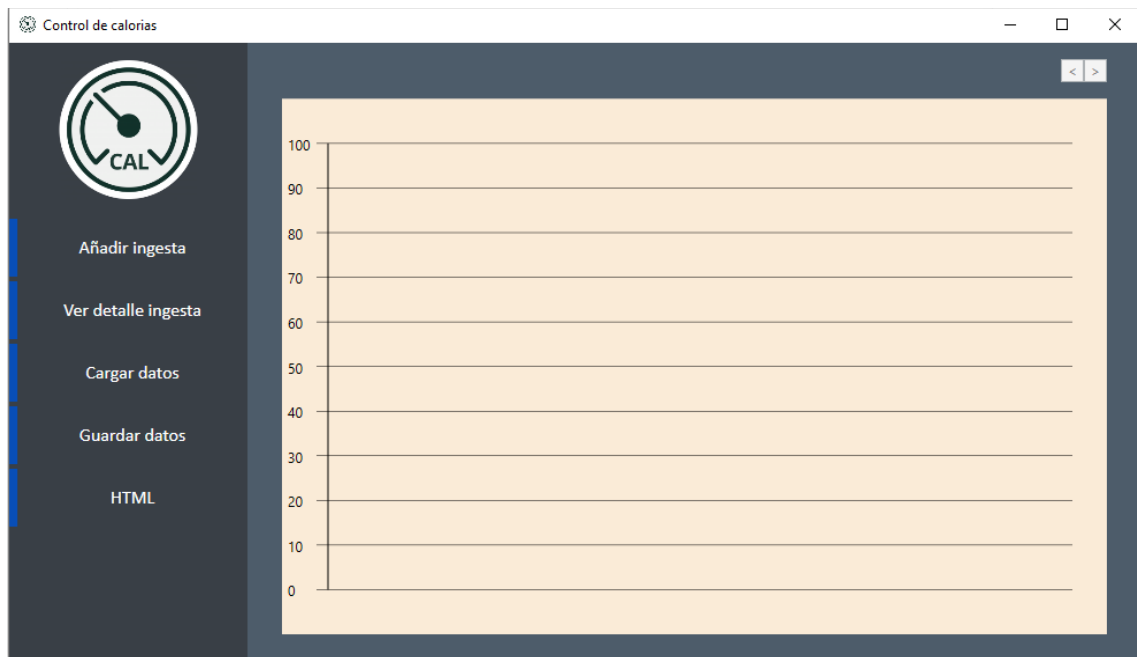


Memoria Ejercicio Práctico

Manual de usuario

Ventana principal

Al ejecutar la aplicación, aparecerá un ventana principal con la siguiente apariencia:



En el menú de la izquierda, hay una serie de botones los cuales realizan diferentes funcionalidades:

- El botón Añadir ingesta abre una nueva ventana en la que se presenta una interfaz para introducir datos de nuevas ingestas.
- El botón Ver detalle Ingesta abre una nueva ventana que permite ver con mayor detalle los datos de las distintas ingestas almacenadas.
- El botón Cargar datos permite cargar datos de ingestas almacenados en un fichero de texto.
- El botón Guardar datos permite guardar los datos de las ingesta de la aplicación en un fichero de texto.
- El botón HTML genera un archivo HTML que contiene una tabla con la información detallada de cada ingesta

La funcionalidad de estos botones se explicará con mayor detalle en las siguientes páginas.

Agregar ingesta

Cuando el usuario pulse el botón Añadir ingesta aparecerá una ventana como esta:

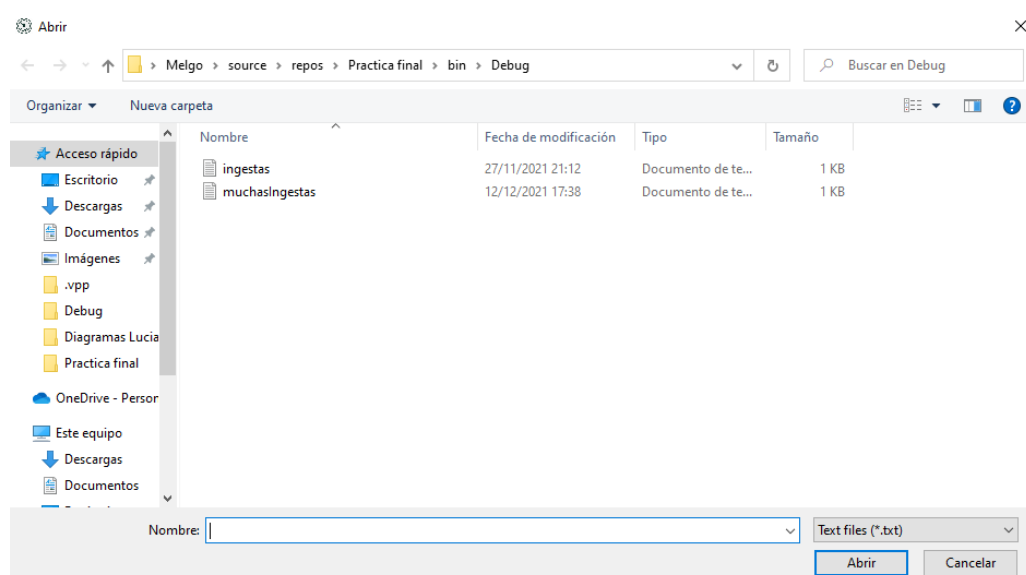
The screenshot shows a window titled 'Ingesta' with a close button (X) in the top right corner. The window is divided into two main sections. The left section, titled 'Introduccion de la fecha de la ingesta', has a blue background and contains a text input field labeled 'Seleccione una fecha' with the value '15' and a calendar widget for December 2021. The right section, titled 'Introduccion de las calorías de la ingesta', has a dark gray background and contains six text input fields labeled 'Calorias desayuno', 'Calorias aperitivo', 'Calorias comida', 'Calorias merienda', 'Calorias cena', and 'Calorias otros'. At the bottom of the right section are two buttons: 'Cancelar' and 'Aceptar'.

Por una parte, hay un campo donde se va a introducir la fecha de la ingesta y un calendario que va a facilitar la introducción de esa fecha (basta con hacer click en la fecha deseada en el calendario).

Por otra parte, hay una serie de campos donde se van a introducir las calorías de cada una de las ingesta del día.

Cargar datos

Cuando el usuario pulse el botón Cargar datos aparecerá una ventana como esta:



Esta ventana va a permitir al usuario explorar en sus archivos y seleccionar un fichero de texto con los datos de las ingestas que quiere cargar. Los ficheros de texto deberán cumplir el siguiente formato:

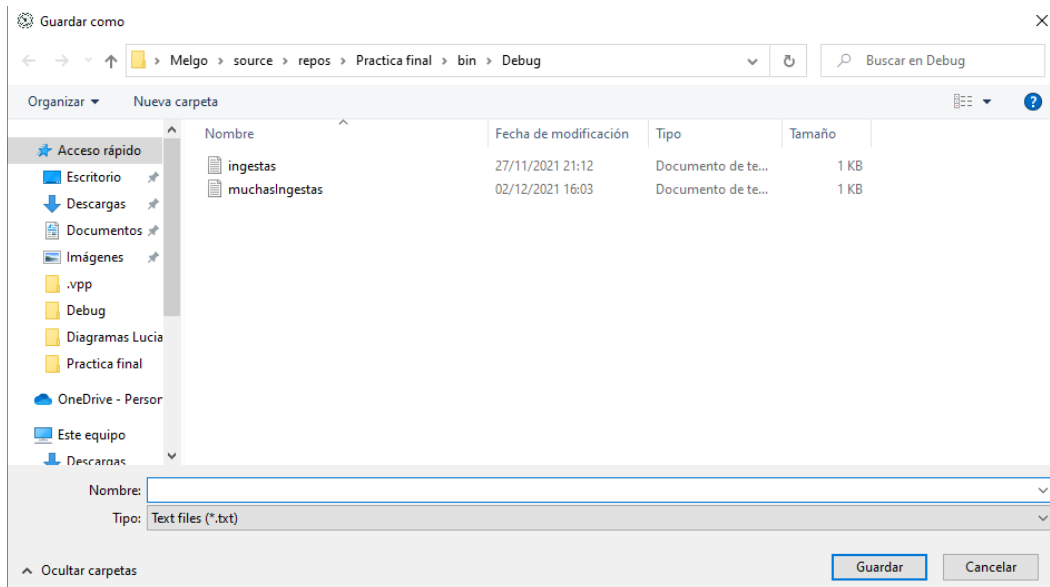
- En primer lugar, tiene que aparecer la fecha de la ingesta.
- En segundo lugar, deberán aparecer las calorías de las distintas ingestas separados por un espacio en blanco.
- Por último, en cada línea del fichero deben aparecer los datos de una única ingesta.

Ejemplo:

```
12/12/2001 100 200 300 400 500 600
01/10/1959 600 500 400 300 200 100
25/05/1967 200 300 400 500 600 700
```

Guardar datos

Cuando el usuario pulse el botón Guardar datos aparecerá una ventana como esta:

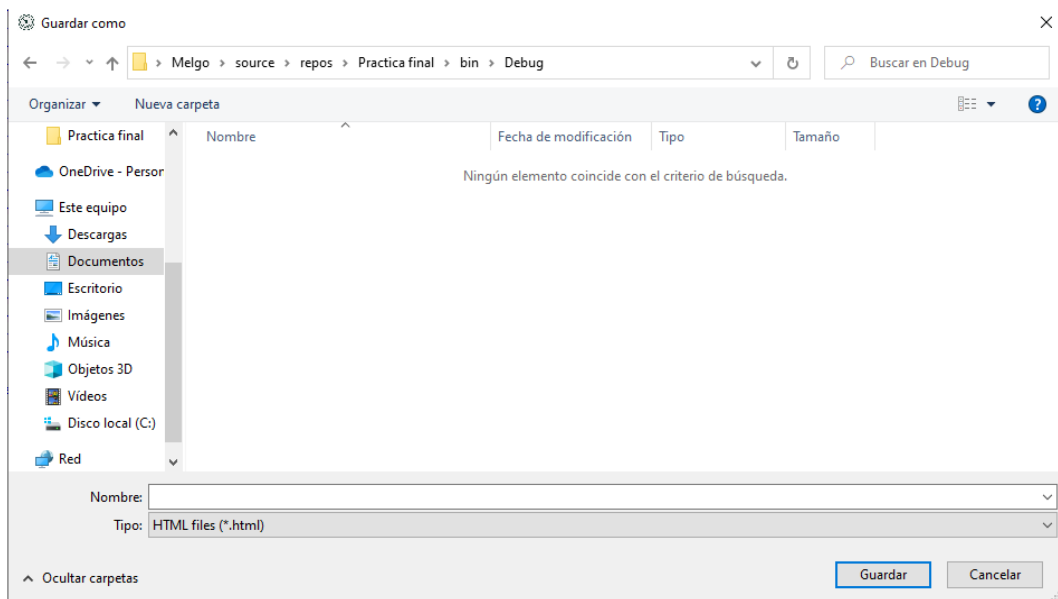


Esta ventana va a permitir al usuario explorar en sus archivos y crear un fichero de texto en el que quiere guardar los datos de las ingestas. Este fichero va a tener el mismo formato que los ficheros de texto de Cargar Datos.

El formato será el mismo porque es posible que un usuario quiera guardar una colección de ingestas y más adelante quiera recuperar esa colección de ingestas para continuar editando la colección de ingesta tal y como la dejó.

HTML

Cuando el usuario pulse el botón HTML aparecerá una ventana como esta:



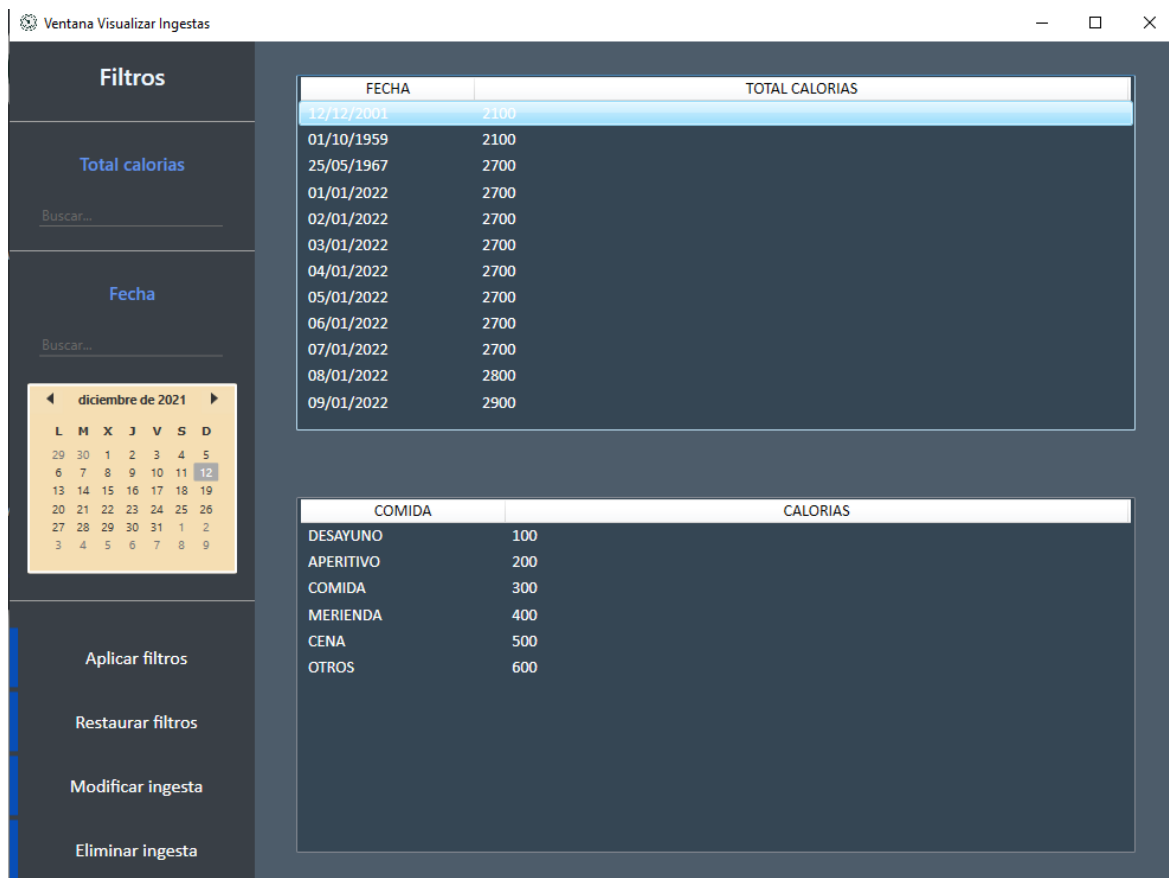
Esta ventana va a permitir al usuario explorar en sus archivos y crear un archivo HTML el cual va a contener una tabla con los datos de las ingestas almacenadas en la aplicación. El archivo HTML tendrá una apariencia similar a esta:

Registro ingestas

FECHA	DESAYUNO	APERITIVO	COMIDA	MERIENDA	CENA	OTROS	TOTAL CALORIAS
12/12/2001	100	200	300	400	500	600	2100
01/10/1959	600	500	400	300	200	100	2100
25/05/1967	200	300	400	500	600	700	2700
01/01/2022	200	300	400	500	600	700	2700
02/01/2022	200	300	400	500	600	700	2700
03/01/2022	200	300	400	500	600	700	2700
04/01/2022	200	300	400	500	600	700	2700
05/01/2022	200	300	400	500	600	700	2700
06/01/2022	200	300	400	500	600	700	2700
07/01/2022	200	300	400	500	600	700	2700
08/01/2022	200	300	400	500	600	800	2800
09/01/2022	200	300	400	500	600	900	2900

Ver detalle ingesta

Cuando el usuario pulse el botón Añadir ingesta aparecerá una ventana como esta:



Ventana Visualizar Ingestas

Filtros

Total calorías

Buscar...

Fecha

Buscar...

diciembre de 2021

FECHA	TOTAL CALORIAS
12/12/2001	2100
01/10/1959	2100
25/05/1967	2700
01/01/2022	2700
02/01/2022	2700
03/01/2022	2700
04/01/2022	2700
05/01/2022	2700
06/01/2022	2700
07/01/2022	2700
08/01/2022	2800
09/01/2022	2900

Aplicar filtros

Restaurar filtros

Modificar ingesta

Eliminar ingesta

COMIDA	CALORIAS
DESAYUNO	100
APERITIVO	200
COMIDA	300
MERIENDA	400
CENA	500
OTROS	600

En la primera tabla aparece un listado con la fecha y el total de calorías de todas las ingestas almacenadas en la aplicación. Cuando se seleccione una ingesta (click en la ingesta en la primera tabla, como se ve en la imagen), en la segunda tabla aparecerá un listado con los detalles de esa ingesta seleccionada, con las distintas comidas de la ingesta y las calorías correspondientes.

En el menú de la izquierda, hay una serie de botones los cuales realizan diferentes funcionalidades:

- El botón Aplicar filtros actualiza la tabla de forma que solo aparezcan las ingestas que cumplen con los filtros introducidos por el usuario.
- El botón Restaurar filtros deshace los filtros aplicados por el usuario de forma que en la tabla aparecen de nuevo todas las ingestas almacenadas en la aplicación.
- El botón Modificar ingesta permite modificar los datos de la ingesta seleccionada. Es imprescindible seleccionar una ingesta ya que si no se selecciona ninguna este botón no aparecerá.
- El botón Eliminar ingesta permite eliminar los datos de la ingesta seleccionada. Es imprescindible seleccionar una ingesta ya que si no se selecciona ninguna este botón no aparecerá.

Aplicar filtros

Antes de que el usuario pulse el botón Aplicar filtros, el usuario introducirá en los campos los filtros que desea aplicar.

El usuario tiene la posibilidad de filtrar las ingestas en función del total de calorías de las ingestas, en función de la fecha de la ingesta o en función de las dos.

Una vez el usuario haya introducido los filtros correspondientes, pulsará el botón Aplicar filtros y la primera tabla se actualizará apareciendo solamente las ingestas que cumplen con los filtros.

Restaurar filtros

Cuando el usuario pulse el botón Restaurar filtros, se borrarán los filtros introducidos y se actualizará la primera tabla de forma que volverán a aparecer los datos de todas las ingestas que almacena la aplicación.

Modificar ingesta

Cuando el usuario pulse el botón Modificar ingesta aparecerá una ventana como esta:

Modificar Ingesta

Introduccion de la fecha de la ingesta

12/12/2001 15

diciembre de 2001

L	M	X	J	V	S	D
26	27	28	29	30	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

Introduccion de las calorías de la ingesta

100
200
300
400
500
600

Cancelar Aceptar

Esta ventana es similar a la ventana donde se introducen los datos de las ingestas, con la diferencia de que en esta ventana, se van a cargar automáticamente los datos de la ingesta que se va a modificar.

Es imprescindible seleccionar una ingesta ya que si no se selecciona ninguna este botón no aparecerá.

Eliminar ingesta

Cuando el usuario pulse el botón Eliminar ingesta, se eliminará la ingesta de los datos de la aplicación y se actualizará la primera tabla de forma que la ingesta eliminada ya no aparecerá en la lista.

Es imprescindible seleccionar una ingesta ya que si no se selecciona ninguna este botón no aparecerá.

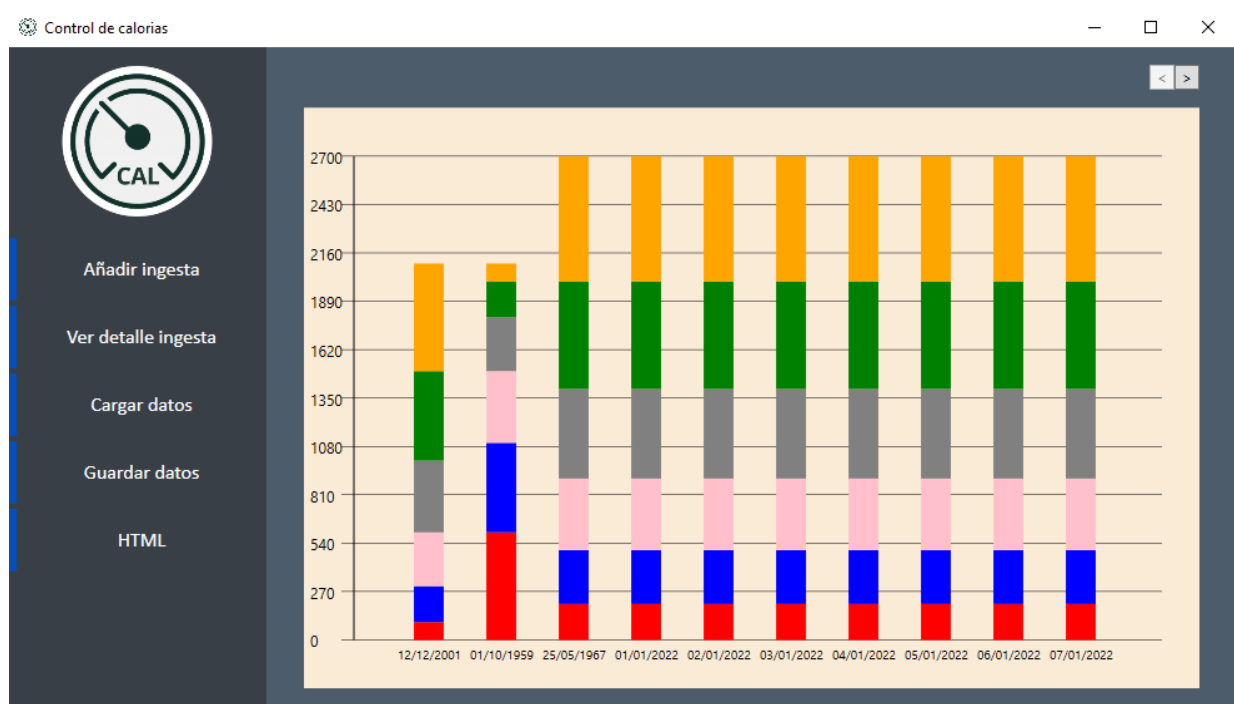
Visualización de gráficos en la ventana principal

Gráfico de barras global

Cuando el usuario haya agregado una ingesta a través de la ventana de Añadir Ingesta o cargando los datos de un fichero de texto, se dibujará en la ventana principal un gráfico de barras con los datos de todas las ingestas almacenadas en la aplicación. Este gráfico va a tener una barra por cada ingesta almacenada. A su vez cada barra va a estar dividida en tantos segmentos como comidas estén consideradas (desayuno, comida, ...).

En la ventana principal, solo hay hueco para representar el gráfico de barras de 10 ingestas. De esta manera, en el caso de que la aplicación almacene más de 10 ingestas, para ver el resto de ingestas que no se han podido representar, existen un par de botones que nos permiten ir a la derecha para ver las siguientes ingestas e ir a la izquierda para ver las ingestas anteriores. Cuando no esté permitido ir a la izquierda o ir a la derecha estos botones estarán desactivados.

La apariencia del gráfico de barras global es el siguiente:

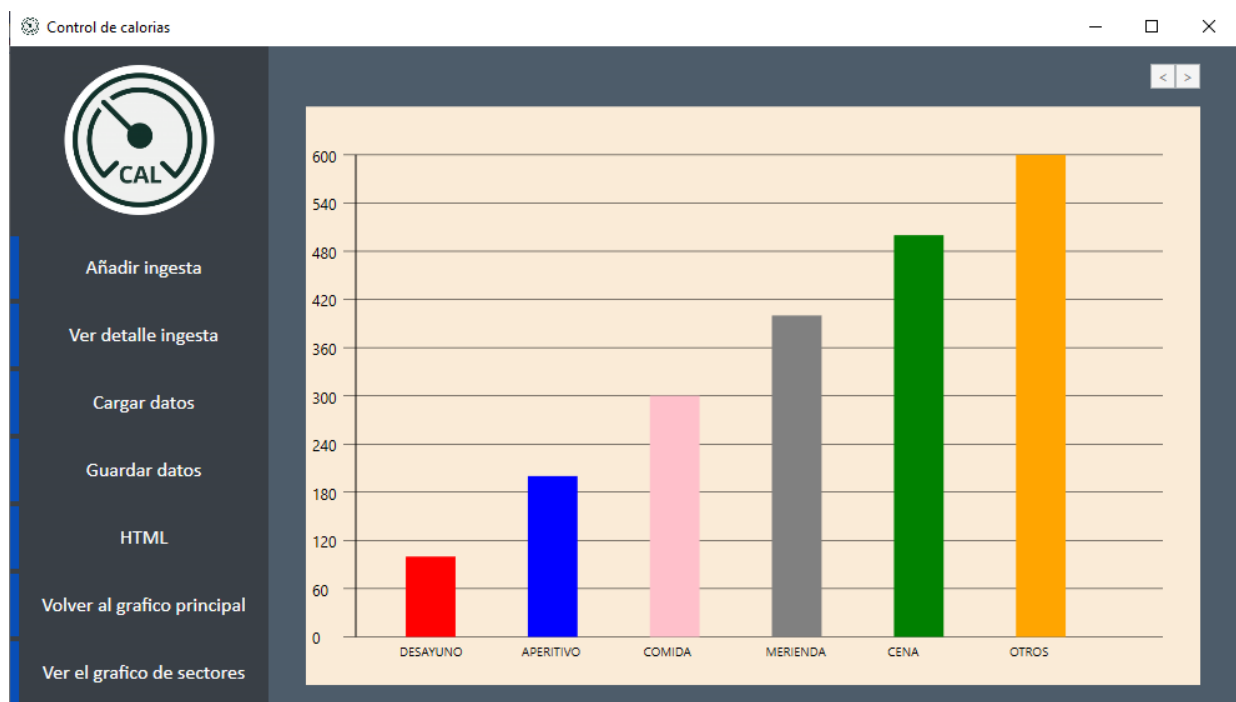


En este caso, como se representa el gráfico de barras de las 10 primeras ingestas, se permite ir a la derecha para ver las siguientes ingestas pero no se permite ir a la izquierda.

Gráfico de barras de una ingesta

Cuando el usuario seleccione un ingesta en la ventana que muestra los detalles de la ingesta, en la ventana principal se dibujara un gráfico de barras con los detalles de esa ingesta. Este gráfico va a tener una barra por cada comida que tenga la ingesta.

La apariencia del gráfico de barras del detalle de una ingesta es el siguiente:

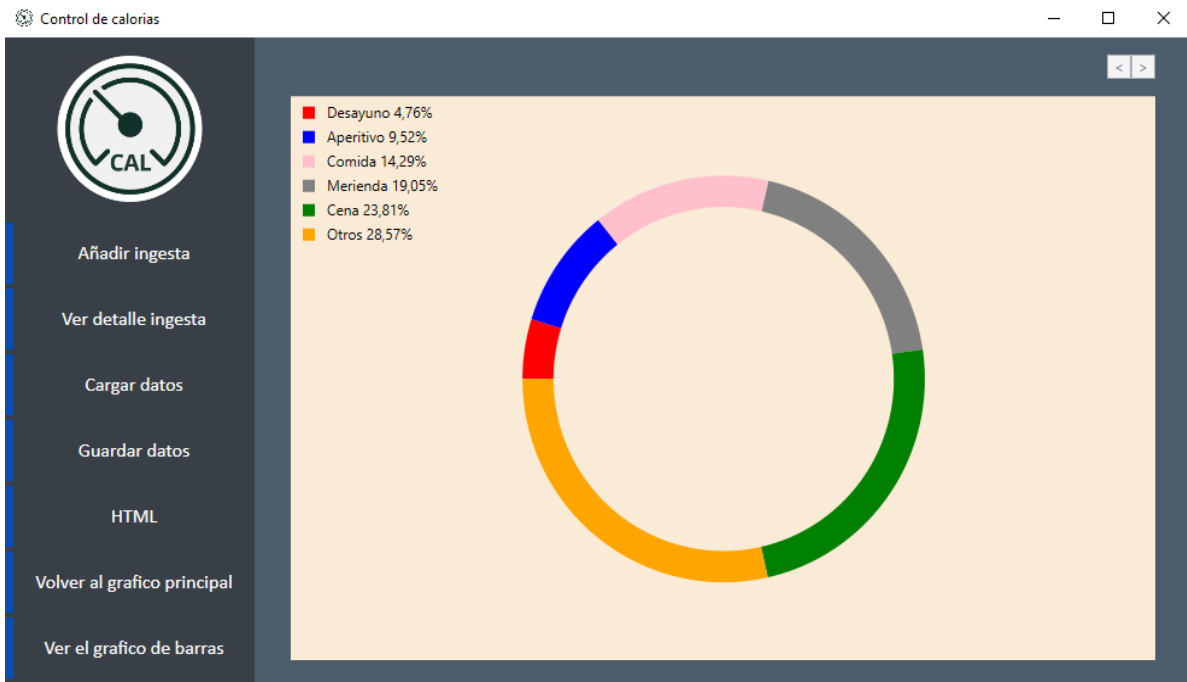


Al representar este gráfico, van a aparecer dos nuevos botones en la ventana principal. El botón **Volver al gráfico principal** permitirá al usuario volver al gráfico principal y el botón **Ver el gráfico de sectores** dibujara el gráfico de sectores del detalle de esa ingesta.

Gráfico de sectores de una ingesta

Cuando el usuario seleccione un ingesta en la ventana que muestra los detalles de la ingesta, en la ventana principal se dibujara un gráfico de sectores con los detalles de esa ingesta. Este gráfico va a tener una segmento por cada comida que tenga la ingesta.

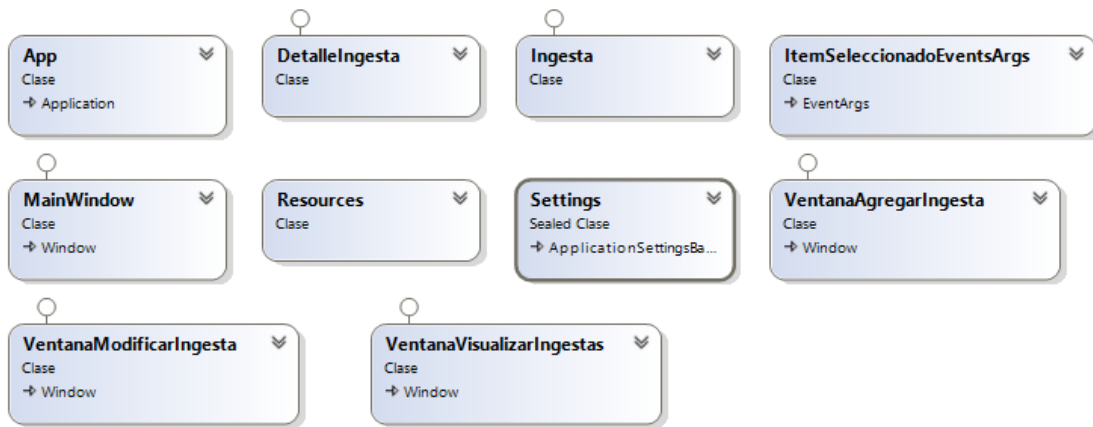
La apariencia del gráfico de barras del detalle de una ingesta es el siguiente:



Al representar este gráfico, van a aparecer dos nuevos botones en la ventana principal. El botón **Volver al gráfico principal** permitirá al usuario volver al gráfico principal y el botón **Ver el gráfico de barras** dibujara de nuevo el gráfico de barras del detalle de esa ingesta.

Manual del programador

La estructura de la aplicación está basada en las siguientes clases:



Como se puede observar, la aplicación tiene 6 clases principales las cuales tienen una línea y un círculo encima.

Por un lado, las clases **MainWindow**, **VentanaModificarIngesta**, **VentanaAgregarIngesta** y **VentanaVisualizarIngesta** son ventanas ya que heredan de Window y van a proporcionar la funcionalidad de las distintas ventanas de la aplicación. Por otro lado, las clases **Ingesta** y **DetalleIngesta** van a actuar como un modelo y se van a encargar de almacenar los datos de la aplicación.

Clase MainWindow

Esta clase va a contener la implementación de la funcionalidad de la ventana principal de la aplicación y por lo tanto es el fichero más importante del proyecto.

En primer lugar, se definen una serie de variables privadas accesibles para todos los métodos de la clase **MainWindow**. Entre ellas destaca un ObservableCollection, el cual es una lista que va a almacenar los datos de todas las ingestas. También destaca el atributo vvi el cual va a almacenar una instancia del cuadro de diálogo no modal llamado VentanaVisualizarIngestas.

El resto de atributos van a ayudar a realizar la funcionalidad de la ventana principal pero no merecen una mención especial.

Más adelante, en el constructor de la clase se va a registrar un controlador de eventos llamado **Ingestas_CollectionChanged** el cual se va a disparar cuando se produzca algún cambio en el ObservableCollection que almacena las ingestas. También se llama al método **dibujarEjes** el cual se va a encargar de dibujar los ejes principales del gráfico de barra en el canvas.

Método dibujarEjes

Este método se va a encargar de dibujar en el canvas de la ventana principal los ejes del gráfico de barras y sus correspondientes etiquetas con los valores de esas guías.

En primer lugar, se van a dibujar 11 líneas horizontales que recorren casi todo el ancho del canvas y para ello se va a dividir la altura del canvas en 12. La primera línea horizontal va a representar el eje X.

En segundo lugar, se va a dibujar una línea vertical la cual va a representar el eje Y y por eso tiene un grosor mayor al resto de líneas.

Por último, se van a dibujar 11 etiquetas las cuales van a contener cada una el valor asociado de cada una de las guías horizontales.

Estas líneas y etiquetas se van a almacenar en dos vectores accesibles por el resto de métodos de **MainWindow** ya que van a ser utilizadas en otros métodos.

Método dibujarIngesta

Este método se va a encargar de dibujar las barras de las comidas de una ingesta la cual se va a recibir como parámetro. También recibe un parámetro de tipo booleano para determinar si el dibujo de las barras de esa ingesta van a tener animaciones.

La clave de este método es obtener la posición x en la que se va a dibujar la ingesta, ya que solo se van a poder dibujar 10 ingestas al mismo tiempo. Esto es lo primero que se calcula.

Una vez calculada la posición x, se procede a dibujar las barras de la ingesta de forma que la coordenada Y2 de la primera línea corresponde con la coordenada Y1 de la siguiente línea y así sucesivamente. También es importante resaltar que para calcular la altura de cada barra se utiliza una regla de 3 entre el valor numérico de la ingesta a dibujar, el valor del total del calorías de la ingesta máxima representada (para ello se utiliza la función *buscarDietaMaximaRedondeadaCentena*) y la altura de representación del canvas.

Por último se actualiza el valor de las etiquetas en función de la ingesta máxima representada en el canvas.

Método Ingestas_CollectionChanged

Como ya hemos dicho, el método *Ingestas_CollectionChanged* es el controlador de eventos que se va a disparar cuando se produzca algún cambio en el ObservableCollection que almacena las ingestas de la aplicación.

La misión principal de este método es actualizar los gráficos del canvas de la ventana principal en función de los cambios que se hayan producido en el ObservableCollection.

Para ello, se van a recorrer los elementos añadidos o eliminados de forma que se van a dibujar las nuevas ingestas en el caso de que se hayan añadido ingestas y se van a redibujar todas las ingestas que aún permanecen en el ObservableCollection en el caso de que se han eliminado ingestas.

Si al añadir una ingesta, esa ingesta tiene un total de calorías mayor al resto de ingestas dibujadas, se tendrá que redibujar todas las ingestas para adaptar su altura en función de la nueva ingesta.

Al recorrer los nuevos elementos, se registra un controlador de eventos llamado *Ingesta_PropertyChanged* el cual se ejecutará cuando se produzca el evento PropertyChanged en la ingesta correspondiente. Este método *Ingesta_PropertyChanged* vuelve a dibujar el gráfico de barras del detalle o de sectores (depende de cual estuviera dibujado antes) de un ingesta cuando esa ingesta seleccionada sufra alguna modificación.

Método BotonIngesta_Click

Este método se va a encargar de mostrar la ventana que se encarga de agregar las ingestas.

Cuando el usuario haya introducido los datos de la ingesta, y pulse en aceptar, se creará una instancia de una ingesta en la que se cargaran los datos introducidos por el cliente. Por último, se añadirá esa ingesta a la lista de ingestas.

Si las ingestas no cumplen una serie de condiciones (por ejemplo, que todas las ingestas deben tener fecha), estas no serán agregadas a la lista de ingestas.

Método MenuItem_Click

Este método se va a encargar de mostrar la ventana que proporciona los detalles de las ingestas.

Es importante resaltar que cuando se crea una instancia de la clase **VentanaVisualizarIngestas**, se pasa como parámetro la colección de ingestas almacenada en el ObservableCollection.

Dentro de este método se va registrar un controlador de eventos llamado *Cdnm_Closed* que se va a ejecutar cuando se cierre la ventana y lo único que va a hacer es poner la instancia de la ventana a null para evitar dobles apariciones de la ventana.

También se va a registrar otro controlador de eventos llamando *Vvi_SeleccionItem* que se va a ejecutar cuando se seleccione un ítem en la ListView1 de la ventana secundaria que muestra los detalles de las ingestas. Este método se va a encargar de dibujar el detalle de la ingesta seleccionada en el canvas de la ventana principal. Su funcionamiento se explicará a continuación.

Método Vvi_SeleccionItem

Como ya hemos dicho, el método *Vvi_SeleccionItem* es el controlador de eventos que se va a disparar cuando se seleccione un ítem en la ListView1 de la ventana secundaria que muestra los detalles de las ingestas.

Cuando el usuario haya seleccionado un ítem en la ventana secundaria, se va a dibujar en la ventana principal el detalle de la ingesta seleccionada bien en forma de gráfico de barras o bien en forma de gráfico de sectores. Esto va a depender del estado de la ventana principal antes de seleccionar un ítem en la ventana secundaria.

Método dibujarDetalleIngesta

Este método se va a encargar de dibujar las barras de los detalles de una ingesta. Este método va a recibir un parámetro de tipo booleano para determinar si el dibujo de las barras del detalle de esa ingesta van a tener animaciones.

A diferencia de *dibujarIngesta*, en este método es fácil calcular la posición x donde se va a dibujar cada una de las barras. Esta posición se calculará dinámicamente en función del ancho del canvas.

Al igual que en *dibujarIngesta*, también se utiliza una regla de 3 entre el valor numérico de la comida a dibujar, el valor de la comida con más calorías de esa ingesta (para ello se utiliza la función *buscarIngestaMaximaRedondeadaCentena*) y la altura de representación del canvas.

Por último se actualiza el valor de las etiquetas en función de la ingesta máxima representada en el canvas.

Método lienzoGrafico_SizeChanged

Este método se va a encargar de borrar y dibujar el gráfico de barras o de sectores (dependiendo del gráfico que esté representado en cada momento).

Como se puede observar, cuando redibujamos un gráfico se llama al método dibujar correspondiente y se le pasa como parámetro un valor falso. Esto se hace porque en el redibujado del gráfico no se desea que haya animaciones.

Método dibujarGraficoSectores

Este método se va a encargar de dibujar el gráfico de sectores de la ingesta seleccionada. Si no se ha seleccionado ninguna ingesta en el ListView1, el botón que acciona el dibujo del gráfico de sectores no aparecerá. Este método va a recibir un parámetro de tipo booleano para determinar si el dibujo del gráfico de sectores va a tener animaciones.

Para dibujar el gráfico de sectores de una ingesta determinada se han seguido los siguientes pasos:

1. Se calculan los puntos de una circunferencia (para obtener los puntos de la circunferencia se han obtenido los puntos de dos semicircunferencias y después se han juntado).
2. Se calcula el módulo total de esa circunferencia, es decir, el perímetro de la circunferencia.
3. Se calcula la porción de perímetro correspondiente para cada ingesta en función del peso de la comida sobre el total de calorías de esa ingesta (se utiliza una regla de 3 entre el perímetro total de la circunferencia, el número de calorías de la comida y el número total de calorías de la ingesta).
4. A cada comida se le asignan una determinada cantidad de puntos en función de la porción de perímetro que le corresponda a esa comida de la ingesta.
5. Se añade esa colección de puntos a la polilínea correspondiente la cual va a ser dibujada en el canvas.

Clase VentanaVisualizarIngestas

Esta clase va a contener la implementación de la funcionalidad de la ventana encargada de mostrar los detalles de las ingestas guardadas en la aplicación.

Esta clase va a contener dos atributos fundamentales:

- Un **ObservableCollection** que va a contener los datos de las ingestas que almacena la aplicación. Esta colección de ingestas va a ser enviada por la ventana principal como parámetro cuando se cree la instancia de esta clase.
- Un delegado predefinido de WPF llamado **EventHandler** que conecta el evento `SeleccionItem` con el controlador de eventos `Vvi_SeleccionItem` de la ventana principal el cual queremos que se ejecute cuando se produzca el evento.

Como el evento `SeleccionItem` asociado al delegado genera datos, se utiliza un delegado predefinido del tipo `EventHandler<ItemSeleccionadoEventArgs>` donde **ItemSeleccionadoEventArgs** es una clase que deriva de **EventArgs** que proporciona las propiedades necesarias para contener los datos del evento.

Más adelante, en el constructor de la clase se va a producir una asignación entre el `ObservableCollection` de la clase **VentanaVisualizarIngestas** y el `ObservableCollection` pasado como parámetro en la instanciación de esta clase. Por otro lado, la fuente de origen de datos del `ListView1` será el `ObservableCollection` que contiene los datos de las ingestas.

Metodo `ListBox1_SelectionChanged`

Este método se va a encargar de cargar los detalles de una ingesta en la `ListView2` cuando el usuario haya seleccionado una ingesta en el `ListView1`.

Si se detecta que hay algún ítem seleccionado en la ListView1, se asigna la lista de los detalles de la ingesta seleccionada a la fuente de origen de datos de la ListView2 y aparecerán los detalles de la ingesta seleccionada en la ListView2.

Después, se va a crear una instancia de la clase **ItemSeleccionadoEventArgs** y se va a asignar a la propiedad IngestaSeleccionada de **ItemSeleccionadoEventArgs** la ingesta que está seleccionada. Como ya dije anteriormente, el evento SeleccionItem asociado al delegado genera datos y la clase **ItemSeleccionadoEventArgs** que deriva de **EventArgs** proporciona las propiedades necesarias para contener los datos del evento.

Por último, se va a llamar la metodo *OnSeleccionItem* y al que se le pasa como parámetro la instancia de la clase **ItemSeleccionadoEventArgs**. Este método va a generar el evento SeleccionItem de forma que se van a disparar todos los controladores de eventos registrados. En este caso el controlador de eventos registrado es el método *Vvi_SeleccionItem* de la ventana principal. Para ello se utilizará la sentencia invoke que recibe como parámetros la referencia al objeto **VentanaVisualizarIngestas** y la instancia de la clase **ItemSeleccionadoEventArgs** que deriva de **EventArgs** pasada como parámetro y contiene la ingesta que ha sido seleccionada. Esto es lo que le permite a la clase **MainWindow** conocer que ingesta se seleccionó en la clase **VentanaVisualizarIngestas**.

Si se detecta que no hay ningún ítem seleccionado en la ListView1, la fuente de origen de datos de la ListView2 será nulo y no aparecerá ningún tipo de información en la ListView2.

Método AplicarFiltrosCalorias_Click

Este método se va a encargar de filtrar las ingestas almacenadas en la aplicación de forma que solo aparecen en la ListView1 las ingestas que cumplan con los filtros introducidos por el usuario.

Para filtrar las ingestas, se va a utilizar otro ObservableCollection que va a almacenar las ingestas que cumplen con los filtros introducidos. De esta manera, se va a recorrer el ObservableCollection que almacena todas las ingestas y se van a almacenar en el otro ObservableCollection las ingestas que cumplan con los filtros. Después se asigna este ObservableCollection a la fuente de origen de datos del ListView1 para que solo aparezcan las ingestas que cumplen con el filtro.

Cuando se restauren los filtros, la fuente de origen de datos del ListView1 volverá a ser el ObservableCollection que almacena todas las ingestas.

Clase ItemSeleccionadoEventArgs

Esta clase hereda de **EventArgs** y va a proporcionar una propiedad llamada IngestaSeleccionada que va a manejar y contener los datos del evento. En este caso, el dato que va a manejar esta clase **ItemSeleccionadoEventArgs** es una instancia de la clase ingesta que contiene la información de la ingesta seleccionada en el ListView1. De esta manera, la clase **MainWindow** va a poder acceder a la ingesta seleccionada a través del segundo parámetro que recibe el método *Vvi_SeleccionItem* de la clase **MainWindow**,

el cual es un parámetro del tipo ***ItemSelectedEventArgs***. En resumen, esta clase va a permitir intercambiar información entre dos métodos de clases distintas cuando se produzca un determinado evento (en este caso que el usuario seleccione un ítem del ListView1).

Clase VentanaAgregarIngesta

Esta clase va a contener la implementación de la funcionalidad de la ventana encargada de agregar los datos de una nueva ingesta a los datos de la aplicación.

Los métodos de esta clase son muy sencillos y están relacionados la mayoría con la presentación gráfica de la ventana.

El único método reseñable es *AceptarIngesta_Click*, el cual establecerá el valor de la propiedad DialogResult de esta clase en verdadero. Este método se va a ejecutar cuando el usuario presione el botón Aceptar.

Si el valor de la propiedad DialogResult es verdadero, entonces, el método *BotonIngesta_Click* de la clase ***MainWindow*** creará una instancia de la clase ***Ingesta*** y rellenará sus campos con los datos introducidos en la ventana de agregar ingestas. Después, esa instancia de Ingesta que contiene los datos introducidos se añadirá al ObservableCollection que almacena los datos de todas las ingestas.

Clase VentanaAgregarIngesta

Esta clase va a contener la implementación de la funcionalidad de la ventana encargada de modificar los datos de una ingesta ya existente en los datos de la aplicación.

A diferencia de la clase ***VentanaAgregarIngesta***, esta clase recibe como parámetro una instancia de la ingesta que se va a modificar. Esta clase recibe un parámetro con la ingesta que va a modificar porque en el constructor de la clase se registra un manejador de eventos llamado *MainWindow_Loaded* que se ejecutará cuando se produzca el evento Loaded. Este método *MainWindow_Loaded* va a cargar en los distintos campos de la ventana la información de la ingesta a modificar con el objetivo de facilitar la modificación de la ingesta.

Otro método reseñable es *AceptarIngesta_Click*, el cual establecerá el valor de la propiedad DialogResult de esta clase en verdadero. Este método se va a ejecutar cuando el usuario presione el botón Aceptar.

Si el valor de la propiedad DialogResult es verdadero, entonces, el método *BotonIngesta_Click* actualizará los valores de la ingesta que hayan sido modificados. Cuando se produzca esa actualización, el ObservableCollection notificará un cambio en una propiedad del modelo y se actualizarán los datos de la ingesta en las dos ListView y se actualizará el dibujo del detalle de la ingesta. Esto se explicará con mayor detalle en la explicación de las clases del modelo.

Clase Ingesta

Esta clase actúa de modelo ya que va a almacenar los datos de las distintas ingestas y se va a encargar también de notificar los distintos cambios que se produzcan en esos datos. Esta clase hereda de **INotifyPropertyChanged** ya que esta interfaz es la que proporciona las clases necesarias para gestionar los cambios en el modelo.

Esta clase va a tener 4 atributos:

- Una cadena de caracteres donde se va a almacenar la fecha,
- Un entero que va a almacenar el total de calorías de la ingesta.
- Un **ObservableCollection** que va a almacenar una colección de objetos **DetalleIngesta** que van a contener la información de las distintas comidas que forman la ingesta.
- Un delegado predefinido de WPF llamado **PropertyChangedEventHandler** que conecta el evento **PropertyChanged** con el controlador de eventos **Ingestas_CollectionChanged** de la clase **MainWindow** el cual queremos que se ejecute cuando se produzca el evento.

Más adelante, en el constructor de la clase, se van a crear una serie de objetos **DetalleIngesta** y se va a asignar a cada objeto **DetalleIngesta** el nombre de la comida correspondiente y el color asociado a dicha comida cuando se representen las ingestas en el canvas. Después, cada objeto **DetalleIngesta** se va añadir al **ObservableCollection**.

Las propiedades de esta clase permiten establecer y obtener la fecha de la ingesta, las calorías de las diferentes comidas de la ingesta y el **ObservableCollection** que contiene los detalles de la ingesta.

Cuando se establece un nuevo valor para la fecha de la ingesta o se establece un nuevo valor para las calorías de una comida de la ingesta, se llama al método **OnPropertyChanged** el cual va a recibir como parámetro el nombre de la propiedad o propiedades que han cambiado su valor.

Este método va a generar el evento **PropertyChanged** de forma que se van a disparar todos los controladores de eventos registrados. En este caso el controlador de eventos registrado es el método **Ingestas_CollectionChanged** de la clase **MainWindow**. Para ello se utilizará la sentencia **invoke** que recibe como parámetros la referencia al objeto **Ingesta** que produce el evento y una instancia de la clase **PropertyChangedEventArgs**. Para crear la instancia de la clase **PropertyChangedEventArgs** se pasa el nombre de la propiedad que ha cambiado como parámetro.

Nótese que cuando se modifica el valor de las calorías de una comida de la ingesta, también se modifica el valor del total de calorías de la ingesta. Por eso, se llama al método **OnPropertyChanged** dos veces, una para notificar que ha cambiado las calorías de la comida y otra para notificar que ha cambiado el valor del total de calorías. El método **On** para notificar que han cambiado el total de calorías es ejecutado por la clase **Ingesta** y el método **On** para notificar que han cambiado el valor de las calorías de una comida de la ingesta es ejecutado por la clase **DetalleIngesta** como se verá a continuación.

Clase DetalleIngesta

Esta clase forma parte del modelo ya que va a almacenar los datos de las distintas comidas de una ingesta y se va a encargar también de notificar los distintos cambios que se produzcan en esos datos. Esta clase hereda de ***INotifyPropertyChanged*** ya que esta interfaz es la que proporciona las clases necesarias para gestionar los cambios en el modelo.

Esta clase va a tener 4 atributos:

- Una cadena de caracteres donde se va a almacenar el nombre de la comida.
- Un entero que va a almacenar las calorías de esa comida.
- Una brocha que va a almacenar una bocha con el color que va a tener esa comida en los gráficos del canvas.
- Un delegado predefinido de WPF llamado ***PropertyChangedEventHandler*** que conecta el evento ***PropertyChanged*** con el controlador de eventos ***Ingestas_CollectionChanged*** de la clase ***MainWindow*** el cual queremos que se ejecute cuando se produzca el evento.

Esta clase tiene 3 propiedades, las cuales permiten establecer y devolver el valor del nombre de la comida, las calorías de esa comida y el color de la brocha respectivamente. Cuando se establece un nuevo valor para el nombre de la comida o para las calorías de esa comida, se llama al método ***OnPropertyChanged*** el cual va a recibir como parámetro el nombre de la propiedad que ha cambiado su valor.

Este método va a generar el evento ***PropertyChanged*** de forma que se van a disparar todos los controladores de eventos registrados. En este caso el controlador de eventos registrado es el método ***Ingestas_CollectionChanged*** de la clase ***MainWindow***. Para ello se utilizará la sentencia ***invoke*** que recibe como parámetros la referencia al objeto ***DetalleIngesta*** que produce el evento y una instancia de la clase ***PropertyChangedEventArgs***. Para crear la instancia de la clase ***PropertyChangedEventArgs*** se pasa el nombre de la propiedad que ha cambiado como parámetro.

Bibliografía

- <https://docs.microsoft.com/es-es/dotnet/desktop/wpf/graphics-multimedia/how-to-paint-an-area-with-a-solid-color?view=netframeworkdesktop-4.8>
- <https://wpf-tutorial.com/es/46/dialogos/el-dialogo-openfiledialog/>
- <https://docs.microsoft.com/es-es/troubleshoot/dotnet/csharp/read-write-text-file>
- <https://www.it-swarm-es.com/es/c%23/como-puedo-escribir-en-mayuscula-la-primer-letra-del-nombre-y-el-apellido-en-c/957382671/>
- <https://docs.microsoft.com/es-es/dotnet/api/system.eventhandler?view=net-6.0>
- <https://docs.microsoft.com/es-es/dotnet/standard/events/>
- https://studium.usal.es/pluginfile.php/156526/mod_resource/content/6/WPF/WPF%204%20Eventos.pdf
- https://studium.usal.es/pluginfile.php/156527/mod_resource/content/2/WPF%205%20Graficos.pdf
- https://studium.usal.es/pluginfile.php/156527/mod_resource/content/2/WPF%205%20Graficos.pdf
- https://studium.usal.es/pluginfile.php/156529/mod_resource/content/1/WPF/WPF_6_Controles.pdf
- https://studium.usal.es/pluginfile.php/156530/mod_resource/content/2/WPF%207%20Cuadros%20de%20Di%C3%A1logo.pdf
- https://studium.usal.es/pluginfile.php/156531/mod_resource/content/2/WPF%208%20Enlace%20de%20datos.pdf
- https://www.youtube.com/watch?v=en_eqatpDEo&list=RDCMUCEQIGiXhpdO4Qhn0sPviagg&index=6&ab_channel=RJCodeAdvance
- https://www.youtube.com/watch?v=eCSbUCL4teE&ab_channel=RJCodeAdvance