Universidad Don Bosco.
Manual de Instalación y Configuración
SAPH.

Índice.

1. Módulo Administrativo.

Para poder instalar el Módulo Administrativo es necesario que la computadora en donde se desee instalar cumpla con los siguientes requerimientos:

- Una computadora con Windows 2000, Windows 98, Windows ME, Windows NT o Windows XP.
- Microsoft .NET framework versión 1.1 o superior.
- 10 a 20 MB de disco duro libres.
- 20 a 64 MB de memoria RAM disponibles.

Si la computadora cumple con los requisitos listados, se puede proceder a realizar la instalación.

- Primero debemos elegir el lugar en el disco duro donde SAPH quedara instalado, como ejemplo tomaremos la ruta **"C:\Archivos de Programa\SAPH\"**
- Posteriormente inserte el CD de instalación de SAPH en su CD-ROM, luego busque la carpeta llamada **Modulo Administrativo.** En ella encontrara una serie de archivos, los cuales deberá copiar y luego pegar en el lugar elegido (en este caso: "C:\Archivos de Programa\SAPH\")..
- Pasaremos luego a configurar las diferentes opciones del módulo. Para ello, abriremos el archivo **Horarios.exe.config** en un editor de texto como el *notepad*. Una vez abierto podemos ver que el mismo es un archivo XML, en el cual las opciones estarán definidas por los Tags **<add key>.** Encontraremos en el archivo siguientes opciones:
    - cnnStr: esta opción se refiere a la cadena de conexión con el servidor de base de datos, Aquí deberemos escribir la cadena completa de conexión a un servidor de Base de Datos SQL Server 2000.
    - Servidor: esta opción se refiere al URL del servidor Web, el cual contendrá tanto el módulo Web como los reportes del sistema. En esta opción deberá escribir el URL completo incluyendo el http://
    - rutaInst: esta opción se refiere a la ruta en la cual ha quedado instalado el módulo Administrativo. Ej. "C:\Archivos de Programa\SAPH\"
- Opcionalmente, podremos dar click derecho al archivo ejecutable y elegir la opción enviar a… y luego seleccionar escritorio. Esto creara un acceso directo al programa en nuestro escritorio.

2. Módulo de Asignación de Horarios.

Para poder instalar el Módulo de Asignación de Horarios es necesario que la computadora en donde se desee instalar cumpla con los siguientes requerimientos:

- Una computadora con Windows 2000, Windows 98, Windows ME, Windows NT o Windows XP.
- Microsoft .NET framework versión 1.1 o superior.
- 5 a 10 MB de disco duro libres.
- 20 a 64 MB de memoria RAM disponibles.

Si la computadora cumple con los requisitos listados, se puede proceder a realizar la instalación.

- Primero debemos elegir el lugar en el disco duro donde SAPH quedara instalado, como ejemplo tomaremos la ruta **"C:\Archivos de Programa\SAPH\"**
- Posteriormente inserte el CD de instalación de SAPH en su CD-ROM, luego busque la carpeta llamada **Modulo Asignacion Horarios.** En ella encontrara una serie de archivos, los cuales deberá copiar y luego pegar en el lugar elegido (en este caso: "C:\Archivos de Programa\SAPH\").
- Pasaremos luego a configurar las diferentes opciones del módulo. Para ello, abriremos el archivo **Encargados.exe.config** en un editor de texto como el *notepad*. Una vez abierto podemos ver que el mismo es un archivo XML, en el cual las opciones estarán definidas por los Tags **<add key>.** Encontraremos en el archivo siguientes opciones:
    - cnnStr: esta opción se refiere a la cadena de conexión con el servidor de base de datos, Aquí deberemos escribir la cadena completa de conexión a un servidor de Base de Datos SQL Server 2000.
    - Servidor: esta opción se refiere al URL del servidor Web, el cual contendrá tanto el módulo Web como los reportes del sistema. En esta opción deberá escribir el URL completo incluyendo el http://
    - rutaInst: esta opción se refiere a la ruta en la cual ha quedado instalado el módulo Administrativo. Ej. "C:\Archivos de Programa\SAPH\"
- Opcionalmente, podremos dar click derecho al archivo ejecutable y elegir la opción enviar a… y luego seleccionar escritorio. Esto creara un acceso directo al programa en nuestro escritorio.

3. Módulo de Asignación de Aulas.

Para poder instalar el Módulo de Asignación de Aulas es necesario que la computadora en donde se desee instalar cumpla con los siguientes requerimientos:

- Una computadora con Windows 2000, Windows 98, Windows ME, Windows NT o Windows XP.
- Microsoft .NET framework versión 1.1 o superior.
- 2 a 5 MB de disco duro libres.
- 20 a 64 MB de memoria RAM disponibles.

Si la computadora cumple con los requisitos listados, se puede proceder a realizar la instalación.

- Primero debemos elegir el lugar en el disco duro donde SAPH quedara instalado, como ejemplo tomaremos la ruta **"C:\Archivos de Programa\SAPH\"**
- Posteriormente inserte el CD de instalación de SAPH en su CD-ROM, luego busque la carpeta llamada **Modulo Asignacion Aulas.** En ella encontrara una serie de archivos, los cuales deberá copiar y luego pegar en el lugar elegido (en este caso: "C:\Archivos de Programa\SAPH\").
- Pasaremos luego a configurar las diferentes opciones del módulo. Para ello, abriremos el archivo **Aulas.exe.config** en un editor de texto como el *notepad*. Una vez abierto podemos ver que el mismo es un archivo XML, en el cual las opciones estarán definidas por los Tags **<add key>.** Encontraremos en el archivo siguientes opciones:
    - cnnStr: esta opción se refiere a la cadena de conexión con el servidor de base de datos, Aquí deberemos escribir la cadena completa de conexión a un servidor de Base de Datos SQL Server 2000.
    - Servidor: esta opción se refiere al URL del servidor Web, el cual contendrá tanto el módulo Web como los reportes del sistema. En esta opción deberá escribir el URL completo incluyendo el http://
    - rutaInst: esta opción se refiere a la ruta en la cual ha quedado instalado el módulo Administrativo. Ej. "C:\Archivos de Programa\SAPH\"
- Opcionalmente, podremos dar click derecho al archivo ejecutable y elegir la opción enviar a… y luego seleccionar escritorio. Esto creara un acceso directo al programa en nuestro escritorio.

4. Modulo Web y Base de Datos.

Para instalar el Modulo Web, es necesario contar con una computadora la cual cuente con una IP pública, la cual tenga instalado el Internet Information Services.

Si la computadora cuenta con estos requerimientos, procederemos a copiar todos los archivos de la carpeta **Modulo Web**, que se encuentran en el CD de instalación de SAPH, y procederemos a copiarlos en una carpeta donde se tenga configurado acceso (ya sea una carpeta virtual o un Sitio Web del IIS).

Luego procedemos a editar el archivo **Web.config** con un editor de texto y en la parte llamada <appSettings> encontraremos los parámetros **ip** y **db**. Estos parámetros se refieren al nombre o dirección IP del servidor de base de datos y al nombre de la base de datos a la que se tendrá acceso.

Para la Instalación de la Base de Datos, solo es necesario copiar los archivos MDF y LDF que se encuentran en la carpeta **Base Datos** del CD de instalación de SAPH, copiarlos en el servidor de Base de Datos y levantarlos en el servidor SQL Server 2000.

NOTA: Una vez se hayan copiado estos archivos al servidor de base de datos, es necesario sacar respaldos periódicos de los mismos, ya que los archivos que se encuentran en el CD de instalación solo servirán para la primera vez que el sistema se instale.

Índice

Clase Aulas

La clase Aulas representa a la tabla de la base de datos aul_aulas, la cual contiene información referente a las aulas disponibles en la Universidad.

Properties, Events and Methods

Member	Description
cnn	Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.
Codigo	Código con el que se representa un registro en la base de datos.
Estado	Representa el estado de un registro. Si se encuentra en verdadero (***True***), significa que el registro esta activo. Si se encuentra en Falso (***False***) significa que el registro esta inactivo.
nuevo	Indica si el registro (representado por una instancia de la clase) es nuevo (***True***) o ya existe dentro de la base de datos (***False***). Este toma valor automáticamente, dependiendo del valor del parámetro ***codigo*** del metodo new.
cargarDatos	Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro ***codigo*** del metodo new **cargarDatos() as Void**
getEdificio	Regresa un objeto de la clase Edificios, el cual indica el edificio al cual pertenece el aula específica. **getEdificio() as Edificios**
GuardarDatos	Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos. **GuardarDatos() as Void**
New	Método que crea una nueva instancia de la clase Aulas, puede utilizarse de dos maneras: **New(conexion,codigo)**
ToString	Regresa una cadena de datos (***string***) que representa el nombre del aula específica. (ej. "A-23") **ToString() as String**
Capcidad	Esta propiedad indica la capacidad de alumnos para la cual el aula fue diseñada. Se relaciona con el campo de la clase ***_capacidad*** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información.
Edificio	Esta propiedad indica el edificio al cual el aula pertenece. Se relaciona con el campo de la clase ***_edificio*** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información. Ambos representan el código del edificio que se encuentra guardado en la base de datos.
Nombre	Esta propiedad indica el nombre del aula. Se relaciona con el campo de la clase ***_nombre*** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información. Ambos representan nombre con el que se almacena el registro en la base de datos.

Assembly: *comunes.dll*
Namespace: *Comunes*

Inherits from:
System.Object
  Comunes.Aulas

Aulas.cnn

Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.

Syntax: `private o.cnn as SqlConnection`

Aulas.Codigo

Código con el que se representa un registro en la base de datos.

Syntax:
```
public o.Codigo as Int32
```

Aulas.Estado

*Representa el estado de un registro. Si se encuentra en verdadero (**True**), significa que el registro esta activo. Si se encuentra en Falso (**False**) significa que el registro esta inactivo.*

Syntax:
```
public o.Estado as Boolean
```

Aulas.nuevo

*Indica si el registro (representado por una instancia de la clase) es nuevo (**True**) o ya existe dentro de la base de datos (**False**). Este toma valor automáticamente, dependiendo del valor del parámetro **codigo** del metodo new.*

Syntax:
```
private o.nuevo as Boolean
```

Aulas.cargarDatos

*Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro **codigo** del metodo new*

Syntax:
```
cargarDatos() as Void
```

Return: Void

Aulas.getEdificio

Regresa un objeto de la clase Edificios, el cual indica el edificio al cual pertenece el aula específica.

Syntax:
```
getEdificio() as Edificios
```

Return: Edificios

Example:
```
Dim dato as Comunes.Aulas
Dim edificio as Comunes.Edificios

dato = New Comunes.Aulas(padre.cnxMDI,3)
edificio = dato.getEdificio.
messagebox.show(edificio)
```

Aulas.GuardarDatos

Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos.

Syntax:
```
GuardarDatos() as Void
```

Return: Void

Example:
```
Dim datos As Comunes.Aulas

datos = New Comunes.Aulas(padre.cnxMDI)
datos.Nombre = "A-31"
datos.Edificio = 1 'Codigo del Edificio al que pertenece el Aula
datos.Capcidad = 35
datos.Estado = True
datos.GuardarDatos()
```

Aulas.New

Método que crea una nueva instancia de la clase Aulas, puede utilizarse de dos maneras:

1. Cuando se desea crear una instancia vacía para llenar sus atributos.
2. Cuando ya se sabe que registro se desea cargar de la base de datos. En este caso la instancia de la clase se conecta a la base de datos y extrae los datos necesarios.

Syntax:
```
New(conexion,codigo)
```

Parameters: **conexion As SqlClient.SqlConnection**
Optional codigo As Integer

Example:
```
'Para un registro que se sabe que ya existe en la base de datos
Dim dato as Comunes.Aulas
dato = New Comunes.Aulas(padre.cnxMDI, 1)

'Para un registro nuevo que se desea agregar a la base de datos
Dim dato as Comunes.Aulas
dato = New Comunes.Aulas(padre.cnxMDI)

'padre.cnxMDI es la conexion a la base de datos
```

Aulas.ToString

*Regresa una cadena de datos (**string**) que representa el nombre del aula específica. (ej. "A-23")*

Syntax:
```
ToString() as String
```

Return: String

Example:
```
Dim dato as Comunes.Aulas

dato = New Comunes.Aulas(padre.cnxMDI, 4)
messagebox.Show(dato.toString)
```

Aulas.Capcidad

*Esta propiedad indica la capacidad de alumnos para la cual el aula fue diseñada. Se relaciona con el campo de la clase **_capacidad** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información.*

Syntax:
```
public o.Capcidad as Int32
```

Example:
```
Dim dato as Comunes.Aulas

dato = New Comunes.Aulas(padre.cnxMdi,6)
dato.Capcidad = 35
dato.GuardarDatos()
messagebox.Show(cstr(dato.Capcidad))
```

Aulas.Edificio

*Esta propiedad indica el edificio al cual el aula pertenece. Se relaciona con el campo de la clase **_edificio** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información. Ambos representan el código del edificio que se encuentra guardado en la base de datos.*

Syntax:
```
public o.Edificio as Int32
```

Example:
```
Dim dato as Comunes.Aulas

dato = New Comunes.Aulas(padre.cnxMdi,6)
dato.Edificio = 1
```

Aulas.Nombre

*Esta propiedad indica el nombre del aula. Se relaciona con el campo de la clase **_nombre** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información. Ambos representan nombre con el que se almacena el registro en la base de datos.*

Syntax:
```
public o.Nombre as String
```

Example:
```
Dim dato as Comunes.Aulas

dato = New Comunes.Aulas(padre.cnxMdi,6)
dato.Edificio = "B-25"
```

Clase Carreras

La clase Carreras representa a la tabla de la base de datos car_carrera, la cual contiene información referente a las carreras que se imparten en la Universidad.

Properties, Events and Methods

Member	Description
cnn	Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.
Codigo	Código con el que se representa un registro en la base de datos.
Estado	Representa el estado de un registro. Si se encuentra en verdadero (***True***), significa que el registro esta activo. Si se encuentra en Falso (***False***) significa que el registro esta inactivo.
nuevo	Indica si el registro (representado por una instancia de la clase) es nuevo (***True***) o ya existe dentro de la base de datos (***False***). Este toma valor automáticamente, dependiendo del valor del parámetro ***codigo*** del metodo new.
cargarDatos	Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro ***codigo*** del metodo new **cargarDatos() as Void**
GuardarDatos	Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos. **GuardarDatos() as Void**
New	Método que crea una nueva instancia de la clase Carreras, puede utilizarse de dos maneras: **New(conexion,codigo)**
ToString	Regresa una cadena de datos (***string***) que representa el nombre de la carrera. (ej. "Ingeniería en Computación) **ToString() as String**
NivelMaximo	Esta propiedad indica el número de ciclos de una Carrera. Se relaciona con el campo de la clase ***_NivelMaximo*** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información. Ambos representan el numero de ciclos que la carrera tiene definidos en su Plan de Estudios.
Nombre	Esta propiedad indica el nombre de la Carrera. Se relaciona con el campo de la clase ***_nombre*** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información. Ambos representan nombre con el que se almacena el registro en la base de datos.

Assembly: *comunes.dll*
Namespace: *Comunes*

Inherits from:
System.Object
  Comunes.Carreras

Carreras.cnn

Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.

Syntax:
```
private o.cnn as SqlConnection
```

Carreras.Codigo

Código con el que se representa un registro en la base de datos.

Syntax:
```
public o.Codigo as Int32
```

Carreras.Estado

*Representa el estado de un registro. Si se encuentra en verdadero (**True**), significa que el registro esta activo. Si se encuentra en Falso (**False**) significa que el registro esta inactivo.*

Syntax:
```
public o.Estado as Boolean
```

Carreras.nuevo

*Indica si el registro (representado por una instancia de la clase) es nuevo (**True**) o ya existe dentro de la base de datos (**False**). Este toma valor automáticamente, dependiendo del valor del parámetro **codigo** del metodo new.*

Syntax:
```
private o.nuevo as Boolean
```

Carreras.cargarDatos

*Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro **codigo** del metodo new*

Syntax:
```
cargarDatos() as Void
```

Return: Void

Carreras.GuardarDatos

Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos.

Syntax:
```
GuardarDatos() as Void
```

Return: Void

Example:
```
Dim datos As Comunes.Carreras

datos = New Comunes.Carreras(padre.cnxMDI)
datos.Nombre = "Ingeniería Biomedica"
datos.NivelMaximo = 10 'Contados en Numeros de ciclos
datos.Estado = True
datos.GuardarDatos()
```

Carreras.New

Método que crea una nueva instancia de la clase Carreras, puede utilizarse de dos maneras:

1. Cuando se desea crear una instancia vacía para llenar sus atributos.
2. Cuando ya se sabe que registro se desea cargar de la base de datos. En este caso la instancia de la clase se conecta a la base de datos y extrae los datos necesarios.

Syntax:
```
New(conexion,codigo)
```

Parameters: **conexion As SqlClient.SqlConnection**
Optional codigo As Integer

Example:
```
'Para un registro que se sabe que ya existe en la base de datos
Dim dato as Comunes.Carreras
dato = New Comunes.Carreras(padre.cnxMDI, 1)

'Para un registro nuevo que se desea agregar a la base de datos
Dim dato as Comunes.Carreras
dato = New Comunes.Carreras(padre.cnxMDI)

'padre.cnxMDI es la conexion a la base de datos
```

Carreras.ToString

*Regresa una cadena de datos (**string**) que representa el nombre de la carrera. (ej. "Ingeniería en Computación)*

Syntax:
```
ToString() as String
```

Return: String

Example:
```
Dim dato as Comunes.Carreras

dato = New Comunes.Carreras(padre.cnxMDI, 3)
messagebox.Show(dato.toString)
```

Carreras.NivelMaximo

*Esta propiedad indica el número de ciclos de una Carrera. Se relaciona con el campo de la clase **_NivelMaximo** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información. Ambos representan el numero de ciclos que la carrera tiene definidos en su Plan de Estudios.*

Syntax:
```
public o.NivelMaximo as Int32
```

Example:
```
Dim dato as Comunes.Carreras
dato = New Comunes.Carreras(padre.cnxMdi,6)
dato.NivelMaximo = 10
```

Carreras.Nombre

*Esta propiedad indica el nombre de la Carrera. Se relaciona con el campo de la clase **_nombre** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información. Ambos representan nombre con el que se almacena el registro en la base de datos.*

Syntax:
```
public o.Nombre as String
```

Example:
```
Dim dato as Comunes.Carreras
dato = New Comunes.Carreras(padre.cnxMdi,6)
dato.Nombre = "Ingeniería Industrial"
```

Clase Ciclo

La clase Ciclo representa a la tabla de la base de datos hoc_horario_ciclo, la cual contiene información referente a los ciclos.

Properties, Events and Methods

Member	Description
Anio	Esta propiedad indica el año del ciclo. Se relaciona con el campo de la clase ***_anio*** (que es de uso exclusivo de la clase), el cual es de tipo numerico y contiene exactamente la misma información. Como se menciono anteriormente, indica el año del ciclo al cual se desea referir (ej Ciclo 01/**2005**)
cnn	Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.
Codigo	Código con el que se representa un registro en la base de datos.
Estado	Representa el estado de un registro. Si se encuentra en verdadero (***True***), significa que el registro esta activo. Si se encuentra en Falso (***False***) significa que el registro esta inactivo.
nuevo	Indica si el registro (representado por una instancia de la clase) es nuevo (***True***) o ya existe dentro de la base de datos (***False***). Este toma valor automáticamente, dependiendo del valor del parámetro ***codigo*** del metodo new
cargarDatos	Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro ***codigo*** del metodo new **cargarDatos() as Void**
Existe	Retorna verdadero (**True**) si el ciclo que se esta consultando existe, y regresa falso (**False**) si el ciclo que se esta consultando no existe. Esto es útil para el sistema cuando se intenta ingresar un ciclo nuevo, de esta manera si desea ingresar uno ya existente, la función nos indicaría si este ya existe, y de ser así, podríamos mostrar el mensaje de que el ciclo ya se encuentra en la base de datos. **Existe() as Boolean**
GuardarDatos	Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos. **GuardarDatos() as Void**
New	Método que crea una nueva instancia de la clase Ciclo, puede utilizarse de dos maneras: **New(conexion,codigo)**
ToString	Regresa una cadena de datos (***string***) que representa el nombre del ciclo y el estado del mismo (ej. "Ciclo 1/2005(Activo)"). **ToString() as String**
Ciclo	Esta propiedad indica el numero del ciclo (ya sea 1, 2 o 3). Se relaciona con el campo de la clase ***_ciclo*** (que es de uso exclusivo de la clase), el cual es de tipo numerico y contiene exactamente la misma información. Como se menciono anteriormente, indica el numero del ciclo al cual se desea referir (ej Ciclo **01**/2005)

Assembly: *comunes.dll*
Namespace: *Comunes*

Inherits from:
System.Object
  Comunes.Ciclo

Ciclo.Anio

*Esta propiedad indica el año del ciclo. Se relaciona con el campo de la clase **_anio** (que es de uso exclusivo de la clase), el cual es de tipo numerico y contiene exactamente la misma información. Como se menciono anteriormente, indica el año del ciclo al cual se desea referir (ej Ciclo 01/**2005**)*

Syntax:
```
public o.Anio as Int32
```

Example:
```
Dim dato as Comunes.Ciclo
dato = New Comunes.Ciclo(padre.cnxMdi,6)
dato.Anio = 2005
```

Ciclo.cnn

Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.

Syntax:
```
private o.cnn as SqlConnection
```

Ciclo.Codigo

Código con el que se representa un registro en la base de datos.

Syntax:
```
public o.Codigo as Int32
```

Ciclo.Estado

*Representa el estado de un registro. Si se encuentra en verdadero (**True**), significa que el registro esta activo. Si se encuentra en Falso (**False**) significa que el registro esta inactivo.*

Syntax:
```
public o.Estado as Boolean
```

Ciclo.nuevo

*Indica si el registro (representado por una instancia de la clase) es nuevo (**True**) o ya existe dentro de la base de datos (**False**). Este toma valor automáticamente, dependiendo del valor del parámetro **codigo** del metodo new*

Syntax:
```
private o.nuevo as Boolean
```

Ciclo.cargarDatos

*Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro **codigo** del metodo new*

Syntax:
```
cargarDatos() as Void
```

Return: Void

Parameters:

Ciclo.Existe

*Retorna verdadero (**True**) si el ciclo que se esta consultando existe, y regresa falso (**False**) si el ciclo que se esta consultando no existe. Esto es útil para el sistema cuando se intenta ingresar un ciclo nuevo, de esta manera si desea ingresar uno ya existente, la función nos indicaría si este ya existe, y de ser así, podríamos mostrar el mensaje de que el ciclo ya se encuentra en la base de datos.*

Syntax:
```
Existe() as Boolean
```

Return: Boolean

Example:
```
Dim dato As Comunes.Ciclo

dato.Anio = 2005
dato.Ciclo = 1

If dato.Existe = False Then
        dato.Estado = True
        dato.GuardarDatos()
Else
        MessageBox.Show("Este ciclo ya existe en la Base de Datos",
"Ciclos", MessageBoxButtons.OK, MessageBoxIcon.Exclamation)
End If
```

Ciclo.GuardarDatos

Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos.

Syntax:
```
GuardarDatos() as Void
```

Return: Void

Parameters:

Example:
```
Dim datos As Comunes.Ciclos

datos = New Comunes.Ciclos(padre.cnxMDI)
dato.Anio = 2005
dato.Ciclo = 1
datos.Estado = True
datos.GuardarDatos()
```

Ciclo.New

Método que crea una nueva instancia de la clase Ciclo, puede utilizarse de dos maneras:

1. Cuando se desea crear una instancia vacía para llenar sus atributos.
2. Cuando ya se sabe que registro se desea cargar de la base de datos. En este caso la instancia de la clase se conecta a la base de datos y extrae los datos necesarios.

Syntax:
```
New(conexion,codigo)
```

Parameters: **conexion As SqlClient.SqlConnection**
Optional codigo As Integer

Example:
```
'Para un registro que se sabe que ya existe en la base de datos
Dim dato as Comunes.Ciclo
dato = New Comunes.Ciclo(padre.cnxMDI, 1)

'Para un registro nuevo que se desea agregar a la base de datos
Dim dato as Comunes.Ciclo
dato = New Comunes.Ciclo(padre.cnxMDI)

'padre.cnxMDI es la conexion a la base de datos
```

Ciclo.ToString

*Regresa una cadena de datos (**string**) que representa el nombre del ciclo y el estado del mismo (ej. "Ciclo 1/2005(Activo)").*

Syntax:
```
ToString() as String
```

Return: String

Example:
```
Dim dato as Comunes.Ciclo

dato = New Comunes.Ciclo(padre.cnxMDI, 3)
messagebox.Show(dato.toString)
```

Ciclo.Ciclo

*Esta propiedad indica el numero del ciclo (ya sea 1, 2 o 3). Se relaciona con el campo de la clase **_ciclo** (que es de uso exclusivo de la clase), el cual es de tipo numerico y contiene exactamente la misma información. Como se menciono anteriormente, indica el numero del ciclo al cual se desea referir (ej Ciclo **01**/2005)*

Syntax:
```
public o.Ciclo as Int32
```

Example:
```
Dim dato as Comunes.Ciclo
dato = New Comunes.Ciclo(padre.cnxMdi,6)
```

```
dato.Ciclo = 2
```

Clase clProfesores

La clase clProfesores representa a la tabla de la base de datos pro_profesores, la cual contiene información referente a los docentes de la Universidad.

Properties, Events and Methods

Member	Description
cnn	Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.
Codigo	Código con el que se representa un registro en la base de datos.
Encargado	Un valor booleano que nos indica si el docente al que nos estamos refiriendo es encargado de realizar los horarios, lo cual significa que podrá tener acceso tanto al sistema de disponibilidad de tiempo como también al sistema creación de grupos y asignación de horarios.
Estado	Representa el estado de un registro. Si se encuentra en verdadero (***True***), significa que el registro esta activo. Si se encuentra en Falso (***False***) significa que el registro esta inactivo.
Instructor	Indica si la persona que estamos ingresando en el sistema es un instructor . Si se encuentra en verdadero (***True***), significa que el registro pertenece a un instructor. Si se encuentra en Falso (***False***) significa que el registro es un Docente.
nuevo	Indica si el registro (representado por una instancia de la clase) es nuevo (***True***) o ya existe dentro de la base de datos (***False***). Este toma valor automáticamente, dependiendo del valor del parámetro ***codigo*** del metodo new
cargarDatos	Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro ***codigo*** del metodo new **cargarDatos() as Void**
getCuartosHorasEnClase	Retorna una colección de objetos de enteros, que representan todas los códigos de cuartos de horas que el docente tiene ocupadas porque esta impartiendo clases. **getCuartosHorasEnClase(dia as Int32, ciclo as Int32, grupo as Grupos&) as Collection**
getCuartosHorasLibres	Al igual que el método getCuartosHorasEnClase, este método también regresa una colección que contiene todos los códigos de cuartos de horas que el docente tiene libre (así como lo estableció en su disponibilidad de tiempo). **getCuartosHorasLibres(dia as Int32, ciclo as Int32) as Collection**
getEscuela	Retorna un objeto de tipo escuela, que representa la escuela a la cual el docente pertenece dentro de la Universidad. **getEscuela() as Escuelas**
getGruposArr	Retorna un Arraylist, el cual contiene objetos del tipo Grupos, y representan los grupos en los cuales el docente imparte clases. **getGruposArr(ciclo as Int32) as ArrayList**
getHorasEnClase	Este método regresa una colección de objetos del tipo HorasClase, que indican todas las horas en las cuales el docente se encuentra impartiendo clases. **getHorasEnClase(dia as Int32, grupo as Grupos) as Collection**
getTodasHorasEnClase	Retorna una colección con todas las horas en las cuales el docente se encuentra impartiendo clases. Esto lo hace teniendo en cuenta el día y el ciclo en que se encuentra trabajando la aplicación. **getTodasHorasEnClase(dia as Int32, ciclo as Int32) as Collection**
GuardarDatos	Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos. **GuardarDatos() as Void**
HoraLibre	Este método regresa un valor numérico indicando si el docente no cuenta con tiempo disponible (0), si el docente tiene la hora disponible (1) o si se encuentra impartiendo una clase a esa hora (2). **HoraLibre(dia as Int32, cuarto_hora as Int32, ciclo as Int32) as Collection**

New	Método que crea una nueva instancia de la clase clProfesores, puede utilizarse de dos maneras: **New(conexion,codigo)**
soy_encargado	Este método regresa una cadena de texto que contiene todos los datos de contacto del docente. Esto es útil en el sistema cuando se sabe que un docente es encargado y se desean mostrar todos sus datos de contacto. **soy_encargado() as String**
ToString	Regresa una cadena de datos (***string***) que representa el nombre del Docente (ej. "Juan Cruz Rodríguez"). **ToString() as String**
Apellido	Esta propiedad indica el apellido del docente. Se relaciona con el campo de la clase ***_apellido*** (que es de uso exclusivo de la clase), el cual es de tipo carácter y contiene exactamente la misma información.
Contrasena	Esta propiedad indica la contraseña que se le a asignado al docente (esta es la contraseña que utilizara en el modulo Web, para ingresar su disponibilidad de tiempo). Se relaciona con el campo de la clase ***_contrasena*** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información. Ambos representan la contraseña que el docente utiliza.
Email	Esta propiedad indica el correo electrónico del docente. Se relaciona con el campo de la clase ***_email*** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información. Ambos representan el correo electrónico con el que se almacena el registro en la base de datos.
Escuela	Esta propiedad indica la escuela a la cual pertenece el docente. Se relaciona con el campo de la clase ***_escuela*** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información. Ambos representan codigo de la escuela con el que se almacena el registro en la base de datos.
Login	Esta propiedad indica el nombre de usuario que se le a asignado al docente (esta es el nombre de usuario que utilizara en el modulo Web, para ingresar su disponibilidad de tiempo). Se relaciona con el campo de la clase ***_login*** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información.
Nombre	Esta propiedad indica el nombre el docente. Se relaciona con el campo de la clase ***_nombre*** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información.
Telefono	Esta propiedad indica el telefono al cual se puede contactar al docente. Se relaciona con el campo de la clase ***_telefono*** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información. Ambos representan el número de telefono del docente que se almacena el registro en la base de datos.

Assembly: *comunes.dll*
Namespace: *Comunes*

Inherits from:
System.Object
  Comunes.clProfesores

clProfesores.cnn

Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.

Syntax: `private o.cnn as SqlConnection`

clProfesores.Codigo

Código con el que se representa un registro en la base de datos.

Syntax: `public o.Codigo as Int32`

clProfesores.Encargado

Un valor booleano que nos indica si el docente al que nos estamos refiriendo es encargado de realizar los horarios, lo cual significa que podrá tener acceso tanto al sistema de disponibilidad de tiempo como también al sistema creación de grupos y asignación de horarios.

Syntax:
```
public o.Encargado as Boolean
```

Example:
```
Dim datos As Comunes.clProfesores

datos = New Comunes.clProfesores(padre.cnxMDI)

'Seteamos algunas de sus propiedades
datos.Nombre = "Luis"
datos.Apellido = "Rodriguez"
datos.Estado = True
datos.Encargado = True
datos.GuardarDatos()
```

clProfesores.Estado

*Representa el estado de un registro. Si se encuentra en verdadero (**True**), significa que el registro esta activo. Si se encuentra en Falso (**False**) significa que el registro esta inactivo.*

Syntax:
```
public o.Estado as Boolean
```

clProfesores.Instructor

*Indica si la persona que estamos ingresando en el sistema es un instructor . Si se encuentra en verdadero (**True**), significa que el registro pertenece a un instructor. Si se encuentra en Falso (**False**) significa que el registro es un Docente.*

Se utiliza esta propiedad para que la clase sea capaz de diferenciar entre un Docente que imparte las clases teóricas y un instructor que imparte las clases de laboratorio.

Syntax:
```
public o.Instructor as Boolean
```

Example:
```
Dim datos As Comunes.clProfesores

datos = New Comunes.clProfesores(padre.cnxMDI)

'Seteamos algunas de sus propiedades
datos.Nombre = "Luis"
datos.Apellido = "Rodriguez"
datos.Estado = True
datos.Instructor = True
'La instancia estara representando a un instructor
datos.GuardarDatos()
```

clProfesores.nuevo

*Indica si el registro (representado por una instancia de la clase) es nuevo (**True**) o ya existe dentro de la base de datos (**False**). Este toma valor automáticamente, dependiendo del valor del parámetro **codigo** del metodo new*

Syntax:
```
private o.nuevo as Boolean
```

clProfesores.cargarDatos

*Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro **codigo** del metodo new*

Syntax:
```
cargarDatos() as Void
```

Return: Void

clProfesores.getCuartosHorasEnClase

Retorna una colección de objetos de enteros, que representan todas los códigos de cuartos de horas que el docente tiene ocupadas porque esta impartiendo clases.

Este método es util a la hora de la asignación de horarios, en el modulo de Asignación de Grupos y Horas Clase.

Esto se realiza consultando la base de datos por todas las horas en que el docente imparte clase en un día especifico, de un ciclo que se encuentra activo y de todas la materias que sean diferentes a la del grupo especifico en el que se esta trabajando.

Una vez obtenidos estos datos, se comienza a llenar una collection que contendra todos los codigos de cada cuarto de hora que se encuentre entre los rangos de hora de inicio y hora de fin de los registros encontrados en la base de datos (asi por ejemplo si un registro de la base de datos tiene como hora de inicio las 7 de mañana y como hora fin las 9 de la maña, y asumiendo que el cuarto de hora de 7:00 a 7:15 tiene el codigo 1, la colección se llenar con los siguientes valores [1,2,3,4,5,6,7,8]).

Syntax: `getCuartosHorasEnClase(dia as Int32, ciclo as Int32, grupo as Grupos&) as Collection`

Return: Collection

Parameters: **dia as Int32**
ciclo as Int32
grupo as Grupos&

clProfesores.getCuartosHorasLibres

Al igual que el método getCuartosHorasEnClase, este método también regresa una colección que contiene todos los códigos de cuartos de horas que el docente tiene libre (así como lo estableció en su disponibilidad de tiempo).

Este método es útil a la hora de la asignación de horarios, en el modulo de Asignación de Grupos y Horas Clase.

Esto se realiza consultando la base de datos por todas las horas que el docente tiene libres en un día específico, de un ciclo que se encuentra activo.

Syntax: `getCuartosHorasLibres(dia as Int32, ciclo as Int32) as Collection`

Return: Collection

Parameters: **dia as Int32**
ciclo as Int32

clProfesores.getEscuela

Retorna un objeto de tipo escuela, que representa la escuela a la cual el docente pertenece dentro de la Universidad.

Syntax: `getEscuela() as Escuelas`

Return: Escuelas

Example:

```
Dim miEscuela as Comunes.Escuelas
Dim Docente as Comunes.clProfesores

Docente = New Comunes.clProfesores(padre.cnxMDI,1)
miEscuela = Docente.getEscuela()

Messagebox.Show(miEscuela)

'padre.cnxMDI es la conexion a la base de datos
```

clProfesores.getGruposArr

Retorna un Arraylist, el cual contiene objetos del tipo Grupos, y representan los grupos en los cuales el docente imparte clases.

Syntax: `getGruposArr(ciclo as Int32) as ArrayList`

Return: ArrayList

Parameters: **ciclo as Int32**

Example:

```
Dim Docente as Comunes.clProfesores
Dim grupos as New ArrayList()

Docente = New Comunes.clProfesores(padre.cnxMDI,1)
grupos = Docente.getGruposArr

'padre.cnxMDI es la conexion a la base de datos
```

clProfesores.getHorasEnClase

Este método regresa una colección de objetos del tipo HorasClase, que indican todas las horas en las cuales el docente se encuentra impartiendo clases.

Esto lo hace en dos pasos, primero consulta a la Base de datos en busca de los registros filtrando por el Docente, día y Materia. Luego se realiza otra consulta, pero en esta ocasión los resultados se filtran por Docente, día y Grupo.

Syntax: `getHorasEnClase(dia as Int32, grupo as Grupos) as Collection`

Return: Collection

Parameters: **dia as Int32**
grupo as Grupos

Example:

```
Dim Docente as Comunes.clProfesores
Dim clases as New Collection()

Docente = New Comunes.clProfesores(padre.cnxMDI,1)
clases = Docente.getHorasEnClase

'padre.cnxMDI es la conexion a la base de datos
```

clProfesores.getTodasHorasEnClase

Retorna una colección con todas las horas en las cuales el docente se encuentra impartiendo clases. Esto lo hace teniendo en cuenta el día y el ciclo en que se encuentra trabajando la aplicación.

Syntax: `getTodasHorasEnClase(dia as Int32, ciclo as Int32) as Collection`

Return: Collection

Parameters: **dia as Int32**
ciclo as Int32

clProfesores.GuardarDatos

Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos.

Syntax: `GuardarDatos() as Void`

Example:

```
Dim datos As Comunes.clProfesores

datos = New Comunes.clProfesores(padre.cnxMDI)
datos.Nombre = "Coralia"
datos.Apellido = "Rodríguez"
datos.Escuela = 1
datos.Login = "crodri"
datos.Contrasena = "12345"
datos.Email = "crodri@dominio.com"
```

```
datos.Telefono = "251-5000"
datos.Instructor = False
datos.Estado = True
datos.GuardarDatos()
```

clProfesores.HoraLibre

Este método regresa un valor numérico indicando si el docente no cuenta con tiempo disponible (0), si el docente tiene la hora disponible (1) o si se encuentra impartiendo una clase a esa hora (2).

Esto lo realiza filtrando la base de datos con los datos que se le proporcionan a la entrada, los cuales son: el día, el código de cuarto hora que se desea verificar y el ciclo en el que se desea verificar. Después de una serie de pruebas con los datos obtenidos con las consultas, se determina que número a regresar.

Syntax: `HoraLibre(dia as Int32, cuarto_hora as Int32, ciclo as Int32) as Collection`

Return: Collection

Parameters: **dia as Int32**
cuarto_hora as Int32
ciclo as Int32

clProfesores.New

Método que crea una nueva instancia de la clase clProfesores, puede utilizarse de dos maneras:

1. Cuando se desea crear una instancia vacía para llenar sus atributos.
2. Cuando ya se sabe que registro se desea cargar de la base de datos. En este caso la instancia de la clase se conecta a la base de datos y extrae los datos necesarios.

Syntax: `New(conexion,codigo)`

Parameters: **conexion As SqlClient.SqlConnection**
Optional codigo As Integer

Example:

```
'Para un registro que se sabe que ya existe en la base de datos
Dim dato as Comunes.clProfesores
dato = New Comunes.clProfesores(padre.cnxMDI, 1)

'Para un registro nuevo que se desea agregar a la base de datos
Dim dato as Comunes.clProfesores
dato = New Comunes.clProfesores(padre.cnxMDI)

'padre.cnxMDI es la conexion a la base de datos
```

clProfesores.soy_encargado

Este método regresa una cadena de texto que contiene todos los datos de contacto del docente. Esto es útil en el sistema cuando se sabe que un docente es encargado y se desean mostrar todos sus datos de contacto.

A pesar de su nombre, este método puede ser utilizado en cualquier docente que se desee.

Syntax: `soy_encargado() as String`

Return: String

clProfesores.ToString

*Regresa una cadena de datos (**string**) que representa el nombre del Docente (ej. "Juan Cruz Rodríguez").*

Syntax: `ToString() as String`

Return: String

Example:
```
Dim dato as Comunes.clProfesores

dato = New Comunes.clProfesores(padre.cnxMDI, 3)
messagebox.Show(dato.toString)
```

clProfesores.Apellido

*Esta propiedad indica el apellido del docente. Se relaciona con el campo de la clase **_apellido** (que es de uso exclusivo de la clase), el cual es de tipo carácter y contiene exactamente la misma información.*

Syntax: `public o.Apellido as String`

Example:
```
Dim dato as Comunes.clProfesores
dato = New Comunes.clProfesores(padre.cnxMdi)
dato.Apellido = "Cruz"
```

clProfesores.Contrasena

*Esta propiedad indica la contraseña que se le a asignado al docente (esta es la contraseña que utilizara en el modulo Web, para ingresar su disponibilidad de tiempo). Se relaciona con el campo de la clase **_contrasena** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información. Ambos representan la contraseña que el docente utiliza.*

***NOTA:** Esta contraseña se guarda encriptada en la base de datos, por cuestiones de seguridad. Esto se hace por medio de las funciones GetEncryptedData y GetDecryptedData.*

Syntax: `public o.Contrasena as String`

Example:
```
Dim dato as Comunes.clProfesores

'Preparando los datos para se guardados en un registro nuevo
dato = New Comunes.clProfesores(padre.cnxMdi)
dato.Contrasena = Comunes.Funciones.GetEncryptedData("HDTV316")

'Leyendo los datos de un registro ya existente, txtContrasena es un
control de tipo TextBox que se encuenta en el formulario
dato = New Comunes.clProfesores(padre.cnxMdi, 25)
txtContrasena.Text =
Comunes.Funciones.GetDecryptedData(datos.Contrasena)
```

clProfesores.Email

*Esta propiedad indica el correo electrónico del docente. Se relaciona con el campo de la clase **_email** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información. Ambos representan el correo electrónico con el que se almacena el registro en la base de datos.*

Syntax: `public o.Email as String`

Example:
```
Dim dato as Comunes.clProfesores
dato = New Comunes.clProfesores(padre.cnxMdi)
dato.Email = "jcruz@udb.edu.sv"
```

clProfesores.Escuela

*Esta propiedad indica la escuela a la cual pertenece el docente. Se relaciona con el campo de la clase **_escuela** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información. Ambos representan codigo de la escuela con el que se almacena el registro en la base de datos.*

Syntax: `public o.Escuela as Int32`

Example:
```
Dim dato as Comunes.clProfesores
dato = New Comunes.clProfesores(padre.cnxMdi)
dato.Escuela = 1
```

clProfesores.Login

*Esta propiedad indica el nombre de usuario que se le a asignado al docente (esta es el nombre de usuario que utilizara en el modulo Web, para ingresar su disponibilidad de tiempo). Se relaciona con el campo de la clase **_login** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información.*

Syntax:
```
public o.Login as String
```

Example:
```
Dim dato as Comunes.clProfesores
dato = New Comunes.clProfesores(padre.cnxMdi)
dato.Login = "jcruz"
```

clProfesores.Nombre

*Esta propiedad indica el nombre el docente. Se relaciona con el campo de la clase **_nombre** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información.*

Syntax:
```
public o.Nombre as String
```

Example:
```
Dim dato as Comunes.clProfesores
dato = New Comunes.clProfesores(padre.cnxMdi)
dato.Nombre = "Juan"
```

clProfesores.Telefono

*Esta propiedad indica el telefono al cual se puede contactar al docente. Se relaciona con el campo de la clase **_telefono** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información. Ambos representan el número de telefono del docente que se almacena el registro en la base de datos.*

Syntax:
```
public o.Telefono as String
```

Example:
```
Dim dato as Comunes.clProfesores
dato = New Comunes.clProfesores(padre.cnxMdi)
dato.Telefono = "251-5000"
```

Clase DetallePensum

La clase DetallePensum representa a la tabla de la base de datos ppm_plan_pensum_materia, la cual contiene información referente a las materias que son impartidas en un pensum.

***NOTA:** Esta clase solo trabaja con códigos que se relacionan con otras clases, en las cuales se encuentra la especificación de cada registro relacionado.*

Properties, Events and Methods

Member	Description
ciclo	Indica el número del ciclo en que se esta trabajando, no se refiere al ciclo activo en el que programa esta trabajando (ej. Ciclo 01/2005), sino más bien al ciclo en el cual se esta deseando agregar una materia en un plan de estudios (ej. Ciclo 8 del Plan 98 de la carrera Ingeniería en Computación).
cnn	Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.
Codigo	Código con el que se representa un registro en la base de datos.
materia	El código de la materia que se desea ingresar en un plan de estudios específico. Así, el código 3 podría representar la materia Matemática 1, el código 5 la materia Lenguajes Algoritmicos, etc.
nuevo	Indica si el registro (representado por una instancia de la clase) es nuevo (***True***) o ya existe dentro de la base de datos (***False***). Este toma valor automáticamente, dependiendo del valor del parámetro ***codigo*** del metodo new
pensum	Un valor numérico representa el código del Plan de Estudios en el cual se estará impartiendo esta materia especifica, así por ejemplo el código 1 podría representar el Plan de Estudios 1998 de la Carrera Ingeniería en Computación, el 2 podría representar el Plan de Estudio 2004 de la Carrera Ingeniería Mecánica, etc.
cargarDatos	Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro ***codigo*** del metodo new **cargarDatos() as Void**
EliminarDatos	Elimina de la base de datos el registro que la instancia de la clase representa. **EliminarDatos() as Void**
getMateria	Regresa un objeto del tipo Materias, el cual contiene toda la información de la materia que el objeto DetallePensum esta representando en un Plan de Estudios específico. **getMateria() as Materias**
getMateriasxCiclo	Retorna una colección de objetos DetallePensum los cuales contienen todas las materias que se imparten en el Plan de Estudios y el ciclo especificados en la instancia de la clase. **getMateriasxCiclo() as Collection**
getNumCiclos	Retorna un dato de tipo entero que indica el número de ciclos que el plan de estudios tiene asignados. **getNumCiclos() as Int32**
GuardarDatos	Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos. **GuardarDatos() as Void**
New()	Método que crea una nueva instancia de la clase DetallePensum, puede utilizarse de dos maneras: **New(conexion,codigo)**
ToString	Regresa una cadena de datos (***string***) que representa el nombre de la Materia a la cual esta clase representa en la posición de un plan de estudios específico. **ToString() as String**

Assembly: *comunes.dll*
Namespace: *Comunes*

Inherits from:
System.Object
Comunes.DetallePensum

DetallePensum.ciclo

Indica el número del ciclo en que se esta trabajando, no se refiere al ciclo activo en el que programa esta trabajando (ej. Ciclo 01/2005), sino más bien al ciclo en el cual se esta deseando agregar una materia en un plan de estudios (ej. Ciclo 8 del Plan 98 de la carrera Ingeniería en Computación).

Syntax:
```
public o.ciclo as Int32
```

DetallePensum.cnn

Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.

Syntax:
```
private o.cnn as SqlConnection
```

DetallePensum.Codigo

Código con el que se representa un registro en la base de datos.

Syntax:
```
public o.Codigo as Int32
```

DetallePensum.materia

El código de la materia que se desea ingresar en un plan de estudios específico. Así, el código 3 podría representar la materia Matemática 1, el código 5 la materia Lenguajes Algoritmicos, etc.

Syntax:
```
public o.materia as Int32
```

DetallePensum.nuevo

*Indica si el registro (representado por una instancia de la clase) es nuevo (**True**) o ya existe dentro de la base de datos (**False**). Este toma valor automáticamente, dependiendo del valor del parámetro **codigo** del metodo new*

Syntax:
```
private o.nuevo as Boolean
```

DetallePensum.pensum

Un valor numérico representa el código del Plan de Estudios en el cual se estará impartiendo esta materia especifica, así por ejemplo el código 1 podría representar el Plan de Estudios 1998 de la Carrera Ingeniería en Computación, el 2 podría representar el Plan de Estudio 2004 de la Carrera Ingeniería Mecánica, etc.

Syntax:
```
public o.pensum as Int32
```

DetallePensum.cargarDatos

*Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro **codigo** del metodo new*

Syntax:
```
cargarDatos() as Void
```

Return: Void

DetallePensum.EliminarDatos

Elimina de la base de datos el registro que la instancia de la clase representa.

Syntax:
```
EliminarDatos() as Void
```

Return: Void

Example:
```
Dim datos As new Comunes.DetallePensum(padre.cnxMDI, 3)
datos.EliminarDatos()

'padre.cnxMDI es la conexion a la base de datos
```

DetallePensum.getMateria

Regresa un objeto del tipo Materias, el cual contiene toda la información de la materia que el objeto DetallePensum esta representando en un Plan de Estudios específico.

Syntax: `getMateria() as Materias`

Return: Materias

Example:

```
Dim Posicion as Comunes.DetallePensum
Dim miMateria as Comunes.Materias

Posicion = New Comunes.DetallePensum(padre.cnxMDI, 5)
miMateria = Posicion.getMateria()
messagebox.show(miMateria)

'padre.cnxMDI es la conexion a la base de datos
```

DetallePensum.getMateriasxCiclo

Retorna una colección de objetos DetallePensum los cuales contienen todas las materias que se imparten en el Plan de Estudios y el ciclo especificados en la instancia de la clase.

Este método puede ser utilizado por una instancia vacía (una instancia que no contenga información extraída de la base de datos).

Syntax: `getMateriasxCiclo() as Collection`

Return: Collection

Example:

```
Dim aux As New Comunes.DetallePensum(padre.cnxMdi)
Dim materias As Collection
Dim i As Integer

aux.ciclo = 1
aux.pensum = 13
materias = aux.getMateriasxCiclo()

'padre.cnxMDI es la conexion a la base de datos
```

DetallePensum.getNumCiclos

Retorna un dato de tipo entero que indica el número de ciclos que el plan de estudios tiene asignados.

Syntax: `getNumCiclos() as Int32`

Return: Int32

Example:

```
Dim numCiclos As New Comunes.DetallePensum(padre.cnxMdi)
Dim ciclos As Integer

numCiclos.pensum = 13
ciclos = numCiclos.getNumCiclos
messagebox.show("Este Plan de estudios tiene " & ciclos & " asignados")

'padre.cnxMDI es la conexion a la base de datos
```

DetallePensum.GuardarDatos

Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos.

Syntax: `GuardarDatos() as Void`

Return: Void

Example:
```
Dim dato As New Comunes.DetallePensum(padre.cnxMDI)
dato.Ciclo = 1
dato.Materia = 3
dato.Pensum = 1
datos.GuardarDatos()

'padre.cnxMDI es la conexion a la base de datos
```

DetallePensum.New

Método que crea una nueva instancia de la clase DetallePensum, puede utilizarse de dos maneras:

1. Cuando se desea crear una instancia vacía para llenar sus atributos.
2. Cuando ya se sabe que registro se desea cargar de la base de datos. En este caso la instancia de la clase se conecta a la base de datos y extrae los datos necesarios.

Syntax:
```
New(conexion,codigo)
```

Parameters: **conexion As SqlClient.SqlConnection**
Optional codigo As Integer

Example:
```
'Para un registro que se sabe que ya existe en la base de datos
Dim dato as Comunes.DetallePensum
dato = New Comunes.DetallePensum(padre.cnxMDI, 1)

'Para un registro nuevo que se desea agregar a la base de datos
Dim dato as Comunes.DetallePensum
dato = New Comunes.DetallePensum(padre.cnxMDI)

'padre.cnxMDI es la conexion a la base de datos
```

DetallePensum.ToString

*Regresa una cadena de datos (**string**) que representa el nombre de la Materia a la cual esta clase representa en la posición de un plan de estudios específico.*

Syntax:
```
ToString() as String
```

Return: String

Example:
```
Dim dato as Comunes.DetallePensum

dato = New Comunes.DetallePensum(padre.cnxMDI, 3)
messagebox.Show(dato.toString)
```

Clase Edificios

La clase Edificios representa a la tabla de la base de datos edi_edificios, la cual contiene información referente a los edificios disponibles en la Universidad.

Properties, Events and Methods

Member	Description
cnn	Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.
Codigo	Código con el que se representa un registro en la base de datos.
Estado	Representa el estado de un registro. Si se encuentra en verdadero (***True***), significa que el registro esta activo. Si se encuentra en Falso (***False***) significa que el registro esta inactivo.
nuevo	Indica si el registro (representado por una instancia de la clase) es nuevo (***True***) o ya existe dentro de la base de datos (***False***). Este toma valor automáticamente, dependiendo del valor del parámetro ***codigo*** del metodo new
cargarDatos	Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro ***codigo*** del metodo new **cargarDatos() as Void**
getAulas	Retorna una colección de objetos de tipo Aulas, que representan todas las aulas que pertenecen al edificio específico. **getAulas() as Collection**
GuardarDatos	Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos. **GuardarDatos() as Void**
New	Método que crea una nueva instancia de la clase Edificios, puede utilizarse de dos maneras: **New(conexion,codigo)**
ToString	Regresa una cadena de datos (***string***) que representa el nombre del edificio (ej. "Edificio A") **ToString() as String**
Nombre	Esta propiedad indica el nombre del Edificio. Se relaciona con el campo de la clase ***_nombre*** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información. Ambos representan nombre con el que se almacena el registro en la base de datos.

Assembly: *comunes.dll*
Namespace: *Comunes*

Inherits from:
System.Object
  Comunes.Edificios

Edificios.cnn

Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.

Syntax: `private o.cnn as SqlConnection`

Edificios.Codigo

Código con el que se representa un registro en la base de datos.

Syntax: `public o.Codigo as Int32`

Edificios.Estado

*Representa el estado de un registro. Si se encuentra en verdadero (**True**), significa que el registro esta activo. Si se encuentra en Falso (**False**) significa que el registro esta inactivo.*

Syntax:
```
public o.Estado as Boolean
```

Edificios.nuevo

*Indica si el registro (representado por una instancia de la clase) es nuevo (**True**) o ya existe dentro de la base de datos (**False**). Este toma valor automáticamente, dependiendo del valor del parámetro **codigo** del metodo new*

Syntax:
```
private o.nuevo as Boolean
```

Edificios.cargarDatos

*Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro **codigo** del metodo new*

Syntax:
```
cargarDatos() as Void
```

Return: Void

Edificios.getAulas

Retorna una colección de objetos de tipo Aulas, que representan todas las aulas que pertenecen al edificio específico.

Para llevar a cabo esto, realiza una consulta a la base de datos y extrae los códigos de las aulas, que están relacionadas con el edificio especificado. Una vez se obtienen todos los códigos, se procede a crear los objetos de tipo aulas y agregarlos a una colección.

Syntax:
```
getAulas() as Collection
```

Return: Collection

Example:
```
Dim dato as comunes.Edificios
Dim aulas as Collection

dato = New Comunes.Edificios(padre.cnxMDI, 3)
aulas = dato.getAulas()
```

Edificios.GuardarDatos

Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos.

Syntax:
```
GuardarDatos() as Void
```

Example:
```
Dim datos As Comunes.Edificios

datos = New Comunes.Edificios(padre.cnxMDI)
datos.Nombre = "Edificio A"
datos.Estado = True
datos.GuardarDatos()

'padre.cnxMDI es la conexion a la base de datos
```

Edificios.New

Método que crea una nueva instancia de la clase Edificios, puede utilizarse de dos maneras:

1. Cuando se desea crear una instancia vacía para llenar sus atributos.
2. Cuando ya se sabe que registro se desea cargar de la base de datos. En este caso la instancia de la clase se conecta a la base de datos y extrae los datos necesarios.

Syntax: `New(conexion,codigo)`

Parameters: **conexion As SqlClient.SqlConnection**
Optional codigo As Integer

Example:
```
'Para un registro que se sabe que ya existe en la base de datos
Dim dato as Comunes.Edificios
dato = New Comunes.Edificios(padre.cnxMDI, 1)

'Para un registro nuevo que se desea agregar a la base de datos
Dim dato as Comunes.Edificios
dato = New Comunes.Edificios(padre.cnxMDI)

'padre.cnxMDI es la conexion a la base de datos
```

Edificios.ToString

*Regresa una cadena de datos (**string**) que representa el nombre del edificio (ej. "Edificio A")*

Syntax: `ToString() as String`

Return: String

Example:
```
Dim dato as Comunes.Edificios

dato = New Comunes.Edificios(padre.cnxMDI, 3)
messagebox.Show(dato.toString)
```

Edificios.Nombre

*Esta propiedad indica el nombre del Edificio. Se relaciona con el campo de la clase **_nombre** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información. Ambos representan nombre con el que se almacena el registro en la base de datos.*

Syntax: `public o.Nombre as String`

Example:
```
Dim dato as Comunes.Edificios
dato = New Comunes.Edificios(padre.cnxMdi)
dato.Nombre = "Edificio A"
```

Clase Escuelas

La clase Escuelas representa a la tabla de la base de datos esc_escuelas, la cual contiene información referente a las diferentes escuelas de la Universidad.

Properties, Events and Methods

Member	Description
cnn	Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.
Codigo	Código con el que se representa un registro en la base de datos.
Estado	Representa el estado de un registro. Si se encuentra en verdadero (***True***), significa que el registro esta activo. Si se encuentra en Falso (***False***) significa que el registro esta inactivo.
nuevo	Indica si el registro (representado por una instancia de la clase) es nuevo (***True***) o ya existe dentro de la base de datos (***False***). Este toma valor automáticamente, dependiendo del valor del parámetro ***codigo*** del metodo new
cargarDatos	Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro ***codigo*** del metodo new **cargarDatos() as Void**
edificiosFavoritos	Este método regresa una colección de objetos del tipo Edificios, los cuales representan los edificios en los cuales las materias de estas escuelas pueden ser impartidas. **edificiosFavoritos() as Collection**
GuardarDatos	Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos. **GuardarDatos() as Void**
Masterias()	Retorna un ArrayList en el cual se contienen objetos del tipo Materias, estos indican todas las materias que la escuela especifica imparte. **o.Escuelas.Masterias()**
New()	Método que crea una nueva instancia de la clase Escuelas, puede utilizarse de dos maneras: **o.Escuelas.New(conexion,codigo)**
ToString	Regresa una cadena de datos (***string***) que representa el nombre de la escuela (ej. "Escuela de Ingeniería Industrial") **ToString() as String**
Nombre	Esta propiedad indica el nombre el docente. Se relaciona con el campo de la clase ***_nombre*** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información. Ambos representan el nombre con el que se almacena el registro en la base de datos.

Assembly: *comunes.dll*
Namespace: *Comunes*

Inherits from:
System.Object
  Comunes.Escuelas

Escuelas.cnn

Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.

Syntax:
```
private o.cnn as SqlConnection
```

Escuelas.Codigo

Código con el que se representa un registro en la base de datos.

Syntax:
```
public o.Codigo as Int32
```

Escuelas.Estado

*Representa el estado de un registro. Si se encuentra en verdadero (**True**), significa que el registro esta activo. Si se encuentra en Falso (**False**) significa que el registro esta inactivo.*

Syntax:
```
public o.Estado as Boolean
```

Escuelas.nuevo

*Indica si el registro (representado por una instancia de la clase) es nuevo (**True**) o ya existe dentro de la base de datos (**False**). Este toma valor automáticamente, dependiendo del valor del parámetro **codigo** del metodo new*

Syntax:
```
private o.nuevo as Boolean
```

Escuelas.cargarDatos

*Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro **codigo** del metodo new*

Syntax:
```
cargarDatos() as Void
```

Return: Void

Escuelas.edificiosFavoritos

Este método regresa una colección de objetos del tipo Edificios, los cuales representan los edificios en los cuales las materias de estas escuelas pueden ser impartidas.

Syntax:
```
edificiosFavoritos() as Collection
```

Return: Collection

Example:
```
Dim datos As Comunes.Escuelas
Dim edificios as New Collection

datos = New Comunes.Escuelas(padre.cnxMDI,5)
edificios = datos.edificiosFavoritos()

'padre.cnxMDI es la conexion a la base de datos
```

Escuelas.GuardarDatos

Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos.

Syntax:
```
GuardarDatos() as Void
```

Return: Void

Parameters:

Example:
```
Dim datos As Comunes.Escuelas

datos = New Comunes.Escuelas(padre.cnxMDI)
datos.Nombre = "Escuela de Biomedica"
datos.GuardarDatos()

'padre.cnxMDI es la conexion a la base de datos
```

Escuelas.Masterias

Retorna un ArrayList en el cual se contienen objetos del tipo Materias, estos indican todas las materias que la escuela especifica imparte.

Syntax: `o.Escuelas.Masterias()`

Example:
```
Dim datos As Comunes.Escuelas
Dim materias as New ArrayList

datos = New Comunes.Escuelas(padre.cnxMDI,5)
materias = datos.Masterias()

'padre.cnxMDI es la conexion a la base de datos
```

Escuelas.New

Método que crea una nueva instancia de la clase Escuelas, puede utilizarse de dos maneras:

1. Cuando se desea crear una instancia vacía para llenar sus atributos.
2. Cuando ya se sabe que registro se desea cargar de la base de datos. En este caso la instancia de la clase se conecta a la base de datos y extrae los datos necesarios.

Syntax: `o.Escuelas.New(conexion,codigo)`

Parameters: **conexion As SqlClient.SqlConnection**
Optional codigo As Integer

Example:
```
'Para un registro que se sabe que ya existe en la base de datos
Dim dato as Comunes.Escuelas
dato = New Comunes.Escuelas(padre.cnxMDI, 1)

'Para un registro nuevo que se desea agregar a la base de datos
Dim dato as Comunes.Escuelas
dato = New Comunes.Escuelas(padre.cnxMDI)

'padre.cnxMDI es la conexion a la base de datos
```

Escuelas.ToString

*Regresa una cadena de datos (**string**) que representa el nombre de la escuela (ej. "Escuela de Ingeniería Industrial")*

Syntax: `ToString() as String`

Return: String

Example:
```
Dim dato as Comunes.Escuelas

dato = New Comunes.Escuelas(padre.cnxMDI, 3)
messagebox.Show(dato.toString)
```

Escuelas.Nombre

*Esta propiedad indica el nombre el docente. Se relaciona con el campo de la clase **_nombre** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información. Ambos representan el nombre con el que se almacena el registro en la base de datos.*

Syntax: `public o.Nombre as String`

Example:
```
Dim dato as Comunes.Escuelas
dato = New Comunes.Escuelas(padre.cnxMdi)
dato.Nombre = "Escuela de Computación"
```

Clase Grupos

La clase Grupos representa a la tabla de la base de datos mgr_materia_grupo, la cual contiene información referente a los grupos de clases que son impartidos en la Universidad.

***NOTA:** Esta clase solo trabaja con códigos que se relacionan con otras clases, en las cuales se encuentra la especificación de cada registro relacionado.*

Properties, Events and Methods

Member	Description
Capacidad	Representa la capacidad de alumnos que posee el grupo.
Ciclo	Representa el ciclo en el cual se ha creado el grupo específico (ej. "Ciclo 1/2005"). El campo guarda el código con el que se representa el ciclo en la base de datos.
cnn	Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.
Codigo	Código con el que se representa un registro en la base de datos.
Estado	Representa el estado de un registro. Si se encuentra en verdadero (***True***), significa que el registro esta activo. Si se encuentra en Falso (***False***) significa que el registro esta inactivo.
Grupo	Representa al número del grupo. (ej. Grupo 01, Grupo 02, etc.)
Materia	Representa la materia de la cual este grupo ha sido abierto. Este campo de la clase guarda únicamente el código con el que la materia es representada en la base de datos.
nuevo	Indica si el registro (representado por una instancia de la clase) es nuevo (***True***) o ya existe dentro de la base de datos (***False***). Este toma valor automáticamente, dependiendo del valor del parámetro ***codigo*** del metodo new
Profesor	Representa al docente que esta impartiendo este grupo de clases. Este campo de la clase guarda únicamente el código con el que el docente es representado en la base de datos.
cargarDatos	Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro ***codigo*** del metodo new **cargarDatos() as Void**
elimnarDato	Elimina de la base de datos el registro que la instancia de la clase representa. **elimnarDato() as Void**
getCiclo	Regresa un objeto del tipo Ciclo, el cual contiene la información relacionada con el ciclo en el cual se esta impartiendo este grupo. **getCiclo() as Ciclo**
getCuartosHoras	Regresa una colección de enteros, los cuales representan todos los códigos de cuartos de horas en los cuales este grupo es impartido. **getCuartosHoras(Optional dia as Int32) as Collection**
getHorarios	Regresa una colección de objetos HorasClase, los cuales representan todos los horarios que este grupo posee. **getHorarios() as Collection**
getHorariosXDia	Regresa una colección de objetos del tipo HorasClase, los cuales representan todos los horarios en los cuales este grupo es impartido, dependiendo del día que se le haya especificado. **getHorariosXDia(dia as Int32) as Collection**
getHorasRestantes	Retorna un entero el cual indica cuantas horas (de las que la materia se supone tiene programadas para impartir a la semana) quedan pendientes de asignar para un grupo. **getHorasRestantes() as Int32**

getMateria	Regresa un objeto del tipo Materia, el cual contiene la información relacionada a la materia de la cual este grupo es dependiente. **getMateria() as Materias**
getNumeroGrupo	Regresa un entero, que indica cual será el número del siguiente grupo que se puede crear para una materia. **getNumeroGrupo() as Int32**
getProfesor	Regresa un objeto del tipo clProfesores, el cual contiene la información relacionada al docente que esta impartiendo este grupo. **getProfesor() as clProfesores**
GuardarDatos	Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos. **GuardarDatos() as Void**
New	Método que crea una nueva instancia de la clase Grupos, puede utilizarse de dos maneras: **New(conexion,codigo)**
ToString	Regresa una cadena de datos (***string***) que representa el nombre grupo al cual la instancia de la clase representa (ej. "Grupo 01"). **ToString() as String**

Assembly: *comunes.dll*
Namespace: *Comunes*

Inherits from:
System.Object
  Comunes.Grupos

Grupos.Capacidad

Representa la capacidad de alumnos que posee el grupo.

Syntax:
```
public o.Capacidad as Int32
```

Grupos.Ciclo

Representa el ciclo en el cual se ha creado el grupo específico (ej. "Ciclo 1/2005"). El campo guarda el código con el que se representa el ciclo en la base de datos.

Syntax:
```
public o.Ciclo as Int32
```

Grupos.cnn

Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.

Syntax:
```
private o.cnn as SqlConnection
```

Grupos.Codigo

Código con el que se representa un registro en la base de datos.

Syntax:
```
public o.Codigo as Int32
```

Grupos.Estado

*Representa el estado de un registro. Si se encuentra en verdadero (**True**), significa que el registro esta activo. Si se encuentra en Falso (**False**) significa que el registro esta inactivo.*

Syntax:
```
public o.Estado as Boolean
```

Grupos.Grupo

Representa al número del grupo. (ej. Grupo 01, Grupo 02, etc.)

Syntax:
```
public o.Grupo as Int32
```

Grupos.Materia

Representa la materia de la cual este grupo ha sido abierto. Este campo de la clase guarda únicamente el código con el que la materia es representada en la base de datos.

Syntax:
```
public o.Materia as Int32
```

Grupos.nuevo

*Indica si el registro (representado por una instancia de la clase) es nuevo (**True**) o ya existe dentro de la base de datos (**False**). Este toma valor automáticamente, dependiendo del valor del parámetro **codigo** del metodo new*

Syntax:
```
private o.nuevo as Boolean
```

Grupos.Profesor

Representa al docente que esta impartiendo este grupo de clases. Este campo de la clase guarda únicamente el código con el que el docente es representado en la base de datos.

Syntax:
```
public o.Profesor as Int32
```

Grupos.cargarDatos

*Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro **codigo** del metodo new*

Syntax:
```
cargarDatos() as Void
```

Return: Void

Grupos.elimnarDato

Elimina de la base de datos el registro que la instancia de la clase representa.

Syntax:
```
elimnarDato() as Void
```

Return: Void

Example:
```
Dim datos As new Comunes.Grupos(padre.cnxMDI, 3)
datos.EliminarDato()

'padre.cnxMDI es la conexion a la base de datos
```

Grupos.getCiclo

Regresa un objeto del tipo Ciclo, el cual contiene la información relacionada con el ciclo en el cual se esta impartiendo este grupo.

Syntax:
```
getCiclo() as Ciclo
```

Return: Ciclo

Example:
```
Dim dato as Comunes.Grupos
Dim ciclo as Comunes.Ciclo

dato = New Comunes.Grupos(padre.cnxMDI, 3)
ciclo = dato.getCiclo
messagebox.show("El ciclo para el grupo " & dato.toString & " es " &
ciclo.toString)

'padre.cnxMDI es la conexion a la base de datos
```

Grupos.getCuartosHoras

Regresa una colección de enteros, los cuales representan todos los códigos de cuartos de horas en los cuales este grupo es impartido.

Si se especifica el parámetro opcional día, regresara todos los cuartos de hora de ese día específico. Si no se le especifica el día, regresará los cuartos de horas de todos los horarios en que se imparte el grupo.

Syntax: `getCuartosHoras(Optional dia as Int32) as Collection`

Return: Collection

Parameters: **Optional dia as Int32**

Example:
```
Dim dato as Comunes.Grupos
Dim CuartosHoraDia as New Collection

'Para un día especifico en este caso Lunes
dato = New Comunes.Grupos(padre.cnxMDI, 3)
CuartosHoraDia = dato.getCuartosHoras(0) '0 representa al día lunes, 1
al día Martes y asi sucesivamente

'Para todos los horarios de este grupo
dato = New Comunes.Grupos(padre.cnxMDI, 3)
CuartosHoraDia = dato.getCuartosHoras()

'padre.cnxMDI es la conexion a la base de datos
```

Grupos.getHorarios

Regresa una colección de objetos HorasClase, los cuales representan todos los horarios que este grupo posee.

Syntax: `getHorarios() as Collection`

Return: Collection

Example:
```
Dim dato as Comunes.Grupos
Dim horarios as New Collection

dato = New Comunes.Grupos(padre.cnxMDI, 3)
horarios = dato.getHorarios

'padre.cnxMDI es la conexion a la base de datos
```

Grupos.getHorariosXDia

Regresa una colección de objetos del tipo HorasClase, los cuales representan todos los horarios en los cuales este grupo es impartido, dependiendo del día que se le haya especificado.

Syntax: `getHorariosXDia(dia as Int32) as Collection`

Return: Collection

Parameters: **dia as Int32**

Example:
```
Dim dato as Comunes.Grupos
Dim horariosDia as New Collection

dato = New Comunes.Grupos(padre.cnxMDI, 3)
horariosDia = dato.getHorariosXDia(0) '0 representa al día lunes, 1 al
día Martes y asi sucesivamente

'padre.cnxMDI es la conexion a la base de datos
```

Grupos.getHorasRestantes

Retorna un entero el cual indica cuantas horas (de las que la materia se supone tiene programadas para impartir a la semana) quedan pendientes de asignar para un grupo.

Syntax: `getHorasRestantes() as Int32`

Return: Int32

Example:
```
Dim dato as Comunes.Grupos
Dim faltan as Integer

dato = New Comunes.Grupos(padre.cnxMDI,26)
faltan = dato.getHorasRestantes()
messagebox.show("Faltan " & faltan & " horas por asignar para este
grupo")

'padre.cnxMDI es la conexion a la base de datos
```

Grupos.getMateria

Regresa un objeto del tipo Materia, el cual contiene la información relacionada a la materia de la cual este grupo es dependiente.

Syntax: `getMateria() as Materias`

Return: Materias

Example:
```
Dim dato as Comunes.Grupos
Dim mat as Comunes.Materias

dato = New Comunes.Grupos(padre.cnxMDI, 3)
mat = dato.getMateria
messagebox.show(mat)

'padre.cnxMDI es la conexion a la base de datos
```

Grupos.getNumeroGrupo

Regresa un entero, que indica cual será el número del siguiente grupo que se puede crear para una materia.

Syntax: `getNumeroGrupo() as Int32`

Return: Int32

Example:
```
Dim dato as Comunes.Grupos

'Creamos un nuevo grupo
dato = New Comunes.Grupos(padre.cnxMDI)
dato.Grupo = dato.getNumeroGrupo 'Le asignamos su numero correlativo
correspondiente

'padre.cnxMDI es la conexion a la base de datos
```

Grupos.getProfesor

Regresa un objeto del tipo clProfesores, el cual contiene la información relacionada al docente que esta impartiendo este grupo.

Syntax: `getProfesor() as clProfesores`

Return: clProfesores

Example:
```
Dim dato as Comunes.Grupos
Dim docente as Comunes.clProfesores

dato = New Comunes.Grupos(padre.cnxMDI, 3)
mat = dato.getProfesor
messagebox.show(docente)
```

```
'padre.cnxMDI es la conexion a la base de datos
```

Grupos.GuardarDatos

Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos.

Syntax: `GuardarDatos() as Void`

Return: Void

Example:

```
Dim datos As Comunes.Grupos
datos = New Comunes.Grupos(padre.cnxMDI)
datos.Ciclo = 3
datos.Grupo = 2
datos.Materia = 5
datos.Profesor = 9
datos.Estado = True
datos.GuardarDatos()

'padre.cnxMDI es la conexion a la base de datos
```

Grupos.New

Método que crea una nueva instancia de la clase Grupos, puede utilizarse de dos maneras:

1. Cuando se desea crear una instancia vacía para llenar sus atributos.
2. Cuando ya se sabe que registro se desea cargar de la base de datos. En este caso la instancia de la clase se conecta a la base de datos y extrae los datos necesarios.

Syntax: `New(conexion,codigo)`

Parameters: **conexion As SqlClient.SqlConnection**
Optional codigo As Integer

Example:

```
'Para un registro que se sabe que ya existe en la base de datos
Dim dato as Comunes.Grupos
dato = New Comunes.Grupos(padre.cnxMDI, 1)

'Para un registro nuevo que se desea agregar a la base de datos
Dim dato as Comunes.Grupos
dato = New Comunes.Grupos(padre.cnxMDI)

'padre.cnxMDI es la conexion a la base de datos
```

Grupos.ToString

*Regresa una cadena de datos (**string**) que representa el nombre grupo al cual la instancia de la clase representa (ej. "Grupo 01").*

Syntax: `ToString() as String`

Return: String

Example:

```
Dim dato as Comunes.Grupos
dato = New Comunes.Grupos(padre.cnxMDI, 3)
messagebox.Show(dato.toString)
```

Clase HorasClase

La clase HorasClase representa a la tabla de la base de datos hcl_horas_clase, la cual contiene información referente a los horarios de clase de los diferentes grupos de clases que son impartidos en la Universidad.

Properties, Events and Methods

Member	Description
Aula	Nombre o número del salón de clases en el cual tienen a eso atrapado.
cnn	Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.
Codigo	Código con el que se representa un registro en la base de datos.
fecha_asignacionAula	Fecha en la cual se le asigna un aula a este horario. Este es un dato que sirve estrictamente de auditoria. Es un dato de tipo DateTime.
fecha_creacion	Fecha en la cual se creo el registro de este horario. Este es un dato que sirve estrictamente de auditoria. Es un dato de tipo DateTime.
fecha_modificacionAula	Fecha en la cual se modifico el registro de este horario. Este es un dato que sirve estrictamente de auditoria. Es un dato de tipo DateTime.
Grupo	Contiene el código de un objeto Grupos, al cual esta instancia HorasClase pertenece.
hfin	Indica la hora en la cual este horario se da por finalizado. Este campo de la clase, al igual que se contraparte en la base de datos, guarda el código del cuarto de hora que representa a la hora de finalización.
hinicio	Indica la hora en la cual este horario se da por iniciado. Este campo de la clase, al igual que se contraparte en la base de datos, guarda el código del cuarto de hora que representa a la hora de finalización.
nuevo	Indica si el registro (representado por una instancia de la clase) es nuevo (***True***) o ya existe dentro de la base de datos (***False***). Este toma valor automáticamente, dependiendo del valor del parámetro ***codigo*** del metodo new
usuario_asignaAula	Usuario que realizo la asignación de aula a este horario. Este es un dato que sirve estrictamente de auditoria.
usuario_crea	Usuario que creo el horario. Este es un dato que sirve estrictamente de auditoria.
usuario_modificaAula	Usuario que modifica el registro de horarios. Este es un dato que sirve estrictamente de auditoria.
cargarDatos	Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro ***codigo*** del metodo new **cargarDatos() as Void**
elimnarDato	Elimina de la base de datos el registro que la instancia de la clase representa. **elimnarDato() as Void**
getAula	Regresa un objeto del tipo Aulas, el cual representa el aula en la cual es impartido este horario. **getAula() as Aulas**
getChoqueGruposMismoHorario	Regresa un entero el cual indica el número de grupos que están siendo impartidos en horarios similares al actual. **getChoqueGruposMismoHorario(ciclo as Int32, materia as Int32) as Int32**
getCodigosAulasEnMiHora	Regresa una colección la cual contiene los códigos de todas las aulas de un edificio específico, las cuales están siendo utilizadas por otro horario que coincide con el actual. **getCodigosAulasEnMiHora(edificio as Int32, capacidad as Int32) as Object**
getGrupo	Regresa un objeto del tipo Grupos, el cual representa el grupo al cual este horario pertenece.

	getGrupo() as Grupos
GuardarDatos	Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos. **GuardarDatos() as Void**
New	Método que crea una nueva instancia de la clase HorasClase, puede utilizarse de dos maneras: **New(conexion, codigo)**
ToString	Regresa una cadena de datos (***string***) que representa el horario en el cual el grupo es impartido (ej. "Lunes de 5:00 pm a 8:00 pm") **ToString() as String**
dia	Esta propiedad indica el día en que este horario es impartido. Se relaciona con el campo de la clase ***_dia*** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información. Ambos representan el código del día en el cual es impartido el horario (ej. 0 para Lunes, 1 para Martes, etc).

Assembly: *comunes.dll*
Namespace: *Comunes*

Inherits from:
System.Object
  Comunes.HorasClase

HorasClase.Aula
Nombre o número del salón de clases en el cual tienen a eso atrapado.

Syntax:
```
public o.Aula as Int32
```

HorasClase.cnn
Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.

Syntax:
```
private o.cnn as SqlConnection
```

HorasClase.Codigo
Código con el que se representa un registro en la base de datos.

Syntax:
```
public o.Codigo as Int32
```

HorasClase.fecha_asignacionAula
Fecha en la cual se le asigna un aula a este horario. Este es un dato que sirve estrictamente de auditoria. Es un dato de tipo DateTime.

Syntax:
```
public o.fecha_asignacionAula as DateTime
```

HorasClase.fecha_creacion
Fecha en la cual se creo el registro de este horario. Este es un dato que sirve estrictamente de auditoria. Es un dato de tipo DateTime.

Syntax:
```
public o.fecha_creacion as DateTime
```

HorasClase.fecha_modificacionAula
Fecha en la cual se modifico el registro de este horario. Este es un dato que sirve estrictamente de auditoria. Es un dato de tipo DateTime.

Syntax:
```
public o.fecha_modificacionAula as DateTime
```

HorasClase.Grupo
Contiene el código de un objeto Grupos, al cual esta instancia HorasClase pertenece.

Syntax:
```
public o.Grupo as Int32
```

HorasClase.hfin

Indica la hora en la cual este horario se da por finalizado. Este campo de la clase, al igual que se contraparte en la base de datos, guarda el código del cuarto de hora que representa a la hora de finalización.

Syntax:
```
public o.hfin as Int32
```

HorasClase.hinicio

Indica la hora en la cual este horario se da por iniciado. Este campo de la clase, al igual que se contraparte en la base de datos, guarda el código del cuarto de hora que representa a la hora de finalización.

Syntax:
```
public o.hinicio as Int32
```

HorasClase.nuevo

*Indica si el registro (representado por una instancia de la clase) es nuevo (**True**) o ya existe dentro de la base de datos (**False**). Este toma valor automáticamente, dependiendo del valor del parámetro **codigo** del metodo new*

Syntax:
```
private o.nuevo as Boolean
```

HorasClase.usuario_asignaAula

Usuario que realizo la asignación de aula a este horario. Este es un dato que sirve estrictamente de auditoria.

Syntax:
```
public o.usuario_asignaAula as Int32
```

HorasClase.usuario_crea

Usuario que creo el horario. Este es un dato que sirve estrictamente de auditoria.

Syntax:
```
public o.usuario_crea as Int32
```

HorasClase.usuario_modificaAula

Usuario que modifica el registro de horarios. Este es un dato que sirve estrictamente de auditoria.

Syntax:
```
public o.usuario_modificaAula as Int32
```

HorasClase.cargarDatos

*Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro **codigo** del metodo new*

Syntax:
```
cargarDatos() as Void
```

Return: Void

HorasClase.elimnarDato

Elimina de la base de datos el registro que la instancia de la clase representa.

Syntax:
```
elimnarDato() as Void
```

Return: Void

Example:
```
Dim datos As new Comunes.HorasClase(padre.cnxMDI, 3)
datos.EliminarDatos()

'padre.cnxMDI es la conexion a la base de datos
```

HorasClase.getAula

Regresa un objeto del tipo Aulas, el cual representa el aula en la cual es impartido este horario.

Syntax:
```
getAula() as Aulas
```

Return: Aulas

Example:
```
Dim dato as comunes.HorasClase
Dim aula as comunes.Aulas

dato = New comunes.HorasClase(padre.cnxMDI, 3)
aula = dato.getAula

'padre.cnxMDI es la conexion a la base de datos
```

HorasClase.getChoqueGruposMismoHorario

Regresa un entero el cual indica el número de grupos que están siendo impartidos en horarios similares al actual.

Syntax: `getChoqueGruposMismoHorario(ciclo as Int32, materia as Int32) as Int32`

Return: Int32

Parameters:
ciclo as Int32
materia as Int32

Example:
```
Dim hora_clase as Comunes.HorasClase
Dim choques as Integer

hora_clase = New Comunes.HorasClase(padre.cnxMDI, 45)
Choques = hora_clase.getChoqueGruposMismoHorario(2,5)

'padre.cnxMDI es la conexion a la base de datos
```

HorasClase.getCodigosAulasEnMiHora

Regresa una colección la cual contiene los códigos de todas las aulas de un edificio específico, las cuales están siendo utilizadas por otro horario que coincide con el actual.

Syntax: `getCodigosAulasEnMiHora(edificio as Int32, capacidad as Int32) as Object`

Return: Collection

Parameters:
edificio as Int32
capacidad as Int32

Example:
```
Dim hora_clase as Comunes.HorasClase
Dim aulas_excluidas a New Collection()

hora_clase = New Comunes.HorasClase(padre.cnxMDI, 45)
aulas_excluidas = hora_clase.getCodigosAulasEnMiHora(1,35)

'padre.cnxMDI es la conexion a la base de datos
```

HorasClase.getGrupo

Regresa un objeto del tipo Grupos, el cual representa el grupo al cual este horario pertenece.

Syntax: `getGrupo() as Grupos`

Return: Grupos

Example:
```
Dim dato as comunes.HorasClase
Dim miGrupo as comunes.Grupos

dato = New comunes.HorasClase(padre.cnxMDI, 3)
aula = dato.getGrupo

'padre.cnxMDI es la conexion a la base de datos
```

HorasClase.GuardarDatos

Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos.

Syntax: `GuardarDatos() as Void`

Return: Void

HorasClase.New

Método que crea una nueva instancia de la clase HorasClase, puede utilizarse de dos maneras:

1. Cuando se desea crear una instancia vacía para llenar sus atributos.
2. Cuando ya se sabe que registro se desea cargar de la base de datos. En este caso la instancia de la clase se conecta a la base de datos y extrae los datos necesarios.

Syntax: `New(conexion, codigo)`

Parameters: **conexion As SqlClient.SqlConnection**
Optional codigo As Integer

Example:
```
'Para un registro que se sabe que ya existe en la base de datos
Dim dato as Comunes.HorasClase
dato = New Comunes.HorasClase(padre.cnxMDI, 1)

'Para un registro nuevo que se desea agregar a la base de datos
Dim dato as Comunes.HorasClase
dato = New Comunes.HorasClase(padre.cnxMDI)

'padre.cnxMDI es la conexion a la base de datos
```

HorasClase.ToString

*Regresa una cadena de datos (**string**) que representa el horario en el cual el grupo es impartido (ej. "Lunes de 5:00 pm a 8:00 pm")*

Syntax: `ToString() as String`

Return: String

Example:
```
Dim dato as Comunes.HorasClase
dato = New Comunes.HorasClase(padre.cnxMDI, 3)
messagebox.Show(dato.toString)

'padre.cnxMDI es la conexion a la base de datos
```

HorasClase.dia

*Esta propiedad indica el día en que este horario es impartido. Se relaciona con el campo de la clase **_dia** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información. Ambos representan el código del día en el cual es impartido el horario (ej. 0 para Lunes, 1 para Martes, etc).*

Syntax: `public o.dia as Int32`

Example:
```
Dim dato as Comunes.HorasClase
dato = New Comunes.HorasClase(padre.cnxMdi)
dato.dia = 0 'Para el caso del día lunes
```

Clase MateriaEdificio

La clase MateriaEdificio representa a la tabla de la base de datos med_materias_edificios, la cual contiene información a los edificios en los cuales cada materia puede ser impartida..

***NOTA:** Esta clase solo trabaja con códigos que se relacionan con otras clases, en las cuales se encuentra la especificación de cada registro relacionado.*

Properties, Events and Methods

Member	Description
cnn	Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.
Codigo	Código con el que se representa un registro en la base de datos.
edificio	Representa el edificio que ha sido asignado a una materia en particular, para que un los gurupos de esta materia sean impartidos en el edificio.
materia	Representa la materia a la cual se le ha asignado el edificio en el cual puede ser impartida.
nuevo	Indica si el registro (representado por una instancia de la clase) es nuevo (***True***) o ya existe dentro de la base de datos (***False***). Este toma valor automáticamente, dependiendo del valor del parámetro ***codigo*** del metodo new.
cargarDatos	Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro ***codigo*** del metodo new. **cargarDatos() as Void**
EliminarDatos	Elimina de la base de datos el registro que la instancia de la clase representa. **EliminarDatos() as Void**
getEdificio	Regresa un objeto del tipo Edificios, el cual representa al edificio que se ha seleccionado para que el grupo pueda ser impartido en él. **getEdificio() as Edificios**
getMateria	Regresa un objeto de tipo Materias, que indica la materia que se ha seleccionado para que sus grupos sean impartidos en un edificio específico. **getMateria() as Materias**
GuardarDatos	Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos. **GuardarDatos() as Void**
New()	Método que crea una nueva instancia de la clase MateriaEdificio, puede utilizarse de dos maneras: **New(conexion, codigo)**
ToString	Regresa una cadena de datos (***string***) que representa el nombre del edificio que se le ha asignado a la materia (ej. "Edificio A") **ToString() as String**

***Assembly:** comunes.dll*
***Namespace:** Comunes*

Inherits from:
System.Object
  Comunes.MateriaEdificio

MateriaEdificio.cnn

Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.

Syntax:
```
private o.cnn as SqlConnection
```

MateriaEdificio.Codigo

Código con el que se representa un registro en la base de datos.

Syntax:
```
public o.Codigo as Int32
```

MateriaEdificio.edificio

Representa el edificio que ha sido asignado a una materia en particular, para que un los gurupos de esta materia sean impartidos en el edificio.

Syntax:
```
public o.edificio as Int32
```

MateriaEdificio.materia

Representa la materia a la cual se le ha asignado el edificio en el cual puede ser impartida.

Syntax:
```
public o.materia as Int32
```

MateriaEdificio.nuevo

*Indica si el registro (representado por una instancia de la clase) es nuevo (**True**) o ya existe dentro de la base de datos (**False**). Este toma valor automáticamente, dependiendo del valor del parámetro **codigo** del metodo new.*

Syntax:
```
private o.nuevo as Boolean
```

MateriaEdificio.cargarDatos

*Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro **codigo** del metodo new.*

Syntax:
```
cargarDatos() as Void
```

Return: Void

MateriaEdificio.EliminarDatos

Elimina de la base de datos el registro que la instancia de la clase representa.

Syntax:
```
EliminarDatos() as Void
```

Return: Void

Example:
```
Dim datos As new Comunes.MateriaEdificio(padre.cnxMDI, 3)
datos.EliminarDatos()

'padre.cnxMDI es la conexion a la base de datos
```

MateriaEdificio.getEdificio

Regresa un objeto del tipo Edificios, el cual representa al edificio que se ha seleccionado para que el grupo pueda ser impartido en él.

Syntax:
```
getEdificio() as Edificios
```

Return: Edificios

Example:
```
Dim dato as Comunes.MateriaEdificio
Dim edif as Comunes.Edificios

dato = New Comunes.Edificios(padre.cnxMDI, 3)
edif = dato.getEdificio
messagebox.show(edif)
```

```
'padre.cnxMDI es la conexion a la base de datos
```

MateriaEdificio.getMateria

Regresa un objeto de tipo Materias, que indica la materia que se ha seleccionado para que sus grupos sean impartidos en un edificio específico.

Syntax: `getMateria() as Materias`

Return: Materias

Example:

```
Dim dato as Comunes.MateriaEdificio
Dim mat as Comunes.Materias

dato = New Comunes.MateriaEdificio(padre.cnxMDI, 3)
mat = dato.getMateria
messagebox.show(mat)

'padre.cnxMDI es la conexion a la base de datos
```

MateriaEdificio.GuardarDatos

Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos.

Syntax: `GuardarDatos() as Void`

Return: Void

Example:

```
Dim datos As Comunes.MateriaEdificio
datos = New Comunes.Grupos(padre.cnxMDI)
datos.Edificio = 3
datos.Materia = 2
datos.GuardarDatos()

'padre.cnxMDI es la conexion a la base de datos
```

MateriaEdificio.New

Método que crea una nueva instancia de la clase MateriaEdificio, puede utilizarse de dos maneras:

1. Cuando se desea crear una instancia vacía para llenar sus atributos.
2. Cuando ya se sabe que registro se desea cargar de la base de datos. En este caso la instancia de la clase se conecta a la base de datos y extrae los datos necesarios.

Syntax: `New(conexion, codigo)`

Parameters: **conexion As SqlClient.SqlConnection**
Optional codigo As Integer

Example:

```
'Para un registro que se sabe que ya existe en la base de datos
Dim dato as Comunes.MateriaEdificio
dato = New Comunes.MateriaEdificio(padre.cnxMDI, 1)

'Para un registro nuevo que se desea agregar a la base de datos
Dim dato as Comunes.MateriaEdificio
dato = New Comunes.MateriaEdificio(padre.cnxMDI)

'padre.cnxMDI es la conexion a la base de datos
```

MateriaEdificio.ToString

*Regresa una cadena de datos (**string**) que representa el nombre del edificio que se le ha asignado a la materia (ej. "Edificio A")*

Syntax:
```
ToString() as String
```

Return: String

Example:
```
Dim dato as Comunes.MateriaEdificio
dato = New Comunes.MateriaEdificio(padre.cnxMDI, 3)
messagebox.Show(dato.toString)
```

Class Funciones

Esta es una clase especial la cual contiene funciones que son de utilidad para todos los sistemas desarrollados.

Properties, Events and Methods

Member	Description
cargar_ciclo_activo	Funcion que extrae la base de datos el ciclo en el cual se encuentran trabajando los sitemas y regresa un objeto Ciclos, el cual representa al ciclo activo **cargar_ciclo_activo(cnn as SqlConnection) as Ciclo**
confirmar_pass	Retorna verdadero si los dos strings de entrada son identicos, esto es util a la hora de pedir las confirmaciones de las contraseñas. **confirmar_pass(contra as String, confirm as String) as Boolean**
deshabilitar_todo	Deshabilita los menus y la barra de herramientas de todos los sistemas cuando estos se encuentras desconectados. **deshabilitar_todo(barra as ToolBar&, el_menu as Menu&, el_smenu as MenuItem&) as Object**
filas_diferente_color	Rellena el fondo de un control del tipo ListView, para que cada una de sus filas contengan colores diferentes. Esto con el objeto de hacer mas facil su lectura. **filas_diferente_color(objListview as ListView, color1 as Color, color2 as Color) as Void**
GetDecryptedData	Función que convierte una cadena de datos, la cual ha sido encriptada, en una cadena normal sin ningún tipo de encriptamiento. **GetDecryptedData(Data as String) as String**
getDia	Funcion que regresa en un string el nombre del día que corresponde al codigo que se le ha ingresado. **getDia(codigo as Int32) as String**
GetEncryptedData	Función que convierte una cadena de caracteres simples a una cadena de caracteres codificada. Esto con el objeto de proteger las contraseñas del sistema. **GetEncryptedData(Data as String) as String**
GetHora	Funcion que regresa en un string la hora correspondiente al codigo que se le ha ingresado. **GetHora(codigo as Int32) as String**
getMateriasEnMismoCiclo	**getMateriasEnMismoCiclo(entrada as Collection, mat as Materias, conexion as String) as Collection**
habilitar_todo	Habilita los menus y la barra de herramientas de todos los sistemas cuando estos se encuentras conectados. **habilitar_todo(barra as ToolBar&, el_menu as Menu) as Object**
limpiar_controles	Limpia la informacion de todos los controles que se encuentren en el formulario, siempre y cuando estos sean del tipo TextBox **limpiar_controles(formulario as Form&) as Object**
RegresarControles	**RegresarControles(c as Control) as ArrayList**
validar_usuarios	**validar_usuarios(user as String, password as String, tipo as String, tabla as DataTable) as Int32**

Assembly: *comunes.dll*
Namespace: *Comunes*

Inherits from:
System.Object
  Comunes.Funciones

Funciones.cargar_ciclo_activo

Funcion que extrae la base de datos el ciclo en el cual se encuentran trabajando los sitemas y regresa un objeto Ciclos, el cual representa al ciclo activo

Syntax: `cargar_ciclo_activo(cnn as SqlConnection) as Ciclo`

Return: Ciclo

Parameters: **cnn as SqlConnection**

Funciones.confirmar_pass

Retorna verdadero si los dos strings de entrada son identicos, esto es util a la hora de pedir las confirmaciones de las contraseñas.

Syntax: `confirmar_pass(contra as String, confirm as String) as Boolean`

Return: Boolean

Parameters: **contra as String**
confirm as String

Funciones.deshabilitar_todo

Deshabilita los menus y la barra de herramientas de todos los sistemas cuando estos se encuentras desconectados.

Syntax: `deshabilitar_todo(barra as ToolBar&, el_menu as Menu&, el_smenu as MenuItem&) as Object`

Return: Object

Parameters: **barra as ToolBar**
el_menu as Menu
el_smenu as MenuItem

Funciones.filas_diferente_color

Rellena el fondo de un control del tipo ListView, para que cada una de sus filas contengan colores diferentes. Esto con el objeto de hacer mas facil su lectura.

Syntax: `filas_diferente_color(objListview as ListView, color1 as Color, color2 as Color) as Void`

Return: Void

Parameters: **objListview as ListView**
color1 as Color
color2 as Color

Funciones.GetDecryptedData

Función que convierte una cadena de datos, la cual ha sido encriptada, en una cadena normal sin ningún tipo de encriptamiento.

Syntax: `GetDecryptedData(Data as String) as String`

Return: String

Parameters: **Data as String**

Funciones.getDia

Funcion que regresa en un string el nombre del día que corresponde al codigo que se le ha ingresado.

El sistema maneja los dias como códigos de acuerdo al siguiente esquema.
0 = Lunes
1 = Martes
2 = Miercoles
3 = Jueves
4 = Viernes
5 = Sabado
6 = Domingo

Syntax: `getDia(codigo as Int32) as String`

Return: String

Parameters: **codigo as Int32**

Funciones.GetEncryptedData

Función que convierte una cadena de caracteres simples a una cadena de caracteres codificada. Esto con el objeto de proteger las contraseñas del sistema.

Syntax: `GetEncryptedData(Data as String) as String`

Return: String

Parameters: **Data as String**

Funciones.GetHora

Funcion que regresa en un string la hora correspondiente al codigo que se le ha ingresado.

Es importante notar que la mayoria del manejo de horas dentro del sistema se realiza mediante cuartos de horas, cada cuarto de hora tiene un codigo específico, asi:

1 = "7:00 am"
2 = "7:15 am"
3 = "7:30 am"
4 = "7:45 am"
5 = "8:00 am"
6 = "8:15 am"
7 = "8:30 am"
8 = "8:45 am"
9 = "9:00 am"
10 = "9:15 am"
11 = "9:30 am"
12 = "9:45 am"
13 = "10:00 am"
14 = "10:15 am"
15 = "10:30 am"
16 = "10:45 am"
17 = "11:00 am"
18 = "11:15 am"
19 = "11:30 am"
20 = "11:45 am"
21 = "12:00 pm"
22 = "12:15 pm"
23 = "12:30 pm"
24 = "12:45 pm"
25 = "1:00 pm"
26 = "1:15 pm"
27 = "1:30 pm"
28 = "1:45 pm"
29 = "2:00 pm"
30 = "2:15 pm"
31 = "2:30 pm"
32 = "2:45 pm"
33 = "3:00 pm"
35 = "3:30 pm"
36 = "3:45 pm"
37 = "4:00 pm"
38 = "4:15 pm"
39 = "4:30 pm"
40 = "4:45 pm"
41 = "5:00 pm"
42 = "5:15 pm"
43 = "5:30 pm"
44 = "5:45 pm"
45 = "6:00 pm"
46 = "6:15 pm"
47 = "6:30 pm"
48 = "6:45 pm"
49 = "7:00 pm"
50 = "7:15 pm"
51 = "7:30 pm"
52 = "7:45 pm"
53 = "8:00 pm"
54 = "8:15 pm"
55 = "8:30 pm"
56 = "8:45 pm"
57 = "9:00 pm"

Syntax: `GetHora(codigo as Int32) as String`

Return: String

Parameters: **codigo as Int32**

Funciones.getMateriasEnMismoCiclo

Syntax: `getMateriasEnMismoCiclo(entrada as Collection, mat as Materias, conexion as String) as Collection`

Return: Collection

Parameters: **entrada as Collection**

mat as Materias

conexion as String

Funciones.habilitar_todo

Habilita los menus y la barra de herramientas de todos los sistemas cuando estos se encuentras conectados.

Syntax: `habilitar_todo(barra as ToolBar&, el_menu as Menu) as Object`

Return: Object

Parameters: **barra as ToolBar&**
el_menu as Menu

Funciones.limpiar_controles

Limpia la informacion de todos los controles que se encuentren en el formulario, siempre y cuando estos sean del tipo TextBox

Syntax: `limpiar_controles(formulario as Form&) as Object`

Return: Object

Parameters: **formulario as Form&**

Funciones.RegresarControles

Syntax: `RegresarControles(c as Control) as ArrayList`

Return: ArrayList

Parameters: **c as Control**

Funciones.validar_usuarios

Syntax: `validar_usuarios(user as String, password as String, tipo as String, tabla as DataTable) as Int32`

Return: Int32

Parameters: **user as String**

password as String

tipo as String

tabla as DataTable

Class Materias

La clase Materias representa a la tabla de la base de datos mat_materias, la cual contiene información referente a las materias que son impartidas en la Universidad.

Properties, Events and Methods

Member	Description
cnn	Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.
Codigo	Código con el que se representa un registro en la base de datos.
Estado	Representa el estado de un registro. Si se encuentra en verdadero (***True***), significa que el registro esta activo. Si se encuentra en Falso (***False***) significa que el registro esta inactivo.
nuevo	Indica si el registro (representado por una instancia de la clase) es nuevo (***True***) o ya existe dentro de la base de datos (***False***). Este toma valor automáticamente, dependiendo del valor del parámetro ***codigo*** del metodo new.
cargarDatos	Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro ***codigo*** del metodo new. **cargarDatos() as Void**
getAulasDeMateria	Este metodo re **getAulasDeMateria() as Int32**
getEdificios	Regresa una coleccion de objetos de tipo Edificios, los cuales representan los edificios en los cuales puede ser impartida esta materia. **getEdificios() as Collection**
getEdificiosMaterias	Regresa una coleccion de objetos de tipo MateriaEdificio, los cuales representan los edificios en los cuales puede ser impartida esta materia. **getEdificiosMaterias() as Collection**
getEscuela	Regresa un objeto de tipo Escuela, el cual contiene toda la información referente a la escuela de la cual es dependiente la materia. **getEscuela() as Escuelas**
getGrupos	Retorna una colección de objetos del tipo Grupos, la cual contiene toda la información de los grupos que se han creado para esta materia. **getGrupos(ciclo as Int32) as Collection**
getGruposArr	Al igual que la función getGrupos, esta funcion regresa los grupos que estan asociados a esta materia, con la diferencia de que el tipo de dato que retorna esta funcion es un ArrayList. **getGruposArr(ciclo as Int32) as ArrayList**
getPosicionPensum	Regresa una colección la cual contiene objetos del tipo DetallePensum, estos representan a todas las ocaciones en que esta materia es encontrada en un Plan de Estudios. **getPosicionPensum() as Collection**
GuardarDatos	Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos. **GuardarDatos() as Void**
New(conexion,codigo)	Método que crea una nueva instancia de la clase Materias, puede utilizarse de dos maneras: **o.Materias.New(conexion,codigo)**
ToString	Regresa una cadena de datos (***string***) que representa el nombre de la materia al cual la instancia de la clase representa (ej. "Matemática 1"). **ToString() as String**
CodigoUDB	Esta propiedad indica el codigo con el que se conoce una materia dentro de la universidad. Se relaciona con el campo de la clase ***_CodigoUdb*** (que es de uso exclusivo de la clase), el cual es de tipo caracter y

	contiene exactamente la misma información.
Escuela	Esta propiedad indica la escuela de la cual la materia es dependiente. Se relaciona con el campo de la clase ***_escuela*** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información.
HorasSemana	Esta propiedad indica el número de horas a la semana que una materia puede ser impartida. Se relaciona con el campo de la clase ***_horas*** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información.
Laboratorio	Esta propiedad indica si la materia que se esta ingresando es un laboratorio. Es un valor booleano que nos indica si la clase es laboratorio (**True**) o si no lo es (**False**).
Nombre	Esta propiedad indica el nombre con el que se conoce una materia dentro de la universidad. Se relaciona con el campo de la clase ***_CodigoUdb*** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información.

Assembly: *comunes.dll*
Namespace: *Comunes*

Inherits from:
System.Object
  Comunes.Materias

Materias.cnn

Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.

Syntax:
```
private o.cnn as SqlConnection
```

Materias.Codigo

Código con el que se representa un registro en la base de datos.

Syntax:
```
public o.Codigo as Int32
```

Materias.Estado

*Representa el estado de un registro. Si se encuentra en verdadero (**True**), significa que el registro esta activo. Si se encuentra en Falso (**False**) significa que el registro esta inactivo.*

Syntax:
```
public o.Estado as Boolean
```

Materias.nuevo

*Indica si el registro (representado por una instancia de la clase) es nuevo (**True**) o ya existe dentro de la base de datos (**False**). Este toma valor automáticamente, dependiendo del valor del parámetro **codigo** del metodo new.*

Syntax:
```
private o.nuevo as Boolean
```

Materias.cargarDatos

*Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro **codigo** del metodo new.*

Syntax:
```
cargarDatos() as Void
```

Return: Void

Parameters:

Materias.getAulasDeMateria

Este metodo re

Syntax:
```
getAulasDeMateria() as Int32
```

Return: Int32

Parameters: Regresa un dato de tipo entero, el cual indica si el número de Aulas en las cuales se etan impartiendo la materia.

Materias.getEdificios

Regresa una coleccion de objetos de tipo Edificios, los cuales representan los edificios en los cuales puede ser impartida esta materia.

Syntax: `getEdificios() as Collection`

Return: Collection

Materias.getEdificiosMaterias

Regresa una coleccion de objetos de tipo MateriaEdificio, los cuales representan los edificios en los cuales puede ser impartida esta materia.

Syntax: `getEdificiosMaterias() as Collection`

Return: Collection

Parameters:

Materias.getEscuela

Regresa un objeto de tipo Escuela, el cual contiene toda la información referente a la escuela de la cual es dependiente la materia.

Syntax: `getEscuela() as Escuelas`

Return: Escuelas

Parameters:

Materias.getGrupos

Retorna una colección de objetos del tipo Grupos, la cual contiene toda la información de los grupos que se han creado para esta materia.

Syntax: `getGrupos(ciclo as Int32) as Collection`

Return: Collection

Parameters: **ciclo as Int32**

Materias.getGruposArr

Al igual que la función getGrupos, esta funcion regresa los grupos que estan asociados a esta materia, con la diferencia de que el tipo de dato que retorna esta funcion es un ArrayList.

Syntax: `getGruposArr(ciclo as Int32) as ArrayList`

Return: ArrayList

Parameters: **ciclo as Int32**

Materias.getPosicionPensum

Regresa una colección la cual contiene objetos del tipo DetallePensum, estos representan a todas las ocaciones en que esta materia es encontrada en un Plan de Estudios.

Syntax: `getPosicionPensum() as Collection`

Return: Collection

Materias.GuardarDatos

Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos.

Syntax: `GuardarDatos() as Void`

Return: Void

Materias.New

Método que crea una nueva instancia de la clase Materias, puede utilizarse de dos maneras:

1. Cuando se desea crear una instancia vacía para llenar sus atributos.
2. Cuando ya se sabe que registro se desea cargar de la base de datos. En este caso la instancia de la clase se conecta a la base de datos y extrae los datos necesarios.

Syntax:
```
o.Materias.New(conexion,codigo)
```

Example:
```
'Para un registro que se sabe que ya existe en la base de datos
Dim dato as Comunes.Materias
dato = New Comunes.Materias(padre.cnxMDI, 1)

'Para un registro nuevo que se desea agregar a la base de datos
Dim dato as Comunes.Materias
dato = New Comunes.Materias(padre.cnxMDI)

'padre.cnxMDI es la conexion a la base de datos
```

Materias.ToString

*Regresa una cadena de datos (**string**) que representa el nombre de la materia al cual la instancia de la clase representa (ej. "Matemática 1").*

Syntax:
```
ToString() as String
```

Return: String

Parameters:

Example:
```
Dim dato as Comunes.Materias
dato = New Comunes.Materias(padre.cnxMDI, 3)
messagebox.Show(dato.toString)
```

Materias.CodigoUDB

*Esta propiedad indica el codigo con el que se conoce una materia dentro de la universidad. Se relaciona con el campo de la clase **_CodigoUdb** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información.*

Syntax:
```
public o.CodigoUDB as String
```

Example:
```
Dim dato as Comunes.Materias
dato = New Comunes.Materias(padre.cnxMdi)
dato.CodigoUDB = "MAT101"
```

Materias.Escuela

*Esta propiedad indica la escuela de la cual la materia es dependiente. Se relaciona con el campo de la clase **_escuela** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información.*

Syntax:
```
public o.Escuela as Int32
```

Example:
```
Dim dato as Comunes.Materias
dato = New Comunes.Materias(padre.cnxMdi)
dato.Escuela = 4
```

Materias.HorasSemana

*Esta propiedad indica el número de horas a la semana que una materia puede ser impartida. Se relaciona con el campo de la clase **_horas** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información.*

Syntax:
```
public o.HorasSemana as Int32
```

Example:
```
Dim dato as Comunes.Materias
```

```
dato = New Comunes.Materias(padre.cnxMdi)
dato.HorasSemana = 5
```

Materias.Laboratorio

*Esta propiedad indica si la materia que se esta ingresando es un laboratorio. Es un valor booleano que nos indica si la clase es laboratorio (**True**) o si no lo es (**False**).*

Syntax:
```
public o.Laboratorio as Int32
```

Example:
```
Dim dato as Comunes.Materias
dato = New Comunes.Materias(padre.cnxMdi)
dato.Laboratorio = True
```

Materias.Nombre

*Esta propiedad indica el nombre con el que se conoce una materia dentro de la universidad. Se relaciona con el campo de la clase **_CodigoUdb** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información.*

Syntax:
```
public o.Nombre as String
```

Example:
```
Dim dato as Comunes.Materias
dato = New Comunes.Materias(padre.cnxMdi)
dato.Nombre = "Matematica 1"
```

Class PensumCarrera

La clase MateriaEdificio representa a la tabla de la base de datos ppc_plan_pensum_carrera, la cual contiene información de los Planes de estudios por materia impartidos en la Unibersidad

Properties, Events and Methods

Member	Description
carrera	Representa la carrera a la cual pertenece el plan de estudios.
cnn	Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.
Codigo	Código con el que se representa un registro en la base de datos.
estado	Representa el estado de un registro. Si se encuentra en verdadero (***True***), significa que el registro esta activo. Si se encuentra en Falso (***False***) significa que el registro esta inactivo.
nuevo	Indica si el registro (representado por una instancia de la clase) es nuevo (***True***) o ya existe dentro de la base de datos (***False***). Este toma valor automáticamente, dependiendo del valor del parámetro ***codigo*** del metodo new.
plan_pensum	Representa el plan al cual pertenece el plan de estudios (ej. pertence al plan 1998 o pertence al plan 2003)
cargarDatos	Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro ***codigo*** del metodo new. **cargarDatos() as Void**
existe	Retorna verdadero (**True**) si el plan de estudios que se esta consultando existe, y regresa falso (**False**) si el plan de estudio que se esta consultando no existe. Esto es útil para el sistema cuando se intenta ingresar un plan de estudios nuevo, de esta manera si desea ingresar uno ya existente, la función nos indicaría si este ya existe, y de ser así, podríamos mostrar el mensaje de que el plan de estudios ya se encuentra en la base de datos. **existe() as Boolean**
getCarrera	Regresa un objeto del tipo Carreras, el cual representa la carrera a la cual pertenece este plan de estudios. **getCarrera() as Carreras**
getPlan	Regresa un objeto del tipo Planes, el cual representa el plan al cual pertenece este plan de estudios. **getPlan() as Planes**
GuardarDatos	Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos. **GuardarDatos() as Void**
New	Método que crea una nueva instancia de la clase PensumCarrera, puede utilizarse de dos maneras: **New(conexion, codigo)**
ToString	Regresa una cadena de datos (***string***) que representa el nombre del Plan de Estudios (ej. "Ingeniería Biomedica - Plan 1998") **ToString() as String**

Assembly: *comunes.dll*
Namespace: *Comunes*

Inherits from:
System.Object
  Comunes.PensumCarrera

PensumCarrera.carrera

Representa la carrera a la cual pertenece el plan de estudios.

Syntax:
```
public o.carrera as Int32
```

PensumCarrera.cnn

Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.

Syntax:
```
private o.cnn as SqlConnection
```

PensumCarrera.Codigo

Código con el que se representa un registro en la base de datos.

Syntax:
```
public o.Codigo as Int32
```

PensumCarrera.estado

*Representa el estado de un registro. Si se encuentra en verdadero (**True**), significa que el registro esta activo. Si se encuentra en Falso (**False**) significa que el registro esta inactivo.*

Syntax:
```
public o.estado as Boolean
```

PensumCarrera.nuevo

*Indica si el registro (representado por una instancia de la clase) es nuevo (**True**) o ya existe dentro de la base de datos (**False**). Este toma valor automáticamente, dependiendo del valor del parámetro **codigo** del metodo new.*

Syntax:
```
private o.nuevo as Boolean
```

PensumCarrera.plan_pensum

Representa el plan al cual pertenece el plan de estudios (ej. pertence al plan 1998 o pertence al plan 2003)

Syntax:
```
public o.plan_pensum as Int32
```

PensumCarrera.cargarDatos

*Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro **codigo** del metodo new.*

Syntax:
```
cargarDatos() as Void
```

Return: Void

Parameters:

PensumCarrera.existe

*Retorna verdadero (**True**) si el plan de estudios que se esta consultando existe, y regresa falso (**False**) si el plan de estudio que se esta consultando no existe. Esto es útil para el sistema cuando se intenta ingresar un plan de estudios nuevo, de esta manera si desea ingresar uno ya existente, la función nos indicaría si este ya existe, y de ser así, podríamos mostrar el mensaje de que el plan de estudios ya se encuentra en la base de datos.*

Syntax:
```
existe() as Boolean
```

Return: Boolean

Example:
```
Dim dato As Comunes.PensumCarrera
dato.Carrera = 2
dato.Plan_pensum = 5
dato.Estado = True

If dato.Existe = False Then
        dato.Estado = True
        dato.GuardarDatos()
Else
```

```
        MessageBox.Show("Este plan de estudios ya existe en la Base de
Datos", "Ciclos", MessageBoxButtons.OK, MessageBoxIcon.Exclamation)
End If
```

PensumCarrera.getCarrera

Regresa un objeto del tipo Carreras, el cual representa la carrera a la cual pertenece este plan de estudios.

Syntax: `getCarrera() as Carreras`

Return: Carreras

Example:

```
Dim dato as Comunes.PensumCarrera
Dim carrera as Comunes.Carreras

dato = New Comunes.PensumCarrera(padre.cnxMDI, 3)
carrera = dato.getCarrera
messagebox.show(carrera)

'padre.cnxMDI es la conexion a la base de datos
```

PensumCarrera.getPlan

Regresa un objeto del tipo Planes, el cual representa el plan al cual pertenece este plan de estudios.

Syntax: `getPlan() as Planes`

Return: Planes

Example:

```
Dim dato as Comunes.PensumCarrera
Dim Plan as Comunes.Planes

dato = New Comunes.PensumCarrera(padre.cnxMDI, 3)
Plan = dato.getPlan
messagebox.show(Plan)

'padre.cnxMDI es la conexion a la base de datos
```

PensumCarrera.GuardarDatos

Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos.

Syntax: `GuardarDatos() as Void`

Return: Void

Parameters:

Example:

```
Dim dato As Comunes.PensumCarrera
dato.Carrera = 2
dato.Plan_pensum = 5
dato.Estado = True
dato.GuardarDatos
```

PensumCarrera.New

Método que crea una nueva instancia de la clase PensumCarrera, puede utilizarse de dos maneras:

1. Cuando se desea crear una instancia vacía para llenar sus atributos.
2. Cuando ya se sabe que registro se desea cargar de la base de datos. En este caso la instancia de la clase se conecta a la base de datos y extrae los datos necesarios.

Syntax: `New(conexion, codigo)`

Parameters: **conexion As SqlClient.SqlConnection**
Optional codigo As Integer

Example:

```
'Para un registro que se sabe que ya existe en la base de datos
Dim dato as Comunes.PensumCarrera
dato = New Comunes.PensumCarrera(padre.cnxMDI, 1)

'Para un registro nuevo que se desea agregar a la base de datos
Dim dato as Comunes.PensumCarrera
dato = New Comunes.PensumCarrera(padre.cnxMDI)

'padre.cnxMDI es la conexion a la base de datos
```

PensumCarrera.ToString

*Regresa una cadena de datos (**string**) que representa el nombre del Plan de Estudios (ej. "Ingeniería Biomedica - Plan 1998")*

Syntax:

```
ToString() as String
```

Return: String

Parameters:

Class Planes

La clase MateriaEdificio representa a la tabla de la base de datos ppe_plan_pensum, la cual contiene información de los diferentes planes que se encuentran vigentes en la Unversidad.

Properties, Events and Methods

Member	Description
cnn	Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.
Codigo	Código con el que se representa un registro en la base de datos.
Estado	Representa el estado de un registro. Si se encuentra en verdadero (***True***), significa que el registro esta activo. Si se encuentra en Falso (***False***) significa que el registro esta inactivo.
nuevo	Indica si el registro (representado por una instancia de la clase) es nuevo (***True***) o ya existe dentro de la base de datos (***False***). Este toma valor automáticamente, dependiendo del valor del parámetro ***codigo*** del metodo new.
cargarDatos	Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro ***codigo*** del metodo new. **cargarDatos() as Void**
GuardarDatos	Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos. **GuardarDatos() as Void**
New	Método que crea una nueva instancia de la clase Planes, puede utilizarse de dos maneras: **o.Planes.New(conexion, codigo)**
ToString	Regresa una cadena de datos (***string***) que representa el nombre del Plan (ej. "Plan 1998") **ToString() as String**
Nombre	Esta propiedad indica el nombre del Plan. Se relaciona con el campo de la clase ***_nombre*** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información. Ambos representan nombre con el que se almacena el registro en la base de datos.

Assembly: *comunes.dll*
Namespace: *Comunes*

Inherits from:
System.Object
  Comunes.Planes

Planes.cnn

Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.

Syntax:
```
private o.cnn as SqlConnection
```

Planes.Codigo

Código con el que se representa un registro en la base de datos.

Syntax:
```
public o.Codigo as Int32
```

Planes.Estado

*Representa el estado de un registro. Si se encuentra en verdadero (**True**), significa que el registro esta activo. Si se encuentra en Falso (**False**) significa que el registro esta inactivo.*

Syntax:
```
public o.Estado as Boolean
```

Planes.nuevo

*Indica si el registro (representado por una instancia de la clase) es nuevo (**True**) o ya existe dentro de la base de datos (**False**). Este toma valor automáticamente, dependiendo del valor del parámetro **codigo** del metodo new.*

Syntax:
```
private o.nuevo as Boolean
```

Planes.cargarDatos

*Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro **codigo** del metodo new.*

Syntax:
```
cargarDatos() as Void
```

Return: Void

Planes.GuardarDatos

Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos.

Syntax:
```
GuardarDatos() as Void
```

Return: Void

Planes.New

Método que crea una nueva instancia de la clase Planes, puede utilizarse de dos maneras:

1. Cuando se desea crear una instancia vacía para llenar sus atributos.
2. Cuando ya se sabe que registro se desea cargar de la base de datos. En este caso la instancia de la clase se conecta a la base de datos y extrae los datos necesarios.

Syntax:
```
o.Planes.New(conexion, codigo)
```

Parameters: **conexion As SqlClient.SqlConnection**
Optional codigo As Integer

Example:
```
'Para un registro que se sabe que ya existe en la base de datos
Dim dato as Comunes.Planes
dato = New Comunes.Planes(padre.cnxMDI, 1)

'Para un registro nuevo que se desea agregar a la base de datos
Dim dato as Comunes.Planes
dato = New Comunes.Planes(padre.cnxMDI)

'padre.cnxMDI es la conexion a la base de datos
```

Planes.ToString

*Regresa una cadena de datos (**string**) que representa el nombre del Plan (ej. "Plan 1998")*

Syntax:
```
ToString() as String
```

Return: String

Planes.Nombre

*Esta propiedad indica el nombre del Plan. Se relaciona con el campo de la clase **_nombre** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información. Ambos representan nombre con el que se almacena el registro en la base de datos.*

Syntax:
```
public o.Nombre as String
```

Example:
```
Dim dato as Comunes.Planes
dato = New Comunes.Planes(padre.cnxMdi)
dato.Nombre = "Plan 2003"

'padre.cnxMDI es la conexion a la base de datos
```

Class Reserva

La clase Representa representa a la tabla de la base de datos res_reserva, la cual contiene información acerca de las diferentes aulas que han sido reservadas en la Universidad.

Properties, Events and Methods

Member	Description
Aula	Representa el Aula que se esta reservando, es un valor numérico y contiene el código de la misma.
cnn	Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.
Codigo	Código con el que se representa un registro en la base de datos.
fecha	Representa la fecha para la cual esta reserva ha sido fijada, es un dato tipo DateTime.
hfin	Indica la hora en que se da por terminada la reserva del aula específica. Este es un valor númerico el cual guarda el código del cuarto de hora.
hinicio	Indica la hora en que se da por iniciada la reserva del aula específica. Este es un valor númerico el cual guarda el código del cuarto de hora.
nuevo	Indica si el registro (representado por una instancia de la clase) es nuevo (***True***) o ya existe dentro de la base de datos (***False***). Este toma valor automáticamente, dependiendo del valor del parámetro ***codigo*** del metodo new.
cargarDatos	Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro ***codigo*** del metodo new. **cargarDatos() as Void**
getAulasOcupadas	Regresa todas las aulas que están siendo ocupadas por grupos de clase en la hora en la cual se desea llevar a cabo la reserva, tomando en cuenta el día y el edificio donde se desea reservar. **getAulasOcupadas(edificio as Int32, ciclo as Int32) as Collection**
getAulasReservadas	Regresa todas las aulas que están reservadas en la fecha y hora en la cual se desea llevar a cabo la reserva, tomando en cuenta el edificio donde se desea reservar. **getAulasReservadas() as Collection**
GuardarDatos	Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos. **GuardarDatos() as Void**
New(conexion,codigo)	Método que crea una nueva instancia de la clase Reserva, puede utilizarse de dos maneras: **o.Reserva.New(conexion,codigo)**
ToString	Regresa una cadena de datos (***string***) que representa la fecha de la reserva y la hora en que fue reservada. (ej. "23/08/2005 de 5:00 pm a 7:00 pm, reservado por Roberto Burgos"). **ToString() as String**
a_nombre	Esta propiedad indica el nombre de la persona que reserva el aula. Se relaciona con el campo de la clase ***_anombre*** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información.
dia	Esta propiedad indica el día en que se reservo el aula. Se relaciona con el campo de la clase ***_dia*** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información.

Assembly: *comunes.dll*
Namespace: *Comunes*

Inherits from:

System.Object
Comunes.Reserva

Reserva.Aula
Representa el Aula que se esta reservando, es un valor numérico y contiene el código de la misma.

Syntax:
```
public o.Aula as Int32
```

Reserva.cnn
Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.

Syntax:
```
private o.cnn as SqlConnection
```

Reserva.Codigo
Código con el que se representa un registro en la base de datos.

Syntax:
```
public o.Codigo as Int32
```

Reserva.fecha
Representa la fecha para la cual esta reserva ha sido fijada, es un dato tipo DateTime.

Syntax:
```
public o.fecha as DateTime
```

Reserva.hfin
Indica la hora en que se da por terminada la reserva del aula específica. Este es un valor númerico el cual guarda el código del cuarto de hora.

Syntax:
```
public o.hfin as Int32
```

Reserva.hinicio
Indica la hora en que se da por iniciada la reserva del aula específica. Este es un valor númerico el cual guarda el código del cuarto de hora.

Syntax:
```
public o.hinicio as Int32
```

Reserva.nuevo
*Indica si el registro (representado por una instancia de la clase) es nuevo (**True**) o ya existe dentro de la base de datos (**False**). Este toma valor automáticamente, dependiendo del valor del parámetro **codigo** del metodo new.*

Syntax:
```
private o.nuevo as Boolean
```

Reserva.cargarDatos
*Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro **codigo** del metodo new.*

Syntax:
```
cargarDatos() as Void
```

Return: Void

Reserva.getAulasOcupadas
Regresa todas las aulas que están siendo ocupadas por grupos de clase en la hora en la cual se desea llevar a cabo la reserva, tomando en cuenta el día y el edificio donde se desea reservar.

Syntax:
```
getAulasOcupadas(edificio as Int32, ciclo as Int32) as Collection
```

Return: Collection

Parameters: **edificio as Int32**
ciclo as Int32

Example:
```
Dim reservas as Comunes.Reserva
```

```
dim aulas_ocupadas as New Collection

reservas = New Comunes.Reserva(padre.cnxMDI)
reservas.Aula = 52
reservas.Fecha = now()
reservas.a_nombre = "Roberto Burgos"
reservas.hinicio = 23
reservas.hfin = 43
aulas_ocupadas = reservas.getAulasOcupadas(3,4)

'padre.cnxMDI es la conexion a la base de datos
```

Reserva.getAulasReservadas

Regresa todas las aulas que están reservadas en la fecha y hora en la cual se desea llevar a cabo la reserva, tomando en cuenta el edificio donde se desea reservar.

Syntax: `getAulasReservadas() as Collection`

Return: Collection

Example:

```
Dim reservas as Comunes.Reserva
dim aulas_reservadas as New Collection

reservas = New Comunes.Reserva(padre.cnxMDI)
reservas.Aula = 52
reservas.Fecha = now()
reservas.a_nombre = "Roberto Burgos"
reservas.hinicio = 23
reservas.hfin = 43
aulas_reservadas = reservas.getAulasReservadas()

'padre.cnxMDI es la conexion a la base de datos
```

Reserva.GuardarDatos

Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos.

Syntax: `GuardarDatos() as Void`

Return: Void

Example:

```
Dim reservas as Comunes.Reserva

reservas = New Comunes.Reserva(padre.cnxMDI)
reservas.Aula = 52
reservas.Fecha = now()
reservas.a_nombre = "Roberto Burgos"
reservas.hinicio = 23
reservas.hfin = 43
reservas.GuardarDatos()

'padre.cnxMDI es la conexion a la base de datos
```

Reserva.New

Método que crea una nueva instancia de la clase Reserva, puede utilizarse de dos maneras:

1. Cuando se desea crear una instancia vacía para llenar sus atributos.
2. Cuando ya se sabe que registro se desea cargar de la base de datos. En este caso la instancia de la clase se conecta a la base de datos y extrae los datos necesarios.

Syntax: `o.Reserva.New(conexion,codigo)`

Example:
```
'Para un registro que se sabe que ya existe en la base de datos
Dim dato as Comunes.Reserva
dato = New Comunes.Reserva(padre.cnxMDI, 1)

'Para un registro nuevo que se desea agregar a la base de datos
Dim dato as Comunes.Reserva
dato = New Comunes.Reserva(padre.cnxMDI)

'padre.cnxMDI es la conexion a la base de datos
```

Reserva.ToString

*Regresa una cadena de datos (**string**) que representa la fecha de la reserva y la hora en que fue reservada. (ej. "23/08/2005 de 5:00 pm a 7:00 pm, reservado por Roberto Burgos").*

Syntax: `ToString() as String`

Return: String

Example:
```
Dim dato as Comunes.Reserva
dato = New Comunes.Reserva(padre.cnxMDI, 3)
messagebox.Show(dato.toString)

'padre.cnxMDI es la conexion a la base de datos
```

Reserva.a_nombre

*Esta propiedad indica el nombre de la persona que reserva el aula. Se relaciona con el campo de la clase **_anombre** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información.*

Syntax: `public o.a_nombre as String`

Example:
```
Dim dato as Comunes.Reserva
dato = New Comunes.Reserva(padre.cnxMdi)
dato.a_nombre = "Roberto Burgos"

'padre.cnxMDI es la conexion a la base de datos
```

Reserva.dia

*Esta propiedad indica el día en que se reservo el aula. Se relaciona con el campo de la clase **_dia** (que es de uso exclusivo de la clase), el cual es de tipo entero y contiene exactamente la misma información.*

Syntax: `public o.dia as Int32`

Example:
```
Dim dato as Comunes.Reserva
dato = New Comunes.Reserva(padre.cnxMdi)
dato.dia = 0 'Representando el dia Lunes

'padre.cnxMDI es la conexion a la base de datos
```

Class Usuarios

La clase Usuarios representa a la tabla de la base de datos usu_usuarios, la cual contiene información de los diferentes docenten que imparten catedra en la Universidad.

Properties, Events and Methods

Member	Description
cnn	Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.
Codigo	Código con el que se representa un registro en la base de datos.
Estado	Representa el estado de un registro. Si se encuentra en verdadero (***True***), significa que el registro esta activo. Si se encuentra en Falso (***False***) significa que el registro esta inactivo.
nuevo	Indica si el registro (representado por una instancia de la clase) es nuevo (***True***) o ya existe dentro de la base de datos (***False***). Este toma valor automáticamente, dependiendo del valor del parámetro ***codigo*** del metodo new.
permiso	Un valor booleando que indica si el usuario tiene permisos para utilizar el sistema administrativo (True) o si tiene permisos de usar el sistema de asignación de aulas (False).
cargarDatos	Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro ***codigo*** del metodo new. **cargarDatos() as Void**
GuardarDatos	Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos. **GuardarDatos() as Void**
New(conexion,codigo)	Método que crea una nueva instancia de la clase Usuarios, puede utilizarse de dos maneras: **o.Usuarios.New(conexion,codigo)**
Apellido	Esta propiedad indica el apellido del usuario. Se relaciona con el campo de la clase ***_apellido*** (que es de uso exclusivo de la clase), el cual es de tipo carácter y contiene exactamente la misma información.
Contrasena	Esta propiedad indica la contraseña que se le a asignado al usuario, ya sea para ingresar al sistema administrativo o al sistema de asignación de aulas. Se relaciona con el campo de la clase ***_contrasena*** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información.
Login	Esta propiedad indica el login que se le ha asignado al usuario y con el cual podra accesar, dependiendo de los permisos al sistema administrativo o al sistema de asignacion de aulas. Se relaciona con el campo de la clase ***_login*** (que es de uso exclusivo de la clase).
Nombre	Esta propiedad indica el nombre el docente. Se relaciona con el campo de la clase ***_nombre*** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información.

Assembly: *comunes.dll*
Namespace: *Comunes*

Inherits from:
System.Object
  Comunes.Usuarios

Usuarios.cnn

Conexión con la cual la clase se conectara a la base de datos para consultar, insertar, modificar o eliminar registros.

Syntax:
```
private o.cnn as SqlConnection
```

Usuarios.Codigo

Código con el que se representa un registro en la base de datos.

Syntax:
```
public o.Codigo as Int32
```

Usuarios.Estado

*Representa el estado de un registro. Si se encuentra en verdadero (**True**), significa que el registro esta activo. Si se encuentra en Falso (**False**) significa que el registro esta inactivo.*

Syntax:
```
public o.Estado as Boolean
```

Usuarios.nuevo

*Indica si el registro (representado por una instancia de la clase) es nuevo (**True**) o ya existe dentro de la base de datos (**False**). Este toma valor automáticamente, dependiendo del valor del parámetro **codigo** del metodo new.*

Syntax:
```
private o.nuevo as Boolean
```

Usuarios.permiso

Un valor booleando que indica si el usuario tiene permisos para utilizar el sistema administrativo (True) o si tiene permisos de usar el sistema de asignación de aulas (False).

Syntax:
```
public o.permiso as Boolean
```

Usuarios. New

Método que crea una nueva instancia de la clase Usuarios, puede utilizarse de dos maneras:

1. Cuando se desea crear una instancia vacía para llenar sus atributos.
2. Cuando ya se sabe que registro se desea cargar de la base de datos. En este caso la instancia de la clase se conecta a la base de datos y extrae los datos necesarios.

Syntax:
```
o.Usuarios.New(conexion,codigo)
```

Example:
```
'Para un registro que se sabe que ya existe en la base de datos
Dim dato as Comunes.Usuarios
dato = New Comunes.Usuarios(padre.cnxMDI, 1)

'Para un registro nuevo que se desea agregar a la base de datos
Dim dato as Comunes.Usuarios
dato = New Comunes.Usuarios(padre.cnxMDI)

'padre.cnxMDI es la conexion a la base de datos
```

Usuarios.cargarDatos

*Realiza una consulta a la base de datos para extraer y llenar la instancia de la clase con los datos correctos pedidos por el usuario. Este método se ejecuta automáticamente dependiendo si se especifica o no el parámetro **codigo** del metodo new.*

Syntax:
```
cargarDatos() as Void
```

Return: Void

Usuarios.GuardarDatos

Inserta o Modifica los datos contenidos en la instancia de la clase a la tabla respectiva en la base de datos.

Syntax: `GuardarDatos() as Void`

Return: Void

Example:

```
Dim Usuario as Comunes.Usuarios

Usuario = New Comunes.Usuarios(padre.cnxMDI)
Usuario.Nombre = "Roberto"
Usuario.Apellido = "Burgos"
Usuario.Login = "rbto1"
Usuario.Contrasena = "12345"
Usuario.GuardarDatos()

'padre.cnxMDI es la conexion a la base de datos
```

Usuarios.Apellido

*Esta propiedad indica el apellido del usuario. Se relaciona con el campo de la clase **_apellido** (que es de uso exclusivo de la clase), el cual es de tipo carácter y contiene exactamente la misma información.*

Syntax: `public o.Apellido as String`

Example:

```
Dim dato as Comunes.Usuarios
dato = New Comunes.Usuarios(padre.cnxMdi)
dato.Apellido = "Burgos"

'padre.cnxMDI es la conexion a la base de datos
```

Usuarios.Contrasena

*Esta propiedad indica la contraseña que se le a asignado al usuario, ya sea para ingresar al sistema administrativo o al sistema de asignación de aulas. Se relaciona con el campo de la clase **_contrasena** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información.*

***NOTA:** Esta contraseña se guarda encriptada en la base de datos, por cuestiones de seguridad. Esto se hace por medio de las funciones GetEncryptedData y GetDecryptedData.*

Syntax: `public o.Contrasena as String`

Example:

```
Dim dato as Comunes.Usuarios
dato = New Comunes.Usuarios(padre.cnxMdi)
dato.Contrasena = "12345"

'padre.cnxMDI es la conexion a la base de datos
```

Usuarios.Login

*Esta propiedad indica el login que se le ha asignado al usuario y con el cual podra accesar, dependiendo de los permisos al sistema administrativo o al sistema de asignacion de aulas. Se relaciona con el campo de la clase **_login** (que es de uso exclusivo de la clase).*

Syntax: `public o.Login as String`

Example:

```
Dim dato as Comunes.Usuarios
dato = New Comunes.Usuarios(padre.cnxMdi)
dato.Contrasena = "roberto"

'padre.cnxMDI es la conexion a la base de datos
```

Usuarios.Nombre

*Esta propiedad indica el nombre el docente. Se relaciona con el campo de la clase **_nombre** (que es de uso exclusivo de la clase), el cual es de tipo caracter y contiene exactamente la misma información.*

Syntax:
```
public o.Nombre as String
```

Example:
```
Dim dato as Comunes.Usuarios
dato = New Comunes.Usuarios(padre.cnxMdi)
dato.Contrasena = "Roberto"

'padre.cnxMDI es la conexion a la base de datos
```

Base de Datos SAPH

A continuación se explica el esquema de la Base de Datos

aul_aulas

Catalogo de Aulas

Table Indexes

- ***PK_adm_aul_aulas***
  Index Key: aul_codigo

Table Fields

Field	Description
aul_capacidad	Capacidad en cantidad de alumnos que posee el aula **int**
aul_codigo	Codigo identificador del registro **int identity**
aul_edificio	Edificio a la que pertenece el aula **int**
aul_estado	Estado del Aula (0=inactiva, 1=activa) **bit**
aul_nombre	Nombre del aula **char(10)**

car_carrera

Catalogo de Carreras exisitentes en la Universidad

Table Indexes

- ***PK_udb_car_carrera***
  Index Key: car_codigo

Table Fields

Field	Description
car_codigo	Codigo de identificacion del registro **int identity**
car_estado	Estado de la carrera (0=inactiva, 1=activa) **bit**
car_nivelmaximo	Indica el nivel maximo de la carrera **int**
car_nombre	Indica el nombre de la Carrera **varchar(40)**

edi_edificios

Catalogo de los edificios en uso en la universidad

Table Indexes

- ***PK_adm_edi_edificios***
  Index Key: edi_codigo

Table Fields

Field	Description
edi_codigo	Codigo identificador del registro **int identity**
edi_estado	Estado del Edificio (0=inactivo, 1=activo) **bit**
edi_nombre	Nombre del edificio **char(20)**

esc_escuelas

Catalogo de escuelas existentes en la Universidad

Table Indexes

- ***PK_udb_esc_escuelas***
  Index Key: esc_codigo

Table Fields

Field	Description
esc_codigo	Codigo identificador del registro **int identity**
esc_estado	Estado de la escuela (0=inactiva, 1=activa)

	bit
esc_nombre	Nombre de la Escuela **varchar(30)**

hcl_horas_clase

Detalle de los Horarios de Materias para un ciclo y año especifico, con sus respectivas aulas

Table Indexes

- ***PK_hor_hcl_horas_clase***
  Index Key: hcl_codigo

Table Fields

Field	Description
hcl_aula	Identifica el aula que utilizará el grupo de materia en un horario **int**
hcl_codigo	Codigo identificador de registro **int identity**
hcl_dia	Identifica el día en un grupo de materia será impartido **int**
hcl_dt_ca	fecha y hora de creacion de las aulas **datetime**
hcl_dt_ch	Fecha y hora de creacion del registro de los horarios **datetime**
hcl_dt_ma	fecha y hora de modificacion de las aulas **datetime**
hcl_Hfin	Indica la hora de finalizacion de clase del grupo de materia **int**
hcl_Hinicio	Indica la hora de inicio de clase del grupo de materia **int**
hcl_materiaq	Identifica el horario de Grupo de Materia **int**
hcl_usr_ca	usuario que asigno las aulas **char(10)**
hcl_usr_ch	Usuario que creo el registro de horarios **int**
hcl_usr_ma	usuario que modifico las aulas **int**

hoc_horario_ciclo

Tabla Maestra de los Horarios de un ciclo en un año especifico.

Table Indexes

- ***PK_adm_cic_ciclos***
  Index Key: hoc_codigo

Table Fields

Field	Description
hoc_anio	Año del Ciclo **int**
hoc_ciclo	Ciclo del año (1=primero,2=segundo o 3=interciclo) **int**
hoc_codigo	Codigo identificador de registro **int identity**
hoc_estado	Estado del horario del ciclo (0= inactivo, 1=activo y 3=en uso) **bit**

mat_materias

Catalogo de Materias

Table Indexes

- ***PK_udb_mat_materias***
  Index Key: mat_codigo

Table Fields

Field	Description
mat_codigo	Codigo Identificador de cada registro **int identity**
mat_codudb	Codigo de la Materia de acuerdo a los estandares de la UDB **varchar(6)**
mat_escuela	Codigo de la materia a que pertenece la materia **int**
mat_estado	Estado de la materia (0=inactiva, 1=activa) **bit**
mat_horas_semana	Cantidad de horas semanales que será impartida la materia **int**
mat_laboratorio	Codigo de la materia de la que es laboratorio **int**
mat_nombre	Nombre de la Materia **varchar(50)**

med_materias_edificios

Table Indexes

- ***PK_med_materias_edificios***
  Index Key: med_codigo

Table Fields

Field	Description
med_codigo	Codigo identificador de cada registro **int identity**
med_edificio	Codigo del edificio **int**
med_materia	Codigo de la materia **int**

mgr_materia_grupo

Detalle de Grupos de Materias

Table Indexes

- ***PK_hor_mgr_materia_grupo***
  Index Key: mgr_codigo

Table Fields

Field	Description
mgr_capacidad	Cantidad de alumnos que recibirá el grupo de materia **int**
mgr_ciclo	Ciclo del año a que pertenece el grupo de materia **int**
mgr_codigo	Codigo identificador de registro **int identity**
mgr_grupo	Grupo identificador del grupo de materia **int**
mgr_materia	Materia del grupo de materia **int**
mgr_profesor	Profesor que impartirá el grupo de materia **int**

pho_profesor_horario

Detalle del horario de disponibilidad de profesores

Table Indexes

- ***PK_web_pho_profesor_horario***
  Index Key: pho_codigo

Table Fields

Field	Description
pho_ciclo	Ciclo en el cual se ha agregado la disponibilidad **int**
pho_codigo	Codigo identificador del registro **int identity**
pho_dia	Indica el día en que tiene algun tiempo disponible **int**
pho_Hfin	Indica la hora final del tiempo disponible **int**
pho_Hinicio	Indica la hora inicial en la que tiene tiempo disponible **int**
pho_profesor	Profesor al que corresponde el registro **int**

ppc_plan_pensum_carrera

Catalogo de Pensums

Table Indexes

- ***PK_adm_ppc_plan_pensum_carrera***
  Index Key: ppc_codigo

Table Fields

Field	Description
ppc_carrera	Identifica la carrera del plan del pensum **int**
ppc_codigo	Codigo identificador del registro **int identity**
ppc_estado	Estado del registro (0=activo, 1=inactivo) **bit**
ppc_plan_pensum	Identifica al plan de pensum que pertenece la carrera **int**

ppe_plan_pensum

Catalogo de Planes de Pensum

Table Indexes

- ***PK_adm_ppe_plan_pensum***
  Index Key: ppe_codigo

Table Fields

Field	Description
ppe_codigo	Codigo identificador de Registro **int identity**
ppe_estado	Estado del plan del pensum (0=inactivo, 1=activo) **bit**
ppe_nombre	Nombre del Plan de Pensum **char(10)**

ppm_plan_pensum_materia

Detalle de Materias en un Plan de Pensum

Table Indexes

- ***PK_adm_plan_pensum_materia***
  Index Key: ppm_codigo

Table Fields

Field	Description
ppm_ciclo	Ciclo en el Plan de Pensum **int**
ppm_codigo	Codigo identificador del registro **int identity**
ppm_materia	Materia del Plan de Pensum **int**
ppm_plan_pensum	Plan de Pensum al que pertenece la materia **int**

pro_profesores

Catalogo de Profesores, incluye sus respectivos login y clave para el modulo Web

Table Indexes

- ***PK_hor_pro_profesores***
  Index Key: pro_codigo

Table Fields

Field	Description
pro_apellido	Apellidos del Profesor **varchar(75)**
pro_clave	Clave del profesor para poder utilizar el modulo web **varchar(25)**
pro_codigo	Codigo identificador de registro **int identity**
pro_email	Correo Electronico del Profesor **varchar(100)**
pro_encargado	Establece a este profesor como usuario encargado de la creación de horarios **bit**
pro_escuela	Escuela a la que pertenece el profesor **int**
pro_estado	Estado de un profesor (0=inactivo, 1=activo) **bit**
pro_es_instructor	La persona es un instructor (1=instructor, 0=profesor) **bit**
pro_login	Login del profesor para poder utilizar el modulo web **varchar(10)**
pro_nombre	Nombres del profesor **varchar(75)**
pro_tel	Telefonos del Profesor **char(8)**

res_reserva

Table Indexes

- ***PK_res_reserva***
  Index Key: res_codigo

Table Fields

Field	Description
res_anombre	A nombre de quien se realiza la reserva **varchar(30)**
res_aula	Aula que se esta reservando **int**
res_codigo	Codigo identificador de registro
res_fecha	Fecha en que se realiza la rerserva **datetime**
res_hfin	Hora en que termina la reserva **int**
res_hinicio	Hora en que inicia la reserva **int**

usu_usuarios

Catalogo de Usuarios para el Modulo de Horarios

Table Indexes

- ***PK_udb_usu_usuarios***
  Index Key: usu_codigo

Table Fields

Field	Description
usu_apellido	Apellido del usuario **varchar(50)**
usu_clave	Clave del usuario **varchar(10)**
usu_codigo	Codigo identificador del registro **int identity**
usu_estado	Estado del usuario (0=inactivo, 1=activo) **bit**
usu_login	Login asignado **varchar(10)**
usu_nombre	Nombre del usuario **varchar(50)**
usu_permiso	Indica si puede usar el modulo administrativo o el asignacion de aulas **bit**