# Estándar de codificación para C# utilizando el IDE Visual Studio

En este estándar se establecerá el formato común para la generación de código utilizando el lenguaje de programación C# y también haciendo uso del IDE Visual Studio. Gracias a este formato permitirá compartir código entre los desarrolladores para que les resulte de una manera más clara y fácil de entender.

## Indentación y ajuste de línea

Se deberá utilizar exclusivamente tabulación para indentar, no espacios en blanco. Cada rama del código terminará en el mismo nivel en el que están alineados.

Las líneas de código cuyo largo supere al de la ventana de código de Visual Studio se ajustarán automáticamente gracias a su función de Ajuste de Línea, sin embargo el programador podrá dividir una línea de código según su criterio.

## Declaraciones

Para los nombres de variables, atributos, métodos, clases y otros identificadores deberán ser lo suficientemente descriptivos como para entender que son qué almacenan o cuál es su función, con excepción de las literales utilizadas en las estructuras como For y los predicados utilizados como parámetros por ciertas funciones como Where o First.

### Declaración de clases y métodos

El nombre de las clases deberá tener aplicado el estilo PasCal.

### Declaración de atributos y variables

El nombre de los atributos y variables deberá ser del estilo caMel.

### Nombres de controles WPF

El nombre de los controles de las ventanas en los archivos XAML hará uso del estilo caMel. Dicho nombre se dividirá en dos partes separadas por un “\_”. La primera parte será el nombre original del tipo de control (es decir, en inglés) y la segunda parte será el nombre que le dará el desarrollador a dicho elemento. Ejemplo: label\_Contraseña, textBox\_CorreoUsuario.

### Nombres de eventos

Se usará el estilo PasCal. El nombre se dividirá en tres partes, separadas por un “\_”. La primera y segunda parte serán las mismas que las del nombre del control al que se le aplicará el evento, solo que con la primera palabra iniciando con mayúscula. La tercera parte será el nombre del evento. La parte uno y tres serán puestas con su nombre original en inglés. Ej: Label\_Contraseña\_MouseDown, TextBox\_Usuario\_KeyDown.

Para los botones, en el nombre de los eventos se puede obviar la tercera parte si el evento es Click. Ej: Button\_IniciarSesion, Button\_Registrarse.

### Convenciones de nombres

Estilo caMel: determina que la primera letra de cada palabra deberá ser mayúscula con excepción de la primera palabra cuya primera letra deberá ser minúscula. Ejemplo: listaPortales.

Estilo PasCal: determina que la primera letra de cada palabra deberá ser mayúscula. Ejemplo: CasillaEspecial.

## Documentación

Para la documentación de código, que incluye la documentación de clases, atributos y métodos, se utilizará el formato de documentación de código XML que utiliza el IDE Visual Studio. Cabe mencionar que se aplicarán algunas reglas para la documentación, las cuáles son las siguientes:

* Se permitirá el uso de comentarios normales dentro de los métodos con dos barras (//) siempre y cuando sea estrictamente necesario de acuerdo al criterio del desarrollador.
* Se utilizaran solo las etiquetas del formato XML que sean estrictamente necesarias para cada caso.
* La documentación de código no será aplicable para los archivos XAML de las ventanas de los proyectos WPF.
* Cada objeto y apartado por documentar tendrá una descripción pequeña y concisa de su función.
* Se tratará de evitar en mayor medida la documentación de atributos de clase ya que estos deberán tener nombres lo suficientemente claros y explicativos; además de que los atributos exponen el comportamiento interno de la clase y estos podrían ser confidenciales.

Para más información acerca de este formato XML, todas las etiquetas disponibles y la función de cada una, véase el siguiente enlace:

<https://docs.microsoft.com/es-es/dotnet/csharp/codedoc#walkthrough>

Ejemplo:

/// <summary>

/// Lógica de interacción para la ventana RegistroUsuario.

/// En esta ventana se registra el nuevo usuario.

/// </summary>

public partial class RegistroUsuario : Window

{

/// <summary>

/// Constructor de la ventana.

/// </summary>

public RegistroUsuario()

{

InitializeComponent();

}

/// <summary>

/// Valida que el correo ingresado tenga el formato correspondiente de un correo verdadero.

/// </summary>

/// <returns>Verdadero si el correo tiene un formato válido o falso en caso contrario.</returns>

private Boolean ValidarFormatoCorreo()

{

String expresion = "\\w+([-+.']\\w+)\*@\\w+([-.]\\w+)\*\\.\\w+([-.]\\w+)\*";

if (Regex.IsMatch(textBox\_CorreoElectronico.Text, expresion))

{

if (Regex.Replace(textBox\_CorreoElectronico.Text, expresion, String.Empty).Length == 0)

{

return true;

}

else

{

MessageBox.Show(Properties.Resources.correoInvalido);

return false;

}

}

else

{

MessageBox.Show(Properties.Resources.correoInvalido);

return false;

}

}

}