HTTP Server s výpisem adresářů

Úlohou je vytvořit jednoduchý HTTP server, který navíc poskytuje výpis adresáře.

Parametry serveru

- V programu se nepoužíva GUI (pro výpis ladících informací použijte konzoli, pro interakci se serverem použijte webový prohlížeč).
- Pri startu serveru se z příkazového řádku načítá:
  - číslo portu, na kterém bude server přijímat spojení z vašeho webového prohlížeče
  - kořenový adresář, ve kterém se nachází publikovaný obsah
- `java Server -p <TCP_PORT> -r <ROOT_DIR>`
  - Příklad: `java Server -p 8080 -r data`
  - Příklad: `java Server -p 80 -r /var/www`

Komunikační protokol

- Server akceptuje připojení klientů na definovaný **TCP** port.
- Po akceptovaní spojení bude server obsluhovat klienta v samostatném vlákně.
  - **Poznámka:** Vlákna mezi sebou nekomunikují a nemusí se synchronizovat.
- Pro komunikaci se používá zjednodušený protokol HTTP 1.0
- Server podporuje pouze jedinou HTTP metodu, a to metodu GET
- Komunikace probíhá ve 2 fázích:
  1. Nejprve klient (webový prohlížeč) pošle na server dotaz (HTTP request)
     Dotaz obsahuje **URL** požadovaného souboru. URL dotazu může obsahovat kromě **cesty k požadovanému souboru** taky parametry uvedené za znakem „?“ a oddělené znakem „&“ (tzv. **Query string**)
  2. Potom server pošle klientovi odpověď (HTTP response)
     Odpověď obsahuje tzv**. status kód**. (my budeme používat pouze některé, viz. dále) a samotná **data odpovědi** (například obsah .jpeg souboru), vše ve formě „textu“ (je to sekvence bytů).
- V rámci protokolu HTTP 1.0 server po odeslání odpovědi klientský socket zavře. Pro každý soubor (např. obrázky v rámci jedné stránky) se navazuje nové spojení.

DŮLEŽITÉ:
Součástí zadání je i **KNIHOVNA IMPLEMENTUJÍCÍ DANÝ PROTOKOL**, takže ho **NEMUSÍTE IMPLEMENTOVAT SAMI**. Knihovnu dostanete jako JAR soubor.

Rozhraní knihovny:

Rozhraní je implementováno v packagi `HttpLibrary`.

Samotný protokol je implementován statickou třídou `HttpProtocol`. Ta obsahuje metody pro čtení dotazů a zapisování odpovědí.

- `public static HttpRequest readRequest(InputStream in)` – přečte z daného vstupního streamu jeden HTTP request a vrátí ho ve formě instance třídy `HttpRequest`, která obsahuje všechny potřebné informace. Může se zablokovat do té doby, dokud nebude co z daného streamu číst.
- `public static void writeResponse(HttpResponse response, OutputStream out)` – zapíše do daného výstupního streamu odpověď předanou ve formě instance třídy `HttpResponse`, kterou vyplníte všemi potřebnými informacemi (status kód a stream obsahující data odpovědi).

Detaily viz. komentáře a zdrojový kód.

Statický obsah

- HTTP dotaz obsahuje název souboru, jehož obsah posílá server klientovi

- Cesta k souborům je relativní vzhledem ke kořenovému adresáři (parametr serveru)

Příklad:

Kořenový adresář je: `/data`
Hlavička HTTP dotazu je: `GET /img/image.jpg HTTP/1.1`
Server vrátí obsah souboru: `/data/img/image.jpg`

Poznámka:

1. Pokud požadovaný soubor neexistuje, server odpoví prázdným tělem se status kódem odpoví status kódem 404 a tělem „404 Not Found“
2. Pokud server nemůže daný soubor číst, odpoví status kódem 403 a tělem „403 Forbidden“

Vypisování adresářů

Pokud HTTP dotaz obsahuje cestu k adresáři, vygeneruje server v odpovědi HTML kód (tzv. index), který bude reprezentovat setřízený obsah tohoto adresáře (parametr, podle kterého má být výpis setříděn bude součástí URL dotazu).

Výpis bude obsahovat:

- v titulku a v h1 nadpisu jméno aktualního adresáře
- `<pre>`
- dále hlavičku seznamu zarovnanou pomocí mezer (detail viz. dále)
- `<hr />`
- Seznam souborů ve formátu
  `<Jméno> <Datum poslední změny> <Velikost v KB>k`
  Zarovnaný zleva pomocí mezer do sloupců, minimální oddělovač sloupců je **8 mezer**.
  Jméno souboru/adresáře bude zároveň HTML odkazem na tento soubor/adresář.
  Adresáře budou před jménem obsahovat předponu „`[DIR] `“.
  - Pokud je soubor menší než 1KB tak velikost udávejte v bytech.
    Velikost adresáře je vždy brána jako 0, ve výpisu se místo toho uvede znak ‘-‘ (mínus).
  - Kromě odkazů na soubory je součástí vygenerovaného seznamu i položka odkazující na nadřazený adresář (viz. obrázek). Pokud se jedná o výpis kořenového adresáře, odkazuje tento odkaz sám na sebe. Tato položka je vždy na vrchu seznamu.
- `</pre>`
- `<hr />`

Obsah testovacího adresáře bude po vypsání vypadat zhruba takto:

Index of /

Name	Last modified	Size
[..] Parent Directory	18-Jan-2011 19:02	-
entry.png	19-Feb-2010 00:32	1k
favicon.ico	18-Feb-2010 23:55	2k
homepage.html	18-Jan-2011 19:18	1k
[DIR] html/	18-Jan-2011 19:18	-

Součástí zadání je i šablona definující požadovaný vzhled. V šabloně se vyskytují různé šablonovací řetězce jako `{{PATH}}`, `{{FILE_LIST}}`, `{{NAME_SORT_TYPE}}`. Můžete počítat s tím, že se tyto řetězce nevyskytnou jako jméno žádného adresáře ani souboru.

Hlavička a třídění souborů

Hlavička indexu bude obsahovat odkazy, které budou v rámci query stringu obsahovat příslušné parametry pro třídění výpisu obsahu. Hodnota těchto parametrů bude buď znak **`A`** pro vzestupné třídění nebo **`D`** pro sestupné třídění.

- Pro třídění podle jména souboru bude použit parametr **`N`** s hodnotami `A` nebo `D`.
- Pro třídění podle data poslední modifikace souboru bude použit parametr **`M`** s hodnotami `A` nebo `D`.
- Pro třídění podle velikosti souboru bude použit parametr **`S`** s hodnotami `A` nebo `D`.

Příklad:
Pro setřízení aktuálního adresáře vzestupně podle jména bude odkaz `Name` obsahovat URL `/?N=A`.
Pro setřízení aktuálního adresáře sestupně podle velikosti bude odkaz `Size` obsahovat URL `/?S=D`.

Poznámka:

- Pokud je zadáno více třídících parametrů, bere se první z nich (zajišťuje knihovna).
- Jako výchozí hodnota pro třídění se bere honota "podle jména vzestupně" (zajišťuje knihovna).

HTTP knihovnu a šablonu indexu najdete spolu s testovacími daty zde:
http://d3s.mff.cuni.cz/~keznikl/download/zapocet-java-2011-19-01.zip

Detaily pro zájemce

HTTP request (posílá klient na server)

1. První řádek dotazu obsahuje 3 části oddělené mezerou: „GET" RequestURL HttpVersion.
2. Potom následuje libovolný počet řádků s HTTP hlavičkami.
3. Nakonec klient pošle prázdný řádek.

Příklad:

```
GET /index.html?param1=value&param2=value HTTP/1.1
Host: localhost:8000
Connection: keep-alive
Cache-Control: max-age=0
Accept-Language: en-US,en;q=0.8
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.3
(prázdný řádek)
```

HTTP response (posílá server klientovi)

1. První řádek odpovědi obsahuje text: `HTTP/1.0 200 OK`
2. Druhý řádek je prázdný.
3. Dále následuje obsah odpovědi, např. data JPEG obrázku

Příklad:

```
HTTP/1.0 200 OK
(prázdný řádek)
<html>
<body>Príklad Obsahu<body>
</html>
```