Bash Automation

CS 246

Objectives

- Students will be able to:
  - explain what is meant by a shell
  - understand the advantages of writing shell scripts
  - read and understand shell scripts
  - write basic shell scripts (IO, loops, conditional statements, etc.)

Overview

- A shell is the program that is running when you are at the command line
- It interprets your statements as you write them
- It can also execute a shell program -- a series of shell statements in a file
- Bash stands for Bourne Again Shell, a play on words because it is based on the Bourne Shell
- Other shells exist, but Bash is the most popular

Creating a Bash Program

- Create a text-based file (e.g., upper):

```
#!/bin/bash
echo "$1" | tr '[:lower:]' '[:upper:]'
```

1. $1 is the first argument
2. echo echoes the argument, but instead of being displayed,
| pipes it into the tr program
3. tr translates lower to uppercase

- Set it to be executable (chmod u+x upper)

```
% chmod u+x upper              # or chmod 711 upper
% ls -l upper
-rwxr--r--@ 1 me  staff  51 Jul 15 15:48 upper
```

rwx ==> the owner can execute the file

- Execute it by writing ./demo at the command line

```
% ./upper Hello
HELLO
```

Motivation (and soon-to-be exercises)

- You're a server administrator, and need to set up 18 Node.js projects
- You're a photographer and need to rename or renumber all the picture files in a directory
- You have a series of files that were written in Fall 2025, and you need to change every occurrence of "Fall 2025" to "Spring 2026"
- You're a web administrator, and want to easily check if your website is up-and-running
- You're a professor, and you need to process submissions on Canvas

Fundamentals - IO by Example

- Use echo for output, read for input
- Use $ to reference variables once they have been defined

```
#!/bin/bash

echo "What is your name (first last)?"
read first_name last_name

echo "Where do you live?"
read city

echo "Hello, $first_name $last_name, here's your weather: "

curl wttr.in/"$city"
```

```
What is your name (first last)?
Michael Rogers
Where do you live?
Oshkosh
Hello, Michael Rogers, here's your weather:
Weather report: Oshkosh

      \   /     Sunny
       .-.      +27(29) °C
    ― (   ) ―   ↑ 4 km/h
       `-'      16 km
      /   \     0.0 mm

                                  Wed 16 Jul
┌─────────────────────────────┬─────────────────────────────┬─────────────────────────────┬─────────────────────────────┐
│            Morning          │             Noon            │           Evening           │            Night            │
├─────────────────────────────┼─────────────────────────────┼─────────────────────────────┼─────────────────────────────┤
│      \   /     Sunny        │      \   /     Sunny        │ _`/"".-.     Thundery outbr… │                Overcast     │
│       .-.      +27(30) °C   │       .-.      +31(36) °C   │  ,\_(   ).   +17(16) °C     │      .--.      +14(13) °C   │
│    ― (   ) ―   ↗ 3-4 km/h   │    ― (   ) ―   ↙ 6-10 km/h  │   /(___(__)  ← 18-35 km/h   │   .-(    ).    ↙ 22-35 km/h │
│       `-'      10 km        │       `-'      9 km         │     ⚡‘‘⚡‘‘  7 km           │  (___.__)__)   7 km          │
│      /   \     0.0 mm | 0%  │      /   \     0.0 mm | 0%  │     ‘ ‘ ‘ ‘  0.0 mm | 0%   │                0.0 mm | 0%  │
└─────────────────────────────┴─────────────────────────────┴─────────────────────────────┴─────────────────────────────┘

                                  Thu 17 Jul
┌─────────────────────────────┬─────────────────────────────┬─────────────────────────────┬─────────────────────────────┐
│            Morning          │             Noon            │           Evening           │            Night            │
├─────────────────────────────┼─────────────────────────────┼─────────────────────────────┼─────────────────────────────┤
│    \  /       Partly Cloudy │                Overcast     │      \   /     Sunny        │      \   /     Clear        │
│  _ /"".-.     +15(14) °C    │      .--.      21 °C        │       .-.      +23(25) °C   │       .-.      17 °C        │
│    \_(   ).   ↓ 22-30 km/h  │   .-(    ).    ↓ 18-22 km/h │    ― (   ) ―   ← 10-13 km/h │    ― (   ) ―   ← 14-25 km/h │
│    /(___(__)  10 km         │  (___.__)__)   10 km        │       `-'      10 km        │       `-'      10 km        │
│               0.0 mm | 0%   │                0.0 mm | 0%  │      /   \     0.0 mm | 0%  │      /   \     0.0 mm | 0%  │
└─────────────────────────────┴─────────────────────────────┴─────────────────────────────┴─────────────────────────────┘
Location: Oshkosh, Winnebago County, Wisconsin, United States of America [44.0206919,-88.5408574]
```

Fundamentals - for Loops by Example

- Bash has while and for loops
- Use do and done to wrap the loop body
- If do is on a distinct line, the ; is optional
- In command substitution, $(command) gets replaced by the output of the command (and the current directory contained the 4 files shown here)

```bash
#!/bin/bash

# Standard for loop with list
for color in red yellow green; do
    echo "$color"
done

# For loop with range
for i in {1..5}; do
    echo "Processing Photo$i.png"
done

# For loop with command substitution
for file in $(ls *.bash); do
    echo "Processing $file"
done

# C-style for loop
for ((i=1; i<=5; i++)); do
    echo "Count: $i"
done
```

```
red
yellow
green
Processing Photo1.png
Processing Photo2.png
Processing Photo3.png
Processing Photo4.png
Processing Photo5.png
Processing a.bash
Processing b.bash
Processing c.bash
Processing testerama.bash
Count: 1
Count: 2
Count: 3
Count: 4
Count: 5
```

Fundamentals - while Loops by Example

- Boolean statements are enclosed in [ ], and the spaces are required

- Numeric operations use ((operation)) or let

```bash
#!/bin/bash

# Basic while loop
counter=1
while [ $counter -le 5 ]; do
    echo "Counter: $counter"
    ((counter++))
done

# While reading from file
while read line; do
    echo "Line: $line"
done < testerama.bash

# Infinite loop (use with break)
while true; do
    echo "Running..."
    sleep 1
    # break condition here
done
```

```
Counter: 1
Counter: 2
Counter: 3
Counter: 4
Counter: 5
Line: #!/bin/bash
Line:
Line: # Basic while loop
Line: counter=1
Line: while [ $counter -le 5 ]; do
Line: echo "Counter: $counter"
Line: ((counter++))
Line: done
Line:
Line: # While reading from file
Line: while read line; do
Line: echo "Line: $line"
Line: done < testerama.bash
Line:
Line: # Infinite loop (use with break)
Line: while true; do
Line: echo "Running..."
Line: sleep 1
Line: # break condition here
Running...
Running...
Running...
^C
```

Fundamentals - Conditional Statements

- if statement blocks are enclosed in then fi (instead of do done for loops)
- As with loops, the ; is optional if then appears on a separate line
- All the usual relational operators are represented: lt, le, gt, ge, eq, ne
- Boolean operators &&, || and ! are also available

```
echo -n "Enter age: "  # -n suppresses \n
read age
if [ $age > 18 ]; then
    echo "OK, you can vote"
fi

# If-else statement:
# -1 lists files one per line, -l counts # of lines
num_files=$(ls -1 . | wc -l)
if [ $num_files -gt 100 ]; then
    echo "My that is a large number of files"
    # commands if false
else
    echo "OK, this directory is not too big"
fi
```

```
Enter age: 25
OK, you can vote
OK, this directory is not too big
```

Exercises

- You're a server administrator, and need to set up 18 Node.js projects
- You're a photographer and need to rename or renumber all the picture files in a directory
- You have a series of files that were written in Fall 2025, and you need to change every occurrence of "Fall 2025" to "Spring 2026"
- You're a web administrator, and want to easily check if your website is up-and-running
- You're a professor, and you need to process submissions on Canvas

Reference

- https://learnxinyminutes.com/bash/