

LABORATORIO DE PROGRAMACIÓN II

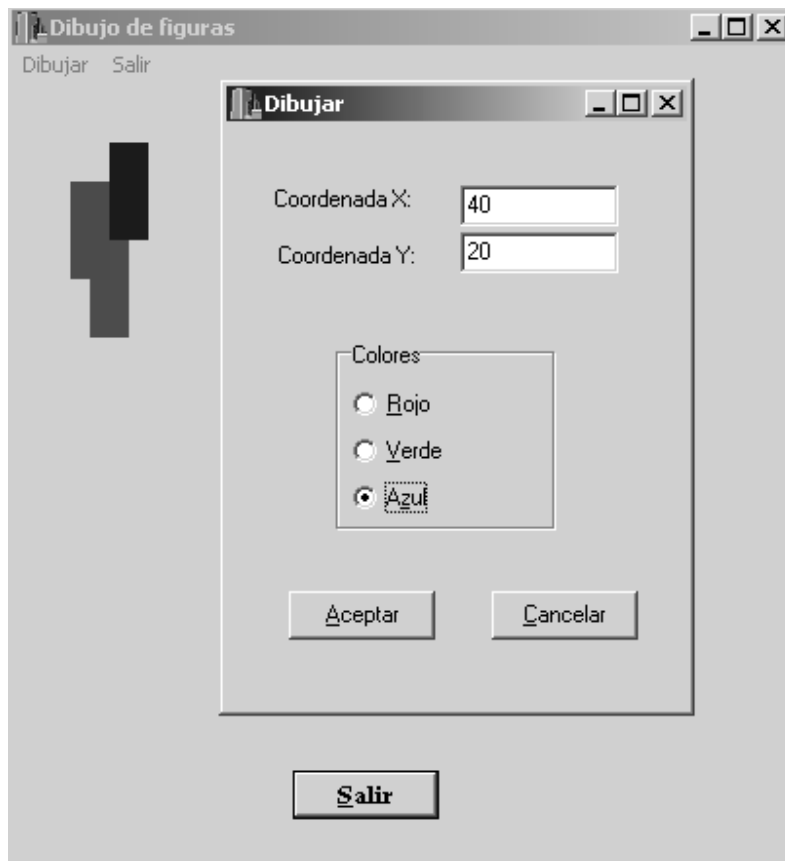
Facultad de Informática

Curso 2003-2004

PRÁCTICA 0

FECHA LÍMITE: esta práctica no necesita ser corregida

El objetivo de esta práctica es introducir el entorno C++Builder. El objetivo de la práctica no es producir una aplicación útil y acabada, sino sólo mostrar algunos pasos comunes para el desarrollo del resto de las prácticas. Vamos a hacer un programa con dos formularios: el formulario principal que incluye un menú con dos opciones (*Dibujar* y *Salir*) y un formulario secundario que se abre al seleccionar la opción *Dibujar* del menú. En este formulario secundario vamos a poder introducir dos valores numéricos (x,y) y seleccionar un color. Al finalizar el formulario secundario la aplicación dibujará sobre el formulario principal rectángulo de 20×50 *pixels* en las coordenadas y el color indicados.



PARTE I: Diseño del formulario principal

Pasos a seguir:

1. Entra en tu cuenta de usuario y crea un directorio de trabajo para la asignatura y un subdirectorío para la práctica (Practica0). Lanzar *C++Builder* y observa los distintos componentes que aparecen en el entorno gráfico: *editor gráfico*, *paleta de componentes visuales* (VCL), *inspector de objetos*, *editor de código*, etc. Pulsando *F11* podemos movernos al *inspector de objetos* y al *editor de código*. El código que vemos con nombre *Uni1.cpp* corresponde al formulario. Para ver su fichero *.h* podemos pulsar *Ctrl.+F6*.
2. Salvar el proyecto con *Save Project As* o *Save All* del menú *File*, eligiendo el directorio de la práctica y los nombres *UFormPrincipal.cpp* y *ProyPract0.bpr*, para los archivos del formulario (*Unit1.cpp*) y del proyecto (*Project1.cpp*) respectivamente.
3. Observa el directorio *Practica0*: los archivos que deben salvarse en el disquete para poder recuperar el proyecto son todos los que tengan las extensiones: *.bpr*, *.res*, *.cpp*, *.h* y *.dfm*.
4. Utilizando el *inspector de objetos* del *Builder* modificar la propiedad del formulario *Name* a *FormPrincipal* y *Caption* a *Dibujo de figuras*. Siempre que se añada una componente se debe poner como *Name* algún nombre nemotécnico. Este nombre aparecerá automáticamente en los métodos asociados a la componente y hay que usarlo para acceder a sus propiedades (el *Caption* sólo es visual). En el código este nombre será el que reciba la variable que apunta al objeto (en *Builder* siempre se utilizan punteros) y que *Builder* declara automáticamente. También en el inspector de objetos modificar las propiedades *ClientHeight* y *ClientWidth* dándoles valores 400 y 400 respectivamente para fijar las dimensiones del formulario.
5. Añadir al código del formulario principal un método *salir* con el siguiente código:

```
void TFormPrincipal::salir() {  
if (MessageDlg("Seguro?",mtConfirmation,TMsgDlgButtons() << mbYes << mbNo,0)==mrYes)  
    Close();  
}
```

No olvidar incluir en la sección *private* del *.h* la cabecera `void salir();`

6. Añadir al formulario un botón (componente *TButton* de la paleta *Standard*) para Salir (pon el *Caption* en negrilla y una fuente más grande). Para darle funcionalidad define el evento *OnClick*, de forma que invoque al método *salir()*. Hay dos formas de hacer esto:
 - Hacer doble click sobre el botón. El cursor saltará al código de un método que *Builder* crea automáticamente. Este será el método al que se llame cuando el usuario pulse sobre el botón.
 - En el inspector de objetos seleccionar la pestaña *Events* y hacer doble click en el espacio en blanco situado a la derecha del suceso *OnClick*.

En cualquiera de los dos casos escribir *salir();* como cuerpo del método.

7. Añade un menú al formulario principal (componente *TMainMenu* de la paleta de componentes *Standard*) con una opción *Dibujar* y otra *Salir*. El evento *OnClick* de la opción *Salir* debe llamar al método *salir()*.

PARTE II: Implementación del formulario de datos

Pasos a seguir:

1. Añadir al proyecto un nuevo formulario (con *File/New/Form*). Ponerle nombre *FormDatos* y salvarlo su unidad con nombre *UFormDatos.cpp*.
2. Añadir los componentes visuales para pedir los datos:
 - a. Un componente *Label* con propiedad *Caption* a "Coordenada X: " seguido de un componente *Edit* con propiedad *Name* tomando el valor *EditX* y propiedad *Text* puesta a 0. Los componentes *Edit* se utilizan para leer los datos de entrada y no tienen propiedad *Caption* sino *Text*.
 - b. En la línea siguiente un componente *Label* con propiedad *Caption* a "Coordenada Y: " seguido de un componente *Edit* con propiedad *Name* tomando el valor *EditY* y propiedad *Text* puesta a 0.
 - c. Un componente *RadioGroup* de la paleta de componentes *Standard* con propiedad *Caption* a *Colores*. Dentro de este componente situar 3 componentes *RadioButton*, con propiedad *Caption* tomando los valores *Rojo*, *Verde* y *Azul*, y propiedad *Name* a *BotonRadioRojo*, *BotonRadioVerde*, *BotonRadioAzul*, respectivamente. Colocar la propiedad *Checked* del primero a *true*.
 - d. Dos botones con propiedad *Name* a *BotonAceptar* y *BotonCancelar* respectivamente, y con propiedad *caption* *&Aceptar* y *&Cancelar*, respectivamente. En el método asociado al suceso *OnClick* del primer botón incluir el código *ModalResult=mrOk*; (indicando que el diálogo se cierra aceptando los datos) y en el del segundo *ModalResult=mrCancel*;
3. Incluir en la unidad métodos para devolver los valores leídos. Estos métodos pueden ser por ejemplo:

```
// métodos observadores o de consulta
int TFormDatos::getX() {
    return StrToInt(EditX->Text); // StrToInt convierte un string en un entero
}

int TFormDatos::getY() {
    return StrToInt(EditY->Text);
}

TColor TFormDatos::getColor() {
    TColor resultado; // el tipo TColor está predefinido en Builder e incluye
                    // constantes para representar los colores

    if (BotonRadioRojo->Checked)
        resultado = clRed;
    else if (BotonRadioVerde->Checked)
        resultado = clGreen;
    else if (BotonRadioAzul->Checked)
        resultado = clBlue;

    return resultado;
}
```

Recordar definir en la parte *public* de *UFormDatos.h* las cabeceras de estos métodos.

PARTE III: Enlace de las dos partes anteriores. Pasos a seguir:

1. Incluye el archivo cabecera del formulario de datos (`#include "UFormDatos.h"`) en el formulario principal (`UFormPrincipal.cpp`).
2. Asociar a la opción del menú *Dibujar* el siguiente código:

```
// creamos el formulario de datos
TFormDatos* formDatos=new TFormDatos(this);
// lo mostramos
formDatos->ShowModal();
// si el usuario ha pulsado aceptar...
if (formDatos->ModalResult==mrOk)
{
    // obtenemos los datos proporcionados por el usuario
    TColor color = formDatos->getColor();
    int x = formDatos->getX();
    int y = formDatos->getY();

    // guardar el color del pincel actual
    TColor colorPincel = Canvas->Brush->Color;
    // nuevo color para el pincel
    Canvas->Brush->Color = color;
    // dibujar un rectangulo relleno de esquina
    // sup. izq. (x,y) y de tamaño 20x50
    Canvas->FillRect(Rect(x,y,x+20,y+50));
    // recuperar el color del pincel
    Canvas->Brush->Color = colorPincel;
}
```

Interfaz del Builder:

- La Ventana Principal: consta del Menú, de paletas de botones/acciones rápidas y de la Paleta de Componentes.
- Inspector de Objetos: para las Propiedades y Eventos de las componentes de la VCL, en fase de diseño.
- Editor de Código: Con F12 pasamos del formulario a su `cpp`. Manipulación de código estándar, definición de Macros (`Ctrl+May+R` escribir la macro `Ctrl+May+R`). Asistencia en al escritura (*Code Insight*, ayuda on-line del editor). Explorador de código (a la izquierda) para desplazamientos rápidos por el código.
- El gestor del Proyecto: para añadir y eliminar unidades al proyecto.

Uso del depurador del Builder:

- Con el ratón podemos poner y quitar dinámicamente puntos de parada (*break points*) y con F9 podemos ir de punto a punto.
- Se puede ver el contenido de las variables situando el ratón sobre ellas.
- F8 ejecuta paso a paso pero tomando las llamadas a funciones como un sólo paso.
- F4 permite ejecutar hasta el punto del programa en que se encuentra el cursor.
- F7 ejecuta paso a paso.
- *View/Debug Windows*.
- Program reset = `Ctrl F2`.
- Para compilar una unidad `Alt F9`.