

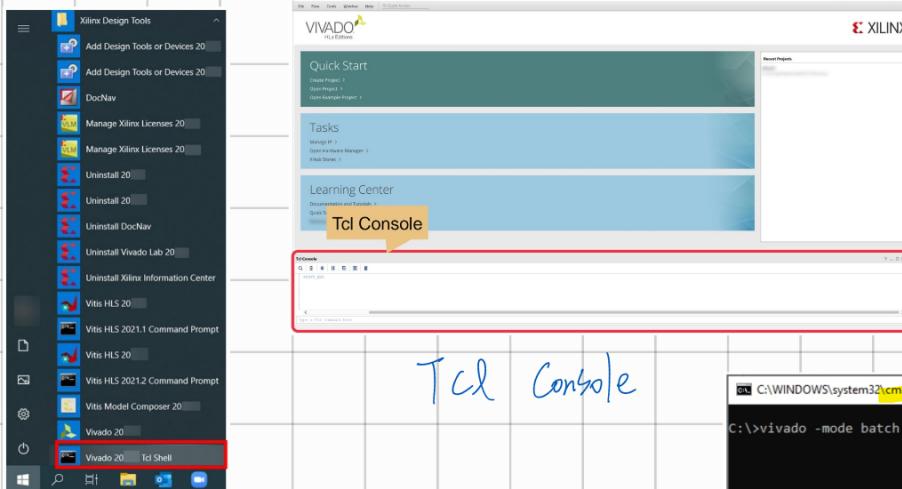
Tool Command Language

在 Vivado 中，並沒有支援所有的 tcl command，但有針對 synthesis, implementation 等等加入特定的 tcl

Safe Base	encoding	if	pid	switch	Safe Base	encoding	if	pid	switch
Tcl	eof	incr	pkg::create	tcl_endOfWord	Tcl	eof	incr	pid	switch
after	info	info	tcl_findLibrary		after	info	incr	pkg::create	tcl_endOfWord
append	interp	join	tcl_startOfNextWord		append	interp	info	pkg::mkIndex	tcl_findLibrary
array	eval	lappend	tcl_startOfPreviousWord		array	eval	join	tcl_startOfNextWord	
auto_execok	exec	pwd	tcl_wordBreakAfter		auto_execok	exec	lappend	tcl_startOfPreviousWord	
auto_import	exit	re_syntax	tcl_wordBreakBefore		auto_import	exit	pwd	tcl_wordBreakAfter	
auto_load	expr	read	tcltest		auto_load	expr	re_syntax	tcl_wordBreakBefore	
auto_mkindex	tblocked	tclvars			auto_mkindex	tblocked	read	tcltest	
auto_mkindex_old	fconfigure	list			auto_mkindex	fconfigure	tclvars	tclvars	
auto_qualify	length	load			auto_qualify	file	time	time	
auto_reset	load	lrange			auto_reset	fileevent	trace	trace	
bgerror	filename	lreplace			bgerror	filename	unknown	unknown	
binary	flush	lsearch			binary	flush	unset	unset	
break	for	lset			break	for	update	update	
catch	foreach	lsort			catch	foreach	variable	variable	
cd	format	memory			cd	format	vwait	vwait	
clock	gets	msgcat			clock	gets	while	while	
close	glob	namespace			close	glob	dde		
concat	global	open			concat	global			
continue	history	package			continue	history			
dde	http	pararray			dde	http			

The highlighted Tcl commands are supported in the Vivado IDE

可在三種情況下運行 tcl



Tcl Shell

```
C:\>vivado -mode batch -source my_tcl_script.tcl
```

Tcl Batch

基本語法範例

```
% expr 3+4
7
% set age 35
35
% puts "I am $age years old"
I am 35 years old
% Set age 42
Invalid command name "Set"
```

expr 算術運算
Result
設定變數與值
Result
puts 相當於在 Console 端 print
摘要數值
不光大寫，代表 tcl 有大小寫不同的 syntax

Script file grandpa.tcl

Variables can store text

set age 40

set name Jonathan

```

set name Jonathan
puts "Hello $name"
puts "Life begins at $age"

```

tcl裡面光寫好tcl

Results

```
C:\training> tclsh grandpa.tcl
```

```
Hello Jonathan
```

```
Life begins at 40
```

W↑ 透過tclsh執行或source xxx.tcl

Run Scripts

linux

Script file grandpa

```

#! /usr/local/bin/tclsh
set age 40
set name Jonathan
puts "Hello $name"
puts "Life begins at $age"

```

Specify which interpreter to use

Running the script

```

training@hannah: ~ chmod 555 grandpa
training@hannah: ~ grandpa

```

Anyone can read and execute

Windows

- 01 Give the script file a .tcl extension so Windows knows its type
- 02 Associate .tcl with the Tcl interpreter (tclsh)
- 03 Double-click the script file to execute it
- 02 Run tclsh from a command window
- 03 Supply the script file name on the command line
- 03 Run Tcl scripts for the Vivado IDE from within the Vivado tool

Command Substitution

```
% set Pi 3.142; set radius 10
```

```
10
```

```
% set area [expr $Pi * $radius * $radius]
```

```
314.2
```

Semicolon is command separator

將好幾句以為在同一句，並用分號分開

[]裡面代表可以抓好幾個變數做運算，並得到一值

Everything as far as the next] is treated as a command

Matching []

```
set a [set b ""]
```

Single] in quotes doesn't match []

```
% set ten 10 ; set twenty 20 ; set thirty 30
```

```
30
```

```
%
```

輸出只会选取後方跟後的值

Quoting

引號裡面公事處於Quoting

```
% set address "22 Market Place"
22 Market Place
% set fulladdress "$address; Ringwood"
22 Market Place; Ringwood
```

Word

變數在裡頭依然有效

Quoting with Braces

{ } 裡面無論什麼都視為 string

```
% set netname {data$bus[31]}
data$bus[31]
```

Word

```
% set Tcl/Tk 8.4
8.4
% set version ${Tcl/Tk}f
8.4f
```

Not recommended

不建議使用包含運算的多複合字作為變數

Backslash

```
% puts "Special characters: \"\n\t\""
Special characters: "
% puts {Not special: \"\n\t\"} 大括號內會視為 string
Not special: \"\n\t\"
```

No substitution inside braces

```
% puts "\tTab and \nnewline"
Tab and
newline
```

Similar to C backslash-escapes

空一大格 (4 or 8 spaces)

\n 有特別意思，代表換行

```
% puts {All on \
one line}
All on one line
```

Must be last character on line

Even inside braces and comments!

表示接續換行後的內容

Comments

使用井號作註解開頭

```
# this is a comment; and so is this
set multiline "Line 1
# But this is just Line 2 of a string! 這段在引號裡面會變 String
Line 3"
set a 5    # This is a syntax error
set b 6 ;  # but this is a valid comment
```

USE COMMENTS to document your scripts!

Tcl Capabilities in Vivado

Find all BUFG primitives:

```
set my_bufgs [get_cells -hier -filter {ref_name == BUFG}]
```

→ 提供更快速的方式來尋找設計中的元件、Constraints、運作等等作處理

Find all drivers(s) for every BUFG:

```
foreach x $my_bufgs { puts "$x: [all_fanin -flat -startpoints_only  
[get_pins -filter {direction == IN} -of [get_cells $x]]]"}
```

Report the ten longest paths from resp_data to bcd_out:

```
report_timing -from [get_cells cmd_parse_i0/send_resp_data*]  
-to [get_cells resp_gen_i0/to_bcd_i0/bcd_out*] -max_paths 10
```

Create a period constraint:

```
create_clock -name clk_pin_p -period 10.000 [get_ports clk_pin_p]
```

Execute programs:

```
place_design, synth_design
```

Tcl allows users to query a design database

Numerous reasons for searching objects in the design database

A common mechanism is used to search for all Vivado Design Suite objects

- Constraining the design
- Generating custom timing reports
- Finding objects in a schematic
- Creating custom schematics
- Setting properties on objects (primitive attributes, I/O standards, etc.)
- Manually placing cells (LOC)
- Floorplanning (Pblocks)
- Controlling the incremental router
- Performing minor changes (ECOs)

Tcl Initialization Scripts

啟動 Vivado 時，其會從三個位置查找 Script

1. 安裝位置

> local user directory for 特定 Vivado 版本

3. local user directory for 一般 Vivado 版本

安裝 Vivado 時並未提供 `Vivado_init.tcl`，需手動創建，因其是取代或增加原版 Vivado 指令的部份

