Unixové systémy a Linux

Jan Outrata

KATEDRA INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

úvod

Literatura, dostupné zdroje

Literatura

- Vychodil V.: *Linux: Příručka českého uživatele*. Computer Press, 2003.
- Ray D. S., Ray Eric J.: *Unix: podrobný průvodce*. Grada, 2009.
- Cannon J.: *Linux for Beginners: An Introduction to the Linux Operating System and Command Line*. CreateSpace Independent Publishing, 2014.
- Kameník P.: *Příkazový řádek v Linuxu: praktická řešení*. Computer Press, 2012.
- Fox R.: *Linux with operating system concepts*. CRC Press, 2015.
- Herold H.: *awk & sed: Příručka pro dávkové zpracování textu*. Computer Press, 2004.
- kolektiv: *Linux: Dokumentační projekt*, 4. aktualizované vydání. Computer Press, 2008.

Literatura, dostupné zdroje

Literatura

- Vychodil V.: *Linux: Příručka českého uživatele*. Computer Press, 2003.
- Ray D. S., Ray Eric J.: *Unix: podrobný průvodce*. Grada, 2009.
- Cannon J.: *Linux for Beginners: An Introduction to the Linux Operating System and Command Line*. CreateSpace Independent Publishing, 2014.
- Kameník P.: *Příkazový řádek v Linuxu: praktická řešení*. Computer Press, 2012.
- Fox R.: *Linux with operating system concepts*. CRC Press, 2015.
- Herold H.: *awk & sed: Příručka pro dávkové zpracování textu*. Computer Press, 2004.
- kolektiv: *Linux: Dokumentační projekt*, 4. aktualizované vydání. Computer Press, 2008.

Další zdroje informací

- **dokumentace, která je (standardní) součástí unixových OS a distribucí Linuxu**
- články, magazíny, fóra, diskuze apod. na webu

Úvod

Unixové operační systémy (Unix-like, UN*X, *nix)

= koncepčně a filozoficky vycházející z operačního systému Unix

Úvod

Unixové operační systémy (Unix-like, UN*X, *nix)

= koncepčně a filozoficky vycházející z operačního systému Unix

- rozšířené

Úvod

Unixové operační systémy (Unix-like, UN*X, *nix)

= koncepčně a filozoficky vycházející z operačního systému Unix

- rozšířené

  Za posledních 15 let se stala zajímavá věc. Když vytáhnete telefon a podíváte se na webovou stránku, na stroji, který vám danou stránku poslal, pravděpodobně běží Linux. Z 500 nejrychlejších superpočítačů na světě 497 používá Linux. A Linux je srdcem Androidu ve více než miliardě telefonů. Místo abychom vyhráli na desktopech, vyhráli jsme všude jinde.

  – Rusty Russel, vývojář linuxového jádra

Úvod

Unixové operační systémy (Unix-like, UN*X, *nix)

= koncepčně a filozoficky vycházející z operačního systému Unix

- rozšířené
- podporované

Úvod

Unixové operační systémy (Unix-like, UN*X, *nix)

= koncepčně a filozoficky vycházející z operačního systému Unix

- rozšířené
- podporované
- populární

Úvod

Unixové operační systémy (Unix-like, UN*X, *nix)

= koncepčně a filozoficky vycházející z operačního systému Unix

- rozšířené
- podporované
- populární
- charakteristický („textový terminál“??)

Úvod

Unixové operační systémy (Unix-like, UN*X, *nix)

= koncepčně a filozoficky vycházející z operačního systému Unix

- rozšířené
- podporované
- populární
- charakteristický („textový terminál“??)

Poznání, porozumění, osvojení

- netriviální!

Úvod

Unixové operační systémy (Unix-like, UN*X, *nix)

= koncepčně a filozoficky vycházející z operačního systému Unix

- rozšířené
- podporované
- populární
- charakteristický („textový terminál“??)

Poznání, porozumění, osvojení

- netriviální!
- využití potenciálu systému a počítače naplno, nejen jako (omezené) uživatelské rozhraní pro (neefektivní) použití několika (grafických) aplikací

Úvod

Unixové operační systémy (Unix-like, UN*X, *nix)

= koncepčně a filozoficky vycházející z operačního systému Unix

- rozšířené
- podporované
- populární
- charakteristický („textový terminál“??)

Poznání, porozumění, osvojení

- netriviální!
- využití potenciálu systému a počítače naplno, nejen jako (omezené) uživatelské rozhraní pro (neefektivní) použití několika (grafických) aplikací

→ příkazový interpret (shell) – tradičně textový, viz dále

Úvod

Unixové operační systémy (Unix-like, UN*X, *nix)

= koncepčně a filozoficky vycházející z operačního systému Unix

- rozšířené
- podporované
- populární
- charakteristický („textový terminál“??)

Poznání, porozumění, osvojení

- netriviální!
- využití potenciálu systému a počítače naplno, nejen jako (omezené) uživatelské rozhraní pro (neefektivní) použití několika (grafických) aplikací

→ příkazový interpret (shell) – tradičně textový, viz dále

= používání, zkoušení, experimentování

Operační systém (OS)

„Operační systém je základní programové vybavení počítače,
které se stará o správu systémových zdrojů.“
– klasická charakterizace operačního systému

Operační systém (OS)

„Operační systém je základní programové vybavení počítače,
které se stará o správu systémových zdrojů.“
– klasická charakterizace operačního systému

Charakteristika

- **správa systémových zdrojů (= hardware)** a **úloh (programů)**
- **systémové programové rozhraní (API)** – pro programy
- **uživatelské rozhraní (UI)** – pro člověka

Operační systém (OS)

„Operační systém je základní programové vybavení počítače, které se stará o správu systémových zdrojů.“
– klasická charakterizace operačního systému

Charakteristika

- **správa systémových zdrojů (= hardware)** a **úloh (programů)**
- **systémové programové rozhraní (API)** – pro programy
- **uživatelské rozhraní (UI)** – pro člověka

Historie OS

- první počítače bez OS
- vedle hlavní úlohy doplňková
- **víceúlohové** (multitask, timesharing) a **víceuživatelské** (multiuser)

Operační systém (OS)

„Operační systém je základní programové vybavení počítače, které se stará o správu systémových zdrojů.“
– klasická charakterizace operačního systému

Charakteristika

- **správa systémových zdrojů (= hardware)** a **úloh (programů)**
- **systémové programové rozhraní (API)** – pro programy
- **uživatelské rozhraní (UI)** – pro člověka

Historie OS

- první počítače bez OS
- vedle hlavní úlohy doplňková
- **víceúlohové** (multitask, timesharing) a **víceuživatelské** (multiuser)

Základní části

- **jádro (kernel)** – základ: správy zdrojů (hardware) a úloh (procesů), příp. tzv. ovladače hardware, systémové API pro programy
- **základní programy**: práce s hardware, daty a běžícími programy
- **uživatelské rozhraní**: základní ovládání počítače

Unix

Historie

- 1965 – Multics, Bellovy telefonní laboratoře (AT&T)
- konec 60. let – Unix („Unics“), K. Thompson, D. Ritchie, B. Kernighan,
- 1973 – Unix v programovacím jazyce C
- konec 70. let – výzkumný (Bell: **System V** Release 4) a univerzitní (UCB: **BSD**) vývoj
- 80. léta – standardizace, norma IEEE **POSIX** (systémové rozhraní, přenositelnost)
- konec 80. let až nyní – komerční (System V) i komunitní (BSD) vývoj

Unix

Historie

- 1965 – Multics, Bellovy telefonní laboratoře (AT&T)
- konec 60. let – Unix („Unics“), K. Thompson, D. Ritchie, B. Kernighan,
- 1973 – Unix v programovacím jazyce C
- konec 70. let – výzkumný (Bell: **System V** Release 4) a univerzitní (UCB: **BSD**) vývoj
- 80. léta – standardizace, norma IEEE **POSIX** (systémové rozhraní, přenositelnost)
- konec 80. let až nyní – komerční (System V) i komunitní (BSD) vývoj

Charakteristika

- víceúlohový a víceuživatelský, terminálová interakce
- univerzální (rozšířenost), inspirující (systém souborů, systémové rozhraní, implementace), nadčasově funkční (více než 50-letá koncepce)

Unix

Historie

- 1965 – Multics, Bellovy telefonní laboratoře (AT&T)
- konec 60. let – Unix („Unics“), K. Thompson, D. Ritchie, B. Kernighan,
- 1973 – Unix v programovacím jazyce C
- konec 70. let – výzkumný (Bell: **System V** Release 4) a univerzitní (UCB: **BSD**) vývoj
- 80. léta – standardizace, norma IEEE **POSIX** (systémové rozhraní, přenositelnost)
- konec 80. let až nyní – komerční (System V) i komunitní (BSD) vývoj

Charakteristika

- víceúlohový a víceuživatelský, terminálová interakce
- univerzální (rozšířenost), inspirující (systém souborů, systémové rozhraní, implementace), nadčasově funkční (více než 50-letá koncepce)

Architektura

- **jádro (kernel)**
- **příkazový interpret (shell)**: (textový) příkazový režim v terminálu, spouštění programů, programovací jazyk $\rightsquigarrow$ automatizace, např. GNU Bash
- **(základní) programy** – tzv. „tiché chování“, i uživatelská rozhraní

Projekt GNU (GNU's Not Unix)

Historie

- 1983 – projekt FSF (Nadace pro svobodný software), R. M. Stallman
- cíl vytvořit volně použitelný a šiřitelný („svobodný“) unixový OS – nový trend ve vývoji OS a software obecně

Projekt GNU (GNU's Not Unix)

Historie

- 1983 – projekt FSF (Nadace pro svobodný software), R. M. Stallman
- cíl vytvořit volně použitelný a šiřitelný („svobodný“) unixový OS – nový trend ve vývoji OS a software obecně

Výstup

- programové vybavení: shell (Bash), základní i další programy (zejm. pro zpracování textu a tvorbu programů, např. editor Emacs a překladače GCC)
- jádro: Hurd $\rightarrow$ OS GNU
- **obecná veřejná licence GPL**, zaručuje práva software:
  1. používat
  2. kopírovat a sdílet
  3. upravovat – implikuje dostupnost tzv. zdrojových kódů
  4. používat, kopírovat a sdílet upravený

  ! za podmínky zachování licence GPL („virovost“)

$\Rightarrow$ **svobodný (free) software** = volně použitelný a šiřitelný, licencovaný např. GPL – tedy dostupný včetně zdrojových kódů (**otevřený, open source**)

Linux

Historie

- 1993 – konec vývoje BSD Unixu, poslední 386BSD (uzavřené), poté FreeBSD (přenositelnost), NetBSD (sítě), OpenBSD (bezpečnost) – otevřené, nový kód
- 1991 – nové jádro (unixového) OS, Linus Torvalds, žádný kód z Unixu

Linux

Historie

- 1993 – konec vývoje BSD Unixu, poslední 386BSD (uzavřené), poté FreeBSD (přenositelnost), NetBSD (sítě), OpenBSD (bezpečnost) – otevřené, nový kód
- 1991 – nové jádro (unixového) OS, Linus Torvalds, žádný kód z Unixu

Charakteristiky

- **unixový**, ale Linux != Unix

Linux

Historie

- 1993 – konec vývoje BSD Unixu, poslední 386BSD (uzavřené), poté FreeBSD (přenositelnost), NetBSD (sítě), OpenBSD (bezpečnost) – otevřené, nový kód
- 1991 – nové jádro (unixového) OS, Linus Torvalds, žádný kód z Unixu

Charakteristiky

- **unixový**, ale Linux != Unix
- nejrozšířenější (škálovatelnost), nejpodporovanější (aktuálnost), nejpopulárnější (univerzálnost), . . . , „in“ (?)

Linux

Historie

- 1993 – konec vývoje BSD Unixu, poslední 386BSD (uzavřené), poté FreeBSD (přenositelnost), NetBSD (sítě), OpenBSD (bezpečnost) – otevřené, nový kód
- 1991 – nové jádro (unixového) OS, Linus Torvalds, žádný kód z Unixu

Charakteristiky

- **unixový**, ale Linux != Unix
- nejrozšířenější (škálovatelnost), nejpodporovanější (aktuálnost), nejpopulárnější (univerzálnost), . . . , „in“ (?)
- svobodný software (licence GPL), komunitní vývoj (podporovaný firmami)

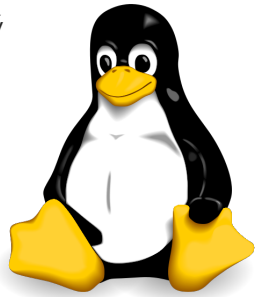
Linux

Historie

- 1993 – konec vývoje BSD Unixu, poslední 386BSD (uzavřené), poté FreeBSD (přenositelnost), NetBSD (sítě), OpenBSD (bezpečnost) – otevřené, nový kód
- 1991 – nové jádro (unixového) OS, Linus Torvalds, žádný kód z Unixu

Charakteristiky

- **unixový**, ale Linux != Unix
- nejrozšířenější (škálovatelnost), nejpodporovanější (aktuálnost), nejpopulárnější (univerzálnost), . . . , „in“ (?)
- svobodný software (licence GPL), komunitní vývoj (podporovaný firmami)
- **Tux**



GNU/Linux

Linux vs. GNU/Linux

- otázka terminologie?
- Linux = jádro OS
- GNU = shell a (základní) programy OS z projektu GNU
- jádro + shell + programy = kompletní OS $\Rightarrow$ GNU/Linux – „složité“ a také mnoho programů ne z GNU
- Linux – jednoduché, ale nepřesné označení celého OS $\rightarrow$ **linux**?

GNU/Linux

Linux vs. GNU/Linux

- otázka terminologie?
- Linux = jádro OS
- GNU = shell a (základní) programy OS z projektu GNU
- jádro + shell + programy = kompletní OS ⇒ GNU/Linux – „složité“ a také mnoho programů ne z GNU
- Linux – jednoduché, ale nepřesné označení celého OS → **linux**?

Distribuce

= „balení“ systému pro snadnou instalaci a správu OS a programů

\+ další programy: administrátorské, specializované, ...

- programy (i jádro) ve formě **balíčků**, závislosti mezi balíčky
- i tzv. „živé“ (live, např. Slax) – pro provoz netřeba instalovat
- např. Ubuntu, Mint, Mandriva, openSUSE, Fedora, **Debian**, Arch, Slackware, Gentoo, ... (i komerční, např. Red Hat Enterprise, SUSE Enterprise)

Základy unixových OS (Linuxu)

- uživatelské – správce systému musí být v prvé řadě dobrý uživatel!
- obecné principy unixových systémů a základy uživatelského stylu práce – místy zaměření se na Linux, nezávisle na distribuci

Základy unixových OS (Linuxu)

- uživatelské – správce systému musí být v prvé řadě dobrý uživatel!
- obecné principy unixových systémů a základy uživatelského stylu práce – místy zaměření se na Linux, nezávisle na distribuci

1. grafické uživatelské prostředí = desktop, základní aplikace
2. příkazový řádek terminálu – shell, příkazy, nápověda
3. systém souborů – soubory a adresáře, programy, přístupová práva, adresářová struktura
4. systém procesů, základy shellu (GNU Bash) – procesy a úlohy, vstup/výstup programu, roury
5. editace a zpracování textu – základní programy a editory

Grafické uživatelské prostředí

= grafická konzole terminálu

- ovládání klávesnicí a myší, popř. dotykovou obrazovkou

Grafické uživatelské prostředí

= grafická konzole terminálu

- ovládání klávesnicí a myší, popř. dotykovou obrazovkou

Přihlášení

- uživatelské jméno (login) + heslo
- místo znaků hesla puntíky (nebo nic)
- po přihlášení: spuštění grafického desktopového prostředí (desktopu) – uživatelské sezení (user session)

Grafické uživatelské prostředí

= grafická konzole terminálu

- ovládání klávesnicí a myší, popř. dotykovou obrazovkou

Přihlášení

- uživatelské jméno (login) + heslo
- místo znaků hesla puntíky (nebo nic)
- po přihlášení: spuštění grafického desktopového prostředí (desktopu) – uživatelské sezení (user session)

Desktop

- menu, okna, plochy, ikony, ... „klasika“ (na PC)
- základní aplikace: prohlížeč souborů, webový prohlížeč, textový editor, **emulátor (textového) terminálu**
- nápověda
- nastavení prostředí a systému (správa počítače)
- ...

Textové uživatelské prostředí

= textová konzole terminálu nebo emulace terminálu (i v grafice)

- původní uživatelské rozhraní unixových OS
- ovládání klávesnicí, myší doplňkově

Textové uživatelské prostředí

= textová konzole terminálu nebo emulace terminálu (i v grafice)

- původní uživatelské rozhraní unixových OS
- ovládání klávesnicí, myší doplňkově

Přihlášení

- pouze (celoobrazovková) konzole terminálu nebo vzdáleně, např. síťová služba SSH: příkaz `ssh login@počítač`
- uživatelské jméno (login, za výzvu login:) + heslo (za výzvu password:)
- znaky hesla se nevypisují
- po přihlášení: výpis informací a spuštění textového příkazového interpretu (shellu) – uživatelské sezení (user session)

Textové uživatelské prostředí

= textová konzole terminálu nebo emulace terminálu (i v grafice)

- původní uživatelské rozhraní unixových OS
- ovládání klávesnicí, myší doplňkově

Přihlášení

- pouze (celoobrazovková) konzole terminálu nebo vzdáleně, např. síťová služba SSH: příkaz `ssh login@počítač`
- uživatelské jméno (login, za výzvu login:) + heslo (za výzvu password:)
- znaky hesla se nevypisují
- po přihlášení: výpis informací a spuštění textového příkazového interpretu (shellu) – uživatelské sezení (user session)

Uživatelé

- běžní, systémoví, root = správce
- uživatelský účet: jméno (login), (zašifrované) heslo, primární skupina, plné jméno, domovský adresář (umístění na disku), shell po přihlášení (umístění na disku)

Příkazový řádek

= textové uživatelské rozhraní příkazového interpretu (shellu) – typicky GNU Bash

- příkazy shellu za výzvu = prompt: typicky `login@počítač:adresář$`
- pohyb kurzorem po textu: ←→, C/M-←, C/M-→, Home = C-a, End = C-e a další
- editace textu: Del = C-d, C-Del = M-d, Backspace, C/M-t, C-u, C-k, C-y, označení textu myší a stisk kolečka, S-C-c a S-C-v a další
- zadání příkazu: text + Enter – vykonání
- rolování terminálu: S-PgUp, S-PgDown, kolečko myši, C-l
- historie příkazů – klávesy ↑↓, M-<, M->, C-r + text, C-g„ příkaz `history`
- ukončení (shellu): příkazy `logout` (odhlášení), `exit`, klávesa C-d na prázdném řádku

Příkazový řádek

= textové uživatelské rozhraní příkazového interpretu (shellu) – typicky GNU Bash

- příkazy shellu za výzvu = prompt: typicky `login@počítač:adresář$`
- pohyb kurzorem po textu: ←→, C/M-←, C/M-→, Home = C-a, End = C-e a další
- editace textu: Del = C-d, C-Del = M-d, Backspace, C/M-t, C-u, C-k, C-y, označení textu myší a stisk kolečka, S-C-c a S-C-v a další
- zadání příkazu: text + Enter – vykonání
- rolování terminálu: S-PgUp, S-PgDown, kolečko myši, C-l
- historie příkazů – klávesy ↑↓, M-<, M->, C-r + text, C-g„ příkaz `history`
- ukončení (shellu): příkazy `logout` (odhlášení), `exit`, klávesa C-d na prázdném řádku

Příkazy a programy

- příkazy shellu vs. programy (např. `bash`)
- case-sensitivity, parametry/argumenty (přepínače, options)
- mnoho (GNU), „úvodní“: `echo -n výstup`, `date '+%x %X'`, `who`, `w`, `finger login`
- změna hesla: `passwd` (→ `yppasswd`)
- více „naráz“ – vykonány po sobě: oddělení `;`

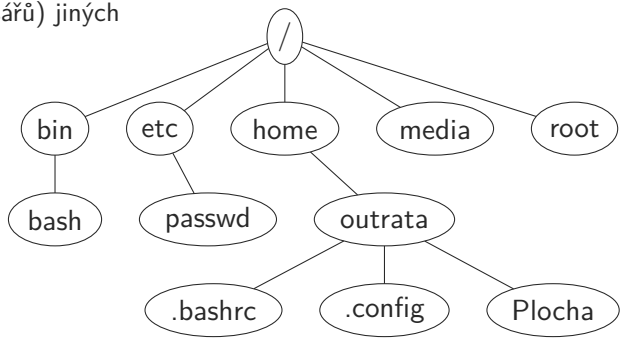
Nápověda

- standardní součást unixových systémů
- shellu (GNU Bash): `help`, `type`, parametr jméno příkazu
- **manuálové stránky**: `man program`, klávesy ↑↓, kolečko myši, PgUp, PgDown, <, >, / + text + Enter, ? + text + Enter, n, N, h, q a další, vyhledávání `whatis jméno`, `apropos slovo` (`man -k slovo`)
- GNU Info: `info`, `info program`, odkazy za *, klávesy ↑↓←→, PgUp, PgDown, Home, End, Enter (odkazy), u, C-s/r + text, C-g, H, q a další
- další dokumentace nainstalovaného software v adresáři `/usr/share/doc/` – často HTML stránky

Souborový systém

= systém organizace dat na (diskových) úložných zařízeních

- hierarchická stromová struktura (strom) **souborů a adresářů**
- podpora mnoha: **UFS** (nativní Unix), **ext2-4** (nativní Linux), XFS, JFS, ZFS, (ex)FAT, NTFS, HFS(+), Reiser4 (Linux), Btrfs (Linux), ISO-9660 (CD), UDF (DVD) a další
- abstrakce **VFS** = jediný (logický) strom souborů a adresářů
  - kořenový souborový systém pod kořenovým adresářem /
  - připojování (stromu souborů a adresářů) jiných pod nějaký adresář, např. `/mnt`



Soubory a adresáře

- **(adresářová) cesta** = posloupnost jmen, oddělených /, souborů a adresářů, ve VFS podřazených (potomků) a nadřazeného (rodiče) předchozímu, od nějakého k danému
  - absolutní: od kořenového adresáře /, např. `/home/tonda`
  - relativní: (typicky) od aktuálního adresáře `.` – (nejen) shell má vždy nějaký nastavený, např. `./tonda/soubor` (nebo i jen `tonda/soubor`)
  - nadřazený (rodičovský) adresář `..`, např. `../tonda`
- tzv. skryté (neviditelné) soubory a adresáře = jméno začíná `.` – běžně se nezobrazují v zobrazení (výpisech) jmen
- case-sensitivity, žádná omezení na jména vyjma znaku / a ne duplicitní v tomtéž adresáři (včetně `.` a `..`!)
- význam částí jmen, např. „přípona“, jen zvyklost (typ obsahu souboru)

Soubory a adresáře

- **(adresářová) cesta** = posloupnost jmen, oddělených /, souborů a adresářů, ve VFS podřazených (potomků) a nadřazeného (rodiče) předchozímu, od nějakého k danému
  - absolutní: od kořenového adresáře /, např. `/home/tonda`
  - relativní: (typicky) od aktuálního adresáře `.` – (nejen) shell má vždy nějaký nastavený, např. `./tonda/soubor` (nebo i jen `tonda/soubor`)
  - nadřazený (rodičovský) adresář `..`, např. `../tonda`
- tzv. skryté (neviditelné) soubory a adresáře = jméno začíná `.` – běžně se nezobrazují v zobrazení (výpisech) jmen
- case-sensitivity, žádná omezení na jména vyjma znaku / a ne duplicitní v tomtéž adresáři (včetně `.` a `..`!)
- význam částí jmen, např. „přípona“, jen zvyklost (typ obsahu souboru)

Příkazový řádek / shell

- **klávesa TAB**: doplnění jména příkazu/programu (příp. i parametru, GNU Bash) a souboru a adresáře!

Soubory a adresáře

- zobrazení (výpis) absolutní cesty k aktuálnímu adresáři: `pwd`
- zobrazení (výpis) jmen v adresáři: `ls` (v aktuálním), `ls cesta`, `ls -l` (podrobný), `ls -a` (i skrytých)
- změna aktuálního adresáře: `cd` (na domovský), `cd cesta`, `cd -` (předchozí v historii aktuálních)
- velikost a vytvoření adresáře: `du -sh`, `mkdir -p`
- obsah souboru: `file`, `cat`, `less` (klávesy jako `man`)
- manipulace: `cp -afirv zdroj(e) cíl`, `mv -fiv zdroj(e) cíl`, `rm -firv` – není žádná možnost obnovy (undelete)!
- hledání:
  `find cesta -name jméno -iname jméno -path cesta -size velikost -mmin mins -mtime dní -newer cesta`

Speciální soubory

Odkaz (link) na soubor nebo adresář

- pevný (hard): jen další jméno, na jiné cestě, v souborovém systému (ne VFS) pro soubor (typicky nelze pro adresář), vytvoření `ln cíl odkaz`
- symbolický (soft): „obsahem“ je cesta k souboru nebo adresáři ve VFS – operace čtení a zápisu (a změny práv) nad ním = následování odkazu, může být neplatný (broken, neexistence cesty), vytvoření `ln -s cíl odkaz`

Speciální soubory

Odkaz (link) na soubor nebo adresář

- pevný (hard): jen další jméno, na jiné cestě, v souborovém systému (ne VFS) pro soubor (typicky nelze pro adresář), vytvoření `ln cíl odkaz`
- symbolický (soft): „obsahem“ je cesta k souboru nebo adresáři ve VFS – operace čtení a zápisu (a změny práv) nad ním = následování odkazu, může být neplatný (broken, neexistence cesty), vytvoření `ln -s cíl odkaz`

Další:

- pro zařízení (hardware) nebo účel: blokové, např. pro disková úložná zařízení (např. `/dev/sda2`), a znakové, např. pro terminály (např. `/dev/tty1`) nebo např. `/dev/null`, `/dev/random` aj.
- a další

(Tradiční unixová) přístupová práva

- každému souboru/adresáři přidělen uživatel (vlastník) a skupina
- zvlášť pro vlastníka (**u**ser), skupinu (**g**roup) a ostatní (**o**ther)
- pro soubor: čtení (**r**ead), zápis (**w**rite), spuštění (e**x**ecute)
- pro adresář: čtení $\sim$ zjištění souborů/podadresářů v něm (r), zápis $\sim$ vytvoření souborů/podadresářů v něm (w), spuštění $\sim$ vstup do něj (x)
- speciální (bity): SUID (s), SGID (s), sticky (t)
- změna: `chmod -R práva cesta`, práva:
  - symbolicky: `[ugoa...]` `[+-=]` `[rwxXst...ugo]``,...`, **a**ll = všichni, např. u-x, go+rw, ug=rw, +r (= a+r), o=
  - numericky (osmičkově): až 4 cifry (zleva nuly) 0–7 jako součet hodnot pro speciální práva (SUID=4, SGID=2, sticky=1), práva pro vlastníka, skupinu a ostatní (r=4, w=2, x=1), např. 660, 755, 4550
- skupiny uživatele: `groups login`, změna vlastníka (může jen `root`) a skupiny:

  `chown -R vlastník:skupina cesta`, `chgrp -R skupina cesta`
- POSIX ACL – rozšíření na seznamy řízení přístupu se záznamy práv pro konkrétní uživatele

Adresářová struktura

Připojení a odpojení (stromu souborů a adresářů) souborového systému [linux]:

- připojení automaticky (v grafickém desktopu) nebo v (grafické) aplikaci/desktopu pod `/media/login/jmeno-disku` nebo `pmount /dev/sd??` do `/media/sd??`, výpis `sd??` pomocí `lsblk -f`
- odpojení v (grafické) aplikaci/desktopu nebo `pumount /dev/sd??`, předtím `sync`!
- zobrazení obsazeného místa: `df -h`

Adresářová struktura

Připojení a odpojení (stromu souborů a adresářů) souborového systému [linux]:

- připojení automaticky (v grafickém desktopu) nebo v (grafické) aplikaci/desktopu pod `/media/login/jmeno-disku` nebo `pmount /dev/sd??` do `/media/sd??`, výpis `sd??` pomocí `lsblk -f`
- odpojení v (grafické) aplikaci/desktopu nebo `pumount /dev/sd??`, předtím `sync`!
- zobrazení obsazeného místa: `df -h`

- ve VFS, tradiční unixová (FHS standard), linuxová (LSB standard)
- `/bin/`, `/boot/`, `/dev/`, `/etc/`, `/home/`, `/lib/`, `/media/` [linux], `/mnt/`, `/opt/`, `/proc/`, `/root/`, `/run/` [linux], `/sbin/`, `/srv/`, `/sys/` [linux], `/tmp/`, `/usr/`, `/var/`
- zajímavé vybrané soubory a adresáře: `/boot/vmlinuz` [linux], `/boot/initrd.img` [linux], `/dev/mem`, `/etc/fstab`, `/etc/passwd`, `/etc/shadow`, `/etc/hostname`, `/etc/init.d/` (`/etc/systemd/` [linux]), `/lib/modules/` [linux], `/proc/cpuinfo`, `/proc/<číslo>/`, `/usr/share/man/`, `/usr/share/doc/`, `/var/log/`, `/var/spool/mail/`, ...

Systém procesů

- program = spustitelný soubor (právo spuštění, x)
- **proces** = spuštěný program, multitasking (timesharing), process ID (PID)
- jediná stromová hierarchie (strom) procesů
  - kořenový proces (programu) `init`, PID 1 – spouští systémové programy a služby, včetně přihlašovacího dialogu
  - z něj shell/desktop a v něm uživatelské programy/aplikace
  - všechny (vyjma init) spouštěny z nějakého procesu jako **potomci rodiče**

Systém procesů

- program = spustitelný soubor (právo spuštění, x)
- **proces** = spuštěný program, multitasking (timesharing), process ID (PID)
- jediná stromová hierarchie (strom) procesů
  - kořenový proces (programu) `init`, PID 1 – spouští systémové programy a služby, včetně přihlašovacího dialogu
  - z něj shell/desktop a v něm uživatelské programy/aplikace
  - všechny (vyjma init) spouštěny z nějakého procesu jako **potomci rodiče**
- zobrazení (výpis) procesů: `ps aux f o U -ef -H -o -u`, `pstree -ap`, interaktivně `top` (klávesy x, P, M, <, >, f, c, u, h/?. q aj.) ⇝ `htop` (klávesy P, M, F6/., F2/S, u, F1/h, F10/q aj.)
- ukončení procesu: (násilně) `kill -(SIG)TERM -9 -l PID`, `killall -u login jméno` – zasílání tzv. signálů
  - návratová hodnota rodiči: 0 při korektním ukončení, jinak nenulové číslo („tiché chování“)
  - ukončeni i jeho potomci – proces může zařídit, aby nebyli, např. shell
- čas běhu příkazu: `time příkaz`

Základy shellu

Shell

= příkazový interpret = vykonávání (textových) příkazů, včetně spouštění programů – tzv. **úlohy (jobs)**

- např. GNU Bash, C, Korn (Ksh), Z shell (Zsh) aj.

Základy shellu

Shell

= příkazový interpret = vykonávání (textových) příkazů, včetně spouštění programů – tzv. **úlohy (jobs)**

- např. GNU Bash, C, Korn (Ksh), Z shell (Zsh) aj.

Řízení úloh

- běh tzv. „na popředí“ ($\sim$ „má terminál“) nebo tzv. „na pozadí“
- (obyčejné) spuštění programu $\rightarrow$ úloha na popředí
- spuštění programu na pozadí: `program &` – zobrazí ID úlohy a procesu (PID)
- výpis úloh: `jobs -l`
- pozastavení úlohy (na popředí): klávesa **C-z**
- přesunutí úlohy z popředí na pozadí: pozastavení + `bg úlohy` – úloha %ID
- přesunutí úlohy z pozadí na popředí: `fg úloha` – bez úlohy poslední (i bg)
- (násilné) ukončení úlohy (na popředí): klávesa **C-c**

Základy shellu

Expanze cest k souborům a adresářům

→ setřízený seznam existujících cest k souborům a adresářům na základě vzoru

- zástupné znaky pro vzor pro (vyhovující) cesty: ∼`login`, `*`, `?`, `[znaky]` – rozsah `znak-znak`, negace `!` nebo `^` za `[`
- potlačení, tzv. kvótování, např. `*`: `\*`, `'*'`, `"*"` – také mezery, `;`, `&`, `\`, `'`, `"`
- **klávesa TAB**, aktuální adresář (relativní cesty)

Základy shellu

Expanze cest k souborům a adresářům

→ setřízený seznam existujících cest k souborům a adresářům na základě vzoru

- zástupné znaky pro vzor pro (vyhovující) cesty: ∼`login`, `*`, `?`, `[znaky]` – rozsah `znak-znak`, negace `!` nebo `^` za `[`
- potlačení, tzv. kvótování, např. `*`: `\*`, `'*'`, `"*"` – také mezery, `;`, `&`, `\`, `'`, `"`
- **klávesa TAB**, aktuální adresář (relativní cesty)

Aliasy

= zástupná jména příkazů

- vytvoření a zrušení: `alias jméno='příkaz'`, `unalias jméno`

Základy shellu

Expanze cest k souborům a adresářům

→ setřízený seznam existujících cest k souborům a adresářům na základě vzoru

- zástupné znaky pro vzor pro (vyhovující) cesty: ∼`login`, `*`, `?`, `[znaky]` – rozsah `znak-znak`, negace `!` nebo `^` za `[`
- potlačení, tzv. kvótování, např. `*`: `\*`, `'*'`, `"*"` – také mezery, `;`, `&`, `\`, `'`, `"`
- **klávesa TAB**, aktuální adresář (relativní cesty)

Aliasy

= zástupná jména příkazů

- vytvoření a zrušení: `alias jméno='příkaz'`, `unalias jméno`

Skupina příkazů

- jako „celek“: `{ příkazy; }`
- např. spuštění na pozadí

Základy shellu

Vstup a výstup procesu

- každý proces má (standardní) vstup, výstup a chybový výstup – z shellu v terminálu napojené na vstup a výstup terminálu
- **filtr** = program, který (typicky neinteraktivně) transformuje data ze vstupu na výstup
- přesměrování vstupu ze souboru: `program < soubor`, ze vstupu terminálu: `≪ značka` a vstup ukončen řádkem = značka, z příkazové řádky: `≪≪ text`
- přesměrování výstupů do souboru: `program > soubor`, chybový `2>`, oba `&>` (`program > soubor 2>&1`), přidání `≫`
- **roura (pipe)** | = systémové propojení výstupu programu nalevo od | se vstupem programu napravo od |: `program1 | program2`

Základy shellu

Vstup a výstup procesu

- každý proces má (standardní) vstup, výstup a chybový výstup – z shellu v terminálu napojené na vstup a výstup terminálu
- **filtr** = program, který (typicky neinteraktivně) transformuje data ze vstupu na výstup
- přesměrování vstupu ze souboru: `program < soubor`, ze vstupu terminálu: `<< značka` a vstup ukončen řádkem = značka, z příkazové řádky: `<<< text`
- přesměrování výstupů do souboru: `program > soubor`, chybový `2>`, oba `&>` (`program > soubor 2>&1`), přidání `>>`
- **roura (pipe)** | = systémové propojení výstupu programu nalevo od | se vstupem programu napravo od |: `program1 | program2`
- pojmenovaná roura (fifo) = speciální soubor reprezentující rouru, vytvoření `mkfifo roura`, použití pomocí přesměrování z/do souboru

Základy shellu

Programovací jazyk (GNU/Bash) – náhled

- příkazy + spouštění programů
- proměnné: automatické, vnitřní, např. PATH (cesty pro hledání programu při spouštění, standardně ne aktuální adresář!, cesta k programu: `which`), PWD, USER, HOME, uživatelské, speciální, např. ?, !, přiřazení (textové) hodnoty `jmého=hodnota`, získání hodnoty `$jméno` nebo `${jméno}`, pole `jméno[index]`, `export jméno=hodnota`
- podmínky: `if`, `test` (`[`) nebo `[[]]`, `-dfrwxL soubor`, `=`, `!=`, `<`, `>`, `!`, `&&`, || aj., `case`
- cykly: `for`, `while`, `until`, `break`, `continue`
- celočíselná aritmetika: `(())` nebo `let ""` (hodnota `$(())`), `*`, `/`, `%`, `+`, `-`, `=`, `!=` aj.
- funkce: `jméno () skupina příkazů`, parametry `$číslo`, volání `jméno argumenty`
- příkazy v „podshellu“: `()` (výstup jako hodnota `$()` nebo `` ` ` ``)
- a mnohé další (za # do konce řádku komentář)
- **skript** = program v (prog. jazyku) shellu
  - lze spouštět jako každý jiný program (s právem spouštění)
  - první řádek: `#!/cesta/k/shellu argumenty`, např. `#!/bin/bash`
  - např. konfigurační skripty shellu: `.bashrc`, `.bash_profile`, `.profile` aj. v ∼

příkazy a (jednoduché) programy – filtry, vstup/výstup

+

roura

+

programovací jazyk

=

„cokoliv“!

Text

- data v unixových systémech tradičně textová
- **(plain) text** = posloupnost řádků = posloupnost tisknutelných a řídících znaků (v textovém terminálu), zakončená znakem pro nový řádek (\n)
- dnes standardně kódování **UTF-8**, konverze mezi kódováními: `enca+enconv -l charsets -x do`, `recode -l z..do`, `iconv -l -f z -t do` (ne konce řádků), `cstocs z do` (ne konce řádků), konverze konců řádků: `dos/mac2unix`, `unix2dos/mac`
- **textový editor** = editor plain textu → uživatelský návyk

Text

- data v unixových systémech tradičně textová
- **(plain) text** = posloupnost řádků = posloupnost tisknutelných a řídících znaků (v textovém terminálu), zakončená znakem pro nový řádek (\n)
- dnes standardně kódování **UTF-8**, konverze mezi kódováními: `enca+enconv -l charsets -x do`, `recode -l z..do`, `iconv -l -f z -t do` (ne konce řádků), `cstocs z do` (ne konce řádků), konverze konců řádků: `dos/mac2unix`, `unix2dos/mac`
- **textový editor** = editor plain textu → uživatelský návyk

Příkazový řádek / shell

- příkaz v editoru: klávesa C-x C-e, z historie `fc -ls prefix`

Textové editory

(G)Vim (Vi)

- `vi(m) -g` (`gvim`), `vimtutor -g`
- módy (režimy) normální, vkládací, vizuální, info C-g, příkazy Esc, šipky (h,j,k,l), w, b, Home (^, 0), End ($), PgUp, PgDown, i/I, a/A, o/O, r/R, c(w,^,$,c), x/X, u, C-r, y(w,^,$,y), d(w,^,$,d), p, v/V/C-v, /, ?, n, N, :(%)s(/co/za/)(g,c), :e! soubor, :w(a) soubor, :r soubor, :bp, :bn, C-^, :bd, :(v)split, C-w šipky, :close, :only, :h/F1, :(w)q! aj.

Textové editory

(G)Vim (Vi)

- `vi(m) -g` (`gvim`), `vimtutor -g`
- módy (režimy) normální, vkládací, vizuální, info C-g, příkazy Esc, šipky (h,j,k,l), w, b, Home (^, 0), End ($), PgUp, PgDown, i/I, a/A, o/O, r/R, c(w,^,$,c), x/X, u, C-r, y(w,^,$,y), d(w,^,$,d), p, v/V/C-v, /, ?, n, N, :(%)s(/co/za/)(g,c), :e! soubor, :w(a) soubor, :r soubor, :bp, :bn, C-^, :bd, :(v)split, C-w šipky, :close, :only, :h/F1, :(w)q! aj.

GNU Emacs

- `emacs -no-splash -nw`
- klávesy pro pohyb kurzorem a editaci jako v příkazovém řádku, ↑↓ (C-b,n,p,f, A-f,b), PgUp, PgDown, C-g, Ins, C-_ (C-/, C-x u), C-space, A-w, C-w, C-y, C-x space, C-x r A-w/k/y, C-s, C-r, A-%, C-x C-f, C-x C-s, C-x C-w, C-x s, C-x i, C-x ←, C-x →, C-x b, C-x k, C-x 2/3, C-x o, C-x 0, C-x 1, C-h/F1 ? (t), C-x C-c aj. (tetris a psychiatr ;-))

Textové editory

(G)Vim (Vi)

- `vi(m) -g` (`gvim`), `vimtutor -g`
- módy (režimy) normální, vkládací, vizuální, info C-g, příkazy Esc, šipky (h,j,k,l), w, b, Home (^, 0), End ($), PgUp, PgDown, i/I, a/A, o/O, r/R, c(w,^,$,c), x/X, u, C-r, y(w,^,$,y), d(w,^,$,d), p, v/V/C-v, /, ?, n, N, :(%)s(/co/za/)(g,c), :e! soubor, :w(a) soubor, :r soubor, :bp, :bn, C-^, :bd, :(v)split, C-w šipky, :close, :only, :h/F1, :(w)q! aj.

GNU Emacs

- `emacs -no-splash -nw`
- klávesy pro pohyb kurzorem a editaci jako v příkazovém řádku, ↑↓ (C-b,n,p,f, A-f,b), PgUp, PgDown, C-g, Ins, C-_ (C-/, C-x u), C-space, A-w, C-w, C-y, C-x space, C-x r A-w/k/y, C-s, C-r, A-%, C-x C-f, C-x C-s, C-x C-w, C-x s, C-x i, C-x ←, C-x →, C-x b, C-x k, C-x 2/3, C-x o, C-x 0, C-x 1, C-h/F1 ? (t), C-x C-c aj. (tetris a psychiatr ;-))

Další

- Jed: `(x)jed`, (výchozí) ovládání jako v GNU Emacs
- GNU Nano (Pico): `nano` (`pico`)

Zpracování (nejen) textu

GNU Textutils (Coreutils)

- text ze/do standardního vstupu/výstupu (ukončení C-d) nebo souborů

= **filtry** ⇒ přesměrování vstupu a výstupů + propojení rourou

- obsah: `cat -n`, `tac`, `rev`, `head -qcn -1`, `tail -qfcn +1`, `wc -cmlw`
- třídění řádků: `sort -fdnhruotk F.C`, `uniq -ci` (setřízené řádky!)
- obsah do/ze sloupců: `cut -bcdf N-M`, `paste -ds`
- náhrada (a odstranění) znaků: `tr -cds set1 set2` – i řídící znaky např. \n, \t, rozsah -
- rozdělení: `split -dblnt`

+ rozdíly: `cmp -bl`, `diff -quyrNi`, `patch -ioup N`

Zpracování (nejen) textu

GNU Textutils (Coreutils)

- text ze/do standardního vstupu/výstupu (ukončení C-d) nebo souborů

= **filtry** ⇒ přesměrování vstupu a výstupů + propojení rourou

- obsah: `cat -n`, `tac`, `rev`, `head -qcn -1`, `tail -qfcn +1`, `wc -cmlw`
- třídění řádků: `sort -fdnhruotk F.C`, `uniq -ci` (setřízené řádky!)
- obsah do/ze sloupců: `cut -bcdf N-M`, `paste -ds`
- náhrada (a odstranění) znaků: `tr -cds set1 set2` – i řídící znaky např. \n, \t, rozsah -
- rozdělení: `split -dblnt`

\+ rozdíly: `cmp -bl`, `diff -quyrNi`, `patch -ioup N`

- filtrace řádků dle vzoru → regulární výrazy (vzor) a `grep`

Zpracování (nejen) textu

GNU Textutils (Coreutils)

- text ze/do standardního vstupu/výstupu (ukončení C-d) nebo souborů

= **filtry** ⇒ přesměrování vstupu a výstupů + propojení rourou

- obsah: `cat -n`, `tac`, `rev`, `head -qcn -1`, `tail -qfcn +1`, `wc -cmlw`
- třídění řádků: `sort -fdnhruotk F.C`, `uniq -ci` (setřízené řádky!)
- obsah do/ze sloupců: `cut -bcdf N-M`, `paste -ds`
- náhrada (a odstranění) znaků: `tr -cds set1 set2` – i řídící znaky např. \n, \t, rozsah -
- rozdělení: `split -dblnt`

\+ rozdíly: `cmp -bl`, `diff -quyrNi`, `patch -ioup N`

- filtrace řádků dle vzoru → regulární výrazy (vzor) a `grep`
- proudová (neinteraktivní) editace → `sed`

Zpracování (nejen) textu

GNU Textutils (Coreutils)

- text ze/do standardního vstupu/výstupu (ukončení C-d) nebo souborů

= **filtry** ⇒ přesměrování vstupu a výstupů + propojení rourou

- obsah: `cat -n`, `tac`, `rev`, `head -qcn -1`, `tail -qfcn +1`, `wc -cmlw`
- třídění řádků: `sort -fdnhruotk F.C`, `uniq -ci` (setřízené řádky!)
- obsah do/ze sloupců: `cut -bcdf N-M`, `paste -ds`
- náhrada (a odstranění) znaků: `tr -cds set1 set2` – i řídící znaky např. \n, \t, rozsah -
- rozdělení: `split -dblnt`

\+ rozdíly: `cmp -bl`, `diff -quyrNi`, `patch -ioup N`

- filtrace řádků dle vzoru → regulární výrazy (vzor) a `grep`
- proudová (neinteraktivní) editace → `sed`
- „cokoliv“ (programovací jazyk) → `awk`

Regulární výrazy a grep

Regulární výraz

= vzor pro (vyhovující) textové řetězce, použití pro jejich vyhledávání

- znak ., \t, \n, [^-[:třída:]] (třídy digit, alpha, blank, space aj.), pozice ^, $, \<, \>
- opakování ?, *, +, {n}, {n,}, {n,m}, zřetězení, volba |, podvýraz () + \n
- potlačení spec. významu \znak
- rozšířené POSIX (ERE), základní (BRE) ?+{}|() s \ pro spec. význam
- zápis často mezi // → \/ pro znak /

Regulární výrazy a grep

Regulární výraz

= vzor pro (vyhovující) textové řetězce, použití pro jejich vyhledávání

- znak ., \t, \n, [^-[:třída:]] (třídy digit, alpha, blank, space aj.), pozice ^, $, \<, \>
- opakování ?, *, +, {n}, {n,}, {n,m}, zřetězení, volba |, podvýraz () + \n
- potlačení spec. významu \znak
- rozšířené POSIX (ERE), základní (BRE) ?+{}|() s \ pro spec. význam
- zápis často mezi // → \/ pro znak /

`grep 'vzor'`

= filtrace řádků textu na pouze obsahující řetězce vyhovující vzoru („vyhovující řádky“)

- vzor: řetězec s -F/`fgrep` nebo **regulární výraz** – výchozí základní (-G), rozšířené s -E/`egrep`, více -e 'vzor', v souboru -f soubor, nevyhovující řádky -v, -i
- výstup: -clLon, N řádků za/před/za i před vyhovujícím (kontext): -ABC N
- text rekurzivně ze souborů adresáře: -r/`rgrep`, -R

Proudový editor sed (základy)

```
sed 'příkazy'
```

= proudová (neinteraktivní) editace řádků textu dle zadaných příkazů (tzv. **sed skript**)

- jeden průchod textem („proudem“) – filtr, opakovaně:
  1. načtení dalšího řádku textu do vstupního bufferu (pattern space)
  2. test vykonání a vykonání editovacích příkazů (v zadaném pořadí) nad vstupním bufferem
  3. výpis vstupního bufferu (s odřádkováním) – potlačení -n

Proudový editor sed (základy)

```
sed 'příkazy'
```

= proudová (neinteraktivní) editace řádků textu dle zadaných příkazů (tzv. **sed skript**)

- jeden průchod textem („proudem“) – filtr, opakovaně:
  1. načtení dalšího řádku textu do vstupního bufferu (pattern space)
  2. test vykonání a vykonání editovacích příkazů (v zadaném pořadí) nad vstupním bufferem
  3. výpis vstupního bufferu (s odřádkováním) – potlačení -n
- příkazy: `adresa1,adresa2 funkce argumenty`, oddělené ; (nebo na samostatných řádcích), více zřetězení -e 'příkazy', v souboru -f skript.sed
  - adresy nepovinné, funkce vykonána pro řádky od vyhovujícího adrese1 po vyhovující adrese2 nebo jedné adrese nebo všechny, ! funkce pro ostatní řádky
  - adresa: číslo řádku od 1 ($ = poslední) nebo `/regulární výraz/` – výchozí základní, rozšířené s -E/-r, vyhovující řádek obsahuje řetězec vyhovující výrazu

Proudový editor sed (základy)

```
sed 'příkazy'
```

= proudová (neinteraktivní) editace řádků textu dle zadaných příkazů (tzv. **sed skript**)

- jeden průchod textem („proudem“) – filtr, opakovaně:
  1. načtení dalšího řádku textu do vstupního bufferu (pattern space)
  2. test vykonání a vykonání editovacích příkazů (v zadaném pořadí) nad vstupním bufferem
  3. výpis vstupního bufferu (s odřádkováním) – potlačení -n
- příkazy: `adresa1,adresa2 funkce argumenty`, oddělené ; (nebo na samostatných řádcích), více zřetězení -e 'příkazy', v souboru -f skript.sed
  - adresy nepovinné, funkce vykonána pro řádky od vyhovujícího adrese1 po vyhovující adrese2 nebo jedné adrese nebo všechny, ! funkce pro ostatní řádky
  - adresa: číslo řádku od 1 ($ = poslední) nebo `/regulární výraz/` – výchozí základní, rozšířené s -E/-r, vyhovující řádek obsahuje řetězec vyhovující výrazu
- vybrané funkce (max. počet adres, jméno, argumenty):
  - 2 `p`: výpis vstupního bufferu, 1 `q`: p (ne s -n) + ukončení, 2 `n`: p (ne s -n) + 1.
  - 2 `d`: výmaz vstupního bufferu a další iterace skriptu
  - 1 `a/i \text`: výpis textu po/před výpisu vstupního bufferu (s odřádkováním)
  - 1 =: výpis čísla řádku textu (s odřádkováním), 2 `{ příkazy }`

Proudový editor sed (základy)

```
sed 'příkazy'
```

- 2 **s/regulární výraz/text/**`příznaky`: nahrazení prvního řetězce obsaženého ve vstupním bufferu vyhovujícímu výrazu textem
  - \n / & v textu = řetězec vyhovující n-tému (1-9) podvýrazu / celému výrazu
  - nepovinné příznaky: n / g – nahrazen n-tý / každý (nepřekrývající se) vyhovující řetězec, p – při nahrazení výpis změněného vstupního bufferu

Proudový editor sed (základy)

```
sed 'příkazy'
```

- 2 **s/regulární výraz/text/**`příznaky`: nahrazení prvního řetězce obsaženého ve vstupním bufferu vyhovujícímu výrazu textem
  - \n / & v textu = řetězec vyhovující n-tému (1-9) podvýrazu / celému výrazu
  - nepovinné příznaky: n / g – nahrazen n-tý / každý (nepřekrývající se) vyhovující řetězec, p – při nahrazení výpis změněného vstupního bufferu
- 2 `y/řetězec1/řetězec2/`: nahrazení znaků z řetězce1 znaky v řetězci2 na stejné pozici ve vstupním bufferu

Proudový editor sed (základy)

```
sed 'příkazy'
```

- 2 **s/regulární výraz/text/**`příznaky`: nahrazení prvního řetězce obsaženého ve vstupním bufferu vyhovujícímu výrazu textem
  - \n / & v textu = řetězec vyhovující n-tému (1-9) podvýrazu / celému výrazu
  - nepovinné příznaky: n / g – nahrazen n-tý / každý (nepřekrývající se) vyhovující řetězec, p – při nahrazení výpis změněného vstupního bufferu
- 2 `y/řetězec1/řetězec2/`: nahrazení znaků z řetězce1 znaky v řetězci2 na stejné pozici ve vstupním bufferu
- další funkce: pro práci s více řádky textu ve vstupním bufferu (P, N, D), se soubory (r, w), s tzv. hold bufferem (hold space, h/H, g/G, x), (podmíněný) skok na místo ve skriptu (:, b, t) aj. – „programování“
- GNU rozšíření: další specifikace adresy (adresa,+N, adresa~N aj.) a funkce (Q, R, W, T aj.)

Programovací jazyk awk („neprogramovací“ základy)

```
awk 'program'
```

= analýza a zpracování řádků textových (typicky tabulkových) dat dle **awk programu**

- jeden průchod textem – filtr, opakovaně:
  1. načtení dalšího řádku textu jako záznamu (record) a jeho rozdělení do sloupců, polí (field) – oddělovač znak za -F, výchozí mezery nebo tabulátory
  2. test vykonání a vykonání akcí programu (v zadaném pořadí) nad záznamem

Programovací jazyk awk („neprogramovací“ základy)

```
awk 'program'
```

= analýza a zpracování řádků textových (typicky tabulkových) dat dle **awk programu**

- jeden průchod textem – filtr, opakovaně:
  1. načtení dalšího řádku textu jako záznamu (record) a jeho rozdělení do sloupců, polí (field) – oddělovač znak za -F, výchozí mezery nebo tabulátory
  2. test vykonání a vykonání akcí programu (v zadaném pořadí) nad záznamem
- program: posloupnost `vzorek { akce }`, oddělené ; (nebo na samostatných řádcích), více zřetězení -e 'program', v souboru -f program.awk, „pretty print“ -o
  - vzorek anebo akce nepovinné, akce vykonána pro záznamy vyhovující vzorku nebo všechny
  - vzorek (pattern): logický výraz – záznam vyhovuje při pravdivém, nebo `/regulární výraz/` (rozšířené) – vyhovující záznam obsahuje řetězec vyhovující výrazu
  - ! vzorek (negace), složení vzorků &&, ||, (), `vzorek1,vzorek2` – vyhovují záznamy od vyhovujícího vzorek1 po vyhovující vzorek2
  - speciální vzorky `BEGIN` a `END`: povinná akce vykonána před prvním záznamem (typicky inicializace), resp. po posledním záznamu (typicky souhrnný výpis)

Programovací jazyk awk („neprogramovací“ základy)

```
awk 'program'
```

= analýza a zpracování řádků textových (typicky tabulkových) dat dle **awk programu**

- jeden průchod textem – filtr, opakovaně:
  1. načtení dalšího řádku textu jako záznamu (record) a jeho rozdělení do sloupců, polí (field) – oddělovač znak za -F, výchozí mezery nebo tabulátory
  2. test vykonání a vykonání akcí programu (v zadaném pořadí) nad záznamem
- program: posloupnost `vzorek { akce }`, oddělené ; (nebo na samostatných řádcích), více zřetězení -e 'program', v souboru -f program.awk, „pretty print“ -o
  - vzorek anebo akce nepovinné, akce vykonána pro záznamy vyhovující vzorku nebo všechny
  - vzorek (pattern): logický výraz – záznam vyhovuje při pravdivém, nebo `/regulární výraz/` (rozšířené) – vyhovující záznam obsahuje řetězec vyhovující výrazu
  - ! vzorek (negace), složení vzorků &&, ||, (), `vzorek1,vzorek2` – vyhovují záznamy od vyhovujícího vzorek1 po vyhovující vzorek2
  - speciální vzorky `BEGIN` a `END`: povinná akce vykonána před prvním záznamem (typicky inicializace), resp. po posledním záznamu (typicky souhrnný výpis)
- akce: posloupnost příkazů oddělených ; (nebo na samostatných řádcích), blok v {}, výchozí (bez i {}) výpis celého záznamu (příkaz `print`)

Programovací jazyk awk („neprogramovací“ základy)

```
awk 'program'
```

- příkazy: `print argumenty >soubor` (s odřádkováním), `printf "formát" argumenty >soubor`, `getline proměnná <soubor` (další řádek textu), `next` (hned další záznam) aj.
- pole (záznamu): `$N` N-té (od 1), $0 celý záznam
- proměnné: automatické, přiřazení hodnoty `jméno=hodnota`, i za -v, získání hodnoty `jméno`, pole `jméno[index]`
  - datové typy: čísla celá i necelá (s .) nebo "textové řetězce"
  - vnitřní: NF = počet polí záznamu, NR = číslo záznamu (řádku, od 1), FS = znak oddělovač polí záznamu (pro výchozí mezera), FILENAME = jméno aktuálního vstupního souboru aj.
- operátory: porovnání (v log. výrazech) ==, !=, <, >, <=, >= (lexikografické), řetězec !~ reg. výraz, arimetické +, −, *, /, % aj., řetězcové spojení = mezera
- funkce: číselné (sin, cos, sqrt, log apod.), řetězcové (length, substr, sub, split apod.) aj.

Programovací jazyk awk („neprogramovací“ základy)

`awk 'program'`

- příkazy: `print argumenty >soubor` (s odřádkováním), `printf "formát" argumenty >soubor`, `getline proměnná <soubor` (další řádek textu), `next` (hned další záznam) aj.
- pole (záznamu): `$N` N-té (od 1), $0 celý záznam
- proměnné: automatické, přiřazení hodnoty `jméno=hodnota`, i za -v, získání hodnoty `jméno`, pole `jméno[index]`
  - datové typy: čísla celá i necelá (s .) nebo "textové řetězce"
  - vnitřní: NF = počet polí záznamu, NR = číslo záznamu (řádku, od 1), FS = znak oddělovač polí záznamu (pro výchozí mezera), FILENAME = jméno aktuálního vstupního souboru aj.
- operátory: porovnání (v log. výrazech) ==, !=, <, >, <=, >= (lexikografické), řetězec !∼ reg. výraz, arimetické +, −, *, /, % aj., řetězcové spojení = mezera
- funkce: číselné (sin, cos, sqrt, log apod.), řetězcové (length, substr, sub, split apod.) aj.
- podmínky: `if (log. výraz)`, `switch`, cykly: `while`, `do while`, `for`
- vlastní funkce: `function jméno(parametry)` (místo `vzorek`), volání `jméno(argumenty)`, `return`
- a mnohé další (za # do konce řádku komentář)