OSY Home 3

Deadline: 31.10.2025, 0:00, Time remaining: 2 hours, 3 minutes left

Submit |< < 2 / 2 > >| - Adam Vlček (VLC028), 30.10.2025 21:54

Select multiple files using CTRL or SHIFT or **drag them to this window**

Assignment | Source code | Upload

Úkol

Pro přípravu použijte připravený soketový server a klienta z github, adresář `socket-demo`. Tyto příklady se budou používat častěji a budou k dispozici i u zkoušky.

Použijte obrázek podzimu.

1. Upravte soketový server tak, aby byl schopen komunikace s více klienty. Pro každé nové spojení si rodičovský proces vytvoří potomka (někde po `accept`u bude fork) a ten se bude starat o spojení s klientem. Stávající kód je potřeba rozdělit tak, aby se rodičovský proces staral jen o příjem klientů a potomci jen o komunikaci s nimi. Ideálně si kód pro potomky dejte do samostatné funkce.

   Nezapomínejte, že stále platí: za fork každý proces zavírá co už nepotřebuje!

   Řešení ověřte zasíláním zpráv z klientů na server. Zprávy ze serveru už nebude možno zasílat.

2. Pro úpravu obrázků lze použít program convert (součástí ImageMagick). Vyzkoušejte si `convert -resize 1500x750! podzim.png out.png ; display out.png`. Lze také provést `convert -resize 1500x750! podzim.png - | display -` a to bez ukládání do souboru.

   Z klienta zašlete na server požadováné rozlišení, např `1500x750`. Na serveru vytvořte další proces - potomka, v něm proveďte `exec` programu `convert -resize 1500x750! podzim.png -` a
   jeho `stdout` přesměrujte do soketu (žádná roura ještě není potřeba, nevymýšlejte AI hlouposti!). Výsledek bude odeslán klientovi. Rodič na ukončení procesu čeká! (U klienta se začnou chaoticky vypisovat přijatá data, obrázek nelze zobrazit na stdout.)

3. Upravte si soketového klienta tak, aby akceptoval 3 parametry, kde třetí parametr bude požadované rozlišení. Toto rozlišení se ihned po navázání spojení odešle na server ve tvaru např. `1400x700\n`.

   Dále z kódu odstraňte použití funkce `poll`.

Všechna data přijatá ze serveru klient uloží do souboru `image.img`. Přenos dat končí uzavřením soketu (read vrátí 0, pokud se read *zasekne*, není soket na serveru uzavřený). Pokud budou přijatá data správná, mělo by být možno zobrazit obrázek příkazem `display image.img` s příkazového řádku.

4. Upravte server tak, aby se obrázek odesílal komprimovaný: `convert -resize 1500x750! podzim.png - | xz - --stdout`. Dva procesy - potomci si obrázek přepoškou rourou a `xz` bude mít přesměrovaný `stdout` do soketu.

5. Na klientovi si přijatá data můžete vyzkoušet zobrazit: `xz -d image.img --stdout | display -`.

   Klienta upravte tak, aby po uložení souboru `image.img` dva příkazy `xz .... | display -` provedl ve dvou potomcích propojených rourou. Rodič na potomky čeká!. Po zavření okna s obrázkem soketový klient končí.

   Celkově si příkazy můžete ověřit: `convert -resize 1500x750! podzim.png - | xz --stdout | xz -d --stdout | display -` (konverze | komprese | dekomprese | zobrazení).