Convenciones de denominación

- **Pascal case:**
  - Inicial en mayúsculas, demás caracteres en minúsculas.
  - Palabras compuestas se separan utilizando la inicial de cada palabra en Mayúsculas.
- **Camel case:**
  - Toda la palabra en minúsculas.
  - Palabras compuestas se separan utilizando la inicial de cada palabra en Mayúsculas.
- **Reglas generales:**
  - Utilizar nombres nemotécnicos (que representen el contenido y / o función que cumplen).
  - Sustantivos para las variables / properties, expresiones verbales para los métodos.
  - Evitar las abreviaturas.
  - Utilizar acrónimos solo cuando se reconozcan fácilmente (ej.: Rut, Iva)
  -

Elemento	**Convención**	**Ejemplo**
Namespace	Pascal case, separando Namespaces anidados con "."	Ort.Facturacion.Dominio
Clase	Pascal (singular)	Factura, CuentaCorriente
Interface	Pascal Case, prefijada con "I"	IComparable, IValidable, IFigura3D

Enum	Nombre y enumeraciones en Pascal Case y singular.	public enum Color {
		Rojo, Verde, Azul
		}

Variables private	Camel case prefijadas con subguión	private string _apellidoCompleto;
Variables privadas estáticas	Camel case prefijadas con "s_"	private static int s_contadorAlumnos
Properties	Pascal case	public string ApellidoCompleto {get;set;}
Métodos	Pascal case	public void Facturar(){…}
Parámetros	Camel case	public void AplicarDescuento (decimal tasaDescuento) {…}

Variables locales	Camel case.	public double CalcularAreaTotal (){
		double area =0;
		…..
		}

Convenciones de código en C#

Comentarios

- Escribir los comentarios en una línea aparte, no al final de la sentencia.
- No armar recuadros con asteriscos
- Incluir en las clases, interfaces, y métodos públicos los comentarios XML que permitan generar la documentación.
  Los comentarios de los métodos y clases deben ser de la forma:

/// <summary> /// …….. /// …….. /// </summary> public class Factura { …..	/// <summary> /// … /// </summary> /// <returns>…</returns> public decimal CalcularArea() { ….. }

Controlador MVC

Cada controlador tiene la responsabilidad de controlar una clase del modelo y el nombre va en singular, sufijado con “Controller”:

Ej.: PropiedadController

Nota: esta es una regla general, que puede no ser siempre aplicable. Por ejemplo, supongamos que tenemos la clase Proyecto y la clase Tarea, cada proyecto contiene varias tareas y quiero que al visualizar el listado de proyectos pueda cliquear sobre uno en particular y ver sus tareas. En este caso desde ProyectoController obtendría las tareas correspondientes.

Foreach

Para recorrer listas usando el foreach no utilizar *var* para declarar el iterador.

```
foreach (Propiedad item in propiedades)
        {….}
```

Tareas pendientes: TODO

Cuando no esté terminado un método o falta implementarlo ponemos comentario con la palabra ToDo

Para ver todos los pendientes podemos usar: ctrl + \ + T (ctrl + º + T) o en el menú Ver > Lista de Tareas.

Ajustar código

Ctr + K + D permite formatear el código del documento que estamos utilizando.

Después de cada método solo debe ir un espacio en blanco

Más información:

https://docs.microsoft.com/en-us/dotnet/csharp/fundamentals/coding-style/identifier-names

https://docs.microsoft.com/en-us/dotnet/csharp/fundamentals/coding-style/coding-conventions