TEcnologias utilizadas en el desarrollo de La US

Front End: HTML, ANGULAR JS , BOOTSTRAP, JQuery,UI\_BOOTSTRAP

BACK END: WEB API, ENTITY FRAMEWORK, SQL SERVER 2016 , Visual Studio 2017

**CODING STANDARD**

A continuación están los estandares y las convenciones de nombrado que se deberán tener en cuenta al escribir el código:

• Usar Pascal Casing para los nombres de las clases, los métodos y sus atributos (Primer letra de cada palabra en mayúsculas).

• Usar un espaciado de 4 unidades (configurar tabulación para 4 espaciados).

• No usar mayúsculas en el nombrado de variables o constantes (por ejemplo: FECHA).

• No usar abreviaciones salvo aquellas excepciones del vocablo usualmente usadas para nombres o abreviaciones. (Por ejemplo: html, uri, xml, uml, ftp, etc...).

• No usar guiones bajos “\_” en la unión de 2 o mas palabras, usar Pascal Casing.

• Usar sustantivos o frases formadas por sustantivos para las clases. (ejemplo: Empleado, FacturaDigital, etc...).

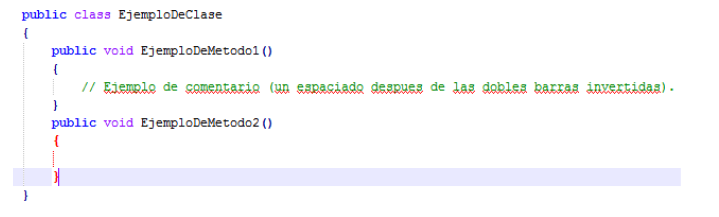
• Usar de prefijo para las interfaces la letra “I”. (ejemplo: IController).

• Usar un simple espaciado luego de las doble barras invertidas de comentarios, por ejemplo:

“// Comentario”.

• Usar alineacion de llaves, verticalmente “{ }”.

Ejemplo:



**Convenciones JavaScript**

**Longitud de una lí­nea**

El lí­mite de las lí­neas de código deberán ser de 80 caracteres, esto ayudará a una mejor lectura del código. Es seguro que tu editor de texto o IDE soporta esta caracterí­stica.

**Comillas:** En ciertos casos se desea tener comillas dentro de un texto, y tendrán otras como delimitadores externos del texto.

***Ejemplo***

**const** foo = ‘bar’;  
console.log(‘bar’);  
**function**(foo, bar){  
 **const** x = ‘hi’ + foo + ‘;’ + ‘bar’;  
}

**Llaves**

La llave de apertura deberá ir en la misma lí­nea de la sentencia. Si colocas en la siguiente lí­nea, en algunos casos muy particulares se puede producir algún error. Más adelante se explica dicho error en el uso de punto y coma en las sentencias.

***Recomendado***

// control flow stament  
**if** ( true ) {  
 //codes goes here  
}

//Anonymous function declaration  
**function** ( args ) {  
 **return** true;  
}

//Named function declaration  
**function** **foo**() {  
 **return** true;  
}

//Anonymous function expression  
**const** bar = **function** ( args ) {  
 **return** true;  
}

//Arrow function  
**const** bar = (arg) => {  
 **return** true,  
};

**const** bar = ( args ) => { true };

**const** bar = ( args ) => true;

**Conclusiones finales**

Recordemos seguir las buenas prácticas de desarrollo para evitar horas innecesarias en reescribir un código mal escrito, llevando con ello la reducción de la productividad de trabajo.

Varios ejemplos y definiciones son tomados de algunas referencias bibliográficas.