POGLAVLJE 3

Programiranje internet klijenata

Ne možete ukloniti nešto sa interneta – to je kao skupljanje urina iz bazena. Kad se jednom nađe tu, ostaje tu.

– Džo Gareli, marta 1996.
(kao „Džo Rogan“, lik iz televizijske emisije *NewsRadio*),

U ovom poglavlju...

- Šta su internet klijenti?
- Prenos datoteka
- Mrežne konferencije
- E-pošta
- Srodni moduli

U poglavlju 2, „Mrežno programiranje“, osvrnuli smo se na mrežne komunikacione protokole niskog nivoa koji koriste utičnice (engl. *sockets*). Ta vrsta mrežne komunikacije predstavlja srž većine klijent/server protokola koji danas postoje na internetu. U njih spadaju protokoli za prenos datoteka (FTP itd.), čitanje sadržaja Usenet diskusionih grupa (Network News Transfer Protocol), slanje e-pošte (SMTP) i preuzimanje e-pošte sa servera (POP3, IMAP) itd. Ovi protokoli funkcionišu slično kao primeri s klijentima i serverima iz poglavlja 2. Jedina razlika je to što sada, povrh protokola nižeg nivoa kao što je TCP/IP, pravimo nove, specifičnije protokole da bismo implementirali navedene servise višeg nivoa.

3.1 Šta su internet klijenti?

Pre nego što razmotrimo pomenute protokole, moramo se zapitati, „Šta je internet klijent?“ Da bismo odgovorili na to pitanje, usvojićemo pojednostavljenu predstavu interneta kao mesta gde se razmenjuju podaci, a u toj razmeni učestvuje neko ko nudi uslugu i klijent koji je traži. U nekim krugovima, koristi se izraz *proizvođač/potrošač* (premda je ova fraza načelno rezervisana za razgovore o operativnim sistemima). Serveri su proizvođači koji pružaju usluge, a klijenti – potrošači koji konzumiraju ponuđene usluge. Za svaku uslugu obično postoji samo jedan server (proces, domaćin itd.) i više od jednog potrošača. Prethodno smo analizirali model klijent/server, i – mada nije potrebno da pravimo internet klijente sa operacijama nad utičnicama niskog nivoa kakve smo ranije videli – taj model je tačan i u tom slučaju.

U ovom poglavlju razmotrićemo razne internet protokole i napraviti klijente za svaki od njih. Kada završimo, trebalo bi da budete u stanju da prepoznate koliko su slični API interfejsi svih tih protokola – za to je zaslužan dizajn, jer konsistentnost interfejsa je važna – i, što je još važnije, da pravite stvarne klijente tih i drugih internet protokola. A iako nam je akcenat na samo tri konkretna protokola, do kraja ovog poglavlja trebalo bi da steknete dovoljno samopouzdanja da pišete klijente za *bilo koji* internet protokol.

3.2 Prenos datoteka

3.2.1 Internet protokoli za prenos datoteka

Jedna od najpopularnijih aktivnosti na internetu jeste razmena podataka. Odigrava se *sve vreme*. Postoji mnogo protokola za prenos podataka preko interneta, a u najpopularnije spadaju protokol za prenos datoteka (engl. *File Transfer Protocol, FTP*), protokol za kopiranje s Unixa na Unix (engl. *Unix-to-Unix Copy Protocol, UUCP*) i, naravno, veb protokol za prenos hiperteksta (engl. *Hypertext Transfer Protocol, HTTP*). Valja pomenuti i Unixovu naredbu za prenos datoteka sa udaljenog sistema, `rcp` (i njene današnje bezbednije i fleksibilnije rođake, `scp` i `rsync`).

HTTP, FTP, i `scp/rsync` i danas su prilično popularni. HTTP se prevashodno koristi za preuzimanje datoteka sa veba i pristup veb servisima. Načelno, nije neophodno da klijenti imaju korisničko ime i/ili lozinku da bi dobili dokumente ili

uslugu. Većinu zahteva za HTTP prenos datoteka čine zahtevi za učitavanje veb strana (preuzimanje datoteka).

S druge strane, `scp` i `rsync` zahtevaju da se klijenti prijave na server. Da bi došlo do prenosa datoteka, tj. da bi datoteke mogle da se šalju (otpremaju) ili učitavaju (preuzimaju), neophodna je provera identiteta klijenta.

Na kraju, tu je FTP. Poput naredbi `scp/rsync`, FTP se može upotrebiti za otpremanje ili preuzimanje datoteka, i koristi Unixove višekorisničke koncepte korisničkih imena i lozinki. FTP klijenti moraju da upotrebe korisničko ime i lozinku postojećih korisnika; međutim, FTP dozvoljava i anonimno prijavljivanje. Razmotrimo pobliže FTP.

3.2.2 FTP protokol

Protokol za prenos datoteka (FTP) osmislili su pokojni Džon Postel i Džojs Rejnolds u dokumentu 959 publikacije Request for Comment (RFC) iz oktobra 1985. Prevashodno se koristi za anonimno preuzimanje javno dostupnih datoteka. Može se primenjivati i za prenos datoteka između dva računara, naročito ako koristite sistem zasnovan na Unixu za skladištenje ili arhiviranje, ili radite na stonom ili prenosivom računaru. Pre nego što je veb postao popularan, FTP je bio jedna od osnovnih metoda za prenos datoteka preko interneta, i za preuzimanje softvera i/ili izvornog koda.

Kao što je ranije pomenuto, da biste pristupili udaljenom računaru s FTP serverom, morate imati korisničko ime / lozinku. Izuzetak su anonimna prijavljivanja, osmišljena za goste. Na taj način, klijenti koji nemaju naloge mogu da preuzimaju datoteke. Administrator servera mora da podesi FTP server sa anonimnim prijavljivanjem da bi to omogućio. Neregistrovan korisnik se prijavljuje pod imenom *anonymous*, a lozinka je – u opštem slučaju – e-adresa klijenta. To je slično javnom pristupu direktorijumima namenjenim za opšte korišćenje, za razliku od prijavljivanja i prenosa datoteka od strane konkretnog korisnika. Spisak naredbi koje dozvoljava FTP protokol načelno je restriktivniji nego kad je reč o realnim korisnicima.

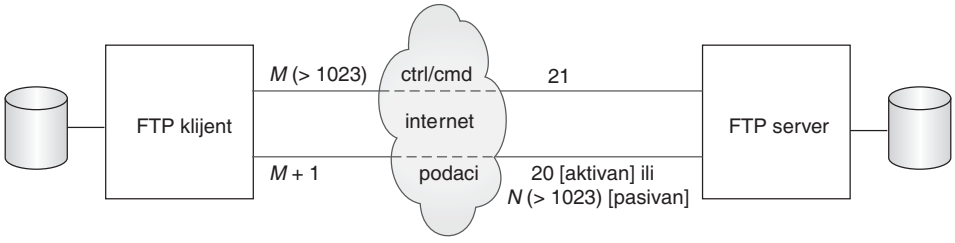
Slika 3-1 prikazuje dijagram protokola. Protokol se odvija na sledeći način:

1. Klijent kontaktira FTP server na udaljenom računaru
2. Klijent se prijavljuje s korisničkim imenom i lozinkom (ili kao „anonymous“, sa e-adresom)
3. Klijent obavlja razne poslove prenosa datoteka ili informacija
4. Klijent okončava transakciju tako što se odjavljuje sa udaljenog računara i FTP servera

Naravno, ovo je uopšten princip funkcionisanja. U izvesnim okolnostima, čitava transakcija se prekida pre nego što se dovrši. To se, na primer, dešava zbog prekida veze s mrežom ukoliko jedan od dva domaćina otkažu ili se javi neki drugi problem s mrežnom vezom. U slučaju neaktivnih klijenata, FTP konekcija će, načelno, isteći posle 15 minuta (900 sekundi) neaktivnosti.

Dobro je znati da FTP koristi isključivo TCP (videti poglavlje 2) – ni na koji način ne primenjuje UDP. Takođe, FTP se može tretirati kao pomalo neobičan primer programiranja arhitekture klijent/server, jer i klijent i server koriste par priključaka za

komunikaciju: jedan je kontrolni ili komandni ulaz (ulaz 21), a drugi ulaz za podatke (ponekad ulaz 20).



Slika 3-1 FTP klijenti i serveri na internetu. Klijent i server komuniciraju pomoću FTP protokola preko komandnog ili kontrolnog ulaza; podaci se prenose preko ulaza za podatke.

Kažemo „ponekad" jer postoje dva FTP režima: aktivni i pasivni, a ulaz 20 je ulaz servera za podatke samo u aktivnom režimu. Pošto server odredi da ulaz 20 bude njegov ulaz za podatke, „aktivno" inicira konekciju sa ulazom za podatke klijenta. U pasivnom režimu, server je jedino obavezan da obavesti klijent gde je njegov ulaz za nasumične podatke; klijent mora sam da inicira konekciju. Kao što vidite, server u ovom režimu ima pasivniju ulogu u konekciji prenosa podataka. Na kraju, pomenimo da postoji podrška za nov, proširen pasivni režim (engl. *Extended Passive Mode*) kako bi se podržala verzija 6 IP (IPv6) adresa— videti RFC 2428.

Python podržava većinu internet protokola, uključujući FTP. Ostale podržane biblioteke klijenata mogu se naći na adresi http://docs.python.org/library/internet. Razmotrimo sada koliko je lako napraviti internet klijent pomoću Pythona.

3.2.3 Python i FTP

Dakle, kako pišemo FTP klijent na Pythonu? Veliki deo odgovora je u prethodnom odeljku. Jedini dodatni posao je uvoženje odgovarajućeg Pythonovog modula i pozivanje odgovarajućih funkcija u Pythonu. Pretresimo protokol u par crta:

1. Povezujemo se sa serverom
2. Prijavljujemo se
3. Pravimo zahteve za uslugu (i, nadamo se, dobijamo odgovore)
4. Završavamo

Da bismo iskoristili Pythonovu podršku za FTP, dovoljno je da uvezemo modul `ftplib` i napravimo instancu klase `ftplib.FTP`. Sva FTP aktivnost – prijavljivanje, prenos datoteka i odjavljivanje – ostvariće se pomoću vašeg objekta.

Evo pseudokoda za Python:

```
from ftplib import FTP
f = FTP('some.ftp.server')
f.login('anonymous', 'your@email.address')
    :
f.quit()
```

Uskoro ćemo razmotriti pravi primer, ali prvo ćemo se upoznati s metodama klase `ftplib.FTP`, koje ćete verovatno koristiti u svom kodu.

3.2.4 Metode klase `ftplib.FTP`

Najpopularnije metode smo naveli u tabeli 3-1. Lista nije sveobuhvatna – pregledajte izvorni kôd klase da biste videli sve njene metode – ali ovde predstavljene metode su one koje čine API za programiranje FTP klijenata na Pythonu. Drugim rečima, nema potrebe da koristite druge metode, jer su to ili uslužne ili administrativne funkcije, ili ih kasnije koriste API metode.

Tabela 3-1 Metode za FTP objekte

Metoda	Opis
`login(`*`user`*`='anonymous', `*`passwd`*`='', acct='')`	Prijavljivanje na FTP server; svi argumenti su opcioni
`pwd()`	Tekući radni direktorijum
`cwd(`*`path`*`)`	Promena tekućeg radnog direktorijuma u *`path`*
`dir([`*`path`*`[,...[,cb]])`	Prikaz listinga direktorijuma za *`path`*; opcioni povratni poziv *`cb`* prosleđen metodi `retrlines()`
`nlst([`*`path`*`[,...])`	Poput metode `dir()`, samo što izlistava imena datoteka umesto da ih prikazuje
`retrlines(`*`cmd`*`[, `*`cb`*`])`	Preuzima tekstualnu datoteku prema FTP naredbi *`cmd`*, na primer, `RETR filename`; opcioni povratni poziv *`cb`* za obradu svakog reda datoteke
`retrbinary(`*`cmd`*`, `*`cb`*`[, `*`bs`*`=8192[, `*`ra`*`]])`	Slična metodi `retrlines()`, sem što je datoteka binarna; povratni poziv `cb` za obradu svakog preuzetog bloka (*`bs`* je podrazumevano 8K) *obavezan* je
`storlines(`*`cmd`*`, `*`f`*`)`	Otprema po FTP komandi *`cmd`*, na primer, `STOR filename`; neophodan otvoren objekat datoteke `f`
`storbinary(`*`cmd`*`, `*`f`*`[, `*`bs`*`=8192])`	Slična metodi `storlines()`, sem što je datoteka binarna; neophodan otvoren objekat datoteke `f`, podrazumevana veličina otpremljenog bloka *`bs`* iznosi 8K
`rename(`*`old`*`, `*`new`*`)`	Preimenuje udaljenu datoteku *`old`* u *`new`*
`delete(`*`path`*`)`	Briše udaljenu datoteku `file` na lokaciji *`path`*
`mkd(`*`directory`*`)`	Pravi udaljeni direktorijum *`directory`*
`rmd(`*`directory`*`)`	Uklanja udaljeni direktorijum *`directory`*
`quit()`	Zatvara konekciju i završava

U metode koje ćete najverovatnije koristiti u normalnoj FTP transakciji spadaju `login()`, `cwd()`, `dir()`, `pwd()`, `stor*()`, `retr*()` i `quit()`. Postoji još FTP metoda izostavljenih iz tabele koje bi mogle da vam budu od koristi. Detaljnije informacije o FTP objektima naći ćete u dokumentaciji Pythona, na adresi http://docs.python.org/library/ftplib#ftp-objects.

3.2.5 Interaktivni FTP primer

Primer korišćenja FTP protokla s Pythonom toliko je jednostavan da ne morate da pišete skript. Možete sve da obavite iz interaktivnog interpretera i da pratite aktivnosti i rezultat u realnom vremenu. Evo primera sesije od pre nekoliko godina kada se na lokaciji `python.org` još uvek izvršavao FTP server. Međutim, danas neće funkcionisati, te ovo služi samo da pokaže kakvo biste iskustvo mogli da imate sa izvršavanjem FTP servera:

```
>>> from ftplib import FTP
>>> f = FTP('ftp.python.org')
>>> f.login('anonymous', 'guido@python.org')
'230 Guest login ok, access restrictions apply.'
>>> f.dir()
total 38
drwxrwxr-x  10 1075     4127         512 May 17  2000 .
drwxrwxr-x  10 1075     4127         512 May 17  2000 ..
drwxr-xr-x   3 root     wheel        512 May 19  1998 bin
drwxr-sr-x   3 root     1400         512 Jun  9  1997 dev
drwxr-xr-x   3 root     wheel        512 May 19  1998 etc
lrwxrwxrwx   1 root     bin            7 Jun 29  1999 lib -> usr/lib
-r--r--r--   1 guido    4127          52 Mar 24  2000 motd
drwxrwsr-x   8 1122     4127         512 May 17  2000 pub
drwxr-xr-x   5 root     wheel        512 May 19  1998 usr
>>> f.retrlines('RETR motd')
Sun Microsystems Inc.   SunOS 5.6       Generic August 1997
'226 Transfer complete.
>>> f.quit()
'221 Goodbye.'
```

3.2.6 Primer programa s FTP klijentom

Već smo pomenuli da uopšte nije potrebno da se gubimo u pisanju koda, jer skript možete izvršiti interaktivno. Ipak, okušaćemo se u kodiranju. Pretpostavimo da treba da napišete kôd za preuzimanje najnovije verzije Bugzille s veb lokacije Mozille. Primer 3-1 pokazuje šta smo uradili. Ovde pokušavamo da napišemo aplikaciju, ali bez obzira na to, mogli biste i ovo da izvršite interaktivno. Naša aplikacija koristi FTP biblioteku da bi preuzela datoteku, a obuhvata i neke mehanizme otkrivanja grešaka.

Primer 3-1 Primer za FTP preuzimanje datoteke (`getLatestFTP.py`)

Ovaj program se koristi za preuzimanje najnovije verzije datoteke s veb lokacije. Možete ga podesiti tako da preuzima vašu omiljenu aplikaciju.

```
#!/usr/bin/env python

import ftplib
import os
import socket

HOST = 'ftp.mozilla.org'
DIRN = 'pub/mozilla.org/webtools'
FILE = 'bugzilla-LATEST.tar.gz'

def main():
    try:
        f = ftplib.FTP(HOST)
    except (socket.error, socket.gaierror) as e:
        print 'ERROR: cannot reach "%s"' % HOST
        return
    print '*** Connected to host "%s"' % HOST

    try:
        f.login()
    except ftplib.error_perm:
        print 'ERROR: cannot login anonymously'
        f.quit()
        return
    print '*** Logged in as "anonymous"'

    try:
        f.cwd(DIRN)
    except ftplib.error_perm:
        print 'ERROR: cannot CD to "%s"' % DIRN
        f.quit()
        return
    print '*** Changed to "%s" folder' % DIRN

    try:
        f.retrbinary('RETR %s' % FILE,
              open(FILE, 'wb').write)
    except ftplib.error_perm:
        print 'ERROR: cannot read file "%s"' % FILE
        os.unlink(FILE)
  else:
       print '*** Downloaded "%s" to CWD' % FILE
  f.quit()
  return

if __name__ == '__main__':
    main()
```

Imajte na umu da ovaj skript nije automatizovan, pa je na vama da ga izvršite kada poželite da preuzmete aplikaciju, ili, ako koristite sistem zasnovan na Unixu, možete podesiti da servis `cron` to radi automatski za vas. Druga stvar je to što će se skript prekinuti ukoliko se promene ime datoteke ili direktorijuma.

Ako se ne javi nijedna greška dok izvršavamo skript, rezultat je ovakav:

```
$ getLatestFTP.py
*** Connected to host "ftp.mozilla.org"
*** Logged in as "anonymous"
*** Changed to "pub/mozilla.org/webtools" folder
*** Downloaded "bugzilla-LATEST.tar.gz" to CWD
$
```

Objašnjenje po redovima

Redovi 1–9

U početnim redovima uvoze se neophodni moduli (pretežno da bi se hvatali objekti izuzetaka) i dodeljuju vrednosti određenim konstantama.

Redovi 11–44

Funkcija `main()` sastoji se od nekoliko koraka operacije: pravimo FTP objekat i pokušavamo da se povežemo na FTP server (redovi 12–17), a u slučaju neuspeha, funkcija okončava izvršavanje. Pokušavamo da se prijavimo kao korisnik *anonymous*, a ako je prijavljivanje neuspešno, odustajemo (redovi 19–25). Sledeći korak je prelaz na direktorijom distribucije (redovi 27–33), i, na kraju, preuzimanje datoteke (redovi 35–44).

Obratite pažnju na red 14 i sve ostale procedure za obradu izuzetaka u ovoj knjizi u kojima se čuva instanca izuzetka (u ovom slučaju `e`) – ukoliko koristite Python 2.5 ili stariju verziju, izmenite **`as`** u zarez, jer je ova nova sintaksa uvedena (ali nije obavezna) u verziji 2.6, da bi se olakšao prelazak na verziju 3.x. Python 3 razume samo novu sintaksu iz reda 14.

U redovima 35–36, prosleđujemo povratni poziv metodi `retrbinary()` koji bi trebalo da se izvrši za svaki preuzeti blok binarnih podataka. Ovo je metoda `write()` objekta datoteke koji pravimo radi pisanja lokalne verzije datoteke. Oslanjamo se na to da će Pythonov interpreter zatvoriti našu datoteku na odgovarajući način posle završenog transfera, tako da se pritom ne izgubi ništa od naših podataka. Iako je tako lakše, pokušavamo da izbegnemo ovakvo rešenje, jer bi programer morao da bude odgovoran za oslobađanje direktno rezervisanih resursa umesto da se oslanja na spoljni kôd. U ovom slučaju, trebalo bi da sačuvamo otvoreni objekat datoteke u promenljivu, na primer `loc`, a potom da prosledimo `loc.write` metodi `ftp.retrbinary()`.

Nakon što se transfer završi, pozvali bismo metodu `loc.close()`. Ako iz nekog razloga nismo u stanju da sačuvamo datoteku, uklanjamo praznu datoteku kako bismo izbegli zakrčavanje sistema datoteka (red 40). Valjalo bi da uvedemo i korak otkrivanja grešaka uz pozivanje funkcije `os.unlink(FILE)` za slučaj da datoteka ne postoji. Na kraju, da bismo izbegli još jedan par redova (redovi 43–44) koji zatvara FTP konekciju i okončava izvršavanje, koristimo odredbu **`else`** (redovi 35–42).