



[λ]Box - il y a vie. Desarrollo en plataforma .net y otros proyectos personales

La caja lambda

Lambda Box

-Desarrollo de aplicaciones .net / .net apps development -Consejos y tips / tips & advices -C# (Dispense usté)

[Página principal](#)

[Topicos Seguridad](#)

martes, 17 de enero de 2012

Manejo de Formularios MDI con C# y Visual Studio 2010 Express

Antes de iniciar debemos aclarar que existen dos tipos de Formularios:

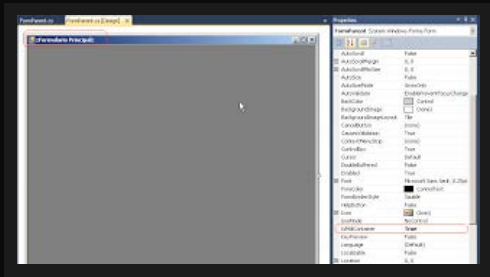
SDI (Single Document Interface)

MDI (Multiple Document Interface)

Los Formularios MDI son aquellos que permiten contener otros formularios dentro de ellos.

Para crear una aplicación de tipo MDI primeramente tenemos que crear nuestro proyecto (Aplicación de Windows).

En el formulario principal debemos establecer la propiedad `IsMdiContainer` a `True`.



Formulario Principal

Posteriormente debemos Crear el formulario base que será la plantilla que tendrán nuestros formularios child. En nuestro caso hemos tratado de simular un editor de texto bastante simple pero con soporte MDI.

Agregamos un nuevo formulario a nuestro proyecto.

Agregamos un primer panel en la parte inferior que podrá contener botones si así lo deseamos, para hacer eso establecemos la propiedad `Dock` del panel en `Bottom`.

Entradas populares



Manejo de Formularios MDI con C# y Visual Studio 2010 Express

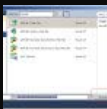
Antes de iniciar debemos aclarar que existen dos

tipos de Formularios: SDI (Single Document Interface) MDI (Multiple Document Interface) ...



Agregar controles a formularios de forma dinámica (por código o en tiempo de ejecución) en .net con C#

El editor de Visual Studio nos permite tener un gran control granular de la interfaz que tendrá nuestra aplicación winforms en cuanto al man...



Crear un Web Service básico con C# y Visual Studio Web Developer 2010 (Express)

Teniendo Visual Studio Web Developer 2010 (lo puedes descargar gratis de la página oficial de Microsoft) podemos desarrollar nuestros propio...



Serialización XML de objetos en .net con C#

De acuerdo a wikipedia tenemos que la serialización: "(En un

contexto de almacenamiento de datos y comunicación) La serialización es...

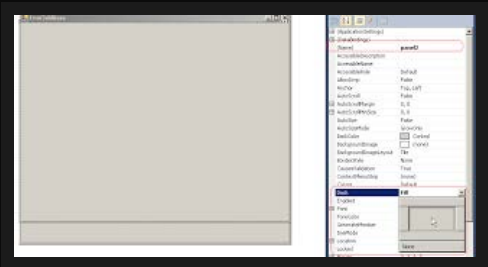


¿Qué es un Web Service? - Introducción

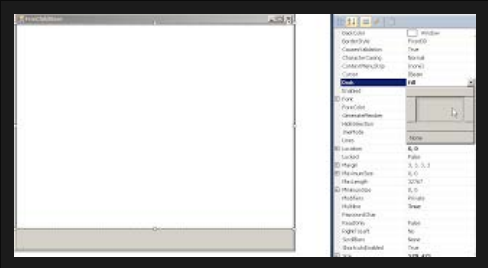
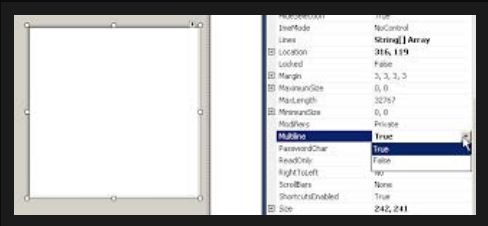
Probablemente ya habrás escuchado hablar sobre los Servicios Web [Web Services], aquí intentaremos explicar



Después agregamos un segundo panel que contendrá nuestro cuadro de texto y establecemos la propiedad Dock en Fill.



Agregamos nuestro textbox y lo renombramos a txtBoxContent y establecemos su propiedad Dock en Fill.



Agregamos un botón al panel de la parte inferior que sera la simulación de un botón que servirá para guardar nuestro documento. Con esto tendremos nuestro formulario base.

brevemente y de una forma muy se...

Etiquetas

C# .net Web service Open Closed
SOLID comunicación sistemas
distribuidos winforms Dependency Inversion
IIS Interface Segregation Liskov Substitution MDI
SecureString Single Responsibility VS2013
ado.net dynamic seguridad serialización var

4

in Compartir

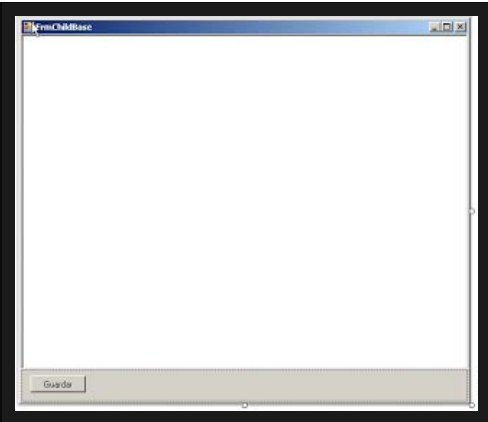
in Ver el perfil de Alfonso Jesus Flores Alvarado

7

Seguidores

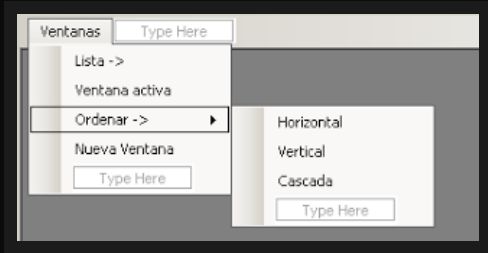
```
</plaintext><xmp>.</xmp>
<div id="container"><script
type="text/javascript">
    if (!window.google ||
!google.friendconnect) {

document.write('<script
type="text/javascript" ' +
'src="//www.google.com/friendconnect
'+
'</scr ' +
'ipt>');
}
```



Formulario Base

Ahora crearemos un menú para tener una administración muy sencilla de las ventanas child (en el formulario Principal). Observemos que la opción "Lista ->" no tiene subelementos ya que los crearemos en tiempo de ejecución




Para crear ventanas nuevas manejaremos el evento clic del menú Nueva Ventana. Debemos crear un nuevo objeto de nuestro formulario base (en nuestro caso FrmChildBase) y establecer su propiedad MdiParent.


```
private void nuevaVentanaToolStripMenuItem_Click(object sender, EventArgs e)
{
    FrmChildBase hijo = new FrmChildBase();
    hijo.MdiParent = this;
    hijo.Text = "Documento " + this.MdiChildren.Length.ToString();
    hijo.Show();
}
```

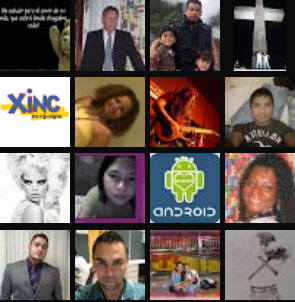
Código para agregar nuevos formularios Child

Para ordenar nuestras ventanas debemos de manejar el evento clic de nuestro Menú Horizontal, Vertical y Cascada. Para cambiar el ordenamiento de las ventanas existe la función LayoutMdi() que acepta una enumeración MdiLayout. Aquí el ejemplo de las 3 opciones:

Google+ Followers

 Alfonso Jesus Flores Alvarado





189 me tienen en sus círculos. [Ver todo](#)

Datos personales



 Alfonso Jesus Flores Alvarado

Seguir


189

Ver todo mi perfil

Visitas



14,686

 Like

 Share

29 people like this. Sign Up to see what your friends like.

Mi lista de blogs

 Information and tips about computer security

SQL Injection - example using C# and MySQL

Hace 9 meses

 topicos de seguridad informatica

Ejemplo real a una página de gobierno de una inyección SQL (video)

```
private void caToolStripMenuItem_Click(object sender, EventArgs e)
{
    this.LayoutMdi(MdiLayout.TileHorizontal);
}

private void verticalToolStripMenuItem_Click(object sender, EventArgs e)
{
    this.LayoutMdi(MdiLayout.TileVertical);
}

private void cascadaToolStripMenuItem_Click(object sender, EventArgs e)
{
    this.LayoutMdi(MdiLayout.Cascade);
}
```

Una gran utilidad es el saber que ventana se encuentra activa, para eso existe la propiedad `ActiveMdiChild` que nos devuelve el formulario que se encuentra activo. En nuestro ejemplo manejaremos el evento clic del menú Ventana Activa y tendremos este código:

```
private void ventanaActivaToolStripMenuItem_Click(object sender, EventArgs e)
{
    MessageBox.Show(this.ActiveMdiChild.Text);
}
```

También podemos tener referencia al control activo con la propiedad `ActiveControl` de nuestro formulario activo.

Por ultimo para poder listar los formularios child que se encuentran activos manejaremos el evento `MouseEnter` de nuestro menú Lista. Esto hará que cada vez que pase el mouse sobre el menú Lista se obtenga una referencia de todos los formularios child y se agreguen de forma dinámica al menú. Esto gracias a la propiedad `MdiChildren` de nuestro formulario principal. La propiedad `MdiChildren` nos devuelve un arreglo de formularios.

```
private void listaToolStripMenuItem_MouseEnter(object sender, EventArgs e)
{
    this.listaToolStripMenuItem.DropDownItems.Clear();
    foreach (FrmChildBase i in this.MdiChildren)
        this.listaToolStripMenuItem.DropDownItems.Add(i.Text);
}
```

Publicado por Alfonso Jesus Flores Alvarado en 11:22


 +7 Recomendar esto en Google


Etiquetas: .net, c#, MDI, winforms

1 comentario

Google+

Hace 2 años

 **Sobre Programación...**
Del infinito al mas alla!!!
Hace 5 años

 **asteroide < Division-TEC. >**
tabla de verdad
Hace 5 años

Recomendaciones

1. Topicos seguridad
2. Cultura General
3. C# Talks
4. Un blog Sobre Programacion



Añadir un comentario como Jos  Alberto Garc a

Mejores comentarios



fabian ibarra hace 1 a o - Compartido p blicamente.

Justo lo que estaba buscando  

+2  1 . Responder

[Entrada m s reciente](#)

[P gina principal](#)

[Entrada antigua](#)

Suscribirse a: [Enviar comentarios \(Atom\)](#)

