- líneas de ayuda, 868
- LineBreak, 224, 290
- LINQ, 471
- LINQ to Entities, 496
- List, 286
- lista, 238, 864
 - acceder a sus elementos, 242
 - borrar elemento, 244
 - de elementos CheckBox, 245
 - de los eventos, 870
 - desplegable, 246, 864
- añadir elemento, 250
- borrar elemento, 250
- diseñar, 248

- diseño, 241
- elemento seleccionado, 240
- selección múltiple, 243
- ListBox, 26, 60, 238, 320
- ListBoxItem, 60, 238
- ListCollectionView, 320
- ListView, 262
- Load documento dinámico, 300
- Loaded, 41, 77, 196, 199
- LoadingRow, 274
- LoadOperation, 832
- LoadSize, 842
- LocationChanged, 44
- LogicalTreeHelper, 40
- Login, 800
- Logout, 801
- LostFocus, 79
- LostKeyboardFocus, 78

M

- maestro-detalle, 339, 354
- mage.exe, 581
- Main, 13
- MainWindow, 41, 55
- manejador de eventos, 37
- marcas de tiempo, 745
- marco, 863
- marco de la ventana, 2
- Margin, 28, 68
- máscaras, 130
- Matrix, 643
- MatrixTransform, 646
- Max, 479
- MaxDropDownHeight, 247
- MaxHeight, 67
- maximizar, 2
- Maximum, 251
- MaxWidth, 67
- MediaElement, 654
- MediaEnded, 655
- MediaFailed, 655
- MediaOpened, 655
- MemberExpression, 484
- membership, 796, 781
- mensaje, mostrar, 144
- Menu, 60, 139
- menú, 138
 - añadir imagen a un elemento, 175
 - contextual, 175

- controlador de un elemento, 142
- de control, 2
- de desbordamiento, 180
- dinámico, 204
- emergente, 175
- señalar elemento, 201
- MenuBase, 60
- MenuItem, 60, 139
- MenuItem.Icon, 175
- menús, 137, 863
 - detalles, 175
 - diseño, 139
 - líneas de separación, 142
- MessageBox, 144, 212
- MessageBox.Show, 91
- MessageBoxResult, valor retornado..., 213
- metadatos, 564
- MethodCallExpression, 484
- método abreviado, 155
- método Main, 13
- método Run, 14
- método Shutdown, 14
- Métodos del generador de consultas, 497
- métodos extensores, 474
- Microsoft Access, 400
- Microsoft.ACE.OLEDB, 401
- Microsoft.Jet.OLEDB, 401
- Microsoft.Win32, 278
- Min, 479
- MinHeight, 67
- minimizar, 2
- Minimum, 251
- MinWidth, 67
- modal o no modal, 212
- Mode, 97, 678
- modelo asincrónico, 724
- modelo de datos, 343
- modelo de entidades, 496
- modelos de contenido, 62
- Model-View-ViewModel, 166
- modificadores de tamaño, 866
- modificar datos, 522
- modificar datos en una tabla, 388
- Modified, 397
- ModifierKeys, 156
- Modifiers, 86, 156
- mostrar datos, 764
- mostrar un mensaje, 144
- Mouse, 81
- MouseBinding, 158

MouseDown, 80
MouseEnter, 80
MouseLeave, 81
MouseLeftButtonDown, 662
MouseLeftButtonUp, 662
MouseMove, 80
MouseUp, 80
MoveCurrentToFirst, 691
MoveCurrentToNext, 335, 351
mover el control, 866
MSL, 496
MultiTrigger, 121
Mutex, 52

N

navegación
 de fragmento, 567
 de Silverlight, 666
 de tipo web, 546
 estructurada, 569
 externa, 674
 falla, 567
 historial, 568
 personalizada de Silverlight, 665
navegar por los elementos de una vista, 351
Navigate, 550, 564, 668
NavigateUri, 565, 669
Navigating, 670, 807, 822
Navigation, 668, 672
NavigationFailed, 671
NavigationService, 547, 551
NavigationUIVisibility, 568
NavigationWindow, 547
nemónico, 67, 144
N-Layer architecture, 744, 775
NotifyOnValidationError, 109
NT AUTHORITY\Servicio de Red, 746
N-Tier architecture, 744, 775
Nullable(Of Boolean), 220

O

Object, 58
ObjectCollection, 243, 249
ObjectContext, 503
ObjectDataProvider, 358
ObjectQuery, 497
ObjectSet(Of), 510
ObjectStateEntry, 523, 536

ObjectStateManager, 523, 536
objeto aplicación, referencia, 55
objetos como orígenes de datos, 343
objetos de negocio, 428
ObservableCollection, eventos, 442
ObservableCollection, InsertItem, 442
OdbcConnection, 399
OdbcDataAdapter, 403
OleDbConnection, 399
OleDbDataAdapter, 403
OleDbDataReader, 402
OnReturn, 574
OnStartup, 52
OpenFile, 280
OpenFileDialog, 279, 700
OpenRead, 700
operadores de consulta, 478
OperationContract, 714
OptimisticConcurrencyException, 532
OracleConnection, 399
OracleDataAdapter, 403
orden, 403
orden enrutada, 145
 información adicional, 159
 parámetros, 162
orden parametrizada, 414
orden Tab, 61
ordenación de los datos, 842
ordenar una vista de elementos, 352
órdenes, 145
 enrutadas comunes, 146
Order By, 479, 487
Orientation, 255
origen de datos implícito, 100, 103
OriginalSource, 75
ORM, 495
OwnedWindows, 43, 226
Owner, 43, 226

P

Padding, 69
Page, 551
 duración, 564
PagedCollectionView, 686, 763
PageFunction, 569
PageSize, 842
página de Silverlight, 668
paginación, 687, 842
páginas, pasar datos entre, 563

Panel, 29, 60
 paneles de diseño, 29
 pantalla de presentación, 53
 Paragraph, 286, 291
 Parameter, órdenes enrutadas, 163
 ParameterExpression, 484
 Parameters, 414
 parámetros de consulta, 841
 parámetros en órdenes SQL, 414
 pasos, 314
 Password, 217
 PasswordBox, 59, 217, 624
 PasswordChar, 217
 Paste, 188
 Path, 98, 101, 316
 patrones de diseño, 501
 perfiles, 808, 821
 plantilla, 123
 plantillas de datos, 320
 pool de conexiones, 407
 Popup, 636
 portapapeles, 186
 posición del ratón, 81
 posición inicial de la ventana, 41
 Position, 656
 PostgreSQL, 400
 predicado, 693
 PresentationHost, 545
 PreviewGotKeyboardFocus, 78
 PreviewKeyDown, 70, 74, 85
 PreviewKeyUp, 74
 PreviewLostKeyboardFocus, 78
 PreviewMouseDown, 80
 PreviewMouseMove, 80
 PreviewMouseUp, 80
 PreviewTextInput, 74
 PRIMARY KEY, 387
 Print, 300
 PrintDialog, 278, 301
 PrintDocument, 283
 PrintVisual, 301
 procedimiento almacenado, 415, 427
 producto cartesiano, 490
 ProfileService, 790, 795, 808
 ProgressBar, 60, 256
 Property, 25
 PropertyChanged, 93, 97, 316
 propiedad
 asociada, 23
 auto-implementada, 473

 cambió, notificar, 93
 como atributo, 19
 como elemento, 20
 Content, 27
 de contenido, 20
 de dependencia, 24, 165
 Font..., 27
 HorizontalContentAlignment, 28
 Icon, 869
 Items, 205
 Margin, 28
 Mode, 97
 PropertyChanged, 97, 316
 Resources, 126
 StartupUri, 14
 ToolTip, 29
 UpdateSourceTrigger, 97
 Visibility, 207
 xmlns, 9
 propiedades
 asociadas, 109
 de navegación, 510
 de un objeto, 867
 del proyecto, 874
 HorizontalAlignment y VerticalAlignment, 28
 proveedor de datos, 398
 proveedor de datos de objetos, 358
 proveedor de entidades, 497
 proveedor XmlDataProvider, 327
 proveedores de LINQ, 493
 publicar un servicio WCF, 769
 publicar una aplicación Silverlight, 730
 Publicar XBAP, 578
 puntero, 862
 punto de inserción, 77
 punto final, 713, 795

Q

QueryParameters, 841

R

RadialGradientBrush, 635
 RadioButton, 26, 60, 232
 RangeBase, 60
 Read, 402
 Rectangle, 61
 recursos, 22, 126
 creados mediante código, 128

- de una aplicación, 47
- del sistema, 129
- Redo, 188
- referencias, 859
- reflexión, 49, 618
- Refresh, 381, 534
- registrar usuario, 819
- registros, 385
- reglas de validación, 107
- rehacer, 188
- rejilla de ayuda, 868
- RelativeSource, 23, 118, 451
- reloj, 303
 - despertador, 309, 312
 - digital, 304
- Remove, 243
- RemoveAt, 243
 - colección, 208
- RemoveBackEntry, 550
- RemovedItems, 438
- RequiresAuthenticationAttribute, 820
- RequiresRoleAttribute, 820
- SizeMode, 44, 217
- ResourceDictionary, 127
- ResourceManager, 47
- Resources, 126
- RestoreBounds, 44
- Result, 574
- ReturnEventArgs, 574
- RIA Services, 813, 825
- RichTextBox, 59, 197, 283, 297
- roles, 820
- RoleService, 790, 795, 804
- RootVisual, 611
- RotateTransform, 646
- RoutedCommand, 146
- RoutedEvent, 69
- RoutedEventArgs, 70
- RoutedEventHandler, 37, 73
- RoutedUICommand, 148
- Row, 33, 439, 446
- RowChanged, 466
- RowDeleted, 466
- RowEditEnding, 439
- RowFilter, 468
- RowSpan, 33
- RowState, 464
- RowValidationRules, 450
- Run, 4, 14, 224, 290, 291

S

- Save documento dinámico, 299
- SaveChanges, 510
- SaveFileDialog, 280
- ScaleTransform, 645
- Scroll, 254
- ScrollBar, 26, 60, 251
- ScrollIntoView, 319, 335, 351
- ScrollViewer, 35, 60, 258
- Section, 286
- SecurePassword, 217
- SecureString, 217
- seguimiento de los cambios, 535
- seguridad en XBAP, 581
- seleccionar datos de una tabla, 389
- seleccionar el contenido de una caja de texto, 79
- seleccionar un objeto, 867
- Select, 80, 188, 487
- SELECT, 389
- selectAll, 79
- SelectAll, 188
- SelectCommand, 403
- SelectedIndex, 240
- SelectedIndexChanged, 520
- SelectedItem, 240, 249
- SelectedItems, 243
- SelectedText, 80, 188
- SelectedValue, 420
- SelectedValuePath, 420
- SelectionChanged, 271, 437
- SelectionLength, 80, 188
- SelectionStart, 79, 188
- Selector, 60
- sentencia Using, 409
- señalar un elemento de un menú, 201
- separadores, 142
- Separator, 59, 139
- seriación/deseriación, 11
- ServiceContract, 714
- servicio, 711
 - de conexiones, 407
 - de dominio, 815, 829, 831
 - de navegación, 551
 - de red, 747
 - de suscripciones, 796
 - WCF, 712
 - WCF de funciones, 804
 - WCF habilitado para Silverlight, 734

- WCF para acceso a datos, 748
- servicios de aplicación, 778, 790
 - configurar, 793
- Servicios de objetos, 498
- servicios webs y LINQ, 743
- SessionEnding, 49
- SetResourceReference, 129
- Setter, 119, 326
- Shape, 61
- Show, 43, 212, 215, 226
- ShowActivated, 44
- ShowDialog, 41, 215, 226, 279, 700
- ShowGridLines, 33
- ShowInTaskbar, 42, 44, 217
- ShowsNavigationUI, 550
- Shutdown, 14
- ShutdownMode, 52
- SignOut, 787
- Silverlight, 605, 708
 - <Object>, 613
 - acceso a datos, 676
 - administrador de identificadores de recursos, 672
 - aplicación fuera del explorador, 675
 - arquitectura de una aplicación, 609
 - arquitectura, 606
 - autenticación y autorización, 675
 - compilación de la aplicación, 612
 - comunicación entre dominios, 728
 - controles, 615
 - crear aplicación, 608
 - diccionario de recursos, 667
 - diseño de una página, 615
 - Frame, 666
 - llamadas asíncronas, 725
 - navegación, 676
 - Page, 666
 - página de entrada, 613
 - SDK, 624, 630
 - vista de una colección, 686
- simultaneidad, 532, 744
- sincronizar con CurrentItem, 333
- sistema de diseño, 29
- sistema de navegación Silverlight, extender, 675
- sitio web, 578
 - administración, 798
- SizeToContent, 44
- SkewTransform, 646
- Slider, 60, 255

- SmallChange, 251
- SOAP, mensaje, 712
- SolidColorBrush, 649
- soluciones y proyectos, 875
- Sort, 468
- SortDescriptors, 842
- Source, 75, 98, 316, 550
- Span, 291
- SplashScreen, 53
- SQL, 386
- SQL Server, 400, 408, 746, 877
- SQL Server Management Studio Express, 880
- SQLCMD, 878
- SqlConnection, 399
- SqlDataAdapter, 403
- SqlDataReader, 402
- SSDL, 496
- StackPanel, 30, 61, 178
- Startup, 49, 51
- StartupUri, 14, 50
- StateChanged, 45
- StaticResource, 22, 127
- StatusBar, 60, 182
- StatusBarItem, 60
- Storyboard, 326, 648, 650
- Stretch, 655
- Style, 58, 119, 270, 326
- subprocesos, 304
- Sum, 479
- suscribirse a un evento, 36, 72
- System.Data.Linq, 471
- System.Linq, 471
- System.Linq.Expressions, 483
- System.Runtime.CompilerServices, 474, 488
- System.Windows, 28
- System.Windows.Markup, 11
- SystemColors, 129
- SystemFonts, 129
- SystemParameters, 129

T

- Tab, 61
- TabControl, 259
- TabIndex, 61
- tabla, 385, 864
- Table, 287
- Take While, 479
- tamaño de los controles, 866
- tamaño de una fuente, 197

tamaño de una ventana, 64
TargetNullValue, 445
TargetType, 119
tecla de acceso, 28, 61, 67
tecla Entrar, 62
tecla pulsada, interceptar, 84
temas, 130
TemplateBinding, 23
temporizador, 303
 resolución, 306
Text, 75, 77, 247
TextBlock, 26, 60, 224, 618
TextBox, 26, 59, 62, 188, 624
TextBoxBase, 59, 188
TextChanged, 76, 92, 235
TextInput, 74
texto seleccionado, 200
TextProperty, 98
TextRange, 298
TextWrapping, 184
Tick, evento, 304
TickFrequency, 255
TickPlacement, 255
ticks, 314
Timer, 304
TimeSpan, 307, 661
tip, 29
tipo anónimo, 472
tipo implícito, 472
tipos SQL, 387
ToByteArray, 697
ToDouble, 77
ToggleButton, 60
ToInt32, 236
ToList, 511
ToolBar, 60, 177
ToolBarTray, 180
Toolkit.dll, 843
ToolTip, 29, 60
Top, 44
Topmost, 45
ToString, 237
ToUpper, 237
transacciones, 417
 explícitas, 421
TransactionScope, 417
transformaciones, 643
TransformGroup, 646
TranslateTransform, 644
TreeView, 264

TreeViewItem, 264
Trigger, 121
Triggers, 325
TryCast, 76
TryFindResource, 128

U

UIElement, 58, 97
Unchanged, 397
Unchecked, 228
Undo, 188, 202
UNIQUE, 387
Unloaded, 199
Update, 404, 466
UPDATE, 388
UpdateCommand, 403
UpdateSource, 114
UpdateSourceTrigger, 97, 101
URI, 567, 672
UriMapper, 672
UriMapping, 672
User, 820
UserControl, 60, 590, 611
Using, 409

V

validación, 366
 de los datos, 107, 625
 personalizada, 116, 370
 proceso, 112, 371
 visualizar errores, 368
validar un campo de texto, 87
validar un grupo de enlaces, 374
Validate, 116, 370
ValidateUser, 801
Validation, 109, 369
Validation.Error, 110
Validation.GetHasError, 110
ValidationAttribute, 621
ValidationRule, 107, 116, 366
ValidationSummary, 625
Value, 251, 257
Value Object, 425, 501
ValueChanged, 251
var, 472
varbinary, 746
variable, declaración implícita, 472
ventana, 1

ventana activa, 41
 modal, 41
 no modal, 41
 posición, 41
 ventanas, 3
 colección, 41
 VerticalAlignment, 68
 VerticalContentAlignment, 28, 68
 VerticalScrollBarVisibility, 185
 vídeo, 654, 662
 View, 319
 Viewbox, 35
 vinculación de datos, 89
 virtualizar, 359
 VirtualizingStackPanel, 359
 Visibility, 69, 207
 vista de colección, 318, 347
 vista de un DataTable, 467
 vista de una colección, 333
 vista PagedCollectionView, 763
 vista predeterminada, 350
 vista
 agrupar sus elementos, 337
 elemento actual, 333
 filtrar sus elementos, 336, 592
 navegar por sus elementos, 334
 obtener, 333
 ordenar sus elementos, 335
 Visual, 58
 Volume, 655

W

WCF, 711
 WCF RIA Services, 813, 825
 WebBrowser, 602
 webcam, 664
 WebContext, 816, 820
 Where, 487

Width, 67
 Window, 9, 60
 Windows, 41
 Windows Forms, 5
 WindowStartupLocation, 41, 45
 WindowState, 45, 55
 WindowStyle, 45
 WindowTitle, 551
 WPF, 5, 57
 WPF, XBAP y Silverlight, 545
 WrapPanel, 31, 61
 WriteableBitmap, 697
 WriteLine, 144

X

x:Array, 23
 x:Name, 592
 x:Null, 22
 x:Static, 23, 130
 x:Type, 23
 x:XData, 331
 XAML, 8, 10
 extensiones de marcado, 21
 información general, 17
 XamlReader, 11, 40
 XamlWriter, 11
 XBAP, 6, 547, 576
 publicar, 578
 seguridad, 581
 y bases de datos, 583
 XCOPY, 877
 XML, simular BBDD, 887
 XmlDataProvider, 327
 XmlElement, 334
 xmlns, 9, 18
 XPath, 316, 330
 XPath=@..., 269, 330
 XSD, 886

Del mismo autor

- Curso de programación con **PASCAL** ISBN: 978-84-86381-36-3
224 págs.
 - Curso de programación **GW BASIC/BASICA** ISBN: 978-84-86381-87-5
320 págs.
 - Manual para **TURBO BASIC** ISBN: 978-84-86381-43-1
Guía del programador 444 págs.
 - Manual para **Quick C 2** ISBN: 978-84-86381-65-3
Guía del programador 540 págs.
 - Manual para **Quick BASIC 4.5** ISBN: 978-84-86381-74-5
Guía del programador 496 págs.
 - Curso de programación **Microsoft COBOL** ISBN: 978-84-7897-001-8
480 págs.
 - Enciclopedia del lenguaje **C** ISBN: 978-84-7897-053-7
888 págs.
 - Curso de programación **QBASIC y MS-DOS 5** ISBN: 978-84-7897-059-9
384 págs.
 - Curso de programación **RM/COBOL-85** ISBN: 978-84-7897-070-4
396 págs.
 - El abecé de **MS-DOS 6** ISBN: 978-84-7897-114-5
224 págs.
 - Microsoft **Visual C ++** (ver. 1.5x de 16 bits) ISBN: 978-84-7897-180-0
Aplicaciones para Windows 846 págs. + 2 disquetes
 - Microsoft **Visual C ++** ISBN: 978-84-7897-561-7
Aplicaciones para Win32 (2ª edición) 792 págs. + disquete
 - Microsoft **Visual C ++** ISBN: 978-84-7897-344-6
Programación avanzada en Win32 888 págs. + CD-ROM
 - **Visual Basic 6** ISBN: 978-84-7897-357-6
Curso de programación (2ª edición) 528 págs. + disquete
 - Enciclopedia de Microsoft **Visual Basic 6** ISBN: 978-84-7897-386-6
1.072 págs. + CD-ROM
 - El lenguaje de programación **Java** ISBN: 978-84-7897-485-6
320 págs. + CD-ROM
 - El lenguaje de programación **C#** ISBN: 978-84-7897-500-6
320 págs. + CD-ROM
-

Del mismo autor

- El lenguaje de programación
Visual Basic.NET ISBN: 978-84-7897-525-9
464 págs. + CD-ROM
 - **Java 2** ISBN: 978-84-7897-745-1
Lenguaje y aplicaciones 392 págs. + CD-ROM
 - Programación orientada a objetos
con C ++ (4ª edición) ISBN: 978-84-7897-761-1
648 págs. + CD-ROM
 - **C/C++** ISBN: 978-84-7897-762-8
Curso de programación (3ª edición) 708 págs. + CD-ROM
 - **Microsoft C#** ISBN: 978-84-7897-813-7
Lenguaje y aplicaciones (2ª edición) 520 págs. + CD-ROM
 - **Java 2.** Interfaces gráficas y aplicaciones para
Internet (3ª edición) ISBN: 978-84-7897-859-5
718 págs. + CD-ROM
 - **Aplicaciones .Net multiplataforma**
(Proyecto Mono) ISBN: 978-84-7897-880-9
212 págs. + CD-ROM
 - Enciclopedia del lenguaje
C ++ (2ª edición) ISBN: 978-84-7897-915-8
902 págs. + CD-ROM
 - Enciclopedia de Microsoft
Visual C# (3ª edición) ISBN: 978-84-7897-986-8
1.110 págs. + CD-ROM
 - Enciclopedia de Microsoft
Visual Basic (2ª edición) ISBN: 978-84-7897-987-5
1.090 págs. + CD-ROM
 - **Microsoft Visual Basic .NET** ISBN: 978-84-9964-020-4
Lenguaje y aplicaciones (3ª edición) 520 págs. + CD-ROM
 - **Java 2** ISBN: 978-84-9964-032-7
Curso de programación (4ª edición) 820 págs. + CD-ROM
 - **Microsoft C#** ISBN: 978-84-9964-068-6
Curso de programación (2ª edición) 850 págs. + CD-ROM
 - **Visual C#.** Interfaces gráficas y aplicaciones con
WPF, WCF y Silverlight ISBN: 978-84-9964-203-1
958 págs. + CD-ROM
-

INSTALACIÓN

Para instalar el kit de desarrollo de Visual Basic y los ejemplos de este libro descargue la plataforma de desarrollo de la dirección: <http://www.microsoft.com/express/>.

PLATAFORMA WINDOWS

Instalación de .NET Framework SDK

Hay que instalar .NET Framework Redistributable Package antes de instalar .NET Framework SDK. El primer paquete incluye todo lo necesario para ejecutar aplicaciones desarrolladas con .NET Framework. El segundo paquete incorpora todo lo necesario para escribir, construir, verificar y desplegar aplicaciones desarrolladas con .NET Framework.

Para realizar la instalación, siga las instrucciones mostradas por el asistente de instalación. Esta instalación se realiza en la carpeta *Microsoft.NET* de Windows y en la carpeta *Microsoft.NET* de archivos de programas.

Instalación de Microsoft Visual Studio

La instalación de Visual Studio (el paquete completo o las versiones *Express* que necesite) no requiere la instalación previa del SDK porque está incluido en éste. Descargue de Internet los paquetes Visual Basic 2010 Express y Visual Web Developer 2010 Express (ambos incluyen SQL Server Express y la ayuda en línea), o bien, si tiene acceso, Visual Studio 2010, e instálos.

Ejemplos del libro

Los ejemplos del libro puede instalarlos en la carpeta *Projects* de Visual Studio o los puede recuperar directamente desde el CD cuando los quiera consultar.

LICENCIA

Todo el contenido de este CD, excepto los ejemplos del libro, es propiedad de las firmas que los representan (Microsoft, etc.). La inclusión en este libro se debe a su gentileza y es totalmente gratuita y con la finalidad de apoyar el aprendizaje del software correspondiente. Para obtener más información y actualizaciones visite las direcciones indicadas en dicho software:

<http://www.microsoft.com/es/es/default.aspx>

Al realizar el proceso de instalación, haga el favor de consultar el acuerdo de licencia para cada uno de los productos.

WEB DEL AUTOR: <http://www.fjceballos.es>

En esta web podrá echar una ojeada a mis publicaciones más recientes y acceder a la descarga del software necesario para el estudio de esta obra así como a otros recursos.