



Operativni sistemi 2011

Osnove Linux-a

Shell scripts

Katedra za računarstvo
Elektronski fakultet u Nišu

Prof. dr Dragan Stojanović
mr Aleksandar Stanimirović
mr Bratislav Predić



Sadržaj

- Šta je shell script?
- Kreiranje shell script-a
- Sistemske promenljive
- Korisničke promenljive
- Pozicioni i specijalni parametri
- Komande
- Programske strukture



Sadržaj

- Šta je shell script?
- Kreiranje shell script-a
- Sistemske promenljive
- Korisničke promenljive
- Pozicioni i specijalni parametri
- Komande
- Programske strukture



Šta je shell script?

Shell script

- Shell script je **script program** napisan tako da se može izvršavati u shell-u operativnog sistema.
- Tipične operacije koje shell script omogućava uključuju:
 - ▶ **manipulaciju datotekama**
 - ▶ **izvršavanje komandi i izvršnih datoteka**
 - ▶ **štampanje tekstualnih poruka**
- Shell script je **tekstualna datoteka** koja omogućava **paketno izvršavanje** komandi Linux operativnog sistema.
- Interpretator komandi izvršava jednu po jednu komandu iz shell script datoteke.
- Namena: **automatizuju najčešće izvršavane operacije.**
- Na Unix/Linux okruženju postoji veliki broj različitih komandnih interpretatora.
- **Bourne SHell (SH)** predstavlja najrasprostranjeniji komandni interpretator za UNIX/Linux okruženje.



Šta je shell script?

Prednosti

- **Jednostavnost** programa.
- Shell script se **razvija brže** nego ekvivalentan program u nekom standardnom programskom jeziku.
- **Nema prevođenja** koda već se shell script **interpretira**.
- Omogućava **jednostavno povezivanje** već postojećih aplikacija i komandi.
- **Interaktivno** otklanjanje grešaka.

Mane

- **Greške** mogu da dovedu do katastrofalnih posledica.
- **Loše performanse** prilikom izvršavanja shell script-ova.
- **Ograničenost** script jezika.
- Problem **kompatibilnosti**.



Sadržaj

- Šta je shell script?
- Kreiranje shell script-a
- Sistemske promenljive
- Korisničke promenljive
- Pozicioni i specijalni parametri
- Komande
- Programske strukture



Kreiranje shell script-a

1. korak – Kreiranje tekstualne datoteke

- Shell script program je tekstualna datoteka.
- Za kreiranje se koristi tekst editora (vi, pico, emacs, joe, ...).
- **Datoteci je potrebno dodeliti ekstenziju sh.**

2. korak – Pisanje programa

```
#  
# pozdrav.sh  
#  
clear  
echo "Pozdravna poruka"
```



Kreiranje shell script-a

3. korak – Privilegije za izvršavanje

- Potrebno je postaviti privilegije za izvršavanje i čitanje.

```
$ chmod +rx script_name  
$ chmod 755 script_name
```

4. korak – Izvršavanje programa

- **sh script_name <argumenti>**

```
$ sh pozdrav.sh poruka 1 23.34
```

- **./script_name <argumenti>**

```
$ ./pozdrav.sh poruka 1 23.34
```




Kreiranje shell script-a

5. korak – Lokacija shell script programa

- Shell script datoteke koje se nalaze u tekućem direktorijumu se izvršavaju bez problema.
- Moguće navođenje putanje (apsolutne ili relativne) do script datoteke.
`$sh /home/neki_user/pozdrav.sh`
- Korišćenjem bin direktorijuma (**PATH promenljiva**)
- Redosled traženja:
 - 1.interne komande
 - 2.tekući direktorijum,
 - 3.direktorijumi navedeni u PATH promenljivoj



Kreiranje shell script-a

Elementi shell script programa

- Osnovni elementi shell script programa su:
 - ▶ unapred definisane (sistemske) promenljive
 - ▶ promenljive definisane od strane korisnika
 - ▶ pozicioni parametri
 - ▶ specijalne promenljive
 - ▶ programske strukture



Sadržaj

- Šta je shell script?
- Kreiranje shell script-a
- **Sistemske promenljive**
- Korisničke promenljive
- Pozicioni i specijalni parametri
- Komande
- Programske strukture



Sistemske promenljive

Definicija

- **Unapred definisane promenljive** čiju vrednost održava operativni sistem.
- Po pravilu ime se sastoji samo od velikih slova.

Promenljiva	Značenje
COLUMNS =80	broj kolona na ekranu
HOME =/home/vivek	putanja do home direktorijum
LINES =25	broj linija na ekranu
LOGNAME =students	datoteka za logovanje
OSTYPE =Linux	tip OS-a
PATH =/usr/bin:/sbin:/bin:/usr/sbin	promenljiva PATH
PS1 =[\u@\h \W]\\$	definicija prompt-a
PWD =/home/students/Common	radni direktorijum
SHELL =/bin/sh	ime shell-a
USERNAME =vivek	Korisnik koji je trenutno ulogovan



Korisničke promenljive

- Šta je shell script?
- Kreiranje shell script-a
- Sistemske promenljive
- **Korisničke promenljive**
- Pozicioni i specijalni parametri
- Komande
- Programske strukture



Korisničke promenljive

Imena

- Počinje alfanumeričkim karakterom ili donjom crtom iza koje slede alfanumerički karakteri.

HOME, SYSTEM_VERSION, no1, v1ar

- Ne treba koristiti **specijalne znake** (?, *, #) za imena promenljivih
- Imena su **case-sensitive**

No, no, nO, NO



Korisničke promenljive

Definisanje i dodeljivanje vrednosti

- Korisničke promenljive se ne moraju posebno deklarirati. Prvo pojavljivanje korisničke promenljive automatski predstavlja deklaraciju.

```
no=10
```

```
ime="sistemski softver"
```

- Prilikom dodeljivanja vrednosti **ne smeju da postoje blanko znaci** ispred i iza znaka jednako (=)

```
no=10
```

```
no = 10 (error)
```

```
no= 10 (error)
```

```
no =10 (error)
```

- NULL vrednost

```
nesto=
```

```
nesto=""
```



Korisničke promenljive

Korišćenje promenljive

- Promenljiva se referencira navođenjem simbola \$ ispred imena.

echo \$no

echo \$ime

- Promenljiva kojoj prethodno nije dodeljena vrednost ima podrazumevanu vrednost NULL.

```
MY_MESSAGE="Hello World"
```

```
MY_SHORT_MESSAGE=hi
```

```
MY_NUMBER=1
```

```
MY_PI=3.142
```

```
MY_OTHER_PI="3.142"
```

```
MY_MIXED=123abc
```




Sadržaj

- Šta je shell script?
- Kreiranje shell script-a
- Sistemske promenljive
- Korisničke promenljive
- Pozicioni i specijalni parametri
- Komande
- Programske strukture



Pozicioni i specijalni parametri

Pozicioni parametri

- Pozicioni paramtri su **ulazni parametri** (argumenti) koji se prosleđuju shell script programu prilikom njegovog izvršavanja.
- \$1, \$2, ... – način obraćanja pozicionim parametrima (**broj pozicionog parametra odgovara rednom broju argumenta**).
- Naredba **set** omogućava programsku dodelu vrednosti pozicionim parametrima
set p1 p2 p3 ... – efekat je \$1=p1, \$2=p2,...

Pozicioni parametri

- \$0 – ime pozvanog shell script programa
- \$# - broj pozicionih parametara
- \$@, \$* - nadovezane vrednosti svih parametara
- \$? - povratni kod prethodnog procesa
- \$\$ - id tekuceg procesa



Pozicioni i specijalni parametri

```
#pozpar.sh
echo "Pozvan sam sa $# parametara"
echo "Moje ime je $0"
echo "Moj prvi parametar je $1"
echo "Moj drugi parametar je $2"
echo "Vrednost svih parametara je $@"
```

```
$ /home/osistemi/pozpar.sh
Pozvan sam sa 0 parametara
Moje ime je /home/osistemi/pozpar.sh
Moj prvi parametar je
Moj drugi parametar je
Vrednost svih parametara je
```

```
$ /home/osistemi/pozpar.sh hello world earth
Pozvan sam sa 3 parametara
Moje ime je /home/osistemi/pozpar.sh
Moj prvi parametar je hello
Moj drugi parametar je world
Vrednost svih parametara je hello world earth
```



Sadržaj

- Šta je shell script?
- Kreiranje shell script-a
- Sistemske promenljive
- Korisničke promenljive
- Pozicioni i specijalni parametri
- Komande
- Programske strukture



Komande

Izdavanje komandi

- **Jednostavne komande** (! - Komanda vraća status sa kojim je završena)

[!] command

- Komande povezane pipeline-om (|)

[!] command1 [| command2...]

- **Grupa komandi**

command;command2 – izvršavaju se jedna za drugom

command || command2 – druga se izvršava samo ako se prva komanda nije izvršila

command && command2 – druga se izvršava samo ako se izvršila i prva komanda

command;command2 & – izvršavaju se u pozadini



Komande

Lista komandi

- (lista_naredbi) – naredbe iz liste se **izvršavaju u subshell-u**
- {lista_naredbi} – naredbe iz liste se **izvršavaju kao grupa**

```
# subshell
# stampa na ekranu
# 5
# 6
ime=6
(ime=5; echo $ime)
echo $ime
```

```
# subshell
# stampa na ekranu
# 5
# 5
ime=6
{ime=5; echo $ime}
echo $ime
```



Komande

echo

- Naredba **echo** ispisuje liniju teksta na ekranu.

echo [string]

- **string** tekst koji se ispisuje na ekranu

```
#stampa.sh
echo "Hello  World"
echo "Hello World"
echo "Hello * World"
echo Hello * World
echo Hello  World
echo "Hello" World
echo Hello "  " World
echo "Hello \"*\" World"
echo `hello` world
echo 'hello' world
```



Komande

read

- Naredba **read** učitava vrednost sa standardnog ulaza i inicijalizuje promenljivu

read <promenljiva>

- **promenljiva** – korisnička promenljiva u koju se smešta učitana vrednost

```
#ucitavanje.sh
echo What is your name?
read MY_NAME
echo "Hello $MY_NAME"
```




Komande

shift

- Naredba **shift** pomera vrednosti pozicionih parametara

shift [n]

- **n** – broj pozicionih argumenata koji se pomeraju ($n \leq \$ \#$)
- Pozicioni parametri $\$1, \dots, \$\#-n$ **preuzimaju vrednosti** pozicionih parametara $\$n+1, \dots, \$\#$. Pozicioni parametri $\$n+1, \dots, \$\#$ **gube svoju vrednost**.



Sadržaj

- Šta je shell script?
- Kreiranje shell script-a
- Sistemske promenljive
- Korisničke promenljive
- Pozicioni i specijalni parametri
- Komande
- Programske strukture



Programske strukture

IF ... THEN ... ELSE ... FI

- Naredba uslovnog grananja definiše tok izvršavanja tok shell script programa.

if tests

then commands

[elif tests

then commands]

[else commands]

fi

- **tests** – lista komandi čija se vrednost ispituje. Uslov je **ispunjen** ako je **vrednost različita od nule**.

- Ključne reč **if** i **then** moraju se nalaziti u **različitim linijama koda**. Ukoliko se nalaze u istoj liniji između njih mora da se nađe ; .



Programske strukture

Testiranje

- Za testiranje vrednosti izraza koristi se naredba **test**.
- Naredba **test** se često **ne poziva direktno** već se koristi **[]**.
- **[]** su simbolički linkovi ka naredbi **test** koji olakšavaju razumevanje programa.
- Operatori testiranja:
 - **!expr** – ispituje da li izraz ima vrednost FALSE
 - **expr1 -a expr2** – dva izraza povezana logičkim operatorom AND
 - **expr1 -o expr2** – dva izraza povezana logičkim operatorom OR
 - **-z str** – ispituje da li je string str dugačak 0B odnosno da li se radi o praznom stringu
 - **-n str** – ispituje da li string str ima dužinu veću od 0B odnosno da nije prazan string
 - **s1 = s2** – ispituje jednakost stringova s1 i s2
 - **s1 != s2** – ispituje nejednakost stringova s1 i s2



Programske strukture

Testiranje

● Operatori testiranja:

- **-d adr** – ispituje da li je adr direktorijum
- **-f file** – ispituje da li je file datoteka
- **-r file** – ispituje da li korisnik ima privilegiju za čitanje
- **-w file** - ispituje da li korisnik ima privilegiju za pisanje
- **-x file** - ispituje da li korisnik ima privilegiju za izvršavanje
- **-s file** – ispituje da li je veličina datoteka file veća od 0KB
- **expr1 -eq|-ne |-lt|-le |-gt|-ge expr2** - poređenje aritmetičkih izraza expr1 i expr2 (=, <>, <, <=, >, >=)



Programske strukture

```
# if struktura
if test $ime = ja
then
    echo Ove dve vrednosti su jednake
else
    ime = ja
fi
```

```
# if struktura
if [ "$X" -lt "0" ]
then
    echo "X je manje od nule"
fi
```

```
# if struktura
if [ "$X" -lt "0" ]; then
    echo "X je manje od nule"
fi
```

```
# if struktura
if [ "$X" -lt "0" ]; then
    echo "X je manje od nule"
fi
if [ "$X" -gt "0" ]; then
    echo "X je vece od nule"
fi
[ "$X" -le "0" ] && \
    echo "X je manje ili jednako nuli"
[ "$X" -ge "0" ] && \
    echo "X je vece ili jednako nuli"
[ "$X" = "0" ] && \
    echo "X je string ili broj \"0\""
[ "$X" = "hello" ] && \
    echo "X je jednak stringu \"hello\""
[ "$X" != "hello" ] && \
    echo "X nije string \"hello\""
[ -n "$X" ] && \
    echo "X is ima dužinu veću od nule"
[ -f "$X" ] && \
    echo "X je datoteka" || \
    echo " Ne postoji datoteka: $X"
[ -x "$X" ] && \
    echo "X je izvršna datoteka "
```



Programske strukture

CASE ... IN ... ESAC

- Naredba **SELECT** omogućava selektivno izvršavanje naredbi u zavisnosti od usliva.
- Naredba **SELECT** menja veći broj **IF** blokova.

case word in

[[(| pattern [| pattern]...) commands ;;]...

esac

- **word** - izraz čija se vrednost upoređuje sa nizom pattern-a. Pattern može biti definisan korišćenjem džoker znaka.
- **pattern** - šablon koji se poredi sa izrazom (vrednosti u šablonu se odvajaju korišćenjem |)
- **commands** - komande koje se izvršavaju. Lista komandi se mora završiti sa ;;



Programske strukture

```
# case struktura
ime=ja
case $ime in
    ja | ti | on)
        echo $ime = jednina;;
    mi | vi | oni)
        echo $ime = mnozina;;
esac
```

```
# case struktura
echo -n "Unesite ime zivotinje: "
read ANIMAL
echo -n "$ANIMAL ima "
case $ANIMAL in
    konj | pas | macka)
        echo -n "cetiri noge";;
    covek | majmun )
        echo -n "dve noge";;
    *)
        echo -n "nepoznat broj nogu";;
esac
```




Programske strukture

FOR ... IN ... DO ... DONE

- Naredba **FOR** omogućava izvršavanje naredbi u petlji.

for name [in word_list] do commands done

for ((expr1 ; expr2 ; expr3)) do commands done

- **name** brojač petlje koji uzima vrednosti iz liste
- **word_lista** - stringovi čije vrednosti uzima brojač; ukoliko se ne navede podrazumeva se lista pozicionih parametara
- **commands** - naredbe koje se izvršavaju u petlji
- **expr1** - aritmetički izraz koji se evaluira pre ulaska u petlju
- **expr2** - aritmetički izraz koji se evaluira dok njegova vrednost ne postane 0. Kad god je expr2 različit od 0 telo petlje se ponavlja.
- **expr3** - aritmetički izraz koji se evaluira kad go je expr2 različit od 0

- Petlja vraća status poslednje komande iz niza ili FALSE ako je neki od izraza petlje nije validan.



Programske strukture

```
#prvi primer  
for ime in aca mika zika  
do  
    echo $ime  
done
```

```
#treci primer  
for i in $(ls );  
do  
    echo item: $i  
done
```

```
# drugi primer  
set ja ti on mi vi oni  
for ime  
do  
    echo $ime  
done
```

```
# cetvrti primer  
for i in /usr/docs/*  
do  
    echo $i  
done
```



Programske strukture

WHILE ... DO ... DONE UNTIL ... DO ... DONE

● Naredba **WHILE** i **UNTIL** omogućava izvršavanje naredbi u petlji.

while tests do commands done

until tests do commands done

- **tests** - uslov koji se evaluira
- **commands** - komande koje se izvršavaju u telu petlje

```
#while  
while test $# -ne 0  
do  
    echo $1$2$3$4$5  
    shift  
done
```