Oznacz jako wykonane

Zapis zmiennych

Na początek trzeba wspomnieć, że aby odczytać zawartość zmiennej w bashu można użyć po prostu zapisu z dolarem na początku: `$VAR`, ale można to zrobić również w postaci `${VAR}`. Taki zapis bywa wygodniejszy niż zwykły, bo na przykład umożliwia bezproblemowe włączanie zmiennej do ciągu znaków np. dodanie rozszerzenia do zawartej w zmiennej nazwy pliku: `${VAR}.txt`. Taki zapis jest również konieczny jeśli chce się wykorzystać moc ukrytą w bashowych zmiennych.

Modyfikacje opisane w tym tekście odbywają się „w locie” – można dokonać zamiany fragmentów, przyciąć tekst do wymaganej długości, zmienić wielkość liter itp. Operacje te nie zmieniają zawartości zmiennej, można więc wielokrotnie wracać do jej początkowej wersji (np. podanej przez użytkownika).

Poniżej przedstawiono kilka najciekawszych przekształceń zmiennych bashowych. Nie są to wszystkie ich możliwości, więcej na ten temat można się dowiedzieć z doskonałego podręcznika Advanced Bash Scripting Guide

Wartość domyślna

Zmienna może mieć wartość domyślną. Oznacza to, że jeśli w momencie jej odczytu jest pusta lub jest równa null to zostanie użyta właśnie ta podana jako domyślna.

Składnia:

```
echo ${VAR-wartość}  # dla pustej zmiennej
echo ${VAR:-wartość} # dla zmiennej równej null
```

Jako wartości domyślnej można użyć… innej zmiennej, która uprzednio zostanie zdefiniowana lub wyliczona:

```
$ DEFAULT="To jest domyślna wartość"
$ echo ${VAR-$DEFAULT}
```

Przykład: podstawianie domyślnego parametru w skrypcie pod zmienną np. $1:

```
filename=${1:-plik_wejsciowy.txt}
```

Jeśli użytkownik nie poda nazwy pliku w pierwszym parametrze to zostanie użyta domyślna, w tym przypadku “plik_wejsciowy.txt”.

Komunikat błędu jeśli zmienna jest pusta

Ten sposób przydaje się najbardziej w skryptach. Jeśli zmiennej nie została przypisana wartość to można wyświetlić komunikat o błędzie. Skrypt jest przerywany (kończy się z kodem 1).

Składnia:

```
${VAR?"komunikat"}
```

Przykład: wymuszenie podania pierwszego parametru:

```
VAR=${1?"Użycie skryptu: $0 <nazwa pliku>"}
```

Jeśli nie podano argumentu (zmienna `$1` jest pusta) to jest wyświetlany komunikat o błędzie i skrypt kończy działanie. Warto zwrócić uwagę, że w komunikacie można użyć zmiennych specjalnych – w tym przypadku `$0`, oznaczającą nazwę skryptu.

Długość zmiennej

Bash umożliwia sprawdzenie długości ciągu przechowywanego w zmiennej.

Składnia:

```
echo  ${#VAR}
```

Przykład: wymuszenie podania pierwszego parametru – sposób drugi.

```
if [[ ${#1} -eq 0 ]]
then
      echo "pierwszy parametr jest wymagany"
      exit 1
fi
```

Jeśli użytkownik nie poda w ogóle parametru to `${#VAR}` zwróci wartość 0 (co jest jeszcze jednym sposobem na sprawdzenie, czy w ogóle został podany wymagany parametr).

Sprawdzenie długości przydaje się jeśli wiemy np., że użytkownik powinien podać nazwę pliku z rozszerzeniem czyli zmienna powinna mieć co najmniej 5 znaków.

Przypadki szczególne:

- dla tablic `${#TABLICA}` podaje liczbę elementów, a nie długość ciągu znaków,
- `${#*}` oraz `${#@}` zwracają liczbę argumentów, którą podano przy uruchamianiu skryptu (przykład wykorzystania został opisany w artykule o automatycznym helpie w skryptach bashowych).

Usunięcie fragmentu ciągu znaków

Ten sposób jest najczęściej używany, by manipulować ścieżkami i nazwami plików – np. by usunąć rozszerzenie z nazwy pliku bądź wydobyć nazwę pliku z pełnej ścieżki. We wzorcu można użyć symbolu * na oznaczenie dowolnego ciągu znaków.

Składnia:

```
${VAR#wzorzec}   #Usuwa najkrótszy fragment pasujący do wzorca na początku zmiennej.
${VAR##wzorzec}  #Usuwa najdłuższy fragment pasujący do wzorca na początku zmiennej.
${var%Pattern}   #Usuwa najkrótszy fragment pasujący do wzorca na końcu zmiennej.
${var%%Pattern}  #Usuwa najdłuższy fragment pasujący do wzorca na końcu zmiennej
```

Przykład: obcięcie ścieżki i pozostawienie wyłącznie nazwy pliku:

```
$ VAR=$(readlink -f test.sh )
$ echo ${VAR}
/home/piotr/tmp/test.sh
$ echo ${VAR##*/}
test.sh
```

Przykład: obcięcie rozszerzenia z nazwy pliku

```
$ VAR="nazwa_pliku.txt"
$ echo ${VAR%%.*}
nazwa_pliku
```

Wybranie fragmentu (substring)

W przypadku, kiedy znana jest struktura ciągu znaków, np. ścieżka, która zawsze zaczyna się od katalogu domowego użytkownika można wybrać tylko określony fragment ciągu, omijając niepotrzebną resztę.

Składnia

```
${VAR:n} #Wartość zmiennej począwszy od znaku n
${VAR:n:x} #Wartość zmiennej, zaczynająca się od znaku n, o długości x znaków.
```

Przykład: wycięcie znanego początku ze ścieżki

```
$ VAR=$(pwd)
$ echo $VAR
/home/piotr/tmp
$ echo ${VAR:12}
tmp
```

Przykład: ograniczenie długości ciągu do 20 znaków

```
echo ${VAR:0:20}
```

Warto zwrócić uwagę, że znaki są liczone od 0, a nie od 1.

Zamiana fragmentu ciągu znaków

Możliwość zamiany części ciągu znaków często przydaje się przy manipulowaniu plikami.

Jeśli zamiennik nie zostanie podany to wzorzec jest zamieniany na pusty ciąg – czyli de facto po prostu usuwany.

Składnia:

```
${VAR/wzorzec/zamiennik}  #zamiana pierwszego wzorca na zamiennik
${VAR//wzorzec/zamiennik} # zamiana wszystkich ciągów pasujących do wzorca na zamiennik.
```

Przykład: konwersja wszystkich obrazków w danym katalogu z jpg na png za pomocą Imagemagick:

```
for i in *.jpg
do
   convert "$i" "${i/.jpg/.png}"
done
```

Zamiana wielkości liter

Zmiana wielkości liter może być przydatna przy porównywaniu ciągów znaków np. by uniezależnić się od tego, czy użytkownik poda parametr wielkimi czy małymi literami, a także przy konstruowaniu zunifikowanych ciągów znaków służących za identyfikatory.

Składnia:

```
${VAR,,}        #zamień wszystkie litery na małe
${VAR^^}        # zamień wszystkie litery na duże
${VAR~~}        # zamienia wszystkie duże litery na małe i odwrotnie
${VAR,,wzorzec} #zamień litery na małe, ale tylko we fragmentach pasujących do wzorca
${VAR^^wzorzec} #zamień litery na duże, ale tylko we fragmentach pasujących do wzorca
```

Przykład:

```
$ VAR="Ala ma Kota, a Kot ma Alę"
$ echo ${VAR,,}
ala ma kota, a kot ma alę
$ echo ${VAR^^}
ALA MA KOTA, A KOT MA ALĘ
$ echo ${VAR,,A}
ala ma Kota, a Kot ma alę
$ echo ${VAR,,K}
Ala ma kota, a kot ma Alę
$ echo ${VAR~~}
aLA MA kOTA, A kOT MA aLĘ
```

Zadanie do samodzielnej realizacji:

Napisz skrypt: (zadanie można zrobić w wersji uproszczonej bez rozszerzenia .ext)

- zamieniający w katalogu bieżącym nazwy plików wg schematu: zamieniając kolejność tego, co jest przed myślnikiem z tym, co jest za myślnikiem, tj: part1-part2.ext na part2-part1.ext
- obcinający nazwę pliku po pierwszym wystąpieniu znaku -, tj dla part1-part2.ext zwróci part1.ext
- zamieniający w katalogu bieżącym nazwy plików wg schematu: part1-part2-part3.ext na part2_part3-part1.ext

Źródło: Piotr Parafiniuk - Tajemnice zmiennych w bashu cz.1: modyfikacje

Ostatnia modyfikacja: poniedziałek, 2 stycznia 2023, 22:55