Tvorba uživatelských rozhraní

Prezentace přednášek

Ústav počítačové grafiky a multimédií

Téma přednášky

Programy ve Windows, zprávy, okna, jejich vlastnosti a překreslování

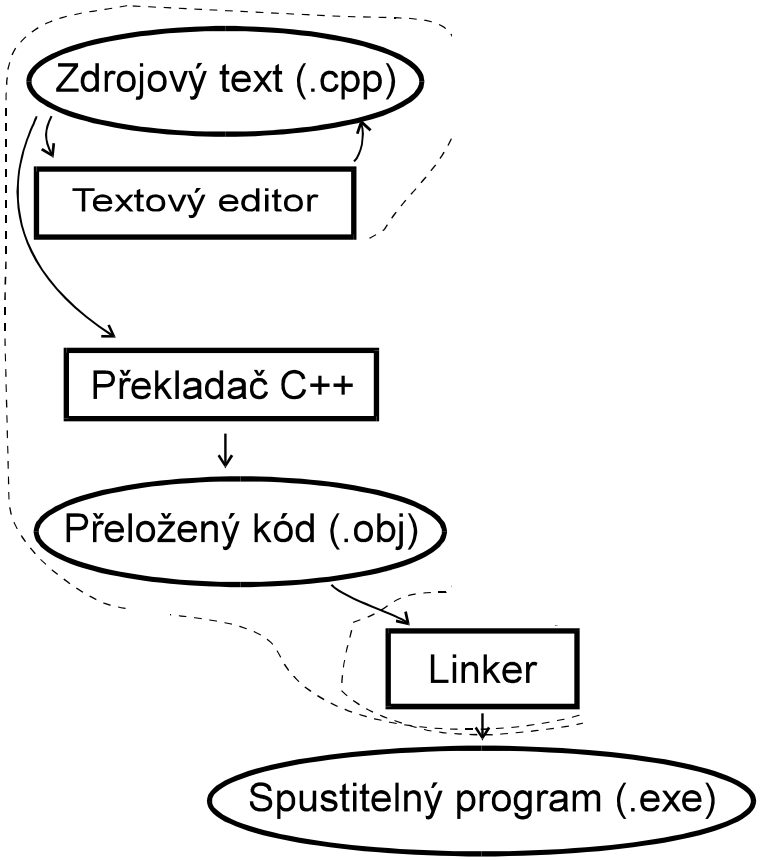
Motto

„Jak je to vlastně s těmi programy pro Windows a jak pracují okna?“

Obsah:

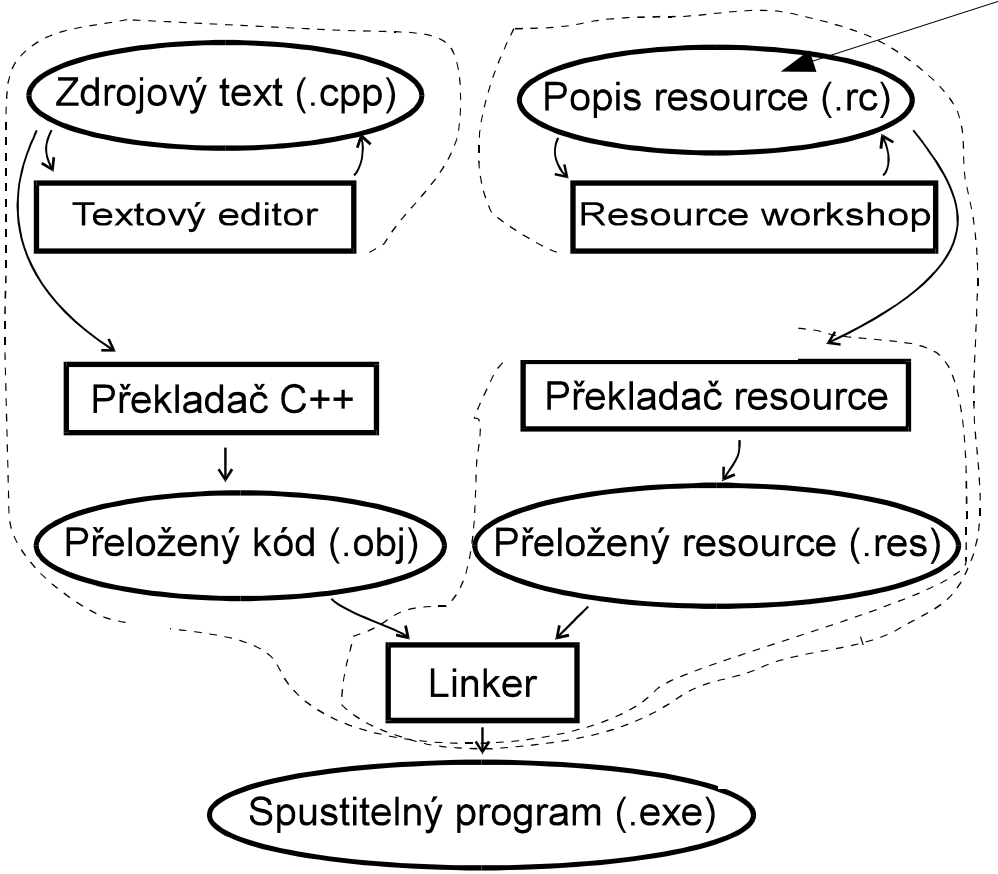
Programy ve Windows

„Klasický” program



Programy ve Windows

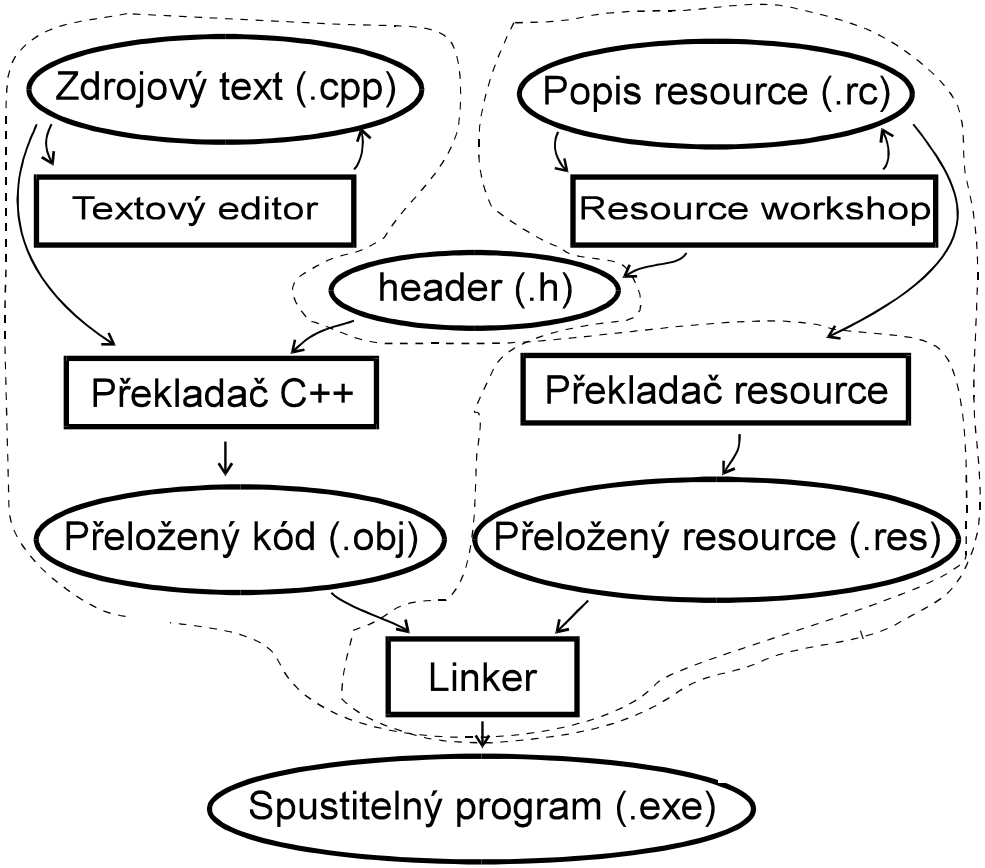
Program s „resource“



- tvar dialog boxů
- bitmapy
- ikony
- kursory
- texty
- binární data

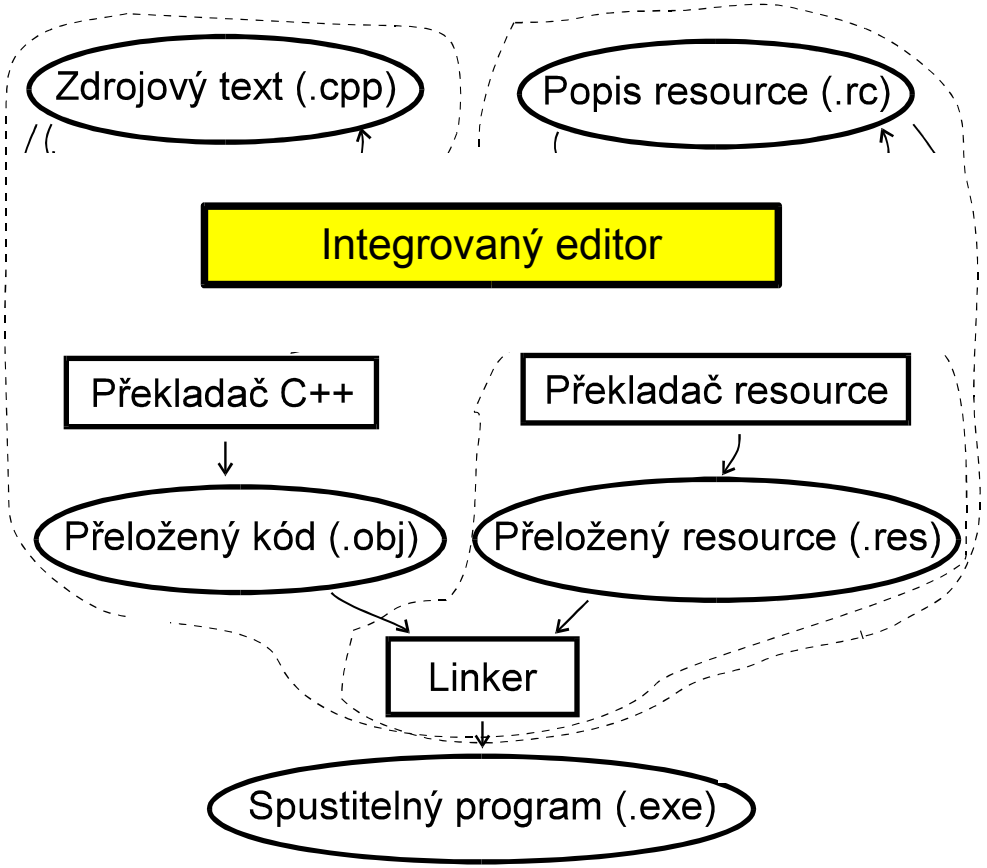
Programy ve Windows

Program s „resource“ - propojení se zdrojovým textem



Programy ve Windows

Současné prostředky – integrace editoru kódu a dat



Zprávy ve Windows

... v minulé přednášce... :-), co k tomu dodat?

```
struct Message {
  int Msg; /* Message type */
  int Window; /* Message target */
  int Param1; /* Message type dependent */
  int Param2; /* Message type dependent */
};
```

- Jak se přidělují čísla zpráv?
- Jaké jsou „předdefinované” zprávy?
- Existují „uživatelské zprávy“?

Zprávy ve Windows

Čísla zpráv:

- Globální – předdefinovaná (nelze shrnout jednoduše, existují jich tisíce, některé budou zmíněny postupně)
- Uživatelské „lokální“ (**`WM_USER`**+malá konstanta) – protože neexistuje „hlídání“ mezi aplikačními programy, lze užívat jen interně uvnitř programu
- Uživatelské „globální“ – zprávu lze zaregistrovat, je jí globálně přiděleno číslo a je zaručeno, že stejná zpráva bude mít ve všech aplikacích stejné číslo

```
UINT RegisterWindowMessage(
    LPCTSTR lpString   // address of message string
   );
```

Zprávy ve Windows

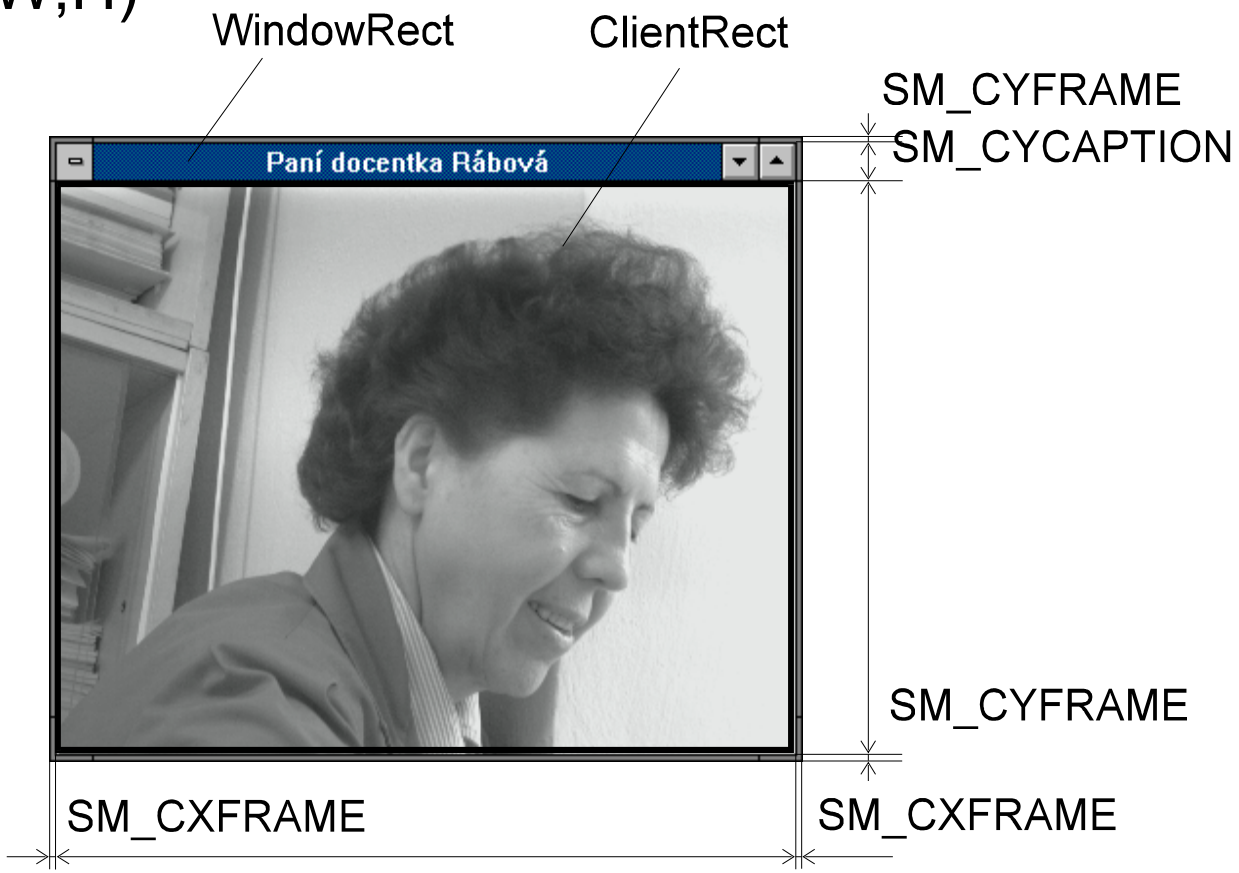
Příklady zpráv:

- **`WM_MOUSEMOVE`** – tuto zprávu dostane okno, pokud se „nad ním pohne“ myší kursor (parametr - X, Y, stav tlačítek)
- **`WM_SIZE`** – tuto zprávu dostane okno, pokud se mění jeho velikost (parametr - nová velikost)
- **`WM_KEYDOWN`** – tuto zprávu dostane okno, pokud je „pro něj“ stisknuto tlačítko na klávesnici (parametr - kód tlačítka)

Okna

Vlastnosti oken:

- Stav (normální, ikona, maximální)
- Souřadnice (X,Y,W,H)
- Klientská oblast
- Titulek
- Okraje
- Ikony
- Typ
- ...



Okna

Vybrané funkce pro geometrii oken:

- `void GetClientRect(HWND Wind,RECT FAR * Rect)`
- `void GetWindowRect(HWND Wind,RECT FAR * Rect)`
- `BOOL IsIconic(HWND Wind)`
- `BOOL IsZoomed(HWND Wind)`
- `(int GetSystemMetrics(int Index))`

- `BOOL ShowWindow(HWND Wind,int CmdShow)`
- `void CloseWindow(HWND Wind)`
- `void OpenIcon(HWND Wind)`
- `BOOL IsWindowVisible(HWND Wind)`

- `BOOL SetWindowPos(HWND Wind,HWND InsertAfter,\`
- `int x,int y,int cx,int cy,UINT Flags)`
- `BOOL MoveWindow(HWND Wind,int Left,int Top,\`
- `int Width,int Height,BOOL Repaint)`

Okna

Třídy oken – window class

„Historická záležitost“, dnes poněkud komplikuje práci :-)

```
typedef struct tagWNDCLASS {
    UINT        style;
    WNDPROC     WndProc;
    int         ClsExtra;
    int         WndExtra;
    HINSTANCE   Instance;
    HICON       Icon;
    HCURSOR     Cursor;
    HBRUSH      Background;
    LPCSTR      MenuName;
    LPCSTR      ClassName;
} WNDCLASS;
```

Okna

Vybrané funkce pro okna:

```
ATOM RegisterClass(WNDCLASS FAR * WndClass)
BOOL UnregisterClass(LPCSTR Name,HINSTANCE In)

HWND CreateWindow(LPCSTR Class,LPCSTR Name,DWORD Style,\
   int x,int y,int Width,int Height,HWND Parent,\
   HMENU DefMenu,HINSTANCE Instance,void FAR * Param)
BOOL DestroyWindow(HNWD Wind)

BOOL GetClassInfo(HINSTANCE Inst,LPCSTR Name,\
   WNDCLASS FAR *WndClassData)
LRESULT DefWindowProc(HWND Wind,UINT uMsg,\
   WPARAM WParam,LPARAM LParam)
WORD SetWindowWord(HWND Wind,int Ofs,WORD Val)
WORD SetWindowLong(HWND Wind,int Ofs,LONG Val)
WORD GetWindowWord(HWND Wind,int Ofs)
LONG GetWindowLong(HWND Wind,int Ofs)
```

Kreslení obsahu oken

Kam se kreslí? - device context:

- Device context je abstraktní kreslící oblast (osový obdélník), která se vyznačuje rozměry a atributy
- Device context není totožný s oknem – abstrakce device contextu je totiž širší a umožňuje sjednotit kreslení na displej, na tiskárny a na virtuální kreslící plochy v paměti počítače

Kreslení obsahu oken

Vybrané funkce pro device context:

```
HDC GetDC(HWND Wind)
int ReleaseDC(HWND Wind,HDC Context)

int SaveDC(HDC Context)
BOOL RestoreDC(HDC Context,int SavedDC)

HDC CreateDC(LPCSTR Driver,LPCSTR Device,\
   LPCSTR Output,const void FAR * InitData)
BOOL DeleteDC(HDC Context)

DWORD SetViewportExt(HDC Context,\
   int XExtent,int YExtent)
DWORD SetViewportOrg(HDC Context,\
   int XOrigin,int YOrigin)
DWORD GetViewportExt(HDC Context)
DWORD GetViewportOrg(HDC Context)
(int SetMapMode(HDC Context,int MapMode))
```

Kreslení obsahu oken

Důležitá fakta:

- Aktuálně zobrazený obsah oken (rastrový obraz, který je vidět na displeji) není ve Windows uchováván
- Každé okno musí být schopno „kdykoli vykreslit svůj obsah“ (děje se to na základě zprávy WM_PAINT)
- Omezení obsahu vykreslování „jen na to, co je vidět“ (clipping) je automatické a nelze jej příliš ovlivnit
- Lze aktivně omezit vykreslování na viditelnou oblast, pokud se to v programu vyplatí (neomezí se obsah samotného vykreslování, ale omezí se volání funkcí a zbytečné výpočty parametrů, související funkce je **`GetUpdateRect`**)

Kreslení obsahu oken

Jak probíhá vykreslování...:

- Windows grafický subsystém uchovává informaci o tom, která oblast kterých oken je správně vykreslena (a která ne – je neplatná – invalid region)
- Oblast se stane neplatnou například „odkrytím dříve neviditelné oblasti okna“ - na popud uživatelského rozhraní
- Oblast se může stát neplatnou též na popud aplikačního programu (mění se obsah okna) – přitom se NEKRESLÍ explicitně, ale použije se API funkce

```
InvalidateRect(
HWND hWnd, // handle of window with changed update region
CONST RECT *lpRect,  // address of rectangle coordinates
BOOL bErase   // erase-background flag
);
```

Kreslení obsahu oken

...jak probíhá vykreslování:

- Windows grafický subsystém „občas“ (“rozumně často”) vykreslí obsah neplatných částí oken (tím přestanou být neplatnými)
- Neplatné části oken se kumulují – několik zneplatnění nemusí vést ke stejnému počtu vykreslení
- V případě potřeby (například při zásadním přeuspořádání mnoha oken) lze kreslení dočasně potlačit funkcí **`LockWindowUpdate`**
- Okno lze „donutit“ k okamžitému překreslení funkcí **`UpdateWindow`**
- „Přímé a okamžité“ kreslení do oken naní zakázáno, je jej však vhodné velmi důkladně zvažovat a využít výjimečně

Kreslení obsahu oken

Vybrané kreslící funkce (opakování z IZG, nezapomeňte na nástroje, paletu apod.):

```
COLORREF SetPixel(HDC Context,int XPos,int Ypos,COLORREF Col)

COLORREF GetPixel(HDC Context,int XPos,int YPos)

DWORD MoveTo(HDC Context,int X,int Y)
BOOL LineTo(HDC Context,int X,int Y)

BOOL Rectangle(HDC Context,int Left,int Top,int Right,int Bot)

BOOL Polyline(HDC Context,const POINT FAR Points,int Num)

BOOL FloodFill(HDC Context,int XStart,int Ystart,COLORREF Col)


int StretchDIBits(HDC Context,int XDest,int YDest,int WDest,\
   int HDest,int XSrc,int YSrc,int WSr,int HSr,void FAR * Bi,\
   LPBITMAPINFO Info,UINT ColorUse,DWORD Rop)

BOOL TextOut(HDC Cotext,int XStart,int YStart,\
   LPCSTR String,int StringLength)
```

Okna - zprávy

Základní zprávy spojené s okny:

`WM_ACTIVATE` - Zpráva je zasílána oknu tehdy, když je aktivováno nebo deaktivováno.

`WM_CLOSE` - Má-li se okno uzavřít, je mu vyslána tato zpráva.

`WM_CREATE` - Tuto zprávu obdrží okno v okamžiku, kdy jsou jeho datové struktury již vytvořeny, ale kdy ještě není zobrazeno.

`WM_DESTROY` - Zpráva přijde při rušení okna ihned po odstranění okna z obrazovky, ale ještě před zrušením jeho datových struktur.

Okna - zprávy

Základní zprávy spojené s okny:

`WM_ERASEBKGND` - Tato zpráva je zaslána oknu v okamžiku, kdy je potřeba připravit jeho podklad před překreslováním.

`WM_GETMINMAXINFO` - Chystá-li se změna velikosti nějakého okna, dostane toto okno uvedenou zprávu a má možnost změnit implicitní minomální i maximální velikost.

`WM_HSCROLL` - Při horizontálním "scrollování" okna do něj přichází tato zpráva s údajem o počtu kroků (i VSCROLL).

`WM_KILLFOCUS` - Předtím, než okno ztratí vstupní "focus" dostane tuto zprávu.

Okna - zprávy

Základní zprávy spojené s okny:

`WM_MOVE` - Je-li s oknem pohnuto, ja mu zaslána tato zpráva.

`WM_PAINT` - Tato zpráva je oknu zaslána v okamžiku, kdy se má překreslit

`WM_PALETTECHANGED` - Informace oknu, že jiné okno realizovalo paletu funkcí RealizePalette.

`WM_PALETTEISCHANGING` - Informace oknu, že jiné okno se chystá realizovat novou logickou paletu funkcí RealizePalette.

Okna - zprávy

Základní zprávy spojené s okny:

`WM_QUERYNEWPALETTE` - Tato zpráva představuje dotaz, jestli program bude realizovat svou paletu poté co získá "focus".

`WM_SETFOCUS` - Touto zprávou se oknu sděluje, že dostalo "input focus".

`WM_SIZE` - Změní-li se velikost okna, je mu zaslána tato zpráva.

`WM_CHAR` - Tato zpráva vzniká na základě zpracování zprávy WM_KEYDOWN nebo WM_KEYUP. Obsahuje kromě virtuálního kódu také informace jako počet opakování při "autorepeatu", "scan code"...

Okna - zprávy

Základní zprávy spojené s okny:

`WM_LBUTTONDBLCLK` - Doubleclick na levém myším tlačítku (může přijít jen po WM_LBUTTONDOWN).

`WM_MBUTTONUP` - Uvolnění středního myšího tlačítka.

`WM_SYSKEYDOWN` - Tato zpráva je vyslána po stisku klávesy v systémovém menu (nezpracováno).

Okna – Default funkce

Default funkce okna DefWindowProc:

- Je volána pokud není jinak obsloužena zpráva pro okno (může se volat i explicitně, pokud se ukáže, že se „nemusí nic zvláštního dělat)
- Provede podle typu okna obsluhu všech předdefinovaných zpráv

```
LRESULT DefWindowProc(
    HWND hWnd,   // handle to window
    UINT Msg,// message identifier
    WPARAM wParam,  // first message parameter
    LPARAM lParam   // second message parameter
   );
```

Shrnutí

- Okna jsou základním stavebním prvkem aplikací ve Windows
- Zprávy jsou základním prostředkem komunikace
- Chování oken je definováno reakcemi na zprávy

Příklady textů užité při přednášce

```
void HandleMessage(Message * Msg)
{
 switch (Msg->Type)
 {
  case WM_PAINT:
   ...kreslení grafu...
   ...použij XSize a Ysize...
   break;
  case WM_SIZE:
   XSize = Msg->Param1;
   YSize = Msg->Param2;
   InvalidateRect(0,0,XSize,YSize);
   break;
  default:
   DefWindowProc(Handle);
 }
}
```

Obsluha zpráv v okně zobrazujícím graf, který vyplňuje celé okno - velikost grafu je v proměnných XSize a YSize

```
class TWindow
{
...
virtual void Paint();
virtual void Size(int X,int Y);
}

class TGraphWindow: public TWindow
{
 ...
 virtual void Paint()
 {
  ...kreslení grafu...
  ...použij XSize a YSize...
 }
 virtual void Size(int X,int Y)
 {
  XSize = Msg->P1, YSize = Msg->P2;
  InvalidateRect(0,0,XSize,YSize);
 }
};
```

Jak to lze provést v objektovém „kabátě“

Příklady textů užité při přednášce

```
BegTable(TWindow,TGraphWindow,1)
SetMessage(MyMessage,WM_USER+10);
EndTable()
```

Zpráva	Metoda
Z TWindow...	Z TWindow...
...z TWindow...	...z TWindow...
WM_USER+10	&MyMessage

„Zděděné“ řádky

Jak se ale „napojí“ uživatelská zpráva, o níž designer knihovny nevěděl? Konstrukce tabulky zpráv

```
int Functions[1000];
int Types[1000];
int Max;

void HandleMessage(Message * Msg)
{
 for (i = 0;i<Max;++i)
 {
  if (Msg->Type==Types[i])
  {
   Function[i](Msg);
   return;
  }
 }
 DefUserProc();
 DefWindowProc(Handle,Msg);
}
```

Funkce pro zpracování zpráv na základě tabulky i s „default“ cestou