Unix Shells and Scripts – Bash

Volker Kuhlmann
<help....@top.geek.nz>

09 Aug 2005

What is a Unix Shell?

- Command interpreter: keyboard / screen
- Command stream from file: shell script
- Script: interpreted, not translated

Interpreter Invocation

- First line of file starts with:

```
        #! /path/to/program
```

  - this specifies the interpreter program
  - interpreter reads whole file from stdin
  - the file must be executable and readable
- Interpreter defaults to $SHELL, the currently running interactive shell.
- No annoying corporate copyright statements as first line!
- Large choice of interpreters (tcsh, python, perl, TCL, ...)

History

- Sh (Bourne shell)
  - original, bare-bones
- Tcsh (tenex C shell)
  - improvement, C syntax
- Bash (born again shell, GNU)
  - reimplementation of sh to make it usable
  - used by all Linux systems
  - copied all good ideas from tcsh (bash v. 2)
- Others
  - ksh (Korn shell), csh (C shell, preceded tcsh)

Bash / Tcsh

Bash (version 2)

- functions, file descriptors, signal handlers
- no data types: only text
- flow control commands

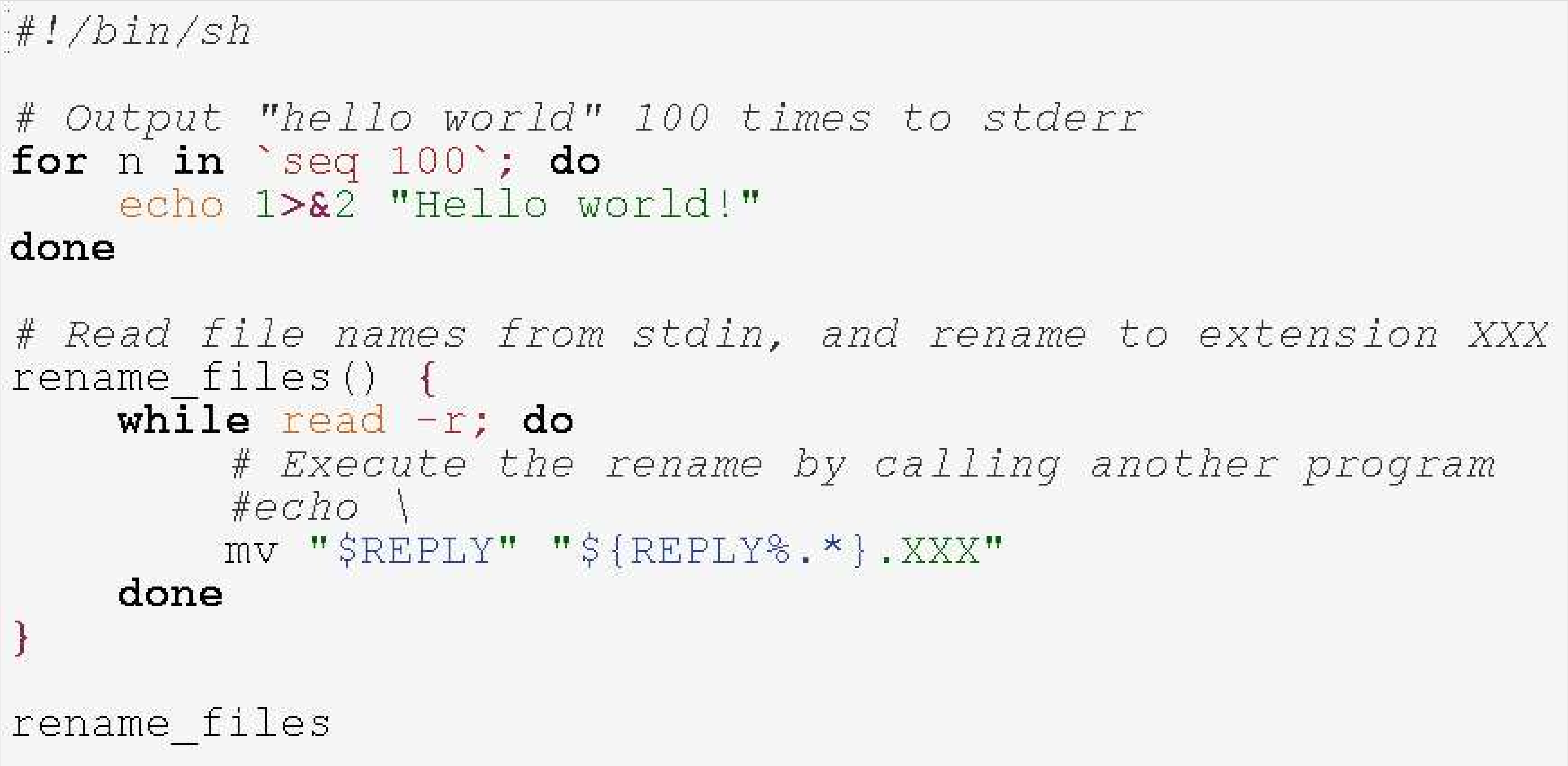
- many things implemented as afterthought

Tcsh

- array variables
- cmd arg completion
- strong alias support
- good interactive shell

- no functions!
- no file descriptors
- weak signal handling

Bash Example



```
#!/bin/sh

# Output "hello world" 100 times to stderr
for n in `seq 100`; do
    echo 1>&2 "Hello world!"
done

# Read file names from stdin, and rename to extension XXX
rename_files() {
    while read -r; do
        # Execute the rename by calling another program
        #echo \
        mv "$REPLY" "${REPLY%.*}.XXX"
    done
}

rename_files
```

Sh/Bash – What is it good for?

- Glue to hold other programs together!
- Automation of smaller jobs – also interactively
- Fast startup time – good for anything small and quick
- System control scripts: /etc/init.d/

Sh/Bash – Limits

- Not for GUI applications
- Not for large applications
  - data handling inefficient (but calling other programs to handle the data is fast!)
  - no binary data
  - however, some larger jobs were written in bash, e.g. lpdfilter (print job filtering)

More Information

- Search for:
  - bash tutorial
  - shell scripting
  - shell programming

  (there is too much material to list here)
- Books