[Bash - Sample Shell Scripts - Google Docs](https://docs.google.com/document/d/115N884qwLieQRNLikyTCyk7EwbNJOH1sjilUihthcj4/edit)

**System Monitoring:**

* Check CPU Usage:
* #!/bin/bash
* echo "CPU Usage: $(top -bn1 | grep "Cpu(s)" | awk '{print $2}')%"
* Check Memory Usage:
* #!/bin/bash
* echo "Memory Usage: $(free -m | awk '/Mem/{print $3}') MB"
* Check Disk Space:
* #!/bin/bash
* echo "Disk Space: $(df -h / | awk '/\//{print $4}') available"
* Check Network Usage:
* #!/bin/bash
* echo "Network Usage: $(iftop -t -s 2 | grep "Total send and receive" | awk '{print $5}')"

**Log Analysis:**

* Tail a Log File:
* #!/bin/bash
* tail -f /var/log/application.log
* Search for Errors in Log:
* #!/bin/bash
* grep -i "error" /var/log/application.log
* Count Occurrences of a Log Pattern:
* #!/bin/bash
* grep -c "pattern" /var/log/application.log

**Automation:**

Automate Git Pull:

#!/bin/bash

cd /path/to/your/repo

git pull origin master

Backup Database:

#!/bin/bash

mysqldump -u username -p password database\_name > backup.sql

**Networking:**

* Check Open Ports:
* #!/bin/bash
* netstat -tuln
* Check DNS Resolution:
* #!/bin/bash
* nslookup example.com
* Ping a Host:
* #!/bin/bash
* ping -c 5 example.com

**Security:**

* Check for OpenSSH Vulnerability:
* #!/bin/bash
* ssh -V
* Check SSL Certificate Expiry:
* #!/bin/bash
* openssl s\_client -connect example.com:443 | openssl x509 -noout -dates

**IoT Monitoring:**

* Read Sensor Data:
* #!/bin/bash
* cat /dev/sensor
* Check Device Connectivity:
* #!/bin/bash
* ping -c 5 iot-device.local
* Monitor IoT Gateway Logs:
* #!/bin/bash
* tail -f /var/log/iot-gateway.log

**System Maintenance:**

* Rotate Logs:
* #!/bin/bash
* logrotate -f /etc/logrotate.conf
* Clean Temporary Files:
* #!/bin/bash
* find /tmp -type f -mtime +7 -exec rm {} \;

**Containerization:**

* List Docker Containers:
* #!/bin/bash
* docker ps -a
* Remove Unused Docker Containers:
* #!/bin/bash
* docker container prune -f
* Check Docker Images:
* #!/bin/bash
* docker images

**Web Services:**

* Check HTTP Response Code:
* #!/bin/bash
* curl -I example.com | grep HTTP/1.1
* Monitor REST API Endpoint:
* #!/bin/bash
* curl -X GET https://api.example.com/health

**Continuous Integration:**

Trigger Jenkins Job:

#!/bin/bash

curl -X POST -u username:token http://jenkins-server/job/job-name/build

Build and Deploy with Maven:

#!/bin/bash

mvn clean install && cp target/\*.war /path/to/deployment/directory

**Miscellaneous:**

* Generate Random Password:
* #!/bin/bash
* echo "$(date +%s | sha256sum | base64 | head -c 16)"
* Check System Uptime:
* #!/bin/bash
* uptime
* Run Cron Job:
* #!/bin/bash
* 0 1 \* \* \* /path/to/script.sh

**Process Management:**

* Check if a Process is Running:
* #!/bin/bash
* if pgrep -x "process\_name" > /dev/null; then
* echo "Process is running."
* else
* echo "Process is not running."
* fi
* Kill Processes Matching a Pattern:
* #!/bin/bash
* pkill -f "pattern"

**Conditional Logic:**

* Check Disk Space and Take Action:
* #!/bin/bash
* threshold=90
* disk\_space=$(df -h / | awk '/\//{print $5}' | tr -d '%')
* if [ "$disk\_space" -gt "$threshold" ]; then
* echo "Disk space is above $threshold%. Taking action..."
* # Add your actions here
* else
* echo "Disk space is within acceptable limits."
* fi
* Nested If-Else Statements:
* #!/bin/bash
* age=25
* if [ "$age" -lt 18 ]; then
* echo "You are a minor."
* elif [ "$age" -ge 18 ] && [ "$age" -lt 60 ]; then
* echo "You are an adult."
* else
* echo "You are a senior citizen."
* fi

**Error Handling:**

* Handle Errors and Log to File:
* #!/bin/bash
* command\_that\_might\_fail || { echo "Command failed" >&2; exit 1; }
* Try-Catch Mechanism:
* #!/bin/bash
* function try {
* "$@"
* }
* function catch {
* "$@"
* }
* try
* (
* # Your main script logic here
* echo "Executing main logic."
* )
* catch || {
* # Handle errors here
* echo "An error occurred."
* }

**File Operations:**

* Check if a File Exists:
* #!/bin/bash
* file\_path="/path/to/file.txt"
* if [ -e "$file\_path" ]; then
* echo "File exists."
* else
* echo "File does not exist."
* fi
* Read Lines from a File:
* #!/bin/bash
* file="/path/to/file.txt"
* while IFS= read -r line; do
* echo "Line: $line"
* done < "$file"

**Advanced Commands:**

* Process Substitution:
* #!/bin/bash
* diff <(command1) <(command2)
* Using trap for Signal Handling:
* #!/bin/bash
* trap 'cleanup' EXIT
* function cleanup {
* # Perform cleanup operations
* echo "Cleaning up..."
* }
* # Rest of the script logic here
* echo "Script running..."

**Functions:**

* Create and Use Functions:
* #!/bin/bash
* function greet {
* echo "Hello, $1!"
* }
* greet "John"
* greet "Alice"
* Return Value from Function:
* #!/bin/bash
* function add {
* echo $(($1 + $2))
* }
* result=$(add 5 3)
* echo "Result: $result"

**Network Operations:**

Check Internet Connectivity:

#!/bin/bash

if ping -c 1 google.com &> /dev/null; then

   echo "Internet is reachable."

else

    echo "Internet is not reachable."

fi

Retrieve External IP Address:

#!/bin/bash

external\_ip=$(curl -s ifconfig.me)

echo "External IP Address: $external\_ip"