MEMORIA WPF

2021-2022

Interfaces Gráficas de Usuario

VNiVERSiDAD D SALAMANCA

CAMPUS DE EXCELENCIA INTERNACIONAL

Fiz Rey Armesto

1 - ÍNDICE

2 - Manual de usuario

La aplicación sirve para crear tablas que guardarán la información de la ingesta de calorías que el usuario especifique.

La ventana principal del programa está formada por el menú desplegable Tablas, 3 botones y un lienzo donde se muestran las gráficas. Los 3 botones que no se utilizarán hasta que se tenga abierta la ventana Tablas a la que se accede desde el menú Ver tabla.

El menú desplegable llamado “Tablas” tiene las opciones:

- **Nueva tabla:** crea una nueva tabla (eliminando los datos que no se hayan guardado durante la sesión)
- **Ver tabla:** abre una ventana secundaria que nos permite introducir y modificar datos de la tabla que se esté utilizando en la sesión
- **Guardar tabla:** permite guardar el archivo que se ha seleccionado previamente en la sesión con la opción “Guardar tabla como…”.
- **Guardar tabla como…:** Abre una ventana nueva que nos permite elegir un archivo donde guardar la tabla actual (también se pueden sobreescribir archivos)
- **Cargar tabla:** Abre una ventana nueva que nos permite elegir un archivo a cargar para utilizar en la sesión actual (eliminando los datos que no se hayan guardado durante la sesión).

El botón central Ver Gráfica Día nos mostrará la gráfica secundaria del día seleccionado. Los botones de la derecha Izq y Dch permiten mover los dos tipos de gráfica hacia el lado que indican (modificando los Días que se muestran en la gráfica o las diferentes comidas dependiendo del gráfico).

La ventana Tablas a la que se accede desde el menú desplegable Ver tabla está formada principalmente por dos listas: una lista de días y otra lista de comidas que muestran su contenido.

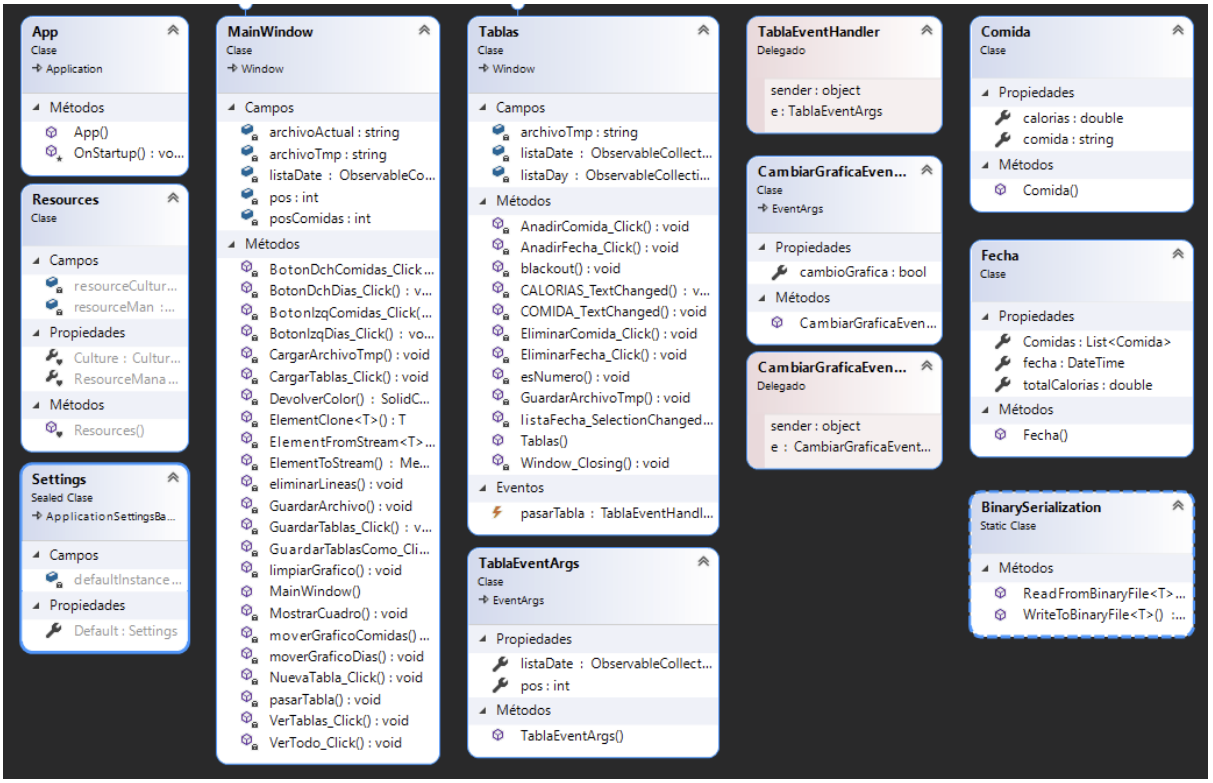
Se pueden añadir días a la lista seleccionando el día en el calendario de la parte superior izquierda y pulsando el botón ”Añadir fecha” de la parte superior derecha. Cuando haya fechas en la lista se podrán clickar y aparecerá su gráfica de días en la ventana principal y su lista de comidas en la lista inferior de la ventana Tablas. Para modificar la lista de comedias de un día se tiene que tener seleccionado el día en la lista superior y después introducir los datos de la comida en las cajas de texto del centro de la ventana, luego de introducir los datos se pulsa el botón añadir comida y ya aparecerá tanto en la lista como en la gráfica. Para eliminar datos de las listas basta con hacer click derecho en el dato que queramos eliminar y seleccionar la opción del menú desplegable eliminar.

La ventana Tablas envía toda la información modificada aunque se cierre la pestaña.

La aplicación tiene un sistema de autoguardado que evita pérdidas de información. El archivo de autoguardado se encuentra en la carpeta \\saves del directorio raíz del programa. Por defecto el programa cargará el archivo de autoguardado al inicio, el autoguardado se produce cada vez que se modifica la tabla. El autoguardado no modifica los archivos que el

usuario elige con Guardar tabla como… o Guardar tabla, solo modifica su propio archivo de autoguardado.

3 - Manual del programador



Partiendo del siguiente diagrama de objetos tenemos los siguientes archivos y clases:

3.1 - MainWindow.xaml.cs:

```
public partial class MainWindow : Window
```

MainWindow es la única clase de este archivo que va a tener todos los métodos descritos a continuación.
Tenemos las siguientes variables globales:

- ```
  ObservableCollection<Fecha> listaDate;
  ```
- ```
  string archivoTmp, archivoActual;
  ```
- ```
  int pos, posComidas;
  ```

- listaDate: Colección de tipo fecha que almacenará los valores que se introduzca sobre las calorías.
- archivoTmp y archivoActual: direcciones de los archivos de autoguardado y del último archivo guardado manualmente por el usuario respectivamente.
- pos y posComidas: posición de la colección en la que se encuentran la fecha y las comidas de cada fecha respectivamente.
  ```

Dentro de la ventana principal tenemos estas funciones:

```
public MainWindow()
```

Inicia el componente, comprueba si hay un directorio \\saves, si no existe lo crea. Después comprueba si hay un archivo temporal y lo carga con el método CargarArchivoTmp(), si no existe inicializa la variable listaDate, y finalmente llama al método limpiarGráfico()

```
private void VerTablas_Click(object sender, RoutedEventArgs e)
```

Inicializa una nueva variable Tablas con el valor de listaDate y la abre, indicando las función delegada pasarTabla

```
private void pasarTabla(object sender, TablaEventArgs e)
```

Función delegada que se encarga de mostrar los gráficos correctos obteniendo la información actualizada de la lista. Primero se actualiza posComidas a 0 y se calculan los valores numComidas (número de Comidas que hay que mostrar en el gráfico) y numFechas (número de fechas que hay que mostrar en el gráfico).
Después se hacen visibles los Canvas de Comida y los botones para mover los canvas de comida también se hacen visibles y se activan. Se llama al método eliminarLineas() y se inicia un bucle que va a cambiar la visibilidad de los nombre y las líneas del gráfico que se necesitan y va a almacenar el número de calorías en un array de double calorías[8] para después calcular el máximo.
Después se calculan los valores que van a tener los puntos de la gráfica de calorías y se hace otro bucle para dar visibilidad a las líneas que forman el gráfico y crear su animación en función de sus calorías con las calorías máximas que se han calculado antes.

Para la parte de la gráfica de días se hace lo mismo hasta llegar a la modificación de las líneas de la gráfica, se tiene que recorrer otro bucle en el interior ya que se crea una línea del gráfico por cada comida que se tiene almacenada en el día. Dentro del bucle se tiene que diferenciar entre la primera ejecución (que modifica la línea del gráfico correspondiente que se tiene en el xaml) y el resto de iteraciones que copian la línea de xaml anteriormente mencionada. Después de clonar la línea se le añade al Uid la cadena “aBorrar” para identificar la línea cuando se quiera eliminar y se le devuelve un color con el método DevolverColor() y finalmente se añade al canvas con canvas.Children.Add()

```
private void moverGraficoComidas(bool direccion)
```

Función análoga a la parte de Comidas de la función pasarTabla quitando que al comienzo no se modifican las visibilidades del canvas ni de los botones sino que se modifica el valor del índice de listaDate comprobando que no sobrepase los límites.

```
private void moverGraficoDias(bool direccion)
```

Función análoga a la parte de Días de la función pasarTabla quitando que al comienzo no se modifican las visibilidades del canvas ni de los botones sino que se modifica el valor del índice de listaDate comprobando que no sobrepase los límites.

```
private void NuevaTabla_Click(object sender, RoutedEventArgs e)
```

Método que sirve para crear otra tabla nueva, se ejecuta al pulsar el menú desplegable Nueva Tabla primero avisa al usuario con un MessageBox de que perderá los cambios no guardados y después comprueba si la ventana Tablas está abierta, en caso de estarlo la

cierra y llama al método limpiarGráfico(), inicializa listaDate y sobreescribe el contenido de archivoActual.

```
private void limpiarGrafico()
```

Método que sirve para borrar las líneas del gráfico. Primero se llama al método eliminarLineas() y a continuación se realiza un bucle que pone la visibilidad de los apartados que pueden cambiar del gráfico (todos menos los guiones) a Hidden.

```
private void CargarTablas_Click(object sender, RoutedEventArgs e)
```

Método que sirve para cargar una tabla, se ejecuta al pulsar el menú desplegable Cargar Tablas. Primero avisa al usuario con un MessageBox de que perderá los cambios no guardados y después comprueba si la ventana Tablas está abierta, en caso de estarlo la cierra y llama al método limpiarGráfico(), inicializa listaDate y sobreescribe el contenido de archivoActual.

```
private void GuardarTablas_Click(object sender, RoutedEventArgs e)
```

Método que sirve para guardar una tabla, se ejecuta al pulsar el menú desplegable Guardar Tablas. Se comprueba que se ha ejecutado el método GuardarTablasComo, sino se ha ejecutado se muestra una ventana de error y si se ha ejecutado se guarda y se muestra un mensaje de éxito.

```
private void GuardarTablasComo_Click(object sender, RoutedEventArgs e)
```

Método que sirve para guardar una tabla, se ejecuta al pulsar el menú desplegable Guardar Tablas como.... Primero crear un diálogo SaveFileDialog(). Si se elige un archivo adecuadamente se guarda y se muestra una ventana de éxito, sino muestra un mensaje de fallo.

```
private void VerTodo_Click(object sender, RoutedEventArgs e)
```

Método que sirve para cambiar la visibilidad entre canvas de gráficos.Se ejecuta al pulsar el botón Ver Gráfica Día. Activa y pone como visibles los elementos relacionados con el gráfico Días y oculta los del gráfico de Comidas.

```
private void GuardarArchivo(String s)
```

Llama al método BinarySerialization.WriteToBinaryFile(s, new List<Fecha>(listaDate)). Como parámetro tiene un string que será la direccion del archivo que queramos guardar, este parámetro se pasa como 1er parámetro de la función indicada anteriormente, el segundo parámetro de esta función será una nueva Lista de tipo Fecha que haremos con el valor de listaDate actual.

```
private void BotonIzqComidas_Click(object sender, RoutedEventArgs e)
```

Método que se ejecuta al pulsar el botón BotonIzqComidas. Llama al método moverGraficoComidas(false) con false como parámetro ya que false indica un movimiento hacia la Izq.

```
private void BotonDchComidas_Click(object sender, RoutedEventArgs e)
```

Método que se ejecuta al pulsar el botón BotonDchComidas. Llama al método moverGraficoComidas(true) con false como parámetro ya que true indica un movimiento hacia la Dch.

```
private void BotonIzqDias_Click(object sender, RoutedEventArgs e)
```

Método que se ejecuta al pulsar el botón BotonIzqDias. Llama al método moverGraficoDias(false) con false como parámetro ya que false indica un movimiento hacia la Izq.

```
private void BotonDchDias_Click(object sender, RoutedEventArgs e)
```

Método que se ejecuta al pulsar el botón BotonDchDias. Llama al método moverGraficoDias(true) con false como parámetro ya que true indica un movimiento hacia la Izq.

```
private void CargarArchivoTmp(string s)
```

Inicializa lista date a una nueva ObservableCollection con el valor devuelto por BinarySerialization.ReadFromBinaryFile<List<Fecha>>(s) siendo s la string del parámetro de la función que será la dirección del archivo que queramos guardar

```
private void MostrarCuadro(string msg, string titulo)
```

Función que simplifica el uso de Cuadros de MessageBox

```
private SolidColorBrush DevolverColor(int s)
```

Función que devuelve colores dependiendo del número que se le pase. Se usa en la creación de lineas para no repetir colores.

```
private static T ElementClone<T>(T element)
```

Clona un elemento. Este método y los 2 siguientes se usan para clonar lineas ya especificadas en xaml con todas sus propiedades.

```
private static MemoryStream ElementToStream(object element)
```

Guarda un elemento como MemoryStream

```
private static T ElementFromStream<T>(MemoryStream elementAsStream)
```

Rehace un elemento desde un Memory Stream

```
private void eliminarLineas()
```

Busca lineas dentro del Canvas CanvasDias en las que su Uid contenga la cadena aBorrar, la cual se ha especificado en la creación de estas. Si encuentra lineas con esa cadena, elimina las lineas del canvas.

3.2- Tablas.xaml.cs:

```
public class TablaEventArgs : EventArgs
```

Es una clase de la ventana no modal tablas que deriva de EventArgs, se utiliza para pasar la lista con las modificaciones que hace el usuario, también se pasa un int que corresponde al índice actual de la lista.

```
public ObservableCollection<Fecha> listaDate {get; set; }
```

Definición de las propiedades get-set para listaDate y pos

```
public int pos { get; set; }
```

```
public TablaEventArgs(ObservableCollection<Fecha> e, int p)
```

Asigna el valor de las variables e y p a listaDate y pos (que se han mencionado anteriormente) respectivamente.

```
public delegate void TablaEventHandler(Object sender, TablaEventArgs e);
```

Delegado para pasar la clase derivada de EventArgs especificada anteriormente, se usa para pasar la tabla actualizada a la ventana principal, siendo this el primer parámetro y una nueva TablaEventArgs el segundo parámetro.

```
public partial class Tablas : Window
```

Tenemos las variables globales:

```
ObservableCollection<Fecha> listaDate;
ObservableCollection<Comida> listaDay;
public event TablaEventHandler pasarTabla;
string archivoTmp;
```

- listaDate: Colección de la clase Fecha que se mostrará en la en la parte superior de la ventana
- listaDay: Colección de la clase Comida que se mostrará en la en la parte superior de la ventana
- pasarTabla: evento del tipo TablaEventHandler pasarTabla
- archivoTmp: dirección del archivo de autoguardado.

```
public Tablas(ObservableCollection<Fecha> l)
```

Inicializa los componentes y se comprueba el directorio de archivo temporal, se crea si no existe y se modifica la variable archivoTmp.

```
private void blackout()
```

Añade todas las fechas de listaDate al blackOut del datePicker, así no se pueden seleccionar al calendario.

```
private void AnadirFecha_Click(object sender, RoutedEventArgs e)
```

Método que se ejecuta al pulsar el botón AnadirFecha. Comprueba si el datePicker tiene alguna fecha seleccionada, si no tiene devuelve un mensaje de error, si tiene fecha seleccionada comprueba que la fecha no exista en la lista y devuelve error en ese caso. Si no existe la fecha se añade,ordena y se guarda la lista con GuardarArchivoTmp();

```
private void listaFecha_SelectionChanged(object sender,
SelectionChangedEventArgs e)
```

Método que se ejecuta cuando se cambia la selección de la lista listaFecha. Se cambia el ItemSource de la listaDia creando una nueva listaDay con el valor de Fecha.Comida de la listaFecha seleccionada. Finalmente se ejecuta la función pasarTabla con los parámetros this y un nuevo TablaEventArgs con listaDate y el índice de la fecha seleccionada.

```
private void AnadirComida_Click(object sender, RoutedEventArgs e)
```

Método que se ejecuta cuando se pulsa el botón AnadirComida. Comprueba que los recuadros de texto tengan algo escrito y se crea una nueva variable comida con los valores

del recuadro, se añaden a la lista, se aumenta el valor de totalCalorias sumandole las calorias de la variable recien creada comida y se ejecuta el método pasarTabla de la misma forma que hemos hecho anteriormente.

```
private void EliminarFecha_Click(object sender, RoutedEventArgs e)
```

Método que se ejecuta cuando se pulsa el botón del menú desplegable Eliminar con click derecho sobre la fecha a eliminar. Borra la fecha de la lista y ejecuta el método guardarArchivoTmp();

```
private void EliminarComida_Click(object sender, RoutedEventArgs e)
```

Análogo al método anterior solo que cambiando por Días por Comida. Además hace una comprobación para ver que no se haya borrado la fecha de la lista anteriormente y resta las calorías totales de la fecha.

```
private void GuardarArchivoTmp()
```

Método ya visto en MainWindow.xaml.cs

```
private void Window_Closing(object sender,
System.ComponentModel.CancelEventArgs e)
```

Método que se ejecuta al cerrar la ventana. Ejecuta el método pasarTabla como hemos visto anteriormente solo que con el índice (pos) a 0.

```
private void esNumero(object sender, TextCompositionEventArgs e)
```

Método que se ejecuta cuando se introduce texto en el TextBox de calorías. Comprueba con un regex que solo se introduzcan números.

```
private void COMIDA_TextChanged(object sender, TextChangedEventArgs e)
```

Método que se ejecuta cuando se introduce texto en el TextBox de comida. Comprueba si el tamaño del texto introducido es mayor que 1 para cambiar la visibilidad de la pista del TextBox

```
private void CALORIAS_TextChanged(object sender, TextChangedEventArgs e)
```

Método que se ejecuta cuando se introduce texto en el TextBox de calorías. Comprueba si el tamaño del texto introducido es mayor que 1 para cambiar la visibilidad de la pista del TextBox

3.3 - Comida.cs:

```
public class Comida
```

Clase comida, almacena los valores de cada comida: una string con el nombre de la comida y un double con el número de calorías.

```
  public string comida { get; set; }


public double calorias { get; set; }


public class Fecha
```

Clase Fecha, almacena todos los datos referentes a un día: DateTime fecha, un double totalCalorias y una lista de Objetos Comida Comidas.

```
public DateTime fecha { set; get; }


public double totalCalorias { set; get; }


public List<Comida> Comidas { set; get; }


public Fecha(DateTime fecha)
```

totalCalorias se inicializa a 0 en el constructor y Comidas se inicializa creando una nueva lista, fecha se obtiene del parámetro del constructor.

3.4 - BinarySerialization.cs:

```
public static class BinarySerialization
```

Clase con los métodos que nos permiten guardar y cargar en disco las listas Serializables con los datos de las tablas.

```
public static void WriteToBinaryFile<T>(string filePath, T
objectToWrite, bool append = false)
```

Escribe a un archivo binario del cual obtiene su dirección por el primer parámetro. Se va a usar un archivo .bin para el autoguardado y el resto de archivos que guarda el usuario se guardan con formato .cal aunque el usuario puede especificar su archivo como el quiera, esto no va a impedir el buen funcionamiento del programa.

```
public static T ReadFromBinaryFile<T>(string filePath)
```

Carga un archivo binario de disco de la dirección especificada por el parámetro.

4 - Referencias

Referencias a 09/02/2022:

Apuntes, diapositivas, clases.

https://stackoverflow.com/questions/31658655/wpf-listview-vertical-scrollbar
https://stackoverflow.com/questions/4351876/c-sharp-list-of-objects-how-do-i-get-the-sum-of-a-property
https://stackoverflow.com/questions/7454024/setting-culture-en-in-globally-in-wpf-application
https://docs.microsoft.com/es-es/dotnet/api/system.windows.forms.savefiledialog?view=windowsdesktop-6.0
https://wpf-tutorial.com/common-interface-controls/contextmenu/
https://blog.danskingdom.com/saving-and-loading-a-c-objects-data-to-an-xml-json-or-binary-file/
https://stackoverflow.com/questions/16920027/looping-over-xaml-defined-labels
https://stackoverflow.com/questions/60135632/wpf-menu-item-not-clickable-with-windowstyle-none-with-windowchrome?noredirect=1&lq=1
https://stackoverflow.com/questions/4280176/wpf-is-there-a-way-to-remove-specific-child-from-canvas-children
https://stackoverflow.com/questions/36803433/how-to-clone-canvas-in-wpf
https://stackoverflow.com/questions/1268552/how-do-i-get-a-textbox-to-only-accept-numeric-input-in-wpf
https://stackoverflow.com/questions/5606887/c-sharp-orderby-over-list-of-objects-using-an-int-attribute-of-object
https://stackoverflow.com/questions/4937060/how-to-check-if-listt-element-contains-an-item-with-a-particular-property-valu
https://stackoverflow.com/questions/7717222/handle-a-wpf-exit-event
https://docs.microsoft.com/en-us/dotnet/desktop/wpf/app-development/how-to-get-all-windows-in-an-application?view=netframeworkdesktop-4.8
https://qastack.mx/programming/7425618/how-can-i-add-a-hint-text-to-wpf-textbox
https://stackoverflow.com/questions/68168048/how-can-i-add-specific-blackout-dates-in-datepicker-c-sharp-wpf
https://docs.microsoft.com/es-es/visualstudio/ide/class-designer/how-to-add-class-diagrams-to-projects?view=vs-2022
https://workspace.google.com/marketplace/app/code_blocks/100740430168