**Setting up Multi Environments Using Batch Files**

**Introduction**

The ability to run one script or several scripts against several environments, by using one batch file or using a single click proves to be very useful during smoke or regression testing when several machines needs to be tested. Batch files can be used to specific which scripts to be execute and what environment and the name and location of the output files. Scripts can be executed at the bash prompt or with little modification can be executed in the command prompt.

**Command Prompt vs Bash Prompt**

The command prompt is used to execute line commands such as dir, cd, etc., and is based on the DOS prompt from older operating systems. Because the command prompt is not case sensitive commands such as DIR, dir, and Dir, displays the directory of the current folder, and CD, cd, and Cd performs the function of changing directories. To open up the command prompt click run from the start menu and type Cmd (Figure 1), again not case sensitive. This results in a window poping up (Figure 2)

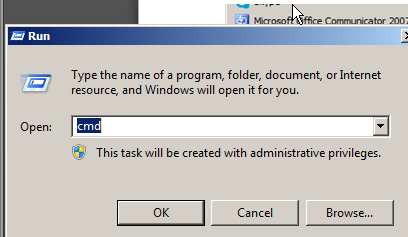


Figure 1 Starting the Command Prompt

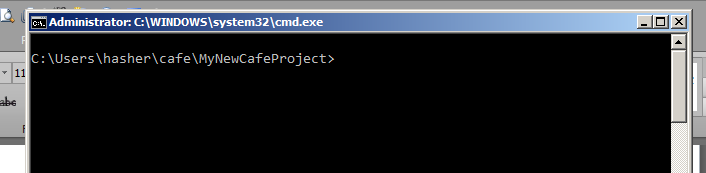


Figure 2 The Command Prompt Window

The Bash prompt window is similar to the command prompt but instead of using the dir command for displaying current directory the bash prompt uses pwd for present working directory, and ls for listing the file in the current directory. To start the Bash prompt click start and select Git Bash (figure 3). After click on Git Bash the prompt window will open (figure 4)

.



Figure 3 Starting Git Bash prompt

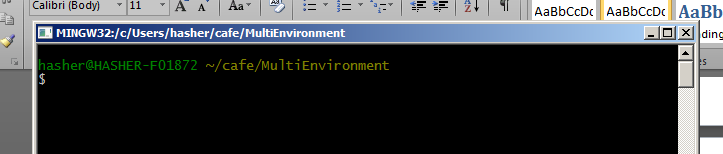


Figure 4 Git Bash Prompt

**Navigating Command and Bash Prompt**

Even though the cd (change directory) command works the same in both prompts, the command prompt used the backwards slash, and the bash command uses the forward slash (figure 5). The slashes are used to specific root and folder names. A nice feature of both commands is the use of autocomplete by using the tab key. If enough unique information is provided the autocomplete feature will fill the rest of the information, such as typing an Us then tab will fill in the rest, completing the word User, type Mult then tab and Multienvionment will be completed.

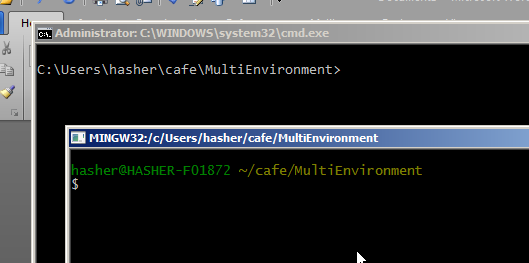


Figure 5 Navigating prompts

**Redirecting script output**

By default the output of the café scripts is directed to the results.html file that is located in the root folder of the script. But by using the –format html –o *outpufilename,* the output can be renamed and relocated if desired. For instance if Systest environment was used then the output could be –format html –o SystTestResults.html, or -–format html –o SystesOut.html. There are two must have parameters the –format html, must be present, no spaces between the two dashes and the word format, and the –o specifics output.



Figure 6 Redirecting script output

**Use of Tags**

By using tags in café scripts allows certain scenarios to be executed or not executed depending of how the tag is used when the café calls the script. A tag is defined inside the café script with a name proceeded by the @ (at sign), most commonly used tag is wip (work in progress), other could be regression, smoke, etc. The syntax would be @wip, @smoke, @regression, or @*anyname.* In figure 6, the tags are even and odd. To specific a scenario to execute use the tag name if skip the scenario use the ~ before the tag name. In figure 6 scenarios with @even are skipped, the rest are executed, then only the @odd are executed in the second statement.

**Bypass the Default Environment**

The default environment is specified in setting.yml by the parameter DEFAULT\_ENVIRONMENT, this can be overrode by specifying the parameter ENVIRONMENT=*testingenvironment*, when calling the café script from either the command or bash prompt. The *testingenvironment* however must be defined in the ENVORONMENT.YML file. In figure 6 the default environment is overrode with SysTest in line one, and B2 in line 2.

**Nuts and Bolts of Café batch files**

Café batch scripts can either be run from the bash or command prompt, but the command prompt need a little move coding, and the bash prompt has more features. The core of a café batch file is a group of café statements that could be executed individually, but through a batch file, can be executed with one click.

Scenarios and café scripts can be specified in a batch file by either using tags or by explicating naming the script inside the batch file. If neither tags nor names are not used, then the batch file will execute all the scripts inside the specification folder. Tags are used by using the –tag @tagsname, to include scenarios, or use the negative ~ to exclude scenarios. To explicit execute a script and all the screnarios inside the script the *features –name filename* is used where the filename is the name of the café script. Figure 7 illustrates an example where the café script *test4* is executed in the bash prompt, and output to *resultsfour.html*. Figure 8 illustrates the identical command but in the command prompt, notice the extra coding of the verb *call*

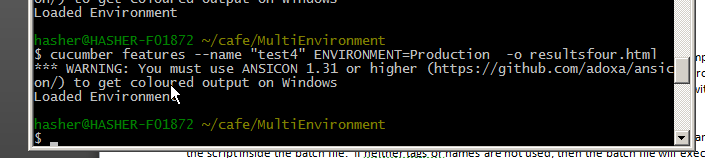


Figure 7 Example of using the --name option from the bash prompt

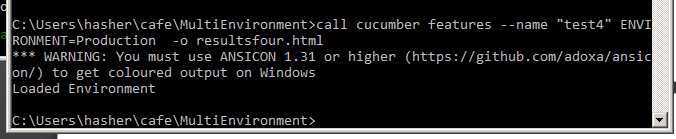


Figure 8 Example of using the -name option from the command prompt

If all the scripts need to be run or to execute scripts with specific tags omit the feature –name option such as in Figure 9, this example also has a redirect output file named *AllScripts.html*

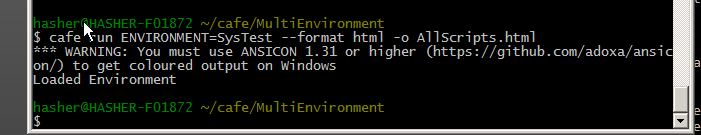


Figure 9 Execute all scripts in specification folder

Comments can be used in a café batch file, a # is used to comment statement running in the bash prompt and a ren is used with command prompt batch. Commented statements will be skipped by café during execution. Figure 10 shows a bash file with Line 2 being commented out whereas Firgure 11 is a command prompt batch file, with line 3 commented out, notice the extra Call verbs.

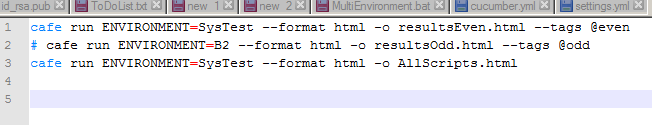


Figure 10 Batch file for Bash prompt

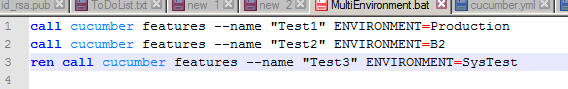


Figure 11 Batch file for Command Prompt

**Using Cucumber vs Café statements**

Café run command is identical to the cucumber command in execution, both will execute the gherkin scripts and produce output according to parameters, either output to the results.html, or to the redirected output using the –o option. Both have their unique command and options, and both can be used in café batch files.

**Building a Café batch file**

A café batch file is simply made up of individual statements, such as in figure 7, 8 , 9. It is important that each statement is valid and can be executed individually, or the batch file won’t work. Copy and paste the statement into either the bash or command prompt to valid the statement before adding to the batch file. Figures 10, and 11 shows a simple batch files for both bash and command prompts. Check on Github for an working example of the MultiEvnvironment project at…

Before the batch file can be successful, each individual statement in the batch must work, as well as each café script. If either fails then the batch run will fail. Double check that each café script is successful, and that each statement in the batch file works independently of the batch run.

**Conclusion**

* Difference between bash and command prompt
* How to redirect script output
* How to use tags
* How to override the default environment
* How to run a single script from batch files
* How to run all the script from batch files