



# Přednáška 8

Proměnné. Psaní a ladění skriptů.

Parametry skriptu. Vstup a výstup.

Konfigurační soubory shellu.



# Proměnné

- Jména nových proměnných by neměly kolidovat se jmény předdefinovaných proměnných.
- Explicitně je hodnota proměnné uložena jako **řetězec**.

- **Lokální proměnná**

- je známá pouze v instanci shellu, kde byla definována

`JMENO=HODNOTA`

- **Proměnná prostředí (exportovaná)**

- proměnná se dědí (při vzniku potomka se překopíruje do prostředí potomka)

`JMENO=HODNOTA ; export JMENO`

`export JMENO=HODNOTA` (ne v sh)

- v sh je třeba export opakovat po každé změně proměnné

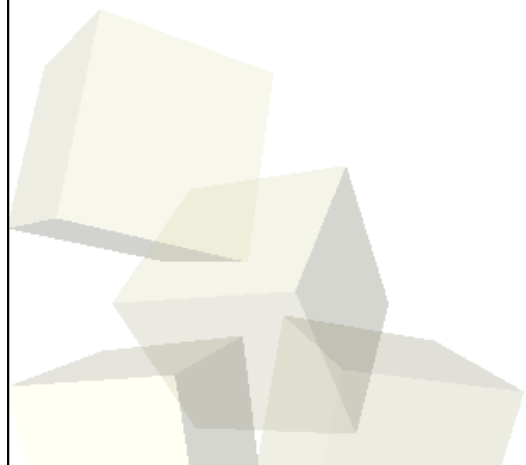
# Předefinované proměnné

- Slouží k **definování prostředí** (např. PS1, SHELL, PATH...).
- Některé proměnné mají hodnotu nastavenou automaticky (např. PWD, HOME, \$#, \$1,...) jiné můžeme modifikovat (např. PATH, PS1,...).
- Úplný seznam předdefinovaných proměnných viz. např. [man bash](#).

Jméno proměnné	Význam proměnné
HOME	domovský adresář
PWD	pracovní adresář
LOGNAME	uživatelské jméno
HOSTNAME	jméno počítače
PS1, PS2, ...	prompt 1. a 2. úrovně
PATH	seznam adresářů, kde se hledají spustitelné programy
MANPATH	seznam adresářů, kde se hledají man. stránky
IFS	vstupní oddělovač (viz. příkaz read)

# Předefinované proměnné II

Jméno proměnné	Význam proměnné
<code>\$#</code>	Počet parametrů skriptu
<code>\$0</code>	Jméno skriptu
<code>\$1, ..., \$9, \${10}, ...</code>	Parametry skriptu (sh pouze \$1, ..., \$9)
<code>\$*</code>	<code>\$1 \$2 \$3 ...</code>
<code>\$@</code>	<code>\$1 \$2 \$3 ...</code>
<code>"\$*"</code>	<code>"\$1 \$2 \$3 ..."</code>
<code>"\$@"</code>	<code>"\$1" "\$2" "\$3" ...</code>



# Hodnoty proměnných

Syntaxe	Význam
<code>\$JMENO</code>	hodnota proměnné
<code>\${JMENO}</code>	hodnota proměnné
<code>\${JMENO:-text}</code>	je-li <code>JMENO</code> prázdné, pak vrátí <code>text</code> , jinak <code>\$JMENO</code>
<code>\${JMENO:=text}</code>	je-li <code>JMENO</code> prázdné, pak <code>JMENO=text</code> a vrátí <code>\$JMENO</code>
<code>\${JMENO:?text}</code>	je-li <code>JMENO</code> prázdné, pak vypíše <code>text</code> a končí ( <code>exit</code> )

- **Zobrazení hodnoty proměnné**  
`echo $JMENO; echo ${JMENO}; ...`
- **Zobrazení všech proměnných (lokální + exportovaných)**  
`set`
- **Zobrazení pouze exportovaných proměnných**  
`env`
- **Zrušení proměnné**  
`unset JMENO`

# Konstanty

- **Definice**

**JMENO=HODNOTA**

**readonly JMENO**

**typeset -r JMENO=HODNOTA** (ne sh)

- Přiřazení do konstanty způsobí chybu.

# Celočíselné proměnné

- Umožňuje pouze ksh a bash.

- **Definice**

```
typeset -i JMENO=HODNOTA
```

- Přiřazení řetězce do celočíselné proměnné způsobí chybu (u ksh) nastaví 0 (u bash).



# Pole

- Jednorozměrné.
- Index je číslo nebo celočíselná proměnná bez \$ v rozsahu 0-4095.

- **Přiřazení**

`JMENO[index]=HODNOTA`

- **Hodnota položky**

`$ {JMENO[index]}`

- **Hodnoty všech definovaných položek**

`$ {JMENO[*]}`

- **Počet všech definovaných položek**

`$ {#JMENO[*]}`



# Vytvoření skriptu

- **Skript**

- obyčejný textový soubor, může obsahovat jednoduché a složené příkazy, definice proměnných, komentáře,...

```
#!/bin/sh

# komentar

A=2 # komentar
B=5

echo "Hodnota promenne A je $A"

echo "Hodnota promenne B je $B"

/bin/echo "Dnes je \c"

date '+%d.%m.%Y'
```

# Spuštění skriptu

- **v aktuální instanci shellu**

- `skript` (min. práva skriptu r--)

- **v nové instanci shellu**

- `/bin/sh skript` (min. práva skriptu r--)

- `./skript` (min. práva skriptu r-x)

# Ladění skriptu

- **v-mód** = původní řádka ze skriptu se nejdříve vytiskne na standardní výstup a pak se spustí
- **x-mod** = provedou se náhrady spec. znaků na řádce, vytiskne se na standardní výstup a pak se spustí
  
- **Ladění celého skriptu**
  - spustíme shell s parametrem **-v**
  - spustíme shell s parametrem **-x**
  
- **Ladění části skriptu**
  - povolení ladících výpisů pomocí  
`set -v` a `set -x`
  - zakázání ladících výpisů pomocí  
`set +v` a `set +x`

# Příklad

- **Skript debug1.sh**

```
#!/bin/sh -vx

# komentar

A=2 # komentar
B=5

echo "Hodnota promenne A je $A"
echo "Hodnota promenne B je $B"

/bin/echo "Dnes je \c"
date '+%d.%m.%Y'
```

# Příklad

- **Skript debug2.sh**

```
#!/bin/sh
# komentar

A=2 # komentar
B=5
echo "Hodnota promenne A je $A"

set -x # zacatek debugovani

echo "Hodnota promenne B je $B"

set +x # konec debugovani

/bin/echo "Dnes je \c"
date '+%d.%m.%Y'
```

# Parametry skriptu I

- Při spuštění skriptu se jméno skriptu a argumenty na příkazové řádce uloží do následujících proměnných:

Jméno proměnné	Význam proměnné
<code>\$#</code>	Počet parametrů skriptu
<code>\$0</code>	Jméno skriptu
<code>\$1, ..., \$9, \${10}, ...</code>	Parametry skriptu (sh pouze \$1,...,\$9)
<code>\$*</code>	<code>\$1 \$2 \$3 ...</code>
<code>\$@</code>	<code>\$1 \$2 \$3 ...</code>
<code>"\$*"</code>	<code>"\$1 \$2 \$3 ..."</code>
<code>"\$@"</code>	<code>"\$1" "\$2" "\$3" ...</code>

# Příklad

- **Skript param1.sh:**

```
#!/bin/sh

echo "Jmeno skriptu:      $0"
echo "Pocet parametru:   $#"
```

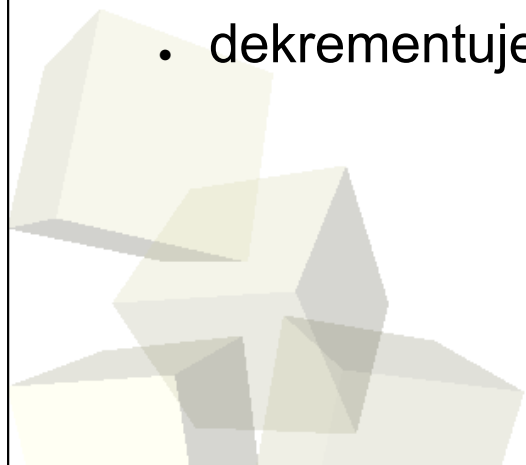
echo "Hodnota \"\\$\*\": \$\*"
echo "Hodnota \"\\$@\": @\$"

```
echo
echo "Hodnota 1. parametru:  $1"
echo "Hodnota 2. parametru:  $2"
```

# Parametry skriptu II

- **Příkaz:** `set -- seznam parametrů`
  - rozdělí seznam parametrů podle `$IFS` a přiřadí je do `$1,$2,$3...`
- **Příkaz:** `shift n`
  - posune hodnoty parametrů vlevo: `$i = ${i+n}`
  - odebere parametry z `$*` a `$@`
  - dekrementuje: `$# = $# - n`





# Příklad

- **Skript param2.sh:**

```
#!/bin/bash

echo "Pocet parametru: $#"
while [ $# -gt 0 ]
do
    echo "Hodnota \$#: $#"
    echo "Hodnota \$*: $*"
    echo "Hodnota \$@: @$"
    echo "Hodnota \$1: $1"
    echo

    shift
done
```

# Příklad

- **Skript param3.sh:**

```
#!/bin/bash

I=1

echo "Pocet parametru: $#"
while [ $# -gt 0 ]
do
    echo "Hodnota parametru $I: $1"
    shift
    I=`expr $I + 1`
done
```

# Příklad

- **Skript param4.sh:**

```
#!/bin/sh

echo "Pocet parametru: $#"  
echo  
  
I=1  
  
for J # in "$@"  
do  
    echo "Hodnota parametru $I: $J"  
    I=`expr $I + 1`  
done
```

# Příkaz read

```
read P1 P2 P3
```

- Přečte jednu řádku ze vstupu.
- Podle proměnné **\$IFS** rozdělí načtenou řádku na jednotlivé hodnoty.
- Uloží první hodnotu do proměnné **P1**, druhou položku do proměnné **P2** a ostatní hodnoty do proměnné **P3** .



# Příklad

- **Skript read1.sh:**

```
#!/bin/sh

while :
do
  /bin/echo "Zadej cele cislo [0,...99][k=konec]: \c"
  read C
  case $C in
    k)
      break
      ;;
    [0-9]|[0-9][0-9] )
      echo "Druha mocnina cisla $C je `expr $C \* $C`."
      ;;
    *) echo "Spatny parametr."
  esac
done
```

# Příklad

- **Skript read2.sh:**

```
#!/bin/sh

echo "Informace o uzivatelich v /etc/passwd"

IFS=":"

while read JMENO X UID GID POPIS DIR LOGSHELL
do
    echo "Ucet $JMENO ma:"
    echo "    UID=$UID"
    echo "    GID=$GID"
    echo "    HOME=$DIR"
    echo "    SHELL=${LOGSHELL:-neni definovan}"
done < /etc/passwd
```

# Příklad

- **Skript heslo.sh:**

```
#!/bin/sh

/bin/echo "Zadej jmeno: \c"
read JMENO

stty -echo

/bin/echo "Zadej heslo: \c"
read HESLO

stty echo

echo
echo
echo "Jmeno je $JMENO"
echo "Heslo je $HESLO"
```

# Funkce I

```
function jméno_funkce
{
    seznam_příkazů
}
```

```
jméno_funkce ( )
{
    seznam_příkazů
}
```

- Funkce se volá jménem a volitelnými parametry
- **Počet a hodnoty parametrů jsou uvnitř funkce dostupné** pomocí **\$#** a pozičních parametrů **\$1**, **\$2**, ...
- Funkce může volat jinou funkci případně sebe rekurzivně.
- **Návrat z funkce** proběhne po provedení všech příkazů s návratovým kódem posledního provedeného příkazu nebo předčasně příkazem **return návratový\_kód**.



# Funkce II

- **Proměnné deklarované uvnitř funkce existují i po návratu z funkce.** Výjimku tvoří proměnné definované příkazem `typeset` (ne `sh`), které jsou po návratu z funkce zrušeny příkazem `unset`.

- **Zobrazení nadefinovaných funkcí:**

`set` (sh)

`typeset -f` (ksh a bash)

- **Zrušení funkce**

`unset -f funkce`

# Příklad

- **Skript funkce1.sh:**

```
#!/bin/sh

a()
{
    echo "Pocet parametru funkce: $#"
```

*(Note: The following lines are indented in the original image)*

```
    echo "Parametry funkce:      $@"
    echo
}
```

*(Note: The following lines are not indented in the original image)*

```
echo "Pocet parametru skriptu:  $#"
```

*(Note: The following lines are not indented in the original image)*

```
echo "Parametry skriptu:       $@"
echo
```

```
a aa bbb "ss ff"
```

```
a 1 2 3 4 "5 6 7"
```

# Příklad

- **Skript funkce2.sh:**

```
#!/bin/sh

maximum()
{
    if [ $1 -gt $2 ] ; then
        echo "$1"
    else
        echo "$2"
    fi
}

if [ $# -ne 2 ] ; then
    echo "Pouziti: $0 num1 num2"
    exit 1
fi

echo "Maximun je `maximum $1 $2` "
```

# Konfigurační soubory shellu

Shell	Globální soubor	Uživatelský soubor pro přihlašovací shell	Uživatelský soubor pro další instanci shellu
/bin/sh	/etc/profile	\$HOME/.profile	
/bin/ksh	/etc/profile	\$HOME/.profile [ ENV=.kshrc ] \$HOME/.kshrc	\$HOME/.kshrc
/bin/bash	/etc/profile	\$HOME/.bash_profile [ . . \$HOME/.bashrc ] \$HOME/.bashrc	\$HOME/.bashrc