**PRIMERA ETAPA PROYECTO FINAL**

Sistema del Servicio Social de la FEI

**Equipo 5: Dulce Regina Saavedra Mata, Arturo Villa López, José María Grijalva Montes**

**Principios de Construcción**

## Estándar de Código

## Introducción

## 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

## 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.

## 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.

## 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 pascal 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;

## Constantes

La definición de los constantes se efectúa siempre y cuando el valor no cambie durante el tiempo de ejecución del software, todo en mayúsculas.

**Ejemplo:**

public const int MESES = 12;

## 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().

## 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.

## Diseño de código

## 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*

}

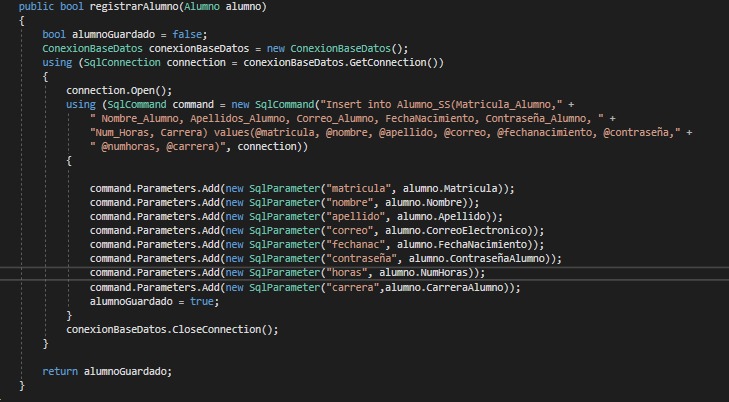
## Uso de Comentarios

Los comentarios deberán seguir las siguientes convenciones:

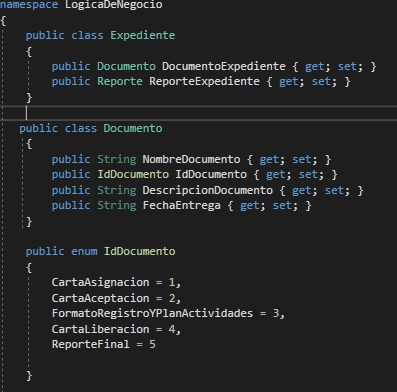
* No usar comentarios para cada línea y/o variable.
* Usar de preferencia // en lugar de /\* …...\*/
* Usar los menos comentarios posibles.
* Usarlos para explicar lógica compleja.
* Usarlos para explicar asignaciones a las variables.
* Tener buena ortografía, puntuación y gramática.

## Buenas prácticas

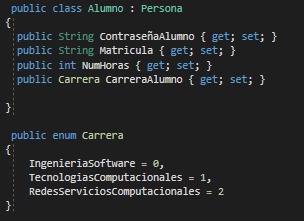
1. Evitar escribir métodos muy largos. Se recomienda que un método deba tener entre 1 a 25 líneas de código, si se llega a tener más de 25 líneas se deberá refactorizar.



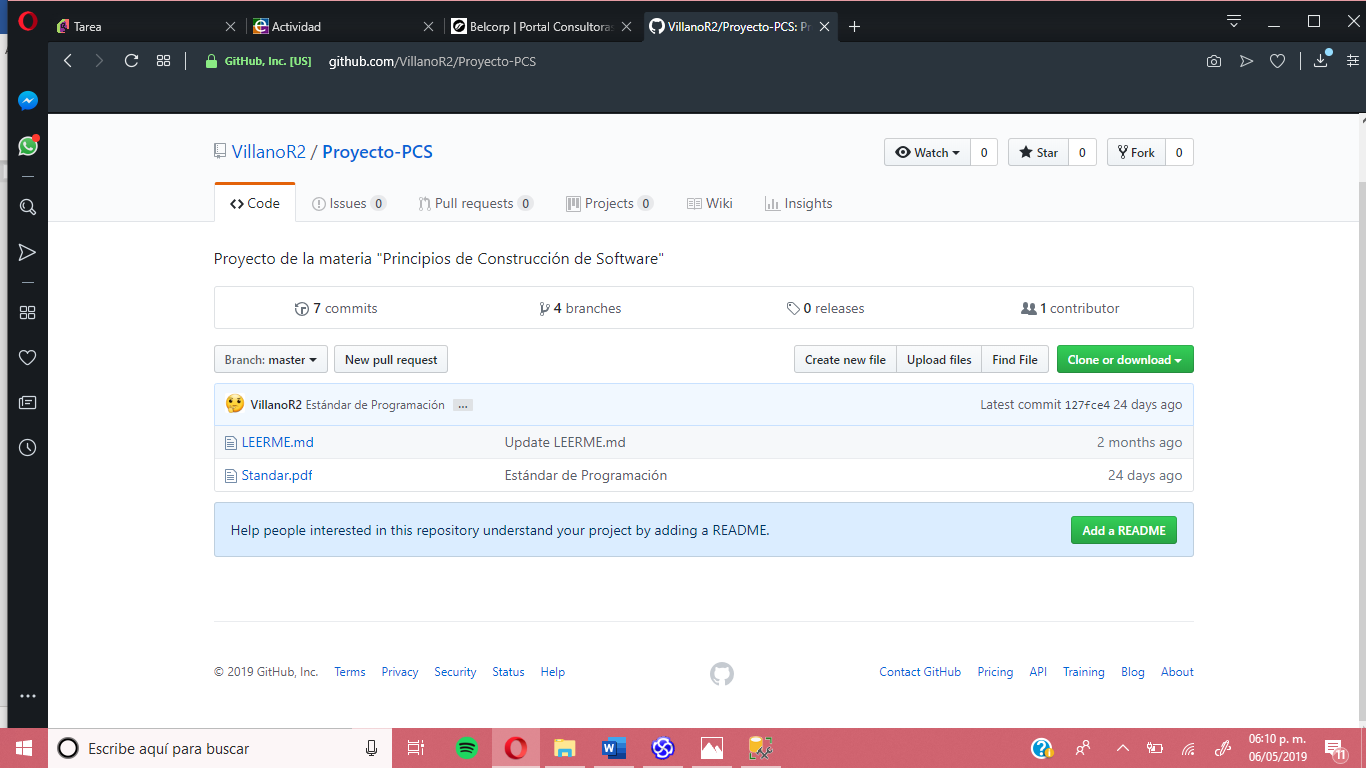
1. Aplicar las sangrías apropiadas a los segmentos de código que lo ameriten, por ejemplo, los ciclos, los selectivos y los condicionales.



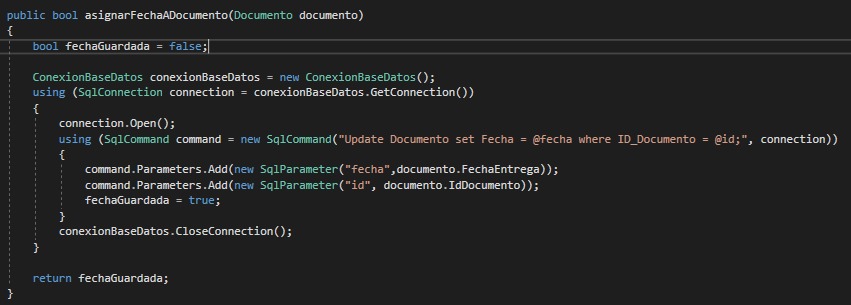
1. Respetar las nomenclaturas definidas para las variables, clases y métodos.



1. Compartir el estándar con todo el equipo.



1. Usar solo los comentarios estrictamente necesarios.  
   **No se han manejado comentarios hasta el momento.**
2. Dejar espacios entre líneas de código y secciones diferentes.



## Modelos Dominio (Visto Bueno Diseño)

## 

Modelo 1 Modelo De Dominio

## Modelo Entidad Relación

## 

Modelo 2 Modelo Entidad Relación

## Prototipos

## 

Pantalla 1 Pantalla de Inicio de sesión

## 

Pantalla 2 Pantalla principal del director

## 

Pantalla 3 Pantalla para registro de coordinador

## 

Pantalla 4 Pantalla principal director con ventana de “cerrar sesión”.

## 

Pantalla 5 Pantalla registro de alumno

## 

Pantalla 6 Pantalla alumno pendiente de validación

## 

Pantalla 7 Pantalla principal alumno

## 

Pantalla 8 Pantalla alumno documentos subidos

## 

Pantalla 9 Pantalla alumno solicitar servicio social

## 

Pantalla 10 Pantalla alumno principal

## 

Pantalla 11 Pantalla principal coordinador

## 

Pantalla 12 Pantalla Coordinador Alumno

## 

Pantalla 13 Pantalla Registro de Técnico

## 

Pantalla 14 Pantalla coordinador con ventana de cerrar sesión

## 

Pantalla 15 Realizar calendarización

## 

Pantalla 16 Principal Coordinador

## 

Pantalla 17 Registrar institución

## 

Pantalla 18 Registrar proyecto

## 

Pantalla 19 Registrar encargado

## 

Pantalla 20 Asignar Proyecto

## 

Pantalla 21 Coordinador principal con cerrar sesión

## 

Pantalla 22 Principal Técnico

## 

Pantalla 23 Subir documento

## 

Pantalla 24 Explorador de Archivos

## 

Pantalla 25 Explorador de archivos

## Propuesta de Capas

## Nuestra propuesta para el sistema del Servicio Social se encuentra dividido en 3 capas:

## Acceso a Datos:

## En esta capa se establece la conexión con la base de datos Servicio Social.

## Lógica del Negocio

## En esta capa se encuentran codificadas todas las clases planteadas en el modelo de dominio así como sus relaciones.

## Se realizará la lectura y escritura en la base de datos.

## Servicio Social App

## En esta capa estará todo aquello que será visible para el usuario.

## Pantallas en las que podrá ingresar datos y visualizar.

## Conclusiones

## Arturo:

## Puedo concluir que con el avance que se logró hasta el día 07/05/2019, se tiene una mejor comprensión de lo podría llegar a convertirse el software final, gracias a el uso de pruebas unitarias se han detectado errores que al momento de compilar y ejecutar el software se pasaban por alto, añadiéndole un manejo de excepciones en diversos casos, se han solucionado algunos problemas que generaba al probar el código, también llegue a la conclusión que algunos fragmentos de código factibles al cambio y puede generar algunos problemas si no se maneja de manera unitaria los fragmentos de código que no dependen entre si.

## Regina:

## Con el avance que el proyecto tiene el día de hoy, se puede observar lo importante que es tener los modelos hechos en diseño, totalmente definidos ya que son la base para tomar muchas de las decisiones en el código y permite saber que métodos y atributos se encuentran en cada clase y como se encuentran relacionadas entre sí, lo que hace que la codificación sea una actividad casi de traducción de estos métodos. También noté que los prototipos son un gran apoyo ya que permite ver las ideas un poco más aterrizadas y poder tener un acercamiento a lo que será el proyecto final, y llegar a esa meta realizando el código.

## Referencias

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