Additional Debugging Tips and Tricks

What You Will Learn

- Variables for debugging
- Manual debugging tips
- Syntax Highlighting
- More Bash built-ins
- File types

Manual Debugging

- You can create your own debugging code.
- Use a special variable like DEBUG
  - DEBUG=true
  - DEBUG=false

```
#!/bin/bash
DEBUG=true
if $DEBUG
then
  echo "Debug mode ON."
else
  echo "Debug mode OFF."
fi
```

```
#!/bin/bash
DEBUG=true
$DEBUG && echo "Debug mode ON."
```

```
#!/bin/bash
DEBUG=false
$DEBUG || echo "Debug mode OFF."
```

```
#!/bin/bash
DEBUG=true
$DEBUG || echo "Debug mode OFF."
```

```
#!/bin/bash
DEBUG="echo"
$DEBUG ls
```

```
#!/bin/bash
#DEBUG="echo"
$DEBUG ls
```

```
#!/bin/bash

debug() {
   echo "Executing: $@"
   $@
}

debug ls
```

Manual Copy and Paste

- Open up a second terminal.
- Copy and paste the commands into the terminal.
- Can be helpful to use "set -x" on the command line.

Syntax Highlighting

- Syntax errors are common.
- Typos, missing brackets, missing quotes, etc.
- Use an editor with syntax highlighting.
  - vi / vim
  - emacs
  - nano
  - gedit
  - kate
  - geany

```
#!/bin/bash

debug() {
  echo "Executing: $@"
  $@
}

debug ls
```

PS4

- Controls what is displayed before a line when using the "-x" option.
- The default value is "+".
- Bash Variables
  - BASH_SOURCE, LINENO, etc

```
PS4='+ $BASH_SOURCE : $LINENO '
```

```
#!/bin/bash -x
PS4='+ $BASH_SOURCE : $LINENO : '
TEST_VAR="test"
echo "$TEST_VAR"
```

```
+ PS4='+ $BASH_SOURCE : $LINENO : '
+ ./test.sh : 3 : TEST_VAR=test
+ ./test.sh : 4 : echo test
test
```

```
#!/bin/bash -x
PS4='+ ${BASH_SOURCE}:${LINENO}:${FUNCNAME[0]}()
'
debug() {
  echo "Executing: $@"
  $@
}
debug ls
```

```
+ ./test.sh:4:debug(): ls
```

DOS vs Linux (Unix) File Types

- CRLF / Carriage Return, Line Feed
- cat -v script.sh

```
#!/bin/bash^M
# This file contains carriage returns.^M
echo "Hello world."^M
```

DOS vs Linux (Unix) File Types

```
#!/bin/bash^M
# This file contains carriage returns.^M
echo "Hello world."^M
```

```
bash: ./test.sh: /bin/bash^M: bad
interpreter: No such file or directory
```

DOS vs Linux (Unix) File Types

- file script.sh
  - script.sh: Bourne-Again shell script, ASCII text executable, with CRLF line terminators
- `dos2unix script.sh`
- file script.sh
  - script.sh: Bourne-Again shell script, ASCII text executable

How does this happen?

- Using a Windows editor and uploading to Linux
  - Some editors can be configured to use just LF
- Pasting from Windows into a Linux terminal
- Pasting from a web browser into a terminal

Summary

- DEBUG variables
- Syntax highlighting
- PS4 and set -x
- File types
  - cat -v
  - dos2unix
  - file