Sistema de Asistencia

Documento de Estándares de Programación

Versión 1.0

Historia de Revisión

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Historial de revisiones | | | | | |
| Ítem | Fecha | Versión | Descripción | Equipo |
| 1 |  | 1.0 | Versión Final. |  |

Contenido

[1. Objetivo 4](#_Toc75897508)

[2. Arquitectura de Software 4](#_Toc75897509)

[3. Patrón Arquitectónico Arquitectura de 3 Capas 5](#_Toc75897510)

[4. Definición de Controles 5](#_Toc75897511)

[4.1 Tipo de datos 5](#_Toc75897512)

[4.2 Prefijo para el Control 5](#_Toc75897513)

[4.3 Nombre descriptivo del Control 6](#_Toc75897514)

[4.4 Declaración de variables, atributos y objetos (Acceso Datos) 7](#_Toc75897515)

[4.5 Declaración de variables, atributos y objetos (Entidad) 8](#_Toc75897516)

[4.6 Declaración de variables, atributos y objetos (Negocio) 11](#_Toc75897517)

[4.7 Declaración de clases (Acceso Datos) 12](#_Toc75897518)

[4.8 Declaración de clases (Entidad) 13](#_Toc75897519)

[4.9 Declaración de clases (Negocio) 14](#_Toc75897520)

[4.10 Declaración de funciones (Acceso Datos) 15](#_Toc75897521)

[4.11 Declaración de funciones (Negocio) 18](#_Toc75897522)

[4.12 Control de versiones de código fuente 21](#_Toc75897523)

[4.13 Controles ADO.NET 21](#_Toc75897524)

[5. Clases. 23](#_Toc75897525)

[6. Métodos, Procedimientos y Funciones definidos por el Usuario. 23](#_Toc75897526)

[8. Formato al código 24](#_Toc75897527)

[8.1 Crear cadenas para MsgBox, InputBox y consultas SQL 24](#_Toc75897528)

[9. Arquitectura de Desarrollo 25](#_Toc75897529)

[9.1 Capa Presentación 25](#_Toc75897530)

[9.2 Capa Lógica de Negocios 25](#_Toc75897531)

[9.3 Capa de Acceso a Datos 26](#_Toc75897532)

[10. Beneficios 26](#_Toc75897533)

[11. Conclusiones 26](#_Toc75897534)

Estándares de Programación

# Objetivo

Reglamentar la forma en que se implementara el código fuente del proyecto, pasando, por las variables, controles, ficheros, archivos y todo aquello que este implicado en el código.

Mejorar y uniformizar a través de las reglas que se proponen, el estilo de programación que tiene un programador.

* Los nombres de variables serán mnemotécnicos con lo que se podrá saber el tipo de dato de cada variable con solo ver el nombre de la variable.
* Los nombres de variables serán sugestivos, de tal forma que se podrá saber el uso y finalidad de dicha variable o función fácilmente con solo ver el nombre de la variable.
* La decisión de poner un nombre a una variable o función será mecánica y automática, puesto que seguirá las reglas definidas por nuestro estándar.
* Permite el uso de herramientas automáticas de verificación de nomenclaturas.

Por tanto, se seguirán dichos patrones para un entendimiento legible del código y para facilitar el mantenimiento de este

# Arquitectura de Software

Comprende la definición y organización de los elementos arquitectónicos de una solución de software, sus interacciones y sus restricciones. Dentro de las tareas a realizar dentro de la arquitectura se encuentran: Definición de los diferentes componentes que integran la solución propuesta, analizando las diferentes interacciones que se presentan entre los mismos. Análisis de componentes pre-existentes en el mercado, que pueden ser reutilizados en la arquitectura que se propone. Análisis del modelo propuesto desde diferentes perspectivas, para validar características tales como performance, concurrencia y aspectos que deberá tener el sistema al ser implementado en redes de comunicaciones. Definir el patrón arquitectónico que logre implementar adecuadamente las características funcionales y no funcionales definidas durante la toma de requerimientos.

# Patrón Arquitectónico Arquitectura de 3 Capas

El patrón arquitectónico en 3 capas es una especialización del patrón Aplicación en Capas. Esto define que la solución debe de dividirse en tres capas lógicas: presentación, lógica de negocio y acceso a datos, con esto se busca especificar el conjunto de responsabilidades de cada una de las capas y los componentes que los conformarán.

**Presentación:** debe provee la interfaz para el usuario. Las tecnologías que se utilizan en.net son Win Forms para aplicaciones de cliente inteligente y ASP.NET para aplicaciones web.

**Lógica de Negocio:** implementa la funcionalidad de la lógica de negocio. Esta capa está conformada de componentes desarrollados en cualquiera de los lenguajes de.net

**Acceso a Datos:** provee el acceso a los repositorios de información persistente y/o sistemas externos. La tecnología utilizada para desarrollar esta capa es ADO.NET, se pueden también utilizar stored procedures y XML.

**Capa de Entidad:** Aunque aparentemente es una cuarta capa realmente no lo es, esta capa se encarga de contener todos aquellos objetos (clases) que representan al negocio, y esta es la única que puede ser instanciada en las 3 capas anteriores, es decir, solo ella puede tener comunicación con el resto, pero su función se limita a únicamente ser un puente de transporte de datos. Esta capa complementa a la Capa de Negocio

# Definición de Controles

Para poder determinar el nombre de un control dentro de cualquier aplicación de tipo visual, se procede a identificar el tipo al cual pertenece y la función que cumple dentro de la aplicación.

## Tipo de datos

|  |  |  |
| --- | --- | --- |
| **Tipo de variable** | **Mnemónico** | **Descripción** |
| Byte | by | Entero de 8 bits sin signo. |
| Integer | in | Entero de 32 bits con signo. |
| Char | ch | Un carácter UNICODE de 16 bits |
| String | st | Cadena de caracteres |
| Date | dt | Formato de fecha/hora |
| Boolean | bl | Valor lógico: verdadero y falso |
| Float | fl | Coma flotantes, 11-12 dígitos significativos. |
| Double | db | Coma flotante, 64 bits (15-16 dígitos significativos) |
| Object | ob | Objeto genérico |

## Prefijo para el Control

El prefijo del control será determinado mediante tres caracteres que estarán conformados por las consonantes más representativas del control, es así, por ejemplo; el control Button, estará asociado al prefijo btn.

## Nombre descriptivo del Control

Formado por la descripción de la función que lleva a cabo el control, esta debe ser descrita en forma específica y clara.

|  |  |  |
| --- | --- | --- |
| **Control** | **Prefijo** | **Ejemplo** |
| Button | btn | btn\_nombre |
| Calendar | cal | cal\_nombre |
| CheckBox | chk | chk\_nombre |
| CheckBoxList | chkl | chkl\_nombre |
| ComboBox | cbo | cbo\_nombre |
| DataGrid | dg | dg\_nombre |
| DatGridView | dgv | dgv\_nombre |
| DataList | dlst | dlst\_nombre |
| DateTimePicker | dtp | dtp\_nombre |
| DialogmColorDialog,FontDialog | dlg | dlg\_nombre |
| GroupBox | gbx | gbx\_nombre |
| Image | Img | img\_nombre |
| Label | lbl | lbl\_nombre |
| LinkLabel | lbl | lbl\_nombre |
| ListBox | lst | lst\_nombre |
| ListView | mnu | mnu\_nombre |
| Panel | pnl | pnl\_nombre |
| PrintDialog | pdg | pdg\_nombre |
| PrintDocument | pdt | pdt\_nombre |
| ProcessBar | pgb | pgb\_nombre |
| PictureBox | pic | pic\_nombre |
| RadioButton | rad | rad\_nombre |
| RadioButtonList | rbl | rbl\_nombre |
| RichTextBox | rtb | rtb\_nombre |
| Repeater | rep | rep\_nombre |
| TabControl | tab | tab\_nombre |
| TextBox | txt | txt\_nombre |
| Timer | tmr | tmr\_nombre |
| Validators | val | val\_nombre |
| ValisationsSummary | vals | vals\_nombre |

## Declaración de variables, atributos y objetos (Acceso Datos)

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | BOOL Autentificacion |
| **Descripción** | Autentificacion de la base de datos de SQL Server |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | SQLCONECTION URL |
| **Descripción** | Representa una conexión a la base de datos de SQL Server |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | SQLCONECTION sqlCnx |
| **Descripción** | Representa una conexión abierta a la base de datos de SQL Server |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | SQLDATAREADER Resultado |
| **Descripción** | Proporciona una forma de leer un flujo de filas solo hacia adelante desde una base de datos de SQL Server. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | DATATABLE Tabla |
| **Descripción** | Representa una tabla de datos. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | SQLCOMMAND comando |
| **Descripción** | Representa una instrucción Transact-SQL o un procedimiento almacenado para ejecutar en una base de datos de SQL Server. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

## Declaración de variables, atributos y objetos (Entidad)

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | INT ID\_Administrador |
| **Descripción** | Identificación del administrador |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | STRING Nombre |
| **Descripción** | Nombre del administrador |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | STRING Apellido |
| **Descripción** | Apellido del Administrador |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | STRING DNI |
| **Descripción** | DNI del Administrador. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | INT ID\_ Trabajador |
| **Descripción** | Identificación del Trabajador |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | STRING Nombre |
| **Descripción** | Nombre del Trabajador |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | STRING Apellido |
| **Descripción** | Apellido del Trabajador |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | STRING DNI |
| **Descripción** | DNI del Trabajador. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | INT ID\_Asistencia |
| **Descripción** | Identificación de la Asistencia |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | STRING Fecha |
| **Descripción** | Fecha de la Tabla Asistencia |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | STRING Hora |
| **Descripción** | Hora de Asistencia,Tabla Asistencia. |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | STRING Estado |
| **Descripción** | Estado de la Asistencia |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

## Declaración de variables, atributos y objetos (Negocio)

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | ClsHojaDatos objHoja |
| **Descripción** | Objeto Hoja de Acceso a Datos |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | ClsTablaAsistenciaDatos objAsistencia |
| **Descripción** | Objeto Tabla Asistencia de Acceso a Datos |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | ClsTrabajadorDatos objTrabajador |
| **Descripción** | Objeto de Trabajador de Acceso a Datos |
| **Observaciones** | En la declaración de variables o atributos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

## Declaración de clases (Acceso Datos)

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | PUBLIC ClsTrabajadorDatos |
| **Descripción** | Encargada de acceder a los mismos. Reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio. |
| **Observaciones** | En la declaración de clases no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Private Class Empleado  Indica una clase Empleado |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | PUBLIC ClsConexionDatos |
| **Descripción** | Nos permitirá hacer conexión con la Base de Datos SQL. |
| **Observaciones** | En la declaración de clases no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Private Class Empleado  Indica una clase Empleado |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | PUBLIC ClsHojaDatos |
| **Descripción** | Encargada de acceder a los mismos. Reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio. |
| **Observaciones** | En la declaración de clases no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Private Class Empleado  Indica una clase Empleado |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | PUBLIC ClsTablaAsistenciaDatos |
| **Descripción** | Encargada de acceder a los mismos. Reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio. |
| **Observaciones** | En la declaración de clases no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |
| **Ejemplo** | Private Class Empleado  Indica una clase Empleado |

## Declaración de clases (Entidad)

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | PUBLIC ClsHojaNegocio |
| **Descripción** | Donde se reciben las peticiones del usuario y se envían las respuestas tras el proceso. |
| **Observaciones** | En la declaración de clases no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | PUBLIC ClsTablaAsistenciaNegocio |
| **Descripción** | Donde se reciben las peticiones del usuario y se envían las respuestas tras el proceso. |
| **Observaciones** | En la declaración de clases no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | PUBLIC ClsTrabajadorNegocio |
| **Descripción** | Donde se reciben las peticiones del usuario y se envían las respuestas tras el proceso. |
| **Observaciones** | En la declaración de clases no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

## Declaración de clases (Negocio)

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | PUBLIC ClsHojaNegocio |
| **Descripción** | El nombre de las clases tendrá una longitud máxima de 30 caracteres y las primeras letras de todas las palabras estarán en mayúsculas. Tipo se refiere a si la clase será: Private, Public o Protected. |
| **Observaciones** | En la declaración de clases no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | PUBLIC ClsTablaAsistenciaNegocio |
| **Descripción** | El nombre de las clases tendrá una longitud máxima de 30 caracteres y las primeras letras de todas las palabras estarán en mayúsculas. Tipo se refiere a si la clase será: Private, Public o Protected. |
| **Observaciones** | En la declaración de clases no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | PUBLIC ClsTrabajadorNegocio |
| **Descripción** |  |
| **Observaciones** | En la declaración de clases no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ]. * Caracteres tildados: á, é, í, ó, ú. |

## Declaración de funciones (Acceso Datos)

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | SqlConnection establecerConexion() |
| **Descripción** | Establecer conexión con la base de datos SQL. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | ClsConexionDatos getInstancia() |
| **Descripción** | Si la conexión de datos es null no retornara la instancia ClsConexionDatos |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | DataTable Listar() |
| **Descripción** | Listara todos los datos que contiene la tabla Trabajador de la base de dato SQL |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | String Insertar(ClsTrabajadorEntidad ObjTrabajador) |
| **Descripción** | Aqui podremos insertar un trabajador ejecutando el procedimiento almacenado y ejecuta una instrucción en la conexión y devolverá si se pudo insertar correctamente. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | String Modificar(ClsTrabajadorEntidad ObjTrabajador) |
| **Descripción** | Aquí podremos modificar un Trabajador usando su ID\_Trabajador e ingresar los parámetros modificados y ejecuta una instrucción en la conexión y devolverá si se pudo modificar correctamente. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | String Eliminar(int Id) |
| **Descripción** | Podremos eliminar un Trabajador haciendo uso de ID\_Trabajador luego ejecuta una instrucción en la conexión y devolverá si se pudo eliminar correctamente. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | String Verificar(String valor) |
| **Descripción** | Verifica si el trabajador no este registrado.Se ejecuta una instrucción en la conexión y devolverá si se existe o no existe el trabajador. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | Buscar(string Busqueda) |
| **Descripción** | Podremos buscar a un trabajador con su ID\_Trabajador y se ejecuta una instrucción en la conexión y envia el parámetro “Busqueda”. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | String Insertar(ClsTablaAsistenciaEntidad ObjAsistencia) |
| **Descripción** | Aquí Insertaremos la Tabla Asistencia. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | Int FechaActualC() |
| **Descripción** | Nos devolverá la primera columna de la Tabla Asistencia que se creo recientemente. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | DataTable Buscar\_Fecha(string Busqueda) |
| **Descripción** | Podremos buscar una Tabla Asistencia por su fecha, enviara la propiedad. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | String Insertar(ClsHojaEntidad ObjAsistencia) |
| **Descripción** | Al agregar un trabajador a la lista este tendré su estado como preterminado “falta”. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | String Asistencia\_ACT(int ID\_Asistencia,int ID\_Trabajador,string Hora) |
| **Descripción** | El Trabajador al marcar que asistió se modificara el parámetro estado por “Asistio” y la hora en que marco la Hoja de Asistencia. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

## Declaración de funciones (Negocio)

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | String Insertar(int id\_Asistencia, int id\_Trabajador, string estado, string hora) |
| **Descripción** | Ingresaremos los parámetros requeridos para poder hacer uso del método “Insertar” de Accedo a Datos. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | String Asistencia\_ACT(int ID\_Asistencia, int ID\_Trabajador, string Hora) |
| **Descripción** | Ingresaremos los parámetros requeridos para poder hacer uso del método “Asistencia\_ACT” de Accedo a Datos. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | String Verificar\_Hoja(int ID\_Asistencia, int ID\_Trabajador) |
| **Descripción** | Ingresaremos los parámetros requeridos para poder hacer uso del método “Verificar\_Hoja” de Accedo a Datos. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | String Insertar(string Fecha, string Hora) |
| **Descripción** | Ingresaremos los parámetros requeridos para poder hacer uso del método “Insertar” de Accedo a Datos. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | Int FechaActualC() |
| **Descripción** | Ingresaremos los parámetros requeridos para poder hacer uso del método “FechaActualC” de Accedo a Datos. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | DataTable Buscar\_Fecha(string Busqueda) |
| **Descripción** | Ingresaremos los parámetros requeridos para poder hacer uso del método “Buscar\_Fecha” de Accedo a Datos. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | DataTable Buscar\_EncontrarFecha(int Busqueda) |
| **Descripción** | Ingresaremos los parámetros requeridos para poder hacer uso del método “Buscar\_EncontrarFecha” de Accedo a Datos. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | DataTable Listar() |
| **Descripción** | Ingresaremos los parámetros requeridos para poder hacer uso del método “Listar” de Accedo a Datos. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | String Insertar(string Nombre, string Apellido, string Dni, string Rol, string Clave) |
| **Descripción** | Ingresaremos los parámetros requeridos para poder hacer uso del método “insertar” de Accedo a Datos. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | String Modificar(int ID, string Nombre, string Apellido, string Dni, string Rol, string Clave) |
| **Descripción** | Ingresaremos los parámetros requeridos para poder hacer uso del método ”Modificar” de Accedo a Datos. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | String Eliminar(int ID) |
| **Descripción** | Ingresaremos los parámetros requeridos para poder hacer uso del método “Eliminar” de Accedo a Datos. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Sintaxis** | DataTable Loguin(string Usuario, string clave) |
| **Descripción** | Ingresaremos los parámetros requeridos para poder hacer uso del método “Loguin” de Accedo a Datos. |
| **Observaciones** | En la declaración de objetos no se deberá utilizar caracteres como:   * Letra Ñ o ñ. * Caracteres especiales ¡, ^, #, $, %, &, /, (, ), ¿, ‘, +, -, \*, {, }, [, ], \_. * Caracteres tildados: á, é, í, ó, ú. |

## Control de versiones de código fuente

Cada modificación realizada será guardada de la forma:

|  |  |
| --- | --- |
| **Título** | **Descripción** |
| **Formato** | TrabajoGrupal\_30\_06\_2021\_10pm.zip |
| **Descripción** | Se generarán archivos con las siguientes extensiones:.zip o .rar. Por ejemplo: WSTENNIS\_20070421\_2056.zip |

## Controles ADO.NET

Objetos de ADO.NET Aunque hay miles de objetos disponibles como parte de .NET, es probable que se use ADO.NET como parte de las aplicaciones, por lo tanto algunos estándares para nombrar los objetos de ADO.NET más comunes. A continuación, se listan los prefijos que se utiliza:

|  |  |
| --- | --- |
| **Componente** | **Prefijo** |
| **DataSet** | Ds |
| **DataTable** | Dt |
| **DataView** | Dv |
| **DataRow** | Drw |
| **Connection\*** | Cnn |
| **Command\*** | Cmd |
| **DataAdapter\*** | Da |
| **CommandBuilder\*** | Bld |
| **DataReader\*** | Dr |

Ejemplos: de declaración de los objetos ADO.net

* drEmps As New SqlDataReader()
* drCust As New SqlDataReader()
* dsEmps As DataSet
* dsCust As DataSet

# Clases.

El nombre de las clases debe ser auto descriptivo de manera que no se requiera, en lo posible, entrar al código de la función para saber qué es lo que realiza.

El estándar para nombres de clases es usar iniciar con las siglas **cls,** la cual debe estar escrita en minúscula seguido del nombre que identifica la clase, la primera letra del nombre debe iniciar con mayúscula

* Ejemplos: clsCuenta, clsEmpleado, clsFactura

**Nota:**

* No se hará uso de los caracteres: Espacio en blanco " ", Caracter de subrayado "\_".

# Métodos, Procedimientos y Funciones definidos por el Usuario.

El nombre de las funciones y procedimientos debe ser auto descriptivo de manera que no se requiera, en lo posible, entrar al código de la función para saber qué es lo que realiza.

***verbo-Sustantivo***

El estándar para nombres de procedimiento es usar un Verbo que describa la acción realizada seguida por un sustantivo (objeto sobre el cual actúa). Se recomienda:

* Usar un nombre que represente una acción y un objeto. El nombre del procedimiento debe indicar qué hace el procedimiento a... o qué hace el procedimiento con....
* El verbo debe estar en infinitivo.
* Ser consistente en el orden de las palabras. Si se va a usar ***verboNombre***, siempre usar ***verboNombre***.
* Ser consistente en los verbos y sustantivos usados. Por ejemplo, si tiene un procedimiento ***asignarNombre***, en vez de ***colocarNombre***.
* Para la acción **modificar cuentas del cliente** se define:

**modificar Cuenta**

Verbo: modificar

Sustantivo: Cuenta

**Nota:**

* No se hará uso de los caracteres: Espacio en blanco " ", Caracter de subrayado "\_".
* La nomenclatura de argumentos o parámetros pasados a los procedimientos/funciones así como para valores devueltos por funciones sigue las mismas convenciones que la nomenclatura para variables.

# Formato al código

Como muchos programadores usan todavía pantallas VGA, hay que ajustarse al espacio de la pantalla en la medida de lo posible y hacer que el formato del código siga reflejando la estructura lógica y el anidamiento. Estos son algunos indicadores:

* Los bloques anidados estándar, separados por tabuladores, deben llevar una sangría de dos espacios (predeterminado) como mínimo. Los comentarios deben ser alineados todos iguales en la medida de lo posible
* El comentario del esquema funcional de un procedimiento debe llevar una sangría de un espacio.
* Las instrucciones de nivel superior que siguen al comentario del esquema deben llevar una sangría de un tabulador, con cada bloque anidado separado por una sangría de un tabulador adicional.

## Crear cadenas para MsgBox, InputBox y consultas SQL

Cuando esté creando una cadena larga, use el guion bajo para la continuación de línea romper en múltiples líneas de código una línea lógica, de forma que pueda leer o depurar la cadena fácilmente. Esta técnica es especialmente útil cuando se muestra un cuadro de mensaje (MsgBox), un cuadro de entrada (InputBox), o cuando se crea una cadena SQL.

Dim Msj As String

Msj = "Esto es un párrafo q

ue estará en un" \_

& " cuadro de mensajes. El texto está separado en" \_

& " varias líneas de código en el código de origen, " \_

& "lo que facilita al programador la tarea de leer y depurar."

MsgBox Msj

# Arquitectura de Desarrollo

Para el desarrollo de las nuevas aplicaciones de desarrollo se utilizará una arquitectura de 3 capas.

**Capa Presentación:**

Provee la interfaz del usuario. Las tecnologías que se utilizan en .Net son win forms para aplicaciones de cliente inteligente y ASP.Net para aplicaciones web.

**Capa Lógica de Negocios:**

implementa la funcionalidad de la lógica de negocio. Esta capa está compuesta por componentes desarrollados en cualquier lenguaje de .NET.Generalmente se utiliza

COM+ (Enterprise Services) para brindar el soporte transaccional, escalabilidad, etc.

**Capa de Acceso a Datos:**

Provee el acceso a los repositorios de información persistente y/o sistemas externos. La tecnología utilizada para desarrollar esta capa es ADO.NET, es común utilizar stored procedure y XML.

## Capa Presentación

**Componentes de Interfaz de Usuario (UIC)**

En aplicaciones de cliente inteligente se utilizan los componentes del namespace System.Windows.Forms En aplicaciones WEB, los componentes corresponden al namespace System.Web.UI.

**Componentes de Proceso de Interfaz de Usuario (UPC)**

Componentes personalizados, frameworks de navegación, librerías de funciones comunes, que faciliten el desarrollo de la interfaz de usuario.

## Capa Lógica de Negocios

**Componentes de Negocios (BC)**

Son los componentes que implementan las reglas de negocio. Se implementan en clases.net que puedan utilizar los servicios de COM+ y/o Remoting.

**Componentes de Workflow de Negocios (BW)**

Son componentes que representan las actividades de un proceso de negocio. Se implementan las clases.net en conjunto con algún motor de flujos de trabajo u orquestador de procesos.

**Entidades del Negocio (BE)**

Son contenedores de datos. Representan las entidades persistentes del negocio. Se transportan entre las capas. Pueden implementarse en clases.net, DataSets tipados o DataSets sin tipar.

**Interfaces de Servicio (SI)**

La lógica de negocios se expone mediante servicios. Cada servicio posee una interfaz, es decir un protocolo que indica cómo debe ser invocado. Puede implementarse utilizando Web Services o interfaces.net.

## Capa de Acceso a Datos

**Componentes de Accesos a Datos (DAC)**

Abstraen a la lógica de negocios de los detalles específicos de la persistencia. Minimizan el impacto en caso de cambio de motor de base de datos o la representación de los datos. Se implementan con ADO.NET y generalmente se construye un set de clases utilitarias para facilitar el desarrollo y abstraer la utilización de un proveedor específico (SQL Client, ORACLE Client, etc.).

**Agentes de Servicio (SA)**

Se encargan de manejar el acceso a aplicaciones y servicios externos. Las tecnologías utilizadas son muy variadas y dependen de la aplicación externa (Sockets, Biztalk, Web Services.)

# Beneficios

* La documentación hace más legible un programa.
* Al documentar bien un programa desde un principio se evita que para cada modificación deba estudiarse profundamente el funcionamiento del programa, redescubriendo todo lo no documentado, con la ventaja adicional de que generalmente quién modifica el programa no es siempre quién lo escribió.
* Facilita la reutilización de módulos y rutinas desde cualquier otro programa o el mismo.
* Ayuda a determinar cuándo debe ser reescrito un código. Si existen problemas para explicar el código con un comentario, probablemente el código esté mal escrito.

# Conclusiones

* Una buena programación e implementación legible, solo se logra usando y llevando de la mano un buen estándar o patrón de programación.
* Es muy importante que el programador tenga un conocimiento previo del estándar o en su defecto que lea el documento para prever diferencias.
* Al documentar se obtienen dos cosas fundamentales, un documento legible y segundo una buena base para los futuros desarrollos de mantenimiento del código.