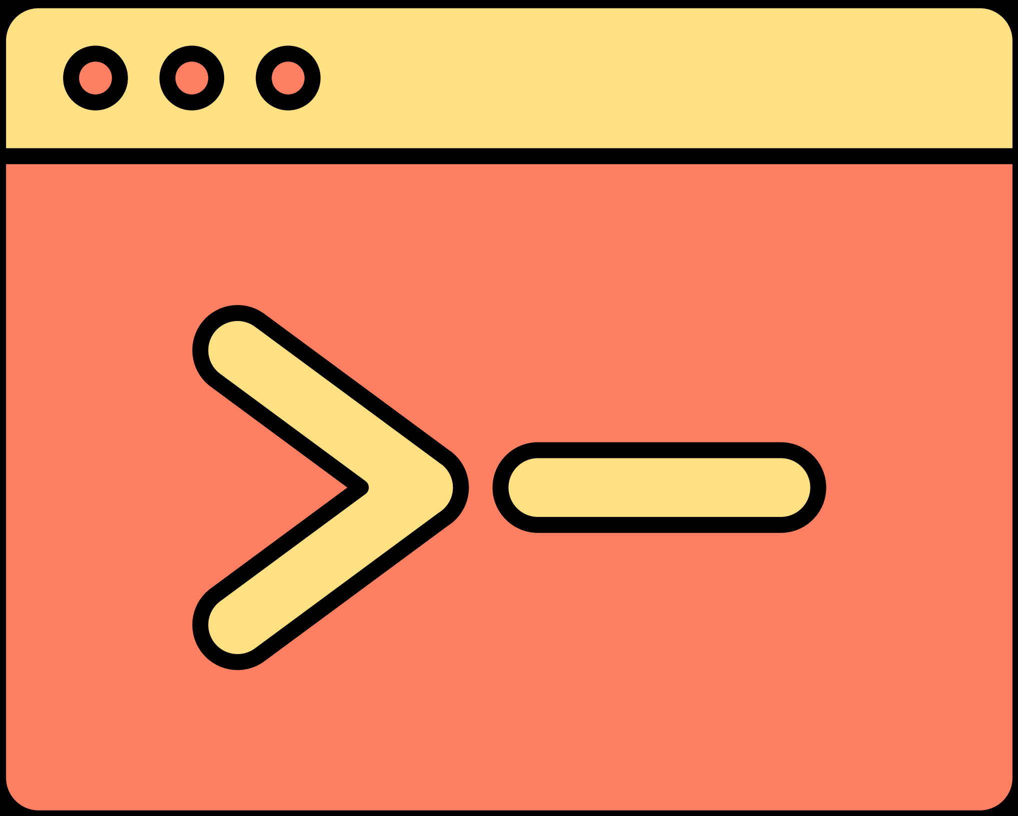
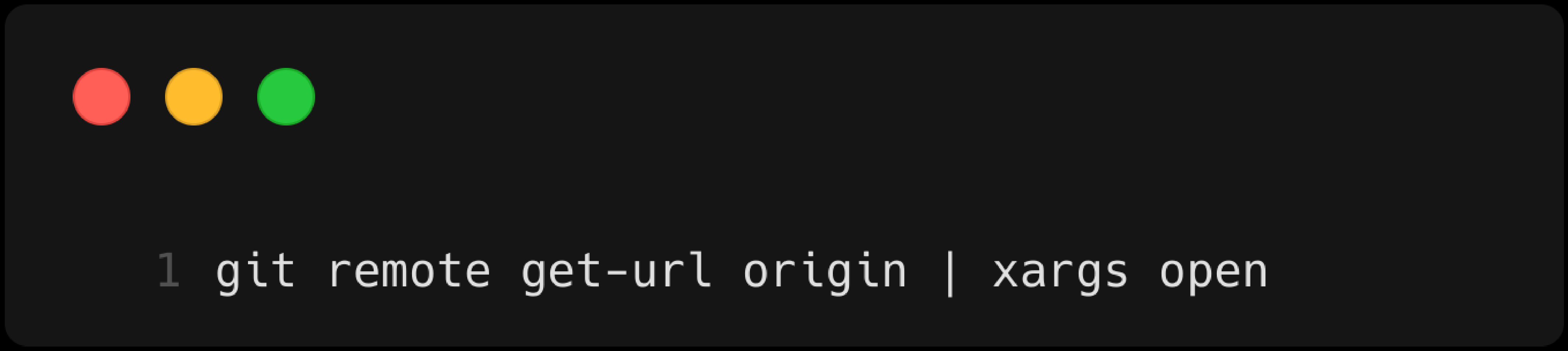
Optimizing Your Development Workflow with Bash Scripts!



Sharing a Powerful Bash Script for Streamlined Productivity 🚀

@itsvinayak

Open Git Repository:



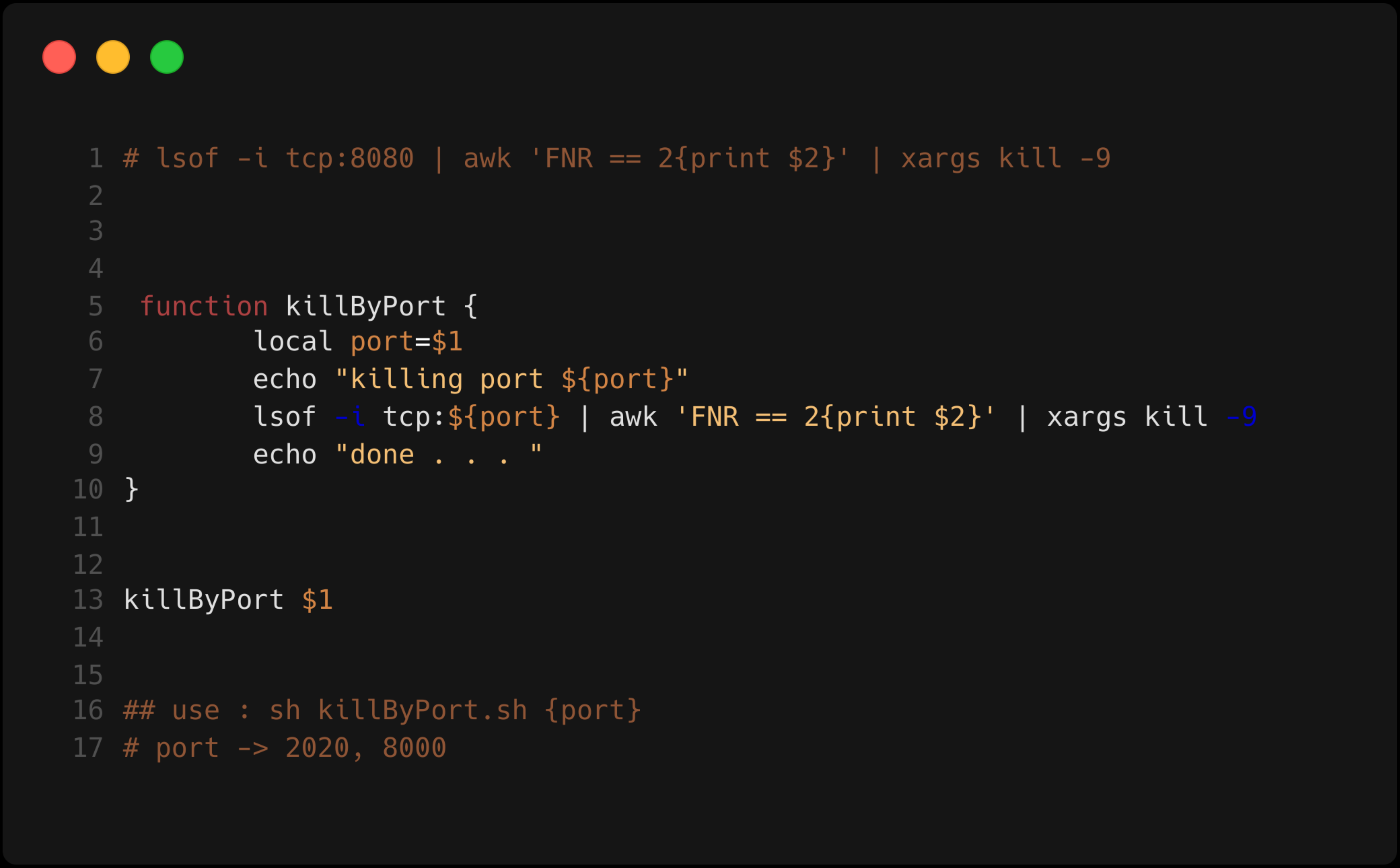
```
git remote get-url origin | xargs open
```

The command retrieves the URL of the "origin" remote repository and opens it in the default web browser. This is useful when you want to quickly visit the remote repository's webpage without having to manually copy and paste the URL into the browser.

@itsvinayak

Kill Process by Port:

The provided script is a Bash shell script designed to kill a process running on a specified port number.

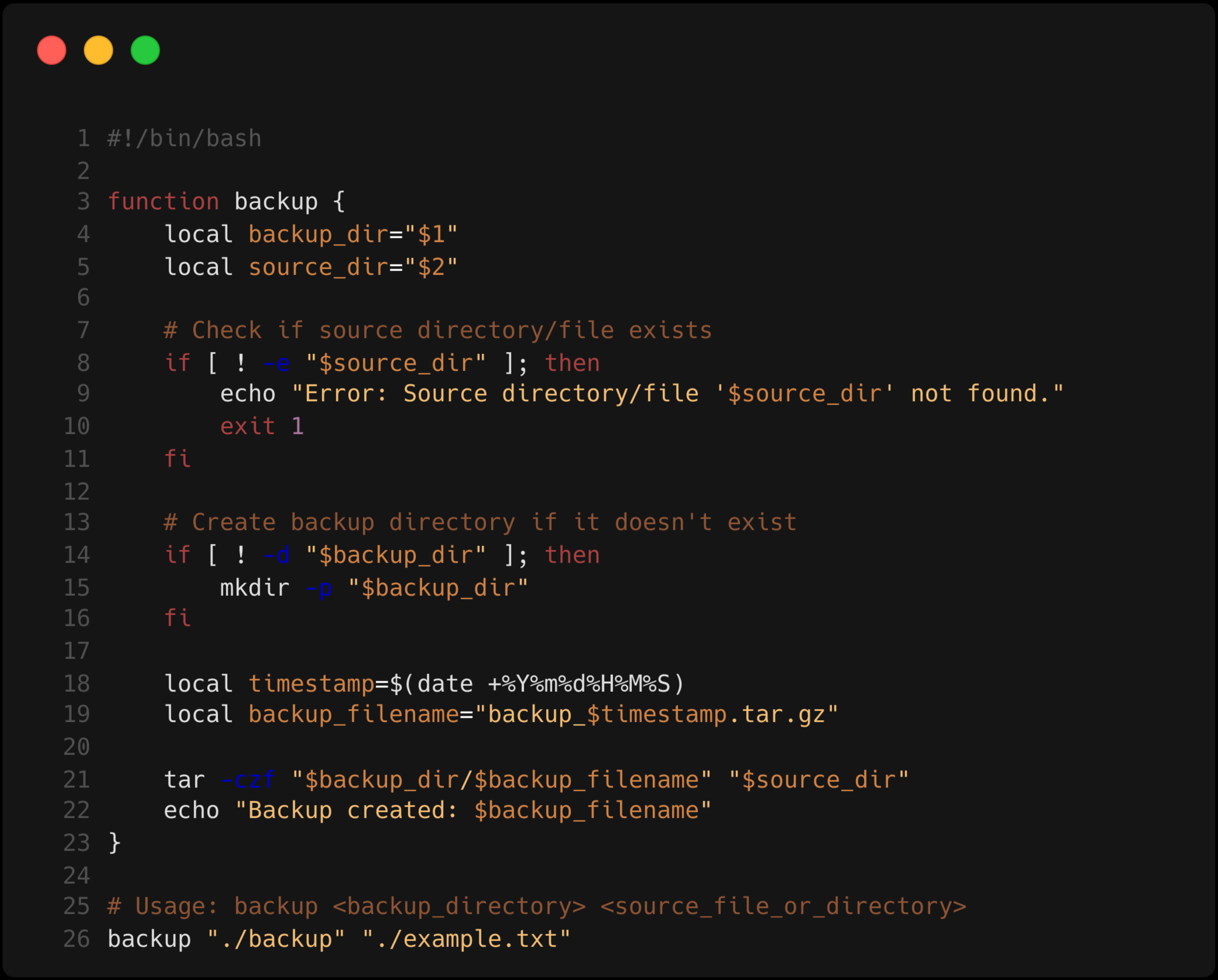


```bash
# lsof -i tcp:8080 | awk 'FNR == 2{print $2}' | xargs kill -9



 function killByPort {
        local port=$1
        echo "killing port ${port}"
        lsof -i tcp:${port} | awk 'FNR == 2{print $2}' | xargs kill -9
        echo "done . . . "
}


killByPort $1


## use : sh killByPort.sh {port}
# port -> 2020, 8000
```

@itsvinayak

Backup Script:

This script can be used to create a backup of specified files or directories.



```bash
#!/bin/bash

function backup {
    local backup_dir="$1"
    local source_dir="$2"

    # Check if source directory/file exists
    if [ ! -e "$source_dir" ]; then
        echo "Error: Source directory/file '$source_dir' not found."
        exit 1
    fi

    # Create backup directory if it doesn't exist
    if [ ! -d "$backup_dir" ]; then
        mkdir -p "$backup_dir"
    fi

    local timestamp=$(date +%Y%m%d%H%M%S)
    local backup_filename="backup_$timestamp.tar.gz"

    tar -czf "$backup_dir/$backup_filename" "$source_dir"
    echo "Backup created: $backup_filename"
}

# Usage: backup <backup_directory> <source_file_or_directory>
backup "./backup" "./example.txt"
```

@itsvinayak

Disk Space Checker:

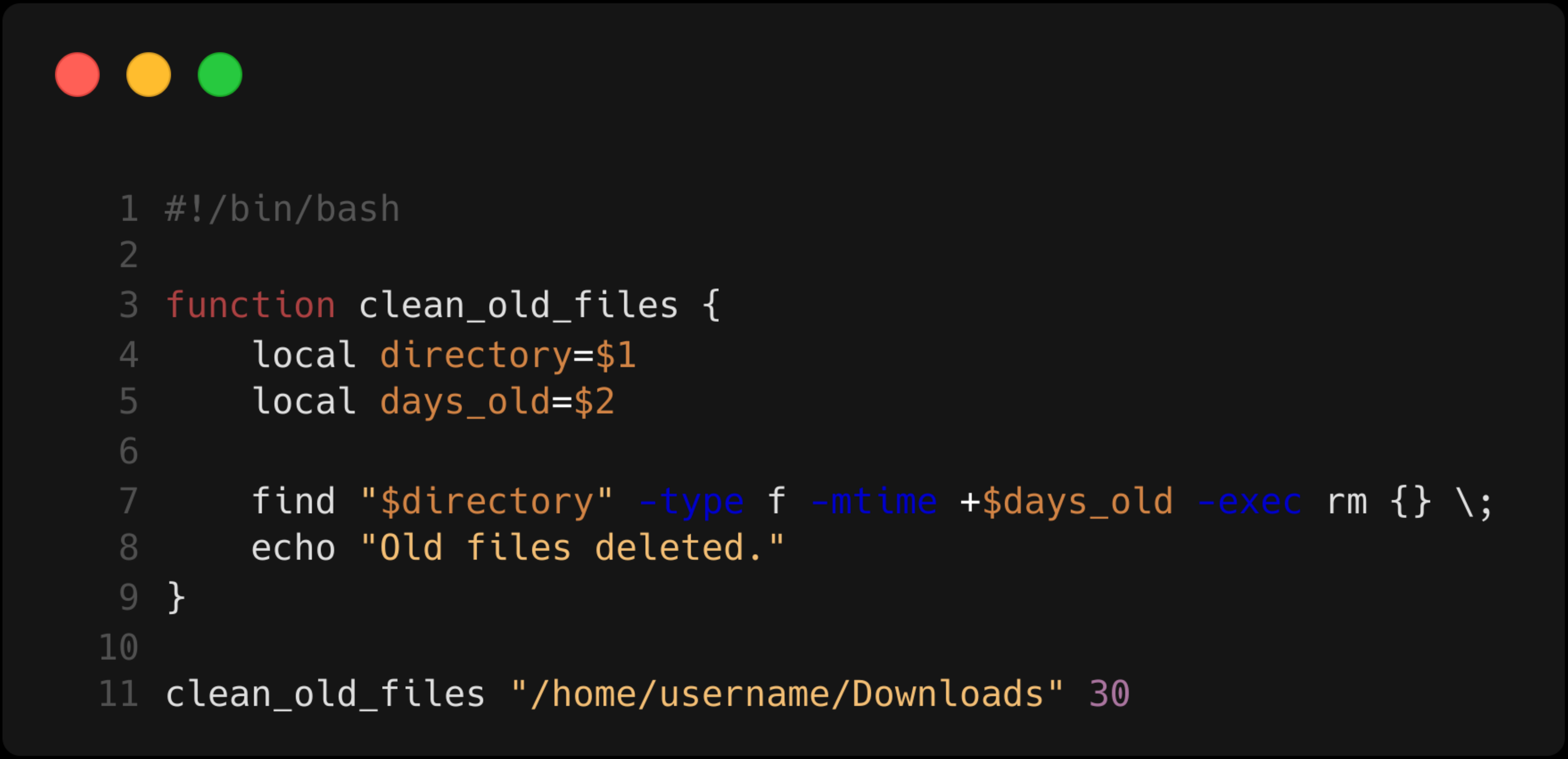
```bash
#!/bin/bash

function disk_space {
    local partition=$1
    local disk_space=$(df -h $partition | awk 'NR==2 {print $4}')
    echo "Available disk space on $partition: $disk_space"
}

disk_space /
```

This script checks the available disk space on the system.

@itsvinayak

File Cleanup Script:

This script deletes files older than a specified number of days in a directory.



```bash
#!/bin/bash

function clean_old_files {
    local directory=$1
    local days_old=$2

    find "$directory" -type f -mtime +$days_old -exec rm {} \;
    echo "Old files deleted."
}

clean_old_files "/home/username/Downloads" 30
```

@itsvinayak

Find and Replace in Files:

```bash
#!/bin/bash

function search_and_replace {
    local search_string="$1"
    local replace_string="$2"
    local directory="$3"

    grep -rl "$search_string" "$directory" | xargs sed -i "s/$search_string/$replace_string/g"
    echo "String '$search_string' replaced with '$replace_string' in files under '$directory'."
}

search_and_replace "exampleText" "exampleText2" ./example.txt
```

The script you provided defines a Bash function **search_and_replace** that takes three arguments: search_string, replace_string, and directory. It searches for files containing the **search_string** in the specified directory and replaces all occurrences of search_string with **replace_string** in those files using **sed -i**

@itsvinayak

@itsvinayak