تعلم البرمجة بواسطة MFC

تأليف عابر ياسين

بسم الله الرحمن الرحيم وبه أستعين ولاحول ولا قوة إلا بالله العلي العظيم ومن الله أستمد المعونة والتوفيق
سبحانك يارب لا علم لنا إلا ما علمتنا إنك أنت العليم الحكيم رب زدني علما وصلى الله على سيدنا محمد وعلى آله وصحبه وسلم تسليما

أخي المبرمج سم الله وتوكل على الله وابدأ معي المشوار فرحلة الألف ميل نبدأ بخطوة

1.- ال MFC اختصار Microsoft Foundation Classes

1.1.- تعريف:

إن ال MFC عبارة عن مجموعة من الفئات صممتها مايكروسوفت خصيصا لإنشاء برنامج نوافذية وذات واجهة رسومية يتعامل معها المستخدم بالفأرة وأدوات المعاينة، إن كتابة برنامج نوافذي يمرنك على إنشاء واستعمال كائنات مكتبات MFC أو كائنات الفئات المشتقة من MFC ، برنامج Visual C++ يضع في متناول المبرمجين الأدوات التي تساعدهم في تسهيل وتخصيص استعمال مكتبات ال MFC كما أنه يحتوي على معالج لتوليد شيفرة ال MFC الأكثر استعمالا مثل هياكل MDI و SDI التي يتعامل معها المبرمجون بكثرة كما أن فيه معالجات تتولى القيام بتوليد شيفرة الرسائل Message وهي الشيفرة التي بواسطتها يتعرف الويندوز على الحدث المرسل مثل النقر بزر الماوس والضغط على لوحة المفاتيح وغيرها من الأحداث مما يجعل المبرمجين يهتمون أو يركزون فقط على برنامجهم المنوط بهم بواسطة استدعاء وظائف ال MFC الجاهزة أو تطوير فئات ترتكز عليها تكون أكثر كفاءة.
وعندما ننشئ مشروع MFC فإن Visual C++ ينشئ إطار عمل يضم جميع الملفات التي يتعامل معها المشروع مثل ملفات الموارد Resource وملفات الفئات التي تنتهي بالامتداد h وغيرها من الملفات.

2.1.-ملاحظة:

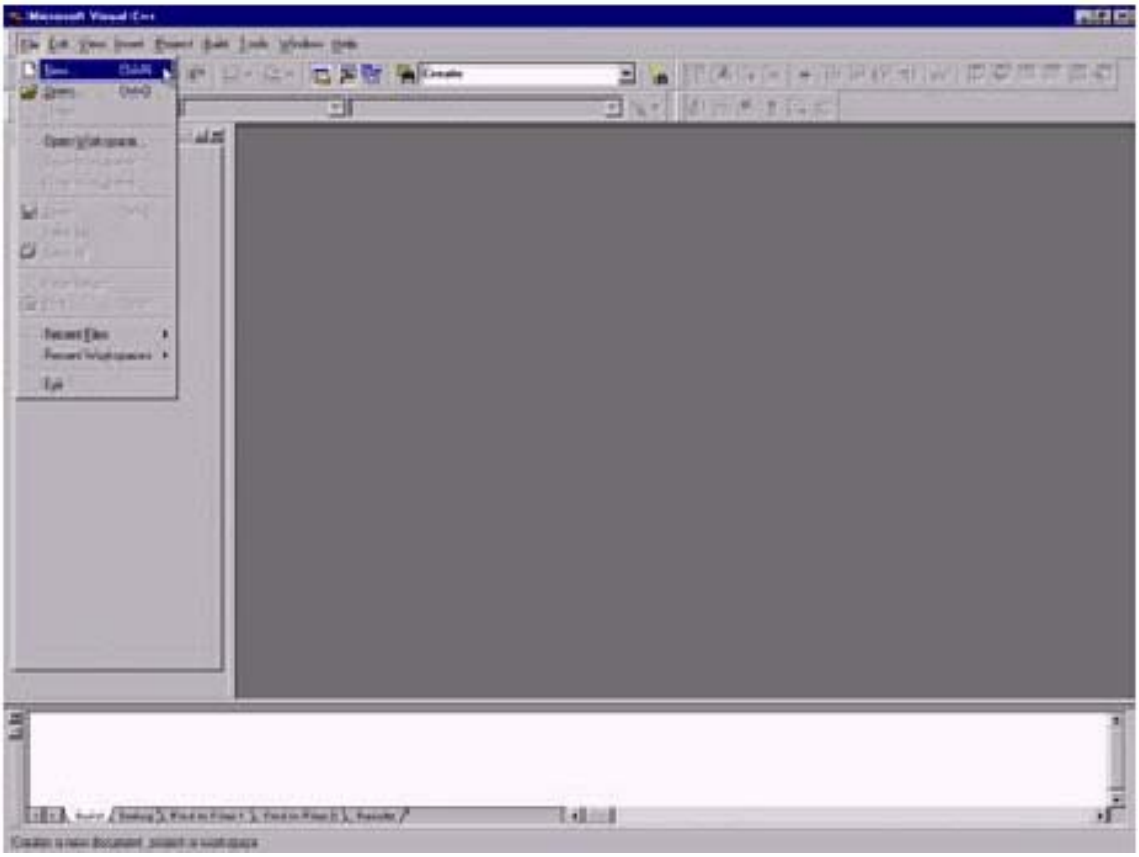
كل فئات MFC تحمل اسما يبتدئ بالحرف C مثل CDocument,CView لهذا ينصح بشدة التزام هذا الأمر لإنشاء فئات نظيفة وفعالة وأيضا قياسية.
أما أعضاء الفئات Members فإنها تستهل بالبادئة m_

23 بناء تطبيق نوافذي (Windows):

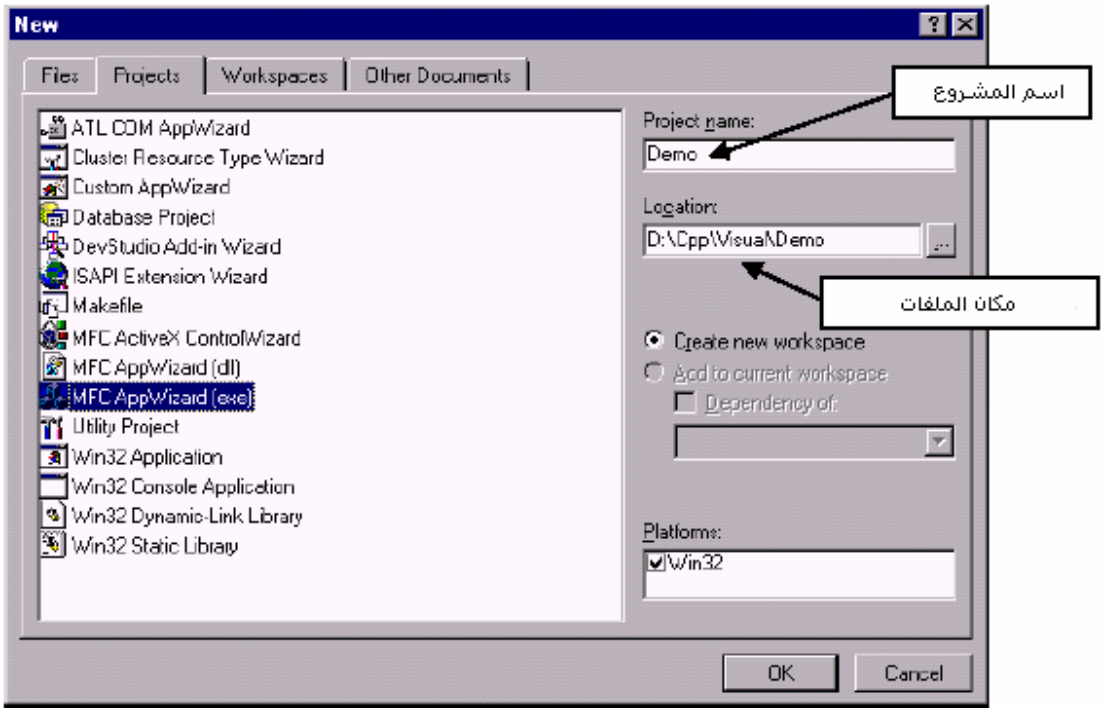
1.3.- ال AppWizard:

يوفر Visual C++ معالجا يدعى AppWizard يتولى ويغنينا عن مهمة عظيمة وهي بناء هيكل البرنامج الذي نود تطويره فهو يعرِّف تلقائيا جميع الفئات الضرورية لتطوير برنامج ويندوز على أي نمط تريده وفيما يلي عرض لكيفية استعمال هذا المعالج السحري اتبع الخطوات التالية:

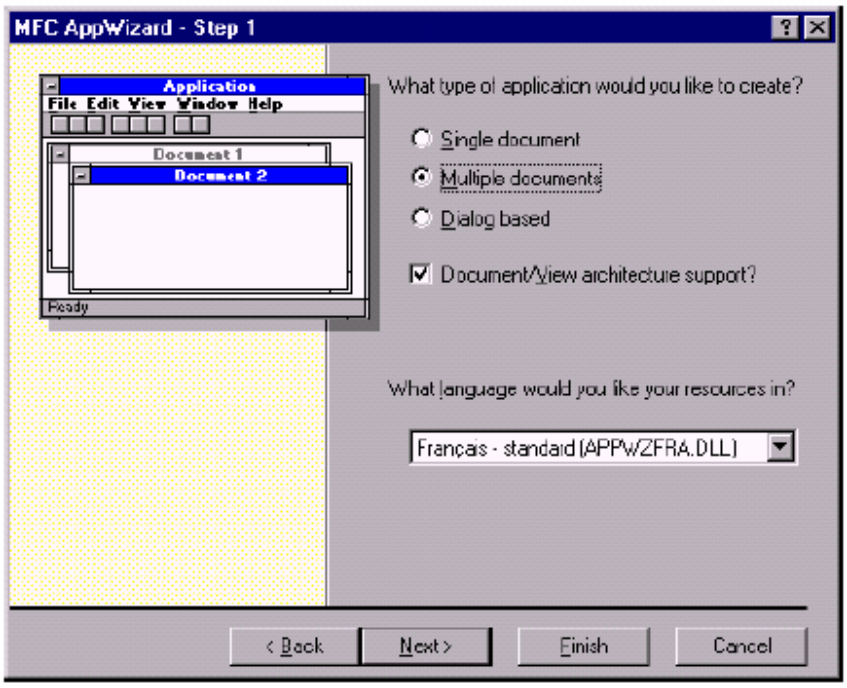
الخطوة 1: فتح مشروع جديد:



الخطوة 2: اختر النوع MFC AppWizard (exe) ، واكتب اسم المشروع بالأحرف اللاتينية وحدد مسار وضع الملفات المشروع واحذر أن يكون ضمن اسم المسار كلمة عربية فقد سبب لي أنا شخصيا مشكلة عند تفسير البرنامج ولم أتفطن لها إلا بعد جهد .

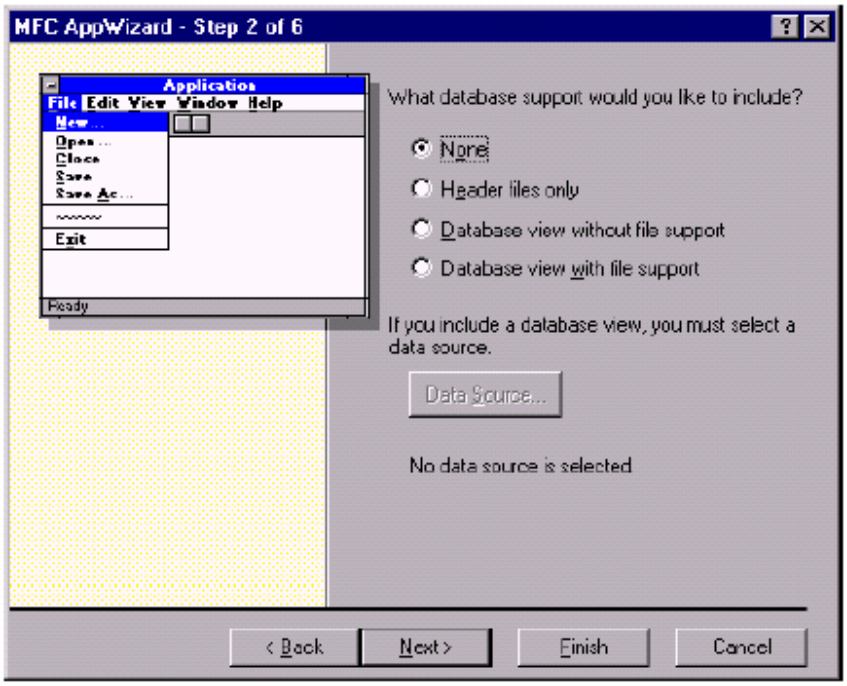


الخطوة 3: اختر نوع المستند الذي تود إنشاءه بسيط SDI أو متعدد MDI :

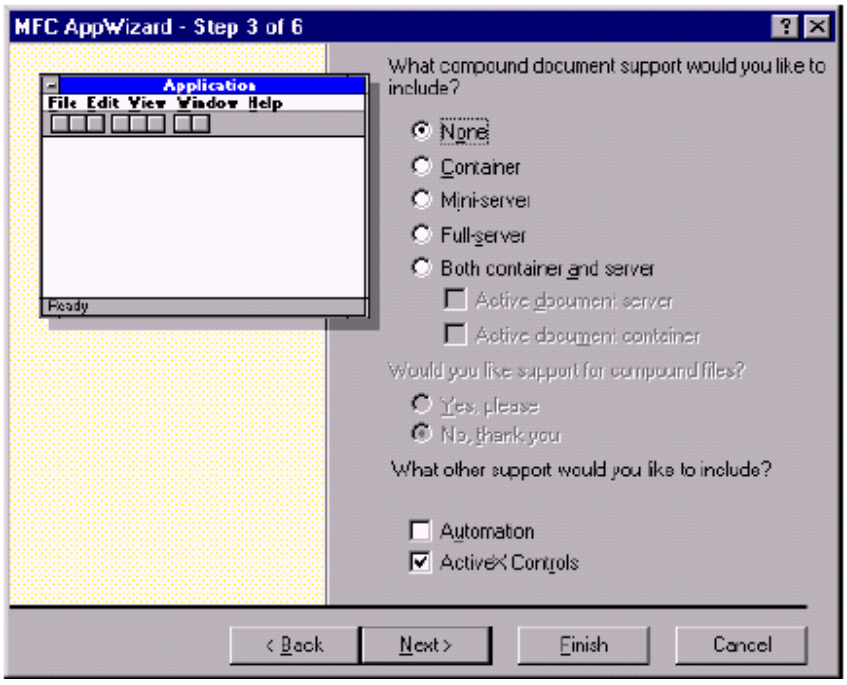


في أغلب الأحيان يستحسن الاحتفاظ بالقيم الافتراضية في الخطوات 4 5 6 و 7

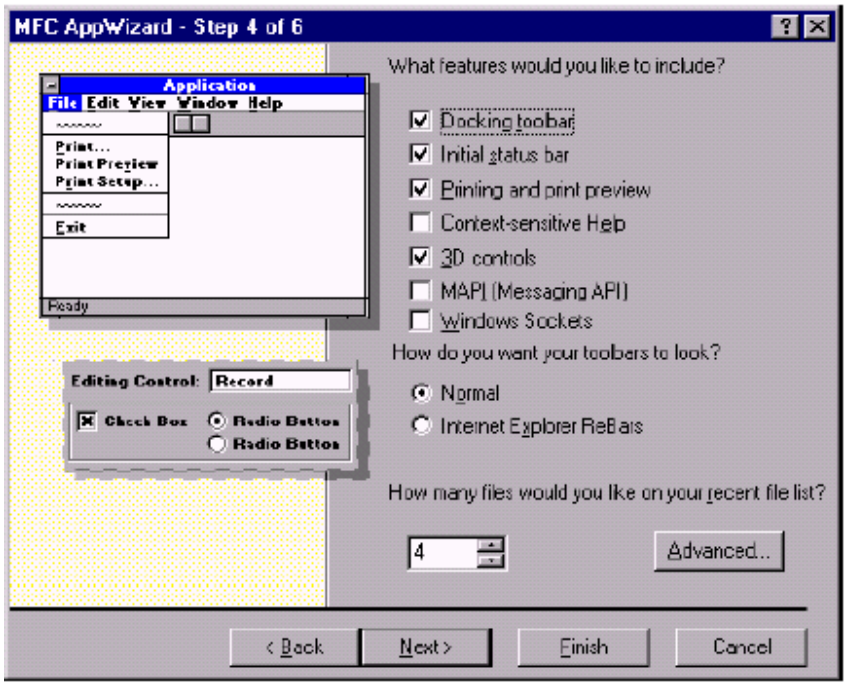
الخطوة 4: تحديد قاعدة البيانات المربوطة بالتطبيق (خيار محتمل عند التعامل مع قواعد البينات).



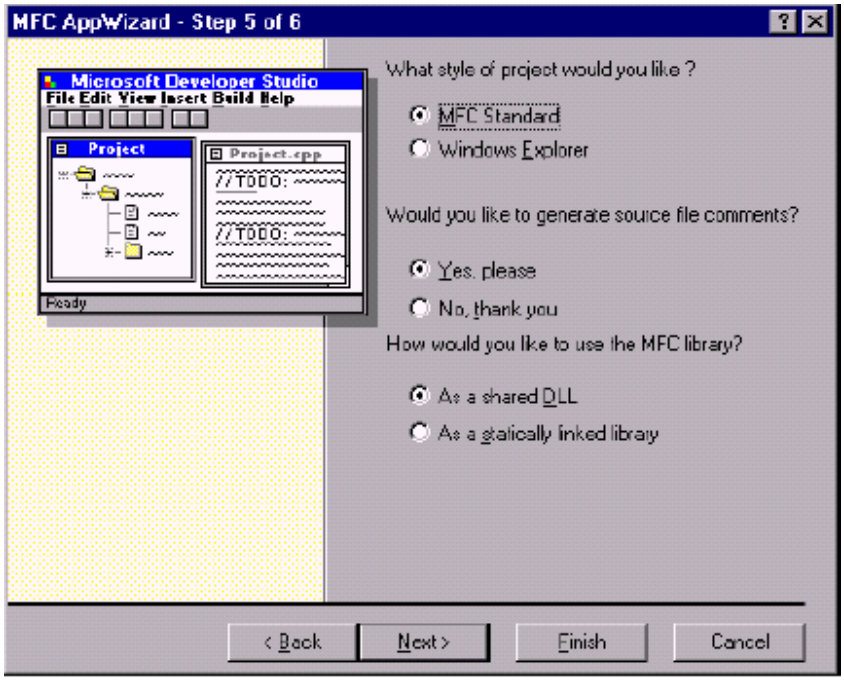
الخطوة 5: هذه الخطوة لربط كائنات OLE و ActiveX بالتطبيق غالبا يكون التطبيق بحاجة إلى ملفات ActiveX أخرى.



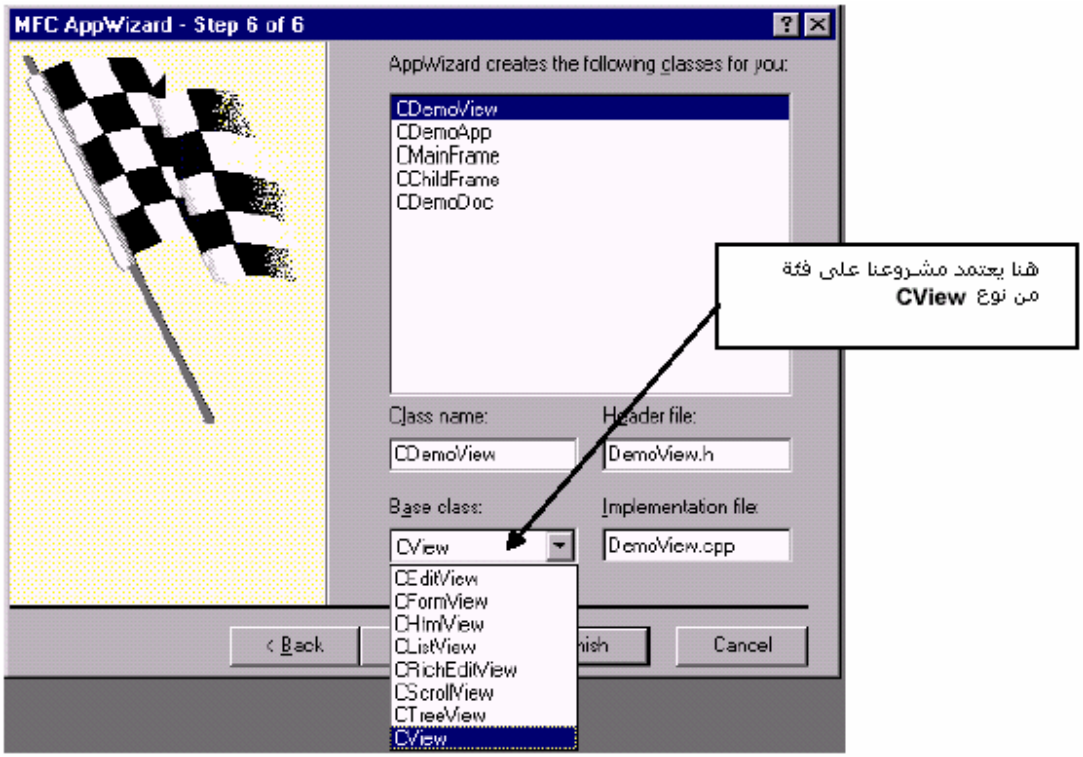
الخطوة 6: قائمة لخيارات متعدد يدمجها المعالج في تطبيقك مثل إدراج شريط الحالة ودعم كائنات ثلاثية الأبعاد وغيرها اختر منها ما تشاء.



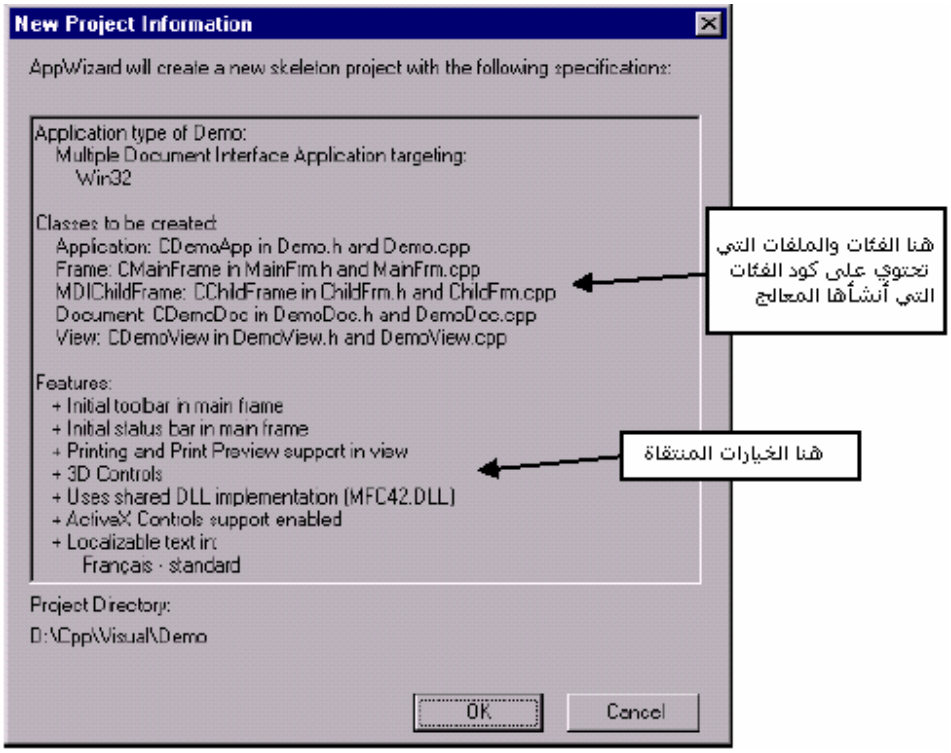
الخطوة 7: اختر نمط Style التطبيق خيار Windows Explorer ينشئ البرنامج شبيه بمستكشف الويندوز.



الخطوة 8: اختر الفئة التي يعتمد عليها التطبيق



وفي الأخير عرض لتقرير البرنامج بخياراته المحددة:



3 . 2 .- نظرة على الفئات المولدة :

ال AppWizard يولد دائما الفئات التالية:

CMainFrame : تمثل النافذة الرئيسية للتطبيق التي تحتوي على القوائم وأشرطة الأدوات وشريط الحالة.

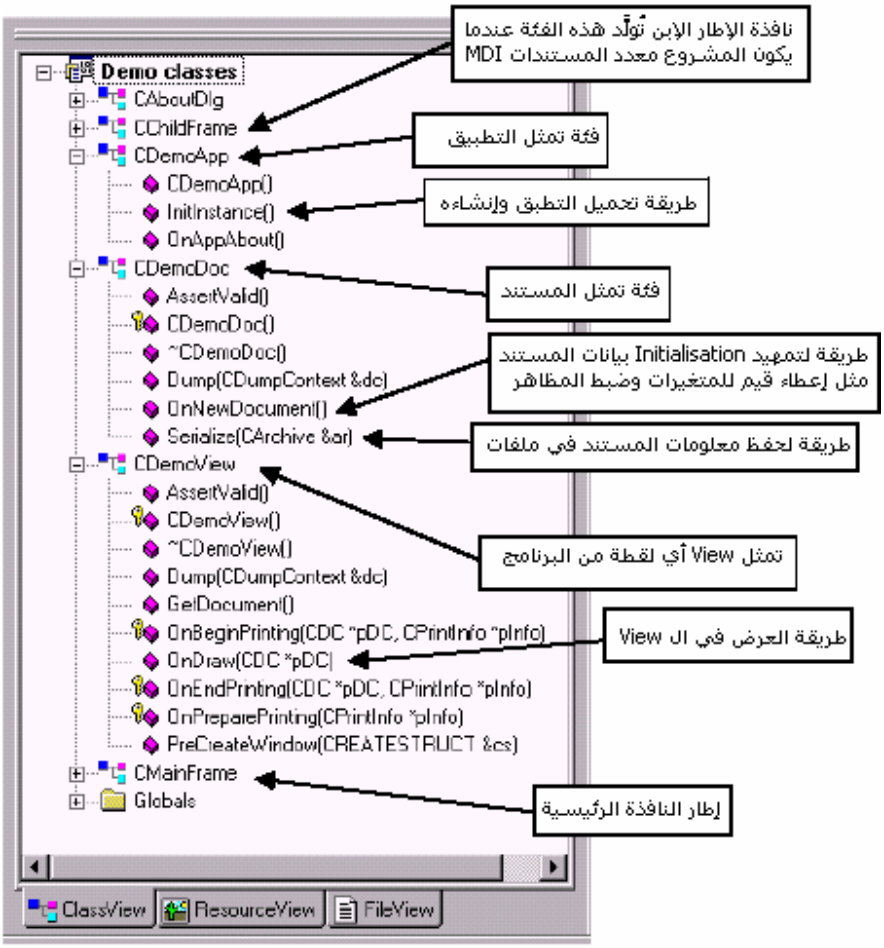
CChildFrame: تولد هذه الفئة عندما يكون نوع التطبيق متعدد المستندات MDI وكل كائن من هذه الفئة يمثل مستندا جديدا للتطبيق.

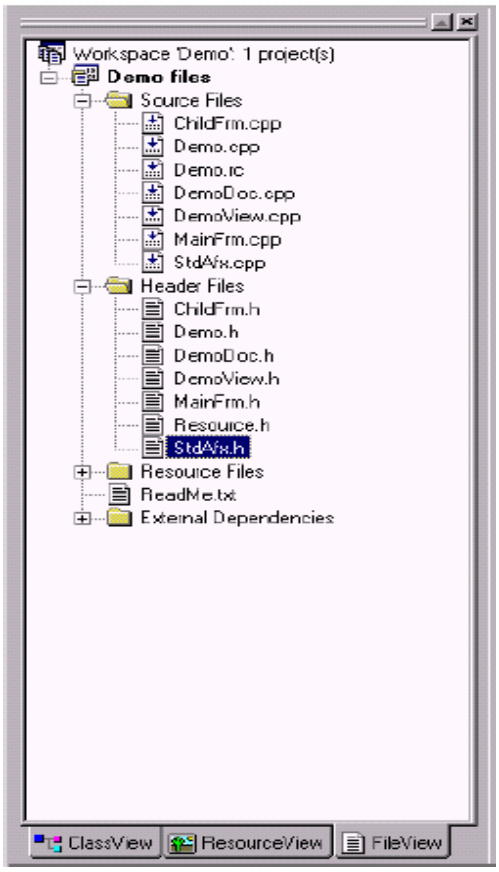
C<Project Name>App : (مثل **CDemoApp)** تمثل هذه الفئة شكل التطبيق وتحتوي على طريقة هامة و هي **InitInstance** وهي تمثل بداية إنشاء البرنامج.

C<Project Name>Doc : (مثل **CDemoDoc)** فئة تمثل كل مستند من التطبيق وهي بدورها تحتوي على أعضاء تمثل المستند

C<Project Name>View : (مثل **CDemoView)** فئة تمثل ال View الرئيسية للمستند وهي المعرفة في الخطوة الثامنة.

يتعلق بكل فئة ملفان ملف بالمتداد ".h" الذي يعرف كل أعضاء الفئة من طرق ومتغيرات والثاني بالامتداد ".cpp" والذي يمثل هيكل الطرق.

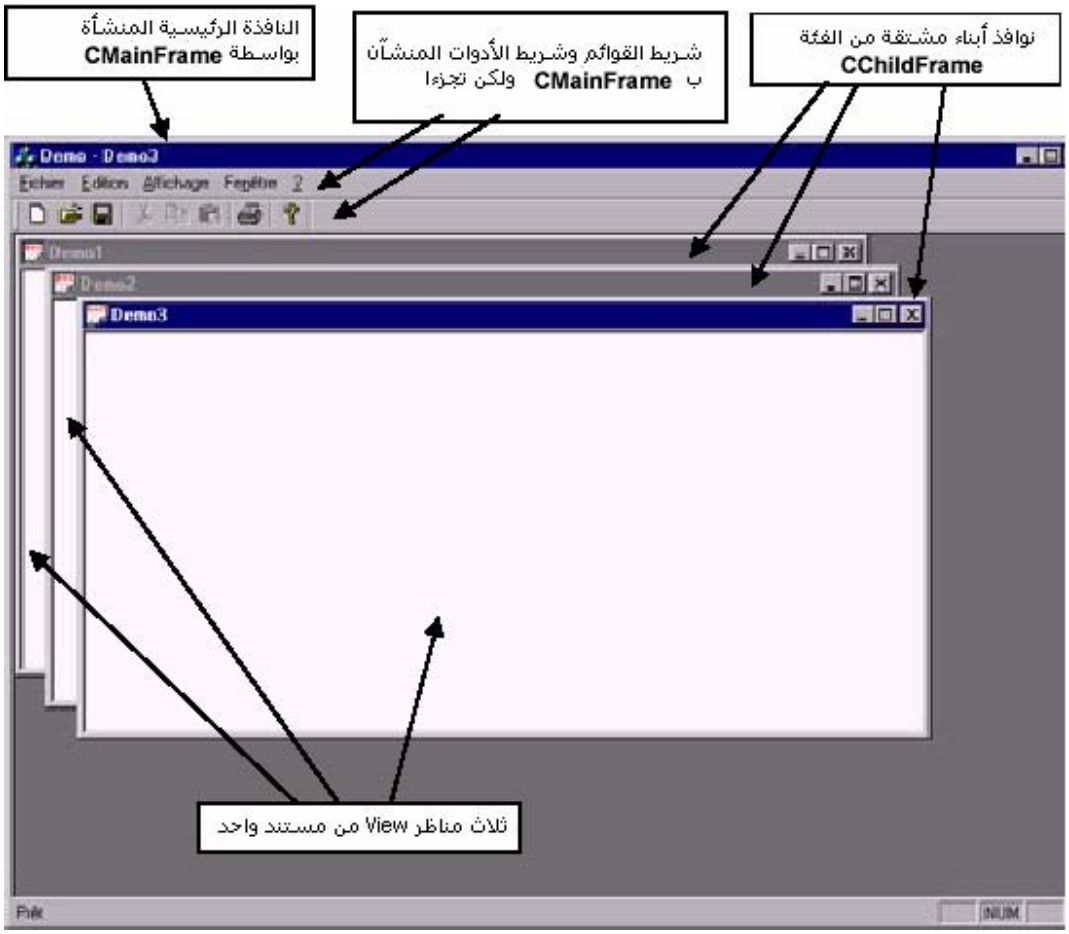

Demo classes
CAboutDlg
CChildFrame
CDemoApp
CDemoApp()
InitInstance()
OnAppAbout()
CDemoDoc
AssertValid()
CDemoDoc()
~CDemoDoc()
Dump(CDumpContext &dc)
OnNewDocument()
Serialize(CArchive &ar)
CDemoView
AssertValid()
CDemoView()
~CDemoView()
Dump(CDumpContext &dc)
GetDocument()
OnBeginPrinting(CDC *pDC, CPrintInfo *pInfo)
OnDraw(CDC *pDC)
OnEndPrinting(CDC *pDC, CPrintInfo *pInfo)
OnPreparePrinting(CPrintInfo *pInfo)
PreCreateWindow(CREATESTRUCT &cs)
CMainFrame
Globals
ClassView
ResourceView
FileView
نافذة الإطار الإبن تُولّد هذه الفئة عندما يكون المشروع معدد المستندات MDI
فئة تمثل التطبيق
طريقة تحميل التطبق وإنشاءه
فئة تمثل المستند
طريقة لتمهيد Initialisation بيانات المستند مثل إعطاء قيم للمتغيرات وضبط المظاهر
طريقة لحفظ معلومات المستند في ملفات
تمثل View أي لقطة من البرنامج
طريقة العرض في ال View
إطار النافذة الرئيسية



3 . 2 .- تفسير البرنامج وتنفيذه:

بعد تفسير البرنامج وتشغيله سنحصل على تطبيق Windows قياسي (النافذة الرئيسية القوائم شريط الأدوات ...)**وذلك بدون كتابة سطر واحد من الكود أليس هذا رائعا!.**

هذا التطبيق مبني على واجهة متعددة المستندات MDI فيمكنك إنشاء أكثر من مستند واحد في آن واحد جرب الأمر New من القائمة File .



4. 1.- تشخيص التطبيق:

4 .1 . بيان كيف يبنى البرنامج والطريقة InitInstance .

الطريقة InitInstance من الفئة CDemoApp تبين هندسة البرنامج العامة وكيف يتم بناؤه كود هذه الطريق يتم توليده بواسطة المعالج AppWizard أما أنا وأنت عزيزي المبرمج فلا يهمنا من هذا الكود سوى السطور التالية:

```
...
// Register the application's document templates. Document templates
// serve as the connection between documents, frame windows and views.
CMultiDocTemplate* pDocTemplate;
pDocTemplate = new CMultiDocTemplate(
IDR_DEMOTYPE,
RUNTIME_CLASS(CDemoDoc),
RUNTIME_CLASS(CChildFrame), // custom MDI child frame
RUNTIME_CLASS(CDemoView));
AddDocTemplate(pDocTemplate);
// create main MDI Frame window
CMainFrame* pMainFrame = new CMainFrame;
if (!pMainFrame->LoadFrame(IDR_MAINFRAME))
return FALSE;
m_pMainWnd = pMainFrame;
...
```

إنشاء مستند من نوع MDI

إنشاء إطار النافذة الرئيسية

الوسـيط الأول للطريقـة **CMultiDocTemplate** وهـو **IDR_DEMOTYPE** يعـرف القائمـة وشـريط الأدوات المستعملان في التطبيق والوسائط الثلاثة التالية تعرف المستند (**CDemoDoc**) النافذة الإبـن (**CChildFrame**) و المنظر View المربوط مع المستند (**CDemoView**)
أما الماكرو RUNTIME_CLASS فهو يسمح بتعيين نوع الفئة أثناء التشغيل

4. 2. - إضافة الكود : مثال مع الطريقة Ondraw.

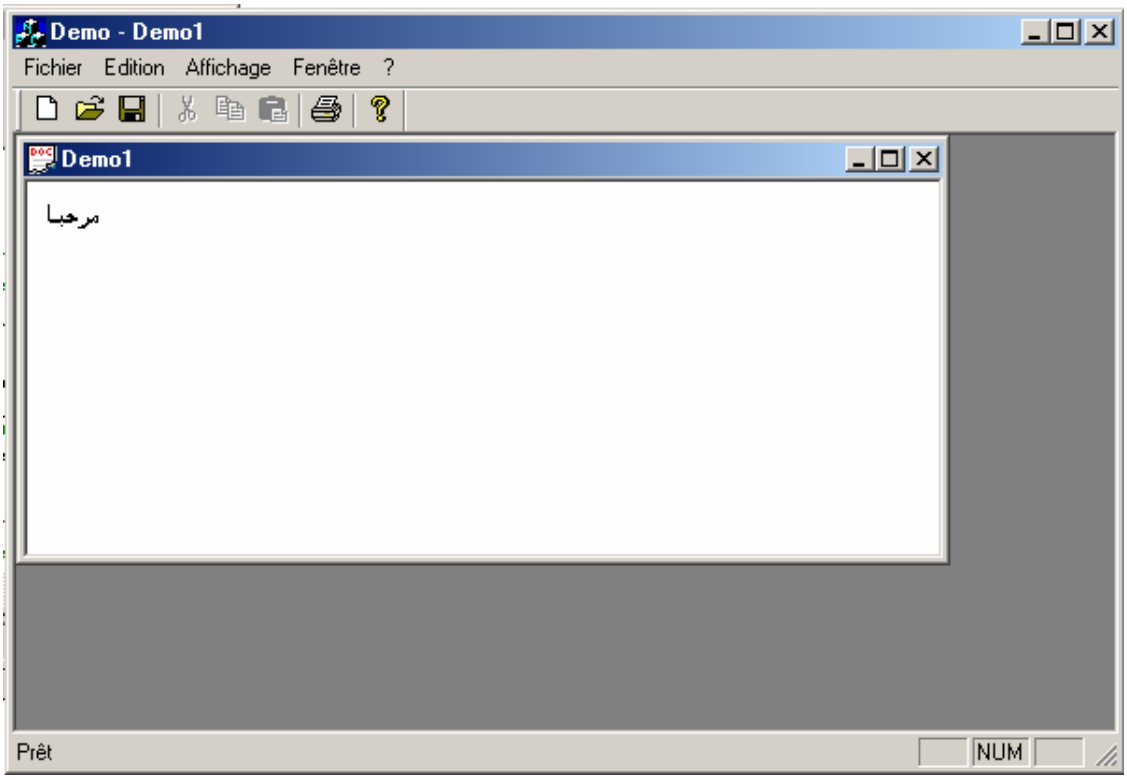
كل فئة من النوع CView تشتمل على الطريقة OnDraw يولدها المعالج AppWizard تسمح بالكتابة أو الرسم في المنظر View .
الطريقة OnDraw تأخذ الوسيط pDC من الفئة CDC أنظر إلى الكود المولد:

```
////////////////////////////////////////////////
// CDemoView drawing
void CDemoView::OnDraw(CDC* pDC)
{
  CDemoDoc* pDoc = GetDocument();
  ASSERT_VALID(pDoc);
// TODO: add draw code for native data here
}
```

ملاحظة: الفئة CDC تحتوي على كل الأدوات الضرورية للرسم والكتابة.
إذا أردنا إضافة سطر من الكود يسمح لنا بكتابة السلسلة "مرحبا"علـى الشاشـة فالطريقـة المناسـبة هي TextOut أضف السطر التالي الموضح بالأزرق:

```
////////////////////////////////////////////////////////////////////////////
// CDemoView drawing

void CDemoView::OnDraw(CDC* pDC)
{
	CDemoDoc* pDoc = GetDocument();
	ASSERT_VALID(pDoc);
	// TODO: add draw code for native data here
	pDC->TextOut(10,10,"مرحبا");
}
```

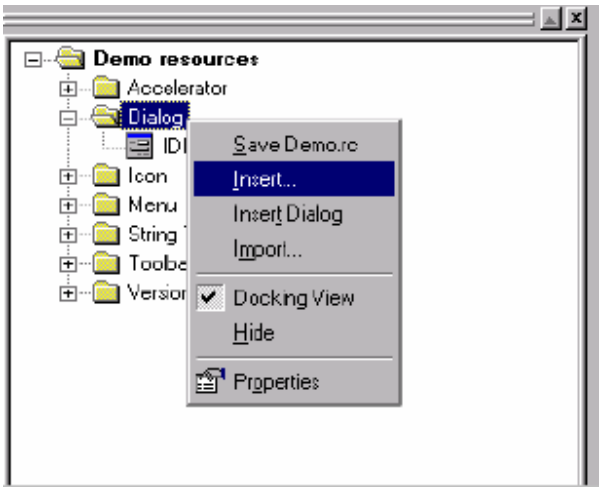
بعد التشغيل تحصل على هذه النافذة:



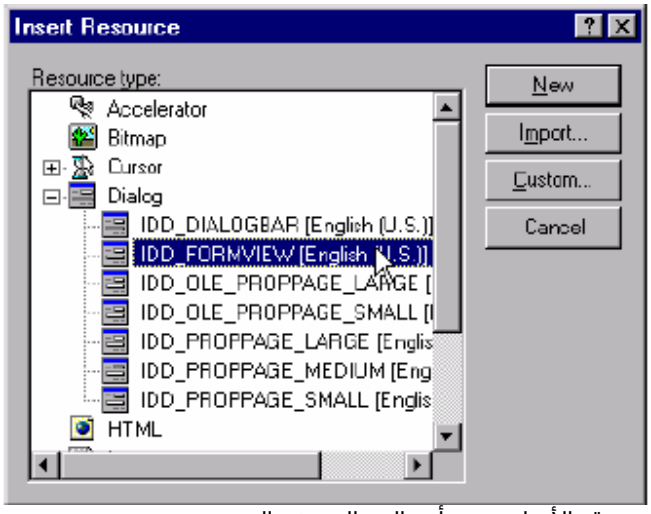
الكلمة مرحبا كتبت بقياس 10 pixels مابين x وy في الجانب العلوي من اليسار للنافذة الرئيسية.

4. 3. - إضافة نموذج مرئي يمكن التعامل معه أثناء التصميم:

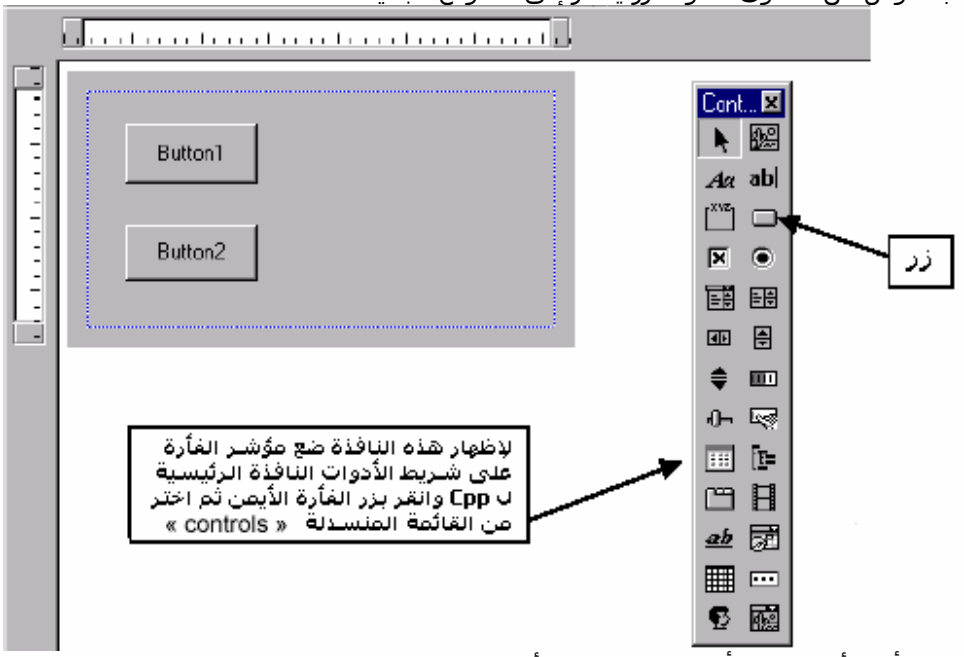
لإضافة عنصر جديد انقر على الشريحة « RessourceView » انشر العنصر « Dialog » وانقر بزر الماوس الأيمن وا ختر من القائمة الفرعية الأمر « Insert ».



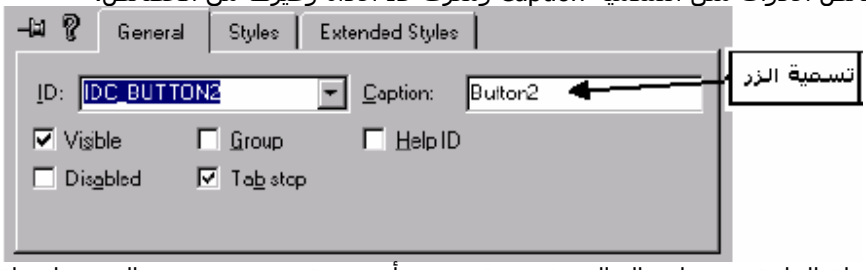
في النموذج Insert Resource انشر العنصر Dialog واختر نوع النموذج وليكن مثلا "FormView".



أضف بالماوس من صندوق الأدوات زري أمر إلى النموذج الجديد.

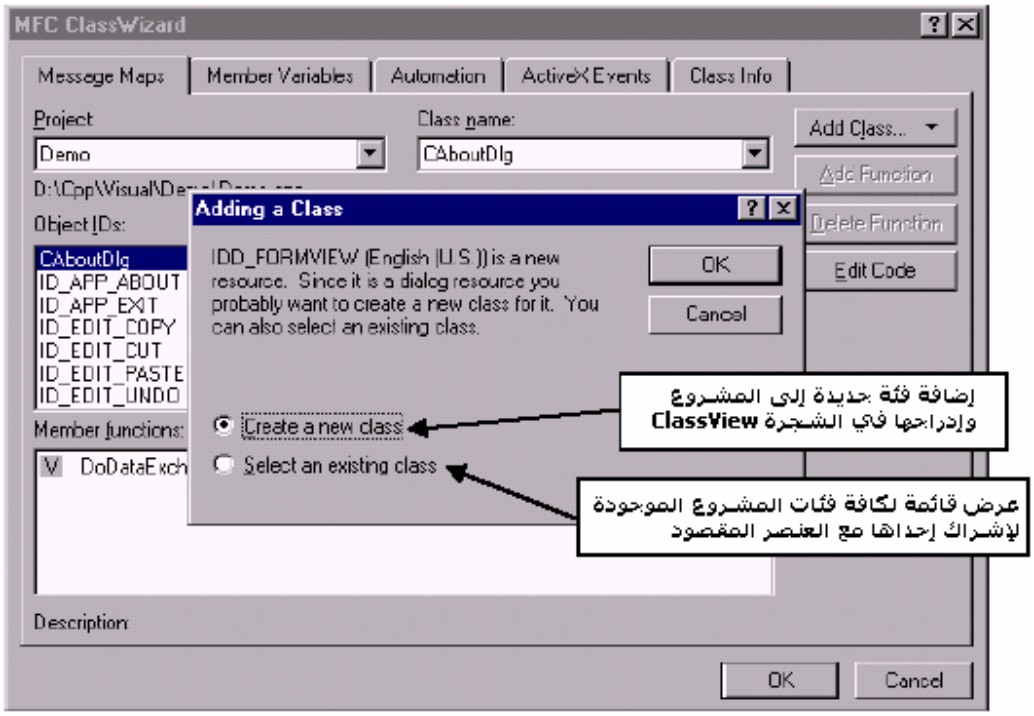


انقر بزر الفأرة الأيمن على أحد الزرين واختر الأمر "Properties" تظهر لك النافذة التالية التي تتيح لك تغيير خصائص الأدوات مثل التسمية Caption ومعرف ID الأداة وغيرها من الخصائص.

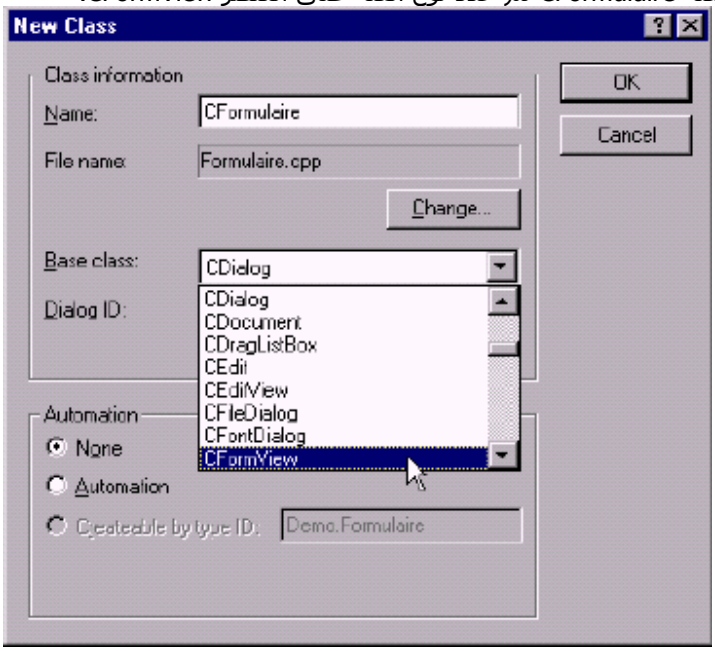


بقيت النقطة الهامة وهي إشراك النموذج بفئة موجود أو جديدة وهي تعتبر روح التي بها حياة النموذج التي يمكننا من خلالها التعامل مع الأحداث والخصائص والطرق

لإشراك فئة جديدة بالنموذج انقر بزر الماوس الأيمن على مكان فارغ في النموذج واختر من القائمة المنسدلة الخيار "ClassWizard"



سننشئ فئة جديدة وعند ظهور النافذة الموالية اكتب اسم الفئة في الخانة Name مع مراعاة البادئة C وليكن اسم الفئة CFormulaire ثم حدد نوع الفئة على العنصر CFormView.



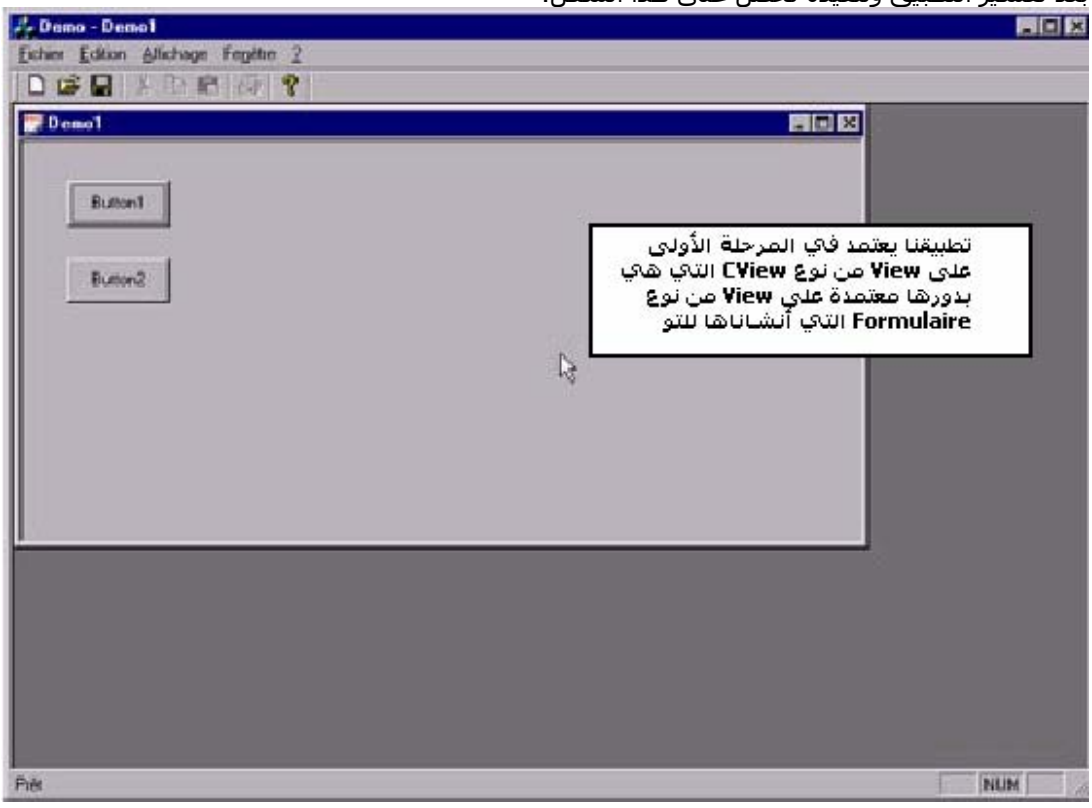
لتغيير كود الطريقة InitInstance التابعة للفئة CDemoApp انقر على الشريحة "Class View" انقر علامة الجمع في الفئة CDemoApp ثم انقر نقرا مزدوجا على اسم الطريقة InitInstance.

```
#include "Formulaire.h"
// Register the application's document templates.  Document templates
// serve as the connection between documents, frame windows and views.

CMultiDocTemplate* pDocTemplate;
        pDocTemplate = new CMultiDocTemplate(
        IDR_DEMOTYPE,
                RUNTIME_CLASS(CDemoDoc),
                RUNTIME_CLASS(CChildFrame), // custom MDI child frame
                RUNTIME_CLASS(CFormulaire));
        AddDocTemplate(pDocTemplate);

        // create main MDI Frame window
        CMainFrame* pMainFrame = new CMainFrame;
        if (!pMainFrame->LoadFrame(IDR_MAINFRAME))
                return FALSE;
        m_pMainWnd = pMainFrame;
```

يضاف في أعلى الفئة

غير ال View إلى CFormulaire

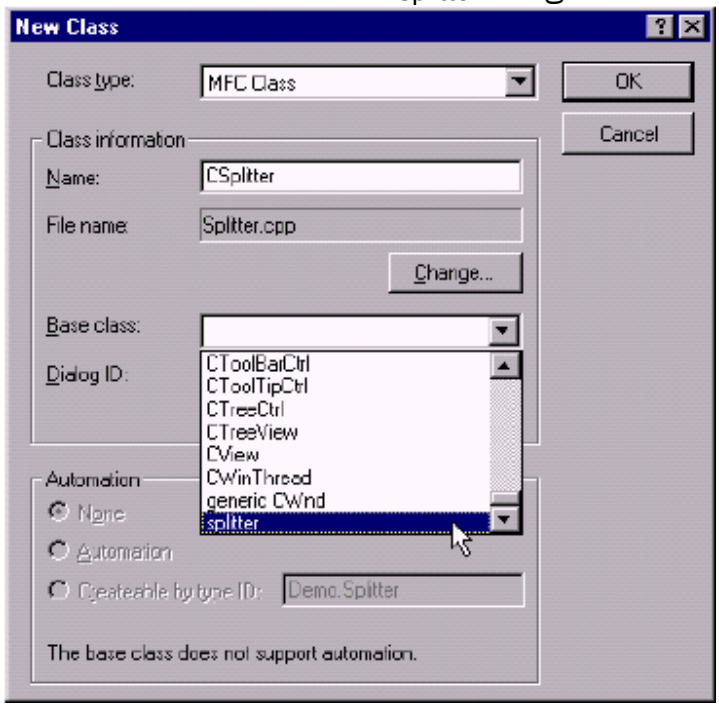
بعد تفسير التطبيق وتنفيذه نحصل على هذا الشكل:



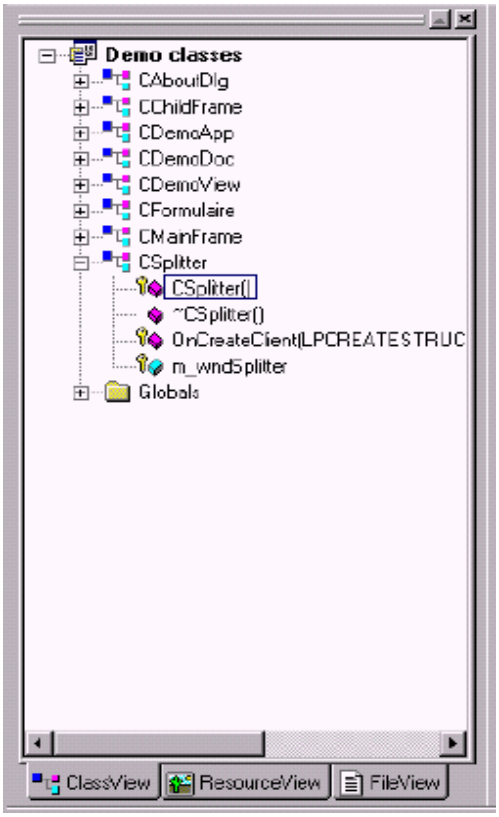
4. 4.-إضافة فاصل ("splitter").

كثيرا ما نحتاج في تطبيقاتنا إلى دمج مستندين في إطار واحد، لفعل ذلك يجب إنشاء فاصل ("splitter") وجعل هذين المستندين بين هذا الفاصل وقد يكون المستندان من نوع مختلف أو من نوع واحد

إذن يجب أولا إنشاء فئة من النوع "CSplitterWnd" المشتقة من الفئة "CWnd".
حدد الشريحة "ClassView" ضع مؤشر الفأرة على اسم المشروع وانقر بالزر الأيمن ثم اختر الأمر New Class أضف فئة تعتمد على الفئة Splitter.



بمجرد إنشاء فئة من نوع Splitter يجب تعريف العناصر التي سندمجها في إطار واحد ونفصلها بالفاصل الفئة CSplitter تحتوي على طريقة تنشأ تلقائيا تقوم بهذه المهمة حسب المتطلبات وهي الطريقة OnCreateClient.



هاهو الكود المولد أثناء إنشاء الفئة CSplitter

```
BOOL	CSplitter::OnCreateClient(LPCREATESTRUCT	/*lpcs*/,	CCreateContext*
pContext)
{
	return m_wndSplitter.Create(this,
		2, 2,	// TODO: adjust the number of rows, columns
		CSize(10, 10),  // TODO: adjust the minimum pane size
		pContext);
}
```

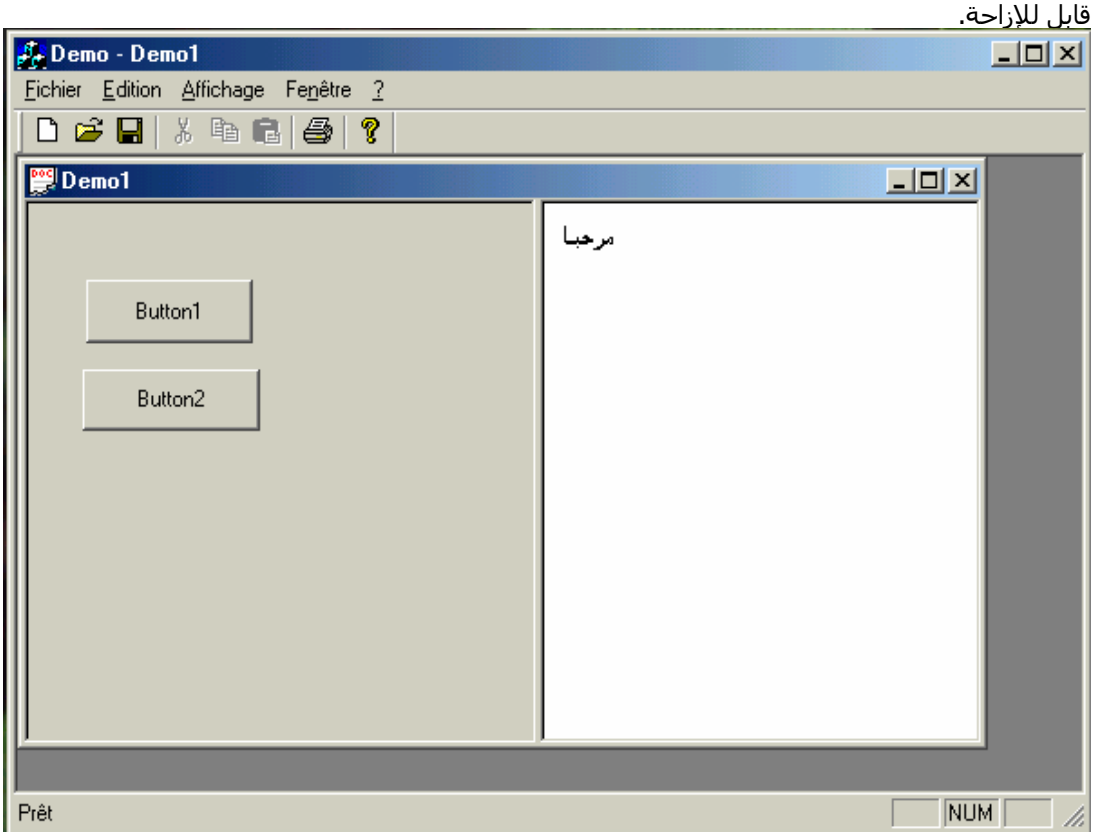
استبدله بالكود التالي:

```
BOOL	CSplitter::OnCreateClient(LPCREATESTRUCT	/*lpcs*/,	CCreateContext*
pContext)
{
	if (!m_wndSplitter.CreateStatic(this, 1, 2))//إنشاء فاصل ذي خط1 وحقلين2
	{
	TRACE0("Failed to CreateStaticSplitter\n");
	return FALSE;
	}
	if (!m_wndSplitter.CreateView(0, 0,//إضافة اللوح الأول من الفاصل
	RUNTIME_CLASS(CFormulaire), CSize(300, 0), pContext))
	{
	TRACE0("Failed to create first pane\n");
	return FALSE;
	}
	if (!m_wndSplitter.CreateView(0, 1,//إنشاء اللوح الثاني من الفاصل
	RUNTIME_CLASS(CDemoView), CSize(0, 0), pContext))
```

موضع الفاصل بين اللوحين

```
        {
        TRACE0("Failed to create second pane\n");
        return FALSE;
        }
        SetActiveView((CView*)m_wndSplitter.GetPane(0,1));// تنشيط الView ال
        return TRUE;
}
                                                كما لا تنسى تصريح هذين الملفين في أعلى الفئة
#include "Formulaire.h"
#include "DemoView.h"
                        وبعد ذلك يجب إعادة تهيئة الطريقة InitInstance وضبطها على الفاصل
        CMultiDocTemplate* pDocTemplate;
        pDocTemplate = new CMultiDocTemplate(
                IDR_DEMOTYPE,
                RUNTIME_CLASS(CDemoDoc),
                RUNTIME_CLASS(CSplitter), // custom MDI child frame
                RUNTIME_CLASS(CDemoView));
        AddDocTemplate(pDocTemplate);
دون أن تنسى تصريح ملف الفئة CSplitter في أعلى الفئة التي تحتوي على هذه الطريقة وهي
                                                                        الفئة CDemoApp
#include "Splitter.h"
                                        كما يجب أيضا إضافة هذا التصريح في أعلى الفئة CDemoView
#include "DemoDoc.h"
```

بعد التفسير والتشغيل ستحصل على ما يلي مستند يحمل نموذجين مفصولا بينهما بفاصل عمودي قابل للإزاحة.



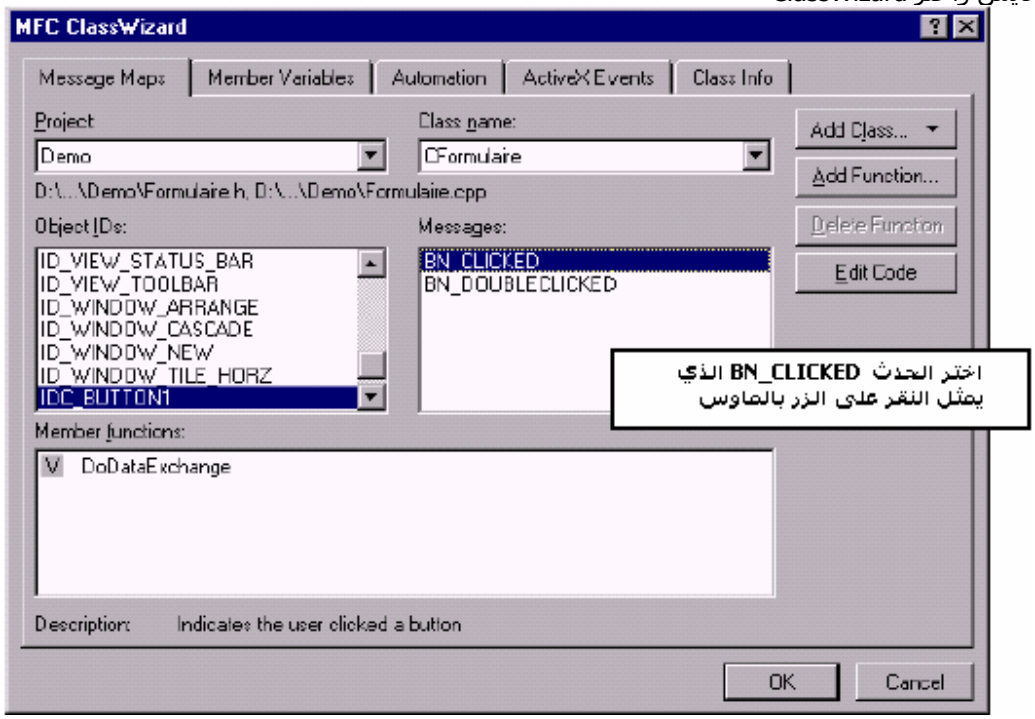
5.- الاتصال بين منظرين:

لنأخذ المثال السابق ونظيف إليه هاتين العمليتين:

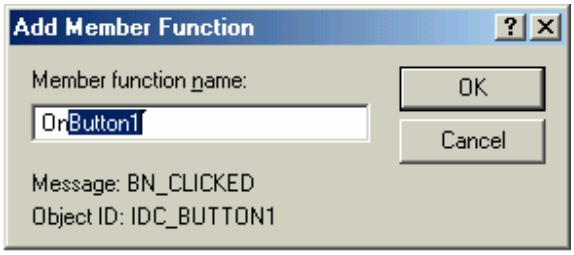
- ✓ بالنقر على الزر الأول في المنظر Formulaire نكتب البسملة في المنظر الثاني
- ✓ بالنقر على الزر الثاني نعرض مربع حوار بسيط على الشاشة.

5. 1.- تصريح حدث بواسطة ClassWizard

لربط حدث النقر بالزر أنقر على الشريحة "ResourceView" افتح علامة الجمع في العنصر "Dialog" واختر النافذة IDD_FORMVIEW وانقر عليها نقرا مزدوجا لإظهارها أنقر على الزر الأول بزر الماوس الأيمن واختر ClassWizard



انقر "Add Function" غير إن شئت اسم الحدث باسم معبر ثم انقر "Edit Code".



الحدث OnButton1 سيتم استدعاؤه في كل مرة ينقر عليه بالماوس.

هذه هي الشيفرة التي يولدها المعالج عند إضافة الحدث

```
void CFormulaire::OnButton1()
{
	// TODO: Add your control notification handler code here

}
```

5. 2.- تفعيل الأحداث:

والآن لنربط الكود بالحدث سيتم إجراء ما يلي عند النقر على الزر

- ✓ استرجاع كائن المستند المشترك مع النموذج
- ✓ إلحاق القيمة بالمتغير الذي ستعرض قيمته على المنظر هذا المتغير من نوع CString
- ✓ تحديث القيمة الجديدة بواسطة الطريقة UpdatData
- ✓ استدعاء طريقة العرض UpDateAllViews التي بدورها تستدعي الطريقة OnDraw

```
////////////////////////////////////////////////////////////////////////////
// CFormulaire message handlers

void CFormulaire::OnButton1()
{
	CDemoDoc *pDoc = (CDemoDoc *)GetDocument();
	pDoc->hello = "بسم الله الرحمن الرحيم";
	UpdateData(TRUE);
	pDoc->UpdateAllViews(this);
}
```

لا تنس تصريح الملف DemoDoc في أعلى الفئة

```
#include "DemoDoc.h"
```

بعد ذلك يجب أن تقوم الطريقة OnDraw التابعة للفئة CDemoView بما يلي:

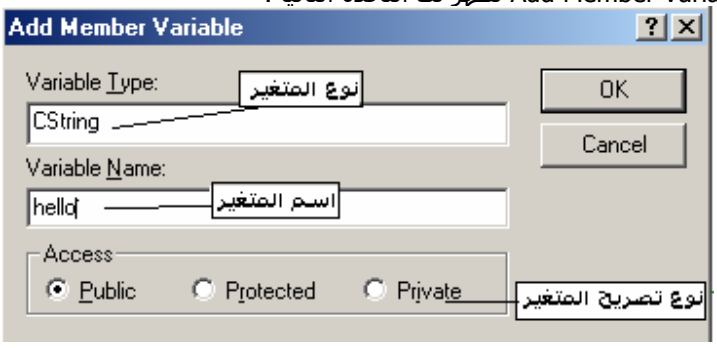
- ✓ استرجاع المنظر View المقصود.
- ✓ عرض القيمة الجديد على الشاشة بواسطة الطريقة TextOut

```
void CDemoView::OnDraw(CDC* pDC)
{
	CDemoDoc* pDoc = GetDocument();
	ASSERT_VALID(pDoc);
	// TODO: add draw code for native data here
	pDC->TextOut(10,10,pDoc->hello);
}
```

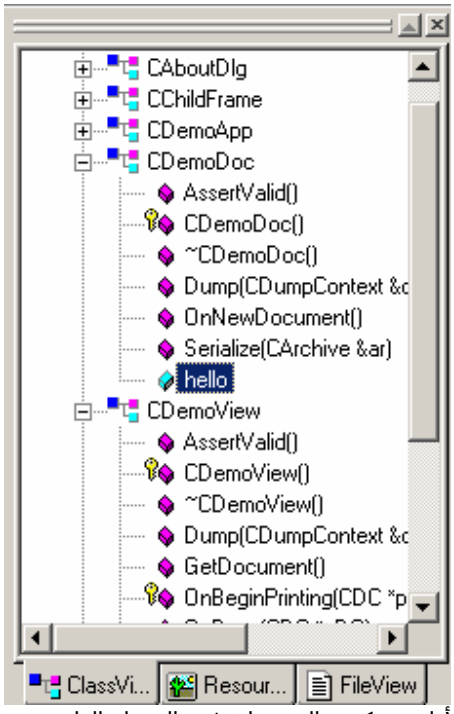
المتغير hello يجب تصريحه عاما في الفئة CDemoDoc من نوع CString وتهيئته على سلسلة فارغة في الطريقة OnNewDocument التابعة للفئة CDemoDoc كما يلي:

```
BOOL CDemoDoc::OnNewDocument()
{
	if (!CDocument::OnNewDocument())

		return FALSE;
	// TODO: add reinitialization code here
	// (SDI documents will reuse this document)
	hello = "";
	return TRUE;
}
```

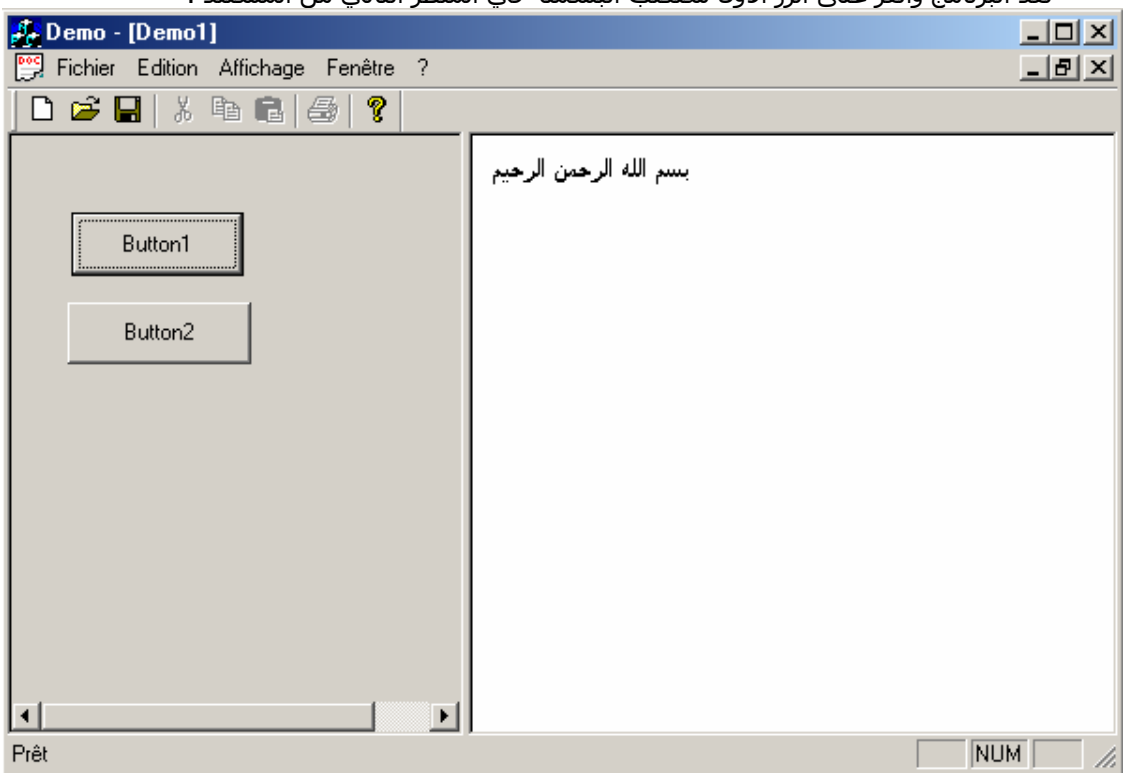
لتصريح المتغير hello انقر على الشريحة Class View ثم انقر على الفئة CDemoDoc بزر الماوس الأيمن وانقر Add Member Variable تظهر لك النافذة التالية:



بعد إدخال المعطيات الصحيحة انقر Ok وسيتم إدراج المتغير وللتحقق من ذلك ستجده ضمن عقد الشجرة ضمن الفئة التابع لها.



نفذ البرنامج وانقر على الزر الأول ستكتب البسملة في المنظر الثاني من المستند :

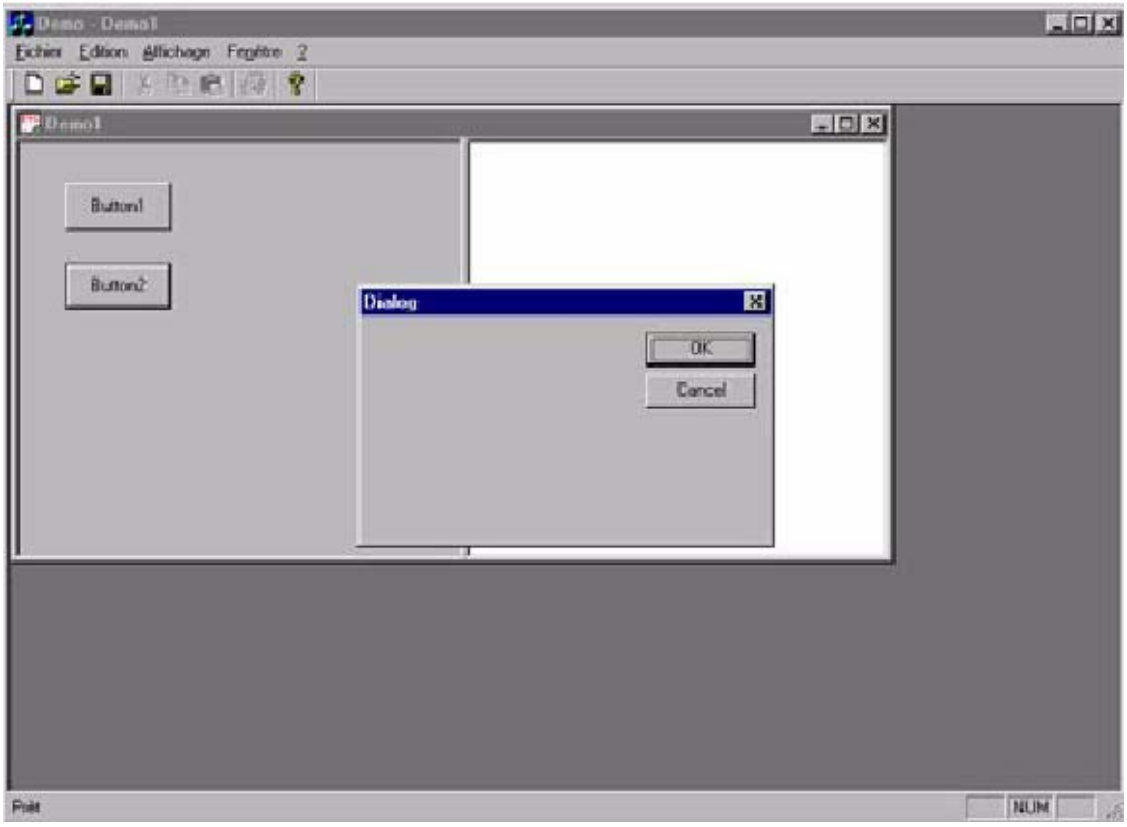


5. 3.- عرض مربع حوار :

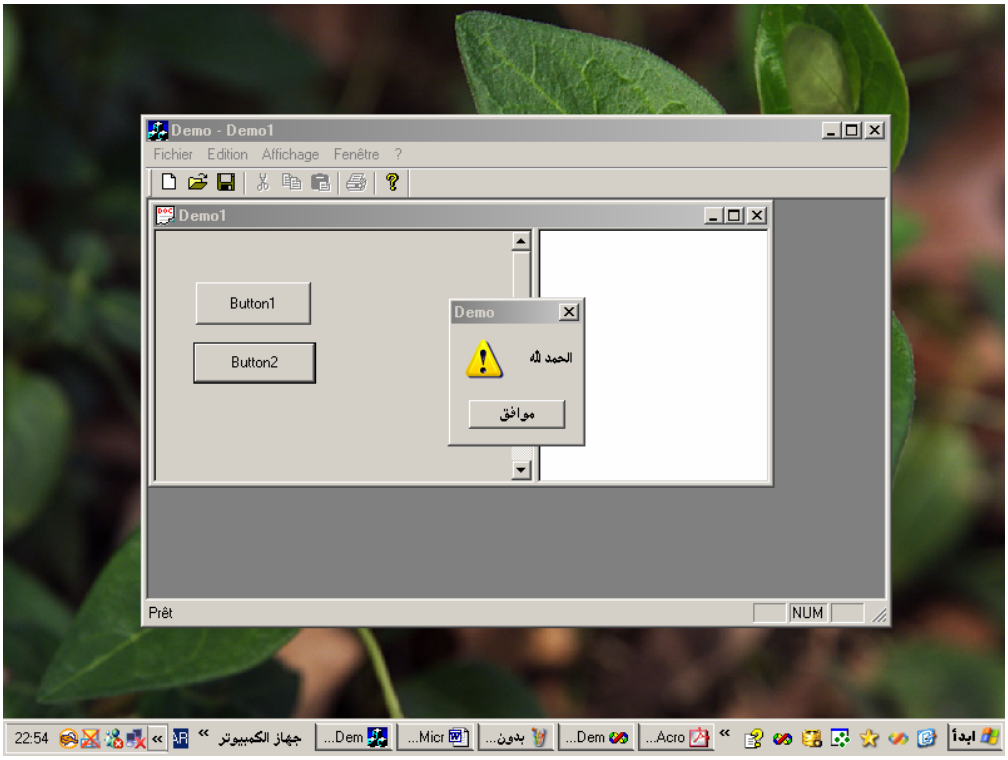
```
void CFormulaire::OnButton2()
{
        // TODO: Add your control notification handler code here
        // TODO: Add your control notification handler code here
        CDialogue *dial = new CDialogue ();
        if (dial->DoModal() == IDOK)
        AfxMessageBox ("الحمد لله");

}
```

الطريقة DoModal تعرض مربغ الحوار وتعيد الثابت IDOK إن نقر المستخدم على OK أو IDCANCEL إن نقر على CANCEL

إن نقر المستخدم على OK سيعرض البرنامج مربع رسالة



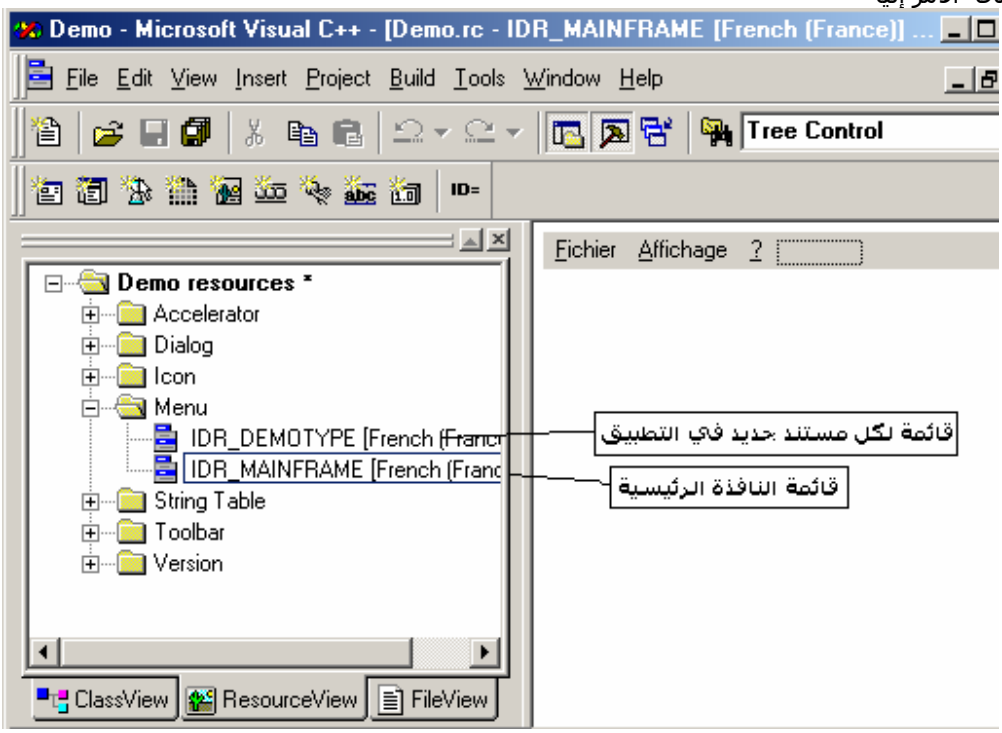
اقر على الزر Ok تظهر الرسالة التالية



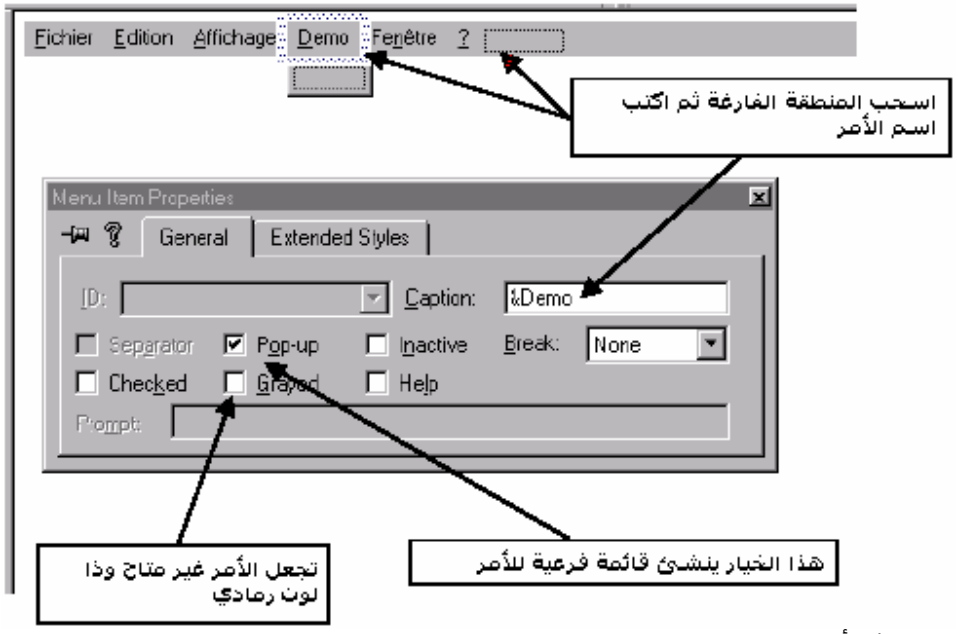
6.- إنشاء القوائم Menu :

6. 1.- إضافة قائمة إلى شريط القوائم:

حدد الشريحة « RessourceView » انقر علامة الجمع في العنصر « Menu » واختر الشريط الذي تريد إضافة الأمر إليه

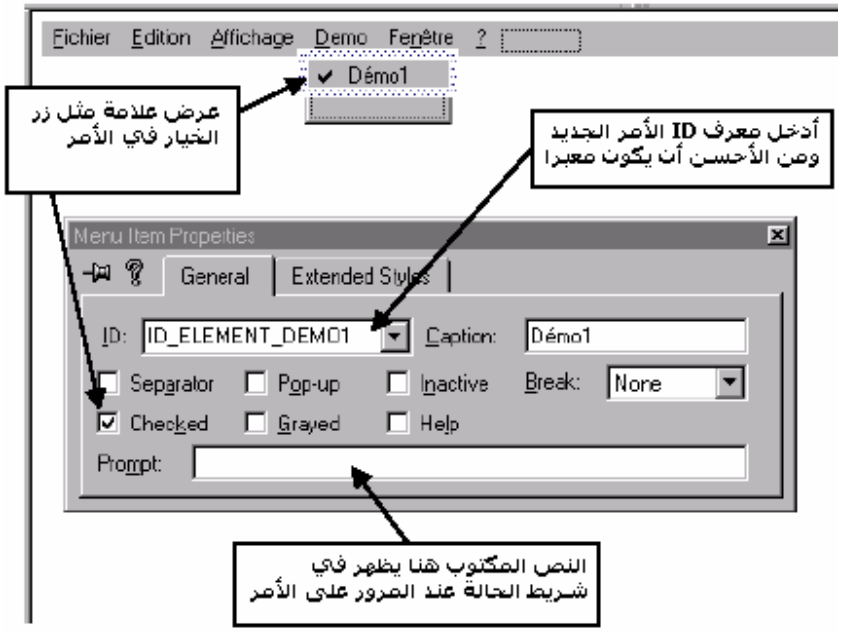


ولإضافة قائمة جديد اسحب المستطيل الفارغ إلى المكان المحدد واكتب اسم القائمة الحرف & ينشئ اختصار في لوحة المفاتيح وذلك بالضغط على Alt والحرف الذي بعد &



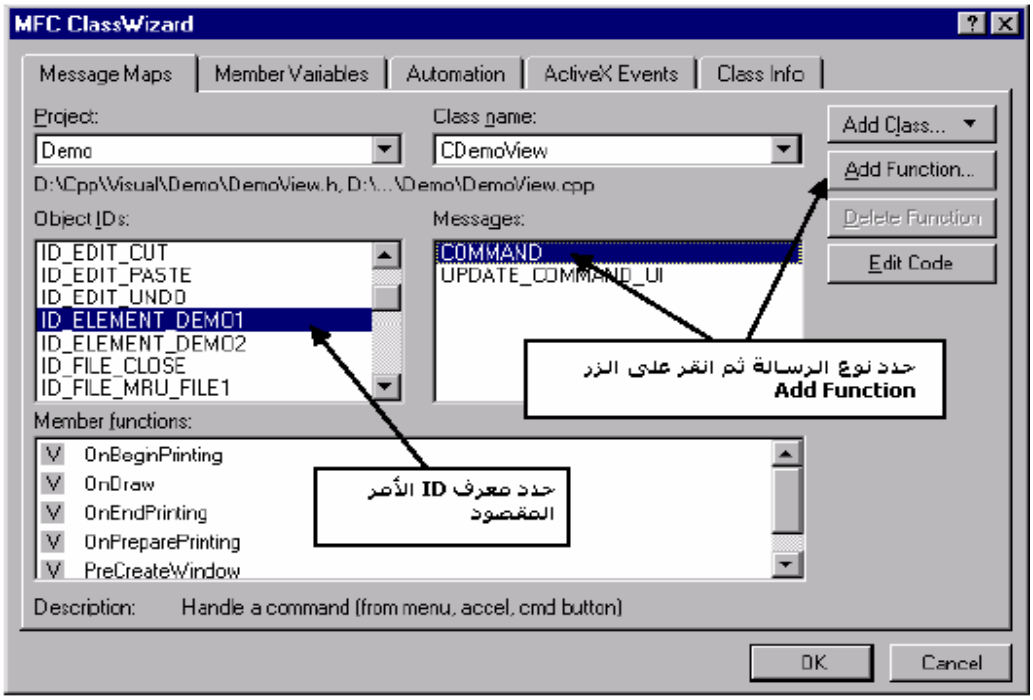
6. 2.- إضافة أمر إلى القائمة:

انقر في المنطقة الفارغة تحت القائمة Demo وأدرج أمرا جديدا بالخصائص التالية:



6. 3.- ربط الأمر برسالة (حدث) جديد بواسطة ClassWizard

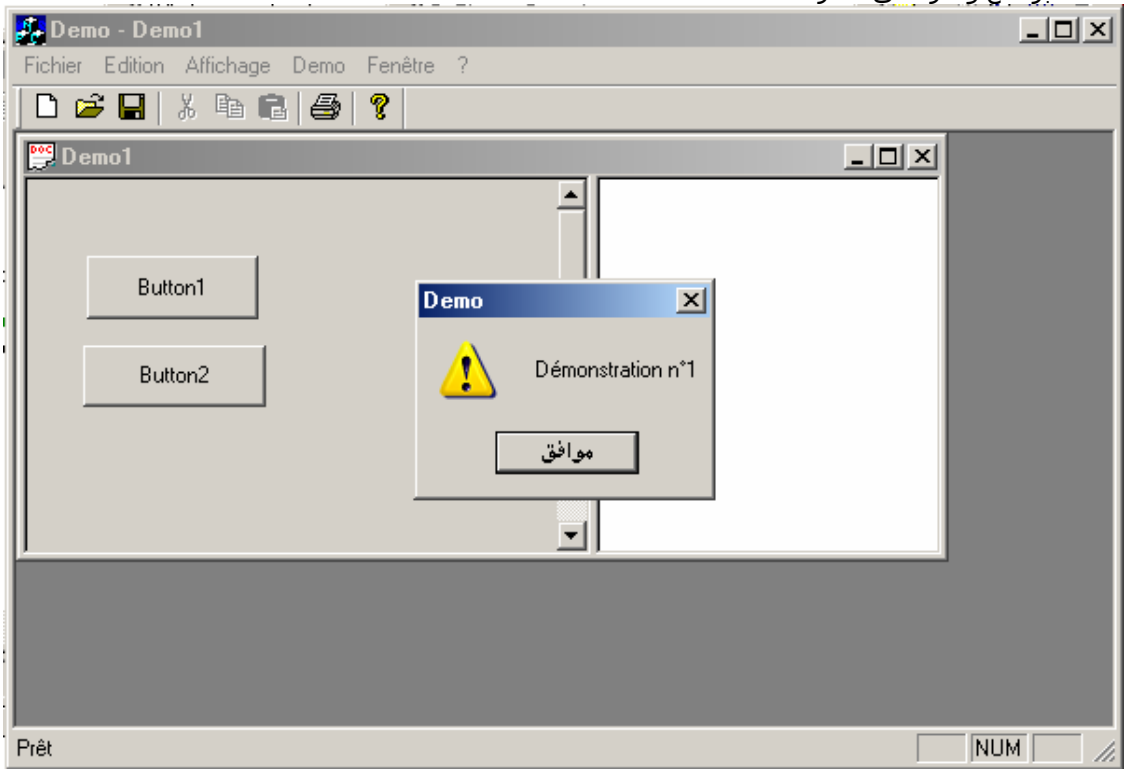
لتشغيل ClassWizard أنقر على الأمر بزر الماوس الأيمن واختر ClassWizard حدد معرف ID الأمر ونوع الرسالة ثم انقر Add Function ثم اكتب اسم الطريقة التي ستُتدعى في كل مرة يتم فيها تحديد الأمر الجديد



بعد كتابة اسم الرسالة انقر على Edit Code واكتب ما يلي:

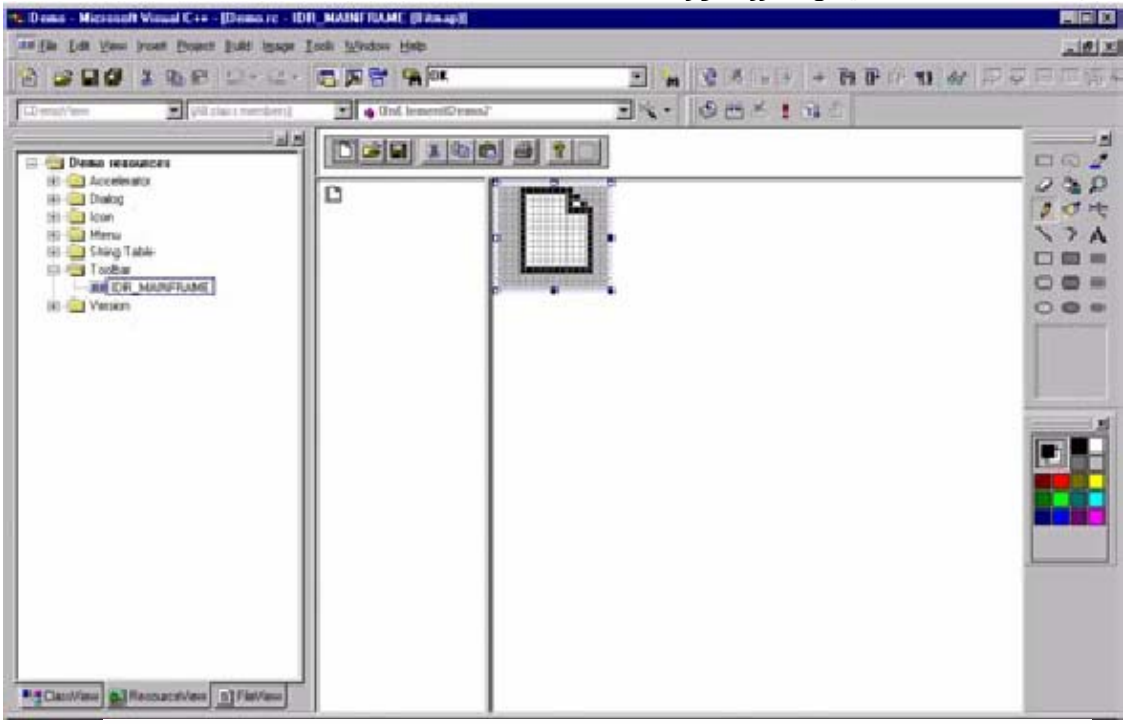
```
void CDemoView::OnElementDemo1()
{
	// TODO: Add your command handler code here
	AfxMessageBox ("Démonstration n°1") ;
}
```

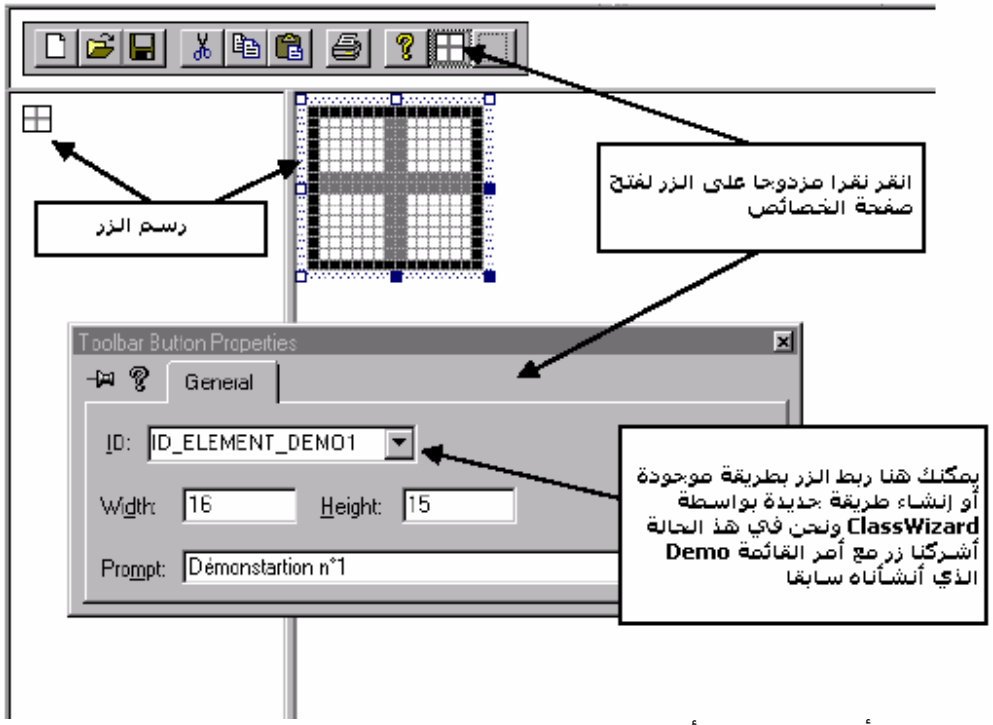
نف> البرنامج وانقر على الأمر :



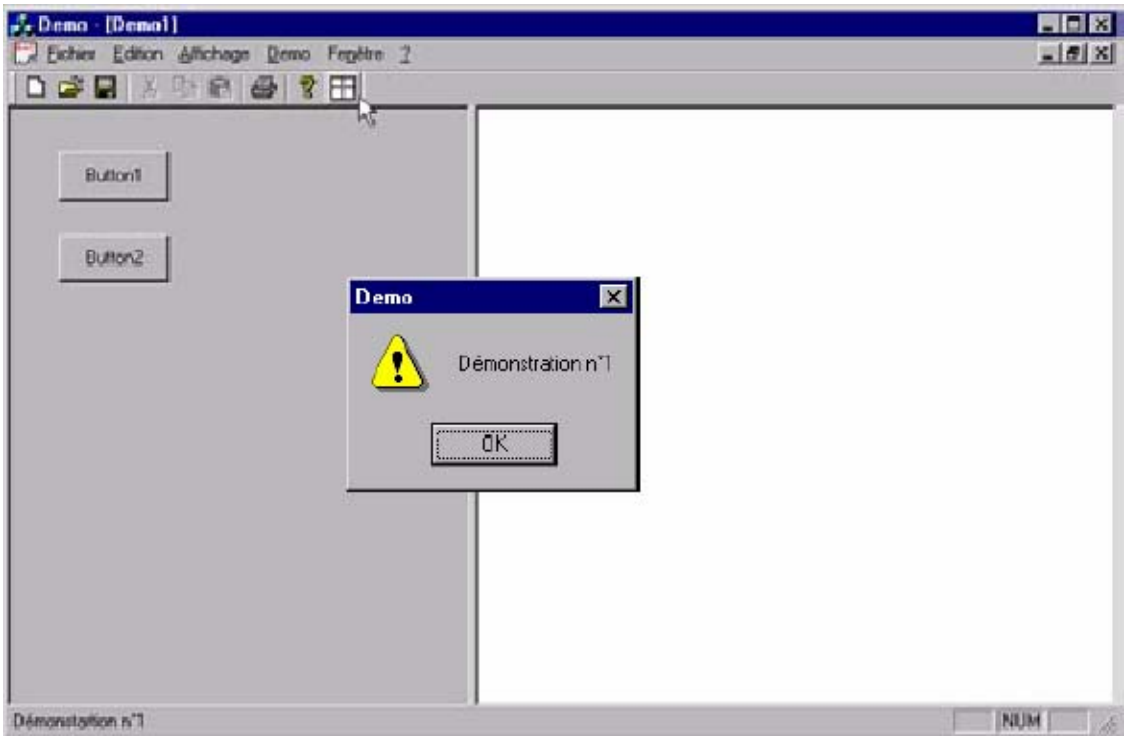
6. 4.- لإضافة زر إلى شريط الأدوات:

حدد الشريحة « RessourceView » وانقر علامة الجمع في العنصر « Toolbar » ثم انقر نقرا مزدوجا على IDR_MAINFRAME يفتح محرر الموارد:





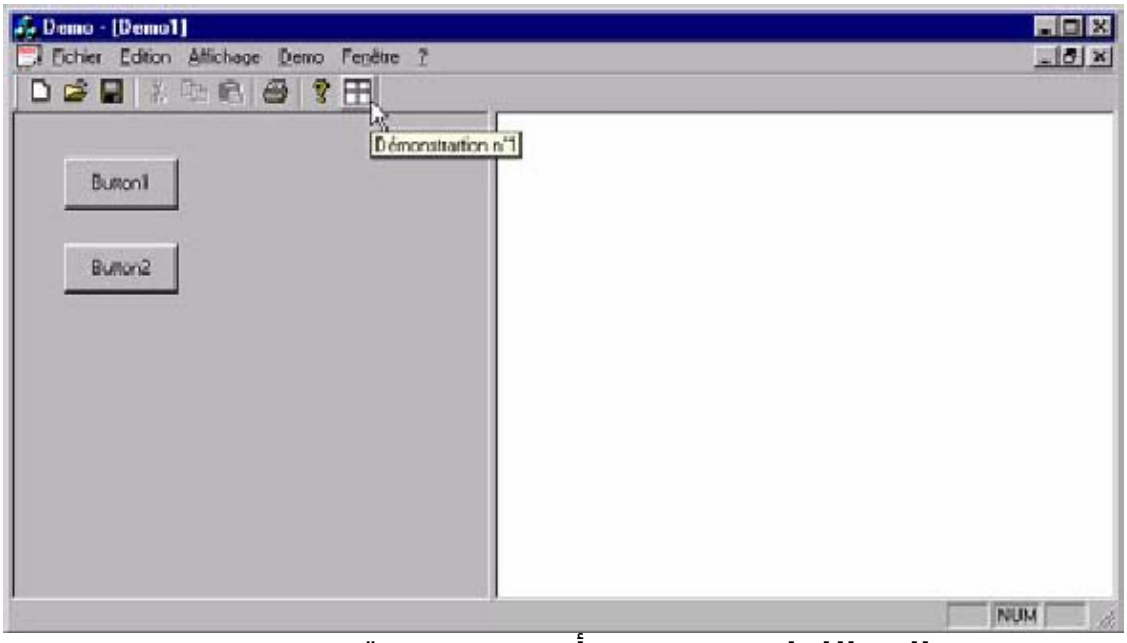
في مثالنا ه>ا أشركنا الزر مع الأمر Demo1 للقائمة Demo



6. 5.- إضافة التلميحات:

يمكنك إضافة تلميح لأزرار شريط الأدوات الذي يعرض عندما نبقي مؤشر الفأرة على الزر بعض الثواني لفعل ذلك حدد الشريحة « RessourceView » وافتح العنصر String Table وانقر مرتين على معرف ID الزر تعرض النافذة التالية:

والنتيجة:



وإلى اللقاء مع دروس أخرى متقدمة وهي:

- ✓ **تعدد الفواصل**
- ✓ **إنشاء صفحات الخصائص ذات الشرائح**
- ✓ **مربعات الخيار**
- ✓ **أزرار الخيار (راديو)**
- ✓ **مربع السرد ListBox**
- ✓ **مربع الكتابة والسرد**
- ✓ **مربع النص**

أرجو منكم الدعاء والسلام عليكم

المشروع مرفق مع الملف