# Shell Scripting with Bash

Fun with strings

Reindert-Jan Ekker
http://nl.linkedin.com/in/rjekker/
@rjekker

pluralsight
hardcore developer training

# Overview

- **Parameter Expansion**
  - Remove a pattern
  - Search and replace
  - Default values
- **Pattern matching in a conditional expression**
- **End of options**

# Parameter Expansion

- Allows powerful string manipulation
- ${#var}: length of $var

# Removing a pattern

- **Removing part of a string**
  - ${var#pattern} Removes shortest match from begin of string
  - ${var##pattern} Removes longest match from begin of string
  - ${var%pattern} Removes shortest match from end of string
  - ${var%%pattern} Removes longest match from end of string

- **Pattern matching is like pathname matching with *, ? and []**

- **Example with i="/Users/reindert/demo.txt"**

| | |
|---|---|
| ${i#*/} | Users/reindert/demo.txt |
| ${i##*/} | demo.txt |
| ${i%.*} | /Users/reindert/demo |
| ${i%/*} | /Users/reindert |

# Search and replace

- **Search and Replace**
  - ${var/pattern/string}
  - Substitute first match with string
  - ${var//pattern/string}
  - Substitute all matches with string
- **Anchor your pattern**
  - ${var/#pattern/string} matches beginning of the string
  - ${var/%pattern/string} matches end of the string

# Default values

- **Default value**
  - ${var:-value}
  - Will evaluate to “value” if var is empty or unset
  - ${var-value}
  - Similar, but only if var is unset
- **Assign a default value**
  - ${var:=value}
  - If var was empty or unset, this evaluates to “value” and assigns it to var
  - ${var=value}
  - Similar, but only if var is unset

- **Several more useful Parameter expansions:**
  - http://goo.gl/xRHo3u

# Conditional Expression Patterns

- **==, != operators in [[ .. ]] do pattern matching**
  - == is the same operator as =
  - [[ $var == pattern ]] returns true when $var matches the pattern
  - Pattern matching is like pathname matching with *, ? and []
  - [[ $filename == *.txt ]]

- **Use quotes to force string matching**
  - [[ $var == "[0-9]*" ]] matches the string "[0-9]*"

# Conditional Expression Patterns

- **=~ does regular expression matching**
  - Very powerful
  - Uses POSIX extended regular expressions
- **? matches the token before it 0 or 1 times**
  - [0-9]? will match a single digit or nothing at all
- *** matches the token before it for any number of times**
  - [a-z]* will match any lowercase text or nothing at all
- **+ matches the token before it for one or more times**
  - [0-9]+ will match 1 or more digits
- **^ matches start of string**
- **$ matches end of string**

# End of options

- **Is denoted by --**
  - Supported by many UNIX commands
  - Arguments after this will not be interpreted as options
- **Makes it safe when working with data that starts with a dash**
  - For example with a file called “-l.txt”
  - rm --  -l.txt
  - for i in *.txt; do touch -- $i; done
- **Good habit:**
  - When you use a variable as an argument for a command
  - And the contents of the variable are not under your control
  - Use -- with the commands you call

# Summary

- **Parameter Expansion**
  - Remove a pattern
  - Search and replace
  - Default values
- **Pattern matching in a conditional expression**
- **End of options**