Skriptovanie (programovanie v shell-i)

Operačné systémy - ZS 2016/2017

michal.vrabel@tuke.sk

Spracovanie textov

Kódy, kódové stránky

- EBCDIC
  - eight-bit character encoding used mainly on IBM mainframe and IBM midrange computer operating systems.
- ASCII
  - developed from telegraph code.
  - Its first commercial use was as a **seven-bit teleprinter** code promoted by Bell data services.
- Extended ASCII
- Kód bratov Kamenických
- CP850, CP852, … - used under DOS
- Windows-1250, WIN-1252, ...
- ISO-8859-1, ISO-8859-2, ..., ISO-8859-x
- UNICODE-16
  - variable-length,
  - code points are encoded with one or two 16-bit code units
- UNICODE-32 (~120 tisíc znakov, pre 129 písem) - fixed-length encoding
- UTF-8

ASCII Table

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	`
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(	72	48	110	H	104	68	150	h
9	9	11		41	29	51	)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[	123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	|
29	1D	35		61	3D	75	=	93	5D	135	]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~
31	1F	37		63	3F	77	?	95	5F	137	_	127	7F	177	

Extended ASCII

- EASCII or high ASCII
- refers to eight-bit or larger character encodings that include the standard seven-bit ASCII characters, plus additional characters
- The meaning of each extended code point can be different in every encoding.
- In order to correctly interpret and display text data (sequences of characters) that includes extended codes, hardware and software that reads or receives the text must use the specific extended ASCII encoding that applies to it.
- Applying the wrong encoding causes irrational substitution of many or all extended characters in the text.

UTF-8

- Character encoding capable of encoding all possible characters, or code points, defined by Unicode
- Variable-length and uses 8-bit code units
- Backward compatible with ASCII-encoded text
- Example - wolf face - 🐺 - Unicode: `U+1F43A` - Bytes: `0xF09F90BA`
  - http://apps.timwhitlock.info/emoji/tables/unicode

Number of bytes	Bits for code point	First code point	Last code point	Byte 1	Byte 2	Byte 3	Byte 4
1	7	U+0000	U+007F	`0xxxxxxx`			
2	11	U+0080	U+07FF	`110xxxxx`	`10xxxxxx`		
3	16	U+0800	U+FFFF	`1110xxxx`	`10xxxxxx`	`10xxxxxx`	
4	21	U+10000	U+10FFFF	`11110xxx`	`10xxxxxx`	`10xxxxxx`	`10xxxxxx`

(https://en.wikipedia.org/wiki/UTF-8)

Kódy, kódové stránky II.

- Utilita iconv pozná 1173 druhov kódovaní

```
iconv - convert text from one character encoding to another

       iconv [options] [-f from-encoding] [-t to-encoding] [inputfile]...

The iconv program reads in text in one encoding and outputs the text
in another encoding.  If no input files are given, or if it is given
as a dash (-), iconv reads from standard input.  If no output file is
given, iconv writes to standard output.

Convert text from the ISO 8859-15 character encoding to UTF-8:

  $ iconv -f ISO-8859-15 -t UTF-8 < input.txt > output.txt
```

man 1 iconv
(http://man7.org/linux/man-pages/man1/iconv.1.html)

Triedenie textov

- Ako triediť?:
  - `if(strcmp(r1,r2)>0) vymena(r1,r2);`
- Zotrieďte:
  - luk, ľad, lad, lak, les
  - mačka, čačka
- ⇒ strcmp() nefunguje pre väčšinu lexikografických triedení (ak vôbec pre nejaké)

Triedenie – locales

- Celý rad definícií pre podporu konkrétneho jazyka (**man 7 locale**):

LC_COLLATE

This category governs the collation rules used for **sorting and regular expressions**, including character equivalence classes and multicharacter collating elements. This locale category changes the behavior of the functions strcoll(3) and strxfrm(3), which are used to compare strings in the local alphabet. For example, the German sharp s is sorted as "ss".

LC_CTYPE

This category determines the interpretation of byte sequences as characters (e.g., single versus multibyte characters), character classifications (e.g., alphabetic or digit), and the behavior of character classes. On glibc systems, this category also determines the character transliteration rules for iconv(1) and iconv(3). It changes the behavior of the character handling and classification functions, such as isupper(3) and toupper(3), and the multibyte character functions such as mblen(3) or wctomb(3).

Triedenie – locales (pokračovanie)

LC_NUMERIC

This category determines the formatting rules used for nonmonetary numeric values—for example, the thousands separator and the radix character (a period in most English-speaking countries, but a comma in many other regions). It affects functions such as printf(3), scanf(3), and strtod(3). This information can also be read with the localeconv(3) function.

LC_TIME

This category governs the formatting used for date and time values. For example, most of Europe uses a 24-hour clock versus the 12-hour clock used in the United States. The setting of this category affects the behavior of functions such as strftime(3) and strptime(3).

locale

```
$ locale
LANG=sk_SK.UTF-8
LANGUAGE=
LC_CTYPE="sk_SK.UTF-8"
LC_NUMERIC=sk_SK.UTF-8
LC_TIME=sk_SK.UTF-8
LC_COLLATE="sk_SK.UTF-8"
LC_MONETARY=sk_SK.UTF-8
LC_MESSAGES="sk_SK.UTF-8"
LC_PAPER=sk_SK.UTF-8
LC_NAME=sk_SK.UTF-8
LC_ADDRESS=sk_SK.UTF-8
LC_TELEPHONE=sk_SK.UTF-8
LC_MEASUREMENT=sk_SK.UTF-8
LC_IDENTIFICATION=sk_SK.UTF-8
LC_ALL=
```

```
$ locale date_fmt
%a %b %e %H:%M:%S %Z %Y

$ locale -ck date_fmt
LC_TIME
date_fmt="%a %b %e %H:%M:%S %Z %Y"

$ locale -k LC_TELEPHONE
tel_int_fmt="+%c %a %l"
tel_dom_fmt=""
int_select=""
int_prefix="421"
telephone-codeset="UTF-8"
```

man 1 locale
(http://man7.org/linux/man-pages/man1/locale.1.html)

Predvolené hodnoty locale parametrov

If the second argument to setlocale(3) is an empty string, "", for the default locale, it is determined using the following steps:

1. If there is a non-null environment variable **LC_ALL**, the value of **LC_ALL** is used.

2. If an environment variable with the same name as one of the categories above exists and is non-null, its value is used for that category.

3. If there is a non-null environment variable **LANG**, the value of **LANG** is used.

man 7 locale
(http://man7.org/linux/man-pages/man7/locale.7.html)

Podporované locales - localedef (1)

```
localedef - compile locale definition files

       localedef [options] outputpath
       localedef --list-archive [options]
       localedef --delete-from-archive [options] localename ...
       localedef --add-to-archive [options] compiledpath
       localedef --version
       localedef --help
       localedef --usage

The localedef program reads the indicated charmap and input files,
compiles them to a binary form quickly usable by the locale functions
in the C library (setlocale(3), localeconv(3), etc.), and places the
output in outputpath.
```

man 1 localedef
(http://man7.org/linux/man-pages/man1/localedef.1.html)

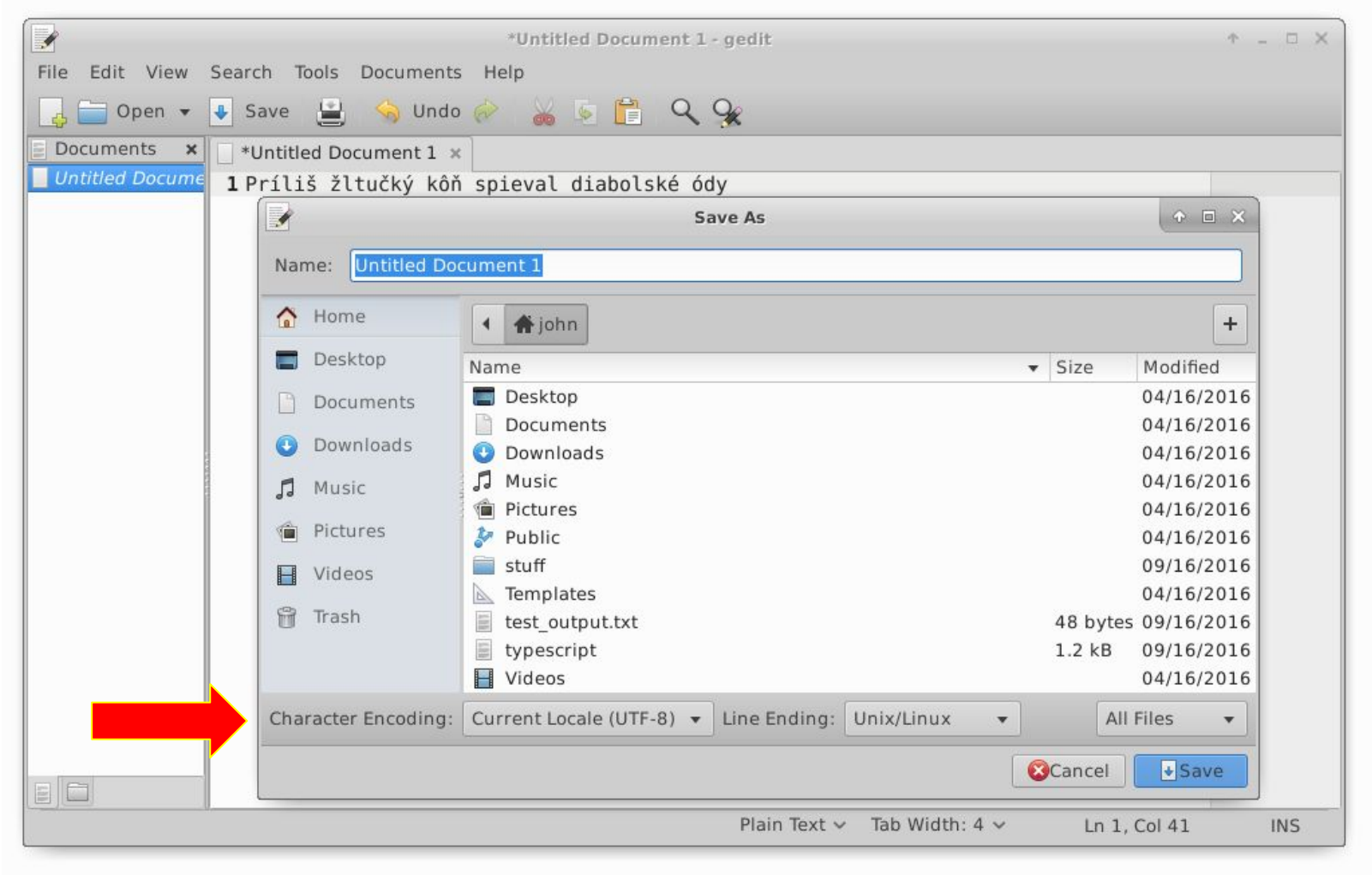
Podporované locales - localedef (2)

```
vrabelm@hron:$ localedef --list-archive
en_US
en_US.iso88591
en_US.utf8
ru_RU
ru_RU.cp1251
ru_RU.iso88595
ru_RU.koi8r
ru_RU.utf8
ru_UA
ru_UA.koi8u
ru_UA.utf8
russian
sk_SK
sk_SK.iso88592
sk_SK.utf8
slovak
```

Prečo „nefunguje“ diakritika

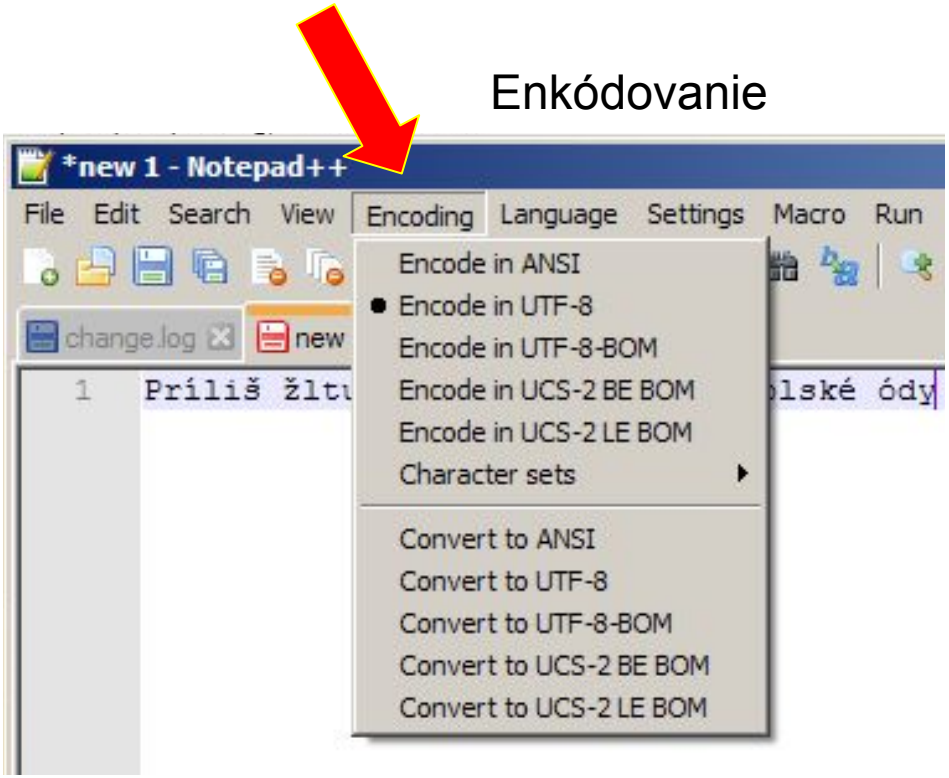
- Musí byť zosúladené:
  - Kódovanie súboru - príkazy file, hd
  - Kódovanie terminálového emulátora
  - Hodnota premennej LANG (LC_...) - `echo $LANG`

Prečo „nefunguje“ diakritika - kódovanie súboru (1)

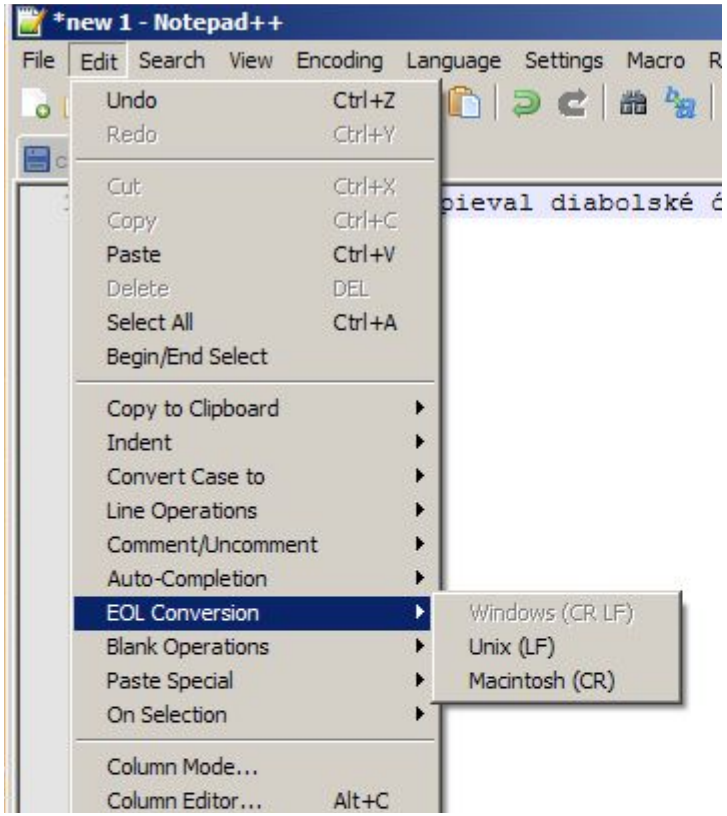


Prečo „nefunguje“ diakritika - kódovanie súboru (2)

Enkódovanie

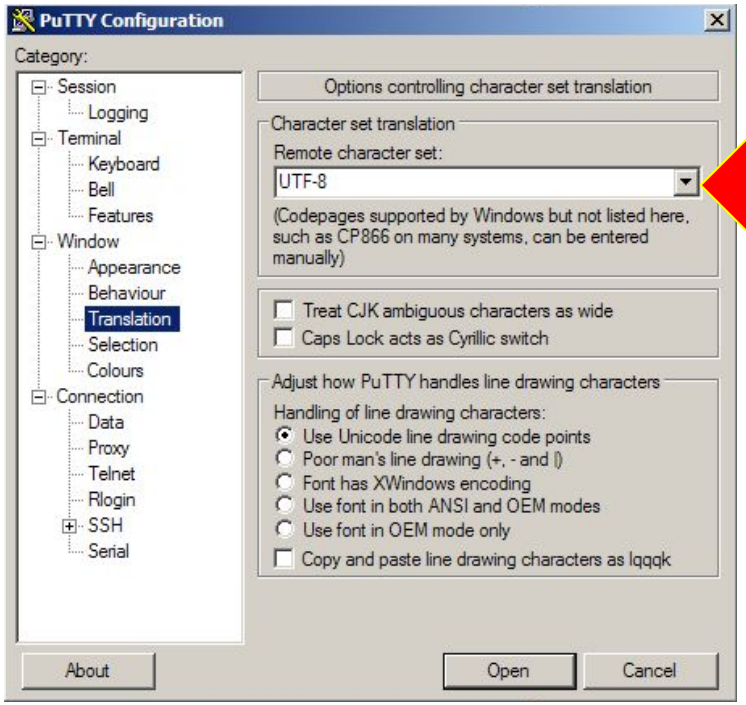


EOL - End of line

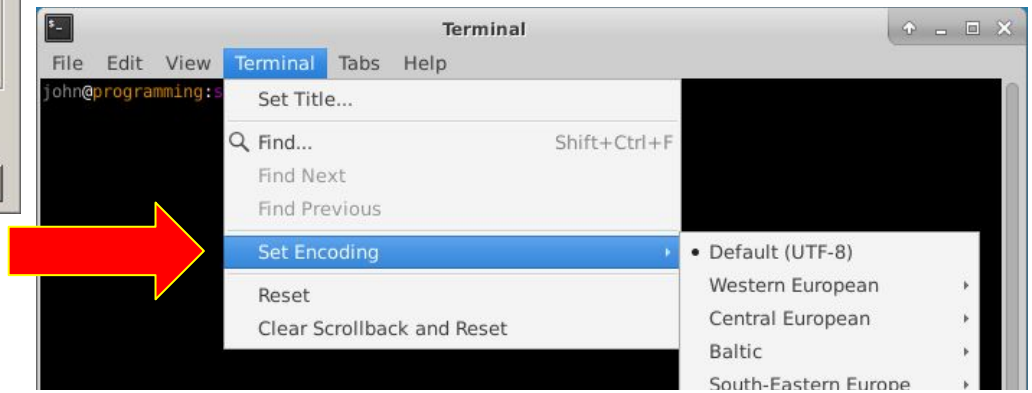


Prečo „nefunguje“ diakritika - terminálový emulátor

- Pozor na lokálne nastavenie terminálu



http://thegreyblog.blogspot.sk/2009/08/configuring-putty-to-use-utf-8.html



Prečo „nefunguje“ diakritika - locale

- SSH - závislé aj od konfigurácie ssh na serveri (`/etc/ssh/sshd_config`)
  - Program ssh* preposiela lokálne nastavenia - **HRON OK**
  - PUTTY - **HRON “problém”**
    - `LC_COLLATE="POSIX"`
- Overte a nastavte si locale pred prácou na hron.fei.tuke.sk

@hron.fei.tuke.sk

```
$ export LANG=sk_SK.UTF-8
$ locale
LANG=sk_SK.UTF-8
LANGUAGE=
LC_CTYPE="sk_SK.UTF-8"
LC_NUMERIC="sk_SK.UTF-8"
LC_TIME="sk_SK.UTF-8"
LC_COLLATE="sk_SK.UTF-8"
LC_MONETARY="sk_SK.UTF-8"
LC_MESSAGES="sk_SK.UTF-8"
LC_PAPER="sk_SK.UTF-8"
LC_NAME="sk_SK.UTF-8"
LC_ADDRESS="sk_SK.UTF-8"
LC_TELEPHONE="sk_SK.UTF-8"
LC_MEASUREMENT="sk_SK.UTF-8"
LC_IDENTIFICATION="sk_SK.UTF-8"
LC_ALL=
```

https://wiki.debian.org/Locale
https://help.ubuntu.com/community/Locale

* `$ ssh -V`
`OpenSSH_7.2p2 Ubuntu-4ubuntu2.1, OpenSSL 1.0.2g  1 Mar 2016`

Prístup na hron.fei.tuke.sk pridelí Ing. Popovec vo Fakultnom počítačovom centre po preukázaní sa ISIC-om

Skriptovanie v shelli

Skriptovanie

- UNIX/Linux
  - Textový súbor príkazov shell-u a príkazov dostupných v OS (je možné dopĺňať) - Bash, C shell, Z shell, a ďalšie
  - Perl
  - Python
  - A ďalšie “dynamické programovanie jazyky”
- MS Windows
  - Povelové súbory (*.bat, *.cmd – slabý odvar oproti UNIX/Linux)
  - VBscript
  - Powershell (mocný nástroj!!!)

Bash (1)

- `#!/bin/bash`
  - `#! interpreter [optional-arg]`
  - man 2 execve (http://man7.org/linux/man-pages/man2/execve.2.html)
- Comments `# Komentáre`
- Variables `variable1="pocitac"`
- Variable expansion `$variable1`
- Quotes `'Single Quotes'` `"$Double Quotes"`
- Using Braces to Protect Your Variables `"${A}a"`
- Conditionals, if/then/elif `if [ "$var1" -eq 12 ]; then cmd; fi`
- “new style of conditional test” `if [[ $filename = *.png ]]; then cmd; fi`
- Control Operators (&& and ||) `mkdir d && cd d`
- Wildcards `mv *.txt *.bak`
- ...

- http://wiki.bash-hackers.org/start
- http://wiki.bash-hackers.org/scripting/tutoriallist
- http://www.panix.com/~elflord/unix/bash-tute.html
- http://www.abclinuxu.cz/serialy/bash

Bash (2)

- For loops, While Loops; break, continue `for i in 1 2 3 4 5; do echo "$i"; done`
- Brace Expansion `echo {1..9}`
- The environment `export LANG=sk_SK.UTF-8`
- File Descriptors `echo "Something went really bad.." >&2`
- Redirection
  - `echo "It was a dark and stormy night.  Too dark to write." > story`
  - `cat < story`
  - `grep proud file 'not a file' > stdout.log 2> stderr.log`
- Pipes
  - `curl -s http://cbsg.sourceforge.net/cgi-bin/live | grep -Eo '^<`
    `sed s,\</\\?li\>,,g | shuf -n 1`
- Process substitution
  - `diff -y <(head -n 1 .dictionary) <(tail -n 1 .dictionary)`
- ...

- http://wiki.bash-hackers.org/start
- http://wiki.bash-hackers.org/scripting/tutoriallist
- http://www.panix.com/~elflord/unix/bash-tute.html
- http://www.abclinuxu.cz/serialy/bash

Bash (3)

- Command Substitution
- Arithmetic Expansion
- Arithmetic Commands
- Arrays
- Parameter Expansion
- Internal variables
- Choices (case and select)
- Functions
- Subshells
- Grouping Statements
- Sourcing
- Aliases, Job control, ...

```
"os-`date "+%Y-%m-%d-%H-%M-%S"`.zip"
i=$((j + 3))
let a="17 + 23"
files=(~/*.jpg); cp "${files[@]}" /backups/
"${parameter:-word}" … "${#parameter}" ...
IFS=","
case $LANG in en*) echo 'Hello' ;; esac
sum() { echo "$(( $1 + $2 ))"; }
(cd /tmp || exit 1; date > timestamp)
... && { rm "$file" || echo "Err..." >&2; }
source setup_environ.sh
```

- http://wiki.bash-hackers.org/start
- http://wiki.bash-hackers.org/scripting/tutoriallist
- http://www.panix.com/~elflord/unix/bash-tute.html
- http://www.abclinuxu.cz/serialy/bash

Parametre povelového riadku - Návratová hodnota

Parameter Name	Usage	Description
0	`"$0"`	Contains the name, or the path, of the script. This is not always reliable.
1 2 etc.	`"$1"` etc.	*Positional Parameters* contain the arguments that were passed to the current script or function.
*	`"$*"`	Expands to all the words of all the positional parameters. Double quoted, it expands to a single string containing them all, separated by the first character of the **IFS** variable (discussed later).
@	`"$@"`	Expands to all the words of all the positional parameters. Double quoted, it expands to a list of them all as individual words.
#	`$#`	Expands to the number of positional parameters that are currently set.
?	`$?`	Expands to the exit code of the most recently completed foreground command.
$	`$$`	Expands to the PID (process ID number) of the current shell.
!	`$!`	Expands to the PID of the command most recently executed in the background.
_	`"$_"`	Expands to the last argument of the last command that was executed.

http://mywiki.wooledge.org/BashGuide/Parameters

Externé príkazy

- grep, egrep
- sed
- awk, gawk
- sort, uniq
- head, tail
- tr (problém UTF-8)
- cut
- wc
- tee
- ...

grep

```
grep, egrep, fgrep - print lines matching a pattern

       grep [OPTIONS] PATTERN [FILE...]
       grep [OPTIONS] [-e PATTERN | -f FILE] [FILE...]

grep searches the named input FILEs for lines containing a match to
the given PATTERN.  If no files are specified, or if the file "-" is
given, grep searches standard input.  By default, grep prints the
matching lines.

In addition, the variant programs egrep and fgrep are the same as
grep -E and grep -F, respectively.  These variants are deprecated,
but are provided for backward compatibility.
```

man 1 grep
(http://man7.org/linux/man-pages/man1/grep.1.html)

grep - mód porovnávania textu

Matcher Selection

```
-E, --extended-regexp
       Interpret PATTERN as an extended regular expression (ERE, see
       below).

-F, --fixed-strings
       Interpret PATTERN as a list of fixed strings (instead of
       regular expressions), separated by newlines, any of which is
       to be matched.

-G, --basic-regexp
       Interpret PATTERN as a basic regular expression (BRE, see
       below).  This is the default.

-P, --perl-regexp
       Interpret the pattern as a Perl-compatible regular expression
       (PCRE).  This is highly experimental and grep -P may warn of
       unimplemented features.
```

man 1 grep
(http://man7.org/linux/man-pages/man1/grep.1.html)

grep - riadenie výberu

```
Matching Control
      -i, --ignore-case
              Ignore case distinctions in both the PATTERN and the input
              files.
      -v, --invert-match
              Invert the sense of matching, to select non-matching lines.
General Output Control
      -c, --count
              Suppress normal output; instead print a count of matching
              lines for each input file.  With the -v
      -o, --only-matching
              Print only the matched (non-empty) parts of a matching line,
              with each such part on a separate output line.
Output Line Prefix Control
      -n, --line-number
              Prefix each line of output with the 1-based line number within
              its input file.
```

man 1 grep
(http://man7.org/linux/man-pages/man1/grep.1.html)

grep - DEMO

```
grep auto /usr/share/dict/words
grep -o auto /usr/share/dict/words

grep -i "a" /usr/share/dict/words

grep -c '\bc...h\b' /usr/share/dict/words
grep '\bc...h\b' /usr/share/dict/words | wc -l

grep "dog|cat" /usr/share/dict/words
grep -E "dog|cat" /usr/share/dict/words

grep -E "cat[^i]+" /usr/share/dict/words
grep -E "(at.){2}" /usr/share/dict/words
egrep -E "[a-ž]+kl?[c-f]" /usr/share/dict/words
```

```
$ head /usr/share/dict/words
A
A's
AA's
AB's
ABM's
AC's
ACTH's
AI's
AIDS's
AM's
```

man 1 grep
(http://man7.org/linux/man-pages/man1/grep.1.html)

sort

```
sort - sort lines of text files

sort [OPTION]... [FILE]...
sort [OPTION]... --files0-from=F

Write sorted concatenation of all FILE(s) to standard output.
With no FILE, or when FILE is -, read standard input.

      -u, --unique
              with -c, check for strict ordering; without -c, output only
              the first of an equal run
      -r, --reverse
              reverse the result of comparisons
      --sort=WORD
              sort according to WORD: general-numeric -g, human-numeric -h,
              month -M, numeric -n, random -R, version -V
...
```

man 1 sort
(http://man7.org/linux/man-pages/man1/sort.1.html)

sort - DEMO

```
sort food.txt
sort -u food.txt
sort -uf food.txt
sort -u -f food.txt
sort -ur food.txt
sort -R food.txt

sort -n numbers.txt
sort -h numbers.txt
sort -g numbers.txt

sort -u food.txt numbers.txt
```

```
$ cat food.txt
pears
čučoriedky
oranges
apples
kiwis
bananas
chlieb
HamBurGer
hamburger
apples
apples
apples green
```

```
$ cat numbers.txt
4564.445
789.45
44564.456
4
1e15
8
88.01
88
1e+15
5e3
lietadlo
123
```

man 1 sort
(http://man7.org/linux/man-pages/man1/sort.1.html)

uniq

```
uniq - report or omit repeated lines

uniq [OPTION]... [INPUT [OUTPUT]]

Filter adjacent matching lines from INPUT (or standard input),
writing to OUTPUT (or standard output).
With no options, matching lines are merged to the first occurrence.

      -c, --count
             prefix lines by the number of occurrences
      -D     print all duplicate lines
      -d, --repeated
             only print duplicate lines, one for each group
      -u, --unique
             only print unique lines
...
```

man 1 uniq
(http://man7.org/linux/man-pages/man1/sort.1.html)

uniq - DEMO

```
uniq food.txt
uniq -i food.txt
uniq -c food.txt
sort food.txt | uniq -c
```

$ cat food.txt

```
pears
čučoriedky
oranges
apples
kiwis
bananas
chlieb
HamBurGer
hamburger
apples
apples
apples green
```

$ cat numbers.txt

```
4564.445
789.45
44564.456
4
1e15
8
88.01
88
1e+15
5e3
lietadlo
123
```

man 1 uniq
(http://man7.org/linux/man-pages/man1/sort.1.html)

head

```
head - output the first part of files

head [OPTION]... [FILE]...

Print the first 10 lines of each FILE to standard output.  With more
than one FILE, precede each with a header giving the file name.
With no FILE, or when FILE is -, read standard input.

       -c, --bytes=[-]NUM
              print the first NUM bytes of each file; with the leading '-',
              print all but the last NUM bytes of each file
       -n, --lines=[-]NUM
              print the first NUM lines instead of the first 10; with the
              leading '-', print all but the last NUM lines of each file
       -z, --zero-terminated
              line delimiter is NUL, not newline
...
```

man 1 head
(http://man7.org/linux/man-pages/man1/head.1.html)

head - DEMO

```
head food.txt
head -n 3 food.txt
head -c 3 food.txt
head -n 3 food.txt

head -n 3 food.txt numbers.txt
```

```
$ cat food.txt
pears
čučoriedky
oranges
apples
kiwis
bananas
chlieb
HamBurGer
hamburger
apples
apples
apples green
```

```
$ cat numbers.txt
4564.445
789.45
44564.456
4
1e15
8
88.01
88
1e+15
5e3
lietadlo
123
```

man 1 head
(http://man7.org/linux/man-pages/man1/head.1.html)

tail

```
tail - output the last part of files

tail [OPTION]... [FILE]...

Print the last 10 lines of each FILE to standard output.  With more
than one FILE, precede each with a header giving the file name.
With no FILE, or when FILE is -, read standard input.

-c, --bytes=[+]NUM
              output the last NUM bytes; or use -c +NUM to output starting
              with byte NUM of each file
-n, --lines=[+]NUM
              output the last NUM lines, instead of the last 10; or use -n
              +NUM to output starting with line NUM
-f, --follow[={name|descriptor}]
              output appended data as the file grows;
              an absent option argument means 'descriptor'
```

...

man 1 tail
(http://man7.org/linux/man-pages/man1/tail.1.html)

tail - DEMO

```
tail food.txt
tail -n 3 food.txt
tail -c 3 food.txt
tail -n 3 food.txt
```

```
bash does_something.sh > something.txt &
tail -f something.txt
```

```
$ cat does_something.sh
#!/bin/bash

a=0
while [ $a -le 30 ]
do
     echo "Doing something ... $a"
     sleep 1
     ((a++))
done
```

man 1 tail
(http://man7.org/linux/man-pages/man1/tail.1.html)

cut

```
cut - remove sections from each line of files

cut OPTION... [FILE]...

       -b, --bytes=LIST
              select only these bytes

       -c, --characters=LIST
              select only these characters

       -d, --delimiter=DELIM
              use DELIM instead of TAB for field delimiter

       -f, --fields=LIST
              select only these fields;  also print any line that contains
              no delimiter character, unless the -s option is specified

...
```

man 1 cut
(http://man7.org/linux/man-pages/man1/cut.1.html)

cut - DEMO

```
cut -f2 MOCK_DATA.txt
cut -f2-5 MOCK_DATA.txt
cut -f2- MOCK_DATA.txt
cut -f-2 MOCK_DATA.txt

cut -f-2 MOCK_DATA.csv
cut -f-2 --delimiter=, MOCK_DATA.csv

cut -c-4 MOCK_DATA.txt
```

man 1 cut

```
N      N'th byte, character or field, counted from 1

N-     from N'th byte, character or field, to end of line

N-M    from N'th to M'th (included) byte, character or field

-M     from first to M'th (included) byte, character or field
```

$ cat MOCK_DATA.txt

```
id      first_name last_name  email      gender     ip_address
1       Ruby       Castillo   rcastillo0@dagondesign.com       Female     192.26.253.135
2       Russell    Rogers     rrogers1@hubpages.com Male       206.196.183.168
3       Jeremy     Wright     jwright2@sina.com.cn  Male       190.59.128.154
4       Carol      Jenkins    cjenkins3@un.org      Female     215.151.184.97
5       Nicole     Wright     nwright4@cpanel.net   Female     103.84.16.204
6       Virginia   Murray     vmurray5@ehow.com     Female     176.203.196.103
7       Catherine  Black      cblack6@bandcamp.com  Female     158.119.148.77
8       Kathy      Brooks     kbrooks7@chicagotribune.com      Female     128.140.137.136
9       William    Castillo   wcastillo8@statcounter.com       Male       145.212.191.16
10      Juan       Hanson     jhanson9@unesco.org   Male       72.87.60.82
```

www.mockaroo.com

$ cat MOCK_DATA.csv

```
id,first_name,last_name,email,gender,ip_address
1,Laura,Austin,laustin0@prweb.com,Female,230.211.14.154
2,Thomas,Torres,ttorres1@bravesites.com,Male,28.241.194.5
3,Wayne,Price,wprice2@amazon.de,Male,16.68.120.119
4,Karen,Cole,kcole3@plala.or.jp,Female,22.49.119.107
5,Ernest,Schmidt,eschmidt4@msn.com,Male,46.89.58.120
6,Larry,Kennedy,lkennedy5@spiegel.de,Male,19.204.23.34
7,Shawn,Alexander,salexander6@multiply.com,Male,229.247.163.18
8,Louise,Reid,lreid7@google.de,Female,133.16.2.23
9,Adam,Hudson,ahudson8@scientificamerican.com,Male,65.62.159.247
10,Amy,Bishop,abishop9@google.cn,Female,157.200.29.112
```

man 1 cut
(http://man7.org/linux/man-pages/man1/cut.1.html)

wc

```
wc - print newline, word, and byte counts for each file

wc [OPTION]... [FILE]...
wc [OPTION]... --files0-from=F

Print newline, word, and byte counts for each FILE, and a total line
if more than one FILE is specified.  A word is a non-zero-length
sequence of characters delimited by white space.
With no FILE, or when FILE is -, read standard input.

      -c, --bytes
             print the byte counts

      -m, --chars
             print the character counts

      -l, --lines
             print the newline counts
...
```

man 1 wc
(http://man7.org/linux/man-pages/man1/wc.1.html)

wc - DEMO

```
wc food.txt
wc -c food.txt
wc -l food.txt
wc -m food.txt
wc -w food.txt

sort food.txt | uniq -c | wc -l

egrep -E "[a-ž]+kl?[c-f]"
/usr/share/dict/words | wc -l
```

$ cat food.txt

```
pears
čučoriedky
oranges
apples
kiwis
bananas
chlieb
HamBurGer
hamburger
apples
apples
apples green
```

$ cat numbers.txt

```
4564.445
789.45
44564.456
4
1e15
8
88.01
88
1e+15
5e3
lietadlo
123
```

man 1 wc
(http://man7.org/linux/man-pages/man1/wc.1.html)

tr

Nepodporuje multi-bajtové znaky*
(napr. slovenské znaky v UTF-8)

```
tr - translate or delete characters
tr [OPTION]... SET1 [SET2]

Translate, squeeze, and/or delete characters from standard input,
       writing to standard output.

       -c, -C, --complement
              use the complement of SET1
       -d, --delete
              delete characters in SET1, do not translate
       -s, --squeeze-repeats
              replace each sequence of a repeated character that is listed
              in the last specified SET, with a single occurrence of that
              character
       -t, --truncate-set1
              first truncate SET1 to length of SET2
...
```

man 1 tr
(http://man7.org/linux/man-pages/man1/tr.1.html)

* Pozri. How to make tr aware of non-ascii(unicode) characters?. Závisí aj na implementácií. Napr. znak ‘č’ má v UTF-8 kód 0xC48D

tr - DEMO

Nepodporuje multi-bajtové znaky*
(napr. slovenské znaky v UTF-8)

```
cat food.txt | tr abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ

cat food.txt | tr "[:lower:]" "[:upper:]"

echo "This   is   for testing" | tr [:space:] '\t'
echo "This   is   for testing" | tr -s [:space:] '\t'

echo "This   is   for testing" | tr -d ' '

echo "my username is 432234" | tr -d "[:digit:]"
echo "my username is 432234" | tr -cd "[:digit:]"
```

```
$ cat food.txt
pears
čučoriedky
oranges
apples
kiwis
bananas
chlieb
HamBurGer
hamburger
apples
apples
apples green
```

man 1 tr
(http://man7.org/linux/man-pages/man1/tr.1.html)

* Pozri. How to make tr aware of non-ascii(unicode) characters?. Závisí aj na implementácií. Napr. znak ‘č’ má v UTF-8 kód 0xC48D

sed

```
sed - stream editor for filtering and transforming text

sed [OPTION]... {script-only-if-no-other-script} [input-file]...

Sed is a stream editor.  A stream editor is used to perform basic
text transformations on an input stream (a file or input from a
pipeline).  While in some ways similar to an editor which permits
scripted edits (such as ed), sed works by making only one pass over
the input(s), and is consequently more efficient.  But it is sed's
ability to filter text in a pipeline which particularly distinguishes
it from other types of editors.

      -r, --regexp-extended
              use extended regular expressions in the script.
...
```

man 1 sed
(http://man7.org/linux/man-pages/man1/sed.1.html)

sed

- seach and replace:
  - `'s/<search>/<replace>/<opt>'`
  - Options:
    - `g` - global, handles all occurencies
    - `i` - ignore case
  - `$ sed 's/help/HELP/' file`
- appending lines:
  - `'Na`LINE OF TEXT'
  - `$ sed '3a`New Line' file.txt
  - `$ sed '/help/aAFTER' file.txt`
- inserting lines:
  - `'Ni`LINE OF TEXT'
  - `$ sed '3i`New Line' file.txt
- deleting lines:
  - `'F,TD'` - deletes lines between lines F and T
    - if no F and T lines are specified, deletes all the lines
  - `$ sed '/help/D' file.txt`

sed - DEMO

```
sed -r 's/čučo.+/rum/' food.txt

sed -r 's#čučo.+#rum#' food.txt

sed -r 's/čučo(.+)/rum \1/' food.txt

sed -r "s/\bR[^\t ]+/Jožo/g" MOCK_DATA.txt

sed '/apple/aJablko' food.txt

sed '/hamburger/ID' food.txt

...
```

```
$ cat food.txt
pears
čučoriedky
oranges
apples
kiwis
bananas
chlieb
HamBurGer
hamburger
apples
apples
apples green
```

https://www.google.sk/search?q=sed%20examples

awk

```
awk - pattern-directed scanning and processing language

awk [ -F fs ] [ -v var=value ] [ 'prog' | -f progfile ] [ file ...  ]

Awk scans each input file for lines that match any of a set of patterns
specified literally in prog or in one or more files specified as -f
progfile. With each pattern there can be an associated action that
will be performed when a line of a file matches the pattern. Each line
is matched against the pattern portion of every pattern-action state-
ment; the associated action is performed for each matched pattern.
```

man 1 awk
(https://www.freebsd.org/cgi/man.cgi?query=awk&sektion=1)

awk

- awk program is sequence of lines: `pattern {action}`
  - pattern - what to search for (enclosed inside of //)
  - action - what to do with
  - `$ awk '/hello/ {print}' file`
- each column in line is indexed with number starting from `$1`
  - `$ awk '{print $1 $2}' file`
  - `$ awk '{print $1,$2}' file`
  - option to identify separator **`-F`**

awk - DEMO

```
awk '{print;}' MOCK_DATA.txt

awk '/Bruce/
/Janet/' MOCK_DATA.txt

awk '{print $2,$3,$6;}' MOCK_DATA.txt
awk -F , '{print $2,$3,$6;}' MOCK_DATA.csv

ls -l | awk '{print $1,"\t",$9;}'

awk 'BEGIN {print "Name\tGender\tEmail";}
{print $1,$2,"\t",$5,"\t",$4,$NF;}
END{print "Všetko\n--------------------";
}' MOCK_DATA.txt
```

$ cat MOCK_DATA.txt

```
id      first_name last_name  email      gender     ip_address
1       Ruby       Castillo   rcastillo0@dagondesign.com       Female     192.26.253.135
2       Russell    Rogers     rrogers1@hubpages.com Male       206.196.183.168
3       Jeremy     Wright     jwright2@sina.com.cn  Male       190.59.128.154
4       Carol      Jenkins    cjenkins3@un.org      Female     215.151.184.97
5       Nicole     Wright     nwright4@cpanel.net   Female     103.84.16.204
6       Virginia   Murray     vmurray5@ehow.com     Female     176.203.196.103
7       Catherine  Black      cblack6@bandcamp.com  Female     158.119.148.77
8       Kathy      Brooks     kbrooks7@chicagotribune.com      Female     128.140.137.136
9       William    Castillo   wcastillo8@statcounter.com       Male       145.212.191.16
10      Juan       Hanson     jhanson9@unesco.org   Male       72.87.60.82
```

www.mockaroo.com

$ cat MOCK_DATA.csv

```
id,first_name,last_name,email,gender,ip_address
1,Laura,Austin,laustin0@prweb.com,Female,230.211.14.154
2,Thomas,Torres,ttorres1@bravesites.com,Male,28.241.194.5
3,Wayne,Price,wprice2@amazon.de,Male,16.68.120.119
4,Karen,Cole,kcole3@plala.or.jp,Female,22.49.119.107
5,Ernest,Schmidt,eschmidt4@msn.com,Male,46.89.58.120
6,Larry,Kennedy,lkennedy5@spiegel.de,Male,19.204.23.34
7,Shawn,Alexander,salexander6@multiply.com,Male,229.247.163.18
8,Louise,Reid,lreid7@google.de,Female,133.16.2.23
9,Adam,Hudson,ahudson8@scientificamerican.com,Male,65.62.159.247
10,Amy,Bishop,abishop9@google.cn,Female,157.200.29.112
```

https://www.google.sk/search?q=awk%20examples

Niektoré ďalšie zaujímavé príkazy

- ps
  - Report a snapshot of the current processes
  - man 1 ps (http://man7.org/linux/man-pages/man1/ps.1.html)
- lsof
  - List open files
  - man 8 lsof (https://linux.die.net/man/8/lsof)
- dd
  - Convert and copy a file
  - man 1 dd (http://man7.org/linux/man-pages/man1/dd.1.html)
  - Disk cloning (e.g. https://wiki.archlinux.org/index.php/disk_cloning)

Regulárne výrazy

Podstata

- Poskytujú mechanizmus na špecifikáciu vzorov
- Najjednoduchšia forma – presne špecifikovaný reťazec (exact match)
- Môžu reprezentovať širokú škálu súboru (postupností) znakov
- Rozlišujeme
  - Basic RE
  - Extended RE

- https://regex101.com/
- http://www.regular-expressions.info/
- https://www.google.sk/search?q=regular%20expressions%20tutorial

Triedy znakov

- Nie exkluzívne len regulárne výrazy
  - ISO C90 - hlavičkový súbor <ctype.h> - isalnum(), isalpha(), islower(), isupper(), …
  - Nástroj **tr** - man 1 tr
- `[:alnum:]` `[:cntrl:]` `[:lower:]` `[:space:]` `[:alpha:]` `[:digit:]` `[:print:]` `[:upper:]` `[:blank:]` `[:graph:]` `[:punct:]` `[:xdigit:]`

Basic Regular Expressions

- Špeciálne znaky
  - `.`
  - `[`
  - `\`
  - `*`
  - `^`
  - `$`
- `grep`
- Character Classes or Character Sets `[a-ž0-9]` `[^AabBc-]`
- Shorthand Character Classes `\s` `\S` `\w` `\W` `\d` `\D`
- Repetition `a*`
- Dot `a..j`
- Start of String and End of String Anchors `^a.+z$`
- ...

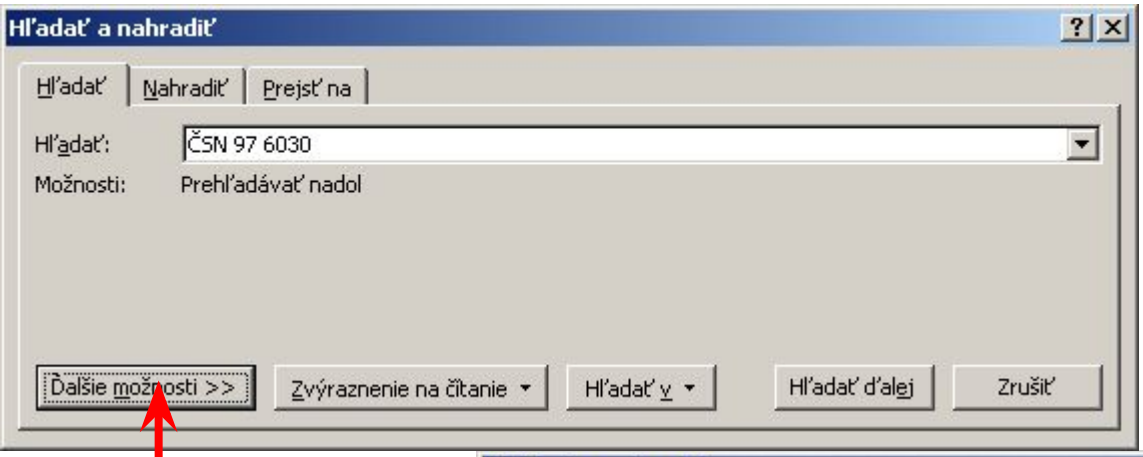
(http://pubs.opengroup.org/onlinepubs/007908799/xbd/re.html#tag_007_003)

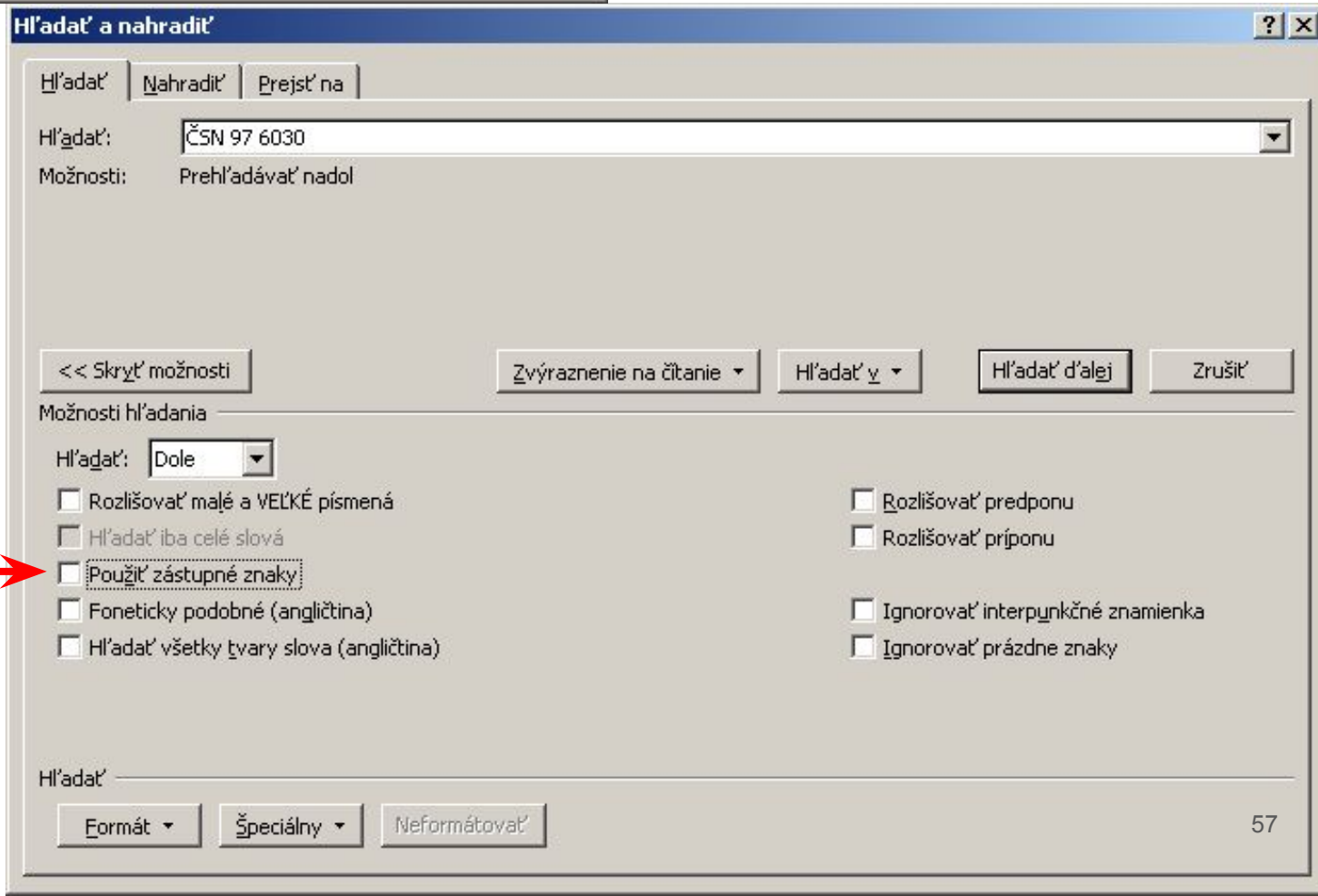
Extended Regular Expressions

- Špeciálne znaky
  - `. [ \ (`
  - `* + ? {`
  - `|`
  - `^`
  - `$`
- `grep -E` alebo `egrep` alebo `grep -P`
- greedy a non-greedy RE
  - greedy - vyberá sa najdlhší možný podreťazec `<.+>`
  - non-greedy - vyberá sa najkratší možný podreťazec `<.+?>`
    - `grep -E` nepodporuje, dostupné v `grep -P` (perl compatible)
- Groups `ah(oj|a)`
- Repetition `a+` `a*` `a{4}` `a{4,8}` `a{4,}`
- Optional items `Feb(ruary)? 23(rd)?` `boys?`
- Alternation `cat|dog|mouse|fish`
- Backreferences `<([A-Z][A-Z0-9]*)\b[^>]*>.*?</\1>`
- Lookahead and Lookbehind (perl-compatible) `q(?!u)` `q(?=u)` `(?<!a)b` `(?<=a)b`

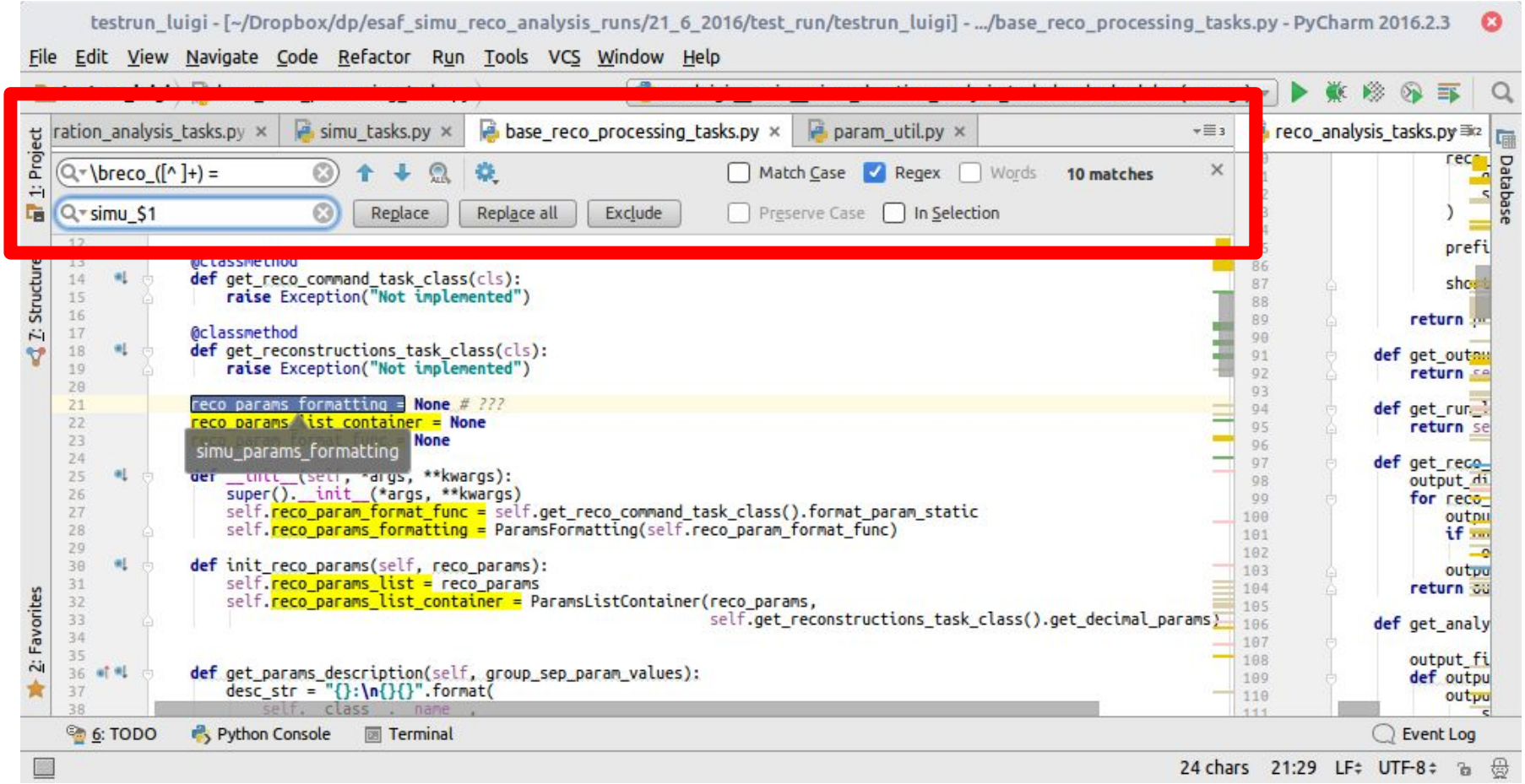
(http://pubs.opengroup.org/onlinepubs/007908799/xbd/re.html#tag_007_004)

Microsoft Word

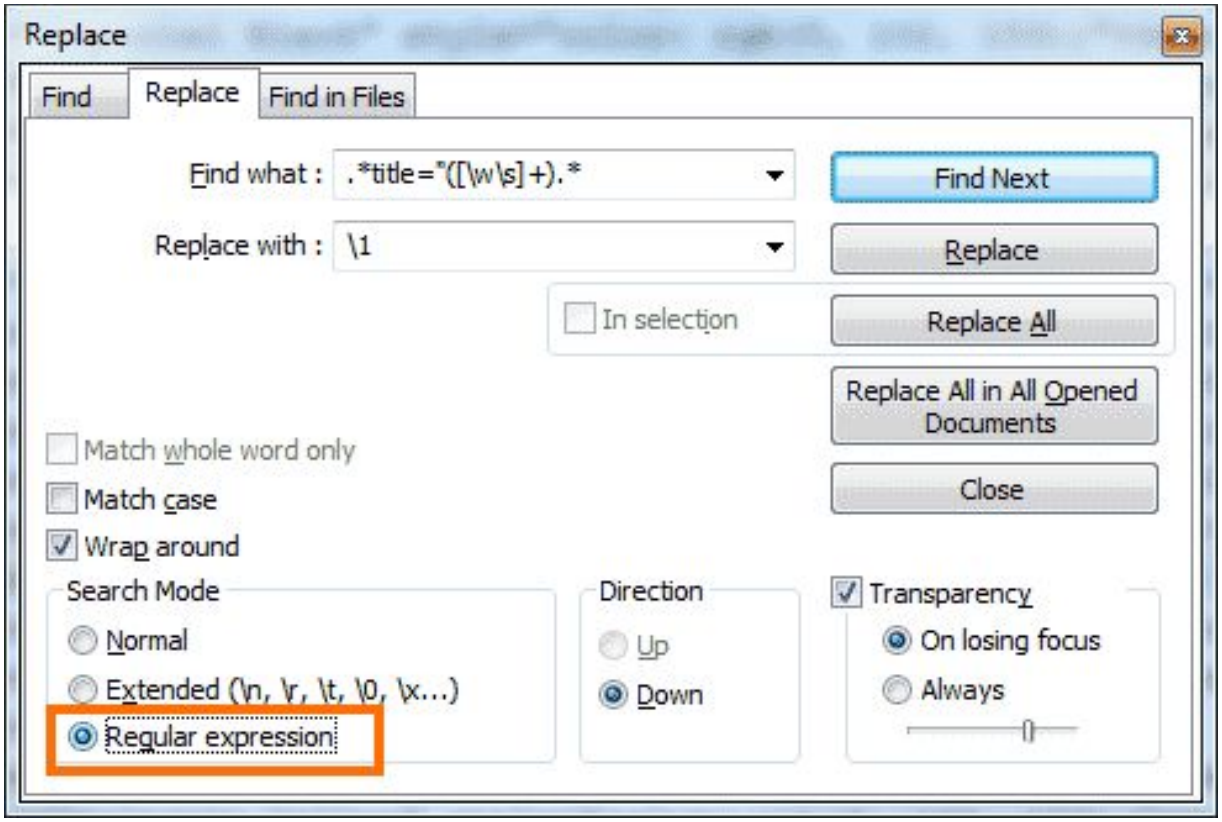




PyCharm


testrun_luigi - [~/Dropbox/dp/esaf_simu_reco_analysis_runs/21_6_2016/test_run/testrun_luigi] - .../base_reco_processing_tasks.py - PyCharm 2016.2.3
File Edit View Navigate Code Refactor Run Tools VCS Window Help
ration_analysis_tasks.py
simu_tasks.py
base_reco_processing_tasks.py
param_util.py
reco_analysis_tasks.py
\breco_([^]+) =
simu_$1
Match Case
Regex
Words
10 matches
Replace
Replace all
Exclude
Preserve Case
In Selection
1: Project
7: Structure
2: Favorites
Database
def get_reco_command_task_class(cls):
raise Exception("Not implemented")
@classmethod
def get_reconstructions_task_class(cls):
raise Exception("Not implemented")
reco_params_formatting = None # ???
reco_params_list_container = None
None
simu_params_formatting
super().__init__(*args, **kwargs)
self.reco_param_format_func = self.get_reco_command_task_class().format_param_static
self.reco_params_formatting = ParamsFormatting(self.reco_param_format_func)
def init_reco_params(self, reco_params):
self.reco_params_list = reco_params
self.reco_params_list_container = ParamsListContainer(reco_params,
self.get_reconstructions_task_class().get_decimal_params)
def get_params_description(self, group_sep_param_values):
desc_str = "{}:\n{}{}".format(
6: TODO
Python Console
Terminal
Event Log
24 chars
21:29
LF
UTF-8

Notepad++



(http://docs.notepad-plus-plus.org/index.php/Regular_Expressions)