V sérii "Na Váš stůl" přinášíme popis obrazovkového editoru "vi", který je používán v operačním systému UNIX. Oproti editorům známým z osobních počítačů má "vi" poněkud nezvyklé ovládání dané tím, že je schopen pracovat prakticky na každém terminálu s libovolnou klávesnicí. Jeho velkou předností je, že je standardní součástí všech verzí operačního systému UNIX. Věříme, že tento stručný přehled funkcí "vi" přijde vhod všem novým uživatelům a zejména programátorům, kteří začínají pracovat s "otevřenými" operačními systémy jako např. SCO UNIX, IBM AIX, atd.

Pracujeme s UNIXovským editorem

vi

O b s a h

Způsob zpracování přehledu

Přehled seznamuje s nejběžnějšími příkazy editoru "vi" v pořadí podle jejich důležitosti a potřebnosti. Možnosti editoru "vi" jsou značně rozsáhlé, a proto jsou některé alternativní způsoby vyvolávání funkcí i některé zřídka užívané funkce, bez kterých se lze obejít, z důvodů přehlednosti textu záměrně vynechány.

Na rozdíl od obecných parametrických popisů jednotlivých příkazů, které jsou uváděny v referenčních manuálech a které jsou vzhledem k nutnosti rozlišování malých a velkých písmen a k přepínání mezi různými módy editoru někdy obtížněji srozumitelné, je zde použita forma příkladů. Způsob používání příkazů je ukázán na konkrétních příkladech vybraných z běžné praxe.

---

1. Módy činnosti editoru vi

a) **příkazový** - v tomto módu je "vi" po spuštění, znaky nejsou vkládány do textu, ale jsou ihned interpretovány jako příkazy - viz kap. 4. a další

Esc ... pípnutím potvrdí nastavení příkazového módu

b) **rozšířený příkazový** - slouží pro vstup složitějších příkazů (např. záměny řetězců v souboru)

: ... přechod do rozšířeného příkazového módu, za znak ":" zobrazený dole na obrazovce se potom píše text příkazu
Enter ... provedení příkazu a návrat do příkazového módu

Q ... trvalý přechod do rozšířeného příkazového módu (někdy se též označuje jako **ex mód**)
vi ... návrat do příkazového módu

c) **vkládací** - umožňuje vkládání textu

a, c, i, o, s,
A, C, I, O, R, S ... přechod do vkládacího módu - viz 6. Vkládání a změny textu
Esc ... návrat z vkládacího módu do příkazového

Poznámka:
Příkazy **/**, **?** (viz 9. Hledání řetězce v souboru) a **!** (viz 13. Spouštění příkazů UNIXu z vi editoru) zadávané v příkazovém módu se zapisují na poslední řádku obrazovky a ukončují stisknutím Enter.

2. Spuštění editoru vi

Práce se souborem "pokus":

vi pokus ... start editoru na první řádce souboru

vi +230 pokus ... start editoru na 230. řádce souboru
vi + pokus ... start editoru na poslední řádce souboru
vi +/novak pokus ... start editoru na prvním výskytu řetězce "novak"

vi -r pokus ... obnoví soubor po event. výpadku systému
vi -r ... zobrazí seznam systémem uložených souborů (r=recovery)

Místo "vi" lze též použít:

vedit pokus ... zobrazuje více hlášení (vhodné pro začátečníky)
view pokus ... soubor lze jen prohlížet, případné změny nelze do právě zobrazeného souboru uložit (stejné jako **vi -R pokus**, R=read only)

3. Ukončení editoru vi

Všechny příkazy kromě **ZZ** se zadávají
v rozšířeném příkazovém módu:

:q ... ukončení bez uložení souboru (quit)
:q! ... dtto u změněného souboru
:x nebo **ZZ** ... ukončení s uložením souboru

:w pokus2 (write)
:q ... zapsání souboru pod jiným jménem a ukončení editoru

:wq pokus2 ... dtto jedním příkazem

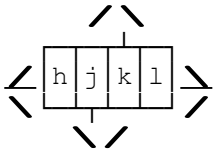
:e! ... zruší všechny změny od posledního uložení souboru (edit)
(stejné jako ukončení bez uložení a nové spuštění editoru)

Další příkazy pro spuštění a ukončení editoru
viz 12. Práce s více soubory najednou

Poznámka:
Pokud v IBM AIXu, který standardně používá Korn shell, místo **Z** (tj. Shift-z) stisknete omylem Ctrl-z, dojde k zastavení procesu a objeví se zpráva [1] + 14103 Stopped vi pokus
a pod ní standardní výzva operačního systému $
Po zadání povelu **fg 14103**, příp. samotného **fg** (=foreground) bez čísla procesu, se vrátíte zpět do editoru.
(Aby po stisknutí Ctrl-z nedocházelo k zastavování procesů, je možno v ksh nastavit **set +m**.)

4. Pohyb kurzoru po obrazovce

h nebo **Šipka_vlevo** ... o jeden znak doleva
j nebo **Šipka_dolů** ... o jednu řádku dolů
k nebo **Šipka_nahoru** ... o jednu řádku nahoru
l nebo **Šipka_vpravo** ... o jeden znak doprava



w ... na začátek následujícího slova (word)
b ... na začátek předcházejícího slova (back)
W, B ... dtto po "velkých" slovech - oddělovačem je pouze space, tab a newline, interpunkční znaménka se ignorují
e, E ... na konec daného nebo následujícího slova - "malého" nebo "velkého" (end)

0 ... na začátek řádky = do prvního sloupce
$ ... na konec řádky = na poslední znak na řádce
^ ... na první znak na řádce
15| ... do 15. sloupce

H ... na začátek první řádky obrazovky (home)
M ... na začátek prostřední řádky obrazovky (middle)
L ... na začátek poslední řádky obrazovky (low)

5. Pohyb po editovaném souboru

Ctrl-f nebo **PgDn** ... posun o celou obrazovku vpřed (forward)
Ctrl-b nebo **PgUp** ... posun o celou obrazovku zpět (backward)

Ctrl-d ... posun o 1/2 obrazovky vpřed, tj. dolů (down)
Ctrl-u ... posun o 1/2 obrazovky zpět, tj. nahoru (up)

Ctrl-e ... posun o 1 řádku vpřed, tj. směrem ke konci souboru
Ctrl-y ... posun o 1 řádku zpět, tj. směrem k začátku souboru
(pozice kurzoru vzhledem k souboru se nemění)

28G ... skok na 28. řádku souboru (goto)
G ... skok na poslední řádku souboru

ma ... označí aktuální řádku značkou "a" (mark)
(jako značky lze používat malá písmena "a" až "z")
'a ... přemístí kurzor na začátek řádky, která byla označena značkou "a"

58j nebo **58***Enter* ... skok o 58 řádek dolů, tj. vpřed
34k nebo **34-** ... skok o 34 řádek nahoru, tj. zpět

z*Enter* ... aktuální řádka, tj. řádka, na níž je kurzor, se nastaví jako první řádka obrazovky (zero) =zacílit
z. ... řádka, na níž je kurzor, se nastaví doprostřed obrazovky
z- ... řádka, na níž je kurzor, se nastaví jako poslední řádka obrazovky

} ... na začátek následujícího odstavce odděleného prázdnou řádkou
{ ... na začátek předcházejícího odstavce

6. Vkládání a změny textu v souboru

i ... následný text bude vložen před kurzor (insert)
a ... následný text bude vložen za kurzor (append)
I ... následný text bude vložen před první znak aktuální řádky
A ... následný text bude vložen na konec řádky
o ... následný text bude vložen za aktuální řádku (open)
O ... následný text bude vložen před aktuální řádku

R ... následný text bude přepisovat řádku od pozice kurzoru (replace)

cw ... následný text bude nahrazovat řádku, a to od pozice kurzoru do konce slova označeného znakem $ (change)
cW ... dtto až po mezeru nebo tabelátor
c$ nebo **C** ... následný text bude nahrazovat zbytek řádky
cc nebo **S** ... následný text bude nahrazovat celou řádku
s ... následný text bude nahrazovat jeden znak (substitute)

Příkazy zadávané ve vkládacím módu:

Ctrl-h nebo **Backspace** ... maže poslední znak
Ctrl-w ... maže poslední slovo
Pozor: Na obrazovce vymazaný text zůstává!
Ctrl-v .. ruší speciální význam následujícího znaku - např. **Ctrl-v Ctrl-m** zapíše **^M**
Pozor: běžně nelze vložit Ctrl-j (=LF), Ctrl-q (=XON) a Ctrl-s (=XOFF)
Enter ... ve vkládacím módu vyvolaném výše uvedenými příkazy ukončí řádku a kurzor přejde na novou řádku
(v příkazovém módu přesune kurzor na první znak na další řádce)
Esc ... ukončí vkládání textu a nastaví opět příkazový mód

Příkazy zadávané v příkazovém módu:

r*znak* ... přepíše znak na aktuální pozici kurzoru znakem *znak* (replace)
. ... zopakuje předcházející příkaz vkládání, přepisování nebo zaměňování textu od nové pozice kurzoru
u ... zruší poslední provedenou změnu (undo)

Další příkazy týkající se vkládání textu do souboru viz 17. Odsazování a zalamování textu

---

Poznámka:
Z rozšířeného příkazového módu (tzv. řádkového) lze příkazem **a**, **i** nebo **c** vyvolat řádkový vkládací mód - na rozdíl od výše uvedeného (vizuálního) vkládacího módu vyvolávaného z (vizuálního) příkazového módu. Normálně se ukončuje zadáním *.Enter* - tj. zapsáním tečky na samostatném řádku, nebo se vkládací mód zruší bez zapsání textu stisknutím **Ctrl-c**.

7. Dělení a spojování řádek

i*Enter*
Esc ... rozdělí řádku na dvě v místě kurzoru

J ... připojí další řádku na konec aktuální řádky (join)
5J ... spojí 5 řádek, tj. 4 následující řádky připojí na konec aktuální řádky

8. Mazání textu v souboru

x ... smaže znak na aktuální pozici kurzoru
X ... smaže znak před aktuální pozicí kurzoru
dd ... smaže aktuální řádku (delete)
5dd ... smaže 5 řádek počínaje aktuální řádkou
dw ... smaže text od pozice kurzoru do konce slova
3dw ... smaže text od pozice kurzoru do konce 3. slova
db ... smaže text od pozice kurzoru do začátku slova, tj. zpět
d$ nebo **D** ... smaže text od pozice kurzoru do konce řádky
d/nechat ... smaže vše - i několik řádek - až k řetězci "nechat"
dt/ ... smaže text od pozice kurzoru až ke znaku "/" na dané řádce (to)
df) ... smaže text od pozice kurzoru až do znaku ")" včetně (find)
. ... zopakuje předcházející příkaz mazání od nové pozice kurzoru

Mazání bloku textu pomocí značek a čísel řádek:

po nastavení kurzoru na první řádku bloku
ma ... označí tuto řádku značkou "a" (mark)
po nastavení kurzoru na poslední řádku bloku
d'a ... smaže blok od značky "a" do aktuální řádky
(stejné jako **:'a,d** v rozšířeném příkazovém módu - viz dále)
:set nu ... zobrazí čísla řádek –
viz 16. Nastavení prostředí editoru
:244,560d ... smaže řádky od 244. do 560. včetně
:25,.d ... smaže řádky od 25. do aktuální včetně
(tečku, která zastupuje číslo aktuální řádky, lze vynechat)
:-2,+2d ... smaže aktuální řádku, 2 předcházející a 2 následující
:.,$d nebo **dG** ... smaže vše od aktuální řádky až do konce souboru
:'a,'bd ... smaže blok řádek od značky "a" do značky "b" včetně

Obnovení smazaných řádek:

p ... vloží naposledy smazaný text (put)
"2p ... vloží předposledně smazané řádky za aktuální řádku
u ... zruší nechtěně vloženou řádku (řádky)
. ... zopakuje příkaz p s bufferem n+1

Poznámka:
Bezprostředně po ukončení libovolného mazání je smazaný text uložen do nepojmenované vyrovnávací paměti (bufferu) a kromě toho se smazané řádky také postupně ukládají do 9 tzv. delete buffers označených čísly 1 až 9. Celkem je tedy uloženo 9 posledních mazání.

9. Vyhledávání řetězců v souboru

/novák ... hledá následující výskyt řetězce "novák"
?josef ... hledá předcházející výskyt řetězce "josef"

n ... opakuje poslední hledání (next)
N ... dtto v opačném směru

/jana/-2 ... vyhledá řetězec "jana" a nastaví kurzor o 2 řádky zpět
/.ana ... hledá řetězce "dana", "hana", "jana" atd.
/an*a ... hledá řetězce "aa", "ana", "anna", "annna" atd.
/[pP]epa ... hledá následující výskyt řetězců "pepa" nebo "Pepa"
?^jana ... hledá předcházející řádku začínající řetězcem "jana"
/nová$ ... hledá řádku, která končí řetězcem "nová"
/jana\.pepa ... hledá řetězec "jana.pepa"
Pozor: Speciální tzv. metaznaky **. * [^ $ ~ & / **
musí být uvozeny znakem ****.
/\<jan\> ... vyhledá jen slovo "jan" a ne např. "krajan" nebo "jana"

Příkazy ovlivňující vyhledávání řetězců:

:set ignorecase ... při hledání se nerozlišují malá a velká písmena
(zkráceně **:set ic**)
:set nowrapscan ... hledá se jen do konce nebo začátku souboru, tedy ne cyklicky
:set nomagic ... nebere **. * [** jako metaznaky, speciální význam mají jen znaky **^** a **$**, ostatní jen jsou-li uvozeny ****

10. Zaměňování řetězců v souboru

:s/josef/pepa/ ... na aktuální řádce nahradí první výskyt řetězce "josef" řetězcem "pepa" (substitute)
:s/josef/pepa/g ... na aktuální řádce nahradí všechny výskyty (global)
:4,8s/josef/pepa/ ... na řádcích 4 až 8 včetně nahradí první výskyty
:13,20s ... dtto na řádcích 13 až 20 včetně
:.,$s/josef/pepa/g ... od aktuální řádky do konce souboru nahradí všechny výskyty
:1,$s/josef/pepa/g ... v celém souboru nahradí všechny výskyty (místo **1,$** stačí napsat jen **%**)

Selektivní nahrazování, tj. s potvrzováním:

:5,18s/josef/pepa/gc ... na řádcích 5 až 18 včetně postupně vyhledává všechny výskyty řetězce, řádku s hledaným řetězcem opíše a nahrazovaný řetězec podtrhne: **josef** (confirm)
^^^^^
Po zadání:
y*Enter* ... nahradí řetězcem "pepa", jinak
Enter ... přeskočí

:%s/josef/pepa/gc ... dtto v celém souboru
:g/josef/s//pepa/gc ... dtto v celém souboru - viz dále

Operace s vybranými řádky:

:g/jana/s/josef/pepa/gc ... na všech řádcích obsahujících řetězec "jana" selektivně nahrazuje řetězec "josef" řetězcem "pepa"
:.,$g/pepek/d ... od aktuální řádky do konce souboru vymaže řádky obsahující řetězec "pepek"
:g/^#/co$ ... všechny řádky začínající znakem "#" zkopíruje na konec souboru - viz dále
:g!/jana/p ... všechny řádky neobsahující řetězec "jana" vypíše (print)

11. Kopírování a přesouvání textů

a) **přes nepojmenovanou paměť - buffer**
(lze jen v rámci jednoho souboru)

Při kopírování se používá příkaz "yank" (škubnout) - viz dále, při přesouvání příkaz "delete" (vymazat) - viz 8. Mazání textu v souboru.

yy nebo **Y** ... uloží aktuální řádku do bufferu (yank)
5yy ... uloží 5 řádek počínaje aktuální
3yw ... uloží 3 slova z aktuální řádky
y$ ... uloží text od pozice kurzoru do konce řádky
yf] ... uloží text od kurzoru až do znaku "]" včetně (find)
2yt, ... uloží text od kurzoru až ke 2. znaku "," (to)

p ... vloží řádku(y) z bufferu za aktuální řádku (put)
P ... vloží řádku(y) z bufferu před akt. řádku,
příp. slovo(a) z bufferu před kurzor

b) **přes pojmenované buffery "a" až "z"**
(lze použít i ke kopírování bloků textu mezi soubory)

Kopírování se provádí příkazem "y", přesouvání textů příkazem "d".

"ayy ... uloží aktuální řádku do bufferu "a"
"b7yy ... uloží 7 řádek počínaje aktuální do bufferu "b"
"B3yy ... připojí 3 řádky na konec bufferu "b"
"cy$ ... uloží text od kurzoru do konce ř. do bufferu "c"
B l o k ř á d e k :
po nastavení kurzoru na první řádku bloku
ma ... označí tuto řádku značkou "a"
po nastavení kurzoru na poslední řádku bloku
"dy'a ... uloží do bufferu "d" blok začínající značkou "a"

"ep ... obsah bufferu "e" vloží za aktuální řádku
"eP ... obsah bufferu "e" vloží před aktuální řádku
nebo - pokud se nevkládá celá řádka - před kurzor

c) **pomocí příkazů "copy" a "move"**

:2,10co25 ... řádky 2 až 10 včetně vloží za řádku 25 (copy)
:2,10m25 ... dtto a na původním místě je smaže (move)
:'a,'bm. ... blok řádek přesune za aktuální řádku
:m$ ... aktuální řádku přesune na konec souboru
:,+1co0 ... aktuální řádku a řádku následující zkopíruje na začátek souboru

12. Práce s více soubory najednou

vi pokus1 pokus2 pokus3 ... postupné editování 3 souborů, nejprve se zobrazí "pokus1"
vi pokus*
vi pokus[1-3] ... jiné způsoby zadání více souborů

:w ... uloží aktuální soubor před přechodem k dalšímu
:n ... zobrazí další soubor v pořadí (next)
Pozor: Nejde to cyklicky!
:n! ... dtto bez uložení změn v aktuálním souboru
:rew ... návrat na začátek seznamu (rewind)
:rew! ... dtto bez uložení změn v aktuálním souboru
:e pokus3 ... skok na soubor mimo pořadí (edit)
:e /u/abc ... otevření souboru, který není uveden v seznamu
(zde je to soubor "abc" v adresáři "/u")
:e# ... návrat na předcházející soubor
(v AIXu též příkazem **Ctrl-a** = alternate file,
v SCO UNIXu **Ctrl-^**)
:e!# ... dtto bez uložení aktuálního souboru
(metaznak **#** zastupuje jméno předcházejícího souboru)
:n test1 test2 test3 ... definuje nový seznam souborů
:ar ... vypíše aktuální seznam souborů (args)
(právě editovaný soubor je v lomených závorkách)

Vkládání a zapisování souborů:

:r jana ... za aktuální řádku vloží soubor (read)
"jana", aktuální řádka se nezmění
:17r hana ... za 17. řádku vloží soubor "hana"

:w posta/alena ... editovaný soubor zapíše do sou- (write)
boru "alena" v adresáři "posta"
:w! pokus2 ... přepíše již existující soubor
:w>>ivan ... aktuální soubor připojí na konec souboru "ivan"
:20,50w karel ... řádky 20 až 50 včetně uloží do souboru "karel"
:20,50w>>pavel ... řádky 20 až 50 včetně připojí na konec souboru "pavel"

Kopírování části jednoho souboru do druhého
~~~~~~

i) Otevřete zdrojový soubor.

ii) Zvolenou část souboru uložte do bufferu "a" (nebo jiného) - viz 11. Kopírování a přesouvání

iii) Příkazem **:w** uložte zdrojový soubor (byl-li změněn).

iv) Příkazem **:e** *soubor2* přejděte do cílového souboru.

v) Nastavte kurzor na řádku, za níž chcete blok vložit.

vi) Příkazem **"ap** vložte buffer "a" (nebo jiný) do cílového souboru.

Vložení začátku nebo konce jiného souboru
viz 13. Spouštění příkazů UNIXu z editoru vi

13. Spouštění příkazů UNIXu z editoru vi

:cd /u/jana ... nastaví zvolený adresář (change directory) - vnitřní příkaz vi
:cd ... nastaví domovský adresář definovaný v proměnné $HOME
:sh ... spustí další shell umožňující vykonávat příkazy operačního systému, po stisknutí **Ctrl-d**, ev. po zadání **exit**, návrat zpět do editoru
:!ls -l ... provede jeden příkaz operačního systému - zde např. "ls -l" vypíše seznam souborů v aktuálním adresáři, po stisknutí **Enter** návrat do editoru
:!pwd ... vypíše jméno aktuálního adresáře (print working directory)

Předání části editovaného souboru na standardní vstup příkazu UNIXu:

:,+120w !lp ... zadaný rozsah řádek je zpracován příkazem operačního systému - zde např. "lp" vytiskne aktuální řádku a 120 následujících

Vložení standardního výstupu příkazu UNIXu do editovaného souboru:

:r !ls ... provede příkaz operačního systému a jeho výstup vloží za aktuální řádku - zde např. "ls" vypíše jména souborů, každé na samostatný řádek
!!ls ... dtto, ale výstup přepíše aktuální řádku a další řádky jsou vloženy

:r !head -15 karel ... vloží prvních 15 řádků ze souboru "karel"
:r !tail -20 pavel ... vloží posledních 20 řádků ze souboru "pavel"
:r !tail -n 130 ivan ... vloží část souboru "ivan" od 130. řádku do konce

Používání filtrů - tj. část editovaného souboru je zpracována příkazem UNIXu a nahrazena:

:5,$!sort ... setřídí od 5. řádky do konce souboru
15!!sort ... setřídí 15 řádek počínaje aktuální řádkou
!Lsort ... setřídí od aktuální řádky do konce obrazovky
!Gsort ... setřídí od aktuální řádky do konce souboru
!'asort ... setřídí od aktuální řádky do řádky označené "a"

14. Různé další příkazy editoru vi

Ctrl-g ... zobrazí informace o právě editovaném souboru na poslední řádce obrazovky:
"pokus" [Modified] The cursor is at line 23 of 50 --46%-- .

Ctrl-l, (clear+redraw)
Ctrl-r ... obnoví obsah obrazovky (redraw)
(např. po zobrazení systémových hlášení)

. ... opakuje poslední příkaz mazání, nahrazování, vkládání textu apod.
u ... zruší poslední příkaz mazání, změny (undo) nebo vkládání nezávisle na délce textu
U ... obnoví aktuální řádku do stavu před editací

% ... přesouvá kurzor na příslušnou otvírací, resp. zavírací závorku kulatou (), hranatou [] nebo složenou {}
~ ... vzájemně zaměňuje malá a velká písmena

15. Příkazy abbrev, map a source

:ab jn Josef Novák ... definuje zkratku "jn" (abbrev)
po zapsání slova " jn ", " jn,", "/jn/" atp.
se zkratka "jn" (pokud není součástí slova)
nahradí řetězcem "Josef Novák"

:ab ... vypíše definované zkratky
:una jn ... zruší zkratku "jn"

Často užívané zkratky je vhodné zapsat do souboru příkazů - viz dále, nebo do konfiguračního souboru - viz 16. Nastavení prostředí editoru.

:map v :q! ... definuje funkci přiřazenou "v" (map)
v příkazovém módu se pak po stisknutí klávesy "v"
napíše příkaz ":q!" a stačí stisknout Enter

:map ... vypíše definované klávesy
:map! ... dtto pro vkládací mód
:unmap v ... zruší definici klávesy "v"

:map # I{zrušit}^[... po stisknutí klávesy "#" se na začátek aktuální řádky vloží řetězec "{zrušit}" a vkládací mód se ukončí *Esc* */

:map s :s/josef/pepa/^M ... po stisknutí "s" se provede příkaz ":s/josef/pepa/", který zamění první výskyt řetězce na aktuální řádce

Ctrl-v s ... vyvolá původní funkci klávesy "s"
(tj. nahrazení jednoho znaku a přepnutí do vkládacího módu)

:map ^[a :!vihlp^M ... klávese F1 na terminálu IBM 3151 **/ přiřadí funkci (shellscript) zobrazující nápovědu k editoru vi

:map ^[g ^B ... klávese F7 přiřadí funkci Backward (PgUp)
:map ^[h ^F ... klávese F8 přiřadí funkci Forward (PgDn)
:map! ^[g ^[^Ba ... dtto pro vkládací mód
:map! ^[h ^[^Fa

Výše uvedené příkazy abbrev, map a případně i jiné příkazy lze zapsat do souboru příkazů:

:so def1 ... provede příkazy zapsané v souboru příkazů "def1" (source)

vi "+so zkr" pokus ... start editoru a provedení příkazů ze souboru "zkr"

Poznámky:

*/ Před vkládáním řídicích znaků je třeba napřed stisknout **Ctrl-v**, čímž se zruší speciální význam následujícího znaku. Například: **^[** se zapíše stisknutím kláves **Ctrl-v Esc**
^M se zapíše stisknutím kláves **Ctrl-v Enter**

**/ Kódy jednotlivých funkčních kláves závisí na typu terminálu. Lze je nalézt v technické dokumentaci k terminálu nebo v popisu emulace terminálu na PC.

16. Nastavení prostředí editoru vi

:set all ... vypíše aktuální nastavení všech voleb
:set ... vypíše pouze změněné volby

Abecední přehled nejdůležitějších voleb:

:set autoindent ... viz 17. Odsazování a zalamování textu
:set ignorecase ... viz 9. Hledání řetězce v souboru
:set list ... viz 17.
:set magic ... viz 9.
:set mesg ... přijímání zpráv ve vi povoleno
:set number ... před řádky souboru jsou jejich čísla
:set readonly ... znemožňuje zapsání souboru
:set report=5 ... po kolika zpracovaných řádkách se vypíše zpráva
:set shiftwidth=8 ... viz 17.
:set showmode ... zobrazuje zprávu "INPUT MODE" a další
:set tabstops=8 ... viz 17.
:set term=$TERM ... nastavuje typ terminálu
:set warn ... varování "No write ..." po zadání !*cmd*
:set wrapmargin=0 ... viz 17.
:set wrapscan ... viz 9.

:set no*option* ... příslušnou volbu vypíná

Po spuštění editoru příkazem	je standardní počáteční nastavení
vi nebo ex	magic nonovice noreadonly report=5 noshowmode
view	" " readonly " "
vedit nebo edit	nomagic novice noreadonly report=1 showmode

Většinu voleb lze zkrátit a jedním příkazem **:set** lze nastavit několik voleb najednou (viz následující příklady).

Trvalá změna nastavení editoru:
- pomocí konfiguračního souboru **.exrc** v domovském adresáři

```
" komentář začíná uvozovkou
set ignorecase sw=4 showmode
"set number ai
abbrev jn Josef Novák
map v :q!
```

- pomocí proměnné **EXINIT**

export **EXINIT='set nu nomesg sw=4 smd | map v :q!'**

Poznámka:
Příkaz **EXINIT=** je možno zadat na příkazové řádce nebo vložit do souboru **.profile**.
Editor se nejprve řídí obsahem proměnné EXINIT. Pokud proměnná EXINIT není nastavena, provedou se příkazy v souboru **$HOME/.exrc** (pokud existuje). Aby editor vzal soubor .exrc v úvahu, je třeba proměnnou EXINIT (pokud existuje) zrušit příkazem **unset EXINIT**.

17. Odsazování a zalamování textu

:set autoindent ... při vkládání nového textu bude každá nová řádka vlevo odsazena stejně jako předcházející
(příkaz **:set noai** automatické odsazování opět vypne)

:set shiftwidth=4 ... nastavuje softwarové tabelátory používané volbou autoindent a dále popsanými příkazy **Ctrl-t**, **Ctrl-d**, **>** a **<**

:set tabstops=8 ... nastavuje hardwarové tabelátory
:set hardtabs=8

:set list ... zobrazí tabelátory jako ^I a konce řádků jako $
(příkaz **:set nolist** tento způsob zobrazování opět vypne)

:%l ... jednorázově vypíše celý soubor tímto způsobem (list)

Změna odsazení při vkládání textu:

Ctrl-i nebo **Tab** ... vloží tabelátor, tj. odsadí dle nastavení "ts"
Ctrl-t ... vloží tabelátor nebo potřebný počet mezer dle "sw"
Ctrl-d ... zruší tabelátor, tj. návrat na předcházející zarážku
^Ctrl-d ... zruší odsazení pro danou řádku

Posunutí již napsaného textu v příkazovém módu:

5>> ... posune 5 řádek o 1 tabelátor (sw) doprava
<< ... posune daný řádek doleva
. ... zopakuje předcházející příkaz
>L ... posune řádky od aktuální až do konce obrazovky doprava
<G ... posune řádky od aktuální až do konce souboru doleva
:37,58> ... posune řádky od 37. do 58. včetně doprava

Zalamování pravého okraje vkládaného textu:

:set wrapmargin=15 ... vkládaná slova překračující nastavený pravý okraj - zde 15 znaků od konce řádky - se budou automaticky přesouvat na novou řádku
(příkaz **:set wm=0** zalamování opět vypne)

18. Vyvolání vi v rámci příkazů more a man

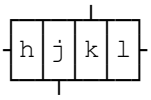
more *file(s)* ... postupně vypisuje soubor(y) po obrazovkách
man tar ... pomocí more zobrazí zvolený popis
Za hlášením na posledním řádku
--More--(6%) *lze zadávat podpříkazy:*
h ... zobrazí seznam podpříkazů příkazu more
v ... spustí editor vi a nastaví na stejný řádek
Potom lze používat příkazy vi, například:
/Examples ... vyhledá příklady
:246,285w tar.examples ... zapíše zvolenou část návodu do souboru
:q ... ukončí vi
q ... ukončí more

19. Editování příkazové řádky v ksh

Aby bylo možné v Korn-shellu příkazy editovat, je nejprve třeba nastavit volbu **-o vi**, což lze udělat příkazem **set -o vi** nebo lépe nastavením proměnné VISUAL nebo EDITOR příkazem **export EDITOR=vi**, neboť pak bude tato volba platit i pro subshell. Uvedený příkaz je vhodné zapsat do souboru "$HOME/.profile" nebo "/etc/profile". (Kromě "vi" lze pro editování příkazové řádky ještě volit editor "emacs" nebo "gmacs".)

Esc ... přepne z vkládacího módu do příkazového (při dalším stisknutí potvrdí pípnutím)

Příkazy zadávané v příkazovém módu:



k ... postupně zobrazuje dříve zadané příkazy
j ... listuje historií příkazů opačným směrem
l ... pohyb kurzoru po znacích vpravo
h ... pohyb kurzoru po znacích vlevo
Pozor: Kurzorové šipky zde nefungují!
w, b ... pohyb kurzoru po slovech vpřed, resp. zpět
0, $ ... přemístění kurzoru na začátek, resp. konec příkazu

/tar*Enter* ... vyhledá naposledy zadaný příkaz obsahující řetězec "tar"
/^t*Enter* ... vyhledá příkaz začínající řetězcem "t"
n, N ... vyhledá další dříve, resp. později zadaný příkaz

x, X ... vymaže znak nad, resp. před kurzorem
D ... vymaže zbytek příkazu od pozice kurzoru
r*znak* ... nahradí jeden znak

i ... přepne do vkládacího módu
A, I ... přemístí kurzor na konec, resp. začátek příkazu a přepne do vkládacího módu
cw ... vymaže slovo a přepne do vkládacího módu
R ... vymaže zbytek příkazu od pozice kurzoru a přepne do vkládacího módu
S ... vymaže celý příkaz a přepne do vkládacího módu (substitute)

Enter ... provede právě zobrazený příkaz (nezávisle na tom, zda je nastaven vkládací nebo příkazový mód)

... před příkaz vloží znak "#" uvozující komentář a řádku odešle (řádka se zaznamená do historie příkazů, aniž by se provedla)
= ... pokud je kurzor na expanzním znaku, tj. na *, ? nebo mezi [ ], vypíše jména generovaných souborů

Speciální příkazy definované pomocí alias, které se zadávají ve vkládacím módu:

r*Enter* ... znovu provede naposledy zadaný příkaz
history ... zobrazí 16 naposledy zadaných příkazů

20. Souhrnný přehled příkazů editoru vi

Příkazy v příkazovém módu

Příkazy, které se nejčastěji potřebují, jsou podtrženy.

1	2	3	4	5	6	7	8	9	0=column1

Quit to ex	WordForwd wordForwd	EndWord endWord ^E=RollDn	ReplEOL replChar ^Redraw	ToCharBck toCharFrw ^Tag	ZZ=save+q zeroScrn	UndoCurL undo ^Up 1/2	Insert(^) insert	OpenNewLB openNewLA	PutBef putAft ^PrevL

Append($) append	SubstLine substChar	DeleteEOL delete ^Down 1/2	FindCharB findCharF ^Forward	GotoLine ^G=status	Home h=cursorL	JoinLines j=cursorD ^J=LF	k=cursorU	LowScreen l=cursorR ^L=clr+^R

YankLine yank ^Y=RollUp	X=delBack x=delChar	ChangeEOL change	V	BackWord backWord ^Backward	NextStrng nextStrng ^NextLine	MiddleScr mark ^M=CR	?=search previous	:=command .=repeat

Speciální znaky

- pro pohyb kurzoru: **0** **^** **|** **$** **%** **'** ` + - ( ) { }
- v módu poslední řádky: **:** **/** **?** **!**
- ostatní: **.** **>** **<** **~** **"** @ ; ,

Příkazy v rozšířeném příkazovém módu

$_r$ =rozsah (range) - viz níže

:**ab**brev	:**f**ile	:**map**	:**pu**t	:**se**t	:**unm**ap	:**z**
:**ar**gs	:$_r$**g**lobal	:**n**ext	:**q**uit	:**sh**ell	:**vi**sual	:**!**
:**c**h**d**ir	:$_r$**j**oin	:$_r$**nu**mber	:**r**ead	:**so**urce	:$_r$**w**rite	:$_r$**<**
:$_r$**co**py	:$_r$**l**ist	:**o**pen	:**rec**over	:**ta**g	:**wq**	:$_r$**>**
:$_r$**d**elete	:$_r$**m**ove	:**pre**serve	:**rew**ind	:**u**ndo	:**x**	:$_r$**&**
:**e**dit	:**ma**rk	:$_r$**p**rint	:$_r$**s**	:**una**bbrev	:$_r$**y**ank	:**=**

Příkazy "global" a "substitute": :$_{rozsah}$**g/**co_vyhledat**/s/**co_nahradit**/**čím_nahradit**/gc**

Před některými příkazy se uvádí rozsah, a to v některém z následujících tvarů: **%** 1,$ adr_1,adr_2 adr_1;adr_2

kde *adr* může být zadáno
- absolutně: **0** **1** *n* **$**
- relativně: **.** *-n* *+n* **-** **+** **--** **++**
- jménem: **'a** ... **'z**
- obsahem: */text/* **?***text***?**

Příkazy ve vkládacím módu

Ctrl-**h** Ctrl-**w** Ctrl-**t** Ctrl-**d** Ctrl-**i** Ctrl-**v** *Esc*

Literatura

[1] IBM AIX Version 3.2 for RISC System/6000:
Editing Concepts and Procedures. (GC23-2212-01)

[2] IBM AIX Version 3.2 for RISC System/6000:
Commands Reference Volume 4 - sa through ypxfr.
(GC23-2393-00)

[3] SCO UNIX System V/386 Operating System R3.2 V2.0:
User's Guide.

[4] SCO UNIX System V/386 Operating System R3.2 V2.0:
User's Reference (C) (M) (F).

[5] Háněl, J.: UNIXovský editor "vi" - Editační minimum
pro začátečníky v UNIXu. Bajt 1/91, s. 34-35.

Připraveno v rámci projektu ADIS (Automatizovaný daňový informační systém) firmy IBM
a uveřejněno v časopise PC WORLD Czechoslovakia.

(HF 11/92)