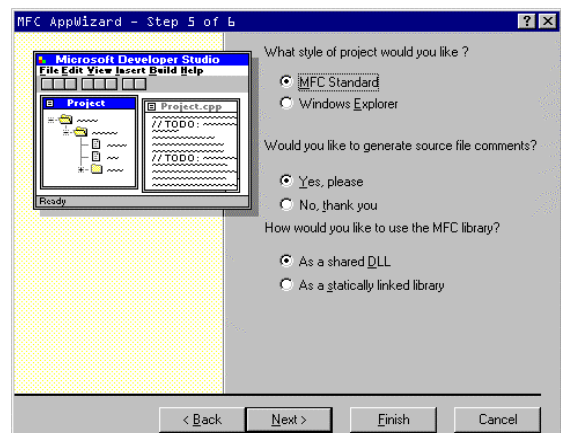
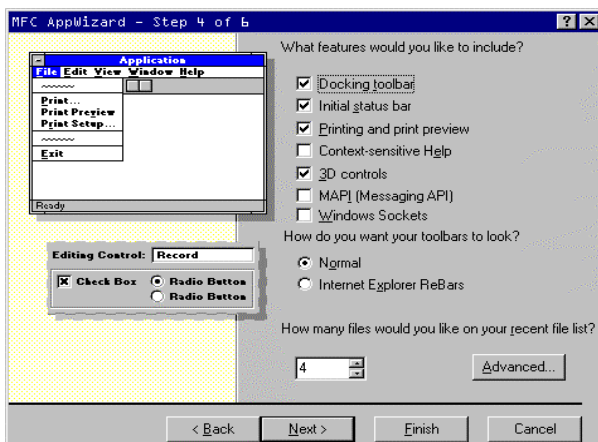
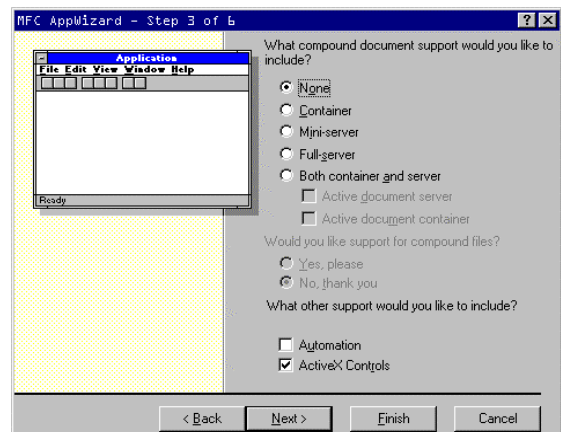
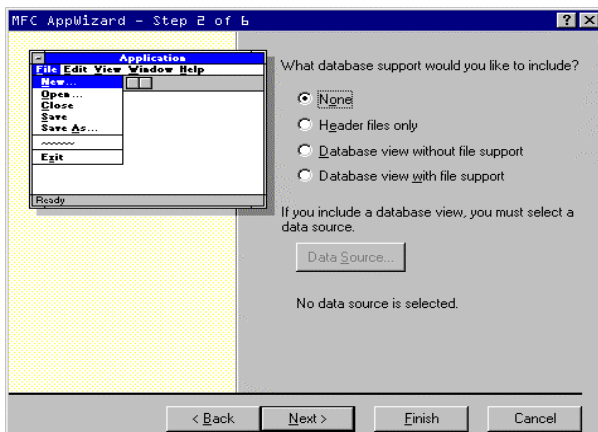
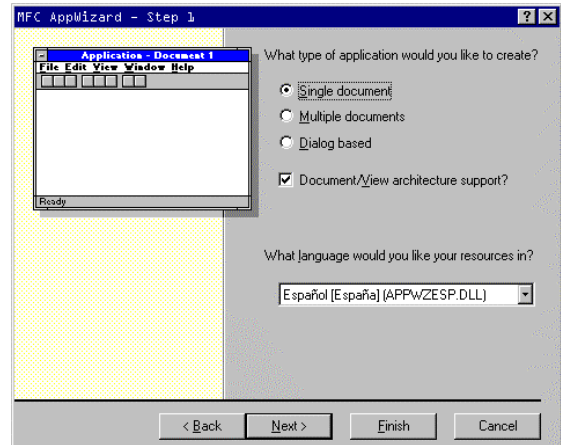
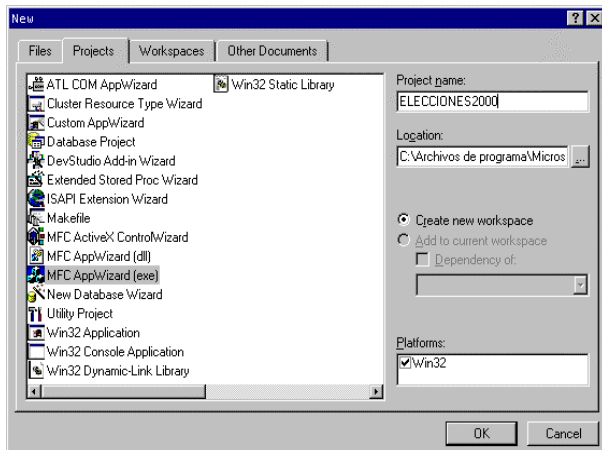


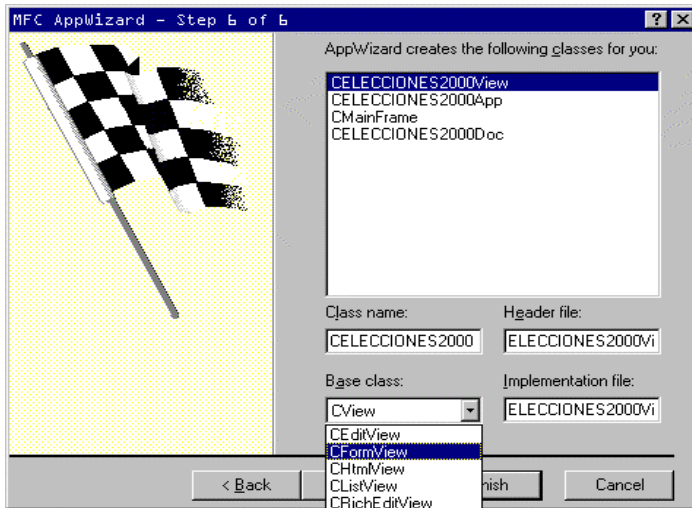
LABORATORIO N° 15

OBJETIVO: Utilizar controles: cajas de texto, botones de pulsación, listas desplegables, etc. asociados con Cajas de dialogo .

PARTE 1**PASO 1.- CREANDO UN NUEVO PROYECTO**

- Antes de que Ud. pueda empezar a codificar, debe crear un nuevo espacio de trabajo y proyecto.
- Para hacerlo, vaya al menú **File** y seleccione **New** .
- Se le presentará un diálogo como este:



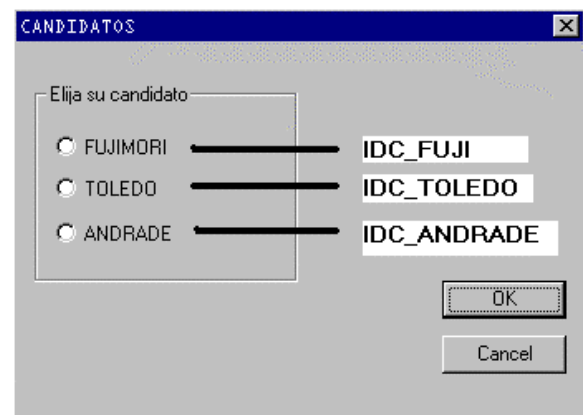


PARTE 2

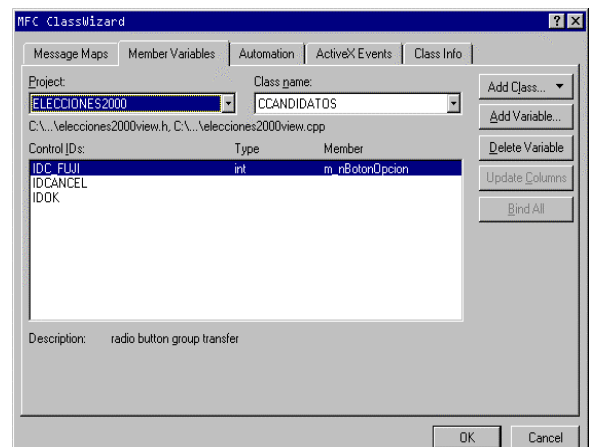
PASO 1. DISEÑO Y CODIGO

```
void CELECCIONES2000View::OnCandidatos()
{
    CCANDIDATOS DLG;
    DLG.DoModal();
}
```

```
void CCANDIDATOS::OnCandidatos()
{
    // TODO: Add your control notification handler code here
    UpdateData(true); //ACTUALIZAR VARIABLES
    if (m_nBotonOpcion==0)
    {
        CFUJIMORI DLG;
        DLG.DoModal();
    }
    else if (m_nBotonOpcion==1)
    {
        CTOLEDO DLG;
        DLG.DoModal();
    }
    else
    {
        CANDRADE DLG;
        DLG.DoModal();
    }
    UpdateData(false); //ACTUALIZAR CONTROLES
}
```



Objeto	Propiedad	Valor
Botón Opción (Fujimori)	Group	si
	TabStop	si
Botón Opción (Toledo)	Group	no
	TabStop	si
Botón Opción (Andrade)	Group	no
	TabStop	si



// CCANDIDATOS dialog

```

class CCANDIDATOS : public CDialog
{
// Construction
public:
    CCANDIDATOS(CWnd* pParent = NULL); // standard constructor

// Dialog Data
    //{AFX_DATA(CCANDIDATOS)
    enum { IDD = IDD_CANDIDATOS };
    int m_nBotonOpcion;
    //}AFX_DATA

// Overrides
    // ClassWizard generated virtual function overrides
    //{AFX_VIRTUAL(CCANDIDATOS)
    protected:
    virtual void DoDataExchange(CDataExchange* pDX); // DDX/DDV support
    //}AFX_VIRTUAL

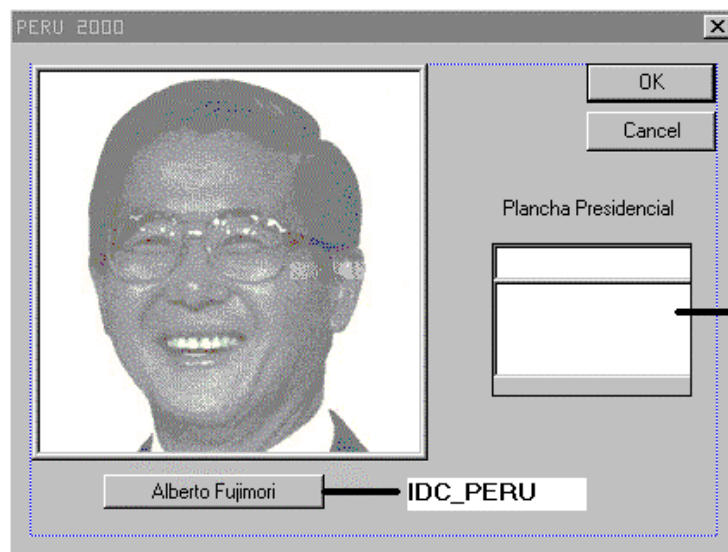
// Implementation
protected:

    // Generated message map functions
    //{AFX_MSG(CCANDIDATOS)
    afx_msg void OnCandidatos();
    //}AFX_MSG
    DECLARE_MESSAGE_MAP()
};

```

PASO 2

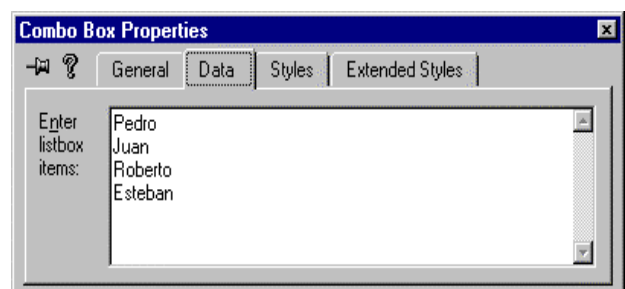
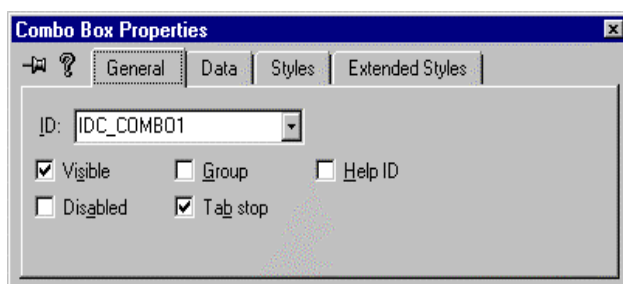
- Diseñe la caja de dialogo "IDD_FUJIMORI":

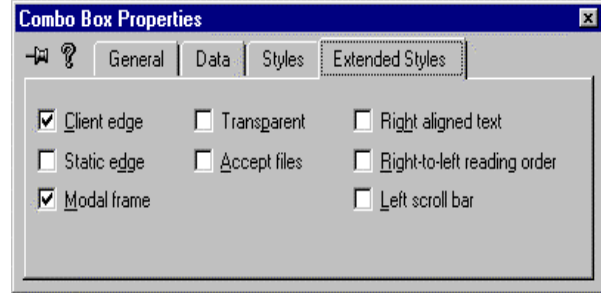
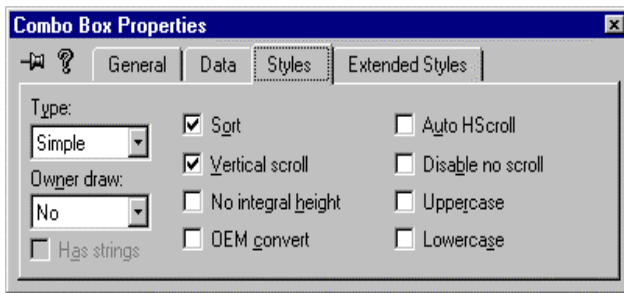


```

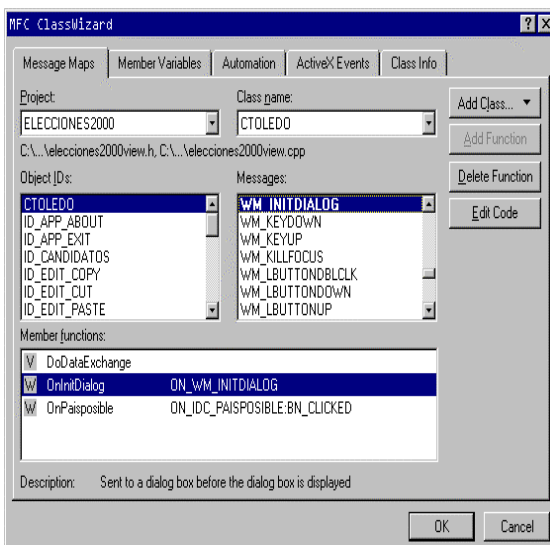
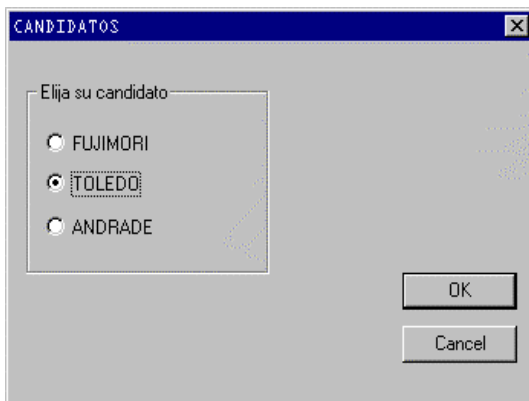
void CFUJIMORI::OnPeru()
{
    // TODO: Add your control notification
    handler code here
    MessageBox("PERU 2000","VIVA!!!",MB_OK);
}

```



**PASO 3:**

- Diseñe la caja de dialogo "IDD_TOLEDO"



```
BOOL CTOLEDO::OnInitDialog()
```

```
{
```

```
    CDialog::OnInitDialog();
```

```
    CListBox*pLB=(CListBox*)GetDlgItem(IDC_PLANCHA);
```

```
    pLB->InsertString(-1,"Juan");
```

```
    pLB->InsertString(-1,"Cesar");
```

```
    pLB->InsertString(-1,"Alex");
```

```
    pLB->InsertString(-1,"Sergio");
```

```
    pLB->InsertString(-1,"Patricia");
```

```
    return TRUE; // return TRUE unless you set the focus to a
```

```
control
```

```
    // EXCEPTION: OCX Property Pages should return
```

```
FALSE
```

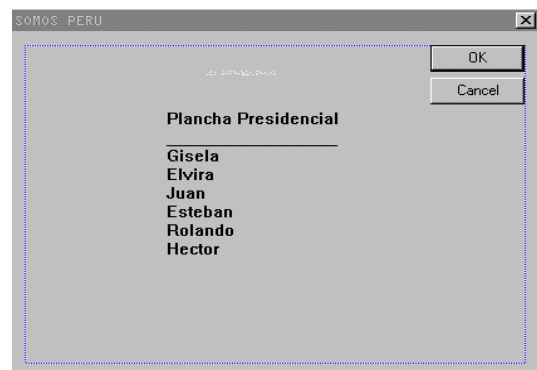
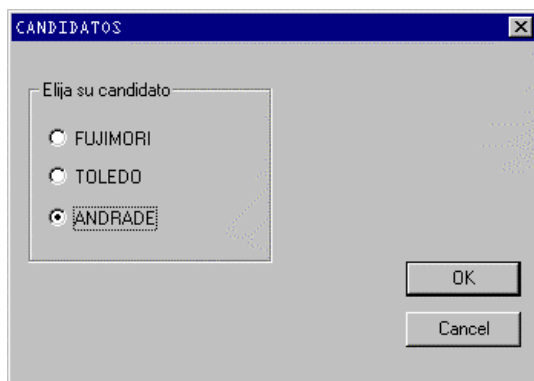
```
}
```

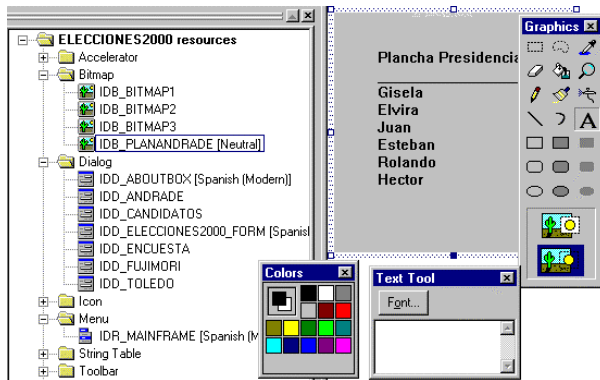
```
void CTOLEDO::OnPaisposible()
```

```
{
```

```
    AfxMessageBox("PAIS POSIBLE",MB_ICONSTOP);
```

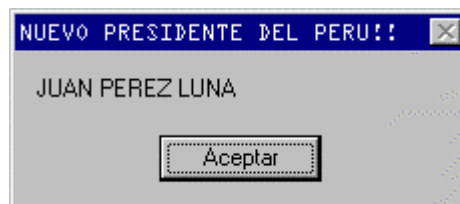
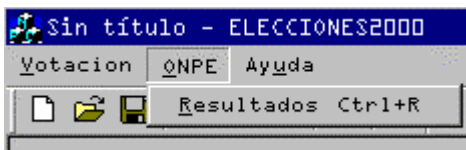
```
}
```

PASO 4: Diseñe la caja de dialogo "IDD_ANDRADE"



- Crear un nuevo **Bitmap**
- Utilizar el caja de herramientas y diseñe la ventana tal como le gusta.

PARTE 3

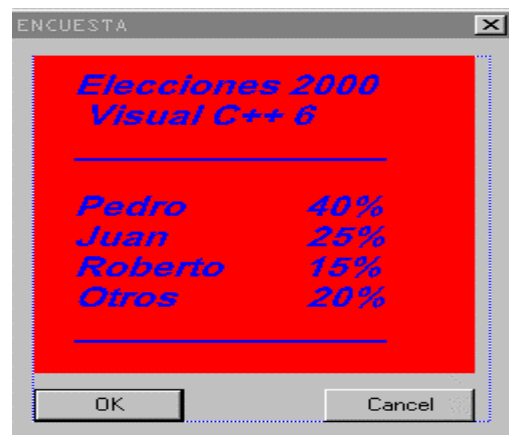


```
void ELECCIONES2000View::OnResultados()
{
    // TODO: Add your command handler code here
    MessageBox("JUAN PEREZ LUNA", "NUEVO PRESIDENTE DEL PERU!!", MB_OK);
}
```

PARTE 4

```
// CENCUESTA message handlers

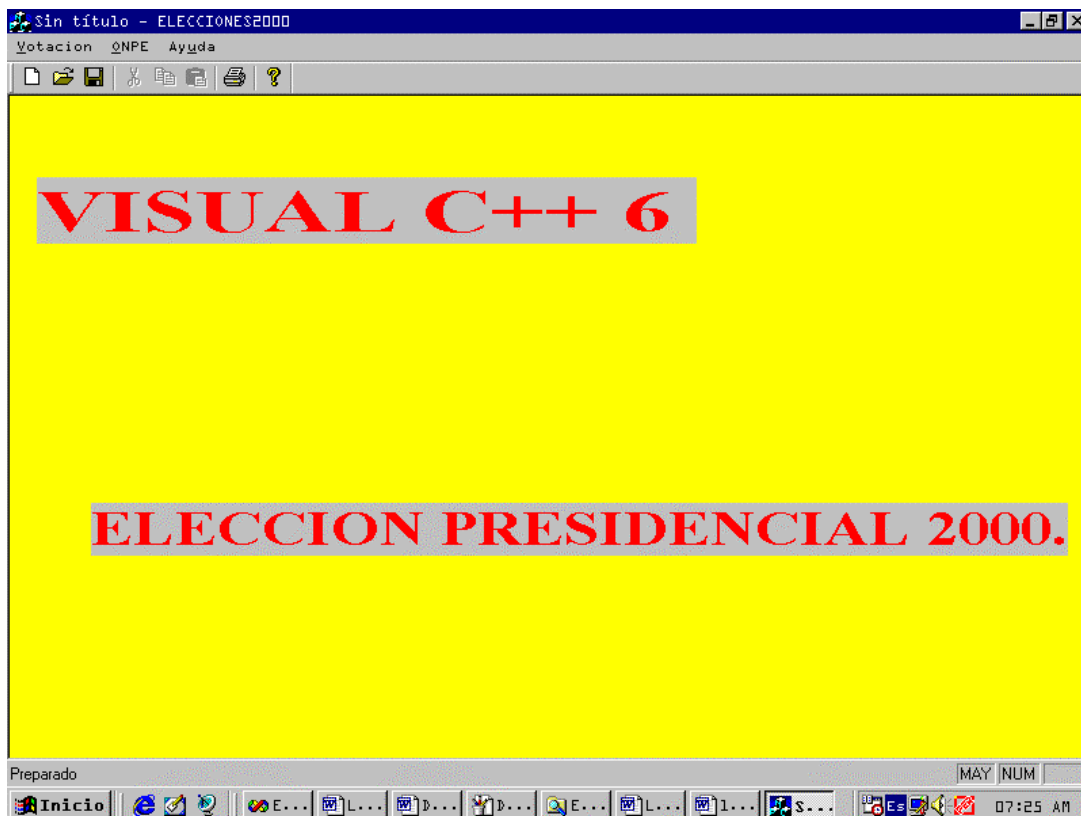
void ELECCIONES2000View::OnEncuesta()
{
    CENCUESTA DLG;
    DLG.DoModal();
}
```



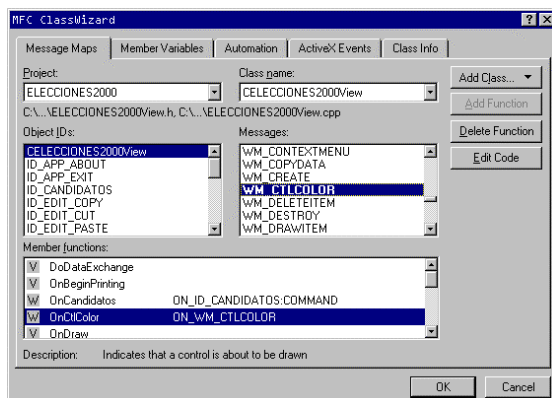
PARTE 5

PASO 1

-Diseñe la figura que se muestra



PASO 2



-Abra ClassWizard(Ctrl+W).

-Seleccione el objeto CELECCIONES2000View.

-Elija el mensaje **WM_CTLCOLOR**

-Haga clic en el botón Add Function.

-ClassWizard añadirá la función **OnCtlColor**.

- Edite esta función

```
HBRUSH CELECCIONES2000View::OnCtlColor(CDC* pDC, CWnd* pWnd, UINT nCtlColor)
{
    HBRUSH hbr = CFormView::OnCtlColor(pDC, pWnd, nCtlColor);

    pDC->SetTextColor(RGB(255,0,0));
    return(HBRUSH)m_Pincel.GetSafeHandle();
    // TODO: Return a different brush if the default is not desired
    return hbr;
}
```

PASO3

-Declare las variables miembros m_Color y m_Pincel

```
class CELECCIONES2000View : public CFormView
{
protected: // create from serialization only
    CELECCIONES2000View();
    DECLARE_DYNCREATE(CELECCIONES2000View)

public:
   //{{AFX_DATA(CELECCIONES2000View)
    enum{ IDD = IDD_ELECCIONES2000_FORM };
        // NOTE: the ClassWizard will add data members here
   //}}AFX_DATA
    COLORREF m_Color; //color
    CBrush m_Pincel; //pincel
// Attributes
```

PASO 4:

- A continuación, modifique la función **OnInitialUpdate()**

```
void CELECCIONES2000View::OnInitialUpdate()
{
    CFormView::OnInitialUpdate();
    GetParentFrame()->RecalcLayout();
    ResizeParentToFit();
    m_Color =RGB(255,255,0);//AMARILLO
    m_Pincel.CreateSolidBrush(m_Color);//inicializar el pincel
}
```

PASO 5:

```
CELECCIONES2000View::~CELECCIONES2000View()
{
    m_Pincel.DeleteObject();//Elimina el pincel virtual
}
```

OBSERVACIONES:

- Cada vez que un control(denominado también ventana filial o ventana hija) esta a punto de repetirse, Windows envía un mensaje WM_CTLCOLOR al procedimiento de ventana padre.
- Esto hace que se llame a la función manipuladora del mensaje, CWnd::OnCtlColor, la cual nos permitirá escribir el código necesario para alterar los colores que el procedimiento de ventana utiliza para pintar los controles.
- Esta función tiene el esquema siguiente:

```
HBRUSH CELECCIONES2000View::OnCtlColor(CDC* pDC, CWnd* pWnd, UINT nCtlColor)
{
    HBRUSH hbr = CFormView::OnCtlColor(pDC, pWnd, nCtlColor);

    // TODO: Return a different brush if the default is not desired
    return hbr;
}
```

pDC Contiene un puntero al contexto de dispositivo del control.
 pWnd Contiene un puntero al control que pregunta por el color
 nCtlColor Especifica el tipo de control. Puede ser:

CTLCOLOR_BTN	Botón
CTLCOLOR_DLG	Caja de diálogo
CTLCOLOR_EDIT	Caja de texto
CTLCOLOR_LISTBOX	Lista
CTLCOLOR_MSGBOX	Caja de mensajes
CTLCOLOR_SCROLLBAR	Barras de desplazamiento
CTLCOLOR_STATIC	Etiquetas

La función debe retornar el handle al pincel que se quiere utilizar para pintar.

Este handle puede obtenerse llamando a la función miembro GetSafeHandle de la clase CGdiObject