Mastering Batch Programming: A Deep Dive into Shell Scripting

Welcome to Script Whiz’s Learning, where we embark on an exciting journey into the world of Batch Programming, commonly known as Batch, Bash scripting, and Shell Scripting. Whether you're a seasoned developer or just starting out, this document aims to equip you with a comprehensive understanding of Batch Programming and its potential to revolutionize your development work.

Introduction

Picture this: you're working on a complex project, and repetitive tasks are eating up your precious time. This is where Batch Programming comes to the rescue. In this session, we'll explore the fundamentals of Shell scripting, discussing how it can lead to superior software development, streamlined processes, and increased overall productivity.

Our journey begins with an introduction to Batch Programming, highlighting its essential features, functionalities, and diverse applications. Throughout this document, we'll demonstrate practical Batch Scripting implementations and guide you through creating a basic game using this script. By the end of our exploration, you'll possess a comprehensive knowledge of Batch Programming and its versatile use cases.

Overview

Batch Programming, often referred to as batch scripting or shell scripting, revolves around the use of BASH commands. It's a specialized form of programming designed to automate tasks within a command-line interface (CLI) environment. With Batch Programming, you can create scripts that execute a sequence of commands in a specific order, simplifying complex operations.

These scripts are composed of a series of commands executed in sequence. They're typically saved with a .bat or .cmd file extension and can be executed with a simple double-click or by running them through a command prompt.

Advantages

Now, let's explore the numerous advantages of Batch Programming. It's a game-changer when it comes to automating repetitive tasks, such as file backups, system maintenance, and software installations. But Batch Programming doesn't stop there; it can handle more complex operations like database backups, data processing, and network management. The real magic lies in its ability to eliminate the tedium of manual execution and reduce the risk of human error. Before executing scripts, you can rigorously test and debug them, ensuring reliability and accuracy.

But enough theory; let's get practical. It's time to dive into the world of scripting commands. This section promises to be both informative and engaging, providing you with a deeper understanding of the subject.

In the world of computer programming, not every solution requires a high-level language like Java, .NET, or C++. Sometimes, simplicity is key, and that's where Batch Programming steps in. Unlike its sophisticated counterparts, Batch Programming is a straightforward, interpreted scripting language that operates on a unique set of commands known as DOS commands. In this comprehensive guide, we'll unravel the world of DOS commands and delve into the realm of Batch Programming, exploring its capabilities and practical applications.

The Foundation: DOS Commands

At its core, Batch Programming is built upon a foundation of DOS commands. These commands, such as echo, cd, dir, and mkdir, are integral parts of the operating system. They serve as the building blocks for creating powerful scripts that can automate a series of tasks.

Exploring Common DOS Commands

Let's take a closer look at some of the commonly used DOS commands that serve as the building blocks for Batch Programming:

1. **ECHO:** The ECHO command is a versatile tool. It can display messages on the screen or control command echoing. "@echo off" turns off command display for the entire script, while "echo on" reverts it. Adding the "@" sign before "echo on" ensures that the command applies to itself as well. You can toggle echo on or off at any point in a batch file.

2. **PAUSE:** Need to pause script execution and wait for user input? The PAUSE command prompts users to press any key to continue. It's not only helpful for user interaction but also invaluable for debugging.
3. **COLOR:** Customize your Command Prompt's appearance with the COLOR command. You can set foreground and background colors to create a unique interface. For example, "color 0A" sets black text on a light green background. The possibilities are endless.

0 = Black	8 = Gray
1 = Blue	9 = Light Blue
2 = Green	A = Light Green
3 = Aqua	B = Light Aqua
4 = Red	C = Light Red
5 = Purple	D = Light Purple
6 = Yellow	E = Light Yellow
7 = White	F = Bright White

Syntax: COLOR [background] [foreground]

4. **TITLE:** When working with multiple Command Prompt windows, keeping track of tasks can be challenging. The TITLE command lets you set a custom title for the Command Prompt window, making organization a breeze.
5. **:: Double Colon:** Similar to REM (Remark), two colons at the start of a line indicate a remark within a batch file. These remarks are never displayed or executed, regardless of whether "ECHO" is on or off.
6. **REM:** Remarks in batch files are crucial for documentation. REM allows you to add comments that won't execute when the batch file runs. Unlike "::," REM comments are visible and affected by echo settings.
7. **CLS:** Clear the Command Prompt screen with the CLS command. It provides a clean slate for your commands and output.
8. **EXIT:** When you're finished with your DOS session, the EXIT command gracefully closes the Command Prompt window.

Labels, Control Flow, and Variables

Batch Programming goes beyond simple commands. It incorporates labels and control flow with commands like GOTO, enabling you to create more complex scripts. Labels, marked by a colon (e.g., ":LABEL"), serve as navigation points in your script.

Variables and conditions also play a pivotal role in Batch Programming. The SET command initializes variables, and IF statements introduce decision-making based on conditions. This flexibility empowers you to create dynamic and responsive scripts.

Syntax: set /P anyname = value

Where, "anyname" is the name of the variable to be set.
"Value" is the value which needs to be set against the variable.
/P allows the user to set a variable equal to a line of input entered by the user. The prompt string is displayed before the user input is read.

The value of the variable declared can be shown on the screen by enclosing in the % sign.
Example: set message = Hello World
echo %message%

Beyond Basic Commands

While DOS commands provide a solid foundation, Batch Programming extends its capabilities with additional commands like Ipconfig, taskkill, and powercfg. These commands open doors to system administration and automation, expanding your scripting possibilities.

Tasklist: Your Task Manager in the Command Line

Tasklist is a command line utility that displays a list of all currently running processes on your Windows system. It's like having Task Manager at your fingertips, but without the need for a graphical interface.

Findstr: The Text Search Ninja

Findstr is your go-to tool when you need to search for specific text patterns within files. It's like having a search engine for your command line. Here's how it works:

```
tasklist | findstr "Teams.exe"
```

Taskkill: Taming Rogue Processes

When a program misbehaves or becomes unresponsive, you may need to terminate it forcefully. Taskkill is the tool for the job. It allows you to end processes gracefully or with a firm hand.

Let's break down some key options:

- /pid processid: Terminates a process by specifying its Process ID (PID).
- /im imagename: Kills processes by image name (e.g., "notepad.exe").
- /f: Forces the process to terminate.
- /t: Terminates the specified process and any child processes it has spawned.

```
Taskkill /f /pid <id>
```

When it comes to the Windows command line, there's a treasure trove of commands that can make your life easier and your system more efficient. In this blog post, we're going to uncover the capabilities of five essential commands: Attrib, Wlan, Subst, Prompt, and CURL. Each of these commands serves a unique purpose, from managing file attributes to handling network connections and enabling data exchange between devices.

Attrib: Mastering File Attributes

Attrib is a versatile command that allows you to manipulate file attributes. With it, you can control whether a file or folder is hidden, marked as a system file, or set as read-only. Here's how it works:

- +h: Makes a file or folder hidden, rendering it invisible to the user.
- -h: Clears the hidden file attribute, making the file or folder visible again.
- +s: Sets the file attribute as a system file.
- -s: Clears the system file attribute.
- +r: Sets the read-only attribute, making the file or folder read-only.
- -r: Clears the read-only attribute, allowing changes to the file or folder.

Example:

To make a folder named "MySecrets" hidden, a system file, and read-only, you can use the following command:

```
attrib +h +s +r MySecrets
```

To revert these attributes and make the folder visible, non-system, and editable, you can use:

```
attrib -h -s -r MySecrets
```

Wlan: Managing Wi-Fi Connections

The Wlan command is your gateway to managing Wi-Fi connections from the command line. It provides insights into the Wi-Fi networks you've connected to and allows you to retrieve network details, including passwords.

- To view a list of Wi-Fi profiles you've connected to:

netsh wlan show profile

- To see the password for a specific Wi-Fi network (replace "ScriptWhiz" with the network name):

netsh wlan show profile "ScriptWhiz" key=clear

This can be especially useful if you need to share a Wi-Fi password or troubleshoot connection issues.

Subst: Substituting Drive Letters

The Subst command lets you substitute a drive letter for a path. It's a handy way to create virtual drives for specific directories. Here's how it works:

- To create a virtual drive (e.g., drive Y:) for a path:
  subst y: "path"
- To remove a virtual drive (e.g., drive Y:):
  subst /d y:

Virtual drives can make navigating to frequently used directories a breeze.

Prompt: Customizing Your Command Prompt

The Prompt command allows you to customize the text displayed in your command prompt. It's a great way to add context or information to your prompt.

- To set a custom prompt (replace "<any text>" with your desired text):
  prompt <any text>$G
- To restore the default prompt:
  Prompt

Custom prompts can help you stay organized and provide reminders or context while working in the command line.

CURL: Data Exchange Made Easy

CURL is a command line tool that facilitates data exchange between devices and servers through a terminal. It's a versatile tool for making HTTP requests, downloading files, and automating data transfer.

- To fetch weather data for India from wttr.in:

```
curl wttr.in/India
```

CURL opens up a world of possibilities for automating tasks and fetching data from the web.

Below are few code snippets to try hands-on:

Color

```
@echo off
color b1
pause
```

Title

```
@echo off
title My Game
pause
```

REM vs ::

```
@echo off
color b1
title My Game
:: This is a random comment
rem This is a second random comment
pause
```

Set

```
@echo off
set t
set a
pause
```

Operators

Relational Operators

```
@echo off
SET /A a = 5
SET /A b = 10
if %a% EQU %b% echo A is equal to than B
if %a% NEQ %b% (
 echo A is not equal to than B
 echo This is a multi lined 'if - NEQ'
)
if %a% LSS %b% echo A is less than B
if %a% LEQ %b% echo A is less than or equal B
if %a% GTR %b% echo A is greater than B
if %a% GEQ %b% echo A is greater than or equal to B

pause
```

Arithmetic Operators

```
@echo off
SET /A a = 5
SET /A b = 10
SET /A c = %a%+%b%
echo %c%
SET /A c = %a%-%b%
echo %c%
SET /A c = %b%*%a%
echo %c%
SET /A c = %b%/%a%
echo %c%
SET /A c =%b% %% %a%
echo %c%

pause
```

Conclusion

Mastering DOS commands and Batch Programming is a valuable skill for anyone seeking to streamline tasks, automate processes, and become proficient with the command line. Whether you're a seasoned developer or just starting your programming journey, understanding the fundamentals of Batch Programming can be a game-changer.

SCRIPT WHIZ

- Yashwanth Byalla

In upcoming articles, we'll dive deeper into Batch Programming, exploring advanced techniques and real-world use cases. So, stay tuned as we unlock the full potential of command-line scripting. Feel free to experiment with these commands to discover their true power and efficiency.

HAPPY LEARNING!

SCRIPT WHIZ

- Yashwanth Byalla