الباب السادس

تحليل برنامج MFC

مفتتح

فى هذا الباب سوف نتجول فى عمق البرنامج "MFCHello" الذى أنشأناه فى الباب السابق باستخدام ساحر MFC ، بهدف تتبع تسلسل الأحداث ، وفهم الدور الذى تلعبه كل فصيلة وكل دالة فى الكود الذى كتبه لنا الساحر. والكود الذى يكتبه الساحر ليس سطوراً متتابعة من عبارات اللغة ، ولكنه بناء متكامل من الفصائل التى تتصل ببعضها البعض من خلال علاقات سابقة الإعداد. وبفهم هذه الفصائل والعلاقات بينها وبين بعضها ، يمكننا أن نضيف إلى البناء ونوجهه إلى التطبيق المقصود.

تذكر:

١. للاطلاع على الصيغة التفصيلية لإحدى الدوال أو الفصائل ، ضع مؤشر الفأر فوقها ثم اضغط زر اللوحة F1.

٢. لعرض الخريطة الكاملة لشجرة فصائل المؤسسة MFC ابحث فى شاشة النجدة (باستخدام البطاقة Search) عن "Hierarchy Chart".

٣. للاطلاع على الكود الكامل لأحد المشروعات ، افتح المشروع من القرص المصاحب للكتاب.

(٦-١) خصائص برامج MFC

قبل الدخول فى التفصيلات ، هناك ملاحظة هامة يجب الالتفات إليها عند قراءة برامج النوافذ عامة ، أو برامج MFC على وجه الخصوص. إن البرامج التى يكتبها الساحر فيها نوعين من الكود:

١. كود يختص بالعمليات التنظيمية التى تتكرر مع كل البرامج ، وتهدف إلى ربط برنامجك بنظام التشغيل وإكسابه الملامح العامة للنوافذ. وهذه الأجزاء تكتب غالبا باستخدام الماكرو (Macro) المميز بالحروف الكبيرة مثل DECARE_MESSGE_MAP. وفى أغلب الأحوال فإن هذا الكود لا يجوز تعديله (كما توصى التعليمات بداخل بلوكات البرنامج).

٢. كود موجه إلى خدمة التطبيق نفسه وهو يتكون من الدوال التى ترى بداخلها التعليق "// TODO" غالباً ، حيث تستطيع أن تضيف إلى هذه الدوال الكود اللازم لخدمة التطبيق المعين.

ولهذا السبب ، فلا تتوقع أن نشرح البرنامج عبارة بعبارة كما تعودنا مع لغات دوس أو برامج الكونصول. بل سنهتم فقط بالأجزاء التى تخدم التطبيق أو تؤثر عليه بصورة مباشرة. أما الأجزاء التنظيمية فسوف نتخطاها ، أو نعلق عليها فى حينه ، عندما يستلزم الأمر أن نفهم شيئاً عن خلفية الأحداث الدائرة.

(٦-٢) مجرى الأحداث فى البرنامج

أول ما نلاحظه فى برنامج MFC أنه لا يحتوى على الدالة الرئيسية **()WinMain** الممثلة لقلب البرنامج النوافذى. وهذه الدالة موجودة ، وتعمل فى الخلفية ، ولكنها محجوبة عن المبرمج. ونتوقع بذلك أن يكون تتابع الأحداث مختلفاً عن البرامج المكتوبة باستخدام دوال API مباشرة. وهذا هو السيناريو العام للأحداث فى برامج MFC:

١. يتم استدعاء دالة بناء فصيلة التطبيق **CWinApp**.

٢. تأخذ الدالة **()WinMain** (التابعة للمؤسسة MFC) دفة التحكم.

٣. يتم خلق وشحن هدف التطبيق باستخدام الدالة **()InitInstance**.

٤. تبدأ الدالة **()WinMain** الحلقة التكرارية لاستلام ومعالجة وتوزيع الرسائل.

٥. عندما يغلق المستخدم نافذة التطبيق ، ينتهى عمل الدالة **()WinMain** (يصطلح على ذلك بعودة الدالة) وينتهى البرنامج.

وفيما يلى نعرض تفصيلات هذا السيناريو من خلال دوال البرنلمج MFCHello ، بادئين بفصيلة التطبيق CMFHelloApp.

(٦-٣) هدف التطبيق The Application Object

يعتبر هدف التطبيق ممثلاً للتطبيق ككل ، وفصيلة التطبيق لمثالنا المطروح CMFHelloApp مشتقة من الفصيلة **CWinApp**. وهذا هو ملف الإعلان الذى يمنحك فكرة عامة عن أعضاء الفصيلة:

```
class CMFCHelloApp : public CWinApp
{
```

```
public:
        CMFCHelloApp();                 دالة البناء
// Overrides
        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CMFCHelloApp)
        public:
        virtual BOOL InitInstance();    دالة عضوة
        //}}AFX_VIRTUAL
// Implementation
        //{{AFX_MSG(CMFCHelloApp)
        afx_msg void OnAppAbout();      دالة عضوة
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};
```

دالة بناء فصيلة التطبيق CWinApp()

يحتوى الملف MFCHello.cpp على الدوال الثلاثة المستخدمة فى بنـاء وشحن التطبيق:

- CMFHelloApp()
- InitInstance()
- OnAppAbout()

وتقوم الدالة CMFHelloApp() ببناء هدف التطبيق باستخدام دالة البناء سابقة التعريف لفصيلة الأساس **CWinApp()** (ولذلك جاء جسم الدالة خاليا من المحتويات). يلى ذلك إعلان هدف واحد فقـــط للتطبيـق بالاسم **theApp** وهو عبارة عن هدف من الفصيلة CMFCHelloApp ، كما نرى فى شريحة البرنامج التالى:

```
CMFCHelloApp::CMFCHelloApp()
{
```

```
        // TODO: add construction code here,
        // Place all significant initialization in InitInstance
                دالة البناء خالية من المحتويات ، حيث تستخدم دالة البناء للفصيلة الأم.
}
// The one and only CMFCHelloApp object
CMFCHelloApp theApp;                    إعلان هدف التطبيق الأوحد
```

وكما نرى أن هدف التطبيق قد تم الإعلان عنه فى النطـــاق العـــام للملف بحيث يصبح هدفا عاما (Global) يمكن التوصل إليه مـــن أى دالة من دوال MFC.

ملاحظة:

إن معظم دوال الفصيلة CWinApp دوال افتراضية يمكن ركوبها من داخـــل التطبيق.

ملاحظة:

دالة شحن الهدف InitInstance()

بالرغم من أن الدالة الرئيسية **WinMain()** محجوبة عن المـــبرمج ، لكنه يتم ربطها بالبرنامج أثناء عمليتى البناء. وهى تقوم باســـتدعاء الدالة **InitInstance()** من فصيلة التطبيق لشحن هـــدف التطبيـــق بالقيم الابتدائية ، وبدء حلقة الرسائل ، وتستمر الدالة فى العمل حتى ينتهى التطبيق. وفيما يلى نص الدالة **InitInstance()**:

```
BOOL CMFCHelloApp::InitInstance()
{
// إعداد عناصر التحكم بالنافذة
        AfxEnableControlContainer();
#ifdef _AFXDLL
```

```
        Enable3dControls();
#else
        Enable3dControlsStatic();
#endif
// إجراءات التسجيل
        SetRegistryKey(_T("Local AppWizard-Generated
Applications"));
        LoadStdProfileSettings();
//                          خلق هدف الوثيقة
        CSingleDocTemplate* pDocTemplate;
        pDocTemplate = new CSingleDocTemplate(
                IDR_MAINFRAME,
                RUNTIME_CLASS(CMFCHelloDoc),
                RUNTIME_CLASS(CMainFrame),
                RUNTIME_CLASS(CMFCHelloView));
        AddDocTemplate(pDocTemplate);
// إجراءات خط الأوامر
        CCommandLineInfo cmdInfo;
        ParseCommandLine(cmdInfo);
        if (!ProcessShellCommand(cmdInfo))
                return FALSE;
//                          عرض وتحديث النافذة
        m_pMainWnd->ShowWindow(SW_SHOW);
        m_pMainWnd->UpdateWindow();
        return TRUE;
}
```

وكما نرى فى شريحة الكـــود الســـابقة أن الدالـــة **InitInstance()** تحتوى على بعض الإجراءات التحضيرية والتى أسبقناها بتعليقـــات مختصرة ، أما سطور الكود التى تمثل قلب الدالة فقد ميزناها بالبنط الثقيل ، وهى تنقسم إلى جزئين:

١. خلق نموذج الوثيقة (Document Template).

٢. عرض وتحديث نافذة التطبيق.

نموذج الوثيقة

أما نموذج الوثيقة فهو عبارة عن هدف يختزن معلومات هامة عن الفصائل المختلفة والموارد التى تستخدم فى خلق كل من أهداف الوثيقة والمشهد والإطار العام. ويتم إعلان نموذج الوثيقة كمؤشر إلى هدف من الفصيلة **CSingleDocTemplate** كالآتى:

```
CSingleDocTemplate* pDocTemplate
```

ثم يتم تخصيص حيز من الذاكرة لهذا الهدف بالاستعانة بالمؤثر new ودالة بناء الفصيلة **CSingleDocTemplate** التى تستخدم أربعة بارامترات كالآتى:

```
pDocTemplate = new CSingleDocTemplate
                    (
                IDR_MAINFRAME,
                RUNTIME_CLASS(CMFCHelloDoc),
                RUNTIME_CLASS(CMainFrame),
                RUNTIME_CLASS(CMFCHelloView)
        );
```

أما البارامتر الأول IDR_MAINFRAME فهو عبارة عن رقم التعارف (أو الهوية) لموارد البرنامج. وأما البارامترات الثلاثة التالية فتحتوى على معلومات فصيلة الوثيقة (CMFCHelloDoc) والإطار (CMainFrame) والمشهد (CMFCHelloView). ويتم الحصول على المعلومات عن هذه الفصائل بالاستعانة بالماكرو RUNTIME_CLASS الذى يستخدم اسم الفصيلة كبارامتر.

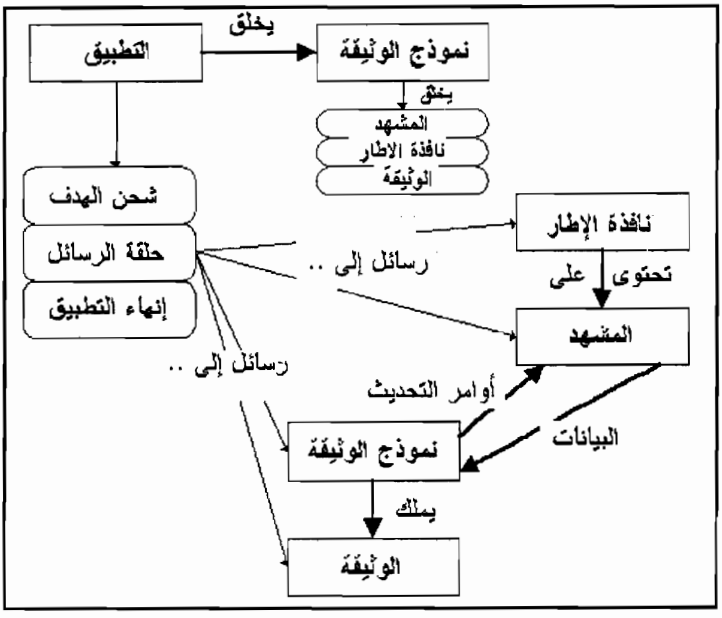
بعد ذلك يتم تخزين نموذج الوثيقة بداخل هدف الوثيقة بحيث يصبح النموذج متاحا عند فتح الوثيقة. ويتم ذلك باستخدام الدالة **AddDocTemplate()** أحد أعضاء فصيلة التطبيق **CWinApp** كالآتى:

عرض وتحديث النافذة

كما نرى فى شريحة برنامج الدالة **InitInstance()** أنها تختتم باستدعاء كل من الدالتين **ShowWindow()** ، **UpdateWindow()** ، حيث تتولى الدالة الأولى إظهار نافذة التطبيق على الشاشة ، بينما تتولى الثانية عرض محتويات النافذة لأول مرة. ويتم استدعاء هذه الدوال بالاستعانة بمؤشر إلى نافذة الإطار العام **m_pMainWnd** (العضو بالفصيلة **CWinThread**):

```
m_pMainWnd->ShowWindow(SW_SHOW);
m_pMainWnd->UpdateWindow();
```

ويوضح التشكل التالى المعالم الرئيسية لأحداث وأهداف التطبيق النوافذى باستخدام MFC.



شكل (٦-٢) الأحداث الرئيسية وأهداف التطبيق

وظائف أخرى لدالة شحن الهدف

تعتبر الدالة **InitInstance()** هى الدالة المناسبة لإعلان المتغيرات العامة فى برنامجك ، كما أنها قابلة للتعديل فى حالة إذا ما أردت الاستغناء عن الوثيقة والمشهد سابقى التعريف ، وفضلت استخدام وثيقة أو مشهدا خاصا ببرنامجك.

وفيما يلى نلخص الإجراءات التنظيمية التى تحتوى عليها الدالة.

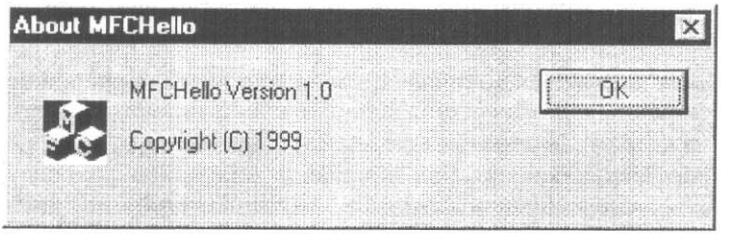
الدالة/الدوال	الوظيفة
Enable3dControls() Enable3dControlsStatic()	تمنح النوافذ والأزرار مظهرا مجسما وهى غير ضرورية مع النوافذ التالية للطراز NT3.51
SetRegistryKey() LoadStdProfileSettings()	لتسجيل اختيارات البرنامج (أو تحميلها) فى سجل النوافذ (Windows Registry).
ParseCommandLine() ProcessShellCommand()	تستخدم الدالة الأولى فى تحليل الأمر فى حالة تشغيل البرنامج فى بيئة خط الأوامر (دوس). وتستخدم الدالة الثانية فى تنفيذ الأمر.

جدول (٦-١) دالة شحن الهدف InitInstance()

دالة المعلومات عن التطبيق OnAppAbout

تؤدى هذه الدالة دورا بسيطا ولكنها تستحق وقفة. إن هذه الدالة تؤدى إلى ظهور النافذة الموضحة بالشكل التالى عندما تستخدم أمر

القائمة Help - About MFCHello ، أو تضغط على الزر المميز بعلامة الاستفهام (؟) بسطر الأدوات.



شكل (٦-٣) صندوق المعلومات عن التطبيق (About)

وهذه هى الدالة نفسها:

```
void CMFCHelloApp::OnAppAbout()
{
	CAboutDlg aboutDlg;
	aboutDlg.DoModal();
}
```

إن الدالة فى مجملها تمثل مقبضا لرسالة ، فهى عبارة عن استجابة لأحد اختيارات القائمة. وهى تحتوى على عبارتين ، الأولى لإعلان هدف صندوق حوار من الفصيلة **CAboutDlg**:

```
CAboutDlg aboutDlg;
```

والثانية لاستدعاء الدالة **DoModal()** العضوة بالفصيلة الأساسية **CDialog**:

```
aboutDlg.DoModal();
```

أما وظيفة الدالة **DoModal()** فهى تنحصر فى عرض الصندوق والاستجابة لرد فعل المستخدم وهو الضغط على الزر OK فى هذه الحالة. والصندوق المستخدم هنا يسمى بالصندوق النمطى (Modal).

يوجد نوعان من صناديق الحوار ، **الصندوق النمطى** (Modal) **والصندوق غير النمطى** (Modeless). والنوع الأول يحتل بؤرة التركيز بحيث لا يستطيع المستخدم أن ينتقل إلى نشاط آخر فى التطبيق قبل الاستجابة للصندوق وإغلاقه. أما النوع غير النمطى فيمكنك تركه مفتوحا والانتقال إلى نافذة التطبيق الرئيسية.

ويوجد إعلان الفصيلة CAboutDlg بملف الكود MFCHello.cpp وهو موضح بشريحة البرنامج التالية:

```
class CAboutDlg : public CDialog
{
public:
  CAboutDlg();

// Dialog Data
        //{{AFX_DATA(CAboutDlg)
        enum { IDD = IDD_ABOUTBOX };
        //}}AFX_DATA
        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CAboutDlg)
        protected:
        virtual void DoDataExchange(CDataExchange* pDX);
  //}}AFX_VIRTUAL
// Implementation
protected:
        //{{AFX_MSG(CAboutDlg)
                // No message handlers
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};
```

والأجزاء الهامة هنا هى:

١.استدعاء الدالة **DoModal()** التابعة للفصيلة **CDialog**.

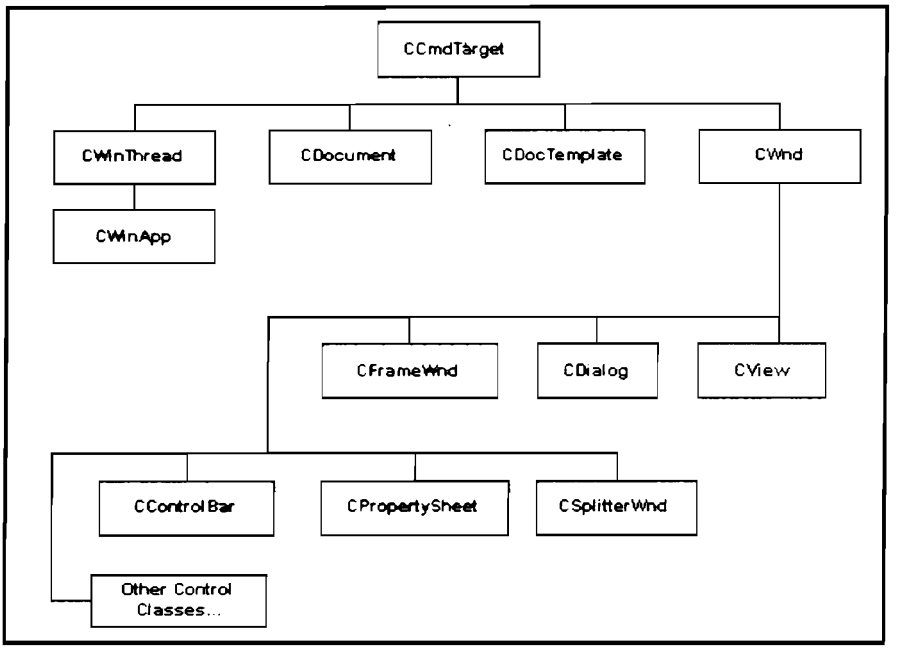
٢. ركوب الدالة الافتراضية **DoDataExchange()** (سيلى الحديث عن الدور الذى تؤديه هذه الدالة فى الفقرات القادمة).

خريطة الرسائل (Message Map)

سنناقش فى هذه الفقرة الأحداث المحركة لعملية الاستجابة لهذه الرسالة والتى تؤدى إلى عرض نافذة المعلومات About.

١. عندما تضغط على اختيار القائمة Help-About فإن النوافذ ترسل إلى التطبيق الرسالة **WM_CMMAND** التى تحدد الأمر الذى تم اختياره بالاستعانة برقم التعارف (ID). وفى هذا المثال فإن رقم التعارف هو ID_APP_ABOUT (وهو موجود بجدول الحرفيات). وعلى هذا الأساس يتم استلام وتوزيع الرسالة بداخل التطبيق.

٢. والرسالة WM_CMMAND (وتسمى **رسالة الأوامر**) تتميز بأهمية خاصة ، لأنها بخلاف الرسائل الأخرى ، لا يتم توزيعها على النوافذ فقط ، بل على هدف التطبيق وما يتبعه من أهداف النوافذ والمشهد والوثيقة. ويطلق على هذه المجموعة من الأهداف اسم أهداف الأوامر (Command Targets) ، وهى تنحدر جميعا من الفصيلة **CCmdTarget**. وفى الشكل التالى نعرض جانبا هاما من شجرة فصائل المؤسسة MFC وهو الجانب المحتوى على الفصيلة **CCmdTarget** والفصائل التى تتبعها بما فى ذلك الفصيلة **CWnd** التى تنحدر منها النوافذ ومشتقاتها.



شكل (٦-٤) الفصائل المشتقة من CCmdTarget

٣.عندما يتلقى أحد الأهداف رسالة أوامر (WM_COMMAND) يتولى مقبض الرسالة (الدالة **OnCmdMsg** الموروثة من الفصيلة **CCmdTarget**) فحص الرسالة لمعرفة إذا ما كان جاهزا للاستجابة لها أم لا. فإن لم يكن ، فيتم إرسالها إلى هدف آخر من أهداف الأوامر وهكذا. أما كيفية فحص الرسالة فيتم باستخدام جدول خاص بمؤسسة الفصائل MFC ، يسمى خريطة الرسائل (Message Map) ، ويربط ما بين كل رسالة وما بين الدالة المخصصة لمعالجتها (المقبض). ويستخدم هذا الجدول فى استدعاء الدالة المناسبة لكل رسالة.

٤. فى شريحة الكود التالية نرى خريطة الرسائل لهدف التطبيق ، والواردة فى بداية الملف MFCHello.cpp. ولأن هدفى التطبيق والوثيقة ليسا بنوافذ ، فهما يستجيبان فقط لرسائل الأوامر (WM_COMMAND).

```
// CMFCHelloApp
BEGIN_MESSAGE_MAP(CMFCHelloApp, CWinApp)
  //{{AFX_MSG_MAP(CMFCHelloApp)
  ON_COMMAND(ID_APP_ABOUT, OnAppAbout)
    // NOTE - the ClassWizard will add and remove mapping macros here.
    //    DO NOT EDIT what you see in these blocks of generated code!
  //}}AFX_MSG_MAP
  // Standard file based document commands
  ON_COMMAND(ID_FILE_NEW, CWinApp::OnFileNew)
  ON_COMMAND(ID_FILE_OPEN, CWinApp::OnFileOpen)
  // Standard print setup command
  ON_COMMAND(ID_FILE_PRINT_SETUP, CWinApp::OnFilePrintSetup)
END_MESSAGE_MAP()
```

وتبدأ خريطة الرسائل بالعبارة BEGIN_MESSAGE_MAP وتنتهى بالعبارة END_MESSAGE_MAP. وتحتوى بداخلها على أوامر ماكرو تبدأ بالكلمة ON_COMMAND (وهو الماكرو المختص بالرسائل WM_COMMAND). ويختص كل ماكرو برسالة معينة ، ويحدد الدالة التى تستجيب لها كالمثال الآتى:

```
ON_COMMAND(ID_APP_ABOUT, OnAppAbout)
```

إن هذا الماكرو يحدد التناظر بين الرسالة ذات رقم التعارف ID_APP_ABOUT وبين المقبض OnAppAbout.

(٦-٤) هدف نافذة الإطار MainFrame Window Object

إن هذا الهدف يمثل الإطار الخارجى لنافذة التطبيق شاملا سطر العنوان (Title bar) والقائمة (Menu bar) وسطر الأدوات (Toolbar) وسطر الحالة (Status bar). أما نافذة المشهد التى تحتوى على العبارة "Hello my MFC friends" (منطقة العميل) فهى نافذة ابنة لنافذة الإطار.

وعندما تبدأ من شجرة الفصائل ، وتضغط على اسم الفصيلة CMainFrame فإنك تشاهد ملف الإعلانات للفصيلة "MainFrm.h" ، وترى به لإعلان الفصيلة CMainFrame الموروثة من الفصيلة **CFrameWnd**. ولو تتبعت شجرة الفصائل الموضحة بالشكل السابق ، فإنك ترى أن هذه الفصيلة ترث العديد من فصائل المؤسسة ، فالفصيلة **CFrameWnd** ترث **CWnd** التى ترث بدورها **CCmdTarget** التى ترث الفصيلة الأم **CObject** وهى الفصيلة التى تأتى منها أغلب فصائل المؤسسة. ومعنى ذلك أن هذه الفصيلة غنية بالبيانات والدوال الأعضاء الموروثة من جميع هذه الفصائل.

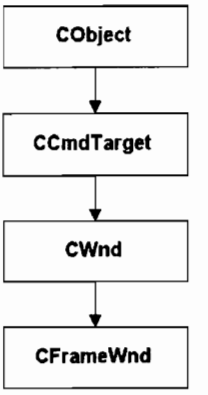
والفارق الأساسى بين التطبيق متعدد الوثائق (MDI) وبين هذا التطبيق (SDI) أن فصيلة نافذة الإطار بالنوع الأول ترث من الفصيلة **CMDIFrameWnd** وليس من **CFrameWnd**.

من الأجزاء الهامة بهذا الملف أيضا إعلان الأعضاء المحمية الممثلة لسطر الحالة (Status bar) وسطر الأدوات (Toolbar) كالآتى:

```
protected:  // control bar embedded members
        CStatusBar  m_wndStatusBar;
        CToolBar    m_wndToolBar;
```

شجرة الوراثة

يوضح الشكل التالى الفصيلة **CFrameWnd** وموقعها فـــى شـــجرة الوراثة.



شكل (٦-٥) فصيلة نافذة الإطار بشجرة الوراثة

خريطة الرسائل

بفتح ملف التطبيق "MainFrm.cpp" فإننا نرى فـــى بدايتـــه خريطـــة الرسائل التى تعتمد على ماكرو واحـــد ON_WM_CREATE ونصـــها كالآتى:

```
BEGIN_MESSAGE_MAP(CMainFrame, CFrameWnd)
  //{{AFX_MSG_MAP(CMainFrame)
    // NOTE - the ClassWizard will add and remove mapping macros here.
    //    DO NOT EDIT what you see in these blocks of generated code !
  ON_WM_CREATE()
  //}}AFX_MSG_MAP
END_MESSAGE_MAP()
```

والرسالة التى يتم معالجتها هنا هى رسالة النوافذ WM_CREATE ويتم التعامل معها بواسطة مقبض الدالة **OnCreate()** الموروثة من الفصيلة **CWnd**.

الدالة OnCreate()

توجد الدالة بملف التطبيق "MainFrm.cpp" أيضا ، وهى تمثل قلب فصيلة نافذة الإطار. وهذا هو نصها:

```
int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
        if (CFrameWnd::OnCreate(lpCreateStruct) == -1)
                return -1;
سطر الأدوات
        if (!m_wndToolBar.CreateEx(this, TBSTYLE_FLAT, WS_CHILD
            | WS_VISIBLE | CBRS_TOP | CBRS_GRIPPER
            | CBRS_TOOLTIPS
            | CBRS_FLYBY | CBRS_SIZE_DYNAMIC)
            || !m_wndToolBar.LoadToolBar(IDR_MAINFRAME))
        {
                TRACE0("Failed to create toolbar\n");
                return -1;      // fail to create
        }
سطر الحالة
        if (!m_wndStatusBar.Create(this) ||
           !m_wndStatusBar.SetIndicators(indicators,
           sizeof(indicators)/sizeof(UINT)))
        {
                TRACE0("Failed to create status bar\n");
                return -1;      // fail to create
        }
خاصية الحوض لسطر الأدوات
        // TODO: Delete these three lines if you don't want the toolbar to
        //  be dockable
        m_wndToolBar.EnableDocking(CBRS_ALIGN_ANY);
        EnableDocking(CBRS_ALIGN_ANY);
        DockControlBar(&m_wndToolBar);

        return 0;
}
```

وكما نرى فى الدالة أن هناك بعض السطور الحاكمة وقد كتبناها باستخدام البنط الثقيل. أما الخطوط الرئيسية لمنطق الدالة فهى:

١. لو أنك حذفت جميع محتويات هذه الدالة فيما عدا العبارة "return 0" الموجودة فى نهاية الدالة ، فإن الترجمة سوف تتم بنجاح وسوف تحصل على نافذة التطبيق فيما عدا سطر الأدوات وسطر الحالة وبالطبع فإنك لن ترى العبارة "Hello my MFC friends"! معنى هذا أن الوراثة تلعب الدور الرئيسى هنا.

٢. لو أنك أبقيت على العبارة الأولى فقط فى الدالة ، وهى:

```
return -1; if (CFrameWnd::OnCreate(lpCreateStruct) == -1)
```

فإنك لا تزال ترى العبارة "Hello my MFC friends" على الشاشة. إن هذه العبارة هى المسئولة عن محتويات النافذة.

٣. أما بقية عبارات الدالة فهى مخصصة لخلق سطر الأدوات وسطر الحالة باستخدام الأعضاء m_wndStatusBar; (هدف من الفصيلة **CStatusBar**) و m_wndToolBar (هدف من الفصيلة **CToolBar**) ، وهذه الأهداف معلنة فى ملف العناوين كما قدمنا. وكما نرى فى هذه العبارات أن الماكرو **TRACE0** قد استخدم هنا لتشخيص الأعطال ، حيث يطبع الرسالة الموجودة بين القوسين فى بطاقة الصيانة (Debug Tab) عند تحقق العبارة الشرطية.

٤. أما عملية خلق سطر الأدوات فهى تتم باستخدام الدالة CreateEx() ، ويتم تحميله باستخدام الدالة LoadToolBar() ، وهما من دوال الفصيلة CToolBar ، كالآتى (حذفنا بعض الشروط للتبسيط):

```
if (!m_wndToolBar.CreateEx(
....
  || !m_wndToolBar.LoadToolBar(IDR_MAINFRAME))
{ ...
```

أما الثوابت المفصولة بالمؤثرات | أو || فهى تحدد خصائص سطر الأدوات (Styles) ، وسوف يلى شرحها فى الأبواب القادمة.

٥. تتم عملية خلق وتحميل سطر الحالة بطريقة مماثلة ، حيث تستخدم الدالتان **Create()** و **SetIndictors()** ، وهما من دوال الفصيلة **CStatusBar**:

```
if (!m_wndStatusBar.Create(this) ||
    !m_wndStatusBar.SetIndicators(...
```

خاصية الحوض لسطر الأدوات (Toolbar Docking)

أما السطور الثلاثة الأخيرة فى الدالة **OnCreate()** فهى تؤدى إلى إكساب سطر الأدوات خاصية الحوض (Docking) بحيث يمكنك نقل السطر بما يحتويه ــ بالسحب و الإسقاط ــ إلى أى موقع على الشاشة:

```
m_wndToolBar.EnableDocking(CBRS_ALIGN_ANY);
EnableDocking(CBRS_ALIGN_ANY);
DockControlBar(&m_wndToolBar);
```

كما يمكنك تعديل أبعاد الحوض الذى يحتوى على الأدوات كما بالشكل التالى. فإذا حذفت هذه السطور الثلاثة ، أصبح سطر الأدوات أفقيا فى مكانه المعروف ويسمى فى هذه الحالة بالسطر الطافى (Floating). ونلاحظ استخدام الدالة **EnableDocking()** مرتين ، الأولى مع سطر الأدوات (وفى هذه الحالة تستدعى من الفصيلة **CControlBar**) والثانية مع نافذة الإطار (وفى هذه الحالة

يحتوى سطر الحالة على ثلاثة صناديق صغيرة على اليمين تسمى المبينات (Indicators) وهى تعكس حالة الأزرار الثلاثة Caps Lock و Scroll Lock و Num Lock كما هو كموضح بالشكل السابق. وتحفظ أرقام التعارف لهذه الأزرار فى المصفوفة Indicators كالآتى:

```
static UINT indicators[] =
{
        ID_SEPARATOR,           // status line indicator
        ID_INDICATOR_CAPS,
        ID_INDICATOR_NUM,
        ID_INDICATOR_SCRL,
};
```

والنمط المستخدم لعناصر هذه المصفوفة هو النمط **UINT** وهو عبارة عن نمط صحيح طويل (٣٢ بت). وتقع هذه العبارة تالية لخريطة الرسائل مباشرة بالبرنامج MainFrm.cpp. وتستخدم هذه المجموعة من أرقام التعارف بواسطة الدالة **SetIndicators()** عند إنشاء سطر الحالة ، كما نرى فى الدالة **OnCreate()**.

دوال أخرى لنافذة الإطار

أغفلنا فى هذه المناقشة بعض الدوال التى يكتبها الساحر ولا تؤثر مباشرة على التطبيق. وهذا هو ملخص سريع بوظائفها:

الدالة	الوظيفة
PreCreateWindow()	لتغيير خصائص نافذة الإطار حسب الطلب. وهى دالة افتراضية من دوال الفصيلة **CWnd**.

AssertValid()	دالة صيانة ، يجوز توظيفها للتحذير من أخطاء معينة عند بناء هدف النافذة. وهى دالة افتراضية تابعة للفصيلة الأم **CObject**.
Dump()	دالة صيانة تستخدم فى طباعة معلومات عن الأهداف فى بطاقة الصيانة (Debug Tab) ، وهى دالة افتراضية من الفصيلة الأم **CObject**.

جدول (٦-٢) دوال أخرى بنافذة الإطار

(٦-٥) هدف الوثيقة The Document Object

استخدمنا من قبل فصيلة الوثيقة CMFCHelloDoc بالملف CMFCHelloDoc.cpp لتخزين المتغير الحرفى m_strHelloText وشحنه بالقيمة "Hello my MFC friends!". كما استخدمنا فصيلة المشهد CMFCHelloView لعرض هذا لحرفى على الشاشة. وتسمى هذه الطريقة بالفصل ما بين البيانات ومشهد البيانات. فالوثيقة تستخدم بصفة عامة فى تخزين البيانات بصرف النظر عن طريقة عرضها. أما المشهد فيستمد بياناته من الوثيقة ويتولى عرضها على الشاشة بصرف النظر عن نوعها ، سواء كانت رسما أو نصا أو جولا إليكترونيا. وهذا الفصل ما بين المشهد والوثيقة يمكنك من عرض نفس البيانات بصور متعددة. وعلى سبيل المثال فإن برامج الجداول الإليكترونية (مثل إكسيل) تمكنك من عرض نفس البيانات فى صورة جدول أو فى صورة خرائط بيانية متنوعة.

أما البيانات التى خزناها فى الوثيقة فى المثال MFCHello فقد أضفناها إلى دالة البناء كالآتى:

```
CMFCHelloDoc::CMFCHelloDoc()
{
        // TODO: add one-time construction code here
        m_strHelloText = "Hello my MFC friends!";
}
```

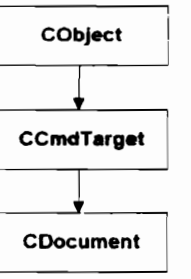
خريطة الرسائل

ونلاحظ فى هذه الفصيلة أن خريطة الرسائل قد جاءت خالية من المحتويات. وسوف نضيف إليها بعض سطور الكود فى الأمثلة القادمة.

```
BEGIN_MESSAGE_MAP(CMFCHelloDoc, CDocument)
  //{{AFX_MSG_MAP(CMFCHelloDoc)
    // NOTE - the ClassWizard will add and remove mapping macros here.
    //    DO NOT EDIT what you see in these blocks of generated code!
  //}}AFX_MSG_MAP
END_MESSAGE_MAP()
```

شجرة الوراثة

أما الشكل التالى فيوضح فصيلة الوثيقة فى شجرة الوراثة من فصائل المؤسسة MFC.



شكل (٦-٧) فصيلة الوثيقة فى شجرة الوراثة

دوال أخرى بفصيلة الوثيقة

أما بقية الدوال التى تحتوى عليها فصيلة المشهد فهى إما خالية (يمكن توظيفها فى تطبيقات أخرى قادمة) أو تختص بالإجراءات التنظيمية العامة ، وهذا ملخصها:

الدالة/الدوال	الوظيفة
CMFCHelloDoc() ~CMFCHelloDoc()	البناء والهدم. لاحظ أن المتغير الحرفى m_strHelloText الذى أضناه إلى دالة البناء يظهر فى قسم الشجرة كعضو من أعضاء الفصيلة.
AssertValid() Dump()	دوال صيانة (راجع جدول فصيلة نافذة الإطار ــ جدول (٦-٢)).
Serialize()	تستخدم هذه الدالة فى عملية القراءة أو الكتابة من/فى الملفات. وسيلى توظيفها فى الأمثلة القادمة.
OnNewDocument()	يمكنك إضافة عبارات شحن الأهداف من النوع SDI بهذه الدالة.

جدول (٦-٣) دوال أخرى بفصيلة الوثيقة

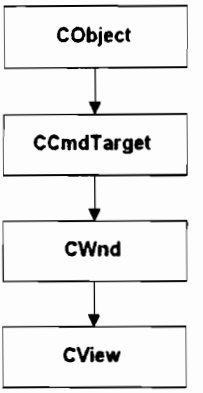
(٦-٦) هدف المشهد The View Object

كما ذكرنا من قبل أن المشهد عبارة عن نافذة تغطى منطقة العميل فى النافذة الرئيسية للإطار وتختص بعرض البيانات على الشاشة.

كما أنه من الجائز تقسيم نافذة المشهد إلى عدة إطارات يعرض كل منها مشهدا مختلفا مستمدا من نفس الوثيقة.

شجرة الوراثة

وفصيلة المشهد من أغنى الفصائل بالدوال والبيانات ، وهى ترث الفصيلة CView و CWnd و CCmdTarget و CObject بالتسلسل. أنظر الشكل التالى.



شكل (٦-٨) فصيلة المشهد فى شجرة الوراثة

ملف العناوين

توضح شريحة الكود التالية الجزء الأساسى من ملف العناوين MFCHelloView.h ونرى فيه مجموعة كبيرة من الدوال الافتراضية المستخدمة فى فصيلة المشهد (تحت التعليق //Overrides). كما نرى فى الجزء الأخير منها دالة خطية ترجع مؤشرا إلى هدف الوثيقة.

```
class CMFCHelloView : public CView
{
protected: // create from serialization only
        CMFCHelloView();
        DECLARE_DYNCREATE(CMFCHelloView)
```

```
// Attributes
public:
        CMFCHelloDoc* GetDocument();

// Operations
public:

// Overrides
  // ClassWizard generated virtual function overrides
  //{{AFX_VIRTUAL(CMFCHelloView)
  public:
  virtual void OnDraw(CDC* pDC);  // overridden to draw this view
  virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
  protected:
  virtual BOOL OnPreparePrinting(CPrintInfo* pInfo);
  virtual void OnBeginPrinting(CDC* pDC, CPrintInfo* pInfo);
  virtual void OnEndPrinting(CDC* pDC, CPrintInfo* pInfo);
  //}}AFX_VIRTUAL

// Implementation
public:
        virtual ~CMFCHelloView();
#ifdef _DEBUG
        virtual void AssertValid() const;
        virtual void Dump(CDumpContext& dc) const;
#endif
...
...
};

#ifndef _DEBUG  // debug version in MFCHelloView.cpp
inline CMFCHelloDoc* CMFCHelloView::GetDocument()
  { return (CMFCHelloDoc*)m_pDocument; }
#endif
```

خريطة الرسائل

تقع خريطة الرسائل كالعادة فى بداية ملف الكود MFCHelloView.cpp وهى تحتوى على إمكانات مختلفة للطباعة (تناظر اختيارات الطباعة بقائمة الملفات للتطبيق MFCHello) وتستخدم دوال الطباعة الموجودة بفصيلة الأساس CView. وهذا هو نصها:

```
BEGIN_MESSAGE_MAP(CMFCHelloView, CView)
  //{{AFX_MSG_MAP(CMFCHelloView)
    // NOTE - the ClassWizard will add and remove mapping macros here.
```

```
    //    DO NOT EDIT what you see in these blocks of generated code!
   //}}AFX_MSG_MAP
   // Standard printing commands
   ON_COMMAND(ID_FILE_PRINT, CView::OnFilePrint)
   ON_COMMAND(ID_FILE_PRINT_DIRECT, CView::OnFilePrint)
   ON_COMMAND(ID_FILE_PRINT_PREVIEW, CView::OnFilePrintPreview)
END_MESSAGE_MAP()
```

الدالة OnDraw()

إن هذه الدالة هى محور فصيلة المشهد وقد استخدمناها فى برنامجنا لطباعة العبارة على الشاشة كالآتى:

```
// CMFCHelloView drawing
void CMFCHelloView::OnDraw(CDC* pDC)
{
        CMFCHelloDoc* pDoc = GetDocument();
        ASSERT_VALID(pDoc);
        // TODO: add draw code for native data here
        pDC ->TextOut(100,100, pDoc->m_strHelloText);
}
```

علاوة على السطر الأخير الذى أضفناه إلى هذه الدالة ، فإن هناك بعض الأجزاء التى تستأهل الاهتمام. فالسطر الأول فى هذه الدالة يستخدم الدالة **GetDocument()** (وهى عضوة بفصيلة الأساس **CView**) للحصول على مؤشر إلى هدف الوثيقة pDoc:

```
CMFCHelloDoc* pDoc = GetDocument();
```

وهذا هو المؤشر الذى استخدمناه فى التوصل إلى المتغير الحرفى m_strHelloText الذى اختزناه من قبل فى هدف الوثيقة. أما العبارة ASSERT_VALID(pDoc) فهى تستخدم لاختبار صلاحية المؤشر.

دوال أخرى بنافذة المشهد

فيما يلى ملخص بالدوال المتبقية فى فصيلة المشهد.

الدالة/الدوال	الوظيفة
CMFCHelloView() ~CMFCHelloView()	دالتا البناء والهدم.
AssertValid() Dump()	دوال صيانة (راجع جدول فصيلة نافذة الإطار ــ جدول (٦-٢)).
OnPreparePrinting() OnBeginPrinting() OnEndPrinting()	دوال تحضير وبداية ونهاية الطباعة.
PreCreateWindow()	لتغيير خصائص نافذة الإطار حسب الطلب. وهى دالة افتراضية من دوال الفصيلة **CWnd**.

الموجز

١. تعرفنا فى هذا الباب بأحداث البرنامج MFC وبأهم الفصائل والدوال الأعضاء التى يتكون منها.

- عرفنا أيضا تفصيلات الأهداف الأربعة الأساسية التى يتكون منها أى تطبيق ، وهى أهداف التطبيق (Application) ، ونافذة الإطار الرئيسية (Main Frame Window) ، والمشهد (View) ، والوثيقة (Document) ، كما عرفنا العلاقة بينهم وبين بعضهم.

٢. تعرفنا أيضا بجزء رئيسى من شجرة فصائل المؤسسة MFC وعلاقة الوراثة بينها وبين بعضها.

٣. عرفنا أيضا هدف التطبيق الأوحد الذى يعلن كمتغير عام **theApp** ، كما عرفنا دالة شحن هدف التطبيق والدور الذى تؤديه فى التطبيق.

٤. تعرفنا بالتطبيق ذى الوثيقة المفردة (SDI) والتطبيق ذى الوثائق المتعددة (MDI) ، وعرفنا أن الأول ترث نافذته من الفصيلة **CFrameWnd** أما الثانى فيرث من **CMDIFrameWnd**.

٥.دوال MFC التى وردت فى هذا الباب:

CCmdTarget::OnCmdMsg()
CControlBar::EnableDocking()
CDC::TextOut()
CDialog::DoModal()
CFrameWnd::DockControlBar()
CFrameWnd::EnableDocking()
CObject::AssertValid()
CObject::Dump()
CObject::Serialize()
CStatusBar::Create()
CStatusBar::SetIndicators()
CToolBar::CreateEx()
CToolBar::LoadToolBar()
CView::GetDocument()
CView::OnBeginPrinting()
Cview::OnDraw()
CView::OnEndPrinting()
CView:OnPreparePrinting()
CWinApp::AddDocTemplate()
CWinApp::InitInstance()
CWnd::DoDataExchange()
CWnd::OnCreate()
CWnd::PreCreateWindow()
CWnd::ShowWindow()
CWnd::UpdateWindow()

٦.البيانات الأعضاء التى وردت فى هذا الباب:

m_pMainWnd
m_pDocument

٧.الرسائل التى وردت فى هذا لباب:

WM_COMMAND
WM_CREATE

٨.الأنماط التى وردت فى هذا الباب:

UINT
CString