Ingreso al sistema y codificación de los sub_sistemas (Parte I)

Sitio:	Agencia de Habilidades para el Futuro	Imprimido por:	Eduardo Moreno
Curso:	Desarrollo de Sistemas Orientado a Objetos 1° D	Día:	miércoles, 14 de mayo de 2025, 23:10
Libro:	Ingreso al sistema y codificación de los sub_sistemas (Parte I)		

Tabla de contenidos

1. Preguntas orientadoras

2. Introducción

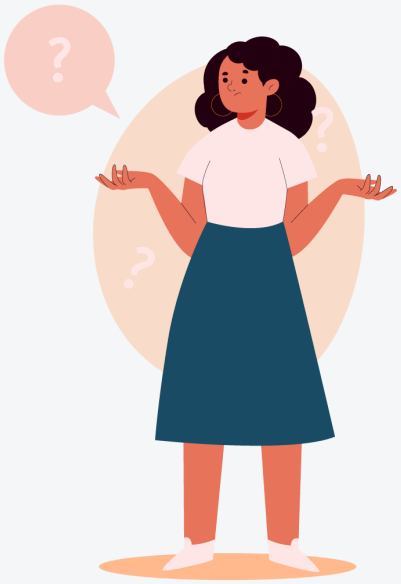
3. Ventana Principal del Sistema
3.1. Llamada a la ventana “Principal”

4. Programación de los Procesos
4.1. Inscribir Postulante (insert a la base)
4.2. Método para el Nuevo Postulante

5. En resumen

Preguntas orientadoras

- ¿Cómo codifico el ingreso de datos a las entidades?
- ¿Creamos plantillas de clases relacionadas a las entidades?
- ¿Usamos sentencias SQL para manipular los datos?
- ¿Es conveniente programas Store Procedure y vincularlos con C#?



Introducción

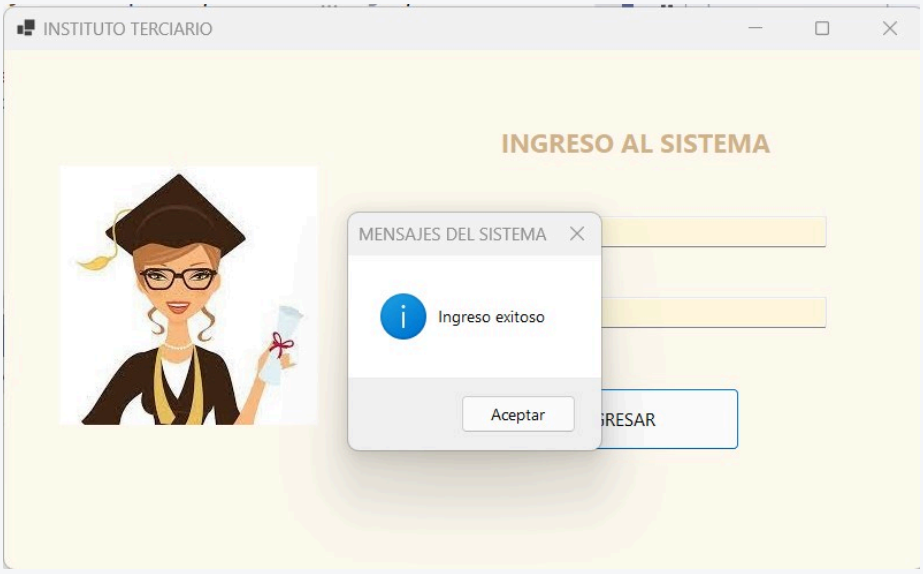
Seguramente a esta altura encuentres en tu mesa de trabajo mucho material y dudes de como continuar.

Ya diste el primer paso con la codificación de la **conexión** y la validación de la ventana de login. El botón de ingresar de esa ventana no tiene funcionalidad y eso es lo que veremos esta semana, además de darle un orden de prioridad a los procesos para ser programados.

En todo momento no olvides colocarte en el tol del usuario de tu desarrollo para interactuar con él.

¿Interactuar con el usuario? ¡Si!, tal cual lo escuchas.

El sistema se comunica con el exterior a través de mensajes, de alertas y esos mensajes o alertas no siempre están plasmadas en la documentación previa (plantillas de caso de uso).



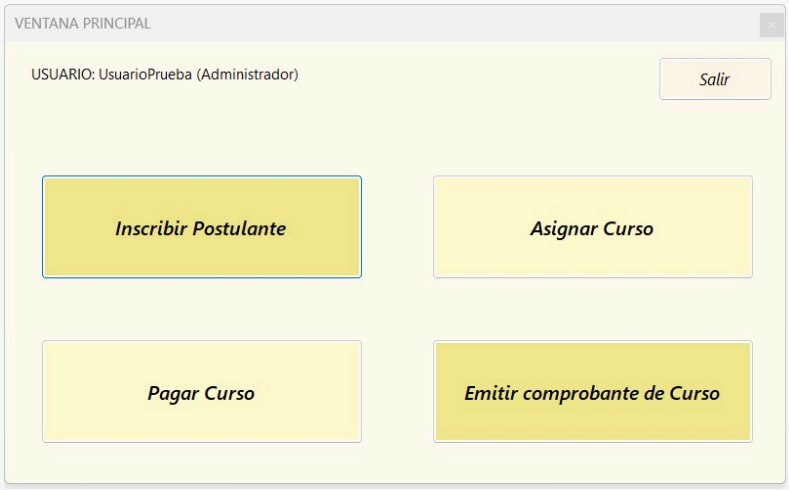
No perdamos tiempo … ¡Vamos a programar la ventana **Principal!**

Ventana Principal del Sistema

Esta ventana es la primera que visualiza el usuario después de la confirmación del login.

El diseño tiene que ser claro y no confundir a la hora de seleccionar las acciones que nos ofrece el sistema.

Sigamos con la explicación del sistema del **Instituto Terciario.**



Como vemos en la imagen encontramos en el centro los 4 procesos que tiene este proyecto:

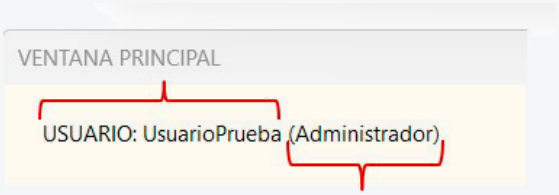
- Inscribir Postulante
- Asignar Curso
- Pagar Curso
- Emitir comprobante de Curso

En el extremo superior derecho aparece un botón con la leyenda **Salir.** En el evento Click la única línea de código dice

```
Application.Exit();
```

Esta sentencia notifica a todos los surtidores de mensajes que deben terminar y, a continuación, cierra todas las ventanas de la aplicación.

En el extremo superior izquierdo muestra el nombre del usuario que ingreso en el login y el rol.



¿Por qué aparece el usuario y el rol? ¿Los sistemas lo deben mostrar?

La respuesta es **No**, depende del diseño del programador y lo que requiera el cliente que solicita el sistema.

Entonces, ¿por qué aparece?

Aparece para que veas en el código como llevar datos de un formulario a otro.

Vamos a loguearnos y pasar a la ventana Principal

Llamada a la Ventana Principal

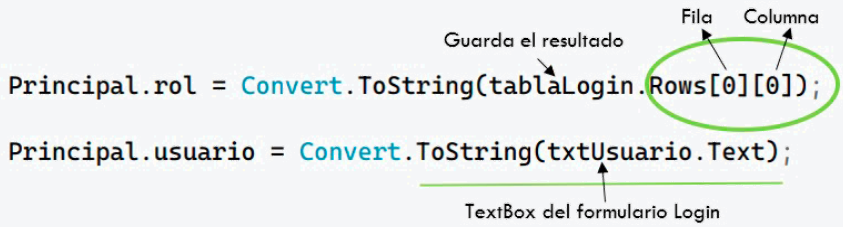
Lo primero que debemos hacer es crear una instancia al formulario que tiene el diseño de la ventana “Principal”, para manipular cualquier objeto instanciado en él y para invocarlo.

```
frmPrincipal Principal = new frmPrincipal();
```

Veamos como escribimos en el **label** que figura en el formulario principal el nombre de usuario y el rol que tiene.

En el formulario principal se declararon dos variables, una para cada uno de los datos.

En la variable **usuario** asignamos el nombre ingresado en el textBox de la ventana del login (ya validado) y en la variable **rol** la salida que consideramos en la query del procedimiento almacenado que es el nombre del rol (única fila y única columna proyectada en el select).



El código de la imagen está en el evento click del botón **Ingresar** del formulario del login, esta aclaración es importante por lo siguiente. Para referenciar a las variables **rol** y **usuario** se antepone el nombre del formulario que los contiene, en este caso es la instancia al formulario principal al que llamamos **Principal**.

Carpeta de materiales. El código completo esta en los siguientes documentos

S9_Codigo_btnIngresar.pdf y S9_Codigo_frmPrincipal.pdf

Pasemos a los procesos, pero, ¿por cuál comenzamos? …

Programación de los Procesos

Llegamos a la ventana principal, están los botones a la espera de recibir el evento que despierte a las líneas de código.

Para saber por cual comenzar a programar te sugiero lo siguiente:

> Mira la mesa de trabajo y ten a la vista las plantillas de caso de uso. Lee con atención las precondiciones y las postcondiciones. Ellas identifican cual es el escenario antes y después de la ejecución del caso, podrás observar que las postcondiciones pueden ser precondiciones de otro caso.

Pues bien, esta observación es la que determina el orden en el cual es conveniente programar los procesos para que a medida vayas avanzado puedas probarlo sin errores.

En nuestro ejemplo debemos comenzar con **Inscribir Postulante**, ya que si no tenemos postulante no podemos asignar curso; si no asignamos curso no lo podemos pagar y si no pagamos no se puede emitir el comprobante de curso.

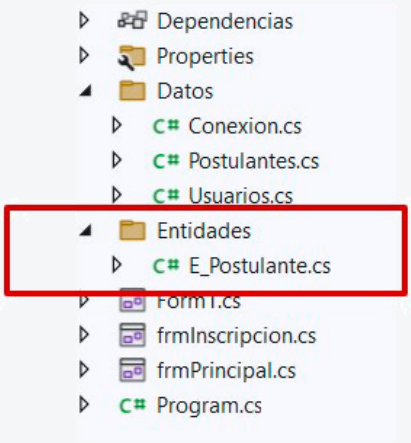
Vamos al Click de Inscribir Postulante

Inscribir Postulante (insert a la base)

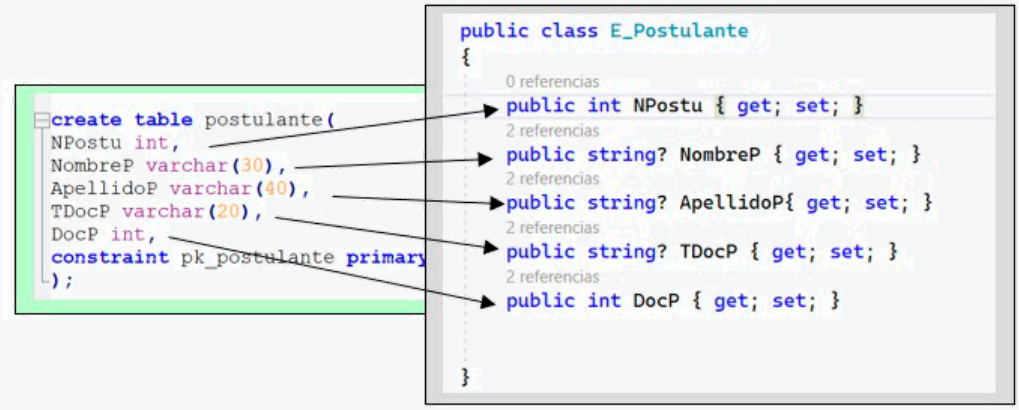
En el proceso de **Inscribir Postulante** se inserta una fila nueva a la tabla Postulante de la base de datos **Instituto.**

El insert lo hacemos a través de la clase Postulante, creando el método correspondiente, y para actualizar la tabla usamos un store procedure ya que la primary key no es auto incrementable y además hay que controlar que no esté almacenado.

Creamos en la Solución una carpeta que llamamos **Entidades** y dentro colocamos una clase por cada entidad de la base de datos que sea actualizada desde el código del sistema. Para identificar estas clases comenzamos a llamarla con la letra E seguida de un guion bajo y el nombre de la entidad propiamente dicha. En nuestro caso es **E_Postulante**.



La clase E_Postulante contiene el get y set de cada uno de los atributos de la tabla **Postulante**, respetando el mismo nombre.



Carpeta de materiales

El código esta en **S9_Clase_E:Postulante.pdf** y la lógica del Store Procedure en **ProcedureNuevoPostulante.sql**

Codifiquemos el método ...

Método para el Nuevo Postulante

Para codificar el método que permite la carga del nuevo postulante creamos la clase Postulante y al método lo llamamos **Nuevo_Postu** y su parámetro es una variable del tipo E_Postulante.

En el código el CommandType es del tipo Store.Procedure.

En la codificación del Store Procedure usamos una variable de tipo **Out** para conocer con que número fue registrado el nuevo ingreso. Esta variable nos permite también conocer si los datos ingresados ya existen en la tabla. En este último caso la variable toma el valor 1.

Una parte de la codificación es igual a la usada con la clase usuario, obviamente cambian los atributos, la parte que es diferente está vinculada al parámetro de salida (Out).

La llamada al método se desarrolla en el evento click del botón **Ingresar** del formulario **Inscripción**.

Se controla que todos los datos requeridos (identificados con un *) tengan información, si están completos se invoca al método, caso contrario se informa con un mensaje para que el usuario complete los datos.

Este formulario presenta el botón **Limpiar** que borra todo el contenido de los objetos y el botón **Volver** para retornar a la pantalla **Principal.**

El diseño es el siguiente:

Carpeta de materiales: El código completo esta en los siguientes documentos: S9_Codigo_frmInscripcion.pdf, S9_Clase_Postulante.pdf

En resumen ...

En resumen

A medida que avances y obtengas respuesta del sistema, te darán ganas de escribir líneas de código que controlen los posibles errores y que tengan un número ilimitados de mensajes. Te sugiero que por ahora te límites a programar la funcionalidad requerida para llegar con los tiempos al final.

Pronto llegará la semana en la que pondremos a punto el sistema, ahora recordá que:

- Cuando queremos pasar de un formulario a otro debemos instanciar un objeto de esa clase.
- Si hacemos referencia a un objeto que se encuentra en otro formulario del que estamos programando se antepone al objeto el nombre del formulario que lo contiene.
- Estamos comunicándonos con los datos de las base de datos con consultas simples mediante store procedure; esto nos permite separa el código de MySQL del código de C#.

A modo de resumen, te dejamos un video donde se muestra la ejecución de lo programado hasta esta semana.