Saját IRCBot Perl nyelven

Napi több óra csevegés közben jól jöhet egy társ, aki a rendszeres, könnyen önműködővé tehető feladatokat ellátja helyettünk.

A botokról röptében annyit, hogy a csevegőként futó programok több egyszerű feladatot láthatnak el: a csevegőszoba foglalását, a elárasztás (flood) elleni védelmet, önműködő kirúgást (kick), illetve opot (az op biztonságos jogokkal rendelkező IRC felhasználó); hivatkozásokat és egyéb adatok megjegyzését stb. Mindehhez emberi beavatkozás nem szükséges, esetleg futás közben befolyásolható a működése. A cikkben egy IRC-re (Internet Relay Chat) írt botot mutatok be. A magyar IRCNeten csak a ***sote.irc.hu*** és a ***hub.irc.hu*** kiszolgálókon szabad botokat futtatni. Arról, hogy az adott IRC-kiszolgálón futtathatunk-e botot, általában a MOTD (Message of the day) tájékoztat. Percdíjas internet esetén érdemes saját IRC-kiszolgálót telepíteni, és a fejlesztés ideje alatt azon próbálgatni a botot. Az IRC-ről további adatokat, az alapfogalmakról leírásokat, illetve az illemtanról a ➲ http://www.irc.hu címen olvashatunk.
Perl nyelven írt botom felépítéséhez a `Net::IRC` (v0.73-2) modult használtam. E csomag fejlesztői felületét fogom ismertetni, természetesen nem kimerítően, csupán amennyire e cikk keretei engedik és amennyi ismeret az induláshoz elengedhetetlen. A csomag négy fő összetevőből áll: `Net::IRC`, `Net::IRC::Connection`, `Net::IRC::Event`, `Net::IRC::DCC`. Az utóbbi bemutatására nem térek ki részletesebben, mert nem feltétlenül szükséges a bot működéséhez, és az előzőek ismeretében könnyen megtanulható. A `Net::IRC` modul adja a keretet a bot működéséhez. Eseményvezérelt felépítésének köszönhetően nem szükséges az IRC-protokollt mélységeiben ismerni, és a kiszolgáló által küldött üzeneteket ellenőrizni, hanem az előre meghatározott eseményekre (connect, join, msg stb.) mindössze egy-egy eljárást kell írni, amelyek önműködően meghívódnak. Ezeket a függvényeket önkényesen hozzárendelhetjük az egyes eseményekhez, de nem is mindegyikhez kötelező. A későbbiek folyamán részletezem, hogy ezt milyen utasításokkal lehet megtenni.

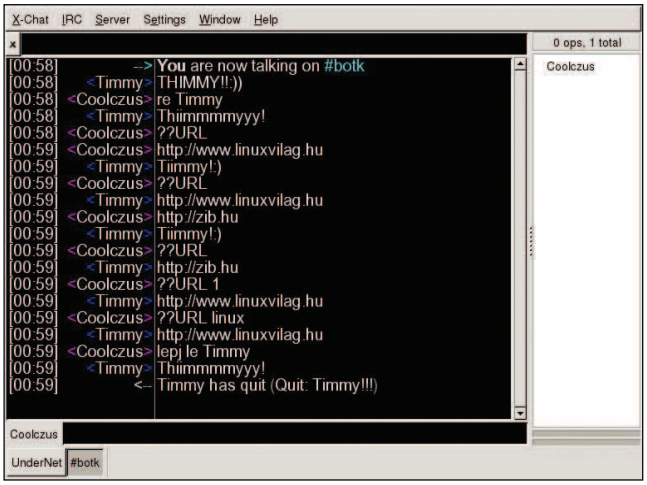
Telepítés

A csomag telepítéséhez Debian GNU/Linux alatt egyszerűen adjuk ki a következő utasítást:

```
apt-get install libnet-irc-perl
```

Ezenkívül szükségünk lesz még a Perl 5.6.0-16-os vagy ennél frissebb változatára is. Más terjesztés esetén is nagy valószínűséggel megtalálható a csomagban, vagy a ➲ http://www.cpan.org weblapról, a ➲ http://search.cpan.org/author/JMUHLICH/Net-IRC-0.74/IRC.pm címről tölthető le. További útmutatást a súgóoldalain, illetve a leírásában (Debian: ***/usr/share/doc/libnet-irc-perl***) lehet találni, sőt a könnyebb megértés kedvéért még egy egyszerű példaprogramot is tanulmányozhatunk.
Az IRC-kiszolgáló csomagból történő telepítése nagyon egyszerű feladat:

```
apt-get install ircd
```



Ezeken kívül választhatjuk még a `dancer-ircd` és a `dancer-services` csomagokat is. Annyival tudnak többet az egyszerűbb IRC-kiszolgálónál, hogy különböző szolgáltatásokat nyújtanak a csevegők számára: csevegőszoba, név (nick) bejegyzése stb. Forrásból való telepítése sokkal bonyolultabb feladat – ennek ismertetése túllép e cikk keretein, de a bátrabbak megpróbálkozhatnak vele.

Első lépések

Az *1. listán* egy nagyon egyszerű IRC bot (Timmy) forrása látható (ez egyébként a CD-mellékleten is megtalálható, az 53. CD Magazin/IRCbot könyvtárában). Alapvetően nem csinál mást, mint kapcsolódik egy IRC-kiszolgálóhoz, belép egy csatornára, és időnként azt mondja, hogy Timmy (a hasonlóság a South Park egyik szereplőjével nem a véletlen műve.) Most tekintsük át lépésről lépésre, hogy melyik utasítás mire való.
A `$DEBUG` változót a hibakeresés megkönnyítésére használom. Futás közben különböző üzeneteket ír ki. Ha a hibaelhárítás be van kapcsolva, a `debug()` függvény az első értékként megadott üzenetet írja ki.
Ezek után létrehozok egy IRC-objektumot, ami ekkor még nem működik, azaz nem kapcsolódik sehova. Ahhoz, hogy a bot elinduljon, a `start()` függvényt kell meghívni, ahogy a forrás legvégén is látni lehet. Az IRC-objektum számára egy kapcsolatot is létrehozok. Megadott értékei, gondolom, nem szorulnak bővebb magyarázatra: `nick`, `server` stb. A `Port` mezőt nem kötelező megadni, alapbeállításként a 6667-et használja. Ha a bot jelszóval védett IRC-kiszolgálóra vagy BNC-re fog csatlakozni, még a `Password` mező is megadható. Természetesen egy IRC-objektumhoz több kapcsolat is létrehozható, de a leírás szerint egyetlen kapcsolattal működik teljesen megbízhatóan. Mire a program eléri az 1.0-s változatot, valószínűleg ezt a gyermekbetegségét is kinövi.
Nem fontos minden kapcsolatot a bot indítása előtt megadni, futás közben (a `start()` függvény meghívása után) is dinami-

1. lista

```
#!/usr/bin/perl -w
use Net::IRC;

# Hibakereses ki/bekapcsolasahoz egy valtozo
my $DEBUG = 1;
sub debug{
      my $msg = shift;
      if ( $DEBUG ){
            print "DEBUG: " . $msg . "\n";
      }
}

# IRC objektum letrehozasa
my $irc = new Net::IRC;

# Kapcsolat objektum letrehozasa.
my $conn = $irc->newconn(
      Server       => '10.0.1.254',
      Port         => '6667',
      Nick         => 'Timmy',
      Ircname      => 'Timmy',
      Username     => 'Timmy'
) or die "Nem lehet kapcsolodni az IRC
      ➥szerverhez.\n";
$conn->{channel} = '#botk';
sub on_connect {
      # Elso argumentumot kiszedjuk.
      my $conn = shift;
      &debug( "Connected..." );
      # Belep a csatszobaba.
      $conn->join($conn->{channel});
      $conn->{connected} = 1;
}
sub on_disconnect {
      # Mindket argumentum kiszedese.
      my ($conn, $event) = @_;
      # Ha megszakadt a kapcsolat ujra
      # csatlakozik.
      &debug( "Disconnected from " .
      ➥$event->from() . " (" .
      (($event->args())[0]) . ").
      ➥Attempting to reconnect...\n");
            $conn->connect();
      }
sub on_join {
      # Mindket argumentum kiszedese.
      my ($conn, $event) = @_;
      # Annak a neve, aki belepett.
      my $nick = $event->{nick};
      &debug( "$nick joined "
      ➥.$conn->{channel}. "." );
      # Ha belepett valaki, akkor Timmy
      # elkialtja magat.
      $conn->privmsg($conn->{channel},
      ➥"THIMMY!!:))");
}
sub on_part {
      # Mindket argumentum kiszedese.
      my ($conn, $event) = @_;
      # Annak a neve, aki kilepett.
      my $nick = $event->{nick};
      &debug( "$nick leaved "
      ➥.$conn->{channel}. "." );
      $conn->privmsg($conn->{channel},
      ➥"Timmmy!!:(");
}
sub on_msg {
      # Mindket argumentum kiszedese.
      my ($conn, $event) = @_;
      # Annak a neve, aki kuldte
      # az uzenetet.
      my $nick = $event->{nick};
      # Az uzenet szovege.
      my $text = $event->{args}[0];
      &debug( "<$nick> $text" );
      # Barmit irnak Timmy mindig ugyanazt
      # valaszolja vissza.
      $conn->privmsg( $nick,
      ➥"Thiimmyyyy!" );
} # on_msg vege
sub on_public{
      # Mindket argumentum kiszedese.
      my ($conn, $event) = @_;
      # Az uzenet kuldoje.
      my $nick = $event->{nick};
      # Maga az uzenet.
      my $text = $event->{args}[0];
      &debug( "<$nick> $text" );
      # most megnezzuk, hogy neki lett
      # e kuldve az uzenet
      if( $text =~ /Timmy/i ){
            # Ha neki jott az uzenet
            # akkor beszol.
            $conn->privmsg
            ➥($conn->{channel},
            ➥"Thiimmmmyyy!");
            # Ha azt irjak neki lepjel le,
            # akkor lelep.
            if ($text =~ /lepj le/){
                  $conn->quit("Timmy!!!");
                  # ezzel lep ki
                  exit 0;
            }
      } # neki szolas vege
} # sub on_public vege

# Esmenykezelok belalitasa.
$conn->add_handler('msg', \&on_msg);
$conn->add_handler('public', \&on_public);
$conn->add_handler('join', \&on_join);
$conn->add_handler('part', \&on_part);
$conn->add_handler('376', \&on_connect);
$conn->add_global_handler('disconnect',
      ➥↓\&on_disconnect);

# IRCBot inditasa.
&debug( "Bot started..." );
$irc->start();
```

IRC-parancsok

- Kapcsolódás a kiszolgálóhoz:
  `/server <szervernev> [port]`
  A kapu értéke elhagyható. Alapértelmezés szerint 6667.
  Például: */server extra.irc.hu*
- Belépés egy szobába: `/join #szobanev`
  Ha az adott szoba nem létezik, akkor létrejön, és mi leszünk az opok.
- Kilépés a szobából: `/part`
- Kilépés a kiszolgálóról: `/quit [uzenet]`
  Ha nem hagyjuk el az üzenetet, kilépéskor kiírja. Ha elhagyjuk, akkor nem ír semmit, illetve néhány IRC-ügyfél (bitchx) véletlen üzeneteket ír be helyettünk.
- Üzenet küldése egy csevegőnek: `/msg <csevego neve>`
- Téma beállítása: `/topic <szoveg>`
  A beállításhoz opra van szükség.
- Op adása: `/op <nick>`
- Kirúgás egy szobából: `/kick <szoba> <nick>`
- Kitiltás egy szobából: `/ban <nev>`
  Itt a név egészen összetett lehet: *nick!ident@hostname*
  Nem kötelező minden részét megadni. Ha például a „valaki” nevű egyént akarjuk kitiltani: `valaki!*@*`
  Ezeket a parancsokat természetesen nem kell minden esetben ismernünk. A legtöbb ügyfél (mIRC, XChat) – elsősorban a grafikusak – esetén gombokkal is elérhetőek.

kusan bővíthető. Erről bővebb útmutatóra a súgóoldalakon lelhetünk. A `$conn channel` mezőben lehet megadni azt a csatornát, amire a bot be fog lépni.
Most az eseménykezelő függvények következnének, de hogy érthetőbb legyen, mit miért tettünk, ugorjunk a forrás végére. Az eseményeket háromféleképpen kezelhetjük: az első esetben, ha nem határoztunk meg saját eseménykezelőt, az alapértelmezett hívódik meg, ami pusztán a legminimálisabb feladatokat látja el. A második esetben meghatározzuk, hogy egy esemény bekövetkeztekor melyik függvény hajtódjon végre (`add_global_handler()`). A harmadik esetben kapcsolatonként határozzuk meg, hogy az adott eseményre melyik függvény fusson le (`add_handler()`). Mivel csak egyetlen kapcsolatom van, bármelyik függvényt felhasználhattam volna a példában, de én mindkettőt alkalmaztam. Az `add_handler()` függvénynek meg kell adni az esemény azonosítóját és az eseménykezelő függvényt. Itt még nincs vége a variálhatóságnak, mivel utolsó értékeként megadható, hogy az alapértelmezett eseménykezelőt felülírja (0), illetve a sajátunk előtt (1) vagy után (2) futtassa le. Ha elhagyjuk ezt az értéket, akkor csak a saját eseménykezelő fut majd le. Hasonlóan kell meghívni az `add_global_handler()` függvényt is. Az események nevei a `Net::IRC::Event` modul forrásában találhatók meg. Ez több mint 160-féle eseményt különböztet meg. Én az 1. listán a következőket használtam:

- `msg`: privát üzenetet kap a bot.
- `public`: a csevegőszobába érkezik egy üzenet.
- `join`: valaki belép a csevegőszobába, még akkor is, ha maga a bot lép be.
- `part`: a csevegőszoba elhagyása.
- `376`: a MOTD végét jelző üzenet – ez egyértelműen azt mutatja, hogy a bot sikeresen csatlakozott.

Minden egyes eseménykezelő két értékkel hívódik meg: az egyik a kapcsolatot (`$conn`) írja le, a másik pedig az eseményt (`$event`). A `$conn->privmsg()` függvénnyel lehet üzenetet küldeni a csatornára, illetve egy adott személynek. Első értékében meg kell adni, hogy kinek (vagy melyik szobának) szól az üzenet, a második érték pedig az üzenet szövege maga. Az `$event` adatokat tartalmaz az eseményről: a csevegő nevét (`nick`), a gépének a nevét stb. A következőképpen lehet például egy csevegőnek opot adni:

```
$conn->mode( $conn->channel, "+o",
➥$event->nick );
```

Az 1. listában látható függvényeket nem részletezem tovább, ugyanis a megjegyzések alapján könnyen el lehet igazodni bennük.

További lehetőségek

Innentől kezdve a határ a csillagos ég: botunk már mindenre képes lehet, amit Perl nyelven meg lehet írni. Megkérdezhetjük az időt, a tévéműsort (➲ http://www.port.hu szűrése) és a napi horoszkópot, sőt botunk akár az összes csatornán elhangzott hivatkozást is megjegyezhetné. Ennek az utóbbinak a megvalósítását mutatja be a 2. lista, amelynek teljes forrása a CD-mellékleten található meg. A hivatkozásokat MySQL-adatbázisban tárolom, DBI segítségével (Debian: `apt-get install libdbi-perl`). Az egyszerűség kedvéért az adatbázis, a felhasználó neve és a jelszó is `timmy` lett. A hivatkozások táblát a következőképpen lehet létrehozni:

```
create table linkek( URL test, datum integer);
```

A MySQL használatának részletes bemutatása meghaladná e cikk kereteit. Aki még nem foglalkozott vele, az egyszerűen egy fájlban is tárolhatja a hivatkozásokat. Ha sikerült beállítani az adatbázist, akkor a botnak be lehet írni a csevegőszobába: ***http://www.linuxvilag.hu***, amit szépen tárol is. A lekérdezés a következőképpen néz ki: ***??URL [minta] [sorszám]***, itt mindkét érték elmaradhat, és mindig csak egy hivatkozást listáz ki. A sorszámozás pedig nullától indul és dátum szerint csökkenő sorrendben halad.
Ha eddig eljutottunk, és a hivatkozásokat már ragyogóan megjegyzi a bot, akkor egy kis átalakítással az is elérhető, hogy a csevegők aranyköpéseit is tároljuk és megőrizzük az utókornak. Aki nem szereti vagy nem ismeri a Perlt, annak a következő írásomat ajánlom, ami a mostanihoz hasonlóan saját IRC bot megírásáról fog szólni, de a Java nyelv használatával.

Kolcza Péter (kpeter@sysconfig.hu)
Imádja a South Parkot. A Miskolci Egyetem informatika szakos hallgatója. Elvakult GNU/Linux-rajongó. Ha egyetemi elfoglaltságai engedik, rendszerépítéssel foglalkozik. A cikkel kapcsolatban minden észrevételt szívesen fogad.

KAPCSOLÓDÓ CÍMEK

- ➲ http://www.irc.hu
- ➲ http://www.cpan.org
- ➲ http://www.mysql.com