Programación Visual en C#.NET

Guion de la práctica 4

OBJETIVOS

- Aprender a manejar otros controles en cajas de diálogos
- Diseñar cajas de diálogos de tamaño variable
- Comprender el mecanismo de seriación

TEMPORIZACIÓN

Recogida del enunciado:	Semana del 18 de octubre
Desarrollo de la práctica:	**1 semana**
Fecha de entrega:	Semana del 25 de octubre junto con las prácticas 1, 2 y 3
Fecha límite de entrega:	Semana del 1 de noviembre

PRÁCTICA 4
Otros controles. Seriación

TABLA DE CONTENIDOS:

8. Introducción

Se continuará trabajando con el programa desarrollado a lo largo de las prácticas anteriores.

En el punto 8.1.1 se añadirá al programa una casilla de verificación que, además de permitir al usuario activar o desactivar la alarma fácilmente, le mostrará en todo momento el estado de la misma.

En el punto 8.1.2 se creará un nuevo diálogo en el que se usará un control de tipo **ListBox**. El diálogo servirá para que el usuario pueda eliminar zonas horarias. La lista mostrará los nombres de las diferentes zonas horarias introducidas previamente.

En el punto 8.1.3 se hará que el diálogo mencionado anteriormente pueda cambiar de tamaño, y que los controles contenidos en él cambien también de tamaño y/o posición para aprovechar todo el espacio disponible. A continuación se completará la funcionalidad del programa en lo que respecta a añadir, usar y eliminar zonas horarias. El programa almacenará todas las zonas horarias en una colección genérica. Habrá que poner especial cuidado en mantener la consistencia entre estos datos y las órdenes del menú.

En el punto 8.2 se implementará la seriación de los datos de las zonas horarias. Cuando el programa se cierre, guardará estos datos en un archivo. Cuando se vuelva a abrir, se cargarán los datos de nuevo y el usuario podrá volver a usar las zonas horarias que introdujo la última vez que usó el programa.

8.1. Otros controles

8.1.1 Casilla de verificación

En una práctica anterior se añadió al formulario principal un botón para probar el formulario `DlgDatosZona` y un manejador para dicho botón. Elimine ambos.

Añada al formulario `RelojDigital` una casilla de verificación (**CheckBox**) con el texto "Despertador activado" y el nombre `cv_Despertador`.

El estado de la casilla de verificación deberá concordar, en todo momento, con el valor de la variable miembro `m_DespertadorActivado`. Para conseguir este efecto, añada un manejador para el evento **Click** de la casilla de verificación, y niegue en él el valor de la propiedad `DespertadorActivado`. Por otro lado, añada la siguiente línea al final del método `set` de dicha propiedad:

```
cv_Despertador.Checked = m_DespertadorActivado;
```

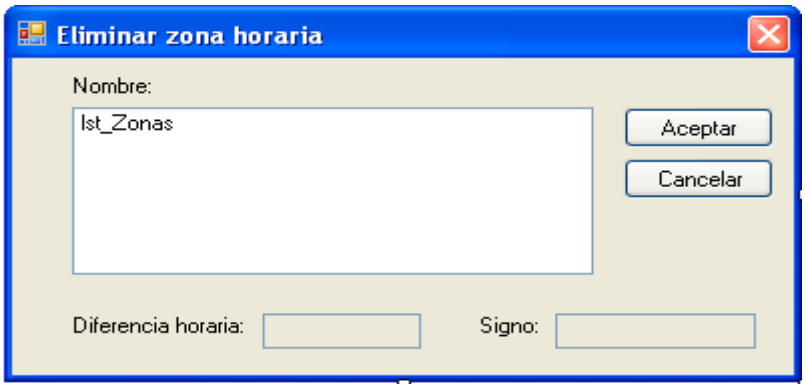
Compilar y ejecutar para ver el resultado.

8.1.2 Listas

Añada un nuevo formulario de nombre `DlgEliminarZona` y título "Eliminar zona horaria". Copie los controles del formulario `DlgDatosZona` y péguelos en el nuevo formulario. Sustituya la caja de texto con máscara por una normal.

Sustituya también la caja de texto `ct_Nombre` por una lista (**ListBox**) con el identificador `lb_Zonas`. Desactive la propiedad **TabStop** y active la propiedad **ReadOnly** en el control `ct_Diferencia`. Sustituya los **RadioButton** por una caja de texto (también **ReadOnly** y no **TabStop**) que se llame `ct_Signo`.

Asigne el valor **Sizable** a la propiedad **FormBorderStyle** de `DlgEliminarZona`, y desactive sus propiedades **Maximize**/**MinimizeBox**. Si lo desea, agregue un icono personalizado para el formulario. Cuando haya terminado, el aspecto del formulario deberá ser parecido al de la siguiente figura:



8.1.3 Cambio de las dimensiones de la ventana de diálogo

Establecer como tamaño mínimo el tamaño inicial del formulario

Añada la siguiente línea en el constructor de `DlgEliminarZona`, para hacer que el límite mínimo en el tamaño del diálogo sea el tamaño inicial de éste:

```
this.MinimumSize = this.Size;
```

Añada a `RelojDigital` un botón provisional que muestre un diálogo modal de la clase `DlgEliminarZona` (también puede usar una orden de un menú si lo prefiere).

Compilar y ejecutar para ver el resultado.

Redimensionar el formulario conservando la posición de los controles

Seleccione, en la propiedad **Anchor** de cada control, a qué bordes del formulario se quiere dejar anclado el control:

- Los controles `ct_Signo` y `ct_Diferencia`, así como sus etiquetas asociadas, deben quedar anclados a los bordes inferior e izquierdo.
- Los botones "Aceptar" y "Cancelar" deben quedar anclados a los bordes superior y derecho.
- La lista de zonas, así como su etiqueta asociada, deben quedar ancladas a los bordes superior e izquierdo y, además, la lista se debe redimensionar según lo haga formulario.

Compilar y ejecutar para ver el resultado.

8.1.4 Nueva clase para almacenar los datos de las zonas horarias

Añada al proyecto una nueva clase pública (**public**) llamada `ZonaHoraria`. Usando la herramienta *Insertar fragmento de código* añada los siguientes campos privados, así como las propiedades públicas que permiten acceder a estos campos:

```
TimeSpan m_Diferencia = new TimeSpan();
bool m_Positivo = true;
string m_Nombre = "";
```

Agregue a `RelojDigital` una colección privada (**private**) de objetos de tipo `ZonaHoraria` llamada `m_Zonas`. Esta colección será un objeto de la colección genérica `List<T>`. Cambie la propiedad `NúmeroZonas` para que devuelva el número de elementos de esta colección y añada una nueva propiedad `Zonas` para que devuelva la colección `m_Zonas`. Obsérvese que utilizando una colección, no es necesario limitar el número máximo de zonas: elimine `MaxZonas`.

Modifique `zonaAñadir_Click` de forma que muestre un diálogo modal de la clase `DlgDatosZona` y, una vez cerrado éste, inserte al final de la colección una zona con los datos introducidos (siempre y cuando el resultado del diálogo haya sido OK). Además,

y al igual que antes, se añadirá una nueva orden al menú "Zona" cuyo manejador será el método `zona_Click`. En este caso el texto de la nueva orden será el nombre de la zona horaria en cuestión.

También dentro de `zonaAñadir_Click`, añada el código necesario para impedir que se añada una zona si ya existe otra con el mismo nombre. La mejor forma de hacer esto es usar el método `Exists` de la plantilla `List<T>`, pasándole un predicado anónimo que verifique si existe en la colección el nombre de la zona que se desea añadir:

```
bool existe = m_Zonas.Exists(
                    delegate(ZonaHoraria zona)  // predicado anónimo
                    {
                       return zona.Nombre == dlgZona.Nombre;
                    }
                );
if (existe)
{
    MessageBox.Show("Ya existe una zona llamada "
                      + dlgZona.Nombre);
    return; // salimos sin modificar nada
}
```

Modifique el manejador `zonaEliminar_Click` de forma que, además de eliminar la última orden del menú "Zona", elimine el elemento correspondiente de la colección de zonas.

Compilar y ejecutar para ver el resultado.

Elimine el botón/orden y el manejador correspondiente utilizados de forma provisional para probar el formulario `DlgEliminarZona`. A continuación, añada a este formulario un atributo `m_Zonas` de tipo `List<ZonaHoraria>`.

Añada un manejador para el evento **Load** de `DlgEliminarZona` y, en él, asigne al atributo privado `m_Zonas` de este diálogo el valor de la propiedad `Zonas` del reloj digital (recuerde la funcionalidad que puede aportar la propiedad `Owner` de un diálogo), y rellene el control `lb_Zonas` con los nombres de todas las zonas almacenadas en la colección `m_Zonas` (use una instrucción `foreach`). Inicialmente, el primer elemento de la lista deberá aparecer seleccionado.

También deberá añadir el código necesario para que el resto de controles (diferencia horaria y signo) muestren los datos de la zona seleccionada. Para ello, maneje el evento **SelectedIndexChanged** del control `lb_Zonas` y use el siguiente código para recuperar una zona horaria de la colección, a partir de su nombre:

```
string nombreZona = ... // zona seleccionada

ZonaHoraria zonaSeleccionada =
        m_Zonas.Find(
               delegate(ZonaHoraria zona)  // predicado anónimo
               {
                  return zona.Nombre == nombreZona;
               }
         );
```

Compilar y ejecutar para ver el resultado.

Agregue a `DlgEliminarZona` la siguiente propiedad:

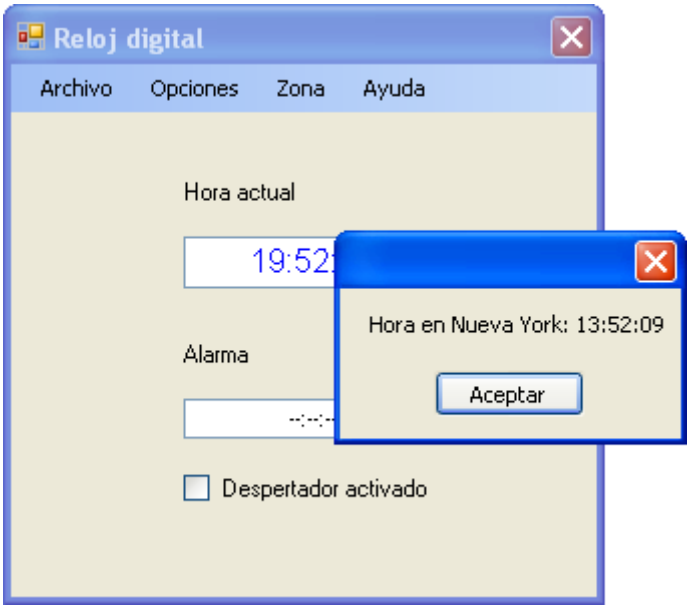
```
private int m_ZonaSeleccionada = -1;

public int ZonaSeleccionada
{
   get { return m_ZonaSeleccionada; }
}
```

Añada un manejador al botón "Aceptar" de `DlgEliminarZona`. Este manejador guardará en `m_ZonaSeleccionada` la posición de la zona que esté seleccionada en el control `lb_Zonas`.

En la clase `RelojDigital`, modifique el método `zonaEliminar_Click` para que, en vez de eliminar siempre la última zona, se muestre un diálogo modal de tipo `DlgEliminarZona` y se elimine la zona seleccionada por el usuario en dicho diálogo.

Compilar y ejecutar para ver el resultado.

Actualmente, cuando el usuario selecciona una zona horaria en el menú "Zona" se muestra una caja de mensaje con el nombre de la zona. Modifique el código que crea este mensaje para que además del nombre de la zona, incluya la hora en dicha zona (se aconseja utilizar el método `Find` de la colección; ver la figura siguiente).



8.2. Seriación

8.2.1 Guardar los datos de las zonas horarias en un fichero.

Añada al comienzo de "RelojDigital.cs" las siguientes directrices *using*, que hacen visibles los elementos de varios espacios de nombres involucrados en el proceso de seriación:

```
using System.IO;
using System.Runtime.Serialization;
using System.Runtime.Serialization.Formatters.Binary;
```

Modifique la clase `ZonaHoraria` añadiendo el atributo **[Serializable]** antes de la declaración de la clase.

Añada un manejador para el evento **FormClosing** de `RelojDigital`, y modifíquelo como se muestra a continuación:

```
private void RelojDigital_FormClosing(object sender,
                                      FormClosingEventArgs e)
{
   try
   {
      using (Stream stream =
                  File.Open("reloj.dat",FileMode.Create))
      {
         BinaryFormatter bin = new BinaryFormatter();
         bin.Serialize(stream, m_Zonas);
      }
   }
   catch(Exception ex)
   {
      Console.WriteLine("No se pudo crear el archivo de zonas. "
                        + " Causa: " + ex.Message);
   }
}
```

Compilar y ejecutar para ver el resultado.

8.2.2 Obtener de un fichero los datos de las zonas horarias

Añada un método que responda al evento **Load** de `RelojDigital`. En éste método se leerá el fichero, y se cargará la matriz `m_Zonas`. Contemple la posibilidad de que el archivo no se pueda abrir.

Compilar y ejecutar para ver el resultado.

8.2.3 Añadir al menú "Zona" los elementos recuperados del fichero

El código anterior se limita a cargar en `m_Zonas` las zonas horarias previamente almacenadas. Modifique el manejador `RelojDigital_Load` para que actualice también el menú "Zona". Probablemente tendrá que reutilizar parte del código que añadía una zona al menú en el método `menúZona_Añadir`. Para ello, seleccione el código que necesita duplicar y utilice la herramienta *Refactorizar* > *Extraer método* para moverlo a un nuevo método llamado `AñadirOrdenZona`. Llame a este método tanto desde `menúZona_Añadir` como desde `RelojDigital_Load`.

Cuando haya finalizado, su método `RelojDigital_Load` deberá tener el siguiente aspecto:

```
private void RelojDigital_Load(object sender, EventArgs e)
{
   try
   {
      using (Stream stream = ...)
      {
         BinaryFormatter bin = new BinaryFormatter();
         m_Zonas = ...; // Deseriamos

         foreach (ZonaHoraria nuevaZona in m_Zonas)
            AñadirOrdenZona(nuevaZona);
      }
   }
   catch(Exception ex)
   {
      // Mensaje de error a la consola
   }
}
```

Compilar y ejecutar para ver el resultado.