Bevezetés a vi használatába

Még akkor is érdemes megismerkedni a mindenütt jelenlévő vi használatának alapjaival, ha munkád zömében másfajta szövegszerkesztőt használsz.

A Linuxszal vagy Unixszal ma ismerkedők többsége már tudja, hogyan kell használni egy grafikus WYSIWYG (amit látsz, azt kapod) szövegszerkesztőt, amely nagymértékben támaszkodik az egér, az ikonok és a legördülő menük használatára. A vi-jal, a Unix/Linux szerkesztőjével való első találkozás alkalmával az nehézkesnek tűnhet, és nem látszik valami hatékonynak sem, de ennek éppen az ellenkezője az igaz.
Ahhoz, hogy a parancsok beviteléhez a teljes billentyűzet rendelkezésre álljon, a vi-nak egy parancs- és egy beszúró üzemmódja van.
Az üzemmódok először zavaróak lehetnek. A legtöbb szerkesztőnél bármi, amit begépelsz, bekerül a dokumentumba, a szerkesztés az egér, esetleg gyorsbillentyűk segítségével történik. A vi-nál parancsmódban a teljes billentyűzet a szerkesztőparancsok bevitelére szolgál. Csak akkor kerül bele szöveg a dokumentumba, ha kiadod a beszúrás parancsot.
A legtöbb új vi-felhasználó csak néhány alapvető parancsot tanul meg: kurzorbillentyűk, `i` a beszúrás, `x` egy karakter törlése, `dd` egy sor törlése, `:wq` mentés és bezárás, esetleg még egy-két más parancsot is megismernek. Hamar megunják, hogy minduntalan ugyanazokra a billentyűkre kell támaszkodni, ugyanakkor félnek attól, hogy hatalmas parancslistákat tanuljanak meg. Ahelyett, hogy egyszerűen egy bemagolandó felsorolást készítenénk, most megpróbáljuk felfedezni a mögöttes szerkezetet, könnyen megjegyezhetővé téve ezáltal a parancsokat, és megpróbáljuk elérni, hogy gyorsan kiemelt vi-felhasználóvá lépjünk elő. A parancsok néhány alapvető kategóriája: szövegbeszúró, kurzormozgató, blokkszerkesztő, kettőspont (ex) parancsok, beállítások és egyéb parancsok.

Szöveg beszúrása

Az első parancs, amit az ember megtanul a vi-ban, az az `i`. Az `i` a szövegszerkesztőt beszúró módba teszi, ahol is minden, amit begépelsz, bekerül a dokumentumba, és egészen addig ott marad, amíg meg nem nyomod az Esc billentyűt. Az `i` azonban nem az egyetlen módja szöveg beszúrásának.
A kurzor mögé vagy a sor végére úgy adhatunk szöveget, hogy az `a` parancsot használjuk, vagy a sor végére ugrunk, és az `A` paranccsal fűzzük hozzá a szöveget. Hasonlóképpen az `I` a sor elejére szúr be szöveget. Ezeknél, valamint a többi beszúróparancsnál a parancs üzemmódba való visszatéréshez mindig nyomd le az Esc-t.
Ha a pillanatnyi sor mögé újabb sort szeretnél beszúrni, és oda szeretnéd tenni a szöveget, használd az `o`-t, vagy a pillanatnyi sor elé való beszúráshoz az `O`-t.
A pillanatnyi karakter cseréjéhez nyomd meg az `s`-t, vagy ha sok karaktert szeretnél cserélni, az `R`-t, a grafikus szövegszerkesztők felülíró üzemmódjához hasonlóan.
Csaknem az összes parancsnál használhatunk számelőtagot, amely általában az adott mennyiségben ismétli meg a parancsot. Például írd be, hogy ***3iHurrá***, majd nyomd le az Enter és Esc billentyűket. Ez 3 sor „Hurrá”-t tesz be az átmeneti tárba. Az `s` parancs különbözik ettől; itt a szám azt mutatja meg, hogy hány karakter kerül lecserélésre: az `5s` öt karaktert cserél le azzal, amit beírsz.
Beszúró üzemmódban néhány vezérlőbillentyűnek különleges jelentése van. A kurzorbillentyűket a mozgáshoz lehet használni, de ne feledjük, hogy még mindig beszúrómódban vagyunk. Ugyanígy a Ctrl-T-t és a Ctrl-D-t a sor elején a szöveg behúzására lehet használni. A Ctrl-T a behúzás következő szintjére mozgatja a szöveget, a Ctrl-D pedig vissza. A legjobb eredmény eléréséhez ezeket az ***autoindent*** és a ***shiftwidth*** beállításokkal együtt kell alkalmazni (lásd lejjebb, a beállítások résznél).

A kurzor mozgatása

A grafikus szövegszerkesztőknél elsősorban egérkattintással mozgunk. Néhány vi-klón támogatja az egér használatát, de a vi és klónjai mind olyan kurzormozgató parancsokon osztoznak egymással, amelyekkel nagyon gyorsan lehet mozogni. A legegyszerűbbek a kurzorbillentyűk vagy az azoknak megfelelő betűk: `h` (balra), j (le), `k` (fel) és `l` (jobbra). Ha azonban csak ezeket alkalmazod, elszalasztod a vi-ban rejlő erő teljes kiaknázását.
A sorban úgy is mozoghatsz, hogy a jobb vagy bal nyilakat (vagy az l-t és h-t) folyamatosan nyomva tartod, de létezik ennél egyszerűbb mód is. A sor elejére való ugráshoz a `^`, a sor végére való ugráshoz pedig a `$` parancs használatos. Ha egyszerre egy szót szeretnél ugrani, használd `w`-t, a `W`-t pedig, ha még tovább akarsz menni. Ha vissza szeretnél ugrani, akkor a `b` vagy a `B` használatos. Az `e`, vagy `E` betűket is lehet a pillanatnyi szó végére való ugrásnál alkalmazni.
A kisbetűs `w`, `b` és `e` a szó alfanumerikus karaktereinek sorozatát jelenti; a nagybetűsek szóközt használnak a szavak elválasztásához.
A pillanatnyi sorban való mozgásnak egy másik módja az, ha egy adott karakterre ugrunk. Például, ha a kurzort a következő `s`-hez szeretnénk mozgatni a sorban, gépeljük be az `fs`-t. Használjuk az `F`-et az előző betű megtalálásához: `Fs`. Vagy használjuk a `ts`-t, hogy pontosan az `s` elé kerüljünk vagy a `Ts`-t, ha a másik irányba szeretnénk menni. Az `s` helyére bármilyen más karaktert is beírhatunk, amit meg szeretnénk keresni. Ha meg szeretnénk ismételni a keresést, a , (vessző) karaktert írjuk be.
Ha a kurzort a képernyő tetejére, közepére, vagy aljára szeretnénk mozgatni, használjuk a `H`, `M` vagy `L` betűket (nagybetű). A `H`-nál vagy `L`-nél a számelőtag azt mondja meg, hogy hány sort szeretnénk lépni az oldal aljától vagy tetejétől nézve; az `M`-nél a számnak nincs hatása.
Mondatonként való előre- és hátraugráshoz használjuk a `(` vagy `)` zárójeleket; a bekezdésenkéntihez pedig a `{` és `}` kapcsos zárójeleket. Ha sok zárójellel rendelkező forráskódot szerkesztesz, a `%` paranccsal próbálkozz: helyezd a kurzort a következő karakterek valamelyikére: `(`, `)`, `[`, `]`, `{` vagy `}`, majd üsd le a `%` jelet; ekkor

a megfelelő zárójelhez fogsz ugrani.
A mozgáshoz gyakran a / paranccsal való keresés a legjobb módszer. Például gépeljük be a /hello parancsot, majd nyomjuk le az ENTER-t, ha a következő „hello”-ra szeretnél ugrani a dokumentumban. A „hello” helyett bármilyen szabályos kifejezést használhatsz. Az n parancs megismétli a keresést; az N az ellentétes irányba teszi ugyanezt.
A ? parancsot a fájlban való visszafelé kereséshez használhatod. A keresés kis- és nagybetűérzékeny; az ignorecase beállítást a cikk egy későbbi részében fogom elmagyarázni.
A legtöbb kurzormozgató parancsnál számelőtagot lehet használni a parancsok ismétléséhez. Például az 5w öt szóval fog előre mozgatni, a 2n nem a következő találathoz visz, hanem az azt követőre, az 5fs az ötödik s-hez fog mozgatni a kurzor jobb oldalán.

Blokkszerkesztő parancsok

A grafikus szerkesztőkben egy szövegblokk módosításához először jelöld ki a szöveget, kattints rá és húzd át az egeret a szöveg felett, majd kattints egy ikonra vagy menüelemre, vagy üss le egy gyorsbillentyűt. Bár a vi nem alkalmazza az egeret, az eljárás meglepően hasonló. Mint az előbb láttuk, a vi-nak széles kurzormozgató parancskészlete van. Ezek közül bármelyik tulajdonságként használható a blokkszerkesztő parancsokhoz (például a dw töröl egy szót). Ahelyett, hogy a vi-ban a szövegblokk meghatározásához először egérrel kijelölnénk a szöveget, majd egy műveletet választanánk, először írjuk be a blokkszerkesztő parancsot (d), majd a kurzormozgató parancsot (w). Gyorsbillentyűként a blokkszerkesztő parancsot kétszer is beírhatjuk, hogy az egész jelenlegi sorra hatással legyen, például a dd kitörli a pillanatnyi sort. Mint láthattuk, a d parancs a törlés. Amikor szöveget törlünk, a szöveg megjegyzésre kerül, a grafikus szerkesztők kivág parancsához hasonlóan. A szöveg beillesztéséhez (a vi szóhasználatában: tenni, „put”) a p-t írjuk be a kurzor mögé való beszúráshoz (vagy nagy P-t, ha a kurzor elé szeretnénk beszúrni). Ez megoldja a kivág–beilleszt parancsokat, de mi a helyzet a másolással? A vi-ban ezt kirántásnak („yank”) hívják, és a y parancsot használják. Ez ugyanolyan, mint a d parancs, azzal a különbséggel, hogy nem töröl semmit: az yy megismétli a pillanatnyi sort, a yw a pillanatnyi szót és így tovább.
Nagy mennyiségű szöveg cseréjénél először kitörölhetjük a szöveget, majd az i-vel beszúrhatjuk az újat, de helyette használhatjuk a c (módosít, change) parancsot, és máris beszúró módban vagyunk és beírhatjuk a módosító szöveget; ha befejezted, üsd le az ESC billentyűt. Használhatod kurzormozgató paranccsal együtt (mint például: cw), vagy az egész sort módosíthatod a cc paranccsal.
A programozók imádják a > és < parancsokat, amelyek egy szöveget tolnak el jobbra vagy balra. Egy blokknyi kód eltolásához helyezd a kurzort a { , vagy } karakterekre a blokk elején vagy a végén, és ha jobbra szeretnéd eltolni a szöveget, írd be a >% parancsot. Bármilyen mozgatóparancsot használhatsz a % helyett, az egész sor eltolásához pedig használd a >> parancsot. A < jel segít, ha balra szeretnéd tolni a szöveget.
Lásd a CTRL-T és a CTRL-D billentyűket (miközben beszúró üzemmódban vagy) és a shiftwidth beállítást lejjebb.
Végül a ! parancsot kell alkalmazni a szöveg külső program általi szűréséhez. Írd be a futtatandó program nevét, és a szerkesztő az állomány kijelölt részét szabványos bemenetként adja át; a program kimenetét pedig beilleszti a dokumentum megfelelő helyére. Például a !! a pillanatnyi sort szűri; a !} egy bekezdést szűr. Különösen hasznos parancs a Unix/Linux-szűrő fmt, amely újraformálja a bekezdéseket. Az állomány módosítása nélküli külső program futtatásához nézd meg a :! parancsot a „kettőspont (ex) parancsok” részben. Bármely számelőtag a parancs kurzormozgató részének kerül átadásra, akár elsőnek adjuk meg, akár a szerkesztő- és kurzormozgató parancs belsejében. Például mind a 3dw, mind pedig a d3w három szót töröl; a 3yy és a y3y parancs három sort másol.

A kettőspont (ex) parancsok

A vi egy parancscsoportot örökölt elődjétől, az ex-től, amit az úgynevezett kettőspont üzemmódon keresztül lehet elérni. Gyakori példa a :wq (mentés és kilépés) és a :s/X/Y/ (helyettesítsd X-et Y-nal). Írd be a : (kettőspontot), amelyet egy ex parancs követ, majd nyomd le az ENTER-t. Miután a parancs befejezte működését, a vi parancsmódba tér vissza.
Számos változata létezik a :w és :q parancsoknak. A fájl mentéséhez, majd a munka folytatásához használjuk a :w parancsot. A vi-ből a :q segítségével lehet kilépni, de csak akkor, ha a munka mentve lett; a minden feltétel nélküli kilépéshez használjuk a :q! parancsot (ez azonban veszélyes!). Próbáljuk meg a :w! parancsot a csak olvasható mód felülbírálásához, ha a :q hibát jelez. Sokan használják a :wq! mentés és kilépés parancsot. A vim-ben vagy az nvi-ben ez ugyanaz, mintha azt írnánk be, hogy :w! majd azt, hogy :q, ha nem volt hiba mentés közben. De a vi néhány más változata ezt úgy kezeli, hogy :w és :q!, ami azt jelenti, hogyha bármilyen hiba történt mentés közben, a munka elveszik. Emiatt jobb szokás a :wq-t használni.
A csere parancs rendkívül hatékony. Legegyszerűbb formájában a :s/X/Y/ az X előfordulását Y-ra cseréli le a pillanatnyi sorban. A pillanatnyi sorban az összes előfordulás cseréjéhez adjuk hozzá a g (global) betűt a parancs végéhez, például :s/foo/bar/g. Rakjuk hozzá a c-t a parancs végéhez, ha meg kívánjuk erősíteni a változtatásokat, például :s/foo/bar/gc. Több sorban való cseréhez két sorszámot írjunk előre, vesszővel elválasztva őket egymástól. Így például a :10,20s/foo/bar/g a „foo” összes előfordulását „bar”-ra cseréli le a 10. és 20. sor között. Az első sorhoz az 1-et, az utolsóhoz pedig a $ jelet kell alkalmazni; az összes sor cseréjéhez viszont a következőt: :1,$s/foo/bar/g (az 1,$, helyett használhatjuk a % jelet, például :%s/foo/bar/g).
Egy másik hasznos parancs a :g/X/. Ez azt teszi lehetővé, hogy bármilyen ex parancsot futtathass az összes X-et tartalmazó sorban. Például az összes olyan sor megjelenítéséhez a képernyőn, amely tartalmazza a „hello” szót, használd a :g/hello/p parancsot (a :p egy ex parancs egy sor képernyőn való megjelenítéséhez). Ezt össze lehet még kapcsolni a :s/// paranccsal. Cseréljük le „foo”-t „bar”-ra azokban a sorokban, amelyek tartalmazzák a „hello”-t: :g/hello/s/foo/bar/g. Mind a :s/X/Y/, mind pedig a :g/X/ parancsokban az X kifejezés egy szabályos kifejezés, nem pedig egyszerűen egy statikus karaktersorozat. Ennek kiküszöbölésére a nomagic beállítást állítsuk kikapcsolt állapotba (lásd lejjebb). A szabályos kifejezések megismeréséhez ajánlom ***Büki András*** Bevezetés a szabályos kifejezések használatába című írását a 69. oldalon, illetve ***Jeoffrey Friedl*** Mastering Regular Expressions című könyvét (O'Reilly & Associates, 2. kiadás, 2002).
Külső program futtatásához használd a :! parancsot. A :!ls megjeleníti a pillanatnyi könyvtár tartalmát. Kényel-

mi szolgáltatás, hogy a parancsban a százalékjel (%) helyettesíti az éppen szerkesztett fájl nevét, azaz a `:!chmod +x %` parancs futtathatóvá teszi a fájlt. A `:!!` parancs begépelése megismétli az előző `:!` parancsot. A parancs-üzemmódbeli `!` paranccsal ellentétben ez nem módosítja a szöveget; a program kimenete, ha egyáltalán van, egyszerűen csak megjelenik a képernyőn.

Beállítások

A vi-ban az `ex` `:set` parancs segítségével be lehet állítani a szerkesztő viselkedését. Például a keresőparancsoknál a kis- és nagybetű alapesetben megkülönböztetett; ennek megváltoztatásához használjuk a `:set ignorecase` vagy a `:set ic` lehetőséget. A legtöbb beállításnak van egy-két karakteres rövidítéssel ellátott változata.

Kétféle beállítás lehetséges: a kétértékű (igaz/hamis) beállítások, és azok, amelyek értéket vesznek fel. Egy kétértékű beállítást a „no” előtaggal lehet kikapcsolni; például a `:set noignorecase` vagy `:set noic` kikapcsolja a kis- és nagybetűfigyelést.

Az értéket felvevő beállítások egyenlőségjelet alkalmaznak, amely mögött a beállítandó érték szerepel. Például a `shift` parancs alapértelmezett szóközértéke (>, <, ^T és ^D) nyolc szóköz (egy tab karakter), a legtöbb programozó azonban két vagy négy szóközt részesít előnyben. A `:set shiftwidth=4` (vagy a `:set sw=4`) parancsot kell használni ahhoz, hogy az összes behúzásparancs négy szóközzel rendelkezzen.

Íme néhány egyéb hasznos beállítás:

- ***wrapmargin=N***
  Bekapcsolja a sortörést; a szám azt határozza meg, hogy hány karakterrel a sor vége előtt kell sortörést alkalmazni, például `:set wm=8`.
- ***az autoindent (kétértékű)***
  Minden sort az azt megelőző értékkel húzza be, például `:set ai`.
- ***magic (kétértékű, alapértelmezett igen, nincs rövidítése)***
  Ez szabályozza a szabályos kifejezések viselkedését a keresőparancsokban, például `:set nomagic`.

Több beállítás egyszeri alkalmazásához a parancsokat eggyé össze lehet kapcsolni, mint például `:set noic wm=8 sw=4 nomagic ai`.

A későbbi szerkesztési tevékenységhez a beállításokat menteni lehet, ekkor be kell rakni őket a *.exrc* fájlba. Mikor a vi-t futtatod, az állományban fellelhető összes `ex` (kettőspont üzemmód) parancs végrehajtásra kerül. Hagyd figyelmen kívül a : kettőspontot, egyszerűen írd be a `set` parancsokat.

Egyéb parancsok

A grafikus szerkesztők gördítősávja azt mondja meg, hol tartózkodsz a szövegben; s rá lehet kattintani, ha a fájl egyes részeit szeretnéd látni. Mivel a vi-nak nincs gördítősávja, az ennek megfelelő billentyűzetparancsokat használjuk.

A CTRL-G a fájl állapotát a pillanatnyi sor számával mutatja meg. A fájl egy meghatározott sorára az `XG` paranccsal lehet eljutni. A `G` önállóan a fájl végére ugrik, vagy az `XG` helyett használhatjuk a `:X`-et.

A CTRL-F egy oldallal előremozgatja a képernyőt, a CTRL-B egy oldallal vissza. Ha félképernyőnként szeretnénk mozogni, a CTRL-D (le) és a CTRL-U (fel) parancsokat kell használnunk.

Néhány egyéb parancs:

- Az `x` parancs letörli a pillanatnyi karaktert (a nagybetűs X a bal oldali karaktert törli le). Számelőtaggal több karaktert is letörölhetünk. Az ezzel a paranccsal törölt karakterek ugyanúgy kerülnek kivágásra, mint a `d` parancs esetében.
- A `~` a pillanatnyi karaktert nagybetűről kisbetűre alakítja vagy fordítva.
- Egy hiba visszavonása az `u` parancscsal történik. A `vim`-ben többszintű visszavonás működik. Az `u` parancs megismétlése a sorban minden egyes lépést visszavon; a CTRL-R billentyű leütése pedig visszateszi őket (redo). A vi más változatainak csak egyszintű visszavonása van; az `u` újbóli kiadása egyszerűen visszavonja az előző lépést, visszaállítva a változtatásokat.
- A nagybetűs `U` minden változtatást visszavon, amit eddig a pillanatnyi sorban tettünk; de miután kiléptél az adott sorból, a szolgáltatás már nem működik többé.
- Írd be a . (pont) parancsot az utolsó szerkesztőparancs megismétléséhez.
- Az utolsó `:s///` parancs megismétléséhez gépeljük be a `&` parancsot (ez azonban csak a pillanatnyi sorban viszi véghez a módosításokat).

További olvasnivaló

A `vim` szövegszerkesztőnek van egy jól használható oktatómodulja, amellyel a parancsok nagy részét gyakorolni lehet. Írd be a `vimtutor` parancsot a parancssorba, vagy add ki a `:help tutor` parancsot a `vim`-en belül.

Linux Journal 2003. 114. szám

William R. Ward (wrw@bayview.com)
1987 óta használja a vim+et. Tanár, író és programozó, feleségével, Hollyval a kaliforniai Mountain View-ban él. Cégük, a Bay View Consulting Services (➲ http://www.bayview.com) Perl, Linux/Unix és az ezekhez kapcsolódó témák oktatásával foglalkozik.

A vi története

A vi szövegszerkesztőt (a név a vizuális szó rövidítése) eredetileg *Bill Joy* írta 1976-ban a kaliforniai Berkeley Egyetemen. A program kezdettől fogva szerepelt a Unix BSD-változatában (Berkeley Standard Distribution). Más Unix-változatok, többek között a Linux is, az évek folyamán átvette a programot. A vi szövegszerkesztő az `ex` leszármazottja, ami viszont az `ed`-en alapult. Ezeket a régebbi szövegszerkesztőket arra tervezték, hogy a távírón használják őket, és egyszerre csak egy sor szöveget voltak képesek megjeleníteni – a vi volt az első unixos teljes képernyőt kihasználó szövegszerkesztő.

Az eredeti vi-forráskód a szerzői jogok miatt nem volt nyilvános (egészen néhány hónappal ezelőttig), de sok klónját elkészítették. Az egyik, amelyik a legtöbb Linux-terjesztésben szerepel, a `vim` (továbbfejlesztett vi – vi improved). A jelenlegi szabványos BSD szövegszerkesztő az nvi (új vi – new vi). További vi-szövegszerkesztő például az elvis, a vile és a stevie. Ezek tartalmazzák a vi szolgáltatásait és a saját kiegészítéseiket.

KAPCSOLÓDÓ CÍMEK

A vi szövegszerkesztő és klónjai
➲ http://www.math.fu-berlin.de/~guckes/vi/index.php3
A vi-kedvelők honlapja ➲ http://www.thomer.com/vi/vi.html