File Manager

6.7. 2018
Autor: David Burian

Přehled

Mým cílem je vytvořit jednoduchý ale efektivní program na pracování se soubory. Efektivity by mělo být dosaženo skrz použití více vláken, různých heureistik. S důrazem na jednoduchost bude tento program realizován pomocí Windows Forms s velmi minimálním grafickým rozhraním.

Cíle programu

1. Program by měl efektivně kopírovat, mazat, přesouvat a vyhledávat soubory.
2. Program by uživateli měl nabízet třízení dle různých parametrů soboru v aktuálním adresáři.
3. Uživateli by mělo být umožněno efektivně přejmenovávat větší množství souborů najednou.
4. Program by měl poskytnout k práci dvě okna, ve kterých uživatel bude moci navigovat ve dvou adresářích.

Požadavky

1. Pokud uživatel bude chtít kopírovat nebo přesouvat soubory, bude to z jednoho adresáře, do druhého. Tedy nebude umožněno kompírovat do více adresářů zároveň.
2. Hromadné přejmenovávání souborů, bude umožněno na základě číslování souborů s možností definování předpony tj. jméno souboru před číslem.

Úvaha nad řešením

- Grafické rozhraní
    - Uživatel bude potřebovat prvky grafického rozhraní, které spustí již dříve zmíněné akce, tedy:
        - Kopírování souborů
        - Mazání souborů a adresářů
        - Přesouvání souborů
        - Přejmenování souboru/ů
        - Třídit soubory v rámci jednoho adresáře
        - Vytvářet adresáře

      plus prvky navigace, tedy přesun do rodičovského adresáře.
    - Ikdyž nabízených funkcí je málo a dalo by se například navrhnout rozumné grafické rozhraní na bázi tlačítek (buttons), je pravděpodobné, že množina nabízených funkcí se bude zvětšovat, a proto bude lepší grafické rozhraní implementovat jako menu.
    - Grafické rozhraní bude tedy děleno na klasický menu bar a dvě okna pro práci se soubory.
    - Každé okno by mělo zobrazovat aktuální adresář, všechny subory v něm a několik základních atributů souborů, které omezím na velikost,

typ souboru, datum poslední modifikace a datum vytvoření. Ostatní atributy se snadno doimplementují.
    - Protože přesun do rodičovského adresáře bude při průměrném používání file manageru velmi častý, bude lepší tuto akci implementovat ve snadno dosažitelném tlačítku.
    - Hromadné přejmenování je složitější akce, kterou by bylo přehlednější implementovat pomocí nějakého pop-up okna, které by se uživatele zeptalo na:
        - Předponu jména souboru
        - Začáteční příponu jména souboru ve formě číslice, která se pro každý soubor navýší
        - Výběr třídění, dle kterého se bude při přejmenovávání bude postupovat
    - V případě kopírování, mazání nebo přesouvání více souborů může akce trvat dlouho a uživatle by toto spoždění mohlo mást. Proto by bylo nejlepší uživateli poskytnout nějakou formu zpětné vazby.

- Používání vláken a heureistiky
    - Vícevláknové práce se soubory nejsou vždy nejrychlejší, proto je potřeba se rozhodnout, zda použít pro práci více vláken a pokud ano, tak kolik.
    - Rozhodování o tom, kolik vláken pro práci použít, musí být rychlé, aby například vyhledávání ve dvou kratičkých dokumentech netrvalo zbytečně dlouho. Proto implementace rozhodovacího algoritmu, bude používat heureistiku, tedy rozhodování bez 100% pravděpodobností správného rozhodnutí.
    - Je potřeba vzít do úvahy konkrétní parametry systému uživatele. Například na dvoujádrovém procesoru nebude efektivní (a rychlé) vytvářet tolik vláken pro práci, jako na osmijádrovém procesoru.
    - Práce se soubory nesmí zatěžovat schopnost interaktivity grafického prostředí, proto práce se soubory si vyžádá vždy alespoň jedno vlákno.
    - Není pravděpodobné, že uživatel bude pracovat se dvěmi okny zároveň, naopak se čeká, že mezi nimi bude často přepínat. Proto grafické rozhraní obou oken bude v jednom vlákně.
    - Rychlost některých prací se soubory bude silně limitována, rychlostí rozhraní paměťového systému. Proto vícevláknové kopírování, přesouvání a mazání neposkytuje zrychlení oproti jednovláknové variantě. Tedy tyto funkce bude program implementovat jednovláknově.

Čím se nebudu zabývat

- Program nebude řešit optimalizace na úrovni celého systému. Nebude například brát v úvahu zatíženost jader nebo paměti.
- Program nebude nabízet funkce plnohodnotného file manageru, jako označování souborů flagy, vytváření nových souborů, funkci "Zpět," rychlou selekci adresářů – "Favorites," zobrazovnání vlastních uživatelem upravených ikon, trvalé smazání souborů, otvírání souborů,…
- Program nebude nabízet propracované grafické rozhraní, ani jeho upravení.
- Program nebude brát v úvahu možné kolize jmen souborů. Pokud při kopírování, přesunu souborů bude již v destinačním adresáři soubor se stejným jménem, bude přepsán.

Otevřené otázky

- Kolik maximálně souborů bude moct program zpracovat?
- Bude selekce souborů pro práci limitovaná do jedonoho adresáře?
- Jaké heureistiky bude rozhodovací algoritmus používat?
- V jaké formě bude program poskytovat zpětnou vazbu při přesunu, kopírování a mazání souborů?

Komponenty programu

Nejjednodušší bude oddělit grafické rozhraní a kód, který bude pracovat se soubory. Vzhledem k pravděpodobnému budoucímu využití/vylepšení kódu, který pracuje se soubory, rozdělím celou aplikaci na dvě části:

1. Grafické rozhraní
2. Knihovna funkcí pro práci se soubory

1. Grafické rozhraní

Grafické rozhraní bude naimplementováno pomocí Windows Forms, které uživateli poskytují již předem vytvořené interaktivní prvky, na jejichž spuštění(event) generují volání metody(event handler). Event handler pak spustí jiné vlákno, které bude pracovat s kódem z knihovny funkcí (2.). Zpětné vazby, které budou pravděpodobně krátkodobé (např.: zobrazení složek) budou řešeny pomocí metody .Invoke (nebo .BeginInvoke) na grafickém prvku.

Pro kopírování, přesouvání a další práce se soubory, které vyžadují, aby uživatel dostal zpětnou vazbu event handler vyvolanému vláknu poskytne datovou strukturu, která bude reprezentovat proces, který vyvolané vlákno provádí. Zobrazení této datové struktury uživateli pak bude zajišťovat zvláštní vlákno. Tímto způsobem se nebude zahlcovat vlákno, které pracuje s grafickým rozhraním a zároveň se logicky oddělí způsob zobrazování průběhu operace od implementace samotné operace.

2. Knihovna funkcí pro práci se soubory

Díky nezávislosti knihovny funkcí, bude tato část programu velmi jednoduchá. Tato knihovna bude poskytovat základní API pro všechny výše zmíněné funkce.

Součástí této knihovny bude i definice datové struktury, která bude reprezentovat probíhající proces práce se soubory.