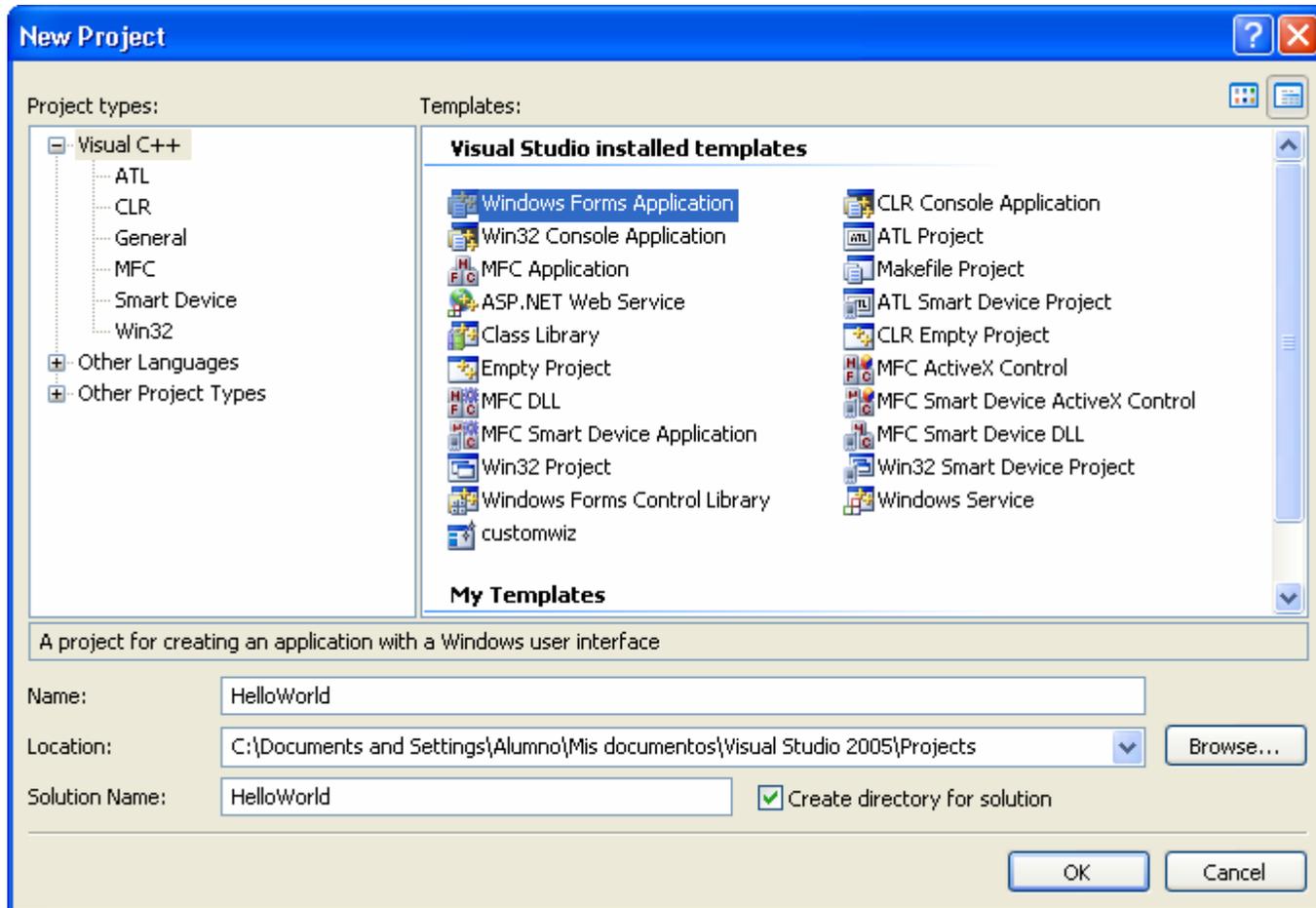


# **Interfaces gráficas con Visual Studio**

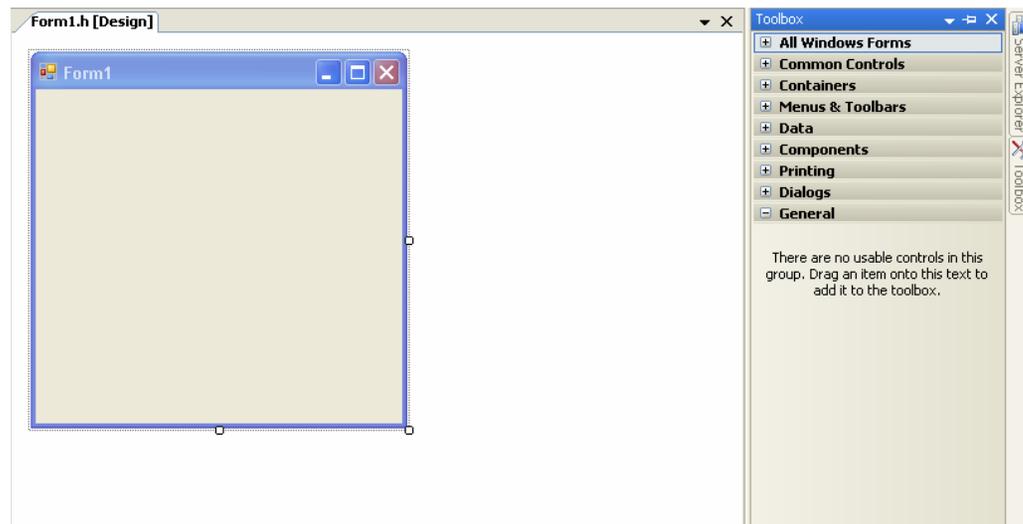
# Creación del proyecto

- Archivo->Nuevo->Proyecto->Aplicación de Windows Form



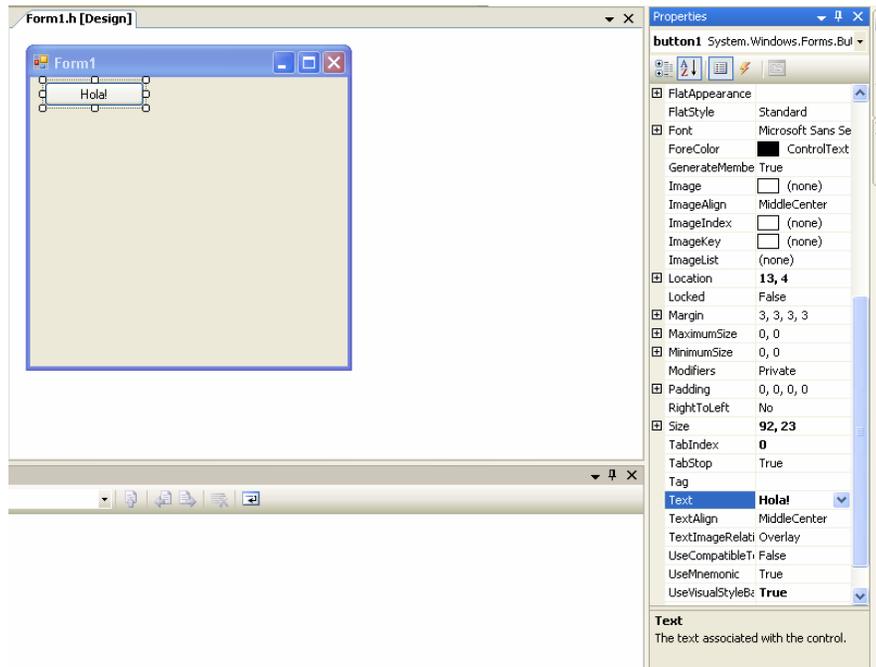
# ToolBox

- Cuando creamos un nuevo proyecto se crea por defecto un Form sobre el que podemos trabajar.
- Para insertar herramientas usaremos el ToolBox.



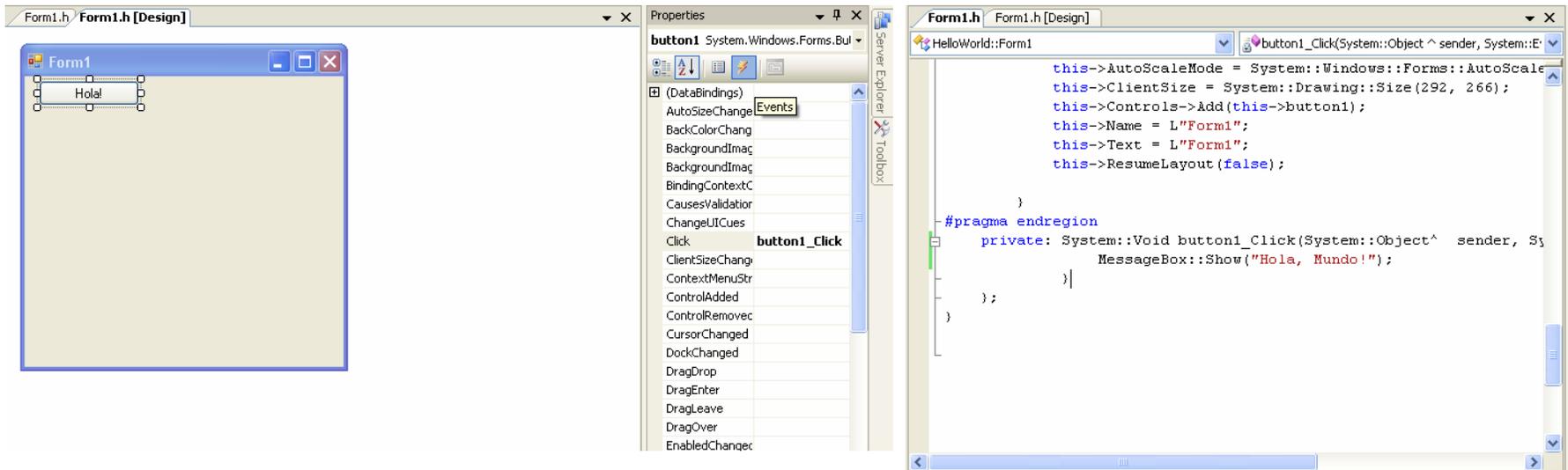
# Incluir elementos

- Para insertar un elemento de la toolBox lo seleccionamos y lo arrastramos al lugar del Form que queremos.
- Seleccionando el elemento en el Form podemos modificar las propiedades de este.



# Eventos

- Seleccionando en las propiedades del elemento también podemos acceder a los eventos que involucran a este.
- Haciendo doble click sobre alguno de ellos nos creara una función que se ejecutara el código que nosotros proporcionemos cuando el evento suceda.



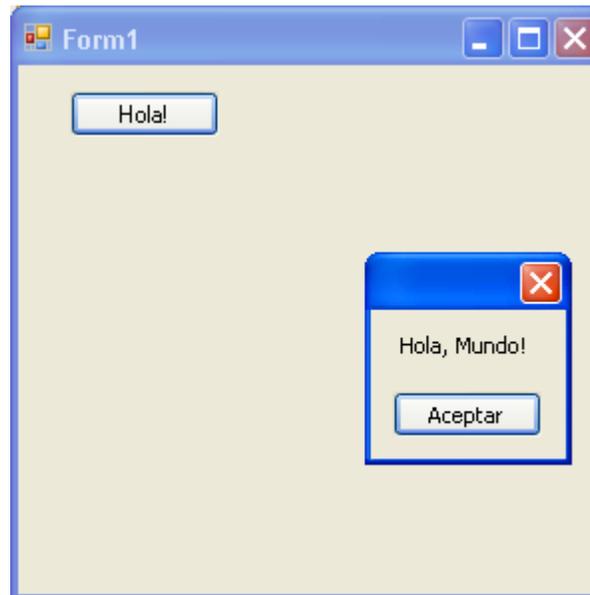
The image displays three windows from the Visual Studio IDE illustrating the process of handling a button click event.

- Form Design:** The leftmost window shows a Windows Form titled "Form1" with a single button labeled "Hola!".
- Properties Window:** The middle window shows the Properties window for the selected "button1". The "Events" section is expanded, and the "Click" event is assigned to the "button1\_Click" handler.
- Code Editor:** The rightmost window shows the code for the "button1\_Click" event handler. The code is as follows:

```
private void button1_Click(System::Object ^ sender, System::EventArgs e) {  
    this->AutoSizeMode = System::Windows::Forms::AutoSizeModeNone;  
    this->ClientSize = System::Drawing::Size(292, 266);  
    this->Controls->Add(this->button1);  
    this->Name = L"Form1";  
    this->Text = L"Form1";  
    this->ResumeLayout(false);  
}  
  
#pragma endregion  
private: System::Void button1_Click(System::Object ^ sender, System::EventArgs e) {  
    MessageBox::Show("Hola, Mundo!");  
};
```

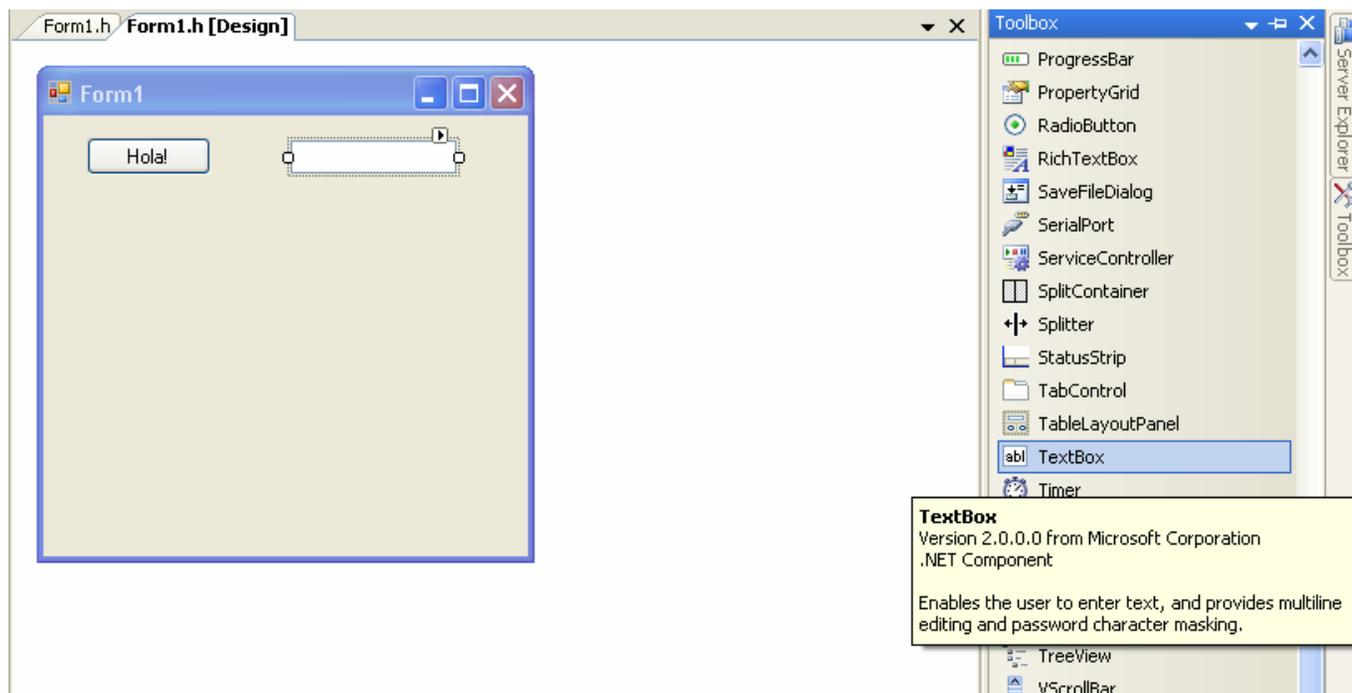
# Resultado

- Cuando ejecutamos el programa y hacemos click en el botón el resultado será el siguiente.



# TextBox(1)

- Para ingresar datos al sistema una buena opción es utilizar un TextBox.



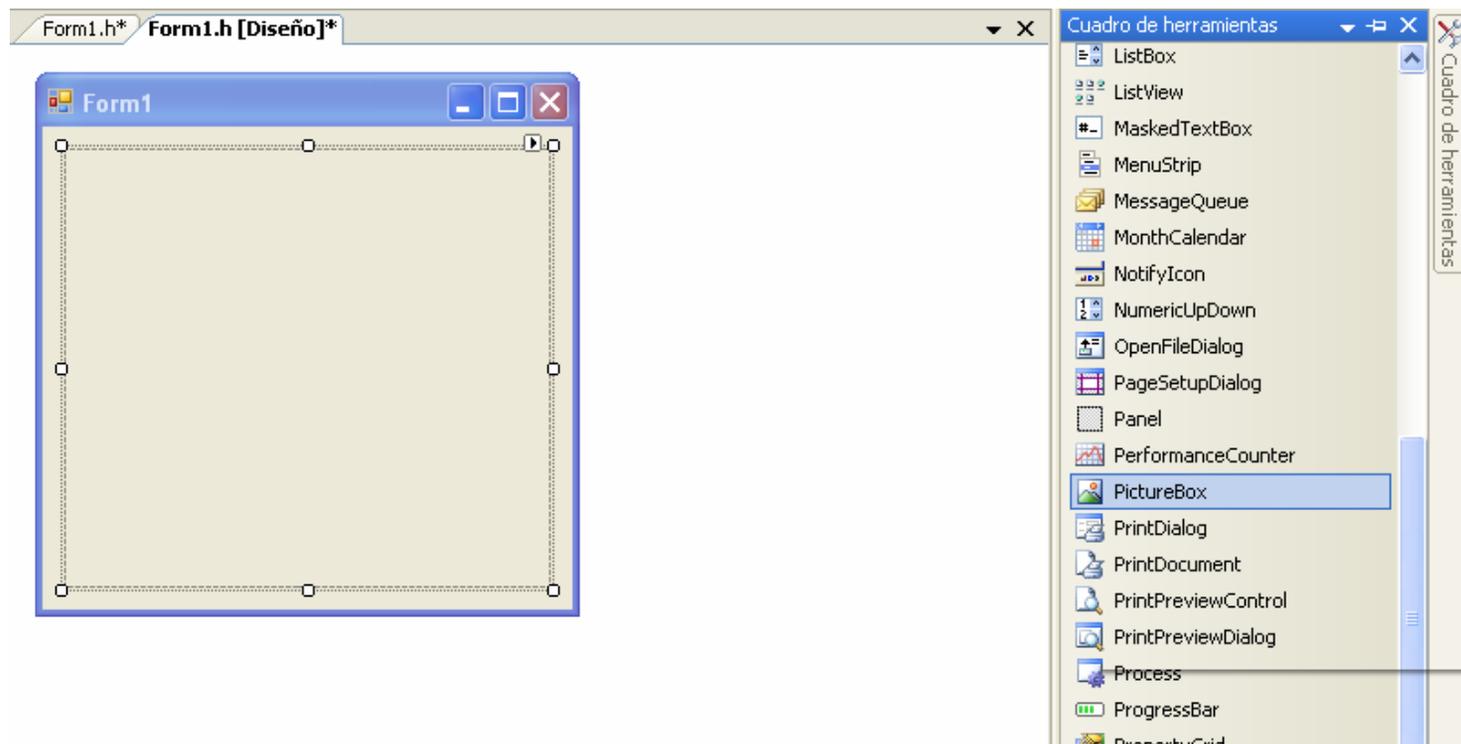
# TextBox(2)

- Para obtener los datos ingresados en el TextBox puede utilizarse la siguiente función `textBox1->Text`.
- De esta forma modificando la función `button1_Click` a `MessageBox::Show(textBox1->Text)` obtendremos:



# Dibujo de figuras (1)

- Para dibujar sobre el form podemos hacer uso de los PictureBox.
- La inserción de este es similar al resto de los elementos del Toolbox. (Por simplicidad en la figura se ve el ejemplo sobre un nuevo proyecto)



# Dibujo de figuras (2)

- Una posible forma de dibujar sobre el PictureBox es incluyendo funcionalidad en el evento Paint de este y el Load del form.

```
private: System::Void Form1_Load(System::Object^ sender, System::EventArgs^ e) {
    pictureBox1 = gcnew PictureBox;
    pictureBox1->Dock = DockStyle::Fill; // Asocia el PictureBox al form.
    // Conecta el Paint event del PictureBox al metodo event handler.
    pictureBox1->Paint += gcnew System::Windows::Forms::PaintEventHandler( this, &Form1::pictureBox1_Paint );
    this->Controls->Add( pictureBox1 ); // Agrega el control del PictureBox al Form.
}
private: System::Void pictureBox1_Paint(System::Object^ sender, System::Windows::Forms::PaintEventArgs^ e) {
    Graphics^ g = e->Graphics; // Crea una version local del graphics object para el PictureBox.
    // Agrega un texto al PictureBox.
    g->DrawString( "Texto", gcnew System::Drawing::Font( "Arial",10 ), System::Drawing::Brushes::Blue, Point(50,30) );
    // dibuja una linea en el PictureBox.
    g->DrawLine( System::Drawing::Pens::Red, pictureBox1->Left, pictureBox1->Top, pictureBox1->Right, pictureBox1->Bottom);
    g->DrawEllipse(System::Drawing::Pens::Blue, 20, 50, 40, 40 ); // dibuja un circulo en el PictureBox.
}
```

# Dibujo de figuras (3)

- Finalmente luego de escribir el código anterior obtenemos:

