KATEDRA
INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

Základní práce s vlákny ve windows

KMI/OS1 Operační systémy I

Mgr. Markéta Trnečková, Ph.D.
www.marketa-trneckova.cz

Základní práce s vlákny ve Windows

Učební text ke cvičení:
`https://phoenix.inf.upol.cz/~krajcap/courses/2025LS/OS1/tutorial10.pdf`

Vlákna

- ve Windows je základní jednotkou vykonávající program **vlákno**
- v každém procesu může běžet více vláken
- vlákno může vykonávat libovolnou část kódu programu
- při spuštění procesu je spuštěno (primární) vlákno – vykonává funkci `main`
- každé vlákno má vlastní kontext – má přidělen zásobník a registry
- další prostředky (paměť a systémové prostředky, ...) jsou sdíleny mezi všemi vlákny
- to umožňuje pohodlnou spolupráci více vláken
- **pozor!** Běh jednotlivých vláken je plánován OS
- činnosti, které vlákna provádí se mohou prolínat
- je nutné zajistit, aby v jednom okamžiku s jedněmi daty nepracovalo více vláken současně
- toho můžeme dosáhnout pomocí synchronizace vláken nebo vhodně navrženým kódem

Vytvoření vlákna

- **vytvoření vlákna** pomocí funkce `CreateThread`
  `https://docs.microsoft.com/en-us/windows/win32/api/processthreadsapi/nf-processthreadsapi-createthread`
- argumenty:
  - atributy definující vlastnosti vlákna
  - odkaz na funkci, která představuje kód vykonávaný vláknem
  - odkaz na data, se kterými vlákno pracuje

Příklad

Podívejte se na zdrojový kód `vlakna.c`.

Vytvoření vlákna

Příklad

```
...
#define COUNT (20)

DWORD WINAPI ThreadFunc(LPVOID lpParam){
    _tprintf(_T("Spusteno vlakno s id: %i\n"), GetCurrentThreadId());
    DWORD x = *(DWORD *)lpParam;
    for (int i = 0; i < COUNT; i++) {
        _tprintf(_T("thr #%i: %i\n"), x, i);
        Sleep(1);
    }
    return 0;
}

int _tmain(){
    DWORD dwThreadId;
    DWORD dwThrdParam = 1;

    HANDLE hThread = CreateThread(
            NULL, // bezpecnostni atributy
            0, // velikost zasobniku (0 -> implicitni hodnota)
            ThreadFunc, // funkce provadena vlaknem
            &dwThrdParam, // argument predany vlaknu
            0, // priznaky pro vytvorene vlakno
            &dwThreadId); // vraci id vlakna

    if (hThread == NULL) {
        _tprintf(_T("Vytvoreni vlakna selhalo\n"));
        ExitProcess(0);
    } else {
        _tprintf(_T("Vytvoreno vlakno s id: %i\n"), dwThreadId);
    }
    for (int i = 0; i < COUNT; i++) {
        _tprintf(_T("thr #0: %i\n"), i);
        Sleep(1);
    }

    WaitForSingleObject(hThread, INFINITE); // ceka na skonceni vlakna
    CloseHandle(hThread); // ukonci praci s vlaknem

    return 0;
}
```

Vytvoření vlákna

- `CreateThread` – 3. argument udává funkci, kterou bude vlákno vykonávat
- `CreateThread` – pomocí 4. argumentu předáváme vláknu data pomocí ukazatele
- `CreateThread` – 6. argument slouží ke zjištění identifikátoru vlákna
- `CreateThread` – vrací ukazatel na dané vlákno, se kterým můžeme dále pracovat
- měli bychom otestovat, zda se vlákno podařilo vytvořit (pokud ne handle je `NULL`)
- měli bychom počkat na dokončení běhu vlákna (`WaitForSingleObject`
- měli bychom ukončit práci s vláknem `CloseHandle`
- v příkladu ve funkci `main` (v primárním vláknu) vypisujeme ve smyčce čísla od `0` do `COUNT` a po každém výpisu proces uspíme na 1 ms (`Sleep`)
  `https://docs.microsoft.com/en-us/windows/win32/api/synchapi/nf-synchapi-sleep`

Vytvoření vlákna

- funkce vykonávající vlákno je typu `DWORD WINAPI ThreadFunc(LPVOID lpParam)`
- `lpParam` je ukazatel předaný funkcí `CreateThread`
- v těle funkce je možné vykonávat libovolný kód
- v našem příkladu vypíšeme id vlákna
- při spuštění programu si můžeme všimnout, že moment vytvoření a spuštění vlákna nemusí být stejný
- projeví se to tak, že primární vlákno vstoupí do smyčky a až následně dojde ke spuštění vlákna, které až pak začne vypisovat text
- funkce `ThreadFunc` vrací hodnotu typu `DWORD`, kterou můžeme předat informaci související s ukončením vlákna
- tuto hodnotu můžeme vyzvednout pomocí funkce `GetExitCodeThread` `https://docs.microsoft.com/en-us/windows/win32/api/processthreadsapi/nf-processthreadsapi-getexitcodethread`

Vytvoření vlákna

Příklady

Příklad

Odstraňte z kódu volání `Sleep`, případně změňte jeho argumenty, a sledujte, jak se změní průběh programu.

Příklad

Rozšiřte ukázkový program tak, aby vyzvedl a vypsal návratovou hodnotu spuštěného vlákna. Zamyslete se nad tím, kam volání funkce `GetExitCodeThread` umístit.

Příklad

Rozšiřte ukázkový program, aby pracoval obecně s `N` vlákny, kde `N` je konstanta zadaná v kódu programu.

Manipulace s vlákny

Další funkce pro práci s vlákny

- `GetCurrentThread` – vrací pseudo-handle sloužící k manipulaci s aktuálním vláknem. Pseudo-handle zde funguje jako zvláštní konstanta odkazující na aktuální vlákno a má platnost pouze v rámci daného vlákna.
  `https://docs.microsoft.com/en-us/windows/win32/api/processthreadsapi/nf-processthreadsapi-getcurrentthread`
- `DuplicateHandle` – převádí pseudo-handle na plnohodnotný handle
  `https://docs.microsoft.com/en-us/windows/win32/api/handleapi/nf-handleapi-duplicatehandle`
- `SuspendThread` – slouží k uspání vlákna. Může být opakovaně zavolána i na již uspané vlákno. Každé vlákno má přiřazeno počítadlo, které indikuje počet uspání daného vlákna
  `https://docs.microsoft.com/en-us/windows/win32/api/processthreadsapi/nf-processthreadsapi-suspendthread`
- `ResumeThread` – slouží k probuzení vlákna. Pokud je vlákno uspáno víckrát, snižuje počítadlo uspání, a pokud toto počítadlo klesne na 0, je vlákno probuzeno.
  `https://docs.microsoft.com/en-us/windows/win32/api/processthreadsapi/nf-processthreadsapi-resumethread`

Manipulace s vlákny

Další funkce pro práci s vlákny

- `ExitThread` – ukončí právě prováděné vlákno.
  `https://docs.microsoft.com/en-us/windows/win32/api/processthreadsapi/nf-processthreadsapi-exitthread`
- `TerminateThread` – ukončí jiné vlákno na základě předaného handlu. Ukončení vlákna touto funkcí není korektní, může k němu dojít v libovolném bodě daného vlákna, a tím pádem se může program dostat do nekonzistentního stavu.
  `https://docs.microsoft.com/en-us/windows/win32/api/processthreadsapi/nf-processthreadsapi-terminatethread`

Příklady

Příklad

Ukázkový příklad upravte tak, že po provedení `(COUNT / 2)` cyklů se vlákno uspí a je probuzeno v momentě, kdy primární vlákno zpracuje celou smyčku.

Příklad

Naprogramujte funkci `int parfib(int)`, která spočítá Fibonacciho číslo rekurzivním způsobem s využitím právě dvou vláken. Dvě počáteční větve výpočtu spusťte v samostatných vláknech.

Příklad

Naprogramujte funkci `int pmin(int *numbers, unsigned int count, unsigned int threads)`,která použije `threads` vláken k tomu, aby v poli `numbers`, které obsahuje `count` hodnot, našla nejmenší hodnotu.