Hrůzostrašný Vim,

aneb jak jsem objevoval funkce Vimu.

Leoš Maršálek

31. prosince 2003

Obsah

Kapitola 1

Úvodem

1.1 Pravidla pro distribuci tohoto dokumentu

Tento dokument je public domain. Může být tištěn a distribuován zdarma pouze v nezměněné podobě. Je-li dokument měněn nebo je-li jeho část užita v jiném dokumentu, potom seznam autorů musí obsahovat jméno autora a také musí obsahovat jména autorů kteří tyto změny provedly. (Nejlépe i s kontaktem na dané autory.) Pro komerční využití platí zásady uvedené v GNU Public Licence.

1.2 Kontakty na autora

Připomínky, návrhy a komentáře k tomuto dokumentu posílejte na mou E-mailovou adresu

Leos.Marsalek@tiscali.cz.

Adresa mých Webových stránek je :

www.sendme.cz/goro, *www.sendme.cz/sklad*,
a *www.goro.pescz.cz*

1.3 Důvody proč vznikl tento dokument

Důvody jsou velice prosté. Prvním důvodem bylo se konečně přinutik k *pořádnému* naučení LaTeXu [1]. Tím druhým a hlavním důvodem bylo to, že jsem nenašel v Linuxu žádný textový editor, který by mi vyhovoval. Ne že bych kladl na textove editory nějaké vysoké nároky, ale podle mě by měl být editor na úpravu textů především rychlý a to jak spouštěním tak prací sním. Tak jsem objevil Vim a sepsal jsem své trapení sním, v domnění že bych tím mohl někomu ušetřit pár horkých chvilek kterých jsem si užil s učením Vimu nespočet.

1.4 Poděkování

Zde chci poděkovat všem, kteří mě podporovali a také všem, kterým jsem psaním tohoto dokumentu *lezl na nervy svým rozmlouváním* s výpočetní technikou. Zejména ve chvílích kdy nechtěla dělat to co se po ní chtělo.

Kapitola 2

Objevování síly editoru Vim

2.1 Co mě přivedlo k Vimu?

O počítače jsem se začal zajímat na střední škole, kde jsem odrostl na MS DOSu, což byl ve té době jediný systém který byl hojně osazován na počítače a který znali tehdejší učitelé. V roce 1995 jsem navštívil poprvé Invex[1] a zrovna Microsoft tam prezentoval svůj nový operační systém Windows 95. Tenkrát jsem byl nadšen jednoduchostí ovládání a grafikou, jenže za nějaký čas jsem začal preferovat efektivitu a ergonomii ovládání před jednoduchostí. A náhle mi svět Windowsů připadal malý. V té době jsem začal prvním rokem navštěvovat vysokou školu a tam jsem narazil na operační systém Linux. Tenkrát mě několik linuxových nadšenců se mě snažilo přesvedčit že Linux je lepší než Windows, ale já jsem jim nějak nechtěl dát zapravdu. Přece jen okýnka se mě v té době zdály jednodušší na ovládání. Jenže jak čas plynul, tak jsem se o Linuxu dozvýdal čím dál tím více informací a také jsem si vyzkoušel některé věci standartně dodávané v distribucich Linuxu a začal jsem se v Linuxu čím dál tím více pohybovat. Především mě nadhly tyto věci: LaTeX 2$_\varepsilon$, TeX, firewall, postscript práce s PDF a spousta dalších věcí.

Jenže stale mi chyběl editor který by splňoval mé nároky. Kwrite mi připadal těžkopádný, gedit nebyl všude nainstalován a editor v Midnight commanderu si na mém počítači nerozuměl s češtinou. Tak jsem si pročetl několik diskuzních fór a vždy jsem slyšel na Vi nebo Vim samou chválu. Tak jsem si řekl, že jej musím vyzkoušet, a tady jsou mé dojmy.

2.2 První dojmy

Při prvním spuštění editoru Vim jsem si říkal co za šílence tento editor používá, vždyť ani nedá standartně ani opustit. Žádná z mě dosut známých klavesových zkratek nefungovala, a mě nezbylo než použít mé v Linuxu málo používáné funkce kill Opět jsem sednul k internetu a pročítal jsem si diskuzní fóra. Tenkrát mě docela zaujal jeden názor, že když

[1] Veletrh v Brně s výpočetní technikou

se člověk donutí a překoná svou prvotní averzi k ovládání vimu, tak že bude velmi mile překvapen, protože Vim byl psán především pro programátory a jeho příkazy dokáží velmi z efektivnit práci. Jen to chce se pořádně prokousat ovládáním.

2.3 Rozdíl mezi editorem Vi,Vim a GVim.

Editor Vi je nejstarší a byl nasazován na první UNIXové stroje. Byl velice jednoduchý, ale přitom byl velmi efektivní editor. Díky němu jsou dnes ve Vimu a GVimu implementovány prvky, které nám na první pohled připadají zbytečné, ale v tehdejší době měly svůj význam. Například k pohybu k textu využíváme šipek na klavesnici, ale můžeme i využít klávesy j,k atd.

Vim je mladší a díky tomu jsou vněm implementovány i věci co se objevili teprve nedávno. Obecně se dá říci že rozdíly mezi Vi a Vim jsou minimální a kdo umí psát ve Vi tak umí i Vim. Opačně to ale už neplatí.

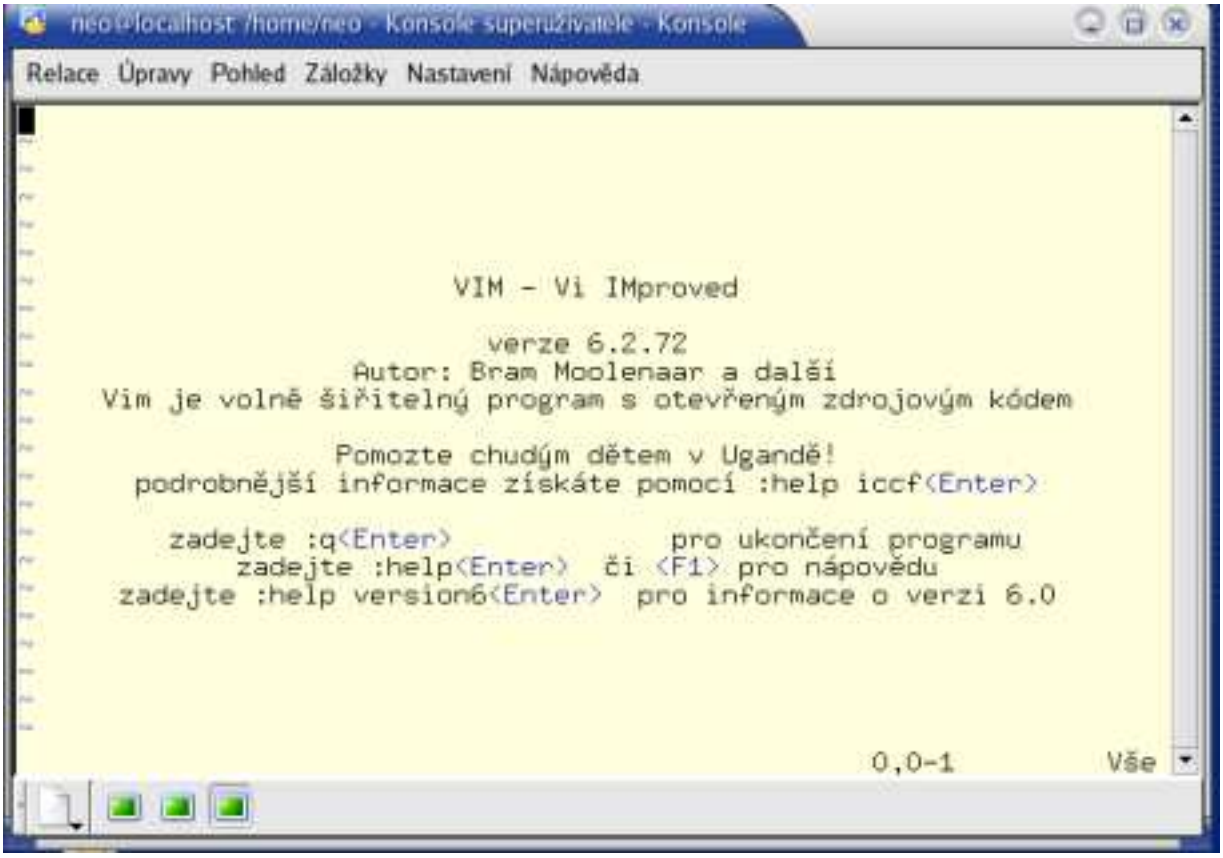
Posledním editorem je GVim. Je to vlastně Vim s grafickým rozhraním. Je v něm možnost obsluhovat Vim pomocí tlačítek. Asi největší změna je využití myši k pohybu po textu. GVim je již trochu náročnější na hardware, ale v dešní době jsou již počítače tak výkonné že tuto otázku dnes nemusíme řešit.

Vim a Gvim jsou stále vyvíjeny a lze je implementovat pod jakýkoliv operační systém. Od Windows počínaje přes Linux, Unix až po Irix[2]. Vim i GVim si lze bezplatně stáhnout z *www.vim.org*

[2]Irix je operační systém založený na UNIXu vyvýjený firmou SGI

2.4 Tak teda začínáme.

Při prvním spuštění se pře Vámi objeví tato obrazovka.



Obrázek 2.1: úvodní obrazovka Vimu

Pokud vidíte Vim prvně, tak Vám tato obrazovka asi toho moc neřekne. Prvním problémem se kterým jsem se potýkal bylo to jak tento program regulérně ukončit, aniž bych využíval funkce kill . Ukončit Vim se dá příkazem `:q!` nebo `ZZ`, ale nepředbíhejme. Nejprve si musíme ujasnit režimy práce ve Vimu. Vim má různé pracovní režimy.

Příkazový režim slouží k zadávání příkazů, a je základním režimem. Do tohoto režimu se vždy dostanete stiskem klávesy `ESC`

Ex režim – příkazový řádek slouží také k zadávání příkazů, ale tyto příkazy jsou daleko složitější. Do tohoto režimu přepíná z příkazového režimu `:`. Tento režim je jednorázový, což znamená že po potvrzení klavesou `Enter`↩ se Vim automaticky přepne do příkazového režimu.

Insert/replace režim tento režim slouží k vkládání/přepisování znaků. Jedná se o režim ve kterém se dá normálně psát. Do tohoto režimu se dá dostat více způsoby. Nejčastěji se přepínám do tohoto režimu stiskem klavesy `Insert`, ale i další klávesy přepínají do tohoto režimu. např `a,i,I,A`

vizuální režim do tohoto režimu se dostanete stiskem klávesy `v`. Re6im se opouští vykonáním nějaké akce nebo klávesou `ESC`. Tento režim bude dále vysvětlen.

Příkazy a režim je signalizován ve stavovém řádku. Jen malý proužek ve spod okna. Několik příkladů stavoveho řádku.

Obrázek 2.2: Příkazový režim

Obrázek 2.3: Insert/replace režim

Obrázek 2.4: Příkazová řádka

Malé vysvětlení příkazové řádky. Čísla 21,1 nebo 31,1 udávají pozici kurzoru v textu. První číslo označuje řádek a to druhé číslo označuje pozici od leveho okraje. Označení začínající – patří označení režimu ve kterém se Vim nachází. Například `--INSERT--`. Pokud příkazová řádka začína : tak to znamená že Vim očekává nějaký příkaz, který musíme potvrdit klavesou `Enter` nebo ukončit tento režim klavesou `ESC`.

2.5 A můžeme se pustit do editace textů

2.5.1 Nový soubor

Asi první co mě zajímalo na Vimu, bylo to jak se vněm vytváří nový soubor. Existuje několik možností jak vytvořit nový soubor, ale asi ta nejednoduší je přímo dosadit název nového souboru jako parametry při spouštění Vimu. Ale pokud již máme spuštěný Vim, tak nový soubor vytvoříme příkazem `:new Jmeno.souboru`

2.5.2 Uložení souboru

Pokud máme vytvořený soubor, tak musíme soubor i uložit. Ktomu slouží příkaz `:w` nebo `:write`.Pokud přidáme k příkazu parametry, slouží tento příkaz jako uložit jako. Ve Vimu je i příkaz `:saveas` který editovaný soubor uloží pod jménem které je za příkazem. Pokud

editujeme soubor který je pouze ke čtení, tak Vim nám nepovolí tento soubor přepsat. My to musíme přikázat vykřičníkem za příkazem. Tedy pokud zadám `:w!` tak Vim soubor uloží ikdyž sobor byl pouze ke čtení.

2.5.3 Ukončení Vimu

Ukončení Vimu se provádí příkazem `:q`, ale pouze pokud není soubor modifikován. Jinak tato operace skončí hláškou `E37:No write since last change`. Pro ukončení modifikovaného souboru musíme soubor napřed uložit a pak Vim ukončit příkazem `:q`. To můžeme sloučit a zadat tento příkaz `:wq` nebo vim ukončit zádáním ZZ. Tento příkaz je ekvivalentní příkazu `:wq`. Tyto komplikace při ukončování mají svůj význam. Donutí nás totiž se rozhodnout co chceme se svou prací provést. Je tím odstraněna nahodilá ztráta práce kvůli ukončení bez uložení.

2.5.4 Otevření souboru

Soubor se dá otevřít stejně jako vytváření nového souboru. Vim se spouští s parametry, které jsou jménem otevíraných souborů. Ve Vi ještě existuje příkaz `:open`, ale ten bohužel není imlementován ve Vimu.

2.5.5 Opakování příkazu

Editor vim umí opakovat příkazy. Například chceme-li se posunout o 10 znaků do prava zadáme `10h`. Dokonce není problém napsat 100x Už nebudu zlobit. Stačí jen zadat `100oUž nebudu zlobit!` a stisknout `ESC`

2.5.6 Pohyb v textu

V textu se můžeme pohybovat šipkami, nebo můžeme používát klávesy `h`– vlevo, `j`–dolů, `k`–nahoru, `l`–vpravo. tyto klávesy mají především vyznam při opakování příkazů. Ve Vimu můžeme používat standartní funkční klávesy `Home`, `End`, `PageUp`, `PageDown`. Místo kláves `PageDown,PageUp` můžeme využít klávesových zkratek `Ctrl+F` a `Ctrl+B`.Pro přesun po řádcích jde také použít klávesy `+` a `-`. Klávesa plus posouvá o řádek dolů a klavesa minus o řádek nahoru. Pro skok na určitý řádek slouží klavesa `G`. Například `18G` přesune kurzor na 18 řádek. Ke skoku na začátek řádku slouží klávesa `Home` nebo `0`. (Jedná se o nulu nikoliv o) K přesunu na konec řádku slouží klávesa `End` nebo `$`.

Docela zvláštností Vimu je přesun na pozici určitého znaku v řádku. Vyhledávání znaku se provádí dopředu nebo dozadu od kurzoru (záleží na příkazu).

`f+znak` na následující výskyt znaku

`F+znak` na předchozí výskyt znaku

`t+znak` na před následující výskyt znaku

`T+znak` na před předchozí výskyt znaku

`;` Opakuje poslední hledání na řádku

`,` Opakuje hledání na řádku, ale v opačném směru

Vim umí posouvat kurzor i po slovech a dokonce i po větách. Jako konec věty bere !?.

`w` posune kuzor na začátek následujícího slova

`e` posune kurzor na konec tohoto slova

`b` na nejbližšípředchozí začátek slova

`)` přesune kurzor na začátek nejbližší následující věty

`(` přesune kurzor na začátek nejbližší předcházející věty

`}` přesune kurzor na nejbližší následující konec ostavce

`{` přesune kurzor na nejbližší předchozí konec ostavce

2.5.7 Mazání částí textu

Pokud uděláme ve Vimu chybu a jsme v vkládacím režimu můžeme použít klávesy `Backspace`. Ale když opustíme vkládací režim, tak musíme sáhnou k jiným prostředkům. Pokud chceme smazat znak pod kurzorem, tak můžeme využít klávesy `Delete` nebo `x`. Ke smazání celého řádku slouží příkaz `dd`.

2.5.8 Kopírování textu

Vim využívá vlastní registr, do kterého ukládá veškeré mazací akce. To znamená že pokud někde něco smažeme, tak ještě to není úplně ztraceno, ale Vim to pouze uložil do registru. A právě tohoto registru se využívá při kopírování textu. Příkazem `p` vkládáme text z registru za aktuální pozici kurzoru. Naopak `P` Vkládá informace z registru před aktuální pozici kurzoru. Pro kopírování do registru bez smazání slouží příkaz `yy` takže `yyp` nám zdvojí řádek.

2.5.9 Vrácení změn v dokumentech

Pro klasické undo[3] je ve Vimu příkaz `u`, který vrátí poslední změnu. Většinou, pokud není nastaveno jinak, jde vrátit více jak 100 kroků. Pokud to s undem přeženeme a potřebovali by jsme se vrátit, tak k tomu slouží klávesová zkratka `Ctrl+R`(Redo), které můžeme také používat opakovaně. Vim má ještě jednu specialitu kterou je vrácení všech změn na posledy měněném řádku. Pro tuto akci platí příkaz `U` .

[3]Undo je funkce která vrátí poslední změnu v dokumentu

2.6 Nápověda

Původně editor Vi neměl žádnou nápovědu, ale naštěstí editor Vim se snaží být uživatelsky co nejpřívětivější a disponuje docela velmi dobrou nápovědou. Do nápovědy se dostanete příkazem `:help` nebo `:h`. Nápověda rozpůlí okno a můžete se vní pohybovat podle zvyklostí Vimu. Pokud už Vás nebaví procházení nápovědy, tak stačí zadat přákaz `:q` a nápověda se zavře. Aby se nemusela vždy procházet celá nápověda když nás jen zajímá funkce příkazu pro uložení tak stačí zadat jen `:help :w`.

2.7 Vizuální režim

Zapíná se klávesou `v`. Po zapnutí a pohybu kurzorem se nám začne text označovat. S takto označeným blokem můžeme dělat tyto akce.

`d` Vymaže a uloží blok do registru

`c` Zamění blok s obsahem registru

`y` Kopíruje blok do registru.

`~` Prohodí malá písmena za velká a naopak

`u` Převede blok na malá písmena

`U` Převede blok na velká písmena

`J` Spojí blok do jednoho řádku

`<` Posune blok doleva

`>` Posune blok doprava

`:` Přejde do příkazového řádku

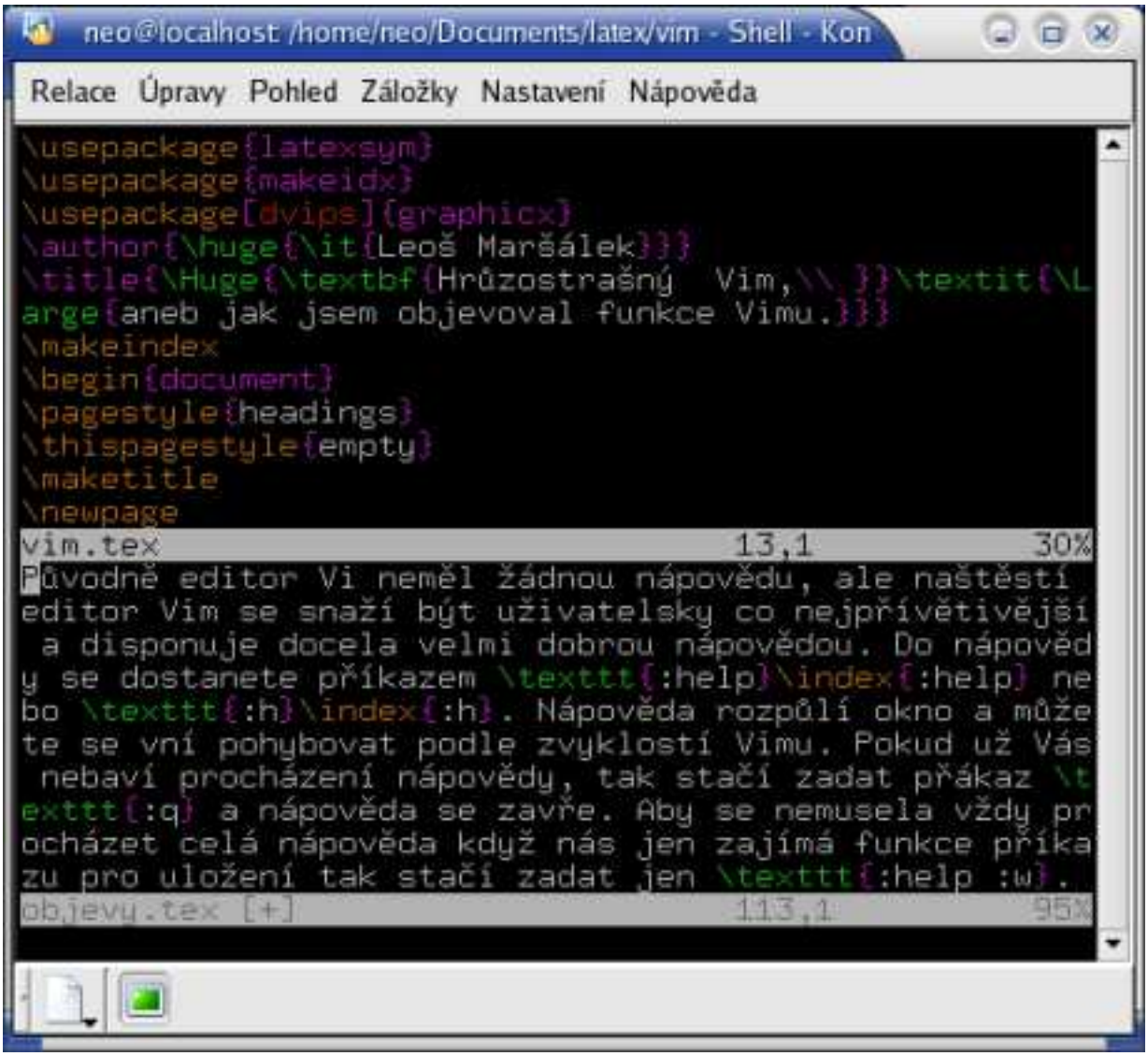
Při využití `Ctrl+v` bude označování textu čtvercové, což znamená že označování začne tam kde byl kurzor a skončí kde kurzor přesuneme.

2.8 Okna Vimu

Editor má velmidobrou vlastnost a tou je, že dokáze otevřít více editovaných souborů najednou. Dokonce umí i to že je jeden soubor otevřen vicekrát a zobrazují se různé jeho části. Jak ukazuje obrázek 2.5. Pro vytvoření nového okna je tady příkaz `:split`. Pokud je tento příkaz bez parametrů,tak vytvoří nové prázdné okno. Pokud k příkazu dáme jako parametr jméno souboru, Vim vytvoří nové okno a do něj otevře soubor který byl zadán jako parametr. Pokud zadaný soubor neexistuje Vim jej vytvoří. Pro přepínání mezi okny

slouží klávesová zkratka `Ctrl+w`. Tady je dobré vždy podívat v jakém režimu se vim zrovna nachází, protože ve vkládacím režimu tento příkaz maže slovo před kurzorem.

Pro okno plati příkazy jako pro normální nedělený Vim. Například uložení okna je taky `:w`. Jen je nutné se při rozdělených oknech dívat kde zrovna se nachází kurzor, jinak si třeba zavřeme soubor který chceme editovat.



Obrázek 2.5: Rozdělení oken

Kapitola 3

Závěrem

Tento dokument si neklade za cíl naučit Vim, ale jen by měl poskytnou záchrané kolo pro ty, kteří ve Vimu tápají stejně jako kdysi já. Je zde nastíněno jen to základní co je potřeba znát k trochu smysluplné práci s Vimem. Úplně zde chybí popis maker, které velmi dokáží ulehčit psaní dokumentů a další věci. Pokud Vás Vim chytl a myslíte si že má pro Vás smysl jej využívat, tak se určitě podívejte na *Vim skutečný editor textů*[3]. Závěrem bych Vám popřál rychlé učení Vimu, pokud jste se pro něj rozhodli. Mě osobně přineslo naučení Vimu rychlejší editaci textu, skriptů a dnes jej používám i jako editor pro programovaci jazyky, protože má v sobě zabudované zvýrazňování syntaxe jazyka a dokáže velmi pěkně zvýraznit téměř všechny programovací jazyky.

Mým původním záměrem byl trochu obsáhlejší dokument, ale bohužel z nedostatku času jsem jej musel zkrátit. Takže až se někdy budu nudit, tak jej určitě rozšířím :-).

Tento dokument byl vytvořen pomocí LaTeX 2ε

Příkaz	**Popis**
:h	otevře nápovědu
:help	otevře nápovědu
:new	vytvoří nové prázné okno
:split	rozdělí vim na více
:w	uloží obsah prace na
:wq	uloží a ukončí Vim
:q	ukončí Vim pokud je soubor nezměněn
;	opakuje hledání znaku na řádku
,	opakuje hledání znaku na řádku ale v opačném směru
+	posouvá kurzor o jeden řádek dolů
-	posouvá kurzor o jeden řádek nahoru
$	přesouvá kurzor na konec řádku

Příkaz	**Popis**
0	přesouvá kurzor na začátek řádku
)	přesune kurzor na začátek nejbližší následující věty
(	přesune kurzor na začátek nejbližší předcházející věty
}	přesune kurzor na nejbližší následující konec odstavce
{	přesune kurzor na nejbližší předchozí konec odstavce
~	prohodí malá písmena za velká a naopak u bloku
>	posune blok doprava
<	posune blok doleva
c	zamění blok s registrem
d	smaže blok
dd	smaže celý řádek
f+znak	postaví kurzor na následující výskyt znaku
F+znak	postaví kurzor na předchozí výskyt znaku
h	posune kurzor vlevo
i	zapína režim vkládání
j	posouvá kurzor o řádek dolů
J	spoji blok do jednoho řádku
k	posouvá kurzor o řádek nahoru
l	posouvá kurzor vpravo
p	vkládá informace z registru z aktuální pozici
P	vkládá informace z registru před aktuální pozici
t+znak	postaví kurzor před následující výskyt znaku
T+znak	postaví kurzor před předchozí výskyt znaku
u	undo – vrací o změny v souboru
U	vrací všechny změny na posledním měněném řádku
v	zapíná vizualní režim
x	smaže znak pod kurzorem
yy	kopíruje informace do registru
ZZ	uloží a ukončí Vim

Literatura

[1] Michal Kočer, Pavel Sýkora : **Neříliš stručný úvod do systému LaTeX 2ε** *Neboli LaTeX 2ε v 73 minutách*

[2] Radek Bednář : LaTeX

[3] Pavel Satrapa : **Vim** *Skutečný editor textů*

Index