Tema 12; Acceso desconectado BBDD

Entorno desconectado

-los datos pueden modificarse de forma independiente y los cambios se escriben posteriormente en la base de datos
-Las conexiones se utilizan durante el menor tiempo posible
-menos conexiones den servicio a más usuarios
-mejora la escalabilidad y el rendimiento de las aplicaciones
-Los datos no siempre están actualizados, pueden producirse conflictos
-varios usuarios pueden modificar los datos de los mismos registros al mismo tiempo
- En modo desconectado SIEMPRE necesitamos recuperar los datos (SELECT) para poder trabajar con ellos (INSERT, UPDATE, DELETE)
-Para recuperar datos se necesita: Un DataAdapter que ejecute la SQL y un DataSet donde guardar el resultado
-Mejor si se va a trabajar con más de una tabla o más de una base de datos

DataSet

-representación de una base de datos relacional en memoria. Crea una BD virtual
-Almacena datos en un caché distinto de la base de datos
-No necesidad de conexión continua
-permite trabajar con una copia de las partes de la BD con las que queremos trabajar, liberando la conexión
-System.Data.OleDb y System.Data.SqlClient NO proporcionan clases para DataSet
-Recomendado para acceso a datos complejo; recibir y almacenar datos (guardar artículos por tipo de un proveedor)

DataAdapter

-DataAdapter se utiliza para insertar datos en un DataSet
-Utiliza comandos para actualizar el origen de datos después de hacer modificaciones en el objeto DataSet
-Abre y cierra la conexión automáticamente
-CommandBuilder recibe un DataAdapter y crea los comandos SQL para actuar sobre la BD
-System.Data.OleDb y System.Data.SqlClient proporcionan clases para DataAdapter y CommandBuilder

DataRow

-Representa una única fila de información de la tabla
-Se puede acceder a los valores individuales usando el nombre de campo o un índice

DataColumn

-No contienen ningún dato real.
-Almacenan información sobre la columna (tipo de datos, valor predeterminado, restricciones..)

DataRelation

-Especifica una relación padre/hijo entre dos tablas diferentes de un DataSet

DataView

-Proporciona una vista sobre una DataTable.
-Cada DataTable tiene al menos un DataView, que se usa para la vinculación de datos.
-DataView muestra los datos de DataTable sin cambios o con opciones especiales de filtro u ordenación

DataTable

-Se utiliza en los DataSet para modificar las tablas
-Tables[“tabla”] devuelve la tabla reclamada

GridView

-presentación de datos en forma de tabla
-permite: Selección, Paginación, Ordenación, Edición y es extensible mediante plantillas

-.DataSource es el origen de datos. Podemos asignarle una tabla de la BD como origen de datos, o los registros obtenidos de una SQL
-.DataBind() muestra el contenido del GridView
-Para la paginación en GridView debemos usar AllowPaging = true y PageSize = X (número de elementos por página). Si no se usa el asistente se debe usar en el código del GridView un PageIndex
-Los tipos de columnas son:

BoundField	Muestra el texto de un campo de la BBDD
ButtonField	Muestra un botón para cada item
CheckBoxField	Muestra un checkbox para cada item
CommandField	Proporciona funciones de selección, edición y borrado
HyperLinkField	Muestra el texto de un campo de la BBDD como un hipervínculo
ImageField	Muestra una imagen
TemplateField	Permite especificar múltiples campos y controles personalizados

-EmptyDataText Se utiliza para mostrar un mensaje cuando no existen datos que mostrar en el GridView
-EmptyDataTemplate permite personalizar el mensaje mostrado cuando el GridView está vacío

Gestión de conflictos

-Concurrencia pesimista: Cuando una fila es leída, esta queda bloqueada para su lectura para cualquier otro que la demande hasta que aquel que la posee la libere
-Concurrencia positiva: Las filas están disponibles para su lectura en todo momento, pueden ser leídas por distintos usuarios al mismo tiempo. Cuando alguno intenta modificar una fila ya modificada se produce un error y no se modifica
-Last win: No existe control. El último cambio en escribirse es el que permanece

Concurrencia positiva

-el DataSet mantiene dos versiones de las filas que leímos: la original y la actualizada
-Cuando se actualiza la fila, se comparan los valores originales con la fila real de la BD, para ver si ha sido modificada. Si ha sido modificada, hay que capturar una excepción, si no, se actualiza

El evento RowUpdated

-Se produce Al actualizar una fila: después de cada operación pero antes de lanzar cualquier excepción
-permite examinar los resultados e impedir que se lance una excepción para conservar los cambios