/\* Script Language Allows C-Style comments. Code-lines end with semicolon or LF. \*/

// Directives

#Title 'A real script'

**#Include** 'path/file.scr'

// Variables

$var = 'hello there'

$num = 2

$var = [ Object Dictionary Notation X=Y, Y, “X”=Y ]

$var = @range(1,5)

GOTO Label

Label:

GUI-OBJECT = { IDENT|\* [LITERAL:]expr], [LITERAL=expr] }

// a GUI-object is a first-class language element -- { **GuiType** [*criteria name=value*] }

$myMenu = { Menu 'RNA|Partition Function RNA' }

$text = { **TextField** ***label***='File Save' }

$custom = { **GUI** ***type***='CustomGUIType', ***parent***=$myPanel, ***text***='/RNA .\*/' }

// GUI-object searches can be refined using relationships with other objects

$button = { **Panel** 'Settings' } => { **Button** 'Save' } //button is a descendant\* of panel

$button = { **Panel** 'Settings' } ==> { **Button** 'Save' } //button is a direct child of panel

$button = { **Button** 'Save' } inside { **Panel** 'Settings' } //button is a direct child of panel

$button = { **Button** 'Save' } childof { **Panel** 'Settings' } //button is a direct child of panel

\*in the sense of component containment (not object inheritance) i.e. Given a panel that contains a sub-panel that itself contains a button, the button and sub-panel are both descendants of the main panel. Similarly the main panel and the sub-panel are both ancestors of the button.

$button = $container **->** { **Panel** $panelName }

TEST: [“TestName“=]EXPR [, EXPR, EXPR]

FAIL: statements;;

PASS: statements ;;

// Call built-in (global) methods

Warning 'Could not locate Button!'

Info 'Running Partition Function.'

Abort Reason

repeat 1..6 as $i

;;

REPEAT UNTIL EXPR

REPEAT WHILE EXPR

FOR $range as $var : Statement (use colon for 1-line statement)

LOOP (require if colon is not present)

FOR $range as $key=>$val :

;;

if $button is null goto X ||

if ($button is null)

goto X

else

goto Y

end

.. range

=== (direct equality)

Obj.Func

@Action [param1, param2, …] [preposition: into on at in with of] subject

--Determine if action has been defined for subject

As an addon or member

FUNC params obj 🡪 obj.Func(params)

i.e. func is called on last expression within params.

ACTION: SUBJECT IDENT EXPR

ACTION PARAM : SUBJECT IDENT EXPR : EXPR

ACTION [PARAMS] : SUBJECT IDENT EXPR, EXPR : EXPR

// Gui-Actions: ACTION [(*parameters* …)] {GUI-Object}

**CLICK:** $myMenu //same as $myMenu->Click()

**TYPE ‘**bmorivector.fasta' : { **TextField** 'File Save' } // $myMenu.Type( ['bmorivector.fasta'] )

**MOUSE (RIGHT, DOWN, x=50, y=50)**: { **CustomGUIScreen** ***title***=’Alignment’ } *inside* $dialog;

**MOUSEMOVE (x=0, y=25)** : GUI ('a', 'b', 45e+7) inside $guiRef

**SELECT** { **Option** 'RNA' }

//Define and call custom methods

ShowSaveDialog 'bmorivector.fasta'

ShowSaveDialog ('bmorivector.fasta')

If there is an identified subject (via colon) then look for direct members

//Call as a function ShowSaveDialog('bmorivector.fasta')

#define ShowSaveDialog($fileName)

**CLICK** { **Menu** ‘File|Save As’ }

**TYPE** $fileName INTO { **TextField** 'File Name' }

**CLICK** { **Button** ‘Save’ }

#end

//Function ‘pointers’

$myFunc = ‘ShowSaveDialog’;  
@$myFunc 'bmorivector.fasta'

---------------------------- Command-line interface ------------------------------------------

**GuiTest** ‘ScriptFile1’ ‘ScriptFile2’ … *(prefix with* ***@*** *for inline code)*

**GuiTest** **@**'**#Include** ‘design-test-script’**;** **$param1**=42; **$out**=‘design-out1.txt’; @**runtest**;'