

COMMAND TUBE

v2.0.5

Automated Yourself

Michael Han

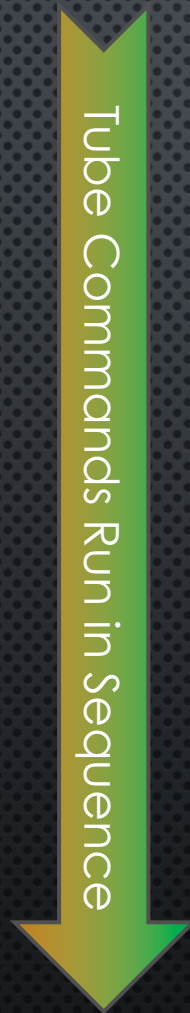
2022

Command Tube

Command Tube can run a group of sequenced commands

Each command has general arguments and self arguments

Easy access built-in help document



01

COMMAND TYPE: CONTENT
[--continue [m][n]] [--redo [m]]

02

COMMAND TYPE: CONTENT
[--continue [m][n]] [--redo [m]]

03

COMMAND TYPE: CONTENT
[--continue [m][n]] [--redo [m]]

04

RUN_TUBE: TUBE-1
[--continue [m][n]] [--redo [m]]

05

COMMAND TYPE: CONTENT
[--continue [m][n]] [--redo [m]]

06

COMMAND TYPE: CONTENT
[--continue [m][n]] [--redo [m]]




07

COMMAND TYPE: CONTENT
[--continue [m][n]] [--redo [m]]

08

...

Support 55+ command types for current
>>> tube help commands

-  Command run successfully.
-  Command is skipped.
-  Command run failed.

Tube Variables

Variables

root_folder	:	C:\workspace\fin-trunk\trunk
disk_drive_name	:	W
my-app-version	:	2022.0.0.1
line_number	:	100

Define Variables

- Manually added in the tube file
- Use SET tube command
- Updated via some tube commands

Pass Variables

- From terminal, use syntax '-v var1=value1, var2=value2' to pass variables from console to tube
- Pass variable to sub tube when use RUN tube command

Tube

PATH: {root_folder}
SET: x = 1
PRINT: Hello World --if x == 1
RUN: SubTube -v number = {line_number}
PRINT_VARS: x, root_folder

Use Variables

- Tube command content can use syntax {variable-name} as the placeholder
- In --if condition or --while condition and set tube command, you can omit the curly brackets

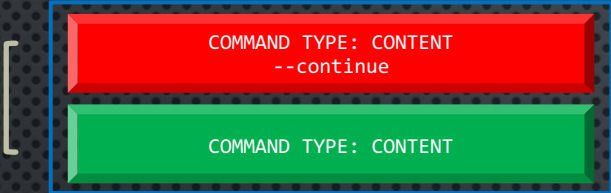
View Variables

- Use PRINT_VARS tube command
- From terminal, use syntax 'help vars' to view all variables in console

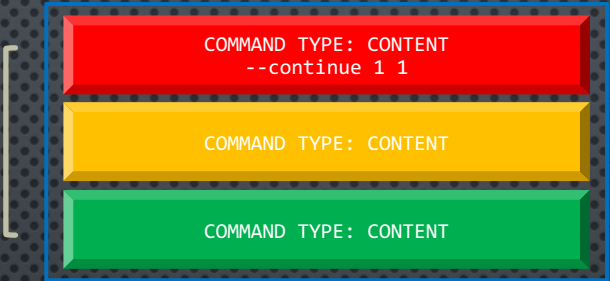
continue & redo

General Arguments I

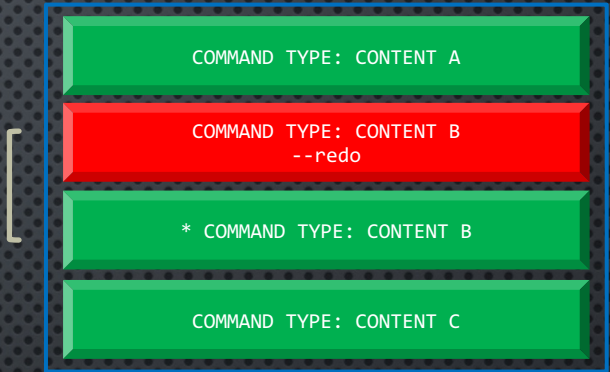
continue



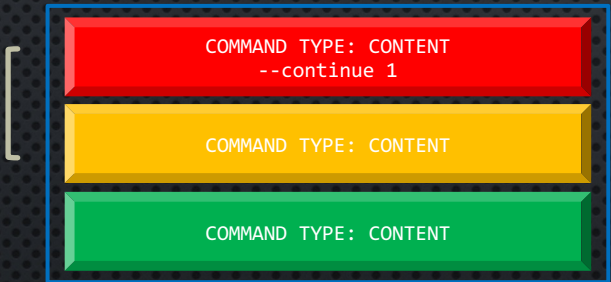
continue [m] [n]



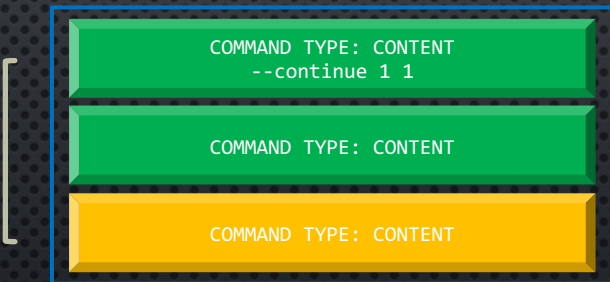
redo



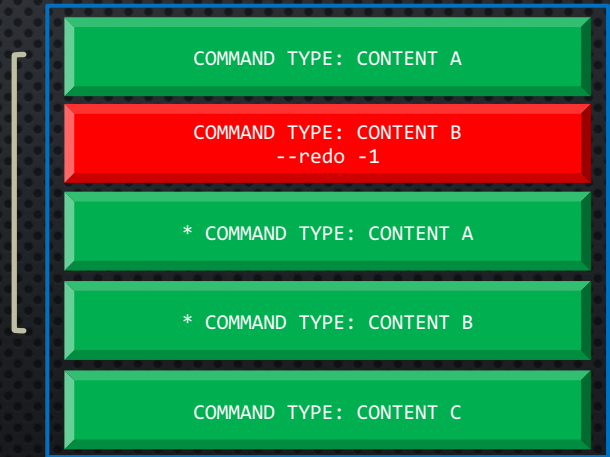
continue [m]



continue [m] [n]



redo [m]



--continue: Looks like IF..ELSE..




Decides which command will be skipped and which command executed, based on command execute result (SUCCESS/FAIL).

Syntax: --continue [m] [n]

--redo: Redo your command

Redo fail command or repeat success command multiple times

Syntax: --redo [m]

-  Command run successfully.
-  Command is skipped.
-  Command run failed.

General Arguments II

Variables

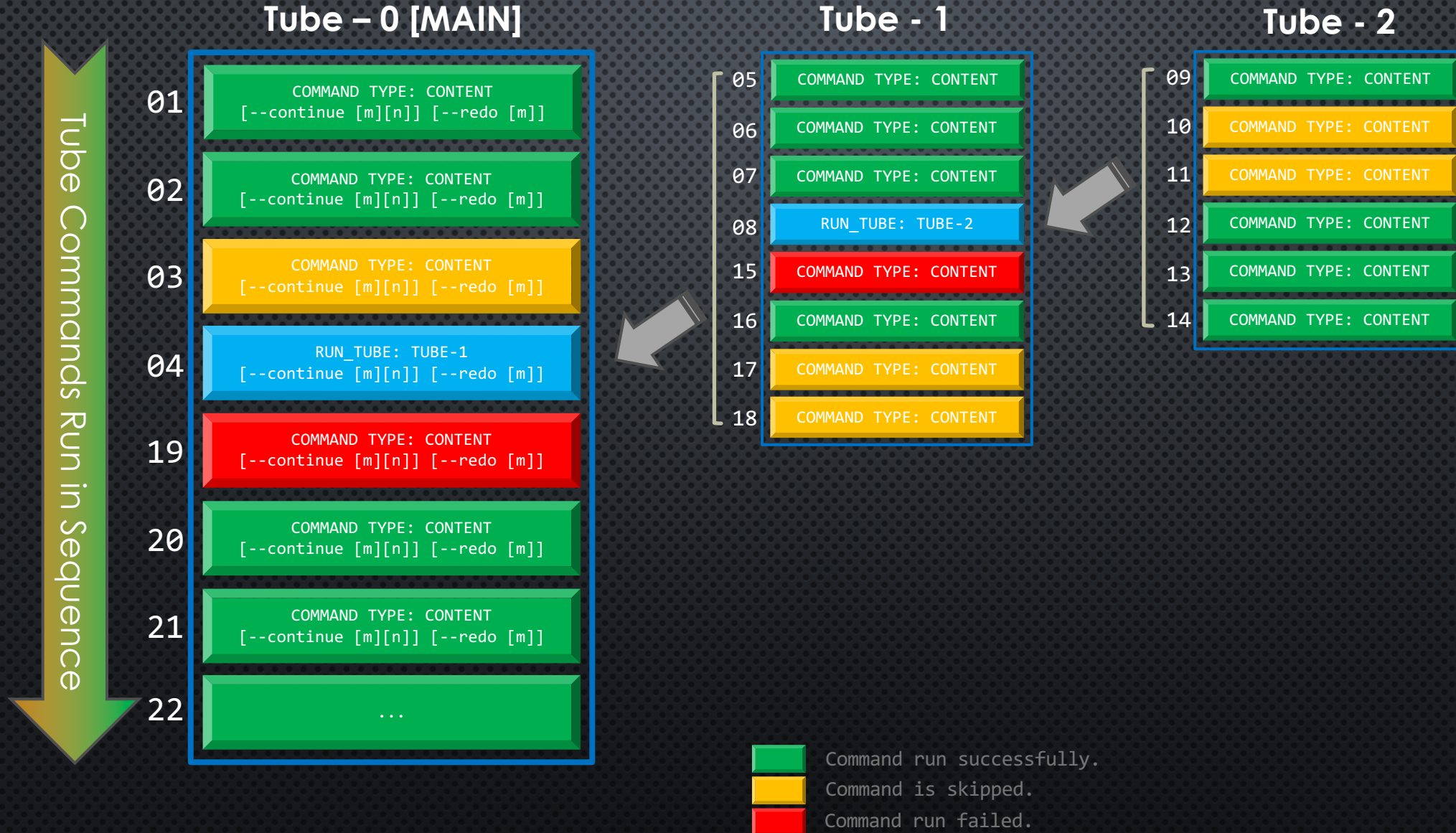
not_run	:	no
run_command	:	yes
run-type	:	server
s or S	:	SPACE-CHAR

Tube

PATH: C:\users --if not_run == no
PATH: C:\dev --if run_command
PATH: C:\dev --if not_run run_command
PATH: C:\dev --if "server"==run-type

--if: Check if run a command
The {variable-name} will be replaced by the variable value. Only the following values will skip the running: 'false', 'False', 'No', 'no'
Syntax: --if <condition>

Sub Tubes



Tube Examples

e.g. Hello World

```
Tube:  
- print: Hello World
```

e.g. If condition

```
Variables:  
  x: 1  
Tube:  
- print: Hello World --if x == 1
```

e.g. Run Sub Tube

```
Tube:  
- print: output from main tube  
- run: SubTube  
SubTube:  
- print: output from sub tube
```

e.g. Loop

```
Tube:  
- set: ls = ["hello", "world"]  
- run: SubTube --each i, item in ls  
SubTube:  
- print: >  
  I'm in loop: {i}, value: {item}
```


Run Command Tube

A: Run command tube via source code (SRC mode)

Run at once:

```
>>> python command-tube.py -t tube.yaml -f
```

Run at next 6:00 o'clock:

```
>>> python command-tube.py -t tube.yaml --datetime n6
```

Run at 6:00 for 7 days:

```
>>> python command-tube.py -t tube.yaml --datetime n6 -l 1d --times 7
```

Show arguments usage:

```
>>> python command-tube.py -h
```

```
>>> ...
```

B: Run programming tube via package (BIN mode)

Run at once:

```
>>> tube -t tube.yaml -f
```

```
>>> ...
```


More Samples

Please refer to:

Sample-refresh-dev.yaml

Sample-conditional-build.yaml

Sample-FFMPEG.yaml

under templates folder

Tube Commands List

[1]: BREAK	[26]: FILE_READ	[51]: SET_VARIABLE
[2]: CHECK_CHAR_EXISTS	[27]: FILE_SORT	[52]: SET_XML_TAG_TEXT
[3]: COMMAND	[28]: GET_FILE_KEY_VALUE	[53]: SFTP_GET
[4]: CONNECT	[29]: GET_XML_TAG_TEXT	[54]: SFTP_PUT
[5]: CONTINUE	[30]: IMPORT_MODULE	[55]: TAIL_FILE
[6]: COUNT	[31]: LINUX_COMMAND	[56]: WRITE_LINE_IN_FILE
[7]: CREATE_OBJECT	[32]: LIST_DIRS	
[8]: DELETE_LINE_IN_FILE	[33]: LIST_FILES	
[9]: DELETE_VARIABLE	[34]: PATH	
[10]: DIR_CREATE	[35]: PAUSE	
[11]: DIR_DELETE	[36]: PRINT	
[12]: DIR_EXIST	[37]: PRINT_VARIABLES	
[13]: EMAIL	[38]: READ_LINE_IN_FILE	
[14]: EXEC	[39]: REPLACE_CHAR	
[15]: FILE_APPEND	[40]: REPORT_PROGRESS	
[16]: FILE_COPY	[41]: REQUESTS_DELETE	
[17]: FILE_CREATE	[42]: REQUESTS_GET	
[18]: FILE_DELETE	[43]: REQUESTS_HEAD	
[19]: FILE_DOWNLOAD	[44]: REQUESTS_OPTIONS	
[20]: FILE_EMPTY	[45]: REQUESTS_PATCH	
[21]: FILE_EXIST	[46]: REQUESTS_POST	
[22]: FILE_INSERT	[47]: REQUESTS_PUT	
[23]: FILE_MOVE	[48]: RUN_TUBE	
[24]: FILE_POP	[49]: SET_FILE_KEY_VALUE	
[25]: FILE_PUSH	[50]: SET_TUBE	

View command details:
>>> tube help command-name