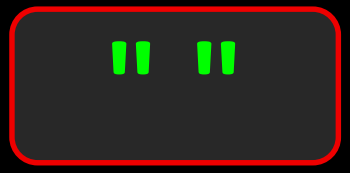
TCL VARIABLES MASTERCLASS

Understanding Substitution, Literals & Command Execution

Prepared By – Kunal Saraswat

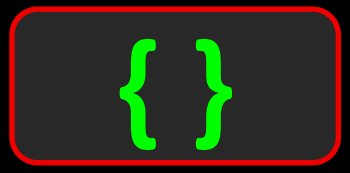
Follow Me on LinkedIn :
https://www.linkedin.com/in/kunalsaraswat/

TCL SYNTAX FUNDAMENTALS



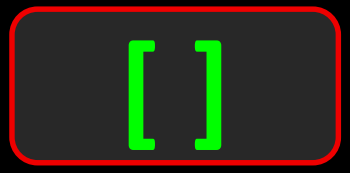
Double Quotes

Enables substitution of variables ($var) and command execution ([cmd])



Curly Braces

Pass content as-is (literal) - NO substitution or execution



Square Brackets

Execute content as a command and return the result

Command #1 of 10

TCL Terminal

```
$ set clock_signals "clk_core clk_io"
```

EXPLANATION:

Creates a variable 'clock_signals' with value 'clk_core clk_io'

OUTPUT:

```
clk_core clk_io
```

KEY CONCEPT:

Double quotes allow variable substitution and command execution within the string

Command #2 of 10

TCL Terminal

```
$ set main_clocks $clock_signals
```

EXPLANATION:

Assigns the VALUE of clock_signals to main_clocks using $ substitution

OUTPUT:

```
clk_core clk_io
```

KEY CONCEPT:

The $ symbol substitutes the variable with its actual value

Command #3 of 10

TCL Terminal

```
$ set literal_clocks {$clock_signals}
```

EXPLANATION:

Assigns the LITERAL string '$clock_signals' to literal_clocks

OUTPUT:

```
$clock_signals
```

KEY CONCEPT:

Curly braces prevent substitution - passes everything as-is (literal)

Command #4 of 10

TCL Terminal

```
$ #set exec_clocks [$clock_signals]
```

EXPLANATION:

COMMENTED OUT - Would try to execute 'clk_core clk_io' as a command

OUTPUT:

```
ERROR (if uncommented)
```

KEY CONCEPT:

Square brackets execute content as a command. This would fail because 'clk_core clk_io' is not a valid command

Command #5 of 10

TCL Terminal

```
$ set exec_clocks [list $clock_signals]
```

EXPLANATION:

Creates a proper TCL list with the value of clock_signals

OUTPUT:

```
clk_core clk_io
```

KEY CONCEPT:

[list ...] command creates a properly formatted list. Here it returns the substituted value

Command #6 of 10

TCL Terminal

```
$ set main_clocks "$clock_signals"
```

EXPLANATION:

Re-assigns main_clocks with clock_signals value (with quotes)

OUTPUT:

```
clk_core clk_io
```

KEY CONCEPT:

Similar to earlier - double quotes allow substitution, result is the same

Command #7 of 10

TCL Terminal

```
$ set report_cmd "report_timing $clock_signals"
```

EXPLANATION:

Assigns the LITERAL STRING 'report_timing clk_core clk_io' to report_cmd

OUTPUT:

```
report_timing clk_core clk_io
```

KEY CONCEPT:

Double quotes substitute $clock_signals but don't execute the command, creating a string

Command #8 of 10

TCL Terminal

```
$ set report_cmd [list {$clock_signals}]
```

EXPLANATION:

Creates a list containing the LITERAL string '$clock_signals'

OUTPUT:

```
$clock_signals
```

KEY CONCEPT:

The [list] command with curly braces inside preserves the literal text without substitution

Command #9 of 10

TCL Terminal

```
$ set report_cmd {report_timing $clock_signals}
```

EXPLANATION:

Assigns the LITERAL string 'report_timing $clock_signals' to report_cmd

OUTPUT:

```
report_timing $clock_signals
```

KEY CONCEPT:

Curly braces prevent any substitution - everything is literal

Command #10 of 10

TCL Terminal

```
$ set report_cmd [puts $clock_signals]
```

EXPLANATION:

EXECUTES 'puts clk_core clk_io' and assigns its return value to report_cmd

OUTPUT:

```
Prints: clk_core clk_io
```

Variable report_cmd = (empty string)

KEY CONCEPT:

Square brackets execute the command. 'puts' prints to console but returns empty string

QUICK REFERENCE TABLE

Command	Output	Key Point
`set clock_signals "clk_core clk_io"`	`clk_core clk_io`	**" " = Substitution**
`set main_clocks $clock_signals`	`clk_core clk_io`	**$ = Variable value**
`set literal_clocks {$clock_signals}`	`$clock_signals`	**{ } = Literal (no sub)**
`#set exec_clocks [$clock_signals]`	`ERROR`	**[ ] = Execute cmd**
`set exec_clocks [list $clock_signals]`	`clk_core clk_io`	**[list] = Make list**
`set main_clocks "$clock_signals"`	`clk_core clk_io`	**" " = Substitution**
`set report_cmd "report_timing $clock_signals"`	`report_timing clk_core clk_io`	**String (not exec)**
`set report_cmd [list {$clock_signals}]`	`$clock_signals`	**List with literal**
`set report_cmd {report_timing $clock_signals}`	`report_timing $clock_signals`	**{ } = No parsing**
`set report_cmd [puts $clock_signals]`	`clk_core clk_io (printed)`	**[ ] = Execute now**