OLE

- Dostrzeżenie ograniczeń schowka [koniec lat 80-tych]
- OLE 1.0 [1991]
  - *Object Linking & Embedding*
  - osadzanie obiektów w dokumentach
  - dokumenty jako kontenery obiektów
- OLE 2.0 [1993]
  - COM - *Component Object Model*
  - wielokrotne użycie kodu skompilowanego do postaci komponentów

OLE

- Składowe:
  - *Component Objects i Component Object Model*
  - *Compound Files*
  - *Monikers*
  - *Uniform Data Transfer*
  - *Automation*
  - *Drag & Drop*
  - *Embedding*
  - *Linking*
  - *In-Place Activation*

Component Object Model (COM)

- System tworzenia binarnych komponentów
  - niezależny od platformy i języka programowania
  - rozproszony
  - obiektowo zorientowany
- Interfejsy
  - grupy metod
  - identyfikowane nazwą z przedrostkiem „I”
  - dostępne w postaci wskaźnika
  - IUnknown, QueryInterface()

Component Objects

- Obiekty
  - serwer klasy o podanej nazwie jest automatycznie lokalizowany przez COM (na podstawie wpisów w rejestrze)
  - serwer tworzy obiekt i zwraca wskaźnik do interfejsu
  - COM odpowiada za komunikację z obiektem (być może w innym procesie, na innej maszynie)
- Komponenty
  - dzielone
  - wielokrotnego użytku

Monikers

- Specjalny typ obiektu *(component object)* do zarządzania abstrakcyjnymi referencjami innych obiektów
  - informacje niezbędne do zlokalizowania lub stworzenia obiektu (szczególnie istotne w przypadku połączeń sieciowych)
  - kod tworzący na podstawie tych informacji rzeczywisty obiekt
- Podstawowe implementacje monikerów są dostarczane przez OLE, można tworzyć własne
- Interfejs IMoniker

Compound Files

- Pliki ze strukturalną zawartością *(„file system within a file”)*
  - *storage objects* - funkcje podkatalogów [IStorage]
  - *stream objects* - funkcje pojedynczych plików [IStream]
- Cechy:
  - *incremental access* - zarządzanie zmianą objętości pojedynczych strumieni w obrębie obiektu
  - obsługa transakcji - zapis obiektu na dysku dopiero po zatwierdzeniu transakcji
  - łatwy dostęp do poszczególnych strumieni jako tablicy bajtów
  - odzyskiwanie nieużywanego miejsca

Uniform Data Transfer

- Mechanizm wymiany danych
  - połączenie źródła danych z ich odbiorcą
- Cechy:
  - wykorzystanie *data objects* [IDataObject]
  - bogaty opis danych (znacznie większy niż w przypadku schowka)
  - niemal dowolne dane (plik, *compound file*, strumień, obiekt GDI, globalna pamięć)
  - niezbyt wydajne w przypadku danych wielkich rozmiarów lub o bardzo skomplikowanej strukturze
  - wykorzystywany w schowku, DDE, OLE *Drag and Drop*, dokumentach OLE

OLE Drag and Drop

- Inny sposób wymiany danych
- Źródło danych
  - dostarcza obiekt danych
  - implementuje IDropSource dla pewnych obiektów (np. dokumentów aplikacji)
- Odbiorca danych
  - implementuje IDropTarget dla pewnych obiektów i rejestruje to w OLE
- Cały mechanizm obsługi łapania lewym przyciskiem myszy, przeciągania i puszczania jest obsługiwany przez OLE

OLE Automation

- Pozwala na dostęp z zewnątrz do metod i właściwości aplikacji
  - *automation server* - aplikacja
  - *automation client* - np. zewnętrzny program wykorzystujący funkcjonalność aplikacji
  - *automation object* - dowolny obiekt udostępniany przez aplikację [IDispatch]
- Możliwości:
  - tworzenie aplikacji i narzędzi programistycznych udostępniających obiekty
  - tworzenie i manipulowanie obiektami stworzonymi w innej aplikacji
  - tworzenie narzędzi do manipulowania udostępnianymi obiektami

OLE Documents

- Mechanizm dzielenia danych pomiędzy aplikacjami
  - serwer definiuje obiekt (dane, sposób wyświetlania, właściwości edycyjne)
  - kontener daje miejsce, w którym obiekt będzie umieszczony i wyświetlony
- Aplikacje nie muszą znać siebie nawzajem - wszystkim zarządza OLE
  - dzielone dane zawierają wszystkie dane potrzebne do ich stworzenia (także identyfikator klasy serwera)
- Aktywacja dokumentu w kontenerze
  - *in-place activation*
  - *embedded*
  - *linked*

Embedding and Linking

- *Embedding*
  - wszystkie dane potrzebne do aktywacji i wyświetlenia obiektu są zawarte w dokumencie OLE
  - wykorzystywany dla niedużych obiektów
- *Linking*
  - tylko referencja do obiektu
  - dane istnieją w innym miejscu niż kontener
  - jeśli obiekt ma wiele referencji, to modyfikacja z jednego kontenera oddziałuje na pozostałe
- Nie ma różnic funkcjonalnych

In-Place Activation

- Umożliwia wyświetlenie narzędzi edycyjnych dotyczących obiektu (np. menu, pasków narzędzi)
  - serwer umieszcza swoje narzędzia edycyjne w kontekście kontenera
  - wymaga obsługi po stronie kontenera i obiektu
  - ukierunkowanie na dokument, a nie na aplikację
- Kontener
  - IOleInPlaceFrame, IOleInPlaceUIWindow, IOleInPlaceSite
  - jeśli kontener nie obsługuje *in-place activation*, obiekt zostanie osadzony standardowo
- Obiekt
  - IOleInPlaceObject, IOleInPlaceActiveObject

Osadzanie dokumentów OLE w .NET

- Kontrolka ActiveDocumentHost
  - zapowiadana w .NET Framework 2.0, ale usunięta po testach
- Kontrolka WebBrowser [2.0]
  - wykorzystuje kontrolkę ActiveX IE Browser

- Kontrolka ActiveX DSOFramer
  - http://support.microsoft.com/?id=311765
  - brak wsparcia firmy Microsoft

Obsługa błędów

- HRESULT
  - typ zwracany przez większość funkcji COM
  - najstarszy bit określa wynik operacji (sukces lub porażka)
  - makra sprawdzające wynik:
    SUCCEEDED()
    FAILED()
  - makra dające dostęp do składowych HRESULT:
    HRESULT_CODE
    HRESULT_FACILITY
    HRESULT_SEVERITY
  - interfejsy wykorzystywane w obsłudze błędów:
    ICreateErrorInfo, IErrorInfo, ISupportErrorInfo

DCOM - Distributed COM

- Obsługa komunikacji pomiędzy obiektami umieszczonymi na różnych komputerach
  - LAN *(local area network)*
  - WAN *(wide area network)*
  - Internet
- Windows NT 4.0, 9x
- Cechy:
  - niezależność od umiejscowienia
  - duża liczba małych komponentów zwiększa ruch w sieci, mała liczba dużych komponentów zmniejsza elastyczność
  - niezależność od języka programowania
  - automatyczne sprawdzanie trwania połączenia
  - skalowalność

COM+

- Powiązanie *Component Object Model* i *Microsoft Transaction Server*
- Wymagania serwera:
  - 1.0: Windows 2000
  - 1.5: Windows XP, Windows 2003 Server
- Wymagania klienta:
  - Windows NT, 98

ActiveX

- Zbiór technologii umożliwiających współdziałanie komponentów w środowisku sieciowym
- Wykorzystywany na stronach internetowych i w aplikacjach

- Przedstawiony w marcu 1996 roku pod hasłem *„Activate the Internet”*
  - powszechnie używany w celach marketingowych
- Synonim OLE
  - rozszerzenie OLE na Internet, komercyjny intranet oraz aplikacje i narzędzia ich tworzenia

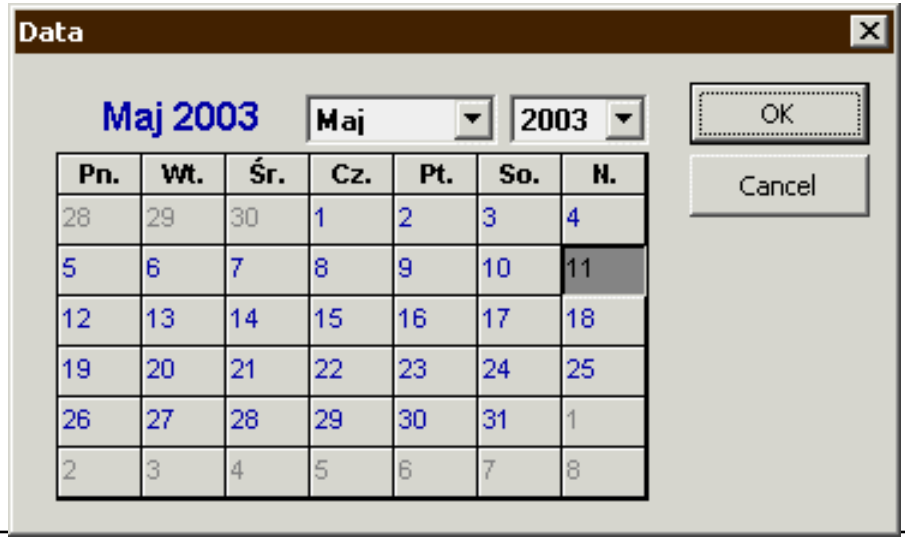
ActiveX Control

- Komponent
  - może być wykorzystany w aplikacjach i na stronach WWW
  - wielokrotnego użytku
  - binarny (skompilowany w dowolnym języku)
  - implementujący interfejs IUnknown
  - implementujący funkcje DllRegisterServer() i DllUnregisterServer
- Synonimy: *OLE control, OCX control*
- Ogromny zbiór istniejących kontrolek dostępnych dla programistów

Tworzenie

- MFC - Microsoft Foundation Classes Library [C++]
  - obiektowa biblioteka dostępu do funkcji API ze zbiorem własnych klas pomocniczych
  - tworzone kontrolki są stosunkowo małe, ale do uruchomienia wymagają biblioteki MFC
- BaseCtl [Visual Basic]
  - bardzo trudne tworzenie kontrolek ActiveX wymagające dużej wiedzy o COM
  - najmniejszy z możliwych rozmiar kontrolek
- ATL - ActiveX Template Library [C++]
  - biblioteka stworzona specjalnie dla kontrolek ActiveX
  - bardzo małe i szybkie kontrolki
  - potrzeba głębszej wiedzy o COM niż w przypadku MFC

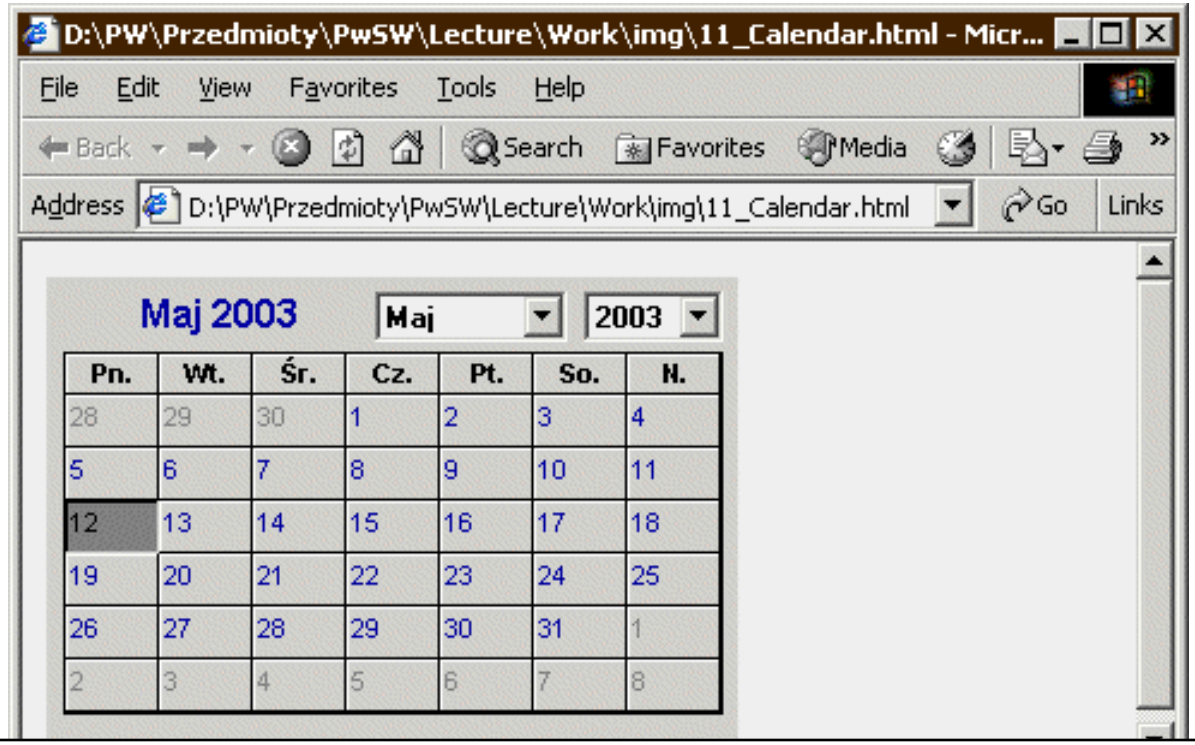
Wykorzystanie w aplikacjach




```
IDD_DATA_DIALOG DIALOGEX 0, 0, 244, 121
...
BEGIN
  DEFPUSHBUTTON   "OK",IDOK,187,7,50,16
  PUSHBUTTON      "Cancel",IDCANCEL,187,25,50,16
  CONTROL         "",IDC_CALENDAR1,
      "{8E27C92B-1264-101C-8A2F-040224009C02}",
      WS_TABSTOP,7,7,173,111
END
```

Wykorzystanie na stronach WWW

- Kontrolka ActiveX jest uruchamiana przy wyświetleniu strony




```
<OBJECT id="Calendar1" classid="clsid:8E27C92B-
                    1264-101C-8A2F-040224009C02">
  <PARAM NAME="Year" VALUE="2003">
  <PARAM NAME="Month" VALUE="5">
  <PARAM NAME="Day" VALUE="12">
</OBJECT>
```

Kontener

- Środowisko, w którym kontrolka może zostać uruchomiona
- Dostęp do wszystkich metod, właściwości i zdarzeń
  - bez obowiązku implementowania wszystkich składowych

Rejestracja

- Autorejestracja
  - DllRegisterServer() - tworzy wpisy w rejestrze Windows dla wszystkich klas zawartych w module
  - DllUnregisterServer() - usuwa wszystkie wpisy w rejestrach dodane przez DllRegisterServer()
- regsvr32.exe
  - wykorzystuje autorejestrację kontrolki

Właściwości

- Cechy kontrolki, które mogą być ustawiane podczas jej pracy
- *Property sheet* - okno dialogowe z zakładkami dającymi dostęp do *property page*
  - ściśle określony sposób definiowania
    IPropertyPage, IPropertyPage2
    ISpecifyPropertyPages
    IPropertyPageSite
  - wygląd niezależny od kontenera
  - dwa standardowe rozmiary
- Okno z właściwościami jest dostępne dla użytkownika uruchamiającego kontrolkę

Metody

- Dowolne zaimplementowane przez twórców kontrolki
  - funkcjonalność kontrolki
- Standardowe metody, które mogą być obsługiwane przez kontrolkę:
  - Refresh()
  - DoClick()
  - AboutBox()

Zdarzenia

- Rodzaje zdarzeń:
  - *request* - zapytanie o pozwolenie wykonania metody
  - *before* - powiadomienie przez akcją
  - *after* - powiadomienie po akcji
  - *do* - pozwala na zmianę akcji, która ma zostać wykonana
- Standardowe zdarzenia, które mogą być obsługiwane przez kontrolkę:
  - Click, DblClick, MouseMove, MouseUp
  - KeyDown, KeyPress, KeyUp
  - Error

Bezpieczeństwo

- Kontrolka ActiveX jako obiekt COM może wszystko - jest uruchamiana bez ograniczeń
- Cyfrowy podpis kontrolki - certyfikat bezpieczeństwa
- Licencje
  - *design-time* - używana przez kontener, weryfikuje licencję dla programistów wykorzystujących kontrolkę
  - *run-time* - weryfikacja licencji podczas uruchomienia kontrolki