Gliwice, Kwiecień 2004

Programowanie w środowiskach graficznych

System operacyjny Windows

Marek Mittmann

Plan wykładu

- Historia
- Cechy systemu Windows
- Program sterowany zdarzeniami
- Rysowanie
- Kontrolki

Historia systemu Windows

- Prace nad GUI w Xerox PARC – lata 70
- Pierwsza wersja Windows – listopad 1985
- Windows 2.0 - listopad 1987
- Windows/287 i Windows/387
- Windows 3.0 – maj 1990 – wsparcie dla trybu chronionego procesorów 80x86
- Windows 3.1 – kwiecień 1992 – nowe technologie: TrueType, OLE, multimedia
- Windows NT – lipiec 1993 – pierwsza wersja pracująca w trybie 32-bitowym
- Windows 95 – kwiecień 1995
- Windows 98, Windows ME, Windows 2000, Windows XP, ...

Cechy systemu Windows

- Interfejs graficzny
- Wielowątkowość
- GUI oparte na zestawie standardowych komponentów graficznych
- Elementy OOUI
- Niezależność od sprzętu, sterowniki urządzeń
- Biblioteki dołączane dynamicznie (DLL)
- Aplikacje sterowane zdarzeniami

Windows API

- Podsystemy: Kernel, User, GDI
- Korzystanie z funkcji Windows API
  - Konwencje
  - Typy danych, struktury, uchwyty
  - Używanie w różnych językach programowania
- Dokumentacja
  - MSDN, Platform SDK

Programy sterowane zdarzeniami

- Zdarzenia
  - Rodzaje zdarzeń
  - Generowanie komunikatów
  - Funkcje *SendMessage* i *PostMessage*
- Reagowanie na zdarzenia
  - *Don't Call Me, I'll Call You*
  - Kolejka komunikatów
  - Procedura okna

Przykładowa aplikacja

```
int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
                   PSTR szCmdLine, int iCmdShow)
{
     /* ... */
     RegisterClass(&wndclass);                  //
     hwnd = CreateWindow( /* ... */ );          // Stworzenie
     ShowWindow(hwnd, iCmdShow);                // i otwarcie okna
     UpdateWindow(hwnd);                        //
     while (GetMessage(&msg, NULL, 0, 0)) {
          TranslateMessage(&msg);               // Pobieranie komunikatów
          DispatchMessage(&msg);                // z kolejki
     }
     return msg.wParam;
}

// Procedura okna
LRESULT CALLBACK WndProc(HWND hwnd, UINT message, WPARAM wParam, LPARAM lParam)
{
     switch (message) {
          case WM_xxxx:                         // Obsługa komunikatów
               /* ... */
               return 0;
     }
     return DefWindowProc(hwnd, message, wParam, lParam);
}
```

Rysowanie

- Zdarzenie WM_PAINT
- Regiony zaktualizowane i unieważnione
- Podsystem GDI
  - Kontekst urządzenia (DC)
  - Funkcje GDI
  - Atrybuty kontekstu urządzenia
  - Obiekty GDI

8

Rysowanie - przykład

```
/* ... */

LRESULT CALLBACK WndProc(HWND hwnd, UINT message, WPARAM wParam, LPARAM lParam)
{
     HDC         hdc;
     PAINTSTRUCT ps;
     RECT        rect;

     switch (message) {
          /* ... */
          case WM_PAINT:
               hdc = BeginPaint(hwnd, &ps);
               GetClientRect(hwnd, &rect);
               SelectObject(hdc, GetStockObject(GRAY_BRUSH));
               Rectangle(hdc, 10, 10, rect.right-10, rect.bottom-10);
               DrawText(hdc, "Hello world", -1, &rect,
                        DT_SINGLELINE | DT_CENTER | DT_VCENTER);
               EndPaint(hwnd, &ps);
               return 0;
     }
     return DefWindowProc(hwnd, message, wParam, lParam);
}
```

9

Urządzenia wejściowe

- Klawiatura
  - Akceleratory
  - Ognisko wejściowe
  - Komunikaty związane z klawiaturą
  - Kody klawiszy – *Virtual Key Codes*
- Mysz
  - Komunikaty związane z myszą
  - Kursory
  - Przechwytywanie myszy – funkcja *SetCapture*

10

Okna dialogowe

- Funkcja *MessageBox*
- Okna modalne i niemodalne
- Procedura obsługi komunikatów
- Okna z zakładkami i kreatory
- Standardowe okna dialogowe systemu Windows

11

Kontrolki

- Okna potomne
- Standardowe kontrolki Windows
- Używanie kontrolek
  - Umieszczanie w oknie
  - Wywoływanie funkcji kontrolek
  - Obsługa komunikatów
- Kontrolki ActiveX

12

Kontrolki - przykład

```
/* ... */

LRESULT CALLBACK WndProc(HWND hwnd, UINT message, WPARAM wParam, LPARAM lParam)
{
    HWND hEdit;
    switch (message) {
        /* ... */
        case WM_CREATE:
            // Utworzenie kontrolki
            hEdit = CreateWindow("edit", /* ... */);
            return 0;

        case WM_COMMAND:
            // Obsługa komunikatu generowanego przez kontrolkę
            if (LOWORD(wParam) == ID_EDIT)
               if (HIWORD(wParam) == EN_ERRSPACE ||
                   HIWORD(wParam) == EN_MAXTEXT)
                   MessageBox(hwnd, "Edit control out of space", NULL, 0);
            return 0;
    }
    return DefWindowProc(hwnd, message, wParam, lParam);
}
```


Multimedia

- Odtwarzanie plików multimedialnych
  - Funkcja *PlaySound*
  - Media Control Interface (MCI)
  - DirectShow, DirectSound, DirectMusic, ...
- Grafika 3D i gry komputerowe
  - DirectX

Style wizualne w Windows XP

- Co jest potrzebne?
  - Biblioteka *Comctl32.dll* w wersji 6.0 (dostarczana z Windows XP)
  - Plik manifestu – plik XML dołączony do aplikacji jako zasób lub umieszczony w katalogu z plikiem wykonywalnym
  - W kontrolkach, które mają właściwość *FlatStyle*, należy ustawić wartość *FlatStyle.System*

Plik manifestu

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<assembly xmlns="urn:schemas-microsoft-com:asm.v1" manifestVersion="1.0">
<assemblyIdentity
    version="1.0.0.0"
    processorArchitecture="X86"
    name="Microsoft.Winweb.Executable Name"
    type="win32"
/>
<description>.NET control deployment tool</description>
<dependency>
    <dependentAssembly>
        <assemblyIdentity
            type="win32"
            name="Microsoft.Windows.Common-Controls"
            version="6.0.0.0"
            processorArchitecture="X86"
            publicKeyToken="6595b64144ccf1df"
            language="*"
        />
    </dependentAssembly>
</dependency>
</assembly>
```

Nazwa pliku wykonywalnego aplikacji (bez rozszerzenia)

Dodawanie manifestu do aplikacji

- Jako plik zewnętrzny
  - Zmienić nazwę pliku manifestu na *NazwaAplikacji.exe.manifest*
  - Umieścić w katalogu aplikacj, razem z plikiem wykonywalnym
- Jako zasób zawarty w pliku wykonywalnym
  - Dołączyć do pliku wykonywalnego plik manifestu jako zasób typu *RT_MANIFEST*
  - Zmienić wartość identyfikatora zasobu na 1

Dziękuję za uwagę