

Universidad Nacional de Ingeniería

Facultad de Electrotecnia y Computación

Ingeniería en Computación

Algoritmización y estructuras de datos

• ~ RA. 8 ~ •

Alumno:

Marcel Enmanuel Díaz Largaespada (2020-1384U)

Docente:

Adilson González López

Fecha de entrega:

Jueves 02 de septiembre del 2021

Índice

1. Introducción.....	3
2. Diseño del <i>form</i>	4
3. Código Fuente.....	6
3.1. Declaraciones y métodos iniciales.	6
3.2. Métodos de interacción con la interfaz.	7
3.2.1. Formulario principal.....	7
3.2.2. Formulario Empleado.....	8
3.2.3. Formulario Solicitud	9
4. Ejecución del programa	10
5. Conclusiones	14

1. Introducción

Esta actividad práctica es distinta a las anteriores, puesto que aquí no se pondrán en práctica algoritmos de ordenamiento o estructuras de datos, como hemos estado viendo, sino que se explorarán las diferentes y potentes herramientas con las que cuenta el IDE de Visual Studio para la creación de formularios con interfaces mucho más vistosas.

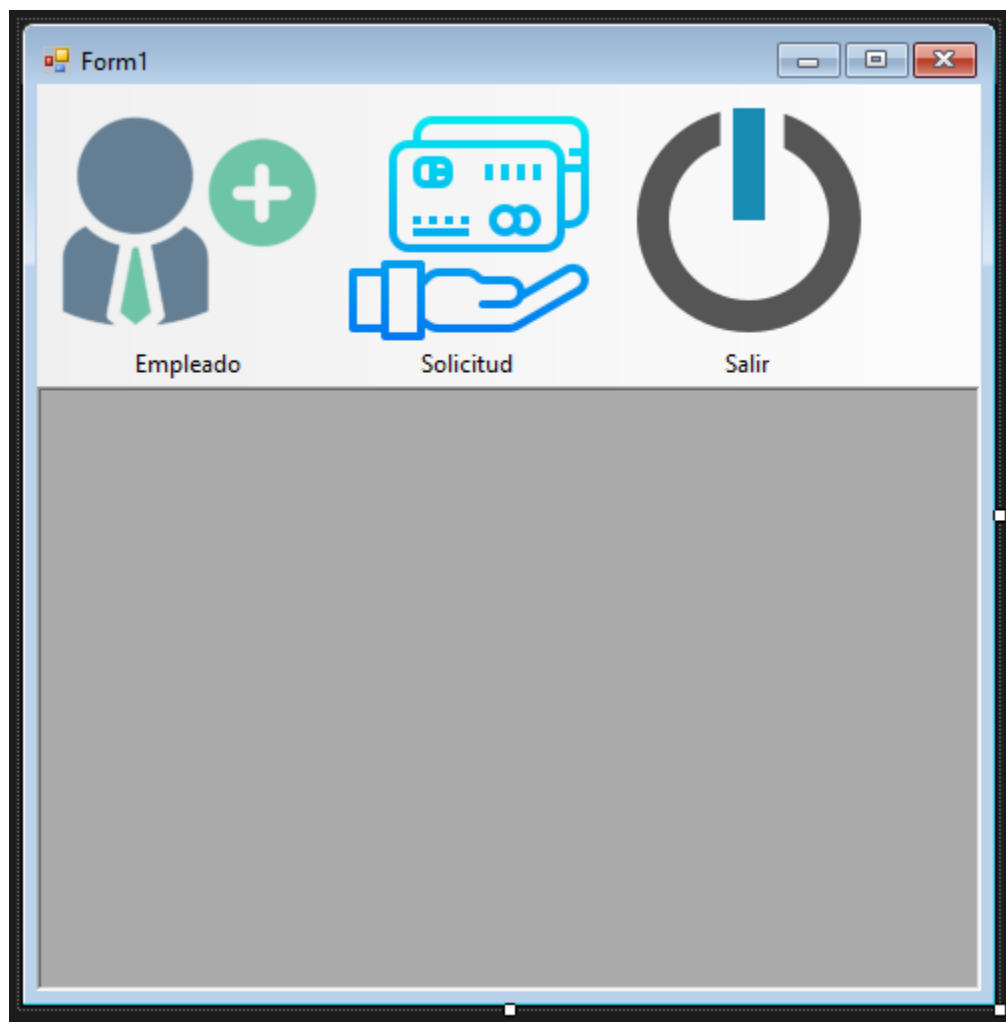
El proceso, por tanto y demás, fue muy entretenido e interesante. Adjuntas en este documento, están las capturas de pantalla del proceso de creación del formulario.

2. Diseño del *form*

La finalidad de este ejercicio es trabajar con más de un *form*: la interfaz principal, el formulario de empleado y el de solicitud.

A través de la interfaz principal es que se accederá a los *forms* mencionados haciendo click en sus botones correspondientes. Para ello, empleamos un **Menustrip**, al cual se le añadieron 3 elementos: empleado, solicitud y salir. En seguida, a cada uno se le asignó un icono proporcionado por el docente y se modificaron un par de propiedades, de tal forma:

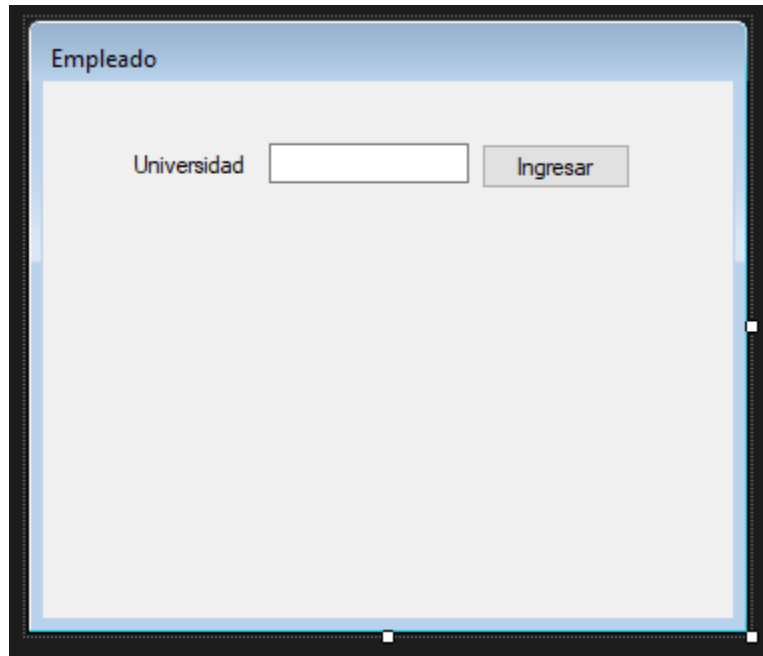
- **ImageScaling:** None.
- **TextImageRelation:** ImageAboveText.
- **Image:** para esta propiedad se procedió importando el icono correspondiente y seleccionándolo.



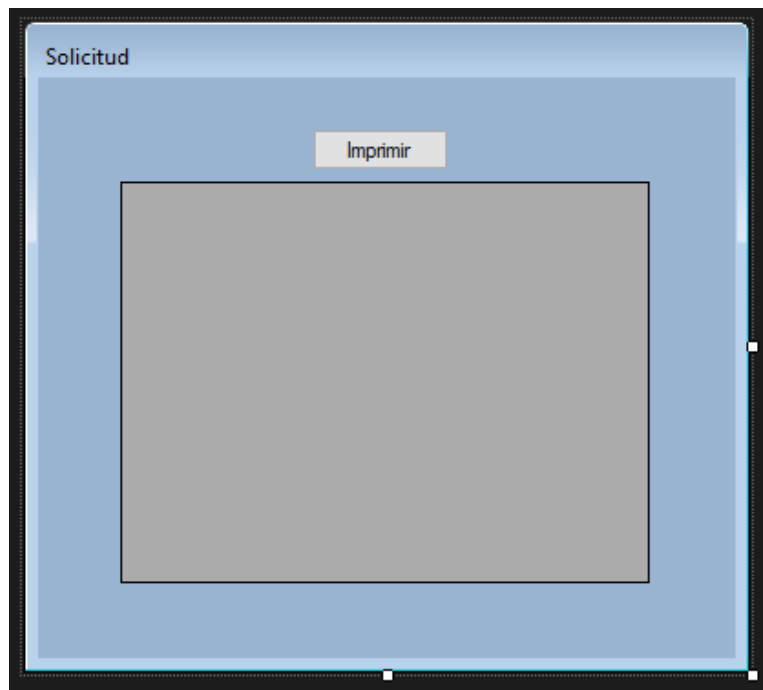
Interfaz principal

En los siguientes dos formularios, que corresponden a la interfaz de empleado y la de solicitud, se les modificaron dos propiedades:

- **StartPosition:** CenterScreen
- **ControlBox:** False. De esta forma, se le impide al usuario cerrar la ventana del *form* para impedir cualquier error.

The image shows a Windows-style form titled "Empleado". The form has a light blue header bar with the title. The main area is white. It contains a label "Universidad" in a standard font. To the right of the label is a rectangular text input field. Further to the right is a button labeled "Ingresar". The form has a standard Windows border with a title bar and a small maximize button on the right.

Formulario de Empleado

The image shows a Windows-style form titled "Solicitud". The form has a light blue header bar with the title. The main area is white. It contains a large, solid gray rectangular area that occupies most of the space. Above this area, centered, is a button labeled "Imprimir". The form has a standard Windows border with a title bar and a small maximize button on the right.

Formulario de solicitud

3. Código Fuente

3.1. Declaraciones y métodos iniciales.

Como estamos trabajando con C#, todos los elementos que lleguemos a manipular son objetos. Esto quiere decir que los *forms* que nosotros creamos también deberán de seguir el mismo tratamiento. De hecho, nosotros, cuando creamos un nuevo formulario, estamos instanciando un objeto de tipo **Form** que hereda sus propiedades y métodos. Por lo cual, para poder emplear a los formularios “*derivados*” del menú principal, primeramente, debemos de declararlos e instanciarlos dentro de dicho formulario:

```
Empleado Emp = new Empleado();  
Solicitud Soli = new Solicitud();
```

Paralelamente, se crea una clase pública **estática** –esto, con el fin de poder acceder a sus propiedades de forma global–, dentro de la cual declaramos e instanciamos una lista de cadenas:

```
3 referencias  
public static class Declaraciones  
{  
    public static List<String> lUniversidad = new List<string>();  
}
```

3.2. Métodos de interacción con la interfaz.

3.2.1. Formulario principal

Elemento Empleado del ToolStripMenu – clic: Se emparenta al formulario **Empleado** dentro del formulario principal. De este modo, el primero aparecerá encapsulado dentro del segundo. Entonces, se muestra y se esconde el resto de formularios.

```
1 referencia
private void empleadoToolStripMenuItem_Click(object sender, EventArgs e)
{
    Emp.MdiParent = this;
    Emp.Show();
    Soli.Hide();
}
```

Elemento Solicitud del ToolStripMenu – clic: Contiene exactamente la misma finalidad que la función anterior, pero para su elemento correspondiente.

```
1 referencia
private void solicitudToolStripMenuItem_Click(object sender, EventArgs e)
{
    Soli.MdiParent = this;
    Soli.Show();
    Emp.Hide();
}
```

Elemento Salir del ToolStripMenu – clic: Muestra un mensaje de confirmación. Si el usuario asiente, el programa termina.

```
1 referencia
private void salirToolStripMenuItem_Click(object sender, EventArgs e)
{
    DialogResult confirmacion = MessageBox.Show("¿Estás seguro que deseas salir?", "Salir", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);
    if (confirmacion == DialogResult.Yes)
        Close();
}
```

3.2.2. Formulario Empleado

Botón Ingresar – clic: Ingresa en la lista **lUniversidad** perteneciente a la instancia **Declaraciones** lo que esté escrito en la caja de texto **txtUniversidad**.

```
1 referencia
private void btnIngresar_Click(object sender, EventArgs e)
{
    Declaraciones.lUniversidad.Add(txtUniversidad.Text);
    txtUniversidad.Clear();
}
```

Caja de texto txtUniversidad – al presionar una tecla: Si la tecla presionada es *enter*, entonces efectúa lo mismo que la función anterior.

```
1 referencia
private void txtUniversidad_KeyPress(object sender, KeyPressEventArgs e)
{
    if (e.KeyChar == 13)
    {
        Declaraciones.lUniversidad.Add(txtUniversidad.Text);
        txtUniversidad.Clear();
    }
}
```

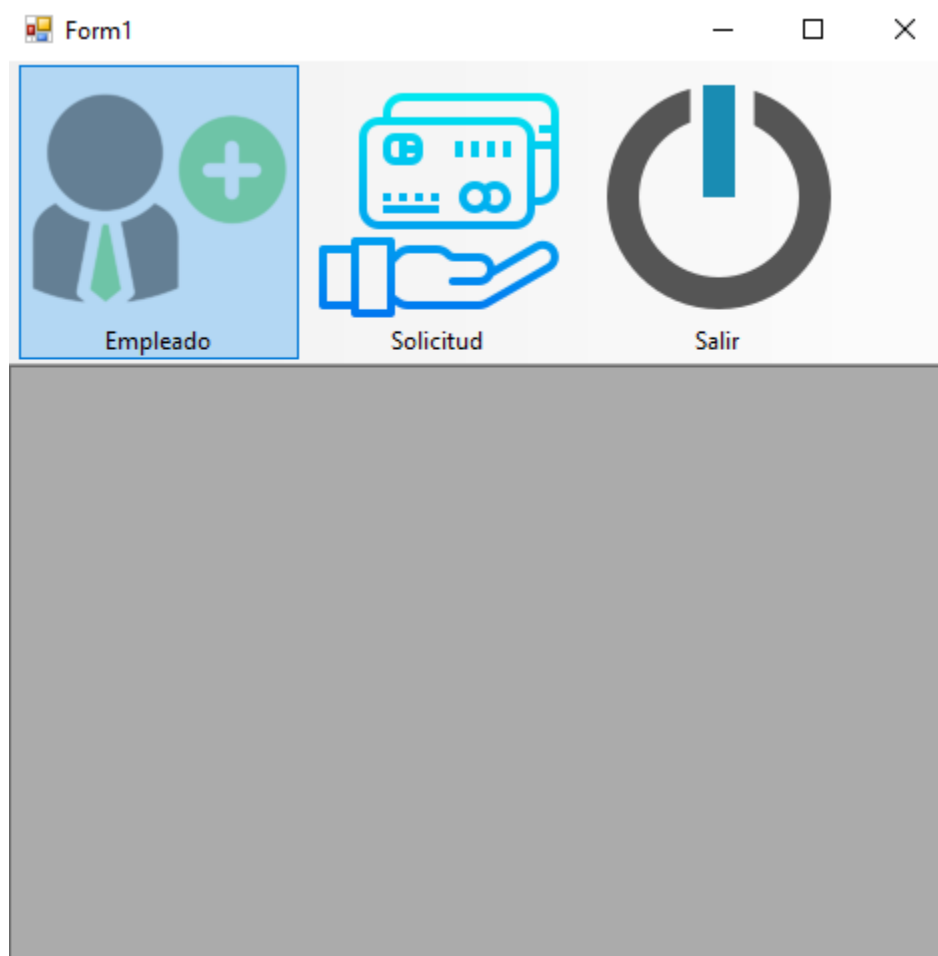

3.2.3. Formulario Solicitud

Botón Imprimir – clic: Crea una columna e imprime en las celdas los datos que contiene la lista dentro de la instancia **Declaraciones**:

```
1 referencia
private void btnImprimir_Click(object sender, EventArgs e)
{
    int i = 0;
    dgImprimir.Columns.Add("", "Universidades");
    foreach(var item in Declaraciones.lUniversidad)
    {
        dgImprimir.Rows.Add();
        dgImprimir.Rows[i].Cells[0].Value = item;
        i++;
    }
}
```

4. Ejecución del programa

Finalmente, el resultado de nuestro laboratorio arrojó la siguiente interfaz:



Inicio del programa¹

¹ El programa inicia en modo pantalla completa, pero se redujo a tamaño de ventana para fines ilustrativos. Ante el escrutinio metódico, es buena la mención.

Form1

Empleado

Solicitud

Salir

Empleado

Universidad

Ingresando datos en el formulario empleado

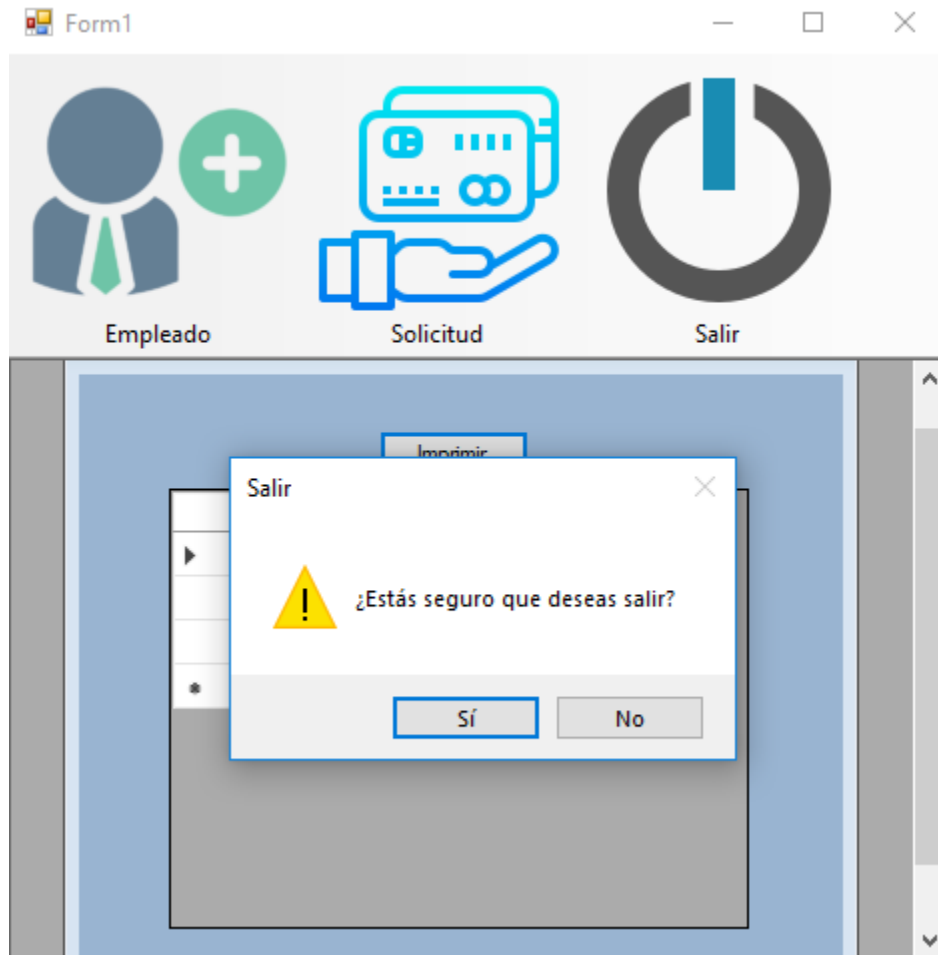
Form1

Empleado Solicitud Salir

Imprimir

	Universidades
▶	UNI
	UCA
	UNAM
*	

Imprimiendo los datos dentro del formulario Solicitud



Finalizando el programa

5. Conclusiones

Fue grandemente agradable haber experimentado con funcionalidades anteriormente desconocidas del IDE de Visual Studio. Sin duda, estamos trabajando en un entorno de desarrollo bastante completo con el cual podemos comenzar a diseñar programas mucho más complejos y exigentes; este, en mi opinión, es un muy buen paso para alcanzar dicha meta.