Aplicaciones con interfaz gráfica

Un inicio rápido utilizando wxWidgets y C++ con code::blocks – parte 2

Una vez que instalamos las wxWidgets podemos comenzar a desarrollar aplicaciones con interface gráfica. En la parte 1 avanzamos hasta tener la aplicación que crea el asistente. Algo como la figura siguiente:

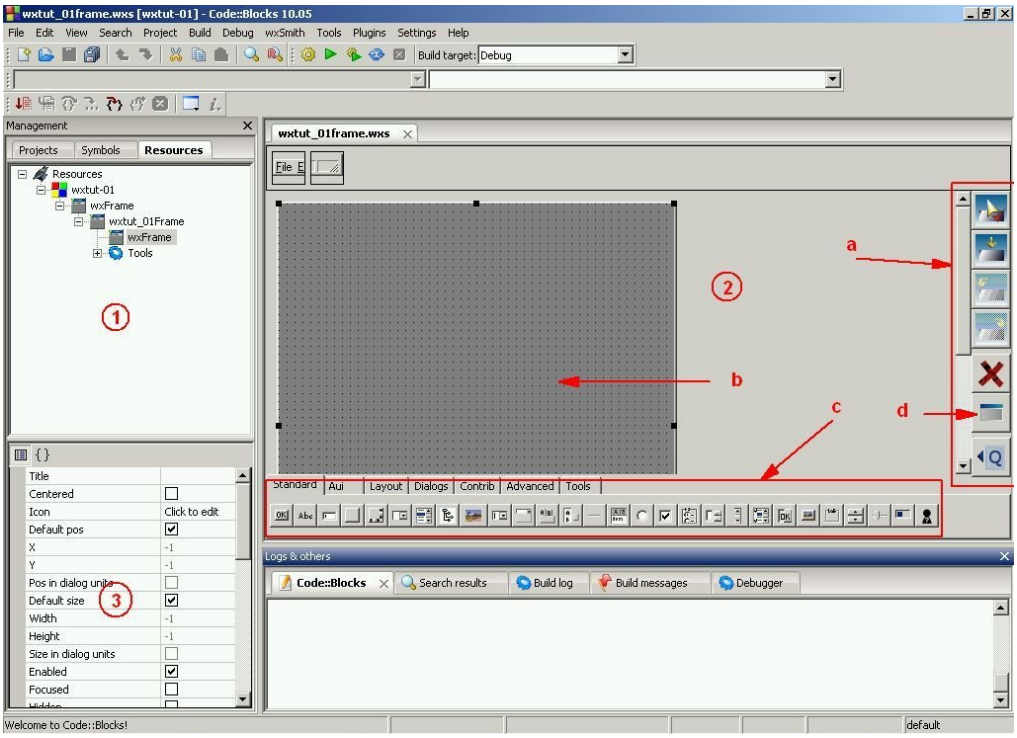

fig. 1 – el code::blocks al finalizar el asistente para proyecto usando wxWidgets

Vamos a desarrollar una aplicación sencilla que sirva como introducción a los controles más básicos y sus métodos y propiedades. La idea es desarrollar un programa que muestre el uso básico de botones, cuadro de texto y etiqueta.

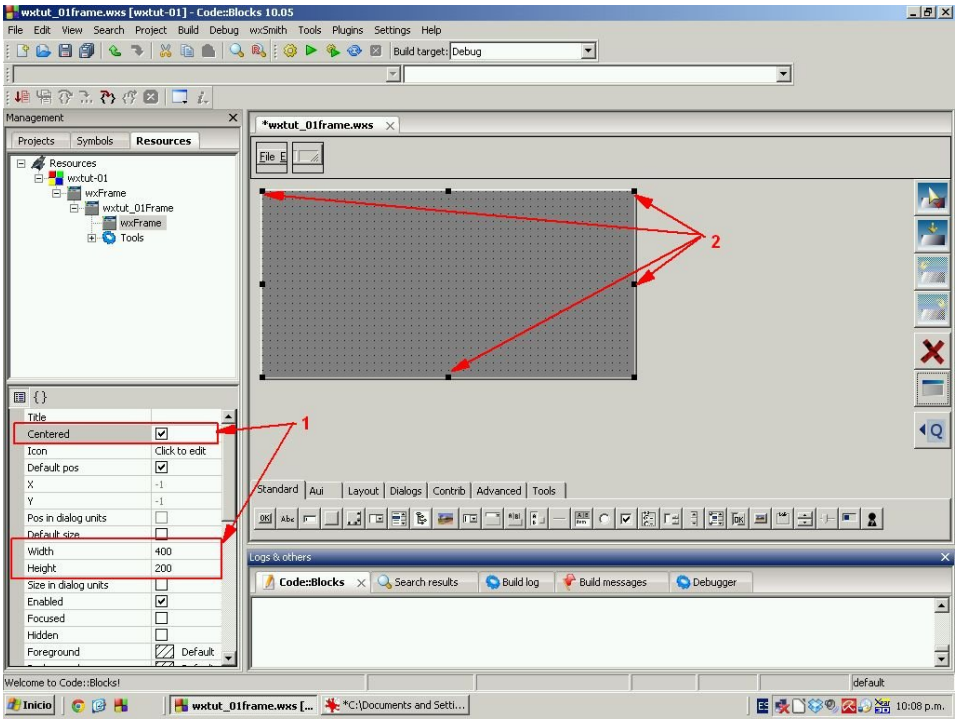

Fig. 2 – Fijando tamaño de nuestra ventana

Si queremos fijar el tamaño de la ventana, lo podemos hacer arrastrando los cuadros (1) que aparecen alrededor del wxFrame o desde la ventana de propiedades (2)

Continuamos agregando un control wxPanel que hará de contenedor; seleccionamos el control (ver fig. 3) y arrastramos hasta el wxFrame.

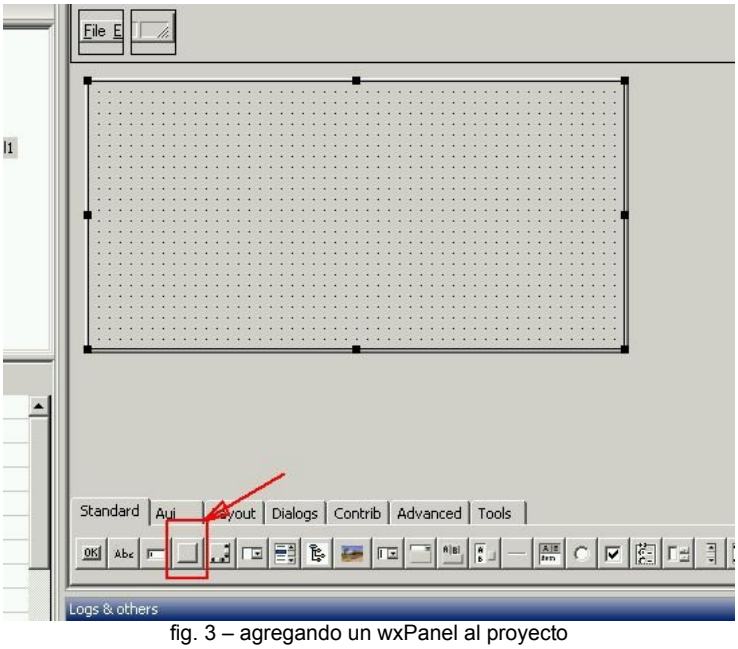

fig. 3 – agregando un wxPanel al proyecto

Agregamos dos controles más: un botón (wxButton) y una etiqueta (wxStaticText) como muestra la figura 4

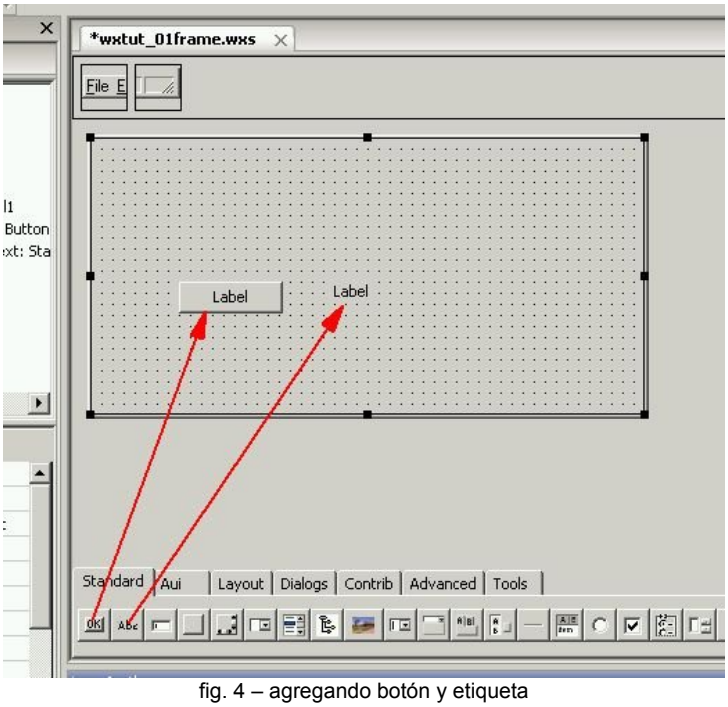

fig. 4 – agregando botón y etiqueta

Podemos cambiar algunas propiedades desde el entorno de desarrollo (ver zona 3 en fig. 1) Por ejemplo, para cambiar el texto que aparece en el botón cambiamos su propiedad “Label” (fig. 5)

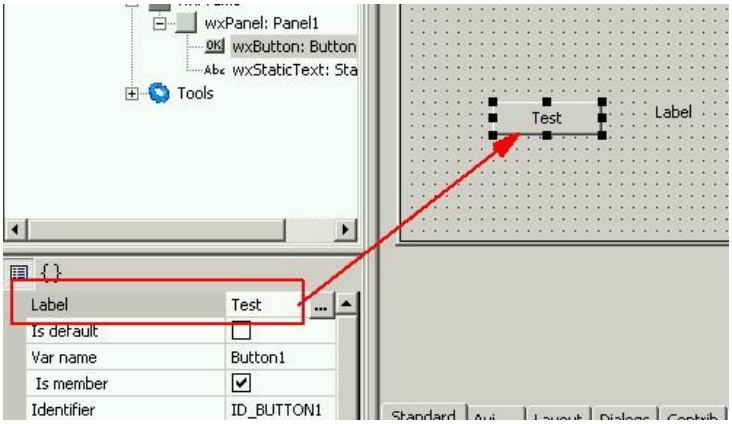

fig. 5 – cambiando el texto de un botón

Supongamos que al hacer click sobre el botón se desea que en la etiqueta aparezca un texto... Tenemos que asignar un conjunto de acciones (función) a un evento. El evento por default del botón es click, entonces para agregarle código hacemos doble click sobre el botón y aparece una ventana de edición de código...

```
109 }
110
111 void wxej2Frame::OnButton1Click(wxCommandEvent& event)
112 {
113 }
114
```

fig. 6 – se crea una función para reaccionar a un evento

Aquí escribimos el código que queremos que se ejecute al presionar el botón. Si queremos que el textro de la etiqueta sea por ejemplo “2011.11.28” escribimos:

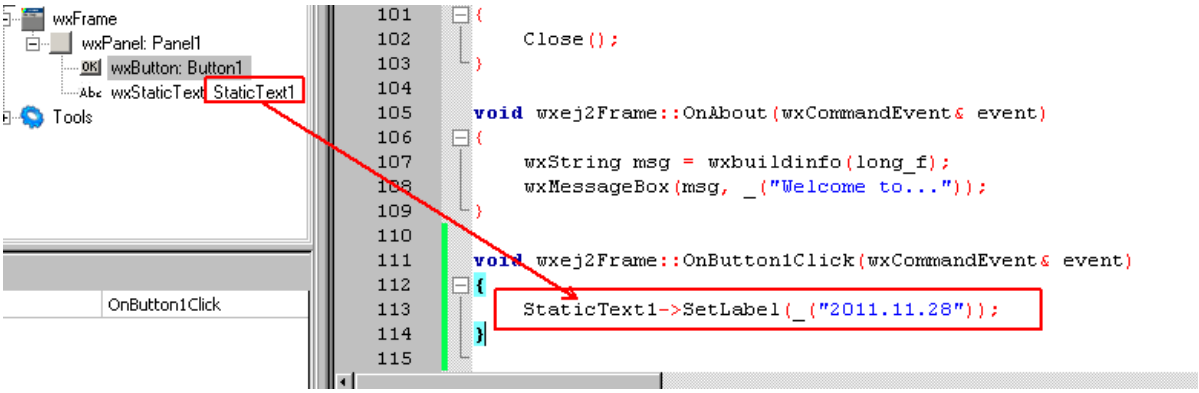

fig. 7 – modificar lo que muestra la etiqueta.

El texto que muestra la etiqueta se puede especificar con el método SetLabel. Como parámetro le pasamos un wxString (El tipo string de wxWidgets). Un wxString se indica entre _(y)

En la fig. 8 vemos cómo queda el proyecto. En (1) el código que acabamos de tipear; en (2) vemos en la pestaña “Resources” la estructura de la ventana: StaticText1 y Button1 están contenidos en Panel1, que está dentro de un wxFrame.

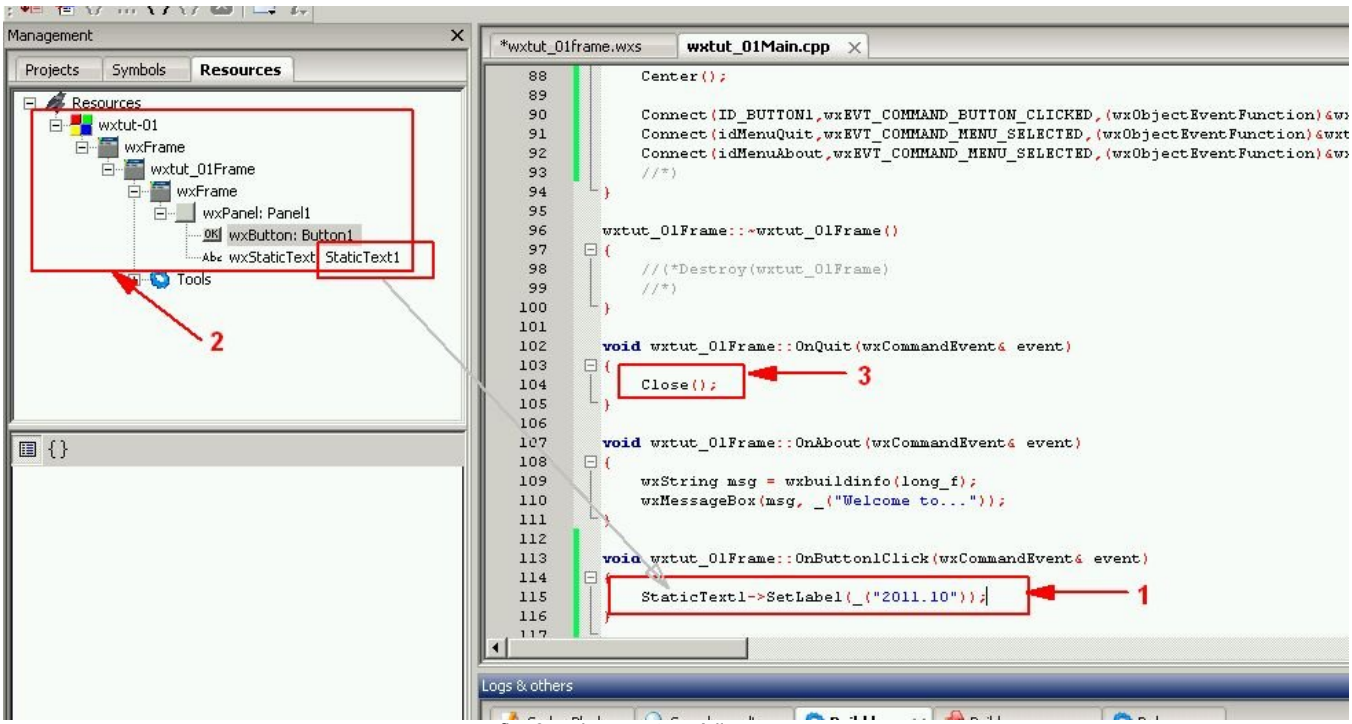

fig. 8 – el proyecto: código y recursos

En (3) vemos una función generada automáticamente para cerrar la ventana: Close()

Si queremos que el usuario pueda introducir un texto el StaticText no sirve. Tenemos que usar un cuadro de texto o TextCtrl. Agregamos dos cuadros de texto y dos botones (a los que les cambiamos el texto) y obtenemos algo como la fig. 9

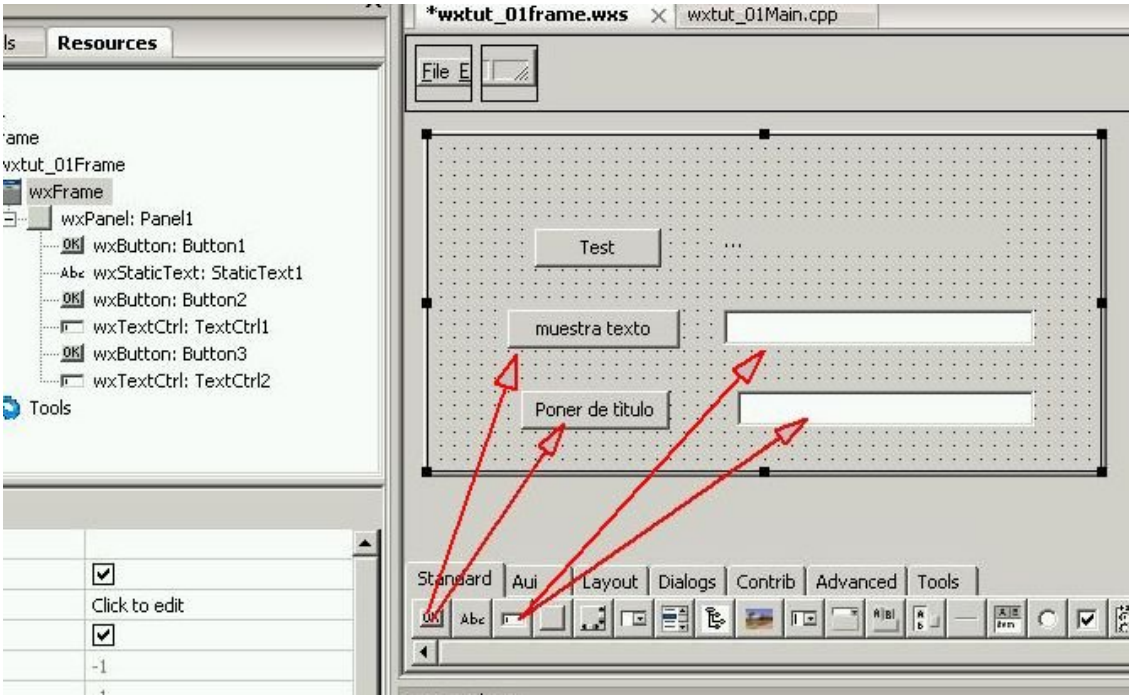

fig. 9 – dos botones más, y dos cuadros de texto.

Vamos a hacer que al hacer click en el botón Button2 (“Muestra texto”) se muestre un messagebox con el texto ingresado en el cuadro de texto. Hacemos doble click sobre este botón e introoducimos el siguiente código:

```
122
123  void wxtut_01Frame::OnButton2Click(wxCommandEvent& event)
124  {
125      wxString s;
126
127      s = TextCtrl1->GetValue();
128
129      wxMessageBox(s, _("El texto..."), wxYES_NO | wxICON_INFORMATION);
130  }
131
```

fig. 10 – Código para el evento “Click” del segundo botón.

Declaramos una variable s de tipo wxString. La usamos para guardar el contenido del cuadro de texto. Para obtener el texto contenido en un cuadro de texto usamos el método GetValue() (ver línea 127 de fig. 10)

Luego mostramos un messagebox donde el primer parámetro es el contenido de la ventana (s en este caso); el segundo parámetro es el título de la ventana y el tercero es un conjunto de bits que indican los botones que mostrará el messagebox y el gráfico que presentará al lado del contenido.

Para finalizar, vamos a hacer que al presionar el tercer botón (“Poner de título”) se muestre otro messagebox que pida confirmar la acción, que será poner como título de la ventana el texto ingresado en un cuadro de texto. Para esto hacemos doble click sobre el tercer botón y agegamos código como muestra la fig. 11

```
void wxtut_01Frame::OnButton3Click(wxCommandEvent& event)
{
    wxString s;

    s = TextCtrl2->GetValue();

    if (wxMessageBox(s, _("Confirma?"), wxYES_NO | wxICON_QUESTION) == wxYES)
        this->SetLabel(s);
}
```

fig. 11 – Cambiamos el título de la ventana

Este código ejemplifica otro uso de los messagebox: devolver un valor según el botón presionado. Además, muestra cómo acceder al título de la ventana (SetLabel)