# Batch Script - Comments

It's always a good practice to add comments or documentation for the scripts which are created. This is required for maintenance of the scripts to understand what the script actually does.

For example, consider the following piece of code which has no form of comments. If any average person who has not developed the following script tries to understand the script, it would take a lot of time for that person to understand what the script actually does.

```
ECHO OFF
IF NOT "%OS%"=="Windows_NT" GOTO Syntax
ECHO.%* | FIND "?" >NUL
IF NOT ERRORLEVEL 1 GOTO Syntax
IF NOT [%2]==[] GOTO Syntax
SETLOCAL
SET WSS=
IF NOT [%1]==[] FOR /F "tokens = 1 delims = \ " %%A IN ('ECHO.%~1') DO SET
FOR /F "tokens = 1 delims = \ " %%a IN ('NET VIEW ^| FIND /I "\\%WSS%"') DO
"tokens = 1 delims = " %%A IN ('NBTSTAT -a %%a ^| FIND /I /V "%%a" ^| FIND
DO ECHO.%%a %%A
ENDLOCAL
GOTO:EOF
ECHO Display logged on users and their workstations.
ECHO Usage: ACTUSR [ filter ]
IF "%OS%"=="Windows_NT" ECHO Where: filter is the first part
of the computer name^(s^) to be displayed
```

## Comments Using the Rem Statement

There are two ways to create comments in Batch Script; one is via the Rem command. Any text which follows the Rem statement will be treated as comments and will not be executed. Following is the general syntax of this statement.

### Syntax

```
Rem Remarks
```

where 'Remarks' is the comments which needs to be added.

The following example shows a simple way the **Rem** command can be used.

### Example

```
@echo off
Rem This program just displays Hello World
set message=Hello World
echo %message%
```

### Output

The above command produces the following output. You will notice that the line with the Rem statement will not be executed.

```
Hello World
```

## Comments Using the :: Statement

The other way to create comments in Batch Script is via the :: command. Any text which follows the :: statement will be treated as comments and will not be executed. Following is the general syntax of this statement.

### Syntax

```
:: Remarks
```

where ‘Remarks’ is the comment which needs to be added.

The following example shows the usage of the "::" command.

### Example

```
@echo off
:: This program just displays Hello World
set message = Hello World
echo %message%
```

### Output

The above command produces the following output. You will notice that the line with the :: statement will not be executed.

```
Hello World
```

**Note** − If you have too many lines of Rem, it could slow down the code, because in the end each line of code in the batch file still needs to be executed.

Let's look at the example of the large script we saw at the beginning of this topic and see how it looks when documentation is added to it.

```
::===============================================================
:: The below example is used to find computer and logged on users
::
::===============================================================
ECHO OFF
:: Windows version check
IF NOT "%OS%"=="Windows_NT" GOTO Syntax
ECHO.%* | FIND "?" >NUL
:: Command line parameter check
IF NOT ERRORLEVEL 1 GOTO Syntax
IF NOT [%2]==[] GOTO Syntax
:: Keep variable local
SETLOCAL
:: Initialize variable
SET WSS=
:: Parse command line parameter
IF NOT [%1]==[] FOR /F "tokens = 1 delims = \ " %%A IN ('ECHO.%~1') DO SET
:: Use NET VIEW and NBTSTAT to find computers and logged on users
FOR /F "tokens = 1 delims = \ " %%a IN ('NET VIEW ^| FIND /I "\\%WSS%"') DO
"tokens = 1 delims = " %%A IN ('NBTSTAT -a %%a ^| FIND /I /V "%%a" ^| FIND
"<03>"') DO ECHO.%%a %%A
:: Done
ENDLOCAL
GOTO:EOF
:Syntax
ECHO Display logged on users and their workstations.
ECHO Usage: ACTUSR [ filter ]
IF "%OS%"=="Windows_NT" ECHO Where: filter is the first part of the
computer name^(s^) to be displayed
```

You can now see that the code has become more understandable to users who have not developed the code and hence is more maintainable.