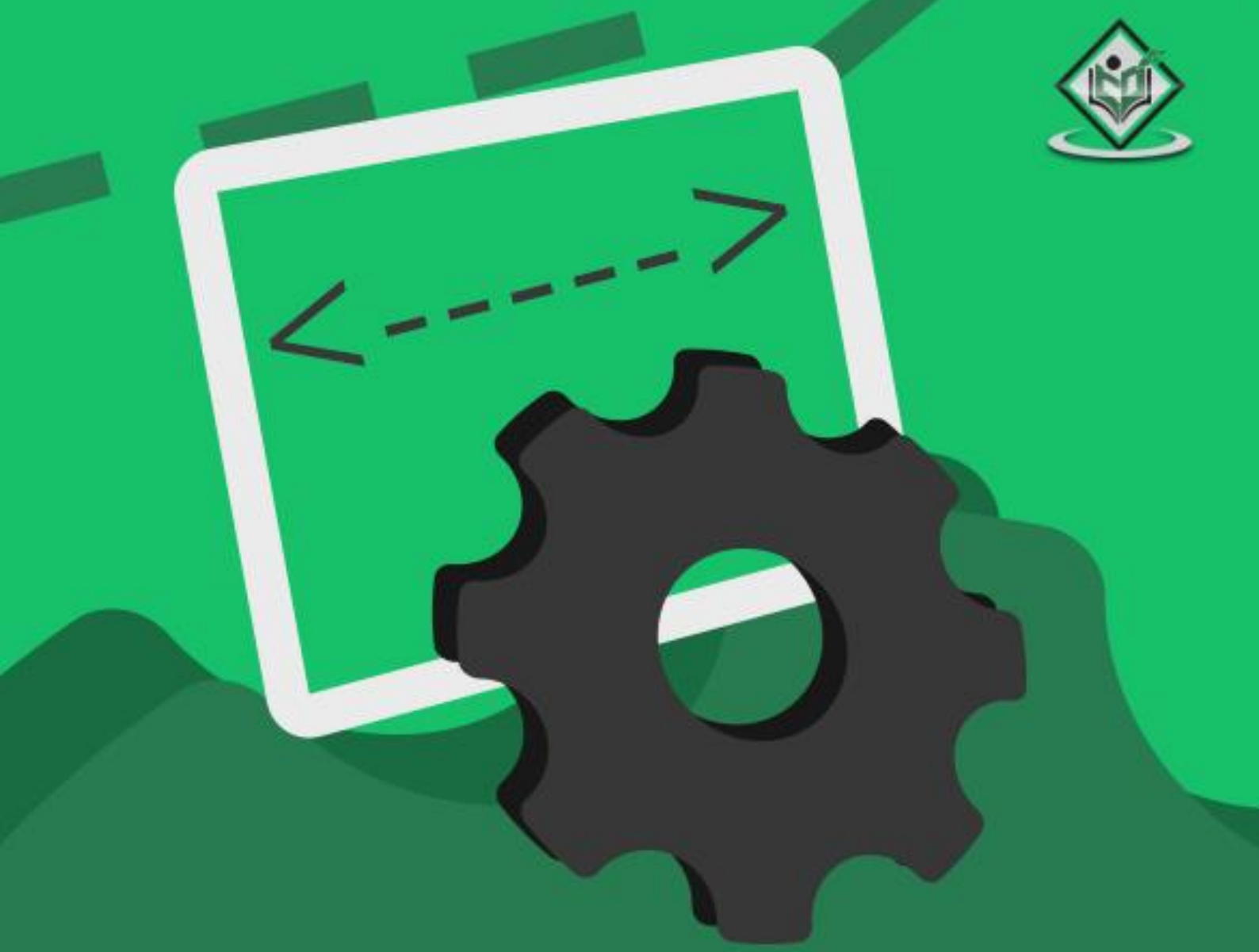


BATCH SCRIPTING

tutorialspoint

SIMPLYEASYLEARNING

www.tutorialspoint.com

https://www.facebook.com/tutorialspointindia

https://twitter.com/tutorialspoint

About this Tutorial

Batch scripts are stored in simple text files containing lines with commands that get executed in sequence, one after the other. Scripting is a way by which one can alleviate this necessity by automating these command sequences in order to make one's life at the shell easier and more productive.

This tutorial discusses the basic functionalities of batch scripting along with relevant examples for easy understanding.

Audience

This tutorial has been prepared for beginners to understand the basic concepts of batch scripting.

Prerequisites

A reasonable knowledge of computer programming and concepts such as variables, commands, syntax, etc. is desired.

Copyright & Disclaimer

Table of Contents

1. Batch Scripting – Overview

Batch scripting is incorporated to automate command sequences which are repetitive in nature. Scripting is a way by which one can alleviate this necessity by automating these command sequences in order to make one's life at the shell easier and more productive. In most organizations, batch scripting is incorporated in some way or the other to automate stuff.

Some of the features of batch scripting are:

- Can read inputs from users so that it can be processed further.
- Has control structures such as for, if, while, switch for better automating and scripting.
- Supports advanced features such as Functions and Arrays.
- Supports regular expressions.
- Can include other programming codes such as Perl.

Some of the common uses of batch scripting are:

- Setting up servers for different purposes.
- Automating housekeeping activities such as deleting unwanted files or log files.
- Automating the deployment of applications from one environment to another.
- Installing programs on various machines at once.

Batch scripts are stored in simple text files containing lines with commands that get executed in sequence, one after the other. These files have the special extension BAT or CMD. Files of this type are recognized and executed through an interface (sometimes called a shell) provided by a system file called the command interpreter. On Windows systems, this interpreter is known as cmd.exe.

Running a batch file is a simple matter of just clicking on it. Batch files can also be run in a command prompt or the Start-Run line. In such case, the full path name must be used unless the file's path is in the path environment. Following is a simple example of a batch script. This batch script when run deletes all files in the current directory.

```
:: Deletes All files in the Current Directory With Prompts and Warnings
::(Hidden, System, and Read-Only Files are Not Affected)
```

```
:: @ECHO OFF
DEL . DR
```

2. Batch Scripting – Environment

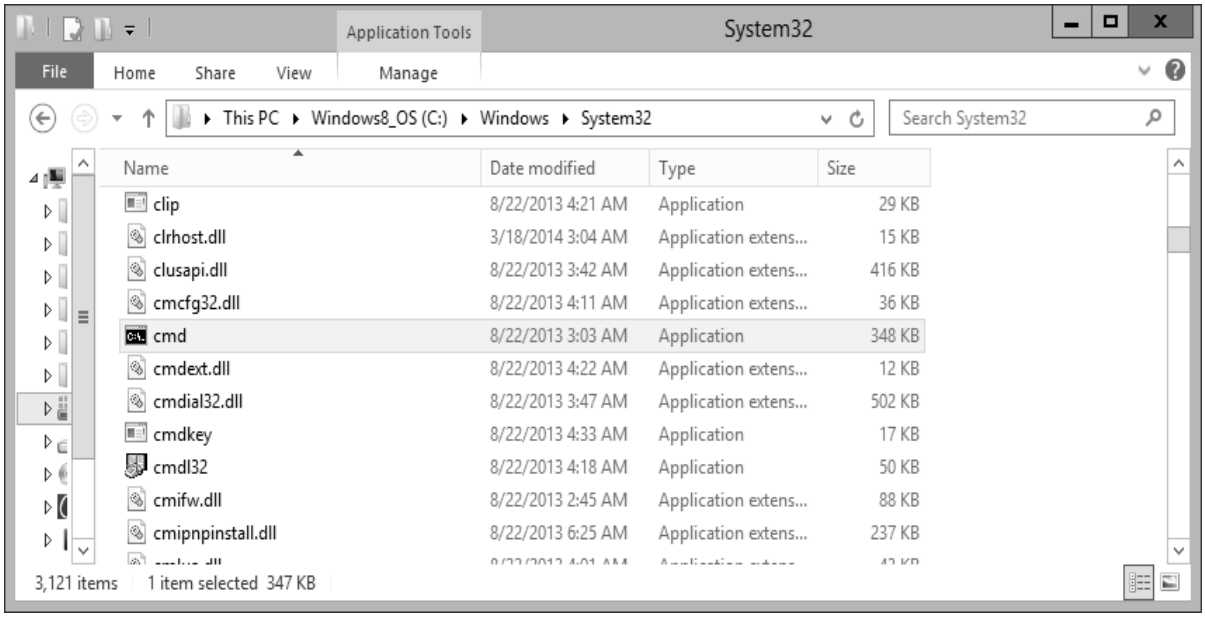
This chapter explains the environment related to batch scripting.

Writing and Executing

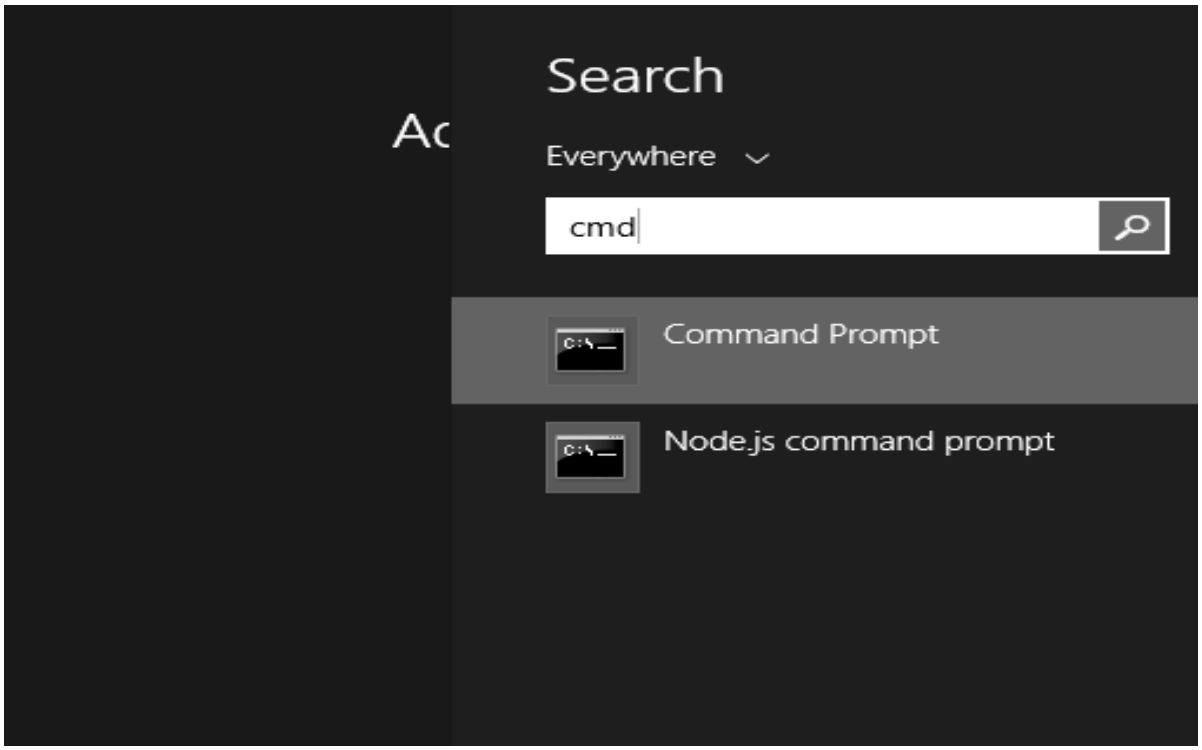
Typically, to create a batch file, notepad is used. This is the simplest tool for creation of batch files. Next is the execution environment for the batch scripts. On Windows systems, this is done via the command prompt or cmd.exe. All batch files are run in this environment.

Following are the different ways to launch cmd.exe:

Method 1: Go to C:\Windows\System32 and double click on the cmd file.



Method 2: Via the run command – The following snapshot shows to find the command prompt(cmd.exe) on Windows server 2012.



Once the cmd.exe is launched, you will be presented with the following screen. This will be your environment for executing your batch scripts.

Environment Variables

In order to run batch files from the command prompt, you either need to go to the location to where the batch file is stored or alternatively you can enter the file location in the path environment variable. Thus assuming that the batch file is stored in the location C:\Application\bin, you would need to follow these instructions for the PATH variable inclusion.

OS	Output
Windows	Append the String; C:\Application\bin to the end of the system variable PATH.

3. Batch Scripting – Commands

In this chapter, we will look at some of the frequently used batch commands.

ver

This batch command shows the version of MS-DOS you are using.

Syntax

```
ver
```

Example

```
@echo off
ver
```

Output

The output of the above command is as follows. The version number will depend upon the operating system you are working on.

```
Microsoft Windows [Version 6.3.9600]
```

ASSOC

This is a batch command that associates an extension with a file type (FTYPE), displays existing associations, or deletes an association.

Syntax

```
assoc – Displays all the file extensions
assoc | find ".ext" – Displays only those file extensions which have the extension
ext.
```

Example

```
@echo off
assoc > C:\lists.txt
```

```
assoc | find “.doc” > C:\listsdoc.txt
```

Output

The list of file associations will be routed to the file lists.txt. The following output shows what is there in the listsdoc.txt file after the above batch file is run.

```
.doc=Word.Document.8
.dochtml=wordhtmlfile
.docm=Word.DocumentMacroEnabled.12
.docmhtml=wordmhtmlfile
.docx=Word.Document.12
.docxml=wordxmlfile
```

CD

This batch command helps in making changes to a different directory, or displays the current directory.

Syntax

```
cd
```

Example

The following example shows how the cd command can be used in a variety of ways.

```
@echo off
Rem The cd without any parameters is used to display the current working directory
cd
Rem Changing the path to Program Files
cd\Program Files
cd
Rem Changing the path to Program Files
cd %USERPROFILE%
cd
Rem Changing to the parent directory
```

```
cd..
cd
Rem Changing to the parent directory two levels up
cd..\..
cd
```

Output

The above command will display the following output after changing to the various folder locations.

```
C:\Users\Administrator
C:\Program Files
C:\Users\Administrator
C:\Users
C:\
```

CLS

This batch command clears the screen.

Syntax

```
cls
```

Example

```
@echo off
Cls
```

Output

The command prompt screen will be cleared.

Copy

This batch command is used for copying files from one location to the other.

Syntax

```
Copy [source] [destination]
```

The files will be copied from source to destination location.

Example

The following example shows the different variants of the **copy** command.

```
@echo off

cd

Rem Copies lists.txt to the present working directory. If there is no destination
identified , it defaults to the present working directory.

copy c:\lists.txt

Rem The file lists.txt will be copied from C:\ to C:\tp location

copy C:\lists.txt c:\tp

Rem Quotation marks are required if the file name contains spaces

copy “C:\My File.txt”

Rem Copies all the files in F drive which have the txt file extension to the
current working directory

copy F:\*.txt

Rem Copies all files from dirA to dirB. Note that directories nested in dirA will
not be copied

copy C:\dirA dirB
```

Output

All actions are performed as per the remarks in the batch file.

DEL

This batch command deletes files and not directories.

Syntax

```
del [filename]
```

Example

The following example shows the different variants of the **del** command.

```
@echo off

Rem Deletes the file lists.txt in C:\
```

```
del C:\lists.txt

Rem Deletes all files recursively in all nested directories

del /s *.txt

Rem Deletes all files recursively in all nested directories , but asks for the
confirmation from the user first

Del /p /s *.txt
```

Output

All actions are performed as per the remarks in the batch file.

DIR

This batch command lists the contents of a directory.

Syntax

```
dir
```

Example

The following example shows the different variants of the **dir** command.

```
@echo off

Rem All the directory listings from C:\ will be routed to the file lists.txt

dir C:\>C:\lists.txt

Rem Lists all directories and subdirectories recursively

dir /s

Rem Lists the contents of the directory and all subdirectories recursively, one
file per line, displaying complete path for each listed file or directory.

dir /s /b

Rem Lists all files with .txt extension.

dir *.txt

Rem Includes hidden files and system files in the listing.

dir /a

Rem Lists hidden files only.

dir /ah
```

Output

All actions are performed as per the remarks in the batch file.

DATE

This batch command help to find the system date.

Syntax

```
DATE
```

Example

```
@echo off
echo %DATE%
```

Output

The current date will be displayed in the command prompt. For example,

```
Mon 12/28/2015
```

ECHO

This batch command displays messages, or turns command echoing on or off.

Syntax

```
ECHO “string”
```

Example

The following example shows the different variants of the dir command.

```
Rem Turns the echo on so that each command will be shown as executed
echo on
echo "Hello World"
Rem Turns the echo off so that each command will not be shown when executed
@echo off
```

```
echo "Hello World"
Rem Displays the contents of the PATH variable
echo %PATH%
```

Output

The following output will be displayed in the command prompt.

```
C:\>Rem Turns the echo on so that each command will be shown as executed

C:\>echo on

C:\>echo "Hello World"
"Hello World"

C:\>Rem Turns the echo off so that each command will not be shown when executed

"Hello World"
C:\Users\ADMINI~1\AppData\Local\Temp
```

EXIT

This batch command exits the DOS console.

Syntax

```
Exit
```

Example

```
@echo off
echo "Hello World"
exit
```

Output

The batch file will terminate and the command prompt window will close.

MD

This batch command creates a new directory in the current location.

Syntax

```
md [new directory name]
```

Example

```
@echo off
md newdir
cd newdir
cd
Rem “Goes back to the parent directory and create 2 directories”
cd..
md newdir1 newdir1
cd newdir1
cd
cd..
cd newdir2
cd
```

Output

The above command produces the following output.

```
C:\newdir
C:\newdir1
C:\newdir2
```

MOVE

This batch command moves files or directories between directories.

Syntax

```
move [source] [destination]
```

The files will be copied from source to destination location.

Example

The following example shows the different variants of the move command.

```
@echo off

Rem Moves the file list.txt to the directory c:\tp

move C:\lists.txt c:\tp

Rem Renames directory Dir1 to Dir2, assuming Dir1 is a directory and Dir2 does not
exist.

move Dir1 Dir2

Rem Moves the file lists.txt to the current directory.

move C:\lists.txt
```

Output

All actions are performed as per the remarks in the batch file.

End of ebook preview

If you liked what you saw…

Buy it from our store @ **https://store.tutorialspoint.com**