

Operační systémy 2

Základy Windows API

Aleš Keprt

Univerzita Palackého

říjen 2008, říjen 2010

Dnešní program

- Visual Studio 2005/2008
 - Základy programování ve Windows
 - Přesněji „Win32 API“ či „Windows API“
- Dnešní programování: Práce s textem

Visual Studio 2005/2008

- Základní jazyk programování ve Windows: C++
 - Nejsou tam téměř objekty, takže funguje i C
- Základní nástroj: Visual Studio 2005/2008
- File / New / Project / C++ / Win32
 - Win32 Project / Application
 - Standardní aplikace s grafickým prostředím
 - Event-driven (programování s událostmi)
 - Toto ve cvičení OS2 používat nebudeme
 - Win32 Console Project / Application
 - Konzolová aplikace (jen text, tj. bez grafiky)
 - Klasický (starobylý) styl programování
 - Toto budeme používat ☺

Application settings

- Create directory for solution
 - Toto vypněte (jinak by se vytvořil adresář navíc)
- Precompiled header
 - Zapne používání stdafx.h
 - Zrychlí kompilaci, ale je to „Microsoft-specific“
 - Toto zapněte a do stdafx.h dopište

```
#define WIN32_LEAN_AND_MEAN
#include <windows.h>
```
- Empty project
 - Vytvoří úplně prázdný projekt (bez souborů)
 - Sami pak musíte napsat main(), apod.

Literatura

- Programování ve Windows dnes už jen v C#
- Pro jazyk C++ jsou prakticky jen staré knihy
- Doporučená četba:
 - Petzold: *Programming Windows*. (5.vydání)
Microsoft Press, 1998.
 - Jeffrey Richter: *Programming Applications for Microsoft Windows*. (4.vydání)
Microsoft Press, 1999.
 - Mark Russinovich, David Solomon: *Microsoft Windows Internals*. (4.vydání)
Microsoft Press, 2004.

Některé rozdíly Win NT a Win 95

- NT podporuje více procesorů, 95 jen jeden
- NT podporuje bezpečnost, 95 ne
- NT je plně 32bitový (či 64bitový), 95 je 16bitový
- NT je plně reentrantní, 95 používá system lock
- NT pouští 16bitový kód v odděleném prostoru, 95 jej pouští v globálním prostoru
- NT sdílí paměť jen mezi určenými procesy, 95 sdílí paměť mezi všemi procesy
- NT chrání jádro systému proti zápisu, 95 ne
- NT má jen emulátor MS-DOSu, 95 spouští programy pro MS-DOS přímo

Stručně o Windows Vista UAC

- UAC = User Account Control
 - Česky „řízení uživatelských účtů“
- Procesy běží s oprávněním skupiny Users
 - I když jste administrátoři, nemáte vyšší práva
- Staré aplikace běží ve virtuálním režimu
 - Přesměrování c:\Program Files a c:\Windows
 - Přesměrování větví registru HKEY_ROOT a HKEY_LOCAL_MACHINE
- Nové aplikace jsou označeny manifestem
 - Manifest určuje požadovanou úroveň pro běh
 - Zároveň se tím vypne virtualizace

Texty ve Windows

- Typ `char` = 1 bajt → MBCS (ANSI/OEM)
 - České znaky používají „Windows 1250“
 - MBCS může být i vícebajtové (jako UTF-8)
- Typ `wchar_t` = 2 bajty → DBCS (Unicode)
 - Ve Windows platí, že `wchar_t` je v kódu UTF-16
- Programy lze v C++ psát tak, aby šly přeložit do MBCS i DBCS (1 zdroják → 2 „ekzáče“)

Jak přežít wchar_t...

- První možnost: vypnout jej
 - Project Properties / Character Set
 - Zvolíme MBCS (multi-byte character set)
- Druhá možnost: nechat zapnuté a používat
- Třetí možnost: psát univerzální kód

Univerzální kód

- Všechny funkce pracující s textem mají vždy 2 verze (lišící se názvem) plus makro
- Makro vede na jednu z těchto funkcí, dle výběru v Project Options
- Všechna tato makra jsou v souboru [tchar.h](#)
- `char` `wchar_t` `_TCHAR`
- `main` `wmain` `_tmain`
- `printf` `wprintf` `_tprintf`
- `strcpy` `wcscpy` `_tcscpy`
- `strcmp` `wcscmp` `_tcscmp`

Univerzální kód a API

- Windows API je univerzální vždy
- Všechny dotčené funkce mají:
 - MBCS verzi s 'A' na konci
 - DBCS verzi s 'W' na konci
 - Makro bez přípony
- Příklad: [CreateFileA](#), [CreateFileW](#), [CreateFile](#)
- Na toto pozor! Při chybě debugger píše skutečné jméno funkce, ne jméno makra!!

Podrobnosti k printf

- Formátovací řetězce %s a %S jsou opačné:
 - `printf` : %s je `char*`, %S je `wchar_t*`
 - `wprintf`: %s je `wchar_t*`, %S je `char*`
- Výstup do konzoly je vždy v kódu OEM
 - OEM = kódování MS-DOSu (tj. CP852)
 - Otravný přežitek z minulosti
- Konverze `tstring` $\leftarrow \rightarrow$ OEM:
 - `CharToOem()`
 - `OemToChar()`

System handlů

- Libovolný objekt se ve Win32 identifikuje pomocí tzv. „hendlu“ (anglicky handle – rukojeť/držátko)
- Handle je něco jako „this“ pointer na nějaký objekt
 - Win32 API je jakoby objektově orientované, ale je slabě typované
 - Každý handle je v C++ ve skutečnosti typu void*
- Kromě handlů potřebujeme ještě security descriptor – většinou se ale s bezpečností neoperuje, takže dáváme NULL a máme výchozí chování (jako ve Windows 95)
- Na závěr handly uvolníte pomocí `CloseHandle()`

Další datové typy Windows API

- Windows API má vlastní datové typy
 - tj. nepoužívá jména typů jazyka C
- Jsou to všechno zkratky (nutno naučit!)
- Příklady:
 - **LPSTR** = (long) pointer to a string
 - každý pointer začíná LP, takže toto je prostě **char***
 - **LPCTSTR** = (long) pointer to a constraint tstring
 - toto je jednoduše **const _TCHAR***
 - **HANDLE** je obecný hendl
 - **HWND** = handle to a window
 - **INT_PTR** = integer for pointer precision
 - toto je integer o velikosti pointeru

Bezpečné CRT ve Visual Studiu

- VS2005/2008 používá bezpečnou CRT knihovnu
- Je to rozšíření ANSI C, čeká se na standardizaci
- Týká se to funkcí, kde se pracuje s bufferem
- Čtěte v MSDN: „security-enhanced CRT“
- Funkce obvykle mají:
 - Původní funkce se nemění
 - Nová bezpečná funkce má na konci názvu `_s`
 - Navíc je délka bufferu jako parametr
 - Při použití `char[]` lze délku vynechat
 - př: `strcpy_s(do_pole, odkud);` nebo `strcpy_s(do_pointeru, odkud, délka);`

Úlohy pro cvičení

1. Načtěte text z klávesnice, uložte do souboru ve Windows 1250
2. Načtěte soubor (viz výše) a vypište na obrazovku
3. fopen umí pracovat i s unicode – vyzkoušíme
Načtěte a zobrazte Unicode soubor na obrazovce.
(Soubor si vytvořte v Notepadu.)

Poznámka: Nejprve si přečtěte „fopen“ v MSDN. Kompilaci si pochopitelně přepněte do MBCS, tj. bez unicode. Konzola zobrazí jen češtinu, ale cizí unicode znaky ne (omezení by design).

© Mgr. Aleš Kepřt, Ph.D., 2006,2007,2008,2010

Vytvořeno pro potřeby výuky na UP Olomouc. Tento text není určen pro samostudium, ale jen jako vodítko pro přednášku, takže jeho obsah se může čtenáři zdát stručný, nekompletní či možná i chybný. Použití je povoleno dle vlastní libosti, ale jen na vlastní nebezpečí. ☺ V případě dalšího šíření je NUTNO uvádět původního autora a odkaz na původní dokument. Komentáře můžete posílat e-mailem autorovi (adresu najdete pomocí Googlu).