INSTITUTO CEI

RELOJ DIGITAL

NOVIEMBRE 2023

PROFESOR GONZALO DUARTE

Federico Goldaracena

Resumen

En este documento se presenta el desarrollo de una aplicación de Reloj Digital, diseñada para operar como una aplicación de escritorio en el entorno Windows. El proceso se llevó a cabo utilizando el entorno de desarrollo integrado (IDE) Visual Studio, aprovechando la interfaz gráfica Windows Presentation Foundation (WPF) incluida en .NET Framework. Esta tecnología facilita la creación de aplicaciones con un diseño atractivo y amigable para el usuario. WPF ofrece herramientas que simplifican el proceso de diseño mediante el uso del lenguaje de marcado XAML (Extensible Application Markup Language), que define la apariencia visual de la interfaz. Además, se emplea el lenguaje de programación C# para definir las funciones específicas de cada componente y del sistema en su conjunto. El reloj digital resultante de este desarrollo se beneficia de la combinación de la potencia de Visual Studio, la versatilidad de WPF y la eficacia de C#. El enfoque integral de estas tecnologías permite crear una aplicación funcional con una interfaz intuitiva y atractiva para el usuario final.

Contenido

1. Introducción

1.1 .NET

.NET, desarrollado por Microsoft, es un marco de trabajo de código abierto y multiplataforma diseñado para construir aplicaciones y servicios web. Desde su establecimiento en 2002, ha proporcionado a los desarrolladores una variedad de herramientas, lenguajes de programación y bibliotecas para la creación de aplicaciones de escritorio y web. Entre los lenguajes compatibles se encuentran C++, Visual Basic, F#, C# y PowerShell, así como tecnologías web populares como HTML, JavaScript y CSS. La esencia fundamental de .NET radica en proporcionar a los desarrolladores una experiencia de desarrollo unificada, fácil de usar y mantener. Esto les permite desarrollar soluciones de software sólidas y seguras de manera eficiente. Además, .NET ofrece un conjunto completo de características, que incluyen un entorno de desarrollo integrado (IDE), herramientas de depuración, pruebas y despliegue automatizados, junto con bibliotecas de aplicaciones. Esta combinación de herramientas y tecnologías contribuye a la eficiencia y confiabilidad en el desarrollo de software. .NET se destaca como una plataforma integral que impulsa la creación eficiente de aplicaciones, siguiendo las mejores prácticas y patrones de diseño. [1]

1.2 WPF- Windows Presentation Foundation

La infraestructura de desarrollo de Windows Presentation Foundation (WPF) es un marco diseñado para la creación de aplicaciones de escritorio orientadas al cliente. WPF ofrece un robusto conjunto de capacidades de desarrollo de aplicaciones, abarcando un modelo de aplicación, recursos, controles, gráficos, diseños, enlace de datos, documentos y seguridad. Al ser una extensión de .NET Framework, aquellos familiarizados con la creación de aplicaciones mediante Windows Forms o ASP.NET encontrarán que la experiencia de programación resulta reconocible. WPF utiliza el lenguaje XAML para establecer un enfoque declarativo en la programación de aplicaciones. Los temas en esta sección proporcionan una introducción a WPF y facilitan la iniciación en su implementación. [2]

2. Desarrollo

Se presenta el diseño de la interfaz y el código a través del entorno de desarrollo integrado (IDE) Visual Studio, utilizando la tecnología de Windows Presentation Foundation (WPF) y codificación en C# (Figura 2.1).

Figura 2.1: Inicio de proyecto WPF con dos Label en Visual Studio.

2.1 Diseño de la interfaz

Para dar inicio, se genera el diseño de la interfaz visual del reloj. En primer lugar, se ajusta el tamaño del reloj para simular la apariencia de un reloj de mesa virtual. Posteriormente, se incorporan dos elementos tipo Label, los cuales se añaden desde el Cuadro de Herramientas (Figura 2.2).

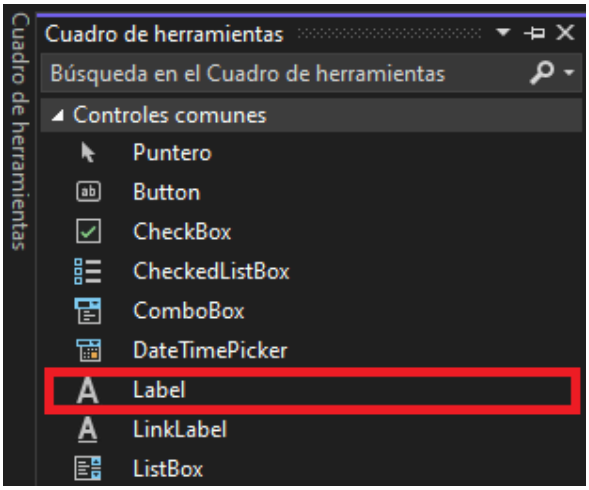

Figura 2.2: Cuadro de herramientas.

La siguiente imagen muestra el diseño final de la interfaz (Figura 2.3)

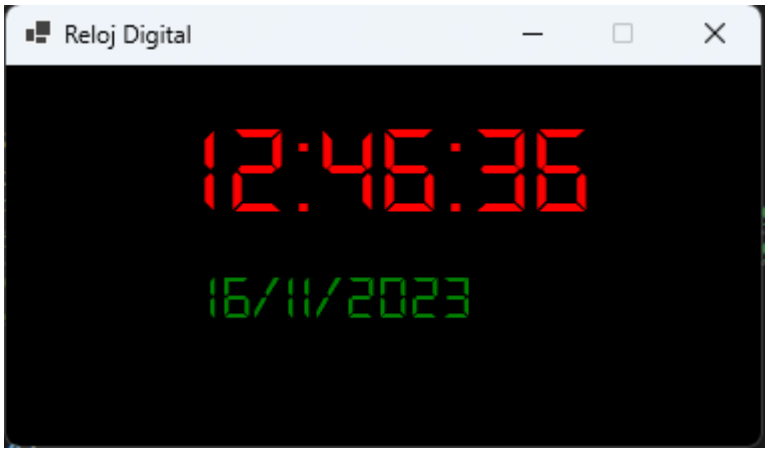

Figura 2.3: Interfaz Reloj Digital.

2.2 Codificación del funcionamiento

Se comenzó por configurar la ventana para dar tamaño fijo, deshabilitar el botón de maximizar y poner un texto a la ventana con el nombre “Reloj Digital” utilizando las funciones de WPF (Figura 2.4).

Luego se configura los dos Label para darle estilo, color y fuente. En el Font, se descargó a parte un archivo llamado DS-Digital, es un estilo que tiene de apariencia de los relojes de display de 7 segmentos (Figura 2.5).

```
public RelojDigital()
{
    InitializeComponent();
    // Configurar el estilo del borde y deshabilitar la opción de maximizar
    this.FormBorderStyle = FormBorderStyle.FixedSingle; // Tamaño fijo de la ventana
    this.MaximizeBox = false; // Deshabilitar el botón de maximizar
    this.Text = "Reloj Digital";  // Cambiar el texto de la barra de título
```

Figura 2.3: Parte del código RelojDigital para dar formato a la ventana.

```
    // Configurar la fuente y el color de fondo para las etiquetas
    labelHora.Font = new Font("DS-Digital", 48, FontStyle.Regular, GraphicsUnit.Point);
    labelFecha.Font = new Font("DS-Digital", 24, FontStyle.Regular, GraphicsUnit.Point);
    labelHora.BackColor = Color.Black;  // Color de fondo negro para el efecto de display de siete segmentos
    labelFecha.BackColor = Color.Black; // Color de fondo negro para el efecto de display de siete segmentos
    labelHora.ForeColor = Color.Red;    // Color rojo para los dígitos de la hora
    labelFecha.ForeColor = Color.Green;   // Color verde para los dígitos de la fecha
}
```

Figura 2.5: Continuación de Figura 2.4.

A continuación, se ha implementado el método "horaFecha_Tick", el cual facilita la presentación de la hora y fecha actuales, obtenidas directamente desde la computadora. Estos datos son visualizados en dos etiquetas designadas como "labelHora" y "LabelFecha". Además, se han creado dos variables, labelHora y LabelFecha, que se emplean en conjunto con la función "Text" para dar formato al texto exhibido. Este procedimiento se realiza mediante el aprovechamiento de dos funciones nativas de C#: "DateTime" y "Now". La función "DateTime" se encarga de

exponer las horas, minutos y segundos actuales, mientras que "Now" proporciona la hora actual del sistema computacional. Posteriormente, se aplican dos funciones adicionales, a saber, "ToLongTimeString()" y "ToShortDateString()". La función "ToLongTimeString()" presenta la hora actual en formato de cadena, detallando horas, minutos y segundos, mientras que "ToShortDateString()" exhibe la fecha en un formato simplificado, como por ejemplo: 01/01/2023. En un esfuerzo por mejorar la robustez del programa, se ha incorporado una instrucción de control de excepciones. En caso de que el programa no funcione correctamente, se captura la excepción y se muestra el error en pantalla (Figura 2.6).

```
private void horaFecha_Tick(object sender, EventArgs e)
{
    try
    {
        // Actualizar las etiquetas con la hora y la fecha actual
        labelHora.Text = DateTime.Now.ToLongTimeString();
        labelFecha.Text = DateTime.Now.ToShortDateString();
    }
    catch (Exception ex)
    {
        // Manejar la excepción, por ejemplo, mostrar un mensaje de error.
        MessageBox.Show("Error al obtener la hora y la fecha: " + ex.Message);
    }
}
```

Figura 2.6: Método que implementa dar hora y fecha.

3. Resultados

Para diseñar la interfaz se utilizaron elementos desde el cuadro de herramientas (Figura 3.1) y luego fueron agregados en el Form1 (Figura 3.2).

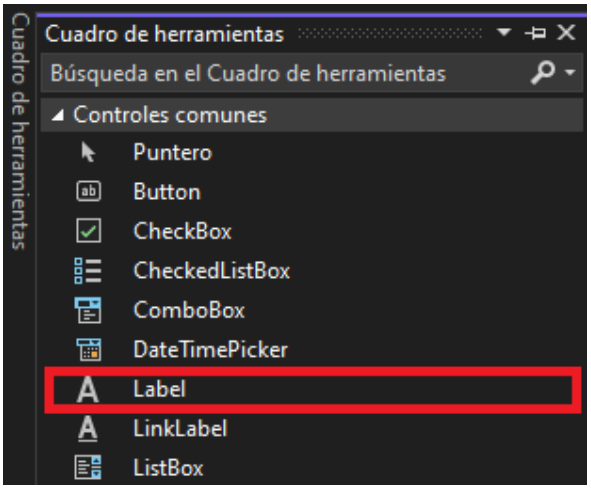

Figura 3.1

Figura 3.2

Diseño final de interfaz de Reloj Digital (Figura 3.3).

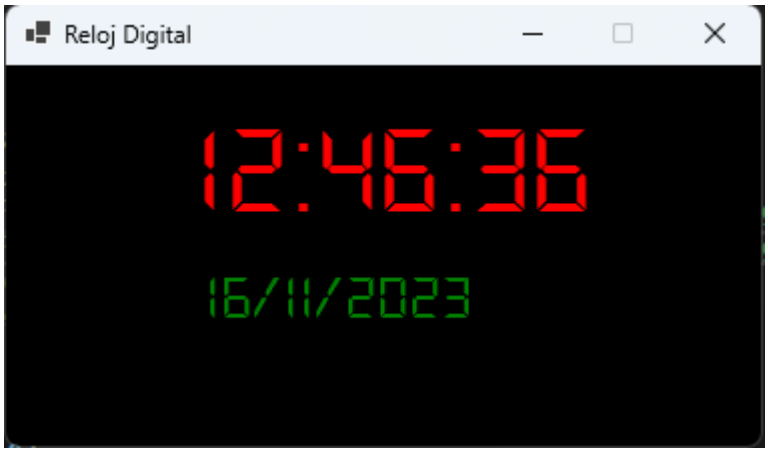

Figura 3.3

4. Conclusiones

La aplicación de WPF simplifica el desarrollo de un reloj digital al proporcionar una interfaz y un sistema de código basado en etiquetas XAML. Permite a los desarrolladores definir estas etiquetas mediante código escrito y facilita la generación automática utilizando herramientas directas, como los elementos Button y Label implementados en la creación del reloj. Al iniciar un proyecto, se crean automáticamente dos archivos XAML interrelacionados: uno para el diseño visual y otro para el código subyacente. Este último recibe la definición de los métodos necesarios para el funcionamiento del reloj digital. Cuando la aplicación se ejecuta, estos archivos se fusionan para dar vida a una interfaz atractiva y funcional para el usuario. En este caso, la interfaz se adapta para mostrar la hora actual y puede ofrecer funciones adicionales propias de un reloj digital. Este tipo de proyectos no solo proporciona a los usuarios una experiencia agradable, sino que también abre espacio para la expansión y modificación futura, aprovechando las diversas posibilidades proporcionadas por las herramientas potentes de WPF.

5. Referencias

[1] https://anywhere.epam.com/es/blog/que-es-dotnet

[2] https://learn.microsoft.com/es-es/dotnet/desktop/wpf/getting-started/?view=netframeworkdesktop-4.8