MFC 프로그램 구조

HCI Programming 2 (321190)
2007년 가을학기
10/1/2007
박경신

Overview

- MFC 개요
  - MFC 발전 과정과 주요 특징 이해
- MFC 구조
  - MFC 구성요소, 클래스 계층도, 주요 클래스
  - MFC 최상위 클래스인 CObject가 제공하는 서비스 이해
  - MFC가 제공하는 전역 함수 사용법
- MFC 응용프로그램 구조
  - MFC Application Framework Class
  - AppWizard를 이용한 MFC 프로그래밍
  - AppWizard에 의해 생성된 객체
  - AppWizard가 자동으로 생성하는 코드의 구조 이해

MFC 발전 과정

- MFC - 윈도우 프로그램 작성에 유용한 클래스의 집합체

연도	개발 도구	MFC 버전	주요 특징
1992	MS C 7.0	1.0	16비트 윈도우 API를 클래스화 OLE 1.0 지원
1993	비주얼 C++ 1.0	2.0	도큐먼트/뷰 구조 도입으로 MFC의 기본 골격 완성 DDX/DDV, 사용자 인터페이스 관련 클래스 추가 정적/동적 MFC 라이브러리 지원
1993	비주얼 C++ 1.5	2.5x	OLE 2.01 지원 ODBC 클래스 추가
1994	비주얼 C++ 2.x	3.x	32비트 윈도우 API로 전환 멀티스레드 지원 윈속(Winsock), MAPI 지원

MFC 발전 과정

연도	개발 도구	MFC 버전	주요 특징
1995	비주얼 C++ 4.x	4.x	공통 컨트롤 클래스 추가 DAO와 32비트 ODBC 지원 인터넷 관련 클래스 추가
1997	비주얼 C++ 5.0	4.21	ATL 추가
1998	비주얼 C++ 6.0	6.0	ATL 업그레이드 여러 개의 새로운 클래스 추가(CHtmlView, ...) OLE DB, ADO 지원 강화
2002	비주얼 C++ .NET	7.0	새로운 MFC DLL 사용(MFC70.DLL) MFC와 ATL의 통합 강화 사용자 인터페이스 클래스 추가 유틸리티 클래스 추가 ATL 서버 클래스 추가
2005	비주얼 C++ 2005	8.0	MFC Windows Forms 지원

MFC 주요 특징

- 생산성 향상
  - 윈도우 응용 프로그램을 작성하는데 드는 수고를 크게 덜어줌
  - 라이브러리 재사용
  - 비주얼 C++에서 제공하는 AppWizard, ClassWizard 같은 자동화된 코드 생성 도구를 사용하여 코딩 시간을 단축
  - 운영체제의 확장된 기술과 연관된 편리한 클래스 제공 - 인쇄 기능, OLE, ActiveX, 등
  - 사용자인터페이스 지원 - 메뉴, 툴바, 상태바 등
  - 네트워크, 데이터베이스 관련 기술 지원
- 안정성
  - 강한 타입 체크,예외처리, 동적 메모리 할당과 해지

MFC 주요 특징

- 코드 크기 최소화
  - MFC가 제공하는 코드는 DLL 형태로 사용할 수 있으므로 실행 파일의 크기가 증가하는 것을 막을 수 있음
- API 함수 사용
  - API 함수를 직접 호출 가능 – 예, ::ReleaseCapture();.
  - C++ 언어를 이용하여 기존의 C 언어에 비해 API를 좀더 편하게 사용 가능 - 오버로딩 (overloading)과 디폴트 인자(default argument) 등을 이용하여 간편화
- SDK 프로그래밍에 대한 기반 지식을 재활용

MFC 구성 요소

- 구성 요소

MFC 클래스

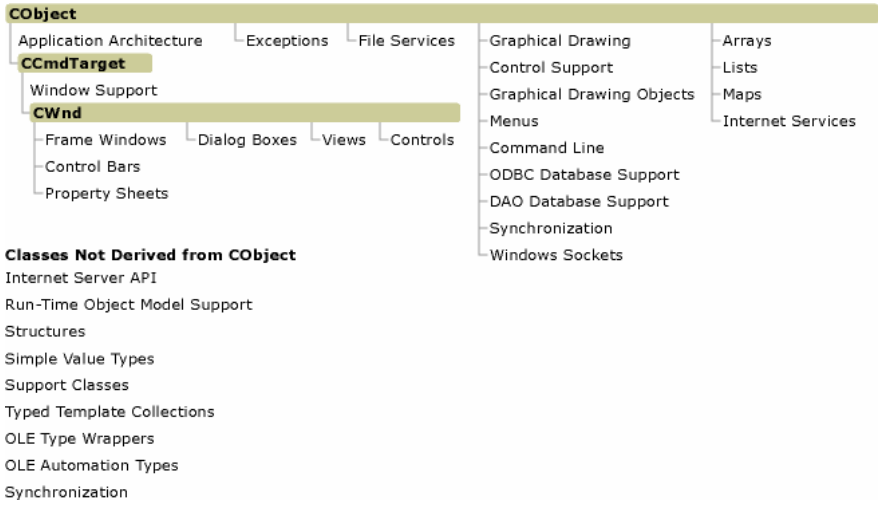
CObject 파생 클래스: 188개

나머지: 40개

(※MFC 6.0 기준)

매크로와 전역 변수 및 전역 함수

MFC 클래스 계층도



MFC 주요 클래스

- CObject 기반 클래스
  - 대부분 MFC클래스가 CObject 클래스로부터 상속됨
  - 실행시간 클래스 경보추출, 동적 객체 생성, 직렬화, 객체 상태 검사, 집합 클래스간 호환성 등을 지원
- Application Framework Class (AFX)
  - 응용프로그램의 기본 구조를 위한 클래스
  - CWinApp – 응용 프로그램을 표현하는 클래스
    - 인스턴스를 초기화하고, 메시지 루프를 설정
  - CFrameWnd – 윈도우의 프레임을 관리
    - 윈도우 이동, 크기 조절 등
  - CView – 윈도우의 클라이언트 영역을 관리하는 클래스
    - 데이터를 보여주는 기능
  - CDocument – 다루는 데이터 구조, 파일을 관리

MFC 주요 클래스

- Window 구성요소 관련 클래스
  - 윈도우, 다이얼로그박스, 콘트롤 등 제공
  - CWnd, CDialog, CEdit CListBox, …
- 그리기와 페인팅관련 클래스
  - 디바이스 컨텍스트와 GDI 객체 등을 제공
  - CDC, CPaintDC, CGdiObject, CBitmap, CBrush, CFont, Cpalette, ..
- 자료형, 자료구조 클래스
  - CPoint, CRect, CSize, CString, CTime , …
  - CArray, CList , CMap, …
- 파일 및 데이터베이스 관련 클래스
  - CFile, CDatabase, CRecordset, CRecordView, …
- 네트워크(인터넷) 관련 클래스
  - CHttpServer , CSocket, CInternetConnection, CHttpConnection

MFC 주요 클래스

- OLE 관련 클래스
  - Object Linking and Embedding
- 예외처리 및 디버깅 관련 클래스
  - CObject::Dump, CException, CMemoryState

CObject 클래스

- CObject 서비스 - <afx.h>

서비스 이름	기능
실행 시간 클래스 정보	프로그램 실행 중 객체 정보(클래스 타입, 크기 등)를 알아낸다.
동적 객체 생성	객체를 동적으로 생성한다.
직렬화	객체를 저장하거나 읽어 들인다.
타당성 점검	객체 상태를 점검한다.
집합 클래스와의 호환성	서로 다른 클래스 객체를 집합 클래스에 저장할 수 있도록 한다.

실행 시간 클래스 정보

- 실행 시간 클래스 정보 기능 추가를 위한 매크로 포함

```
// MyClass.h
class CMyClass : public CObject
{
   DECLARE_DYNAMIC(CMyClass)
   ...
};

// MyClass.cpp
#include "MyClass.h"

IMPLEMENT_DYNAMIC(CMyClass, CObject)
...
```


실행 시간 클래스 정보

- 실행 시간 클래스 정보 사용 예

```
BOOL IsMyClass(CObject *pObj)
{
   // pObj가 가리키는 객체가 CMyClass 타입인지 확인한다.
   if(pObj->IsKindOf(RUNTIME_CLASS(CMyClass)){
      ...
   }
   else{
      ...
   }
}
```


동적 객체 생성

- 동적 객체 생성 기능 추가를 위한 매크로 포함

```
// MyClass.h
class CMyClass : public CObject
{
   DECLARE_DYNCREATE(CMyClass)
public:
   CMyClass();
   ...
};

// MyClass.cpp
#include "MyClass.h"

IMPLEMENT_DYNCREATE(CMyClass, CObject)
...
```


동적 객체 생성

- 동적 객체 생성 사용 예

```
// 객체를 동적으로 생성한다.
CRuntimeClass* pRuntimeClass = RUNTIME_CLASS(CMyClass);
CObject* pObject = pRuntimeClass->CreateObject();

// 객체를 성공적으로 생성했는지 여부를 확인한다.
ASSERT(pObject->IsKindOf(RUNTIME_CLASS(CMyClass)));
```

직렬화

- 직렬화 기능 추가를 위한 매크로 포함

```
// MyClass.h
class CMyClass : public CObject
{
   DECLARE_SERIAL(CMyClass)
public:
   CMyClass();
   virtual void Serialize (CArchive& ar);
   ...
};
```


직렬화

- CObject 의 직렬화 가상함수 재정의

```
// MyClass.cpp
#include "MyClass.h"

IMPLEMENT_SERIAL(CMyClass, CObject, 1)

void CMyClass::Serialize (CArchive& ar)
{
   // CObject가 제공하는 가상 함수인 Serialize() 함수를 재정의한다.
}
...
```


3단계 매크로 기능 정리

단계	매크로 이름	사용 목적	사용 위치
1	DECLARE_DYNAMIC	실행 시간 클래스 정보	클래스 선언부 (*.H)
	IMPLEMENT_DYNAMIC	실행 시간 클래스 정보	클래스 정의부 (*.CPP)
2	DECLARE_DYNCREATE	실행 시간 클래스 정보, 동적 객체 생성	클래스 선언부 (*.H)
	IMPLEMENT_DYNCREATE	실행 시간 클래스 정보, 동적 객체 생성	클래스 정의부 (*.CPP)
3	DECLARE_SERIAL	실행 시간 클래스 정보, 동적 객체 생성, 직렬화	클래스 선언부 (*.H)
	IMPLEMENT_SERIAL	실행 시간 클래스 정보, 동적 객체 생성, 직렬화	클래스 정의부 (*.CPP)

타당성 점검

- 객체의 내부상태를 확인하기 위한 AssertValid() 함수 재정의

```
// MyClass.h
class CMyClass : public CObject
{
   // 멤버 변수
   int m_start;
   int m_end;
public:
   virtual void AssertValid( ) const;
   ...
};
```

타당성 점검

- 타당성 점검 기능 추가

```
// MyClass.cpp
#include "MyClass.h"

virtual void CMyClass::AssertValid( ) const
{
    CObject::AssertValid();
    ASSERT(m_start > 0);          // m_start 값이 0보다 큰 값인지를 검증
    ASSERT(m_end < 100);
}
...
```


집합 클래스와의 호환성

- CObject 포인터를 저장할 수 있는 집합 클래스

종류	클래스 이름
배열	CObArray, CArray(템플릿 클래스)
리스트	CObList, CList(템플릿 클래스)
맵	CMapWordToOb, CMapStringToOb, CMap(템플릿 클래스)

MFC 전역 함수

함수 이름	기능
AfxMessageBox()	메시지 상자를 표시한다.
AfxGetApp()	응용 프로그램 객체의 주소를 리턴한다.
AfxGetMainWnd()	메인 윈도우 객체의 주소를 리턴한다.
AfxGetAppName()	응용 프로그램의 이름을 리턴한다.
AfxGetInstanceHandle()	인스턴스 핸들을 리턴한다.
AfxBeginThread()	스레드를 시작한다.
AfxEndThread()	스레드를 종료한다.

AfxMessageBox()

- 사용 예

```
void CMainFrame::OnLButtonDown(UINT nFlags, CPoint point)
{
    AfxMessageBox("마우스를 클릭했습니다.");
}
```

AfxGetApp(), AfxGetMainWnd(), AfxGetAppName()

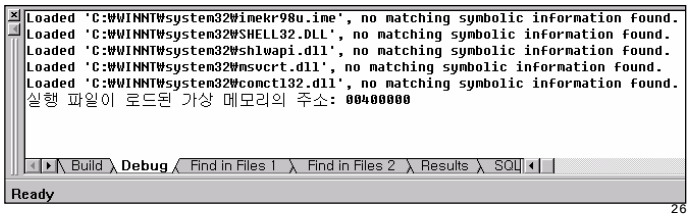
- 사용 예

```
void CMainFrame::OnLButtonDown(UINT nFlags, CPoint point)
{
   TRACE("응용 프로그램 객체의 주소: %p = %p\n",
      AfxGetApp(), &theApp);
   TRACE("프레임 윈도우 객체의 주소: %p = %p\n",
      AfxGetMainWnd(), theApp.m_pMainWnd);
   TRACE("응용 프로그램 이름: %s\n", AfxGetAppName());
}
```

AfxGetInstanceHandle()

- 사용 예

```
void CMainFrame::OnLButtonDown(UINT nFlags, CPoint point)
{
   // 인스턴스 핸들값은 실행 파일이 로드된 가상 메모리의
   // 주소를 나타낸다.
   TRACE("실행 파일이 로드된 가상 메모리의 주소: %p\n",
      AfxGetInstanceHandle());
}
```

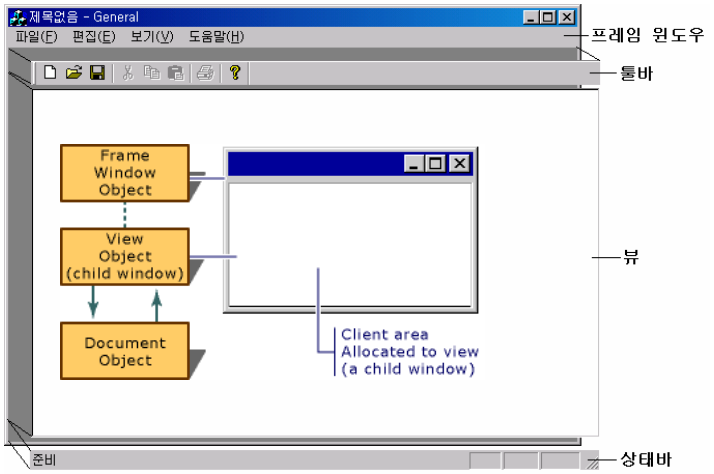


MFC 응용프로그램 구조

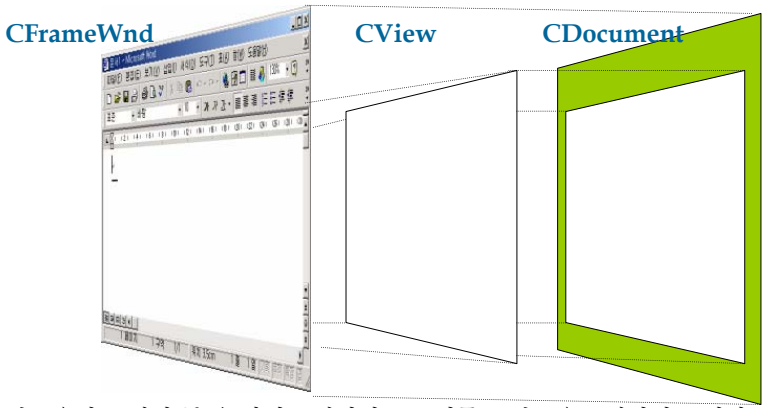
- Application Framework Class (AFX)
  - CWndApp 클래스
  - CWnd 클래스 (CFrameWnd, CView)
  - CDocument 클래스
- AppWizard를 이용한 MFC 프로그래밍
  - AppWizard에 의해 생성된 객체
  - Message Handler

MFC 응용 프로그램 구조

- 일반적인 MFC 프로그램 구성 요소



MFC 응용프로그램의 구조 (AFX)

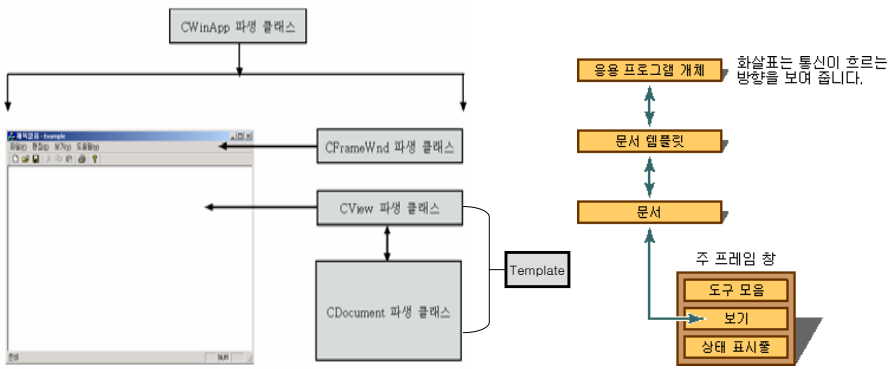


CWinApp : 응용프로그램 관리, 프로그램 구동

Application Framework

- 프레임윈도우와 뷰를 분리한 이유
  - 프레임에 부여된 역할(창의 크기 조절, 확대, 축소 등)을 기본적으로 제공
  - 뷰에서는 실제 보여주어야 할 데이터 만을 적절하게 처리하여 제공
- 도큐먼트와 뷰를 분리한 이유
  - 데이터 저장 및 처리(CDocument)와 이를 보여주는(CView)을 분리하여 클래스의 역할을 분리하여 클래스를 단순화
  - 하나의 데이터(Document)로 여러 개의 뷰(View) 형태를 제공

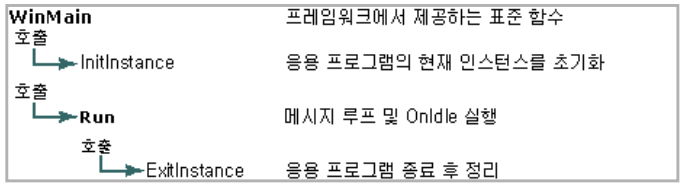
SDI (Single Document Interface)



AFX Class 계층구조

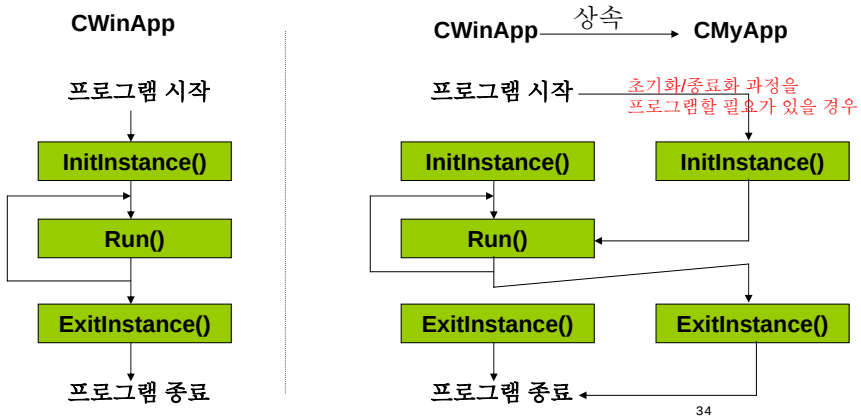
CWinApp 클래스

- CWinApp Class 역할
  - 프로그램 전체를 대표하는 기능
  - 프로그램의 시작과 종료를 담당
  - 프로그램이 시작될 때, 메인 프레임 윈도우를 생성시킴
  - 메시지 루프
- 응용프로그램 실행단계
  - theApp라는 CWinApp의 인스턴스가 전역 변수로 선언됨으로써 실행됨
  - WinMain에서 CWinApp의 함수를 순서대로 호출하면서 실행



CWinApp 멤버함수 재정의

- 새로운 MFC 응용프로그램을 생성하면, CWinApp를 상속받는 클래스가 생성되며, 필요 시 관련 함수 재정의

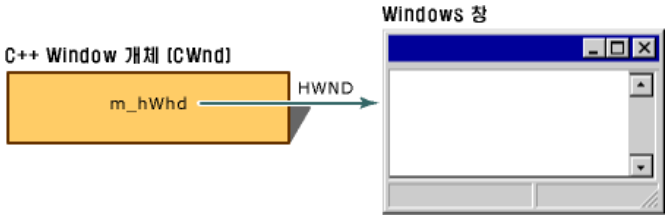


CWnd 클래스

- 프레임 창, 대화 상자, 자식 창, 컨트롤 및 도구 모음과 같은 컨트롤 막대 등의 모든 창에 대한 Base class
- 멤버함수
  - 윈도우 관리에 필요한 Windows API를 캡슐화
  - 윈도우의 크기, 위치, 모양, 상태 등을 제어하기 위한 함수
  - 윈도우에서 발생하는 메시지를 처리하기 위한 함수(메시지 핸들러)
- 하위클래스에서 구체적으로 이용
  - CFrameWnd
    - 윈도우의 스타일 관리(크기, 위치,..)
    - 메뉴, 툴바, 상태바, 뷰 생성 및 관리
  - CView
    - 프로그램에서 다루는 데이터를 화면에 출력
    - 클라이언트 영역에서 발생되는 마우스, 키보드 메시지 처리를 위한 메시지 핸들러 오버라이딩

CWnd & HWND

- CWnd 개체는 C++ 윈도우(창) 개체이면서 Windows 윈도우(창)를 나타냄
  - CWnd 생성자 : C++ 윈도우 개체를 할당하고 초기화
  - Create 멤버 함수 : Windows 윈도우 생성
- HWND는 Windows 윈도우의 핸들
  - HWND 인수를 사용하는 대부분의 Windows 윈도우 관리 API는 CWnd의 멤버 함수로 캡슐화



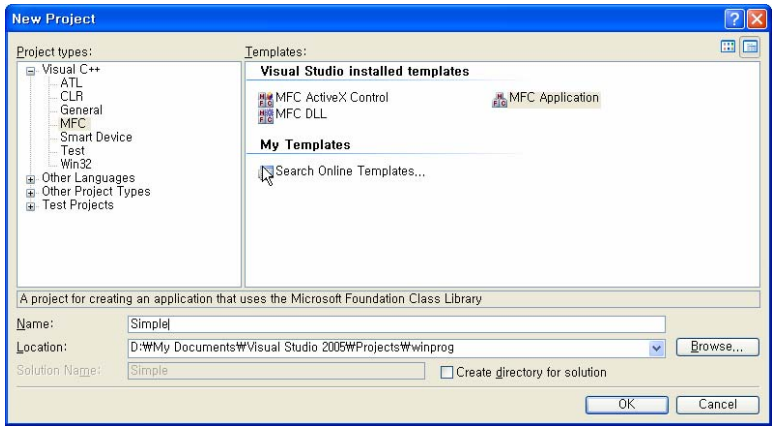
CWnd 하위 클래스

CDocument 클래스

- 프로그램에서 사용되는 데이터를 저장하고, 읽고, 처리하기 위한 모든 기능을 수행
  - 파일로부터 데이터를 읽어 오는 기능("파일" 메뉴의 "열기" 기능)
  - 파일로부터 데이터를 저장 하는 기능("파일" 메뉴의 "저장" 기능)
  - 새로운 파일 생성 기능("파일"메뉴의 "새로 만들기" 기능)
  - 작업중인 파일 닫기 기능 ("파일"메뉴의 "닫기" 기능)
  - 변경된 데이터를 뷰 객체에 알려주는 기능 등

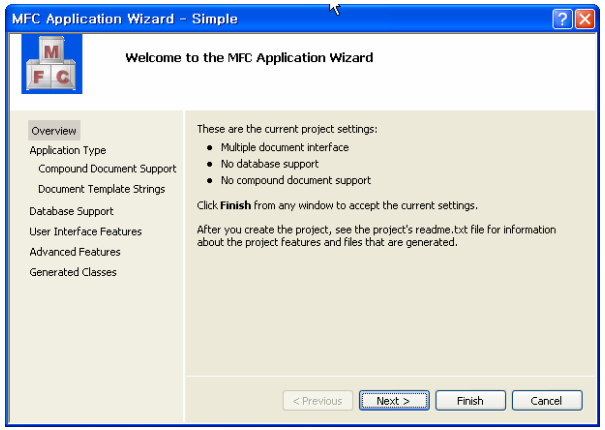
AppWizard를 이용한 MFC 응용 프로그램 생성

- AppWizard 시작 - 프로젝트 종류 선택



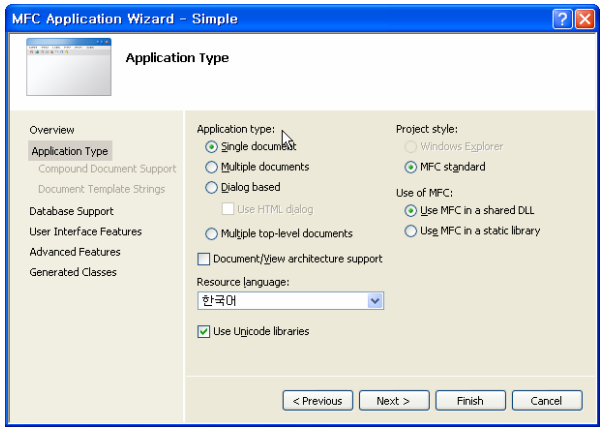
AppWizard를 이용한 MFC 응용 프로그램 생성

- AppWizard



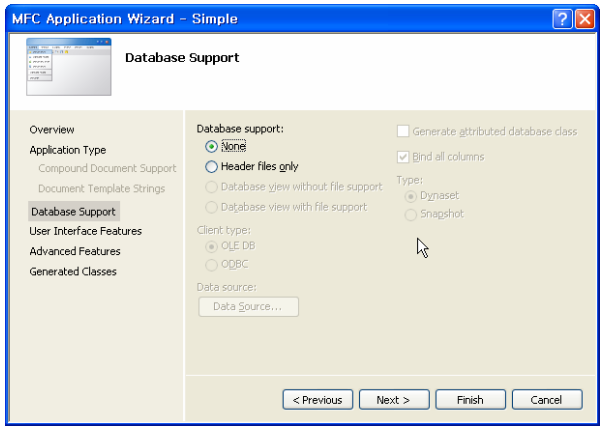
AppWizard를 이용한 MFC 응용 프로그램 생성

- AppWizard – SDI, Use MFC in a Shared DLL, 한국어 선택



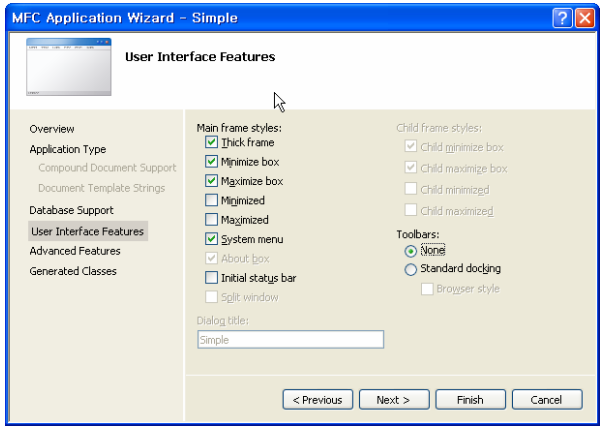
AppWizard를 이용한 MFC 응용 프로그램 생성

- AppWizard – 데이터 베이스



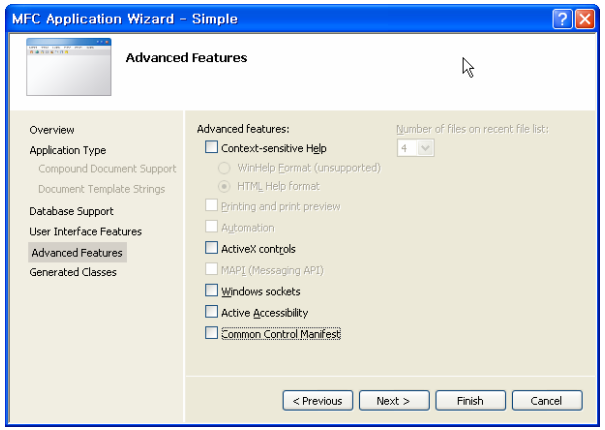
AppWizard를 이용한 MFC 응용 프로그램 생성

- AppWizard – 사용자 인터페이스



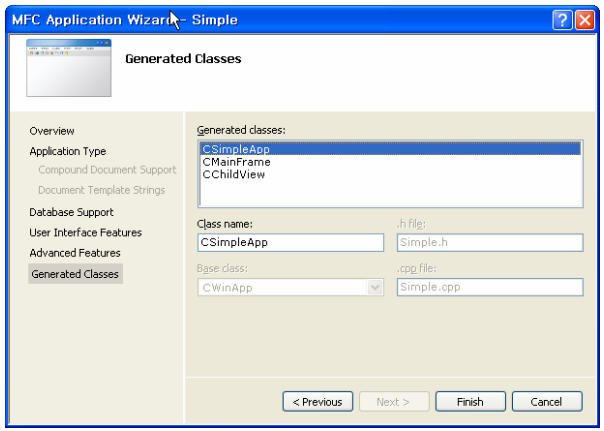
AppWizard를 이용한 MFC 응용 프로그램 생성

- AppWizard – ActiveX 를 선택해제



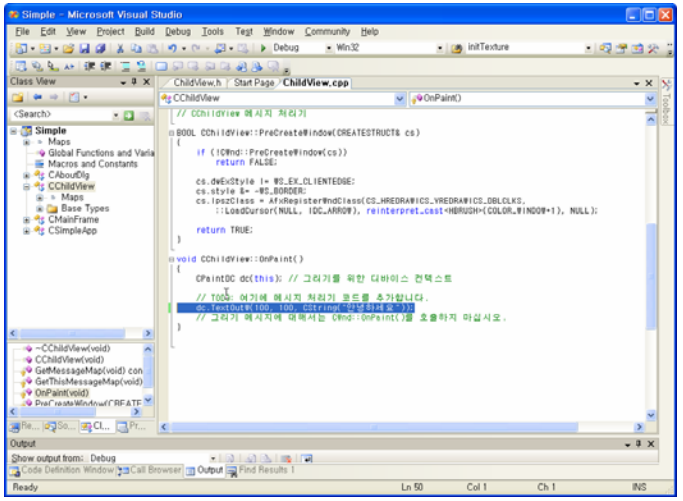
AppWizard를 이용한 MFC 응용 프로그램 생성

- AppWizard

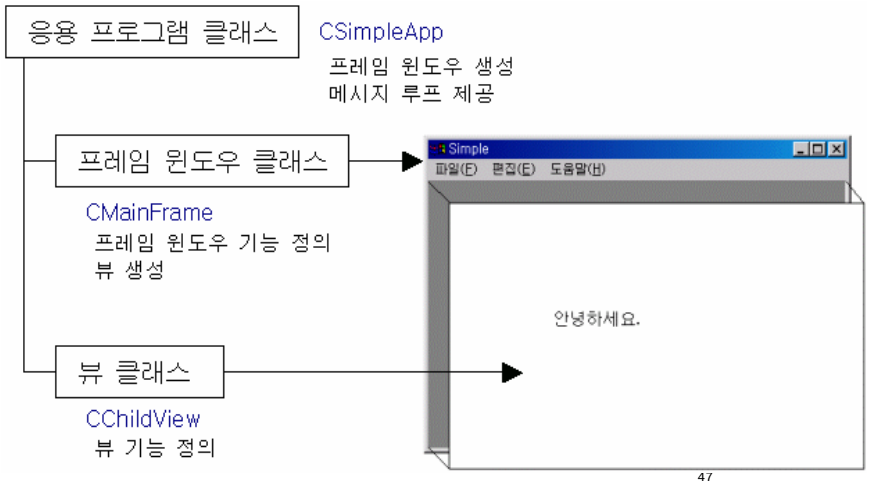


AppWizard를 이용한 MFC 응용 프로그램 생성

- 코드 추가



주요 클래스



응용 프로그램 클래스

```
// Simple.h
class CSimpleApp : public CWinApp
{
public:
    CSimpleApp();
    virtual BOOL InitInstance();
    afx_msg void OnAppAbout();
    DECLARE_MESSAGE_MAP()
};
```

응용 프로그램 클래스

```
// Simple.cpp
BEGIN_MESSAGE_MAP(CSimpleApp, CWinApp)
    ON_COMMAND(ID_APP_ABOUT, OnAppAbout)
END_MESSAGE_MAP()

CSimpleApp::CSimpleApp()
{
}

CSimpleApp theApp;
```


응용 프로그램 클래스

```
BOOL CSimpleApp::InitInstance()
{
    SetRegistryKey(_T("Local AppWizard-Generated Applications"));

    CMainFrame* pFrame = new CMainFrame;
    m_pMainWnd = pFrame;

    pFrame->LoadFrame(IDR_MAINFRAME,
        WS_OVERLAPPEDWINDOW | FWS_ADDTOTITLE, NULL,
        NULL);

    pFrame->ShowWindow(SW_SHOW);
    pFrame->UpdateWindow();

    return TRUE;
}
```


응용 프로그램 클래스

```
// 대화상자 관련 클래스 선언 및 정의 부분 - 생략
// ...

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


프레임 윈도우 클래스

```
// MainFrm.h
class CMainFrame : public CFrameWnd
{
public:
    CMainFrame();
protected:
    DECLARE_DYNAMIC(CMainFrame)
public:
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
    virtual BOOL OnCmdMsg(UINT nID, int nCode, void* pExtra,
        AFX_CMDHANDLERINFO* pHandlerInfo);
    virtual ~CMainFrame();
    CChildView m_wndView;
protected:
    afx_msg int OnCreate(LPCREATESTRUCT lpCreateStruct);
    afx_msg void OnSetFocus(CWnd *pOldWnd);
    DECLARE_MESSAGE_MAP()
};
```

프레임 윈도우 클래스

```
// MainFrm.cpp
IMPLEMENT_DYNAMIC(CMainFrame, CFrameWnd)

BEGIN_MESSAGE_MAP(CMainFrame, CFrameWnd)
    ON_WM_CREATE()
    ON_WM_SETFOCUS()
END_MESSAGE_MAP()

CMainFrame::CMainFrame()
{
}

CMainFrame::~CMainFrame()
{
}
```


프레임 윈도우 클래스

```
int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
    if (CFrameWnd::OnCreate(lpCreateStruct) == -1)
        return -1;
    if (!m_wndView.Create(NULL, NULL, AFX_WS_DEFAULT_VIEW,
        CRect(0, 0, 0, 0), this, AFX_IDW_PANE_FIRST, NULL))
    {
        TRACE0("Failed to create view window\n");
        return -1;
    }

    return 0;
}
```


프레임 윈도우 클래스

```
BOOL CMainFrame::PreCreateWindow(CREATESTRUCT& cs)
{
    if( !CFrameWnd::PreCreateWindow(cs) )
        return FALSE;
    cs.dwExStyle &= ~WS_EX_CLIENTEDGE;
    cs.lpszClass = AfxRegisterWndClass(0);
    return TRUE;
}

void CMainFrame::OnSetFocus(CWnd* pOldWnd)
{
    m_wndView.SetFocus();
}
```


프레임 윈도우 클래스

```
BOOL CMainFrame::OnCmdMsg(UINT nID, int nCode, void* pExtra,
    AFX_CMDHANDLERINFO* pHandlerInfo)
{
    if (m_wndView.OnCmdMsg(nID, nCode, pExtra, pHandlerInfo))
        return TRUE;

    return CFrameWnd::OnCmdMsg(nID, nCode, pExtra, pHandlerInfo);
}
```

뷰 클래스

```
// ChildView.h
class CChildView : public CWnd
{
public:
    CChildView();
protected:
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
public:
    virtual ~CChildView();
protected:
    afx_msg void OnPaint();
    DECLARE_MESSAGE_MAP()
};
```


뷰 클래스

```
// ChildView.cpp
CChildView::CChildView()
{
}

CChildView::~CChildView()
{
}

BEGIN_MESSAGE_MAP(CChildView,CWnd )
    ON_WM_PAINT()
END_MESSAGE_MAP()
```


뷰 클래스

```
BOOL CChildView::PreCreateWindow(CREATESTRUCT& cs)
{
    if (!CWnd::PreCreateWindow(cs))
        return FALSE;

    cs.dwExStyle |= WS_EX_CLIENTEDGE;
    cs.style &= ~WS_BORDER;
    cs.lpszClass = AfxRegisterWndClass (
        CS_HREDRAW|CS_VREDRAW|CS_DBLCLKS,
        ::LoadCursor(NULL, IDC_ARROW),
        HBRUSH(COLOR_WINDOW+1), NULL);

    return TRUE;
}
```


뷰 클래스

```
void CChildView::OnPaint()
{
    CPaintDC dc(this);
    dc.TextOut(100, 100, CString("안녕하세요."));
}
```

요약

클래스 종류	베이스 클래스 이름	핵심 함수 - 주 역할
응용 프로그램 클래스 (CSimpleApp) Simple.cpp, Simple.h	CWinApp	InitInstance() - 프레임 윈도우 생성 Run() - 메시지 루프 제공
프레임 윈도우 클래스 (CMainFrame) MainFrm.cpp, MainFrm.h	CFrameWnd	OnCreate() - 뷰 생성
뷰 클래스 (CChildView) ChildView.cpp, ChildView.h	CWnd	OnPaint() - 화면 출력 OnPaint()에서 OnDraw() 호출 구조

MFC 응용프로그램 실행단계