Logotipo

Descripción generada automáticamente

**Benemérita Universidad Autónoma de Puebla**

*Materia: Sistemas Operativos II*

*Profesor: Gilberto López Poblano*

*Alumno: Christian Amauri Amador Ortega - 201927821*

**- Proyecto Word -**

*Para esta serie de prácticas con código utilizamos el lenguaje C# en el IDE visual studio (la versión más reciente en septiembre de 2022). Puede haber variaciones en el funcionamiento de exactamente los mismos códigos pero con otra versión de visual studio u otro IDE. De todas formas estas variaciones deben de ser mínimas e insignificantes.*

*!!! Documentaremos el funcionamiento de las aplicaciones (en su ejecución) y haremos algunos comentarios / notas / consideraciones / advertencias al respecto. Para ahorrar tiempo y esfuerzo, trataremos de no ahondar demasiado en la estructura del código, sino simplemente explicar brevemente cómo funciona* ***(se asume que el lector tiene ya conocimientos suficientes sobre conceptos varios de programación estructurada, ambiente gráfico, variables, datos, y demás…)***

*Finalmente hay que tomar en cuenta que C# y Visual studio son herramientas pesadas y caprichosas. Y que el equipo que vayamos a utilizar debe tener cierta capacidad de rendimiento superior a sólo básica.*

Primero, el objetivo del proyecto es crear un programa en C# con el que podamos simular un pequeño programa tipo Microsoft Word, para crear documentos de texto con variaciones de fuente en color, tamaño, tipo y alineado, además de la opción de insertar imágenes. Guardar, y abrir archivos con un formato definido por nosotros.

*La ventaja que tenemos en este proyecto es que aparentemente va a estar construido enteramente con funciones predefinidas por el lenguaje, así que mucho trabajo de ingeniería no debemos hacer realmente, simplemente usar y conocer las herramientas que ya fueron hechas por alguien más en algún momento (lo cual es muy bueno para la práctica, pero considerablemente desfavorable para la documentación).*

En cualquier caso, el repositorio en el que este código se encuentra (y otros códigos en C# del curso de Sistemas operativos 2, BUAP FCC, Otoño 2022) está disponible haciendo click en el siguiente enlace:

<https://drive.google.com/drive/folders/1yZujMI51XAnEZBMj-4ykUWMfHXBC9uYp?usp=sharing>

Y estará disponible al menos durante el resto de 2022. Gracias. 😊

***Ejemplo rápido del funcionamiento general del proyecto, para resumir:***

* Podemos crear documentos con nuestro propio tipo de formato (BUAP files)
* Estos documentos consisten en textos modificables en color, fuente, estilo, y la opción de insertar imágenes.

Una captura de pantalla de una red social

Descripción generada automáticamente

* Tenemos la opción de guardar documentos de forma rápida, o específica.

Captura de pantalla de computadora

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Captura de pantalla de computadora

Descripción generada automáticamente

(Podemos guardar cambios sobre un mismo archivo predefinido. Como todos los editores de documentos de diversos formatos)

Ahora, los archivos .cs del proyecto (que nosotros creamos) corresponden a 2 diseños y 2 scripts. Comencemos por los diseños…

***- Diseños –***

Captura de pantalla de computadora

Descripción generada automáticamente

#1 **Form1** (Form main)

**Contiene los botones y las opciones con las que editaremos nuestros textos**, tales como; Alinear a la izquierda, centro o derecha. Abrir archivo, Guardar archivo. establecer letra como: negrita, cursica, subrayada. Cambiar el color y tamaño de la fuente. E insertar imágenes. Todas ellas corresponden a las funciones básicas de todo editor de textos / documentos.

Captura de pantalla de computadora

Descripción generada automáticamente Una captura de pantalla de una computadora

Descripción generada automáticamente Captura de pantalla de computadora

Descripción generada automáticamente

Captura de pantalla de computadora

Descripción generada automáticamente

#2 **Form2** (Acerca de)

**Contiene información sobre el autor del código** (no posee funcionalidad extra). Es invocada por la opción “Acerca de” en la opción “Ver” del menú strip de la ventana principal.

***- Scripts –***

#1 ***Form1.cs***

La estructura inicial de este script consta de:

* 2 librerías
* namespace definido como: wordNuevo
* public partial class Form1 : Form

Captura de pantalla de computadora

Descripción generada automáticamente

Dentro de Form1 : Form. se encuentran los siguientes componentes:

Captura de pantalla de computadora

Descripción generada automáticamente Captura de pantalla de computadora

Descripción generada automáticamente

(de la imagen de la derecha, cada una de esas líneas es un método comprimido)

* 3 variables (tipo String y Form2)
* un constructor
* 43 métodos

Comencemos por las opciones de la opción “Archivo” del menu strip de la ventana principal:

El método nuevoToolStripMenuItem\_Click() corresponde a la opción “Nuevo”

Captura de pantalla de computadora

Descripción generada automáticamente

El método abrirToolStripMenuItem\_Click() corresponde a la opción “abrir”. Manda a llamar al método abrir(), en donde se usa un openFileDialog para leer archivos con filtro .BUAP files:

Captura de pantalla de computadora

Descripción generada automáticamente

El método guardarToolStripMenuItem\_Click() corresponde a la opción “Guardar”. está hecho para guardar de forma rápida sobre el archivo que se esté trabajando, pero si se está trabajando sobre un documento nuevo, se manda a llamar al método guardar().

Captura de pantalla de computadora

Descripción generada automáticamente

El método guardar() funciona como cada método de guardado de archivos que hemos trabajdo antes, Usa un filtro de formato de fichero para verificar que se guarden adecuadamente y mediante un SaveFileDialog() respaldado por un pequeño algoritmo de confirmación, realiza el proceso correspondiente:

Captura de pantalla de computadora

Descripción generada automáticamente

El método guardarComoToolStripMenuItem\_Click() corresponde a la opción “Guardar como”. Manda a llamar al método guardar() :

Captura de pantalla de computadora

Descripción generada automáticamente

El método salirToolStripMenuItem\_Click() corresponde a la opción “Salir”. Este termina la ejecución de la aplicación:

Captura de pantalla de computadora

Descripción generada automáticamente

Sigamos con las opciones de la opción “Editar” del menú strip de la ventana principal. Estas son relativamente sencillas y parecidas, en algunos casos cada una manda a llamar a un sub-método que cumple su propósito (casi casi como meramente paso extra), de modo que por cada opción en este menú, tenemos dos métodos (el que llama al método real, y el que es llamado en primer lugar):

El método blodToolStripMenuItem\_Click() manda a llamar al método negrita(), en donde se realiza el proceso real, el cual consiste simplemente en usar funciones predefinidas del lenguaje para darle a la fuente el efecto que deseamos:

Captura de pantalla de computadora

Descripción generada automáticamenteCaptura de pantalla de computadora

Descripción generada automáticamente

Como dijimos, este concepto se repite para las opciones de cursiva, subrayada y agregar imágen:

Captura de pantalla de computadora

Descripción generada automáticamenteCaptura de pantalla de computadora

Descripción generada automáticamenteCaptura de pantalla de computadora

Descripción generada automáticamente

Para las opciones de alinear a la derecha, centro o izquierda; se ejecuta el cambio en una simple línea de código, en un solo llamado:

Captura de pantalla de computadora

Descripción generada automáticamente

Finalmente en esta parte, las opciones de la opción “Ver”; La primeras tres opciones de esta opción corresponden a los métodos que establecen como visibles o NO visibles los tres conjuntos de íconos mostrados en la ventana principal. Esta acción se ejecuta en una sola línea, negando una vez el estado actual de visibilidad del conjunto de íconos:

* “Principal” corresponde a los íconos de color, tamaño y tipo de fuente: Interfaz de usuario gráfica, Texto, Aplicación

  Descripción generada automáticamente
* “Párrafo” corresponde a los íconos de alineado del texto:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

* “Imagen” corresponde al ícono de imagen:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Métodos respectivos:

Captura de pantalla de computadora

Descripción generada automáticamente

La opción acercaDeToolStripMenuItem\_Click\_1() “Acerca de” manda a llamar al método el cual crea un objeto de tipo Form2, el cual existe única y exclusivamente con el propósito de cubrir esta opción (mostrar información):

Captura de pantalla de computadora

Descripción generada automáticamente

Ahora, para cada método de cada ícono en la ventana main, simplemente se manda a llamar el método correspondiente (otra vez), de ahí que el código parezca tan largo. En realidad no son tantos métodos como parecen, simplemente hay muchas formas de mandarlos a llamar…

Captura de pantalla de computadora

Descripción generada automáticamente

El método Form1\_Load() es un método que se manda a llamar en el Form.1Designer.cs. Este sirve para agregar a la ventana principal los Items y características que esta usa (no hay mucho más que entender, esto es más técnico realmente):

Captura de pantalla de computadora

Descripción generada automáticamente

#2 ***Form2.cs***

La estructura inicial de este script consta de:

* 9 librerías
* namespace definido como: wordNuevo
* public partial class Form2 : Form

Captura de pantalla de computadora

Descripción generada automáticamente

Dentro de Form2 : Form. Simplemente se encuentra el constructor de la calse form2 (ventana de Acerca de) la cual es llamada cuando se presiona la opción “Acerca de” en la opción “Ver” del menú strip de la ventana principal.

Un perro con la lengua de fuera

Descripción generada automáticamente con confianza media

Como comentario final, este proyecto es más didáctico que funcional (no debe ser tomado demasiado en serio). El resultado sigue estando muy pero muy por debajo de cualquier editor de texto o documentos. Si acaso podríamos hacer notas personales simples o algo por el estilo con esto. Y también debemos recordar que al estar construido enteramente con funciones definidas en el lenguaje, hacer una documentación detallada de cada parte del código se convertiría más bien en una guía del lenguaje. Esta documentación se reserva el derecho de mostrar únicamente detalles exteriores sobre la estructura de nuestro producto.