System operacyjny Linux

Paweł Rajba

pawel.rajba@continet.pl
http://kursy24.eu/

Zawartość modułu 6

- Język bash
  - Pierwszy skrypt
  - Rozwinięcia parametryczne
  - Bloki instrukcji
  - Dwa przydatne polecenia
  - Tablice
  - Sprawdzanie warunków
  - Instrukcje sterujące
  - Funkcje

Pierwszy skrypt

- Tworzymy plik z rozszerzeniem .sh i nadajemy mu prawa do wykonywania.
- W pierwszym wierszu wpisujemy #!/bin/bash
- Do skryptu możemy przekazać parametry wywołania:
  - $0 – nazwa skryptu,
  - $1, $2, $3, ..., $9 – kolejne parametry,
  - shift – przesuwa parametry o jeden w lewo
  - $# – liczba parametrów,
  - $* – wszystkie parametry w postaci listy.

Przykład

```
#!/bin/bash
s=0;
for i in $*
do
  s=$[$s+$i];
done
echo $s;
```


Rozwinięcia parametryczne

- Składnia
  - ${parametr:-domyslna} — jeśli parametr jest pusty, to przyjmie wartość domyślną,
  - ${#parametr} — długość parametru,
  - ${parametr%%slowo} — parametr, od którego odcięto z tyłu najdłuższy fragment pasujący do słowa,
  - ${parametr%slowo} — parametr, od którego odcięto z tyłu najkrótszy fragment pasujący do słowa,

Rozwinięcia parametryczne

- Składnia c.d.
  - ${parametr##slowo} — parametr, od którego odcieto z przodu najdłuższy fragment pasujący do słowa,
  - ${parametr#slowo} — parametr, od którego odcięto z przodu najkrótszy fragment pasujący do słowa.
- Przykłady
  - `x="a.b.c.d.e"`
  - `echo ${#x} ${x%%.*} ${x%.*} ${x##*.} ${x#*.}`

Rozwinięcia parametryczne

- Składnia c.d.
  - ${parametr##slowo} — parametr, od którego odcieto z przodu najdłuższy fragment pasujący do słowa,
  - ${parametr#slowo} — parametr, od którego odcięto z przodu najkrótszy fragment pasujący do słowa.
- Przykłady
  - `x="a.b.c.d.e"`
  - `echo ${#x} ${x%%.*} ${x%.*} ${x##*.} ${x#*.}`
- Przykłady c.d.
  - 9 a a.b.c.d e b.c.d.e

Bloki instrukcji

- Składnia
  - (lista) — uruchamia listę instrukcji w nowej powłoce,
  - { lista; } — uruchamia listę instrukcji w bieżącej powłoce,
  - ((wyrażenie)) — wyrażenie jest obliczane zgodnie z zasadami wyrażeń arytmetycznych.

Dwa przydatne polecenia

- read – czyta napis z klawiatury
  - Składnia
    - read [opcje] [zmienna]
  - Opcje
    - -d separator – kończy wczytywanie po pojawieniu się znaku separator (niekoniecznie znaku nowej linii),
    - -n liczba – kończy wczytywanie po liczba znaków,
    - -p prompt – ustawia znak zachęty na prompt,
    - -t timeout – określa, po jakim czasie wczytywanie zostanie przerwane z błędem.
  - Jeśli nie podamy zmiennej, wczytany napis zostanie przekazany do zmiennej REPLY

Dwa przydatne polecenia

- printf – formatowanie napisów
  - Składnia
    - printf format [argumenty]
  - Opcje formatu
    - %d, %i – liczby całkowite,
    - %x – liczby całkowite w postaci szesnastkowej,
    - %f, %F – liczby zmiennoprzecinkowe,
    - %e, %E – liczby zmiennoprzecinkowe w postaci wykładniczej,
    - %s – napis,
    - %c – pojedynczy znak.

Tablice

- Przykład
  - `roadrunner :~ # x[10]=5`
  - `roadrunner :~ # echo ${x[10]}`
  - `5`
  - `roadrunner :~ # x['dwa']=2`
  - `roadrunner:~ # echo ${x['dwa']}`
  - `2`

Sprawdzanie warunków

- test – polecenia do sprawdzania warunków
  - Składnia
    - test wyrazenie
    - [wyrazenie]
  - Operatory liczbowe
    - e1 -eq e2 — prawdziwe wtw., gdy *e*1 = *e*2,
    - e1 -ge e2 — prawdziwe wtw., gdy *e*1 >= *e*2,
    - e1 -gt e2 — prawdziwe wtw., gdy *e*1 > *e*2,
    - e1 -lt e2 — prawdziwe wtw., gdy *e*1 < *e*2,
    - e1 -le e2 — prawdziwe wtw., gdy *e*1 <= *e*2,
    - e1 -ne e2 — prawdziwe wtw., gdy *e*1 <> *e*2.

Sprawdzanie warunków

- test c.d.
  - Operatory plikowe
    - -d plik — prawdziwe wtw., gdy plik istnieje i jest katalogiem,
    - -e plik — prawdziwe wtw., gdy plik istnieje,
    - -f plik — prawdziwe wtw., gdy plik istnieje i jest regularny,
    - -r plik — prawdziwe wtw., gdy plik istnieje i jest z prawem do czytania,
    - -w plik — prawdziwe wtw., gdy plik istnieje i jest z prawem do pisania,
    - -x plik — prawdziwe wtw., gdy plik istnieje i jest z prawem do wykonywania.

Sprawdzanie warunków

- test c.d.
  - Operatory tekstowe
    - s1 = s2 — prawdziwe wtw., gdy teksty s1 i s2 są równe,
    - s1 != s2 — prawdziwe wtw., gdy teksty s1 i s2 są rózne,
    - [-n] s — prawdziwe wtw., gdy długość s jest wiekszą niż 0,
    - -z s — prawdziwe wtw., gdy długość s jest równa 0.
  - Operatory logiczne
    - !w — operator NOT (zaprzeczenie),
    - w1 -a w2 — operator AND,
    - w1 -o w2 — operator OR.

Instrukcje sterujące

- if – instrukcja warunkowa
  - Składnia
    - if warunek; then lista; [elif warunek; then lista;] ...
      [else lista;] fi
  - Przykład
    - ```
      if [ -f "$plik" ]
      then
        echo "Jest pliczek" $plik
      else
        echo "Nie ma, " $plik "chlip"
      fi
      ```
    - ```
      if [ "$haslo" == "tajne" ]; then echo „Hi"; fi
      ```


Instrukcje sterujące

- case – instrukcja wyboru
  - Składnia
    - case $zmienna in
      [wzorzec [| wzorzec] ...) lista ;;] ...
      esac

Instrukcje sterujące

- case c.d.
  - Przykład
    - ```
      case "$dzien" in
          6 | "sat")
            echo "Jest sobota.";
          ;;
          0 | "sun")
            echo "Jest niedziela.";
          ;;
          *)
            echo "To nie jest dobry dzien.";
          ;;
      esac
      ```


      ```

# Instrukcje sterujące

- while – pętla
  - Składnia
    - while warunek; do list; done
  - Przykład
    - ```
      i=0
      while [ $i -le 10 ]
      do
        echo $i
        i=$[$i+1]
      done
      ```


      ```

# Instrukcje sterujące

- for – pętla
  - Składnia
    - for nazwa [ in lista ] ; do lista ; done
    - for (( wyr1 ; wyr2 ; wyr3 )) ; do lista ; done
  - Przykłady
    - `for i in .*; do echo $i; done`
    - ```
      for ((i=0;$i<10;i++))
      do
        echo $[$i%3]
      done
      ```


      ```

# Instrukcje sterujące

- break i continue – instrukcje
  - Składnia
    - break [n]
    - continue [n]
  - Opcje
    - n – określa poziom wyjścia

 

# Funkcje

- Kilka uwag
  - funkcje można wywoływać z parametrami
    np. f 1 2 3 4,
  - dostęp do parametrów wewnątrz funkcji jest taki,
    jak w przypadku skryptów, czyli poprzez $1, ..., $9,
  - instrukcja return przekazujemy wartość funkcji,
  - możemy stosować rekurencje.

 

# Funkcje

- Składnia
  - ```
    [function] nazwa() {
      polecenia;
    }
    ```
- Przykład
  - ```
    #!/bin/bash
    function silnia_aux() {
      if [ "$1" -eq "1" ]
      then
        echo "$2";
      else
        silnia_aux $[$1 - 1] $[$2 * $1];
      fi
    }
    ```


Przykłady

- tabliczka.sh
- pytaj.sh
- sortuj.sh

