Shell Scripting

Писане на shell скриптове

Какво е Shell?

- Команден интерпретатор
- Програмата, част от Линукс, която получава и изпълнява команди
- Има различни видове shell-ове:
  - bash
  - zsh
  - Tcsh
  - Ksh
  - Fish

Какво е shell скриптирането?

- Shell скриптовете позволяват да:
  - Изпълняваме команди от файл
  - Автоматизираме действия чрез поредица от команди
  - Изпълняваме команди в определен момент
  - Използваме преимуществата на Linux

Кой има нужда от shell скриптирането?

- Системни администратори
- Разработчици
- DevOps кадри
- Напреднали потребители

За какво е удобно да ползваме shell скриптове?

- Работа с файлове
- Изпълнение на програми
- Обработка на текст (с помощта на инструменти като grep, sed и др.)
- Още много други неща...

... Но не винаги shell скриптовете са универсално решение - не е удобен за изчисления, работа с двоична информация и графични елементи

Създаване на shell скрипт

- Shell скриптовете се пишат идентично на познатите Линукс команди.
- В рамките на Линукс командите спокойно може да се съдържат променливи, условни конструкции и цикли.
- Shell скриптовете трябва да имат статут на изпълними файлове
    - Има някои особености при тяхното създаване и извикване.

Hello World!

- Отворете удобен текстов редактор и въведете в него следното парче код:

```
#!/bin/bash
echo "Hello world!"
```

- Запазете файла като **hello.sh**
- Опитайте се да изпълните файла през

```
petar@petar-Precision-M4800:~/shell$ hello.sh
```

Ами сега?

- Навярно сте забелязали, че получавате съобщение за грешка:

```
petar@petar-Precision-M4800:~/shell$ hello.sh
hello.sh: command not found
```

- Причината е, че изпълнимите файлове се стартират по различен начин, правилното е да се поставя ./ пред тяхното име:

```
$ ./hello.sh
```

Ами сега... Отново?

- Пак грешка!

```
petar@petar-Precision-M4800:~/shell$ ./hello.sh
bash: ./hello.sh: Permission denied
```

- Линукс изисква да зададем специално право за изпълнение на новосъздадения файл. Нека да ползвам `chmod +x hello.sh`

- Сега я изпълнете отново, както е показано на първата снимка горе :)

Environment variables в Линукс

- Променливите на средата се използват от shell-а, shell скриптовете и ОС за различни цели.
- Някои често използвани environment variables в Линукс:
  - HOME
  - HOSTNAME
  - PATH
  - LANG
  - DISPLAY
  - И др.
- Можете и сами да задавате или редактирате такива променливи

PATH променливата

- Указва в кои директории shell може да търси изпълними файлове, които да съответстват на команди (списък от директории)
- За всяка команда съответства изпълним файл
- Прави изпълнението по-удобно и по-сигурно.
- Една от най-важните environment variables

Практически насоки за изпълнение на shell скриптове

- Ако скрипта не се намира в директория добавена в PATH:
  - Извиквайте скрипта със съответната директория или я задайте като текуща директория чрез командата cd преди извикване.
- Ако скрипта се намира в директория добавена в PATH:
  - Извикайте директно скрипта, все едно е команда
- Съвет: Създайте bin директория в домашната си папка и поставяйте там shell скриптовете
  - Добавете директорията като част от PATH

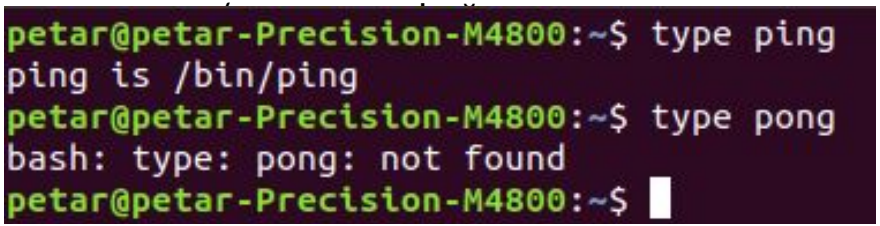
Shebang

- Първия ред от всички shell скриптове
- Започва с #!
- Служи за указване кой интерпретатор да се използва
- Задава опции за интерпретатора

- При bash:
  #!/bin/bash

Какви имена да задаваме на скриптовете?

- Задавайте смислени и значещи имена.
- Избягвайте имена на вече съществуващи команди или на други скриптове.
  - Може да използвате командата type, за да разберете дали дадено име съответства на вече съществуващ



- ~~Как бихме си кръстили скрипта - ping или pong?~~

Променливи в shell

- Като променливите от програмирането :)
- Създават се подобно на Python:
  - име=стойност
  - **ВАЖНО: Не бива да поставяте интервали между името и стойността!**
- Ако стойността на променливата съдържа интервал, то трябва да я оградите с кавички.
- Примери:
  - x=10
  - message=”Welcome back!”

Извличане на стойност от променлива

- За да получите стойността на дадена променлива, трябва да поставите $ пред името ѝ.
- Често се използва заедно с командата echo или в друго присвояване.
- Пример:
  - message=”Hello, $USER”
  - echo $message

Именуване на променливи

- Само букви, числа и долна черта.
- Първият символ в името трябва да е буква или долна черта.
- Има разлика между големи и малки букви (case-sensitive)
- Избягвайте имена съставени от главни букви - обикновено с главни букви са означени environment variables.
- Добър навик е да използвате малки букви за именуването на вашите променливи при писане на shell скриптове.

Добри навици при ползване на променливи

- Ограждайте променливите си също с кавички.
- Използвайте къдрави скоби, особено ако желаете да направите конкатенация:
  - ${**foo**}bar - ще изкара стойността на променливата **foo** последваната от фразата “bar”
  - $**foobar** - ще изкара стойността на променливата **foobar**
- Използвайте $HOME вместо ~

Въвеждане на стойности от клавиатурата

- read
  - Прочита ред и го запзва в променлива
    - Пример: read var
- Може да изведете подканващо съобщение за потребителя:
  - Пример: read -p “Enter your name: ” name

Дебъгване на shell скриптове

- Понякога в shell скриптовете могат да възникнат доста неприятни ситуации, породени от неправилно изписване, грешно извикване или други грешки.
- Shell не показва особено грешки!
- За да използвате дебъгване на всеки ред от shell скрипта, в първия ред задайте -x опцията:
  #!/bin/bash -x

Дебъгване на част от скрипт

- Ако желаете да дебъгнете само част от скрипт:
  #/bin/bash
  #... код, който **не желаете** да дебъгвате
  #...
  set -x
  #код, който **ще бъде** дебъгнат
  set +x
  #още код, който **НЯМА** да бъде дебъгнат

Благодаря за вниманието

Автор:
П. Р. Петров - преподавател по професионална подготовка по Програмиране в ПГЕЕ “К. Фотинов”, гр. Бургас