Estándares de Código

Estándar Personalizado

VILLA LOPEZ ARTURO &
SAAVEDRA MATA DULCE REGINA

Principios de Construcción de Software

Índice

1. Introducción

1.1. Objetivo

En el siguiente documento se establecen algunas normas o reglas que ayudan a la construcción de software para brindar una mejor interpretación.

- Se asegura la calidad del producto de software.
- Facilita el mantenimiento del mismo.
- Simplifica la detección de errores y bugs.
- Facilitar la comprensión para el trabajo en conjunto.

Todos estos conceptos van alineados con el uso de la Metodología Orientada a Objetos en la plataforma Visual Studio.NET

1.2. Alcance

Para este proyecto en específico desarrollador en el lenguaje C# en Visual Studio. Incluye los estándares de nomenclatura, diseño del código, uso de comentarios, así como unas buenas prácticas.

2. Nomenclaturas

Existen varios estilos de estándares para definir los nombres e identificar los elementos del proyecto en C#, entre los cuales destacan 2:

Notación de Pascal (Pascal Case): El primer carácter de todas las palabras se escriben en Mayúscula y todos los demás caracteres en minúsculas. Por ejemplo: CodigoPrueba.

Notación de camello (Camel Case): El primer carácter de todas las palabras excepto de la primera se escriben con Mayúscula y todos los demás en minúsculas. Por ejemplo: codigoPrueba.

2.1. Variables

- Los nombres de variables y de atributos de clase deberán de ser de más de un carácter.

  Una excepción pueden ser los nombres de los iteradores en los ciclos.

Ejemplo:

```
For (int i = 0 ; i  < Variable ; i ++ ) {

}
```

- Usar notación camel case.
- Distinguir los nombres de las variables boleanas.
- Diferenciar entre variables globales y locales.
- No usar palabras reservadas en nombres de variables.

Ejemplo:

Correcto:	**Incorrecto:**
string nombre;	string Nom;
int numeroDeAlumnos;	int NA;

2.2. Constantes

La definición de los constantes se efectúa siempre y cuando el valor no cambie durante el tiempo de ejecución del software

Ejemplo:

```
public const int Meses = 12;
```

2.3. Métodos

La definición del nombre de los métodos este guiado por las siguientes normas:

- Utilizar Camel Case.
- Utilizar verbos o frases descriptivas sobre la acción que tendrá en el programa.
- No se deben usar nombres extraños o engañosos, el nombre debe ser obvio por lo cual no se necesita documentación para explicarlo.

Por Ejemplo: VaciarCubeta(), TirarPelota(), CerrarVentana().

2.4. Clases

La definición del nombre de las clases este guiado por las siguientes reglas:

- Se debe usar un sustantivo en singular, con excepción que la clase represente múltiples cosas.
- Se debe utilizar Pascal Case.
- No se deben utilizar prefijos como C, T o clr.
- Las clases solo representas “cosas” y no “acciones”.

Ejemplo: Empleado, Proveedor, CuentaBancaria.

3. Diseño de código

3.1. Estructura

Estas convenciones tienen la finalidad siguiente:

- Facilitar la lectura, modificación y mantenimiento del código.
- Dar un aspecto coherente al código, de tal manera que el lector centre su atención en el contenido y no en el diseño.
- Permitir la lectura y comprensión con mayor rapidez.
- Utilizar una sola instrucción por línea.
- Utilizar una sola declaración por línea.
- Si a las líneas no se les aplica una sangría automática, aplicarla con una tabulación (cuatro espacios).
- Utilizar un espacio en blanco para separar grupos lógicos de códigos.
- Utilizar paréntesis para cláusulas de una expresión, así se hace más evidente.

Ejemplo:

```
if ((numero1 < numero2) && (numero1 < numero3))
{
   //Ingresar código
}
```

4. Uso de Comentarios

Los comentarios deberán seguir las siguientes convenciones:

- No usar comentarios para cada línea y/o variable.
- Usar de preferencia // en lugar de /* ……*/
- Usar los menos comentarios posibles.
- Usarlos para explicar lógica compleja.
- Usarlos para explicar asignaciones a las variables.
- Tener buena ortografía, puntuación y gramática.

5. Buenas prácticas

1. Evitar escribir métodos muy largos. Se recomienda que un método deba tener entre 1 a 25 líneas de código, si se llega a tener más de 25 líneas se deberá refactorizar.

2. Aplicar las sangrías apropiadas a los segmentos de código que lo ameriten, por ejemplo, los ciclos, los selectivos y los condicionales.
3. Respetar las nomenclaturas definidas para las variables, clases y métodos.
4. Compartir el estándar con todo el equipo.
5. Usar solo los comentarios estrictamente necesarios.
6. Dejar espacios entre líneas de código y secciones diferentes.

Referencias

Este Estándar de Código esta basado en el siguiente estándar de C#, escrito por José Luis Clemente Montes. https://es.calameo.com/read/00443793525640a944420