Netfabb Lua Scripting API Reference

Version: December 2019

Table of Content

1 Lua in Netfabb

Lua is Netfabb's method of choice to facilitate automation tasks inside the software. Lua is currently available inside Netfabb’s range of software products:

- As scripting language for Autodesk Netfabb Ultimate
    - As “Desktop Automation” module
    - Specific to the Slice Commander
    - Specific to the 3S module

1.1 API scopes

In the later API reference part, the different scope of the API calls will be indicated by these tags:

[Desktop Automation]

and

[3S] [Slice Commander]

1.1.1 [3S]

The 3S Lua implementation is accessible as the 3S “script” function and its API only contain the functions documented in this chapter: 2.16. No other Netfabb API calls are available in this implementation. Vice versa, the 3S API functionality is only available in the 3S module and nowhere else.

1.1.2 [Slice Commander]

The Slice Commander module has the option to load a Lua script file via a menu option: Prepare->Run Lua Script. The API this Lua API interface provides is nearly identical with the one for [Desktop Automation]. The only addition is a default “slice” object in the realm of the script, which provides convenient access to the currently loaded slice stack.

1.1.3 [Desktop Automation]

This framework is accessible in Netfabb Ultimate via the 'Lua Script Library' under Prepare->Run LUA Script. It targets people, who want to automate repeatable jobs inside Netfabb. The API also allows functionality extending the out-of-the-box features of Netfabb by more custom workflows. One can also start Netfabb Ultimate with a commandline parameter("/startluascript=<Scriptfile>"), which allows to execute the given lua script after the start of netfabb.

1.2 Supported Lua functionality

Netfabb supports generally the version 5.1 of the Lua language definition, see: http://www.lua.org/manual/5.1/

From the Lua 5.1 standard Netfabb supports the core language specification and some but not all the libraries.

Lua Library	Supported	Comment
Base	Yes, with exceptions	Some exeptions: • dofile:no – use system:executescriptfile() • print: no: use system:log()
Coroutine Manipulation	Yes	
Modules	No	Use system:executescriptfile()
String manipulation	Yes	
Table manipulation	Yes	
Mathematical Functions	Yes	
Input and output Facilities	No	
Operating System Facilities	No	
The Debug library	No	[Desktop Automation] provides a debugger GUI.

1.3 Lua Scripting in Autodesk Netfabb Ultimate

One can access the scripting cababilities of Autodesk Netfabb only in the Ultimate Edition. One can execute general Lua scripts in the main Lua Automation Module in the default module Autodesk Netfabb ([Desktop Automation]) and special scripts working on slices in the [Slice Commander] of Autodesk Netfabb. In addition, the [3S] Module can also execute Lua scripts. However, we refer to the Manual for a description of that feature.

1.3.1 Main Lua Automation Module

[Desktop Automation]

The Lua Automation Module module can be found under -Prepare-Execute Lua Script. In the Lua Automation module, a Lua script can be loaded, saved and manually edited. The syntax can be checked before the script is executed. All the main Lua commands can be accessed within the script. This excludes the special Lua commands for the 3S Module. The Lua Automation module can be used without any connections to the current content of the Desktop application by loading, manipulation and saving again an arbitrary FABBPROJECT. However, in most cases one wants to manipulate the content of the current project opened in Autodesk Netfabb. Therefore, the variable “tray” is predefined with the current platform (and its LUATray object). This allows a direct and fast access to all meshes in the platform and allows to manipulate them. Several example scripts are availiable. In addition, the global variable “netfabbapplication” allows access to the trays of the current running netfabb application.

1.3.2 Lua Execution in Slice Commander

[Slice Commander]

In the slice commander a right click on the Slice opens the Pop-Up menue. Under Extended the entry “Execute Lua script” can be found. Here an script can be directly loaded and immediately executed. The variable “slice” is predefined and corresponds to the selected slice(s). For each selected slice the script is executed. Please note that any changes needs to returned with a new slice using *system:addslicetotree* . The “script SliceCommanderLUAScript_example1_Offset” demonstrate this.

2 Scripting Reference

2.1 General Lua Syntax

The Lua Scripting Language is a compact bytecode programming language, capable of most common language constructions. While including a variety of functionally for example for mathematical calculations or string manipulations, its automatic conversion of data-types guarantees a shallow learning curve and quick results. Therefore, it is also widely used in a lot of applications (e.g. Apache HTTP Server, Adobe Photoshop, Video Lan Client) and especially in commercial games (e.g. Baldur's Gate, World of Warcraft, Crysis). A detailed reference and a general introduction can be found on the Lua website (***http://www.lua.org***). A very comprehensive and recommended book is "Programming in Lua", which is also available on the Web (***http://www.lua.org***). There is also a vivid community, which provides a large source for information and examples.

Netfabb's functions are included by Lua's basic object interface. While not delivering many sophisticated techniques like creating inheritances of classes, there is enough functionality to create a good interface for three-dimensional mesh handling. As a rule, all occurring properties of object instances can be used like generic variables in the code. A simple access by "**object.*property***" is sufficient. Object methods are handled somehow uncommon, since they are characterized by a colon, so the general syntax is always like

result = object:method (param1, param2, param3); *- - generic method call*

2.2 System object

The central connection between Netfabb and the executed Lua script is the system object. As a global variable, it is accessible from anywhere in the script and is the starting point for the creation of all other objects - like meshes, xml files or database connections.

2.2.1 Properties

[Desktop Automation]

Property	Read / Write	Type	Description
buildnumber	read only	string	Build number (e.g. 396)
corecount	read only	number	Number of processor cores
exitcode	read only	number	Exit code
isquiet	read only	Boolean	returns if the logging to stdout is disabled
linebreak	read only	string	Line break characters
majorversion	read only	number	returns the major version of Netfabb (e.g. 5)
minorversion	read only	number	returns the minor version of Netfabb (e.g. 2)
officialversion	read only	string	Returns the official number version (e.g. 2019)

paramcount	read only	number	returns the number of command line parameters
result	read only	Mesh	Resulting Mesh object
timer	read only	number	returns the number of milliseconds since the Lua script has started
versionstring	read only	String	returns the version string of the system (e.g. '5.2.1')
unixtime	read only	number	returns the current time as unix timestamp

[Desktop Automation]

Property	Read / Write	Type	Description
areaunitfactor	read only	number	Returns the conversion factor of the current set area unit to the internal area unit (cm^2): 100 or 25.4*25.4
areaunitstring	read only	String	Returns the string representing the currently set area unit: cm^2 or in^2
lengthunitfactor	read only	number	Returns the conversion factor of the current set length unit to the internal length unit (mm): 1 or 25.4
lengthunitstring	read only	String	Returns the string representing the currently set length unit: mm or in
volumeunitfactor	read only	number	Returns the conversion factor of the current set volume unit to the internal volume unit (cm^3): 1000 or 25.4*25.4*25.4
volumeunitstring	read only	String	Returns the string representing the currently set volume unit: cm^3 or in^3

2.2.2 'system' Method Overview

2.2.2.1 System and file functions

[Desktop Automation]

Name	Syntax	Description
calculatemd5	Md5 = system:calculatemd5(filename:string);	Calculates MD5 hash sum of a file
checklibrary	Result = system:checklibrary(libname:string);	Checks for the desired library
cleargarbage	system:cleargarbage();	Calls the garbage collector
create3mfimporter	AImporter = system: create3mfimporter(AFileName: String; ASplitMeshed: Boolean; AName: String; ATray: TLUATray);	Create an importer for 3mf files. The second paratemeter specifies whether a single mesh should be imported. See **Error! Reference source not found.** for details
create3mfexporter	AExporter = system:create3mfexporter();	Create a new 3mf exporter object. See 3mfExporter for details

createcollisiondetector	ADetector = system:createcollisiondetector(gridsize: Number);	Create a Lua collision detector object. See CollisionDetector for details
createdirectory	Result = system:createdirectory(dirname:string);	Creates a directory
createformatteduuid	UUID = system:createuuid();	Returns a UUID, formatted
creategraph	Graph = system:creategraph(graphs:integer)	Creates a Graph object. Graphs is the number of individual graphs to be painted.
createhistogram	Histogram = system:createhistogram()	Creates a Histogram object. See histogram section.
createprimitivelist	List = system:createprimitivelist()	Create a list-object that contains all available primitives. See primitive section.
createhexbaseplate	ABaseplateMesh = system: createhexbaseplate(AHoleCountX: Number, AHoleCountY: Number, ARadius: Number; AHeight: Number; AWallThickness: Number);	Creates a hex baseplate mesh with a given number of holes in X- and Y-Direction and with a Hole Radius. The Height Parameter defines how high the baseplate should be, the last parameter defines the used wall thickness
createimageprocessing	Ip = system:createimageprocessing();	Creates an image processing object.
createrectbaseplate	ABaseplateMesh = system: createrectbaseplate(AHoleCountX: Number, AHoleCountY: Number; ACellSizeX: Number; ACellSizeY: Number; AHeight: Number; AWallThickness: Number);	Create a baseplate with rectangles. The first two parameters define the number of holes, the next two the size of the holes and the last two the height of the baseplate and the wall thickness
createreportgenerator	Result = system:createreportgenerator(Snapshot : LUAsnapshotcreator);	Creates a LUAReportgenerator, function call needs to have a snapshotcreator as an argument [Desktop Automation]

createscreenshot	system:createscreenshot(AWidth: Number; AHeight: Number; AOptions: JsonObject)	Create a screenshot of the current camera position and returns an *Image Object*. The “AOptions” object can contain the following keys with “true” / “false” values: - show_horizontalruler - show_verticalruler - show_labels - show_viewcube - show_coordsystem - show_platform - show_textures - show_colors - zoom - camera: a json object with 3 optional members “eye”, “up” and “*center*” defining the position, view target and up vector of the camera as array with 3 float elements With these values you can control the Netfabb UI elements to be visible or invisible in the generated image [Desktop Automation]
createsnapshotcreator	Result = system:createsnapshotcreator();	Creates a LUASnapshotcreator. This is used by other objects to create a snapshot. [Desktop Automation]
createstamper	Result = system:createstamper();	Create a LUAStamper, which allows to label meshes [Desktop Automation]
createstringmap	Map = system:createstringmap()	Create a string map for key / value pairs
createuuid	UUID = system:createuuid();	Returns a UUID
createzip	zipobject = system:createzip(zipfile: string)	Creates a new zipfile object. If used without zifile parameter, it creates the object in memory only. See Zipobject.
directoryexists	boolean = system:directoryexists(dirname:string);	Returns TRUE, when directory exists
downloadurl	Result:boolean = System:downloadurl(URL: string, name: string);	Tries to download a file via HTTP GET from ‘URL’ and saves it as ‘name’. Returns TRUE if successful.
excludepathdelimiter	String = system:excludepathdelimiter(inputstring:string);	Returns string excluding the system-specific path delimiter (at the end)

executeapplication	system:executeapplication(app:string [;options:string; wait: boolean]);	Starts ‘app’ application and waits until application ends (blocking). ‘options’ are separated by blank. The optional “wait” parameter defined whether or not Netfabb will wait (true, default) for the external application to finish execution
executescript	system:executescript(script: string, scriptname : string);	Execute a string as a lua script. The scriptname is the name of the script, which should be associated with the script (needed for the lua debugger)
executescriptfile	system:executescriptfile(scriptname:string);	Executes another Lua Script. Scriptname is the name of the lua script on the filesystem.
extractfileext	Filenameext = system:extractfilenameext(inputstring:string);	Returns the file name extension from a string
extractfilename	Filename = system:extractfilename(inputstring:string);	Returns the filename only from a string
extractfilepath	String = system:extractfilepath(inputstring; trailingpathdelimiter: bool)	Returns a string with the file path. If trailingpathdelimiter is true the trailing path delimiter are included, otherwise not
fileexists	boolean = system:fileexists(filename:string);	Returns TRUE, when file exists
formatarea	Result:string = system:formatarea(numval:number [,digits:integer]);	Returns ‘numval’, formatted with number of ’digits’ and trailing currently set area unit [Desktop Automation]
formatlength	Result:string = system:formatlength(numval:number [,digits:integer]);	Returns ‘numval’, formatted with number of ’digits’ and trailing currently set length unit [Desktop Automation]
formattraffic	Result:string = system:formatraffic(numval:number);	Returns ‘numval’ as formatted byte string, e.g.: 3.4 kB [Desktop Automation]
formatvolume	Result:string = system:formatarea(numval:number [,digits:integer]);	Returns ‘numval’, formatted with number of ’digits’ and trailing currently set volume unit [Desktop Automation]
getallfilesindirectory	xmlfilelist = system:getallfilesindirectory (dirname:string);	Returns a xml file with all files in a directory [Desktop Automation]
getdatestring	date = system:getdatestring();	Returns current date as a string
getfilesize	Size = system:getfilesize(filename:string);	Returns the size of a file
gettimestring	time = system:gettimestring();	Returns current time as a string

hideprogressdlg	System: hideprogressdlg();	Hides the progress dialog created by system:showprogressdlg [Desktop Automation]
includepathdelimiter	String = system:includepathdelimiter(inputstring:string);	Returns string with system-specific path delimiter (at the end)
inputdlg	system:inputdlg(Title:string,Label:string,Defaultvalue:string);	Creates a generic dialog for input of one (text) field as well as an OK and a Cancel button. OK returns whatever is currently written in the field (including the default value), Cancel always returns the default value. All fields are mandatory but can be empty. [Desktop Automation]
log	system:log(logsting:string);	Logs logstring to current log output channel. See also: setloggingtooglwindow
logtofile	system:logtofile(filename:string; timestamp: Boolean = true);	Log Standard Output to file. If timestamp is false to time information is printed. The default value is true.
messagedlg	system:messagedlg(message: String);	Show a message dialog [Desktop Automation]
openurl	system:openurl(url: string)	Open a URL in the default browser [Desktop Automation]
openzip	system:openzip(file: String; password: String)	Open an existing ZIP file for reading. Returns an instance of ZipObject. The password parmeter is optional and is used to decrypt password protected archives. [Desktop Automation]
passworddlg	system:passworddlg(Label:string,password:string);	Creates a dialog for inputting a password, hiding the input [Desktop Automation]
registerextension	system:registerextension(Extension: String, LocalizationId: String, Filter: String, DescriptionLocalizationId: String)	Register a new extension to Netfabb. Extension is the extension without the . (“stl”). LocalizationId is the translated string describing the filter (“STL File”). Filter is the filter to actually apply (“*.stl”) [Desktop Automation]
safealphanumeric	Result = system:safealphanumeric(inputstring:string;withpunctuationsigns:18oolean);	Checks and returns a string for standard alphanumeric signs. Boolean flag = TRUE allows also punctuation signs (‘.’,’,’,-‘)
setloggingtooglwindow	system:setloggingtooglwindow(value);	Available for the Lua Automation module. Allows that all output is piped to the OGL warning window. *Value* is a boolean

setprogress	system:setprogress(Percent, Message, [translate]);	Set the progress dialog with a progress value and a message. The third parameter (optional) selects whether or not the message should be translated (true = default) [Desktop Automation]
shellexecute	system:shellexecute(cmd:string, param:string, doWait:19oolean, Show:19oolean, directory:string)	Executes a shell command "*cmd*" with the shell parameters "param" doWait: if TRUE, waits until the command has finished Show: set to "true" if you want the launched application to be shown. "False" is default directory: change to this directory for command execution Example/note for windows: system:shellexecute("cmd", "/c copy cube.stl test.stl",1, 1);
showdirectoryselectdialog	Directorystring = system: showdirectoryselectdialog(bAllowcreate:19oolean, bPerformCreate:19oolean, bPrompt:19oolean)	Opens a directory selection dialog. Parameters: **bAllowcreate**: An edit box allows the user to type in the name of a directory that does not exist. This option does not create a directory: the application must read the name of the selected directory and create it if desired. **bPerformCreate**: Used only in combination with bAllowCreate. If the user enters a directory name that does not exist, the directory selection dialog creates it. **bPrompt**: Used only in combination with bAllowCreate. Displays a message box that informs the user when the entered directory does not exist and asks if the directory should be created. If the user chooses OK, the directory is created if the option set includes bPerformCreate. If the option set does not include bPerformCreate, the directory is not created: the application must read the directory name and create it. Returns string with directory name or empty, if nothing selected [Desktop Automation]

showopendialog	Filename = system:showopendialog(Aextension: String)	Shows a file open dialog. Only one extension kann be passed. If the extension is unknown to Netfabb it can be registered using "system:registerextension". If a file was selected its name is returned, otherwise an empty string is returned. [Desktop Automation]
showprogressdlg	system:showprogressdlg(defaultcallback: Boolean);	Shows the progress dialog [Desktop Automation]
showsavedialog	filename_to_save = system:showsavedialog(ext:string, [ext2: string]);	Open file save dialog with parameter for the file extension, returns filename. The second, optional parameter defines the possible extensions for the save file, the first the default ending. [Desktop Automation]
showsavedialogex	filename_to_save = system:showsavedialogex(ext:string, ext2: string, [name: string]);	Open file save dialog with parameter for the file extension, returns filename. The 2nd parameter defines the possible extensions of the filename, the 1st the default.The third, optional parameter suggests a filename for the save. [Desktop Automation]
slicemesh	system:slicemesh(mesh, layersize, fromZ, toZ, createInMemory:20oolean);	Slices a mesh with the given parameters. Function returns a LUASlice Object
tonumber_safe	system:tonumber_save(value:string, name:string);	Registers "name" as global variable and stores "value" as float in it. [Desktop Automation]
yesnodlg	Result = system:yesnodlg(message: String, [withCancel: 20oolean]);	Displays a "Yes/No" dialog with "message" as text. The optional "withCancel option determines, whether also a "Cancel" option should be shown. Result: 1 for Yes, 0 for No, -1 for other. [Desktop Automation]

Examples

Name	Example	Return value
log	system:log("*Hello World*");	-
logtofile	system:logtofile("*my_file.log*");	-
getparam	filename_to_process = system:getparam(0);	*string*: string containing the desired parameter
shellexecute	system:shellexecute("*mail*", -s subject "*hello@mail.com*")	-
checklibrary	system:checklibrary("*mysql*");	-

showsavedialog	local filename = system:showsavedialog (“wrl”);	String: with the file name, or empty string
createscreenshot	shotJson=”{“show_verticalruler”:true, “show_labels”:true, “show_colors”:true, “show_viewcube”:true, “show_platform”:true, “show_coordsystem”:true, “show_horizontalruler”:true, “show_textures”:true, “camera”:{“zoom”:1, “eye”:[0,1,0], “center”:[0,0,0], “up”:[0,0,1]}}” local shot=system:createscreenshot(400, 300, shotJson)	Image object
inputdlg	local userinput = system:inputdlg("Rename the part","New name:","");	string

2.2.2.2 Mesh Creation and Mesh Loading

[Desktop Automation]

Name	Syntax	Description
createheightmapmesh	Mesh = system:createheightmapmesh(filename:string; width:integer, height:integer; depth:integer; invert:boolean; mingray:integer);	Creates a mesh from a JPG, PNG or BMP heightmapfile in the defined dimension. The mingray defines the minimal grey value between 0 and 255. Invert defines, whether the image should be inverted
createimagemesh	Mesh = system:createimagemesh(filename:string; width:integer, height:integer; depth:integer; treshhold:integer);	Creates a mesh from a JPG, PNG or BMP file in the defined dimension. The threshold is the defined grey value for separation between 0 and 255.
createtextmesh	Mesh=system:createtextmesh(text : string, width:integer, height:integer; depth:integer)	Creates a mesh from a string. The x,y,z size is given by width, height and depth.
createmesh	system:createmesh();	Creates a mesh with no triangles
load3ds	system:load3ds(filename:string);	Loads an 3DS file
load3mf	system:load3mf(filename:string);	Loads an 3MF file
loadamf	system:loadamf(filename:string);	Loads an AMF file
loadgts	system:loadgts(filename:string);	Loads an GTS file
loadncm	system:loadncm(filename:string);	Loads an NCM file
loadobj	system:loadobj(filename:string);	Loads an OBJ file
loadply	system:loadply(filename:string);	Loads an PLY file
loadstl	system:loadstl(filename:string);	Loads an STL file
loadvoxel	system:loadvoxel(filename:string);	Loads an SVX file
loadvrml	system:loadvrml(filename:string);	Loads an VRML file
loadx3d	system:loadx3d(filename:string);	Loads an X3D file
loadzpr	system:loadzpr(filename:string);	Loads an ZPR file

Examples

Name	Example	Return value
createmesh	mesh = system:createmesh();	*mesh object*: an empty mesh object
load3ds	system:loadgts("*test.3ds*");	*mesh object*: a mesh object of the "Autodesk 3D Modeling Format" file
load3mf	system:load3mf("*test.3mf*");	*mesh object*: a mesh object of the "3MF Basic Microsoft" file
loadamf	system:loadamf("*test.amf*");	*mesh object*: a mesh object of the "Additive Manufacturing File"
loadgts	system:loadgts("*test.gts*");	*mesh object*: a mesh object of the "Gnu Tesselated Surfaces" file
loadncm	system:loadncm("*test.ncm*");	*mesh object*: a mesh object of the "Netfabb Compressed Mesh" file
loadobj	system:loadobj("*test.obj*");	*mesh object*: a mesh object of the "Wave Front OBJ" file
loadply	system:loadply("*test.ply*");	*mesh object*: a mesh object of the "Stanford Polygon" file
loadstl	system:loadstl("*test.stl*");	*mesh object*: a mesh object of the "Surface Tesselation Language" file
loadvoxel	system:loadvoxel("*test.svx*");	*mesh object*: a mesh object of the "Simple Voxels" file
loadvrml	system:loavrml("*test.vrml*");	*mesh object*: a mesh object of the "Virtual Reality Modeling Language" file
loadx3d	system:loadx3d("*test.x3d*");	*mesh object*: a mesh object of the "Extensible 3D-ASCII" file

2.2.2.3 XML, Json, CSV, Text Handling and Database connections

[Desktop Automation]

Name	Syntax	Description
connecttoodbc	DBConnection = system:connecttoodbc(connectionstring (DSN), login, password : all strings);	Connects to an ODBC datasource [Desktop Automation]
createcsv	CSVObject = system:createcsv(clearexisting:boolean)	Creates a CSVObject, if flag is TRUE, a subsequent write will overwrite any existing file.
createjson	Jsonfile = system:createjson();	Creates an empty Json file
createtextfile	textfile = system:createtextfile();	Creates an empty text file
createxml	Xmlfile = system:createxml();	Creates an empty XML file
loadjson	Jsonfile = system:loadjson(filename:String);	Loads a Json file from a file
loadjsonfromurl	Jsonfile = system:loadjsonfromurl(URL:String);	Loads a Json file from a URL
loadtextfile	textfile = system:loadtextfile(filename:string);	Loads a text file from disk

loadxml	Xmlfile = system:loadxml(filename:string; AUseAttributesAsChildren: Boolean);	Loads an XML file from disk. If AUseAttributesAsChildren is true, attributes are used as children. The default value is true.
loadxmlfromstring		Loads an XML file from an String
loadxmlfromurl	Xmlfile = system:loadxml(URL:string);	Loads an XML file from an URL

Examples

Name	Example	Return value
createxml	system:createxml();	*XML object:* XML file object for further processing
loadxml	system:loadxml("i*nput.xml*");	*XML object:* XML file object for further processing
connecttoodbc	system:connecttoodbc("netfabb", "user", "pw");	*DB Object*: Database object for further processing

2.2.3 Constants

The Netfabb LUA system object contains some constants:

Name	Type	Description
stCLI	Number	Constant for the "sliceobject:savetofile" type parameter for CLI slice files
stCLS	Number	Constant for the "sliceobject:savetofile" type parameter for CLS slice files
stSLC	Number	Constant for the "sliceobject:savetofile" type parameter for SLC slice files
stSLI	Number	Constant for the "sliceobject:savetofile" type parameter for SLI slice files
stUSF	Number	Constant for the "sliceobject:savetofile" type parameter for USF slice files

2.2.4 Further functionality Methods

2.2.4.1 GL Context

[Desktop Automation]

Name	Syntax	Description
createoglcontext	GLContext = system:createoglcontext(width: integer; heigth: integer; AXServer: string)	Creates GL Context with xidth x height size. AXServer is an optional argument

2.2.4.2 Testsuite Framework

[Desktop Automation]

Name	Syntax	Description
createtestsuite	subsuite = system:createtestsuite(suitename: string)	Creates a new testsuite with specified name.

2.2.4.3 Slicing

2.2.4.3.1 Method overview

[Desktop Automation]

Name	Syntax	Description
addslicetotree	system: addslicetotree(slice:SliceObject);	Adds a slice to the current stack
createemptylayer	SliceLayer = system:createemptylayer();	Returns an empty slicelayer object
createsliceexporter	SliceExporter = system: createsliceexporter(type:String)	Creates a SliceExporter object. Type must be: “cls”.
createslicelist	SliceListObject = system: createslicelist();	Creates a new empty SliceListOject
creategroupedslice	Slice = system:creategroupedslice(SliceObject|SliceListObject, …);	Takes a variable list of either Slice or SliceList Objects and returns a combined Slice object
createslicebuilder	SliceBuilder = system: createslicebuilder(layercount:Integer, Layersize:Number, MinZ: Number);	Creates a Slicebuilder Object
createsliceintersection	Slice = system:createsliceintersection (SliceObject|SliceListObject, …);	Creates an intersection of Slice or SliceList objects and returns a slice object
intersectslices	Slice = system: intersectslices (SliceObject|SliceListObject, …);	Intersects Slice or SliceList objects and returns a single object
loadslice	Slice = system:loadslice(filename:string; type: integer)	Loads a slice from file. Types are the following: 0 = USF 1 = CLI 2 = SLI 3 = CLS 4 = SLC
unifyslices	Slice = system:unifyslices (SliceObject|SliceListObject, …);	Unifies Slice or SliceList objects and returns a slice object

2.2.4.3.2 Properties

None.

2.2.4.4 CadImport

[Desktop Automation]

Name	Syntax	Description
createtecadimport	cadimporter = system:createcadimport()	Creates a new Importer for CAD Files

2.2.4.5 Fabbproject

[Desktop Automation]

Name	Syntax	Description
loadfabbproject()	fabbproject = system:loadfabbproject(filename: String);	Open Fabbproject
newfabbproject()	Newfabbproject = system:newfabbproject()	Create new and empty Fabbproject

2.2.5 3mfExporter

[Desktop Automation]

This class can be used to export .3mf files from a Lua script. An instance is created with the call "system:create3mfexporter();".

2.2.5.1 Properties

Property	Read / Write	Type	Description
writetextures	Read / Write	Boolean	Flag to switch export textures to the 3mf on or off
writematerials	Read / Write	Boolean	Flag to switch export materials to the 3mf on or off
writecolors	Read / Write	Boolean	Flag to switch export colors to the 3mf on or off

2.2.5.2 Method Overview

Name	Syntax	Description
addattachment	Exporter:addattachment(Name: String; Content: String; Namespace: String)	Sets the attachment of the 3mf to be exported
exporttofile	Exporter:exporttofile(Name: String)	Exports the content to a file
setscenethumbnail	Exporter:setscenethumbnail(Thumbnail: Object)	Sets an Image Object as thumbnail for the 3mf

2.2.6 CollisionDetector

[Desktop Automation]

This class is used to detect collisions between two meshes of the build room. It is created with the call "Detector = system:createcollisiondetector(RasterSize);".

2.2.6.1 Properties

None.

2.2.6.2 Method Overview

Name	Syntax	Description
checkmeshcollision	CollisionDetector:checkmeshcollision(Mesh1: Object; Mesh2: Object)	Checks the collision between 2 LUATrayMesh objects,

		returns “0” if now collision was found, a value bigger than 0 otherwise

2.3 Netfabbtrayhandler

[Desktop Automation]

The “netfabbtrayhandler” is not a LUA object which can be created and released. It is a global variable, which is only available in the Desktop Automation Module. It allows access to the trays and to create a new tray in the application.

2.3.1 Properties

Property	Read / Write	Type	Description
traycount	Read	Number	Gives the current number of trays

2.3.2 Method Overview

Name	Syntax	Description
addtray	netfabbtrayhandler:addtray(name: String, machinesize_x: Number, machinesize_y: Number, machinesize_z: Number)	Adds a new tray to the netfabbapplication. “Name” is the name of the tray, machinesize_x, _y, and _z are the sizes of the tray.
gettray	netfabbtrayhandler:gettray(index: Integer)	Retrieve a tray from a netfabbapplication
removetray	Netfabbtrayhandler:removetray(index: Integer)	Removes the tray with the corresponding index. Workspace trays, and the main tray cannot be removed. Also the tray needs to be empty. Function returns 0, if the tray was successfully removed. Otherwise 1 (index not valid), 2 (main tray was tried to be removed), 3 (workspace tray was tried to be removed, 4 (tray was not empty), 5 (internal error).

2.4 Application

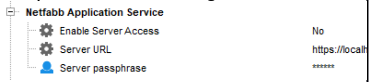
[Desktop Automation]

The “application” is not a LUA object which can be created and released. It is a global variable, which is only avalible in the Desktop Automation Module. It allows more access to the current running netfabb desktop application. In addition, it also can create a GUI out of a dialog. The dialog is the same object as described in Chapter 2.7.

2.4.1 Properties

Property	Read / Write	Type	Description
lengthunitstring	Read	String	Returns the current unit as string
showcolortexture	Write	Boolean	Set if color/texture should be displayed
tier	Read	Number	Gives the current tier

2.4.2 Method Overview

Name	Syntax	Description
createdialog	dialog = application:createdialog();	Create a DialogObject.
createtaskhandler	taskhandler = system:createtaskhandler (URL: String, name:String, userid:String, serverkey: String)	Creates a Netfabb taskhandler object. URL is e.g.: http://localhost:8651/tasks/”, name is the worker displayname, userid to be used and the server key
getenvironmentvariable	Result = application: getenvironmentvariable(Variable : string)	Returns a environment variable as a string
getluascriptfromlibrary	Result = application:getluascriptfromlibrary(scriptname: string);	Returns a lua script as a string from the lua library. The name must match the name in the lua library.
gettimestamp	application:gettimestamp();	Gets current time in seconds since 1.1.1970.
loadappserverhublist	Hublist = application: loadappserverhublist(URL:String, key:String);	Loads the list of available hubs for the current (logged-in) user from the Netfabb Application Server. URL is the server connect string, default: 'http://127.0.0.1:8650/', respective the setting:  Netfabb Application Service Enable Server Access No Server URL https://localh Server passphrase ****** Key ist the option passphrase, default is empty

loadappserverproject	Result:boolean = application:loadappserverproject(URL: String, projectid:String, folderid:String, itemid:String, [userid:String, passphrase:String]);	Loads a project from a local server into the application. This removes any previously loaded project. Userid and passphrase are optional parameters. Returns true is successful.
loadforgehublist	Hublist = application:loadforgehublist()	Loads the list of available hubs for the current (logged-in) user from Autodesk Forge.
overwriteexistingappserverproject	Return:Boolean = application: overwriteexistingappserverproject(Url: String, projectid:String,folderid:string, itemname:string, [userid:string, passphrase:string]);	Saves the current application project on local server, overwrites an exsiting project. Returns true if successful.
savenewappserverproject	Result:boolean application:savenewappserverproject(Url: String, projectid:String,folderid:string, itemname:string, [userid:string, passphrase:string]);	Saves the current project in the application onto a local server. Returns true if successful. Itemname is the name, the project should have.
savecurrentproject	Return:boolean = Application:savecurrentproject();	Saves the current project in the application, returns true when done.
savefabbproject	Result:Boolean = application: savefabbproject (AFilename : string)	Saves the current state of netfabb as fabbproject. The filename gives the saved name of the file.
triggerdesktopevent	application: triggerdesktopevent ('updateparts');	Triggers a event. The event 'updateparts' triggers an update of the displayed meshes in netfabb

2.5 Mesh Object (TLUAMesh)

[Desktop Automation]

A mesh object contains the full information about a triangulated mesh, and may be used for altering, transforming and exporting the mesh data.

2.5.1 Properties

Property	Read / Write	Type	Description
area	read only	Number	Returns the area
badedgecount	read	Number	Get the number of bad edges of the mesh
boundaryedgecount	read	Number	Get the number of boundary edges of the mesh
boundarylength	read	Number	Get the length of the boundary of the mesh
calculationerror	read only	Boolean	Returns whether the last calculation has failed
calculationfailed	read only	String	Returns the message for a failed calculation
edgecount	read only	Number	Returns the number of edges

facecount	read only	Number	Returns the number of triangles
flippedtriangles	read	Number	Get the number of flipped triangles of the mesh
holecount	read only	Number	Returns the number of holes
ismanifold	read only	Boolean	True, if the mesh has a closed surface
isnice	read only	Boolean	Calculates if the mesh is manifold (Deprecated)
isok	read only	Boolean	True, if the mesh is a) 'ismanifold', b) 'isoriented' and c) has a positive volume
isorientable	read only	Boolean	True, if flipped triangles can be repaired
isoriented	read only	Boolean	True, if the mesh has no flipped triangles
loadingfailed	read only	Boolean	Returns, whether a load has failed
loadingerror	read only	String	Returns the loading error
nodecount	read only	Number	Returns the number of nodes/points
outboxvolume	read only	Number	Returns the mesh outbox volume (in mm^3)
outboxbasearea	read only	Number	Returns the mesh outbox base area (in mm^2)
outboxheight	read only	Number	Returns the mesh outbox height (in mm)
shadowarea	read	Number	Get the shadow area of the mesh
shellcount	read only	Number	Returns the number of shells
volume	read only	Number	Returns the volume (in mm^3)

2.5.2 Method Overview

2.5.2.1 Format conversion

Name	Syntax	Description
saveto3ds	mesh:saveto3ds(filename:string);	Saves the mesh as a 3DS file
saveto3mf	mesh:saveto3mf(filename:string);	Saves the mesh as a 3MF file
savetoamf	mesh:savetoamf(filename:string;binary:boolean);	Saves the mesh as a AMF file
savetoasciistl	mesh:savetoasciistl(filname:string, name:string);	Saves the mesh as a ASCII STL file
savetogts	mesh:savetogts(filename:string);	Saves the mesh as a GTS file
savetoncm	mesh:savetoncm(filename:string);	Saves the mesh as a NCM file

savetoobj	mesh:savetoobj(filename:string);	Saves the mesh as a OBJ file
savetoply	mesh:savetoply(filename:string);	Saves the mesh as a PLY file
savetostl	mesh:savetostl(filename:string);	Saves the mesh as a binary STL file
savetovoxel	mesh:savetovoxel(filename:string);	Saves the mesh as a Voxel file
savetovrml	mesh:savetovrml(filename:string;fixlengtheachrow:boolean; endmarkeachrow:boolean);	Saves the mesh as a VRML file
savetox3d	mesh:savetox3d(filename:string, binary:boolean);	Saves the mesh as a X3D file
savetozpr	mesh:saveto3ds(filename:string);	Saves the mesh as a ZPR file
shellasmesh	Newmesh = mesh:shellasmesh(shellnumber: Number)	Extract a shell as new mesh object. Same as "LUATrayMesh.shellasmesh" but without the matrix of a traymesh. See "LUAMeshObject.shellcount" for the number of meshes

All routines return TRUE if successful.

Examples

Name	Syntax	Return value
savetostl	mesh:savetostl("*my_mesh_fixed.stl*");	TRUE if successful
savetoasciistl	mesh:savetoasciistl("*my_fixed_ascii.stl*", "*My fixed ASCII STL*");	TRUE if successful
savetox3d	mesh:savetox3d("*my_mesh_fixed.x3d*", true);	TRUE if successful
savetogts	mesh:savetogts("*my_mesh_fixed.gts*");	TRUE if successful
savetoobj	mesh:savetoobj("*my_mesh_fixed.obj*");	TRUE if successful
savetoncm	mesh:savetoncm("*my_mesh_fixed.ncm*");	TRUE if successful

2.5.2.2 Mesh Transformation

Name	Syntax	Description
addtriangle	Mesh:addtriangle(P1_X, P1_Y, P1_Z, P2_X, P2_Y, P2_Z, P3_X, P3_Y, P3_Z: all number);	Adds a triangle to a mesh
applymatrix	mesh:applymatrix(matrix : LUAMatrix4f);	Applies a 4 times 4 transformation matrix to a mesh. The matrix is passed as LUAMatrix4f object.
calcoutbox	Outbox = mesh:calcoutbox();	Calculates the outbox of a mesh, return Outbox object.
create_partorienter	orienter = mesh:create_partorienter()	Creates and returns new orienter object for this mesh, see PartOrienter
dupe	Newmesh = mesh:dupe();	Creates a copy of the mesh
invert	mesh:invert();	Inverts the orientation of all triangles

merge	mesh:merge(anothermesh:string);	Adds another mesh into the mesh
move	mesh:move(x:number, y: number, z: number);	Translates the mesh by a given vector
movetoorigin	mesh:movetoorigin();	Translates the mesh to the origin
minimizeoutbox	Mesh:minimizeoutbox(mode : string);	Rotates the mesh to minimize the volume or the height and the base area of the mesh outbox. Permitted values of the parameter mode are Volume – the volume of the outbox is minimized; VolumeFlat - the volume of the outbox is minimized and the mesh is oriented in such a way that the shortest edges of the outbox are parallel to the axis z. HeightBase – the height of the outbox is minimized then the outbox base area is minimized at the constant outbox height (i.e. by a rotation around the axis z).
release	mesh:release();	Releases the memory of a mesh. It cannot be accessed after this method
rotate	mesh:rotate(x: number, y: number, z: number, angle: number);	Rotates the mesh by a given angle [degree] around a given axis
scale	mesh:scale(x: number, y: number, z:number);	Scales the mesh by a given factor. Center is the point of origin
zcompensation	Mesh:zcompensation(value : integer);	Applies Z-Compensation to Mesh, Value in mm.

Examples

Name	**Example**	**Return value**
movetoorigin	mesh:movetoorigin();	-
move	mesh:move(*200, 50, -100*);	-
scale	mesh:scale(*0.01*); *[if y and z are not specified, x value is taken]*	-
rotate	mesh:rotate(*1, 2, 0.5, 90*);	-
calcoutbox	my_outbos = mesh:calcoutbox();	*Outbox object*: object representing the outbox of the mesh
invert	mesh:invert();	-
release	mesh:release();	-

dupe	mesh:dupe();	*Mesh object*: returns a copy of the mesh. **Attention**: may use a lot of memory
merge	mesh:merge(*mesh_of_inner_shell*);	-

2.5.2.3 Analyse meshes

Name	Syntax	Description
calculatehausdorffdistanceto	mesh:calculatehausdorffdistanceto(mesh2:string);	Returns the Hausdorffdistance
checksanity	mesh:checksanity(mesh2:string);	Returns TRUE if the mesh is sane
comparewith	mesh:comparewith(mesh2:string, hausdorffdistancethreshold:number, distancethreshold:number, fraction:number in %);	Checks if the mesh is different to mesh2. This is done in two steps: if the haussdorffdistance is larger than the given *hausdorffdistancethreshold* the tests fail or a given *fraction* of the measured distances is larger the *distancethreshold*
createanalyzer	TLUAPartAnalysis analysis = mesh: createanalyzer ()	Creates a new partanalysis object which allows to run different analyses and stores their results
createalignment	TLUAAlignment alignment = mesh:createalignment(mesh2:meshobject, matrix1, matrix2: luamatrix)	Creates a new alignment object which stores the principal axes and the resulting matrix. See 4.28 for a detailed property list mesh2: the mesh that can be transformed with the resulting matrix to overlap the given mesh. Matrix1, matrix2: optional parameters, used as modeltoword matrices when set
fastwallthicknesstest	Boolean result = mesh:fastwallthicknesstest(criticaldistance:number, criticalsurface:number in %;colormesh: boolean);	Checks the wallthickness of a mesh, as above, but stops after fail condition is recognized. If colormesh is true, the results of the wallthickness will be written into the color information of the mesh.
getbuildvolume	mesh:getBuildVolume(offset:number)	Returns the value of the inner buildvolume in mm³. The parameter *offset* gives the thickness of the surface. If the calculation fails, -1 is returned.
getcenterofgravity	Center = mesh:getcenterofgravity();	Returns the center of gravity of the mesh as LUAVector3

getdownskinarea	Areasum = mesh:getdownskinarea(*downskinangle: number*);	Calculates the downskin area. Downskinangle is given in degree (90 - support angle as specified in the Analysis/Up-/Downskin analysis dialog). Units are mm^2 (=cm^2*100)
getsupportvolume	mesh:getsupportvolume(criticleangle:number);	Returns the volume of the support in mm³. The area, which needs support, is calculated using the parameter *criticalangle*. The normal of a triangle with respect to the z-axis defines the critical angle, i.e. a small critical angle means that only a small area needs support. If the calculation fails, -1 is returned.
gettotalcontourlength	mesh:gettotalcontourlength(layersize)	Returns the value of the length of all outer contourlengts in mm. The parameter *layersize* gives the thickness of the layersize. If the calculation fails, -1 is returned.
getupskinarea	Areasum = mesh:getupskinarea(*downskinangle: number*);	Calculates the upskin area
trappedpowderanalysis	result = mesh:trappedpowderanalysis(Holesize: number)	Returns true if powder would be trapped. Holesize gives the sizes of holes which are not large enough to remove the powder, e.g. all holes smaller than this size are ignored. The default value is 0.
wallthicknesstest	Boolean result = mesh:wallthicknesstest(criticaldistance:number, criticalsurface:number in %; colormesh: Boolean; AMulticore: Boolean);	Checks the wallthickness of a mesh. Returns true if the wallthickness is not smaller than criticaldistance for more than the criticalsurface area. If colormesh is true, the results of the wallthickness will be written into the color information of the mesh. If AMulticore is true, multithreading is used.
wallthicknesstestwithoutput	TLuaXML result = mesh:wallthicknesstestwithoutput (criticaldistance:number, criticalsurface:number in %; colormesh: Boolean; AMulticore: Boolean);	The parameters are the same as in wallthicknesstest, however here a xml is returned with the following informations: 'testpassed', 'totalarea', 'testprob', 'areabelowthreshold', 'numberofclusters', 'areaoflargestcluster'

Examples

Name	Example	Return value
wallthicknesstest	wallThicknessPassed = mesh:wallthicknesstest(*0.4, 20*);	*boolean*: returns **true** if the area which has a thinner wallthickness as *criticaldistance* [mm] is less than *criticalsurface* [%] of the total area.
comparewith	testresult = mesh:comparewith(*mesh2, 0.4, 0.1, 10*);	*integer :* For each point of mesh the closest distance to mesh2 is calculated. This distribution of distances is afterward analysed. The distribution is used for two tests. **Test1**: If only distance is larger than *distance* test1 fails. **Test2**: if a *fraction* of the distance distribution is larger than *threshold* test2 fails. Return 0 if meshes are equal, returns 2 if test1 fails, returns 4 if test2 fails, and returns 6 is both test fails.
calculatehausdorffdistanceto	hasudorffdistance = mesh:calculatehausdorffdistanceto(*mesh2*);	*number*: For each point of the mesh the closest distance to mesh2 is calculated. The maximum of these distances, which called Hausdorffdistance, is returned.

2.5.2.4 Boolean Operations

Name	Syntax	Description
bool	Result = mesh:bool(asecondmesh: TLUAMesh; operationtype: number);	Bools the current mesh with a second. If operationtype is 0 the mesh is added, if it is 1 the mesh is subtracted.
insertselfintersections	mesh:insertselfintersections(epsilon:number);	Splits up a mesh along its self intersections, same as: Self-intersections: Split off in Netfabb Professionals Repair actions
intersect	mesh:intersect();	Works similar to the unify function only that it builds the intersect of 2 merged meshes

unify	mesh:unify(epsilon:number);	Unifies shells of a nice mesh. This is actually the 3d boolean function of Netfabb. This does the actual operation of fusing 2 meshes. As an "add" function, if you just merge 2 meshes, as a "substract" operation, when you invert one mesh before. The epsilon parameter is syntactically required, but internally ignored (this was originally a precision parameter, but that has been changed with a scaling algorithm. The syntax is for backward compatibility)
wrap	mesh:wrap();	Wraps a mesh. Only the outer part of a mesh is left. Uses a parallel multicore algorithm
wrapsinglecore	mesh:wrapsinglecore();	Wraps a mesh. Only the outer part of a mesh is left. Uses the old singlecore algorithm

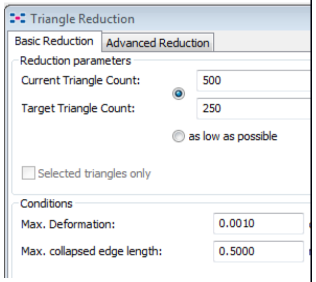
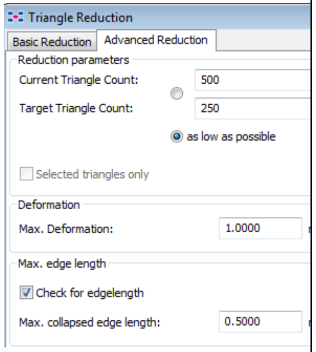
Examples

Name	Example	Return value
unify	found_self_intersections = mesh:unify(*0.05*);	*boolean:* returns if any self-intersections have been found (or if they have been disregarded due to the epsilon value). Dependent on the mesh geometry, this function could create a lot of sall shells, which should be cleared by ”removeghostshells”
intersect	found_self_intersections = mesh:intersect();	boolean: Similar to unify
insertselfintersections	found_self_intersections = mesh:insertselfintersections(*0.05*);	*boolean:* returns if any self-intersections have been found (or if they have been disregarded due to the epsilon value). **Attention**: This function may create a lot of shells, but it is capable of orienting not-orienable surfaces.
wrap	mesh:wrap()	-

2.5.2.5 Mesh Repair

Name	Syntax	Description
closetrivialholes	mesh:closetrivialholes();	Close trivial holes
closeallholes	mesh:closeallholes(timeout:number);	Closes all non-trivial holes. Timout is an optional timeout parameter (in seconds)

cut	Newmesh= mesh:cut(plane:number;height:number;topbottom:number);	Cuts the mesh enlong the x,y, or z plane. Plane 0 corresponds to the z-y plane, 1 to the z-x plane and 2 to the x-y plane. The height gives the position of the plane, topbottom can be either 1, then the top mesh will be returned or 2, then the bottom mesh will be returned. Topbottom is optional, default is that the top mesh is returned.
fixflippedtriangles	mesh:fixflippedtriangles();	Tries to create a uniform mesh orientation
invertnegativeshells	mesh:invertnegativeshells();	Inverts all shells with negative volumes
hollow	Newmesh = mesh:hollow(offset:number, rastersize:number, smoothen:boolean);	Returns a mesh, which is by the offset smaller than the original mesh. The rastersize is the resolution. The smoothing only occurs at the resolution limit. Offset and rastersize should both be positive (in mm). The newly created part has the right orientation to allow the hollowing: in order to hollow an actual part the meshes still have to be merged.
Join	nesh:join();	If the mesh consists of two open shells, which contours are nearby by the mesh is connected. The feature is called “Join contours” in the mesh repair module. Please note that the selection of triangles is done automatically and may not not be correct.
makeorientable	mesh:makeorientable();	Makes Mesh orientable
meshnodesparsification	mesh:meshnodesparsification(Epsilon:number, DegenerationUnits:number);	Sparsificated nodes of a mesh
offset	Newmesh = mesh:offset(offset:number, rastersize:number, smoothen:Boolean; inneroffset: boolean);	Returns a mesh, which is by the offset smaller/larger than the original mesh. The rastersize is the resolution. The smoothing only occurs at the resolution limit. Offset and rastersize should both be positive (in mm). If innerset is true, an inneroffset will be generated, otherwise an outer offset.
preparetopologyforexport	mesh:preparetopologyforexport();	Prepares topology for STL export

reducelod	mesh:reducelod(target:number, deformation:number, edgelength:number);	Reduces number of trianglesFrom Netfabb Pro:  Reduction parameters Current Triangle Count: 500 Target Triangle Count: 250 as low as possible Selected triangles only Conditions Max. Deformation: 0.0010 Max. collapsed edge length: 0.5000 Target: Target Triangle count Deformation: Max. Deformation (cm3) edgelength: Max. Collapsed edge length
reducelodadvanced	mesh:reducelodadvanced(count:number, deformation:number, edgelength:number);	Reduces number of triangles From Netfabb Desktop:  Reduction parameters Current Triangle Count: 500 Target Triangle Count: 250 as low as possible Selected triangles only Deformation Max. Deformation: 1.0000 Max. edge length ☑ Check for edgelength Max. collapsed edge length: 0.5000 Target: Target Triangle count Deformation: Max. Deformation (cm3) edgelength: Max. Collapsed edge length
removedegeneratefaces	mesh:removedegeneratefaces(tol:number);	Removes degenerated faces
removedoublefaces	mesh:removedoublefaces();	Removes double triangles from the mesh
removeghostshells	mesh:removeghostshells(epsilon:number);	Remove small shells
removenegativeshells	mesh:removenegativeshells();	Remove shells with negative volume
removeproblemareas	mesh:removeproblemareas();	Removes tiny faces which are causing many self-intersections
repairenhanced	mesh:repairenhanced();	Same as the Netfabb Professional "default" repair makro script
repairextended	mesh:repairextended();	Same as the Netfabb Professional "extended" repair makro script
repairsimple	mesh:repairsimple();	Same as the Netfabb Professional "simple" repair makro script
stitch	mesh:stitch(tol:number, preserveorientation:boolean);	Stitches open triangle borders

Examples

Name	Example	Return value
removedoublefaces	removed_count = mesh:removedoublefaces();	*integer:* Returns how many triangles have been removed. Two triangles are "double" if they share exactly the same corners. If they share slightly the same corners, this function does not harm them. Stitching and degenerate face removal couldalign these cases.
closetrivialholes	trivial_hole_count = mesh:closetrivialholes();	*integer:* Returns the number of trivially closed holes of the mesh, i.e. all missing triangles or quadrangles. This function does not alter the mesh in any other way and is safe to call anytime.
closeallholes	hole_count = mesh:closeallholes();	*integer:* Returns the number of closed holes of the mesh, i.e. all nontrivial and non-bad holes. This function can be dangerous to call and add unwanted walls into the mesh.
fixflippedtriangles	mesh_is_orientable = mesh:fixflippedtriangles();	*boolean:* returns if the full mesh has been made oriented. This does not mean that the shell as a whole has a positive volume.
invertnegativeshells	negative _shell_count = mesh:invertnegativeshells();	*integer:* number of shells inverted
removedegeneratefaces	mesh:removedegeneratefaces(*0.02*);	-
stitch	mesh:stitch(*0.01*, true);	*integer* numbers of edges having been stitched

2.5.2.6 Mesh Textures

Name	Syntax	Description
hastexture	mesh:hastexture();	Returns a Boolean value, if a mesh has a texture.
savetexturetofile	mesh:savetexturetofile(filepath: string)	Saves the texture of a mesh into a jpg file. Attention: Fileextension is not needed in the filepath as it will be added automatically.

2.5.2.7 Model Package

Name	Syntax	Description

createmodelpackage	mesh:createmodelpackage(barwidth, barthickness, partspacing, gridsizexy, gridsizez: number; text: string);	Creates a model package for the current mesh. The text is printed on the package. The default values 0.8 mm for bar width, 0.8 mm for barthickness, 1.2mm for part spacing, 10 mm for gridsizexy, and 10 mm for gridsizez. The result is the package mesh.

2.5.2.8 Mesh Support

The API has only one method, which uses an automated support definition, which can be generated by the Netfabb Desktop software.

Name	Syntax	Description
createsupport	support = stl:createsupport(xmlfile: string)	Creates automatic Support based on instructions in XML parameter file.

Example:

```
stl = system:loadstl('file.stl')
support = stl:createsupport('support.xml')
support:savetostl('support.stl')
```

2.5.3 Relation to "LUAMeshObject" and "LUATrayMesh"

TLUAMeshObject

[Desktop Automation]

The TLUAMeshObject is the Lua representation of the "TNGM_Mesh_Advanced" class. All modification functions are applied directly to the mesh

"TLUAMeshObject" is used in the Desktop Automation (through "TLU-ATrayMesh").

TLUATrayMesh

[Desktop Automation]

The TLUATrayMesh is the Lua representation of the "TNGM_MeshTreeMesh" class. It contains a TLUAMeshObject as member "FLUAMesh" published in the property "TLUATrayMesh.Mesh" and exported to Lua with the member "LU-ATrayMesh.mesh". Please note that the "LUATrayMesh.mesh is only a copy of the original LUAMesh in the Lua Automation module. All modification functions of the LUATrayMesh do not modify the mesh but only its properties in the tree.

"*TLUATrayMesh*" is used in ***Lua Automation Module***

2.6 Fabbproject API

[Desktop Automation]

The fabbproject API allows to create, load, modify and save fabbproject files used by Netfabb.

2.6.1 Fabbproject Object

[Desktop Automation]

This is the central class of the fabbproject API. It holds references to the trays within the project and allows saving. The methods "System:loadfabbproject" and "System:newfabbproject" (Fabbproject) create instances of the class.

2.6.1.1 Properties

Property	Read / Write	Type	Description
name	Read / write	String	Name of the fabbproject's filename. Is empty when a new fabbproject is created and filled in load/save operations
root	read	String	Options that were not saved. Will be updated in save operations
traycount	read	number	Number of trays within the fabbproject

2.6.1.2 Method overview

Name	Syntax	Description
addtray	fabbproject:addtray(name: String, machinesize_x: Number, machinesize_y: Number, machinesize_z: Number)	Adds a new tray to the fabbproject. "Name" is the name of the tray, machinesize_x, _y, and _z are the sizes of the tray.
gettray	fabbproject:gettray(index: Integer)	Retrieve a tray from a fabbproject
savetofile	fabbproject:savetofile(filename: String)	Saves a fabbproject to a file
totalheight	fabbproject:totalheight()	The sum of heights of all trays (only the height of the space that is occupied by parts is included)

2.6.2 LUATray

[Desktop Automation]

A fabbproject can contain more than one tray. The trays can be obtained with the method "LUAFabbproject:gettray", the number of trays with the member "LUAFabbproject.traycount". Each tray has a property "root" which is the main LUAMeshGroup of the tray

2.6.2.1 Properties

Property	Read / Write	Type	Description
errormessage	Read	String	Error message for tray actions, at the moment only used by "Packing3d"
filling_degree	Read	number	Calculates the percentage of the tray that is filled with the parts currently inside the tray. **Note**: parts that are outside or partly outside the tray will not be included in the calculation **Note**: the filling degree is only calculated to the maximum Z-Value of the parts in the tray. The vertical size of the machine is not taken into consideration.
filling_height	Read	number	Calculates the maximum height of the parts loaded in the tray
root	read	LUAMeshGroup	Get the root of the tray
machinesize_x	Read / write	number	X-Size of the machine
machinesize_y	Read / write	number	Y-Size of the machine
machinesize_z	Read / write	number	Z-Size of the machine
name	Read / write	String	Name of the tray
packingid_2d	Read	number	Constant id for the 2d packer. Used for "LUATray: createpacker "
packingid_3d	Read	number	Constant id for the 3d packer. Used for "LUATray: createpacker "
packingid_montecarlo	Read	number	Constant id for the monte carlo packer. Used for "LUATray: createpacker "
packingid_outbox	Read	number	Constant id for the outbox packer. Used for "LUATray:pack"

2.6.2.2 Method overview

Name	Syntax	Description
checkforcollisions	Result = checkforcollisions(ARasterSize: Number)	Checks the tray for collisions. Returns an instance of "**LUACollisionResult**" (Reference: LUACollisionResult)
createpacker	Packer = tray:createpacker(packer_id: Number);	Create an object of type "LUAPacker" using the implementation defined in "packer_id". See constants "packingid_null", "packingid_outbox", "packingid_2d", "packingid_3d", "packingid_montecarlo" for the correct id. Please note that the createpacker command creates a snapeshoot of the tray with its current parts. If parts are removed or added afterwards you need to create a new packer

exportforsimulation	exportforsimulation(file: string; material: string; laser_spot_mm: number; unify_support: Boolean; workspace_uuid: string; workspace_name: string; laser_Count: number)	Generate a 3mf file that contains all parts of the tray with attached support and hulled support for process simulation.
getuuid	Uuid = tray:getuuid()	Retrieve the UUID of the tray

2.6.2.3 Constant Members

The Netfabb LUA tray object also contains some constant members

Name	Type	Description
packingid_2d	Number	Constant for the 2d packer, see “tray:createpacker”
packingid_3d	Number	Constant for the 3d packer, see “tray:createpacker”
packingid_montecarlo	Number	Constant for the monte carlo packer, see “tray:createpacker”
packingid_null	Number	Constant for the null packer, see “tray:createpacker”
packingid_outbox	Number	Constant for the outbox packer, see “tray:createpacker”

2.6.3 LUAMeshGroup

[Desktop Automation]

LUAMeshGroups are used to organize the meshes within a tray. Each LUAMeshGroup can contain meshes and subgroups. The root group of a tray is stored in "LUATray:root"

2.6.3.1 Properties

Property	Read / Write	Type	Description
meshcount	read	Number	Get the number of meshes within the group
name	Read / write	String	Name of the meshgroup
parent	read	LUAMeshGroup	The parent meshgroup
outbox	Read	LuaOutbox	The outbox of the mesh group

2.6.3.2 Method overview

Name	Syntax	Description
addmesh	Newtraymesh = Tray:addmesh(Mesh: LUAMesh; AName: String; AColor: Integer)	Add a LUAMesh to the group with the given name and color. Returns the new traymesh
addsubgroup	Subgroup = tray:addsubgroup(Name: String)	Add a new subgroup to the group

deletesubgroup	Tray:deletesubgroup(Subgroup: LUASubgroup)	Remove and delete a subgroup. Removes as well subgroups of this group and meshes in these groups.
getmesh	Mesh = tray:getmesh(Index: Integer)	Get a TLUATrayMesh from the group
getsubgroup	Subgroup = tray:getsubgroup(Index: Integer)	Get a subgroup by index
removemesh	Tray:removemesh(Mesh: TLUATrayMesh)	Removes a mesh from the group

2.6.4 LUATrayMesh

[Desktop Automation]

The LUATrayMesh object contains the data of the actual meshes of a fabbproject, i.e. an instance of LUAMesh (Reference: Mesh Objects) and its position, scale and rotation

2.6.4.1 Properties

Property	Read / Write	Type	Description
area	read	Number	Get the area of the tray mesh (including transformation, in mm^2)
color	Read/Write	Number	Set or get the color, (only the color used to display the mesh, not the mesh color information)
matrix	read	LUAMatrix4f	Get the transformation matrix of the mesh
mesh	read	LUAMesh	Get the mesh of the traymesh. Please note that this is only a copy of the LUAMesh in the Lua Automation module.
name	Read / write	String	Name of the mesh
outboxbasearea	read	Number	Get the outbox base area of the tray mesh (including tansformation, in mm^2)
outboxheight	read	Number	Get the outbox height of the tray mesh (including transformation, in mm)
outboxvolume	read	Number	Get the outbox volume of the tray mesh (including transformation, in mm^3)
parent	Read / write	LUAMeshGroup	Get or Set the group the mesh belongs to
uuid	read	String	UUID of the mesh
selected	read / write	Boolean	If the mesh is selected in the tray
volume	read	Number	Get the volume of the tray mesh (including transformation, in mm^3)

2.6.4.2 Method overview

Name	Syntax	Description

assignsupport	Mesh:assignsupport(mesh : LuaMesh; ReleativeCoordinates: Boolean)
	traymesh:assignsupport()
		Assigns a mesh as a support to a tray. Releavtive Coordinates indicates if the coordinates are given in respect to the main mesh. If 'assignsupport' is called without parameters, the existing support is detached.

createsupportedmesh	mesh:createsupportedmesh(mergepart, mergeopensupport, mergeclosedsupport: Boolean; openthickening);	Returns the support of the mesh. Boolean values specifies what meshes should be in the result. The thickening parameter allows the user to extrude open support so it gets closed.

getpackingoption/ setpackingoptions	Option = Mesh:getpackingoption(AOptionIdentifier: String)
	Mesh:setpackingoption(AOptionIdentifier: String; AOptionValue: String)
		Get/set a packing option of the mesh
		
		Note: the following read/write options are available:
		- 'priority' (number, 1-10)
		- 'restriction' ('locked', 'norestriction')
		- 'rotate' ('arbitrary', 'zaxis' , 'forbidden')
		These options have to be set before the start of the packing. All packer respect 'priority' and 'restriction'. 'Rotate' is only respected by the montecarlo packer.
		
		The following read-only option is available:
		- 'state' ('packed', 'leftover', 'not_packable', 'colliding', 'excluded', 'ignored', 'indefinite')
		It describes the state of the part after the packing process:
		'packed' – the part is packed normally;
		'leftover' – the part could not be packed because the number of the parts was too large for the given tray;
		'not_packable' – the part is too large for the given tray;
		'colliding' – the part in its initial position was colliding with a tray wall or with another part, while it was 'locked' or the packer option 'start_from_current_positions' was chosen.
		'excluded' – the part was excluded from the packing process by the user;
		'ignored' – the part was ignored by the packer (for instance, because its mesh was empty or invisible)
		'Indefinite' – the packing process has not been started yet or was aborted.

getuuid	Uuid = mesh:getuuid()	Get the UUID of a mesh

rotate	Mesh: rotate(axis_x, axis_y, axis_z, angle: Number)	Rotate a mesh by a defined angle (angle) around a defined axis (axis_x, axis_y, axis_z); Rotate angle is in radians.
saveto3ds	Mesh:saveto3ds(AFileName: String)	Export the mesh including all transformations to 3ds
saveto3mf	Mesh:saveto3mf(AFileName: String)	Export the mesh including all transformations to 3mf
savetoamf	Mesh:savetoamf(AFileName: String)	Export the mesh including all transformations to amf
savetoasciistl	Mesh:savetoasciistl(AFileName: String)	Export the mesh including all transformations to ascii stl
savetogts	Mesh:savetogts(AFileName: String)	Export the mesh including all transformations to gts
savetoncm	Mesh:savetoncm(AFileName: String)	Export the mesh including all transformations to ncm
savetoobj	Mesh:savetoobj(AFileName: String)	Export the mesh including all transformations to obj
savetoply	Mesh:savetoply(AFileName: String)	Export the mesh including all transformations to ply
savetostl	Mesh:savetostl(AFileName: String)	Export the mesh including all transformations to stl
savetovrml	Mesh:savetovrml(AFileName: String)	Export the mesh including all transformations to VRML
savetox3d	Mesh:savetox3d(AFileName: String)	Export the mesh including all transformations to x3d
savetozpr	Mesh:savetozpr(AFileName: String)	Export the mesh including all transformations to ZPR
scale	Mesh:scale(x, y, z: Number)	Scale a mesh
setmatrix	Mesh:setmatrix(matrix: LUAMatrix4)	Set the matrix of the mesh
setpackingoption	Mesh:setpackingoption(AOption: String; Avalue: String)	Add a packing option to the mesh
shellasmesh	Newmesh = mesh:shellasmesh(shellnumber: Number)	Extract a shell as new mesh object. Same as “LUAMeshObject.shellasmesh” but takes the matrix in the fabbproject into account. See “LUATrayMesh.mesh.shellcount” for the number of meshes
translate	Mesh:translate(x, y, z: Number)	Translate a mesh

2.6.5 LUACollisionResult

[Desktop Automation]

An instance of this object is returned by the “checkforcollisions” function of the LUATray object (Reference: LUATray). It contains information about the collisions within a tray.

2.6.5.1 Properties

Property	Read / Write	Type	Description

meshcount	Read	Number	Number of colliding meshes in the list **Note**: both colliding meshes are inserted into the list, so a meshcount of "2" means there is 1 collision

2.6.5.2 Method overview

Name	Syntax	Description
getcollisionmesh	Mesh = collisionresult:getcollisionmesh(0)	Returns a colliding mesh. **Note**: if a collision is detected the two collising meshes are inserted into the list one by one, i.e. Mesh at index 0 collides with mesh at index 1 and so on **Note**: if a single mesh is more often than once in the list means that the mesh collides with more than one other mesh

2.6.6 LUAPacker

[Desktop Automation]

An instance of this object is returned by the “createpacker” function of the LUATray (Reference: LUATray). It can be used to pack the content of the tray. Please note, that the packer gets a snapshot of the current tray at creation time. If you add meshes afterwards to the packer, they are not packed.

2.6.6.1 Properties

Property	Read / Write	Type	Description
borderspacingxy	Read/Write	Number	The desired minimal distance from the border of the tray. The default value is 0 mm;
borderspacingz	Read/Write	Number	The desired minimal distance from the build platform of the tray. The default value is 0 mm;
defaultpartrotation	Read/Write	Integer	Sets the global rotation behauviour. The values can be 0 (Arbitrary), 1 (z-Axis), 2 (no rotation), Default value is 0. Note that this property is overwritten by the packing property of the single tray meshes.
minimaldistance	Read/Write	Number	The desired minimal distance between the object in the tray. The default value is 1 mm.

optInteger	Read	Number	Constant for parameter type “Integer”
optFloat	Read	Number	Constant for parameter type “Float”
optBoolean	Read	Number	Constant for parameter type “Boolean”
parametercount	Read	Number	The number of parameters of the packer. Use the methods “tray:getparameter…” to get details about any parameters.
[Additional Parameter Name]	Read/Write	Number	Read or write the packer-specific parameters. Please refer to the documentation of the packer implementations for details.

2.6.6.2 Method overview

Name	Syntax	Description
getoutbox	Result = Packer:getoutbox()	Returns the dimensions of the current tray as a TLUAOutbox. The minx,miny, maxy, maxy coordinates of gives the start and end point of the tray.
getparametername	Result = packer:getparemetername(Index: Number)	Returns the name of the parameter with the given index. See tray.parametercount for the number of parameters available.
getparametertype	Result = packer: getparametertype(Index: Number)	Returns the type of the parameter with the given index as string (“integer”,”float”, “boolean”). See tray.parametercount for the number of parameters available.
getparametertypeid	Result = packer: getparametertypeid(Index: Number)	Returns the name of the parameter with the given index as constant (see LUAPacker.optInteger, LUAPacker.optFloat, LUAPacker.optBoolean). See tray.parametercount for the number of parameters available.

getparametervalue	Result = packer: getparametervalue(Index: Number)	Returns the value of the parameter with the given index. Note that the type of the returned value differs depending on the type of the paramter. See tray.parametercount for the number of parameters available.
pack	Result = packer:pack()	Pack the tray with the desired algorithm. Returns a packer-specific error code (general: 0 = no error)
setoutbox	Packer:setoutbox(Outbox : TLUAOutbox);	Sets the dimension of the tray, which should be packed. This can be used to pack only a fraction of the tray.

2.6.7 LUAPacker – Monte Carlo packer

[Desktop Automation]

An instance of this object is returned by the function of the LUATray: object packer = tray:createpacker(tray.packingid_montecarlo); (Reference: LUATray)

2.6.7.1 Properties

This section describes packer-specific properties of the Monte Carlo packer.

Property	Read / Write	Type	Description
packing_quality	Read/Write	Number	Is an integer number ranging from -8 (corresponding to the lowest packing density) to +8 (corresponding to the highest packing density). The higher the packing quality the more time is needed for the packing process. The default value is packing_quality=0
z_limit	Read/Write	Number	z_limit is the height in mm of the platform space that is allowed to contain the parts (the distance from the platform floor to the upper point of the packable region). It is a nonnegative real number which should not exceed the platform height: z_limit<=MachineSizeZ. If the parameter value does not satisfy these conditions, the default value is used instead. The default value is the platform height MachineSizeZ
start_from_current_positions	Read/Write	Boolean	Determines if the packing shoud be started from the current positions of the items (true) or an initial placement procedure has to be run before the packing begins (false). The default value is false.

2.6.7.2 Method overview

Name	Syntax	Description
pack	Result = packer:pack()	Pack the tray with the desired algorithm. Returns an error code that has the following meaning: 0 the packing is done. No problem has been detected. 1 the packing is done. There was not enough space in the tray to fit all items. 2 the packing is done. Some items are too large and cannot be fitted in the tray. 3 all the items are too large. None can be fitted into the tray. 4 there were no items to pack 5 starting the packing from the current item positions was not possible. Possible solutions: a) increase the packing quality b) try smaller values of the parameters min_dist and/or border_spacing_xy; c) do not start from current positions (set start_from_current_positions to false or drop this parameter) 13 unspecified error. Please contact the support team.

2.6.8 LUAPacker – 3d Scanline packer

[Desktop Automation]

An instance of this object is returned by the function of the LUATray: object packer = tray:createpacker(tray.packingid_3d); (Reference: LUATray)

2.6.8.1 Properties

The 3d scanline packer is also using borderspacingxy, borderspacingz, but not defaultpartrotation and minimaldistance of the general properties. This section describes packer-specific properties of the Scanline packer.

Property	Read / Write	Type	Description
allowrotationaxis	Read/Write	Number	Boolean, if true the parts will be rotated around the z-axis. The default value is false.

anglecount	Read/Write	Number	Integer, which gives the number of rotation, e.g. 4 means that 0, 90, 180 and 270 Degrees are tried. Values between 0,7 are allowed. The default value is 0.
coarsening	Read/Write	Number	Integer, increases the rastersize. Values between 1 and 9 are allowed. The default value is 1.
interlockingprotection	Read/Write	Number	Boolean, enables the interlocking protection. The default value is false.
minimizeoutbox	Read/Write	Number	Boolean, if true, the outboxes of the meshes will be minimized. The default value is false.
placeoutside	Read/Write	Number	Boolean, if true parts are moved outside the tray, if there could not be moved. The default value is false.
rastersize	Read/Write	Number	Integer, gives the rastersize of the raster. Values between 1 mm and 5 mm are allowed. The default value is 1 mm.

2.6.8.2 Method overview

Name	Syntax	Description
pack	Result = packer:pack()	Pack the tray with the desired algorithm. Returns an error code that has the following meaning: 0 the packing is done. No problem has been detected. All parts could be packed. 1 Error occurred, please check the indivual mesh pack results

2.6.9 LUAPacker – 2d packer

[Desktop Automation]

An instance of this object is returned by the function of the LUATray: object packer = tray:createpacker(tray.packingid_2d); (Reference: LUATray). Please note that this packer is only available in the Lua Automation Module of the Desktop version

2.6.9.1 Properties

The 2d packer is also using borderspacingxy, but not defaultpartrotation, borderspacingz, and minimaldistance of the general properties. This section describes packer-specific properties of the 2d packer.

Property	Read / Write	Type	Description
anglecount	Read/Write	Integer	Integer, which gives the number of rotation, e.g. 4 means that 0, 90, 180 and 270 Degrees are tried. Values between 0,7 are allowed. Named “Z-Rotation” in the Desktop GUI for the 2dpacker. The default value is 0.
coarsening	Read/Write	Integer	Integer, increases the rastersize. Values between 1 and 9 are allowed. The default value is 0.
packeonlyselected	Read/Write	Boolean	Boolean, if true only parts are packed, which have been selected in the Desktop application. The default value is false.
placeoutside	Read/Write	Boolean	Boolean, if true parts are moved outside the tray, if there could not be moved. The default value is false.
rastersize	Read/Write	Number	Ranges between 0.1 and 10 mm. Gives the voxelsize which is internally used for the algorithm. Named “voxelsize” in the Desktop GUI for the 2dpacker. The default value is 1.
sorttype	Read/Write	Integer	The 2d-Packer packs every part individually in a sequentiell order. The packing starts in one corner and is stopped if either all parts are packed or no more space is available. The parts are sorted before the packing. The sorting can either by size (sorttype =0) or by height (sorttype =1). The size refers to the effective 2d raster size of the corresponding object. The defaullt value is 0.

2.6.9.2 Method overview

Name	Syntax	Description
pack	Result = packer:pack()	Pack the tray with the desired algorithm

2.6.10 LUAPacker – outbox packer

[Desktop Automation]

An instance of this object is returned by the function of the LUATray: object packer = tray:createpacker(tray. packingid_outbox); (Reference: LUATray).

2.6.10.1 Properties

This section describes packer-specific properties of the outbox packer.

Property	Read / Write	Type	Description
pack2D	Read/Write	Boolean	If true, only planar packing is used, i.e. the parts are not packed in z. The default value is false.
rastersize	Read/Write	Number	Float number, ranges between 0.1 and 10 mm. Gives the voxelsize which is internally for the algorithm. Named “voxelsize” in the Desktop GUI for the 2dpacker. The default value is 1 mm.

2.6.10.2 Method overview

Name	Syntax	Description
pack	Result = packer:pack()	Pack the tray with the desired algorithm

2.7 GUI-specific Objects

[Desktop Automation]

2.7.1 ControlContainer Base class

[Desktop Automation]

Descendents of this class are:

- Groupbox
- Splitter
- TabSheet
- Dialog

The control container is the base class for all GUI elements that can contain other GUI elements and provides the following methods to add the corresponding elements:

2.7.1.1 Properties

None.

2.7.1.2 Method Overview

Name	Syntax	Description
addbutton	controlcontainer:();	Add a Button GUI element
addcheckbox	controlcontainer:addcheckbox();	Add a Checkbox GUI element
adddropdown	controlcontainer:adddropdown();	Add a Dropdown GUI element
addedit	controlcontainer:addedit();	Add a Edit GUI element
addfloatspinedit	controlcontainer:addfloatspinedit()	Add a FloatSpinEdit GUI element
addgroupbox	controlcontainer:addgroupbox();	Add a Groupbox GUI element
addimage	controlcontainer:addimage();	Add an Image GUI element
addlabel	controlcontainer:addlabel();	Add a Label GUI element
addmemo	controlcontainer:addmemo();	Add a Memo GUI element
addpicturebutton	controlcontainer:addpicturebutton();	Add a PicturebuttonGUI element
addscrollbar	controlcontainer:addscrollbar();	Add a Scrollbar GUI element
addspacer	controlcontainer:addspacer();	Add a Spacer GUI element
addslider	controlcontainer:addslider();	Add a Slider GUI element
addsplitter	controlcontainer:addsplitter([Name: String, ThreeSections: Boolean]);	Add a Splitter GUI element. Optionam parameter “Name” specifies the name of the element, “ThreeSections” decides whether or not the splitter shall include 3 splits. Use an empty string as name if you want to use three sections but don’t want to specify a name.
addtabcontrol	controlcontainer:addtabcontrol();	Add a TabControl GUI element
addtable	controlcontainer:addtable();	Add a TableObject GUI element
disabletimer	Controlcontainer:disabletimer()	Disables a timer.
enabletimer	Controlcontainer:enabletimer(interval:Number, call:String)	Set up a callback function for a timer that is called all interval milliseconds.

2.7.2 Button

[Desktop Automation]

2.7.2.1 Properties

Property	Read / Write	Type	Description
bold	read / write	boolean	shall the button text be bold?
caption	read / write	string	Caption of the button
enabled	read / write	boolean	shall the button be enabled?
height	read / write	number	height in pixels
hint	read / write	string	Hint / Tooltip for the button
onclick	read / write	string	callback function for a click event
translate	read / write	boolean	if true, 'caption' has to be String Identifier
visible	read / write	boolean	shall the button be visible?

2.7.2.2 Method Overview

None.

2.7.3 Picturebutton

[Desktop Automation]

2.7.3.1 Properties

Property	Read / Write	Type	Description
bold	read / write	boolean	shall the caption be bold?
caption	read / write	string	caption of the picturebutton
enabled	read / write	boolean	shall the button be enabled?
height	read / write	number	height in pixels
hint	read / write	string	Hint / Tooltip for the button
onclick	read / write	string	callback function for a click event
picture	read / write	string	small picture on the button
translate	read / write	boolean	if true, 'caption' must be String Identifier
visible	read / write	boolean	shall the button be visible?

2.7.3.2 Method Overview

None.

2.7.4 Checkbox

[Desktop Automation]

2.7.4.1 Properties

Property	Read / Write	Type	Description
caption	read / write	string	caption of the checkbox
checked	read / write	boolean	if true, Checkbox is checked
enabled	read / write	boolean	if true, Checkbox is enabled
hint	read / write	string	Hint / Tooltip for the button
leftspacing	read / write	number	spacing on the left in pixels
onclick	read / write	string	callback function for a click event
topspacing	read / write	number	spacing on the top in pixels
translate	read / write	boolean	if true, 'caption' has to be String Identifier
visible	read / write	string	Shall the ckeckbox be visible?

2.7.4.2 Method Overview

None.

2.7.5 Groupbox

[Desktop Automation]

Descendent of ControlContainer Base class providing all functions of this base class

2.7.5.1 Properties

Property	Read / Write	Type	Description
borderstyle	read / write	number	style of the border (0, 1, 2 or 3)
caption	read / write	string	caption of the groupbox
enabled	read / write	boolean	Shall the groupbox be enabled?
hint	read / write	string	Hint / Tooltip for the button
horizontalpadding	read / write	number	left and right padding in pixels
minheight	read / write	number	minimum height of the groupbox in pixels
verticalpadding	read / write	number	top and bottom padding in pixels
visible	read / write	boolean	Shall the groubox be visible?

2.7.5.2 Method Overview

None.

2.7.6 Dropdown

[Desktop Automation]

Tips:

- Always create them with id = priority as an ascending integer
- Don't use the write property of selectedindex to change the control's selected item – it doesn't work. Use selecteditem (which references the "id" field of additem()) instead.

2.7.6.1 Properties

Property	Read / Write	Type	Description
backgroundcolor	Read/write	String	Gives the Background color as a string. Format is '$BBGGRR'
caption	read / write	string	caption of the dropdown
captionwidth	read / write	number	width of the caption in pixels
captionwidthpercentage	Read / write	Number	Width of the catpion in percentage
customdraw	Read / write	boolean	Enable custom drawing of dropdown items (parameters "style", "fontcolor" of method "additem" has no effect if this member is remains "false" (default)), also "backgroundcolor" as no effect
count	read	number	Number of items in the dropdown
enabled	read / write	boolean	Shall the dropdown be enabled?
hint	read / write	string	Hint / Tooltip for the button
onchange	read / write	string	callback function for a change event
selecteditem	read / write	number	The id of the selected item (see method "additem")
selectedindex	read / write	number	The index of the selected item (0-based)
spacing	read / write	number	spacing of the dropdown
translatecaption	read / write	boolean	if true, 'caption' has to be String Identifier
visible	read / write	boolean	Shall the dropdown be visible?

2.7.6.2 Method Overview

Name	Syntax	Description
additem	dropdown:additem (caption, id, [priority, translate, style, fontcolor]);	Adds an item with a 'caption' and 'id', sorted by 'priority' number ascending. Translate is a boolean. Style is an integer bitfield for the item's draw style (1=bold, 2=italic, 4=strikeout, 8=underline), fontcolor is the font color. Format is '$BBGGRR'
clear	dropdown:clear();	Clears the items in dropdown list.
getitemparameter	Parameter = dropdown: getitemparameter(Index: Integer);	Get the Parameter (id) of an item at a specific position
getitemtext	Text = dropdown:getitemtext(Index: Integer);	Get the text of an item at a specific position

removeitem	dropdown:removeitem(id);	removes an item
updateitems	dropdown:updateitems();	Updates all the items added or removed in the list.

2.7.7 Edit

[Desktop Automation]

2.7.7.1 Properties

Property	Read / Write	Type	Description
caption	read / write	string	caption of the edit
captionwidth	read / write	number	width of the caption in pixels
captionwidthpercentage	Read / write	Number	Width of the catpion in percentage
customcolor	read / write	string	Custom background color for the edit. Format is '$BBGGRR'
enabled	read / write	boolean	Shall the edit be enabled?
hint	read / write	string	Hint / Tooltip for the button
key	read / write	string	should be read out in a key up/down callback
numbersonly	read / write	boolean	Flag to specify whether or not only numeric characters are valid for the edit field
onchange	read / write	string	callback function for a change event
onkeydown	read / write	string	callback function for a key-down event
onkeyup	read / write	string	callback function for a key-up event
readonly	read / write	boolean	Shall the edit be readonly?
spacing	read / write	number	spacing of the edit in pixels
text	read / write	string	text content of the edit
topspacing	read / write	number	top spacing og the edit in pixels
translate	read / write	boolean	if true, 'caption' has to be String Identifier
visible	read / write	boolean	Shall the edit be visible?

2.7.7.2 Method Overview

None.

2.7.8 Label

[Desktop Automation]

2.7.8.1 Properties

Property	Read / Write	Type	Description
caption	read / write	string	caption of the label
fontcolor	Read/write	string	fontcolor gives color of the text. Format is '$BBGGRR'
hint	read / write	string	Hint / Tooltip for the button

leftspacing	read / write	number	left spacing of the label in pixels
topspacing	read / write	number	top spacing of the label in pixels
translate	read / write	boolean	if true, 'caption' has to be String Identifier
visible	read / write	boolean	Shall the label be visible?

2.7.8.2 Method Overview

None.

2.7.9 Scrollbar

[Desktop Automation]

2.7.9.1 Properties

Property	Read / Write	Type	Description
enabled	read / write	boolean	Shall the scrollbar be enabled?
height	read / write	number	height of the scrollbar in pixels
hint	read / write	string	Hint / Tooltip for the button
max	read / write	number	maximum value of the scrollbar
onchange	read / write	string	callback function for a change event
orientation	read / write	number	0: vertical 1: horizontal
position	read / write	number	position of the bar that can be tracked
spacing	read / write	number	spacing of the scrollbar in pixels
visible	read / write	boolean	Shall the scrollbar be visible?

2.7.9.2 Method Overview

None.

2.7.10 Slider

[Desktop Automation]

Notes: you cannot set the units. Units are always integer between “min” and “max”. You need to implement some more calculation if you need other units.

Example:

```
function SetupUI()
   GEditSlider = interface_main_frame:addedit()
   GEditSlider.caption = "Position"
   GEditSlider.translate = false

   GSlider = interface_main_frame:addslider()
   GSlider.caption = "Slider"
   GSlider.captionwidth = 120
   GSlider.min = 46
   GSlider.max = 93
   GSlider.position = 69
   GSlider.onchange = "SliderCallback"
   SliderCallback()
end

function SliderCallback()
  GEditSlider.text = "Position: " .. GSlider.position
End
```

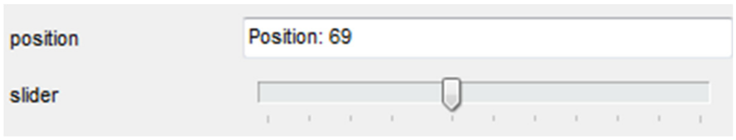


Figure 1 - the result of the slider example code

2.7.10.1 Properties

Property	Read / Write	Type	Description
caption	read / write	string	caption of the slider
captionspacing	read / write	number	spacing of the caption in pixels
captionwidth	read / write	number	width of the caption in pixels
captionwidthpercentage	Read / write	Number	Width of the catpion in percentage
enabled	read / write	boolean	Shall the slider be enabled?
height	read / write	number	height of the slider in pixels
hint	read / write	string	Hint / Tooltip for the button
max	read / write	number	maximum value of the slider
onchange	read / write	string	callback function for a change event
position	read / write	number	position of the slider
spacing	read / write	number	spacing of the slider in pixels
topspacing	read / write	number	top spacing of the slider in pixels
translate	read / write	boolean	if true, 'caption' is String Identifier
visible	read / write	boolean	Shall the slider be visible?

2.7.10.2 Method Overview

None.

2.7.11 Memo

[Desktop Automation]

2.7.11.1 Properties

Property	Read / Write	Type	Description
caption	read / write	string	caption of the memo
captionwidth	read / write	number	width of the caption in pixels
captionwidthpercentage	Read / write	Number	Width of the catpion in percentage

height	read / write	number	height of the memo in pixels
hint	read / write	string	Hint / Tooltip for the button
onchange	read / write	string	callback function for a change event
readonly	read / write	boolean	Shall the memo be readonly?
text	read / write	string	text content of the memo
translate	read / write	boolean	if true, 'caption' is String Identifier
visible	read / write	boolean	Shall the memo be visible?

2.7.11.2 Method Overview

Name	Syntax	Description
addline	memo:addline(text:String)	Adds a text line to the memo field

2.7.12 Spacer

[Desktop Automation]

2.7.12.1 Properties

Property	Read / Write	Type	Description
height	read / write	number	height of the spacer in pixels
visible	read / write	boolean	Shall the spacer be visible?

2.7.12.2 Method Overview

None.

2.7.13 Splitter

[Desktop Automation]

Descendent of ControlContainer Base class providing all functions of this base class

2.7.13.1 Properties

Property	Read / Write	Type	Description
spacing	read / write	number	spacing of the splitter in pixels
splittype	read / write	number	0: default 1: left split is smaller 2: right split is smaller
visible	read / write	boolean	Shall the splitter be visible?
width	read / write	number	width of the splitter in pixels

2.7.13.2 Method Overview

Name	Syntax	Description

settocenter	splitter:settocenter	Sets the active split side to the center one (only available if the “ThreeSections” parameter was set to “true” when creating the splitter (See ControlContainer Base class for details)
settoleft	splitter:settoleft();	sets the active split side to the left one
settoright	splitter:settoright();	sets the active split side to the right one

2.7.14 Dialog

[Desktop Automation]

Descendent of ControlContainer Base class providing all functions of this base class

2.7.14.1 Properties

Property	Read / Write	Type	Description
caption	read / write	string	caption of the dialog
translatecaption	read / write	boolean	if true, caption is String Identifier
width	read / write	number	width of the dialog in pixels

2.7.14.2 Method Overview

Name	Syntax	Description
close	dialog:close();	closes the dialog
rebuild	dialog:rebuild();	rebuilds the dialog
show	dialog:show();	shows the dialog

2.7.15 TableObject

[Desktop Automation]

The table object is a GUI object that allows you to represent data in a column-row grid. It is created using the ":addtable" method of a GUI container (e.g. Dialog).

2.7.15.1 Properties

Property	Read / Write	Type	Description
columncount	Read / Write	number	Get the number of columns of the table
fixedcolumncount	Read / Write	number	Get the number of fixed (headline) columns of the table
fixedrowcount	Read / Write	number	Get the number of fixed (headline) rows of the table
height	Read / Write	number	Get the height of the table
ondoubleclick	Read / Write	String	Lua callback method for double click event

rowcount	Read / Write	number	Get the number of rows of the table
translateheadline	Read / Write	boolean	See whether or not the headlines are translated automatically

2.7.15.2 Method Overview

Name	Syntax	Description
getcelltext	Table:getcelltext(column: integer; row: integer);	Retrieve the text of a table cell
setcelltext	Table:setcelltext(column: integer; row: integer; text: string);	Modify the text of a table cell
setcolumnwidth	Table:setcolumnwidth(column: integer; width: integer);	Modify the width of a column of the table

2.7.16 TabControl

[Desktop Automation]

With the tab control you can arrange dialogs in multiple tabs. Use “addtabcontrol” to add such an element

2.7.16.1 Properties

Property	Read / Write	Type	Description
activepage	Read / Write	number	Get and set the currently active page
pagecount	Read	number	Get the number of pages of the control

2.7.16.2 Method Overview

Name	Syntax	Description
addtabsheet	Tabcontrol:addtabsheet();	Add a new sheet to the tab control. If you add a new sheet at runtime make sure you call the “*rebuild*” function of the containing dialog.

2.7.17 TabSheet

[Desktop Automation]

A TabSheet represents a tab of the TabControl object. It can be created using TabControl:addtabsheet(). Descendent of ControlContainer Base class providing all functions of this base class

2.7.17.1 Properties

Property	Read / Write	Type	Description
caption	Read / Write	string	The caption of the tabsheet
translate	Read / Write	Boolean	Should the caption be translated?

2.7.17.2 Method Overview

It has no methods of its own but supports all methods of the other ControlCon-tainer, e.g. Addbutton.

2.7.18 Image

[Desktop Automation]

Image UI element. It is created by “controlcontainer:addimage()”.

2.7.18.1 Properties

Property	Read / Write	Type	Description
height	Read / Write	Number	Height of the image UI element
hint	read / write	string	Hint / Tooltip for the button
keepratio	Read / Write	Boolean	Keep the aspect ratio. This will only have an effect if only either width or height are set and if the “stretch” parameter is set to true
picture	Read / Write	String / Object	The image to be shown in the UI element. Can either be a string referencing an image file or an Image Object
onclick	Read / Write	String	Onclick listener to handle Onclick event of the image
mousex	Read	Number	X-Position of the mouse, usefull after onclick event
mousey	Read	Number	Y-Position of the mouse, usefull after onclick event
stretch	Read / Write	Boolean	Should the image be stretched to the size defined by “width” and “height”?
width	Read / Write	Number	Width of the image UI element

2.7.18.2 Method Overview

Name	Syntax	Description
setgraph	image:setgraph(graphobject);	Sets a Graphobject as the image. (Set image height and width first!)
setimage	Image:setimage(imageobject);	Sets an image-object as content. See image-processing section

2.7.19 FloatSpinEdit

[Desktop Automation]

A Spin Edit UI element that allows modification of Float numbers using up and downd buttons. It is created by “controlcontainer:addfloatspinedit()

2.7.19.1 Properties

Property	Read / Write	Type	Description
caption	Read / Write	String	The caption that is placed in front of the edit
captionwidth	Read / Write	Number	Width of the edit's caption
captionwidthpercentage	Read / write	Number	Width of the catpion in percentage
customcolor	Read / Write	String	Custom background color for the edit. Format is '$BBGGRR'
decimalplaces	Read / Write	Number	Number of decimal places shown
enabled	Read / Write	Boolean	Flag to enable and disable the edit
height	Read / Write	Number	Height of the edit field
hint	Read / Write	String	Hint to be shown
max	Read / Write	Number	Maximum value
min	Read / Write	Number	Minimum value
readonly	Read / Write	Boolean	Flag to specify whether or not the edit field is read only
spacing	Read / Write	Number	Spacing of the spin edit in pixels
step	Read / Write	Number	Size of the steps applied using the up and down buttons
topspacing	Read / Write	Number	Top spacing of the spin edit in pixels
translate	Read / Write	Boolean	Flag to specify whether or not the caption and the hint should be translated
value	Read / Write	Number	Value of the spin edit.

2.7.19.2 Method Overview

None.

2.7.20 Histogram Object

[Desktop Automation]

The Histogram Object allows the user to plot any values into an histogram or graph. It is created by "system:createhistogram()".

2.7.20.1 Properties

Colors are encoded as Integers '$BBGGRR'

Property	Read / Write	Type	Description
bar_count	Read / Write	Number	Number of bars of the histogram
color_bar	Read / Write	Number	Color of the bar
color_bar_border	Read / write	Number	Color of the bar border
color_background	Read / Write	Number	Color of the background
color_caption	Read / Write	Number	Color of the caption

color_selection	Read / Write	Number	Color of selection
caption_x	Read / Write	String	Horizontal caption string
caption_y	Read / Write	String	Vertical caption string
selection	Read / Write	Number	Current selected bar or graph-section
unit	Read / Write	String	Unit string

2.7.20.2 Method Overview

Name	Syntax	Description
addvalue	histogram:addvalue(value: number)	Add new value
getvalue	Histogram:getvalue(index: number)	Return the value at given index.
drawhistogram	Img = histogram:drawhistogram(width, height: number)	Draw the histogram that contains grouped values. The graphic is drawn into an image with given dimensions. The result is an image object.
drawgraph	Img = histogram:drawgraph(width, height: number)	Draw the orderd values. The graphic is drawn into an image with given dimensions. The result is an image object.
setcustomminmax	histogram:setcustomminmax(min, max: number	Override min and max of added values and set custom min and max for drawing.
selectatpixel	histogram:selectatpixel(width, height, x, y: number)	Select current selected bar or graph-section. Select by given graphic (width and height) and by given pixel position (x and y)

2.7.21 Graph Object

[Desktop Automation]

Image UI element. It is created by "system:creategraph()".

2.7.21.1 Properties

None.

2.7.21.2 Method Overview

Name	Syntax	Description
addrow	graph:addrow(key:float);	Adds a row/key at the position (like the width (and index) of an Excel column)
addvalue	Graph:addvalue(column:integer, row:integer, value:integer)	Sets value in row (key) and column,

setcolor	Graph:setcolor(graph:integer, color:integer)	Sets color for graph. Graph is the index of the graph (the number of graphs is set during the creation. Color is a RGB value encoded in one integer, like: graph:setcolor (0, 256 * 128); -- Green graph:setcolor (1, 65536 * 128); -- Blue
savetopng	Graph:savetopng(name:string, width:integer, height:integer)	Saves the graph to the file:name in the height and width dimensions.

2.8 Slice related Objects

[Slice Commander] [Desktop Automation]

The Slice Commander module has the option to load a Lua script file via a menu option: Prepare->Run Lua Script. The API this Lua API interface provides is nearly identical with the one for [Desktop Automation]. The only addition is a default “slice” object in the namespace of the script, which provides convenient access to the currently loaded slice stack.

There are three main objects regarding slices:

- Slice Layer
- Slice Image Exporter
- Slice Object

Slice object is the main object.

2.8.1 Slice Layer

[Slice Commander] [Desktop Automation]

2.8.1.1 Properties

Property	Read / Write	Type	Description
contourcount	read	number	returns the number of contours of a slice layer
contourlength	read	number	returns the contourlength a slice layer
hatchcount	read	number	returns the number of hatches of a slice layer
hatchlength	read	number	returns the hatchlength a slice layer
maxx	read	number	maximum X of a slice layer
maxy	read	number	maximum Y of a slice layer
minx	read	number	minimum X of a slice layer
miny	read	number	minimum Y of a slice layer

2.8.1.2 Method Overview

Name	Syntax	Description
addpoint	slice:addpoint(?:number, ?:number);	
begincontour	slice:begincontour();	
endcontour	slice:endcontour();	
extractcontour	slice:extractcontour(?:number);	
move	slice:move(?:number, ?:number);	
getpointcount	slice:getpointcount();	returns the amount of points of a slice layer
getpointx	slice:getpointx(?:number, ?:number);	
getpointy	slice:getpointy(?:number, ?:number);	
sort	slice:sort(?:number, ?:number);	

2.8.2 Slice Image Exporter

[Slice Commander] [Desktop Automation]

2.8.2.1 Properties

Property	Read / Write	Type	Description
backgroundcolor	read / write	number	color of the background
basefilename	read / write	string	base file name of the images
closedcontourcolor	read / write	number	color of the closed contours
closedcontourwidth	read / write	number	width of the closed contours in mm
dpix	read / write	number	dpi in X
dpiy	read / write	number	dpi in Y
dofillingbooled	read / write	boolean	
exportmonochromeimages	read / write	boolean	Shall monochrome images be exported?
fillclosedcontours	read / write	boolean	Shall the closed contours be filled?
fillcolor	read / write	number	color of the filling
hatchcolor	read / write	number	color of the hatches
hatchwidth	read / write	number	width of the hatch in mm
height	read / write	number	height of image in pixels
left	read / write	number	left space of the image in pixels
maxx	read	number	maximum X of the image
maxy	read	number	maximum Y of the image
maxz	read	number	maximum Z of the image
minx	read	number	minimum X of the image
miny	read	number	minimum Y of the image
minz	read	number	minimum Z of the image
opencontourcolor	read / write	number	color of the open colours
opencontourwidth	read / write	number	width of the open contours in mm

separatesupports	Read / write	boolean	Create separate slice images for part and support
supportbasename	Read / write	string	Basename of the support images (only used if “separatesupports” is set to “true”)
writeopencontours	read / write	boolean	Shall the open contours be written?

2.8.2.2 Method Overview

Name	Syntax	Description
exportbmp	slice:exportbmp(identifier:string);	export slices as bmp
exportpng	slice:exportpng(identifier:string);	export slices as png
exportpngmulticore	slice:exportpngmulticore(identifier:string);	export slices as png with multicore support for faster calculation

2.8.3 Slice Object

[Slice Commander] [Desktop Automation]

2.8.3.1 Properties

Property	Read / Write	Type	Description
layercount	read	number	returns count of layers
layersize	read	number	returns thickness of the layers (if consistent throughout the slice)
maxx	read	number	maximum X of the slice in mm
maxy	read	number	maximum Y of the slice
maxz	read	number	maximum Z of the slice
minx	read	number	minimum X of the slice in mm
miny	read	number	minimum Y of the slice in mm
minz	read	number	minimum Z of the slice in mm
name	read / write	number	returns / sets name of the slice

2.8.3.2 Method Overview

Name	Syntax	Description
calculateslicearea	slice:calculateslicearea(sliceheight:float);	Calculates the area of a slice; slice is selected by its height; return the value
calculateslicearealayerindex	slice:calculateslicearea(sliceindex:int);	Calculates the area of a slice; slice is selected by its index; return the value
calculateslicecontour	slice:calculateslicecontour(sliceheight:float);	Calculate the length of a slice; slice is selected by its height; return the value

calculateslicecontourlayerindex	slice:calculateslicecontour(sliceindex:int);	Calculate the length of a slice; slice is selected by its index; return the value
conversion_connectopencontours	resultsliceobject = slice: conversion_connectopencontours();	
conversion_contourtohatches	resultsliceobject = slice: conversion_contourtohatches();	
conversion_filter	resultsliceobject = slice: conversion_filter(PreserveHatches : Boolean; PreserveOpenContours : Boolean; PreserveClosedContours : Boolean);	
conversion_filtercontoursbyarea	resultsliceobject = slice: conversion_filtercontoursbyarea (MinContourArea : number)	
conversion_hatchestocontour	resultsliceobject = slice: conversion_hatchestocontour(Accuracy : Number; DoMergeHatches : Boolean; KeepMergeOrder : Boolean)	
conversion_hatchpermutation	resultsliceobject = slice: conversion_hatchpermutation (PermuteEveryNthHatch : integer);	
conversion_randomizeseam	resultsliceobject = slice: conversion_randomizeseam();	
conversion_revertdirection	resultsliceobject = slice:conversion_revertdirection (RevertEveryNthLayer: integer; RevertLayerOffset : integer)	
createaggregation	slice:createaggregation(Offset[mm]:float, Accuracy:float);	Create an aggregation; returns the result as new slice object
createcontoursegmentation	resultsliceobject = slice: createcontoursegmentation(Length := Number; Count :.Integer; Overlap : NumberValue = 0; RandomizeSeam : Boolean = false; IncludeHatches : Boolean = false);	
createdownskin	slice:createdownskin();	Creates a downskin; returns the result as new slice object
createflowsegmentation	resultsliceobject = slice: createflowsegmentation(AAngle : Number;ADirectionTolerance : Number; AInvertNegatives: Boolean; AIncludeHatches : Boolean);	
createimagerenderer	slice:createimagerenderer(layerindex:int, size:float);	Create an image of the slice
createlayerunification	slice:createlayerunification(?:number);	Combine all slice files of the slice objects, starting with the heighest slice

createoffset	slice:createoffset(offset[mm]: float, is_inner_offset:boolean, roundness [degree]:float = 30);	Creates an offset (inner or outer given by *is_inner_offset*) with a given roundness; *;* returns the result as new slice object
createquadfilling	slice:createquadfilling(HatchDistance:float, Angle :float = 0 , AngleIncrement :float =0, OnlyEachLayer : int = 1, HatchOriginIncrement :float = 0, QuadSizeX:float =20 , QuadSizeY:float =20)	Create a quad filling
createrenderer	slice:createrenderer(layerfactor:int);	
createrotatedslice	slice:createrotatedslice (rotationfactor:int);	Creates a rotated slice, rotated, by an angle of 'rotationfactor'
createscaledslice	slice: createscaledslice (factorx:int [, factory:int]);	Creates a scaled slice, if called with only one parameter the scaling factorx is used for x and y.
createsimplehatching	slice:createsimplehatching(hatchdistance:float, angle:float =0, angleincrement:float = 0, onlyeachlayer:int = 1, hatchoriginincrement:float = 0);	Creates a hatching on the sliceobject; you can increment the angle at each layer by setting *angleincrement;* hatches are created only at ech layer by setting *onlyeachlayer;* with the parameter *hatchoriginincrement* the hatchs can be shifted at each layer; returns the result as new slice object
createstripefilling	slice:createstripfilling(HatchDistance:float, StripeWidth:float = 10, StripeGap:float, Angle:float = 0, AngleIncrement:float = 0, OnlyEachLayer:int = 1, SortType:int)	Create a strip filling
createtranslatedslice	slice: createtranslatedslice (factorx:int , factory:int);	Creates slice trlanlated by the factors factorx and factory.
createupskin	slice:createupskin();	Creates an upskinr; returns the result as new slice object
duplicate	slice:duplicate	Duplicate the slice object;
executecalculation	slice:executecalculation(debugmessage:string, layerthickness:number);	Really executes all calculations
getlayerz	slice:getlayerz(layerindex:int);	Returns the Z height of a layer

hatchextension	resultsliceobject = slice:hatchextension(FixedDistance1, AngleFactorA1, AngleFactorB1, AngleFactorC1, AnglePowerA1, AnglePowerB1, AnglePowerC1, FixedDistance2, AngleFactorA2, AngleFactorB2, AngleFactorC2, AnglePowerA2, AnglePowerB2 , AnglePowerC2: number);	
hatchcutting	resultsliceobject = slice:hatchcutting()	
loadlayer	slice:loadlayer(index:int);	Returns the slice layer at the given *index;* returns the result as new slice layer object
moveslice	slice:moveslice(X[mm]:float, Y[mm]:float, Z[mm]:float);	Moves a slice by the given values; returns nothing

multilasersplit	slicelist = slice:multilasersplit (lcount, acc,.mbsize, mdist, perlayershift, gflow, overlap, overlapa);	Sets up the slice for multi-laser processing. Returns slice list.
		Parameters:
		lcount- Laser Count e.g. 2
		acc- Accuracy, e.g. 0.25
		mbsize - Max Blocksize,e.g. 500
		mdist - Max Distance e.g. 20
		perlayershift- Shift per layer e.g. 2.5
		gflow - Gas flow Angle e.g. 45
		overlap - overlap in flow: e.g. 0.3
		overlapa - overlap against flow: e.g. 1.2
		Example:
		`slicelist = slice:multilasersplit (2, 0.25, 500, 20, 2.5, 45, 0.3, 1.2);`
		`-- Laser Count 2`
		`-- Accuracy 0.25`
		`-- Max Blocksize 500`
		`-- Max Distance 20`
		`-- Shift per layer 2.5`
		`-- Gas flow Angle 45`
		`-- overlap in flow: 0.3`
		`-- overlap against flow: 1.2`
		
		`laser1slice = slicelist:getslice (0);`
		`laser1slice.name = "Laser 1";`
		`laser2slice = slicelist:getslice (1);`
		`laser2slice.name = "Laser 2";`
		
		`system:addslicetotree (laser1slice);`
		`system:addslicetotree (laser2slice);`

pointreduction	slice: pointreduction (ATolerance:number);	Applies point reduction on slice

reduce	slice:reduce(tolerance[mm]: float);	Reduces the points of a slice*;* returns the result as new slice object
removeselfintersections	slice:removeselfintersections();	Removes the self intersections; returns the result as new slice object
removeselfintersectionmulticore	slice:removeselfintersections();	Removes the self intersections using multi cpu cores; returns the result as new slice object
savetofile	slice:savetofile(identifier:string, slycetype:int, layersize:float, minz:float, maxZ:float,);	Exports a slice to a given slice format; *identifier* gives the name; *slicetype* gives the slice type; returns nothing Slice type global constants: • stCLI • stCLS • stSLC • stSLI • stUSF
smooth	slice:smooth(value:number);	Smoothes a slicer; returns the result as new slice object
substractslice	slice:subtractslice(sliceobject:object);	Subtracts from slice the given slice *sliceobject.;* returns the result as new slice object
translate	slice:translate(X[mm]:float, Y[mm]:float, Z[mm]:float);	Moves a slice by the given values; returns nothing

2.9 Document related Object types

2.9.1 XML file Object

[Desktop Automation]

An XML file object contains the full information of an XML file. It can be used to read XML information from the disk as well as output information in the XML format to disk or to stdout.

2.9.1.1 Properties

Property	Read / Write	Type	Description
childcount	read only	Number	Returns the number of xml nodes of the root entry
root	read	Object	Query the root node as XMLNode object
rootname	read / write	String	Section name of the root entry
useattributesaschildren	Read / write	Boolean	Should attributes be treated as child nodes?

2.9.1.2 Method Overview

Name	Syntax	Description

addchild	xmlfile:Addchild(childname:string, adddouble:boolean);	Adds child node, adddouble allows for duplicates
addvalue	xmlfile:addvalue(nodename:string, value:string/int);	Adds a value to the XML file
childexists	xmlfile:childexists(nodename:string);	Returns if nodename in the XML file exists
dump	xmlfile:dump()	Dumps the XML file on stdout. Even if the cloud utilities are in quiet mode
findchild	Childnode = xmlfile:Findchild(childname:string);	Finds child node
getchildint	ChildIntvalue = xmlfile:Getchildint(child:string);	Gets the Integer value of child node
getchildindexed	Childvalue = xmlfile:Getchildindexed(childindex:string);	Returns child value by index
getchildintdef	ChildIntvalue = xmlfile:Getchildintdef(child:string;value:number);	Gets the Integer value of child or value
getchildfloat	ChildIntvalue = xmlfile:Getchildfloat(child:string);	Gets Float value of child node
getchildfloatdef	ChildIFloatvalue = xmlfile:Getchildintdef(child:string;value:number);	Gets Float value of child node or value
getchildvalue	Childvalue = xmlfile:Getchildvalue(childname:string);	Returns child value
savetofile	xmlfile:savetofile(filename:string);	Writes the XML file to disk
writetostring	Xmlfile:writetostring()	Returns a string with the contents of the XML

Examples

Name	Syntax	Description
childexists	xmlfile:childexists("*parameters/fixingmode*", 2);	*boolean:* returns **true** if the desired node exists
addvalue	xmlfile:addvalue("*parameters/fixingmode*", 2);	-
savetofile	xmlfile:savetofile("*var/log/test.xml*");	-
dump	xmlfile:dump()	-

2.9.2 XML node Object

[Desktop Automation]

An XML node object contains the full information of an XML node.

2.9.2.1 Properties

Property	Read / Write	Type	Description
childcount	read only	Number	Returns the number of xml nodes of the root entry
section	Read / write	String	Section name of the root entry
value	Read only	String	Value of the node

2.9.2.2 Method Overview

Name	Syntax	Description
addchild	node:Addchild(childname:string, adddouble:boolean);	Adds child node, adddouble allows for duplicates
addvalue	node:addvalue(nodename:string, value:string/int);	Adds a value to the XML file
childexists	node:childexists(nodename:string);	Returns if nodename in the XML file exists
findchild	Childnode = node:Findchild(childname:string);	Finds childnode
getattribut	node:getattribute(nodename:string)	Gets string value of attribute of a node
getattributeint	IntValue = node:getattributeint(nodename:string);	Gets integer value of node
getattributefloat	FloatValue = node:getattributefloat(nodename:string);	Gets float value of attribute of a node
getchildint	ChildIntvalue = node:Getchildint(child:string);	Gets the Integer value of child node
getchildindexed	Childvalue = node:Getchildindexed(childindex:string);	Returns child value by index
getchildintdef	ChildIntvalue = node:Getchildintdef(child:string;value:number);	Gets the Integer value of child or value
getchildfloat	ChildIntvalue = node:Getchildfloat(child:string);	Gets Float value of child node
getchildfloatdef	ChildIFloatvalue = node:Getchildintdef(child:string;value:number);	Gets Float value of child node or value
getchildvalue	Childvalue = node:Getchildvalue(childname:string);	Returns child value
hasattribute	Boolean Value = node:hasattribute(attribute_name: String)	Returns true if a node has a specific attribute, false otherwise

2.9.3 Json file Object

[Desktop Automation]

The Json file object represents a Json file.

2.9.3.1 Properties

None.

2.9.3.2 Method Overview

Name	Syntax	Description
addboolean	Jsonfile:addboolean(name:String, value:Boolean);	Adds a Boolean value to a name-value pair.
addinteger	Jsonfile:addinteger(name:String, value:Integer);	Adds an Integer value to a name-value pair.
addfloat	Jsonfile:addfloat(name:String, value:Float);	Adds a Float value to a name-value pair.

addstring	Jsonfile:addstring(name:String, value:String);	Adds a String value to a name-value pair.
childexists	Ret:boolean = Jsonfile:childexists(name:String);	Returns TRUE if in the Jsonfile a name-value pair exists with the name 'name'.
getboolean	Ret:boolean = Jsonfile:getboolean(name:String);	Returns the boolean value of the 'name' name-value pair
getinteger	Ret:Integer = Jsonfile:getinteger(name:String, value:Integer);	Returns the integer value of the 'name' name-value pair
getfloat	Ret:Float = Jsonfile:getfloat(name:String, value:Float);	Returns the float value of the 'name' name-value pair
getstring	Ret:String = Jsonfile:getstring(name:String, value:String);	Returns the string value of the 'name' name-value pair
loadfromstring	Jsonfile: loadfromstring (string:String);	Loads Jsonfile from 'string'
savetofile	Jsonfile:savetofile(filename:String);	Saves the Jsonfile under the 'filename' name.
writetostring	string:String = Jsonfile: writetostring ();	Returns the Jsonfile as string.

2.9.4 CSV file Object

[Desktop Automation]

The text file object represents a CSV file. The file consists of string entries inside a matrix of rows and colums (also known as fields) . The matrix entries are called cells. The number of colums is defined by the first row. If you add a new row with more columns than the first row a error is thrown.

2.9.4.1 Properties

Property	Read / Write	Type	Description
rowlength	read only	Number	Returns the number of fields
tablelength	read only	Number	Returns the number of lines

2.9.4.2 Method Overview

Name	Syntax	Description
appendfilewithentry	CSVfile:appendfilewithentry (cell:string [, lineending:Boolean]);	Appends one cell to the file. If 'lineending' is TRUE, a newline is added at the end.
appendfilewithline	CSVfile:appendfilewithline (line:string);	Appends one line to the file.
getcell	Field:string = CSVfile:getcell(row:number; field:number [, replace:boolean]);	Reads the cell at 'row' and 'field'. If 'replace' is TRUE, quotationsmarks are removed.
readfile	CSVfile:readfile(filename:string);	Reads the 'filename' from disk.
savetofile	CSVfile:savetofile(filename:string);	Saves the CSV file to 'filename', overwrites an existing file depending on the Boolean value at creation of the object.

2.9.5 Text File Object

[Desktop Automation]

The text file object represents a text file.

2.9.5.1 Properties

Property	Read / Write	Type	Description
linecount	read only	Number	Returns the number of lines

2.9.5.2 Method Overview

Name	Syntax	Description
clear	Txtfile:clear();	Empties the textfile object
getline	resultline = txtfile:getline(linenumber:number);	Returns textline at linenumber
loadfromfile	Result = txtfile:loadfromfile(filename:string);	Loads text file from disk, returns TRUE when successful
savetofile	Result = txtfile:savetofile(filename:string);	Writes textfile, TRUE, when successful
setline	Result = txtfile:setline(linenumber:number; text:string);	Writes 'text' at line number: 'linenumber', TRUE when successful
writeline	txtfile:writeline(text:string);	Appends 'text' to text file

2.10 Database related Objects

2.10.1 Database Connection Object

[Desktop Automation]

Database connection and query handler.

2.10.1.1 Properties

Property	Read / Write	Type	Description
connected	read only	Boolean	Returns TRUE, if the DB connection is established
lasterror	Read only	String	The last DB error

2.10.1.2 Method Overview

Name	Syntax	Description
checkiftableexists	Result = dbconnection:checkiftableexists(tablename:string);	Returns TRUE, if table with tablename exists
createttable	Result = dbconnection:createttable(tablename:string);	Creates a new table
disconnect	dbconnection:disconnect();	Disconnects from the DB
getinsertid	insertid = dbconnection:getinsertid();	Returns the last insert id
getresult	Queryresult = dbconnection:getresult();	Returns the QueryResult Object

getuniquestring	uniquestring = dbconnection:getuniquestring();	Gets unique string
sendquery	Result = dbconnection:sendquery(query:string);	Returns TRUE when successful, the query result can be evaluated wuth "getresult".

2.10.2 Query Result Object

[Desktop Automation]

The result object of a database query.

2.10.2.1 Properties

None.

2.10.2.2 Method Overview

Name	Syntax	Description
getintegerfield	IntResult = queryresult:getintegerfield(fieldnumber:number);	Returns the Integer value of the field in fieldnumber
getfield	StringResult = queryresult:getfield(fieldnumber:number);	Returns the String value of the field in fieldnumber
getfieldcount	Result = queryresult:getfieldcount();	Returns number of fields in row
getfloatfield	FloatResult = queryresult:getintegerfield(fieldnumber:number);	Returns the Float value of the field in fieldnumber
nextrow	queryresult:nextrow();	Put the result pointer to the next row

2.11 Image and Display related Objects

2.11.1 OGLRendering Object

[Desktop Automation]

A render context for previews. The OGLRendering Object is created by calling " system:createoglcontext". The rendering object itself works with:

- models: individual meshes
- scenes: arrangements of models

2.11.1.1 Properties

None.

2.11.1.2 Method Overview

Name	Syntax	Description

addmodeltoscene	oglcontext:addmodeltoscene (modelid:integer);	Adds the model of 'modelID' to the current scene.
createmodel	Modelid:integer = oglcontext:createmodel (mesh:mesh object, loadcolors: boolean, loadtextures: boolean, matrix: tluamatrix)	Puts a mesh object into the oglrender context, returns the modelID. loadcolors: optional parameter, default is true. Flag to allow loading of vertex colors.Setting the flag to false allows the setting of a uniform color using setmodelcolor. loadtextures: optional parameter, default is true. Flag to allow loading of vertex textures. matrix: optional parameter, transforms the model and sets the transformed bounding box when set
freemodel	oglcontext:Freemodel (modelid:integer);	Remoces model from scene and deletes reference
identity	*oglcontext:identity ();*	*DEBUG ONLY: Load the OpenGL identity matrix ("glloadidentity()")*
lookatmodelfromsurroundingsphere	oglcontext:lookatmodelfromsurroundingsphere (modelid, eyex, eyey, eyez, upx, upy, upz, offset: all integer);	Sets the camera position eye (x,y,z) = eye vector up (x,y,z) = the UP vector describes the roll of the camera by saying which point is "up" in the camera's orientation. offset = distance to mesh object (for more information: search for opengl and eye and up vector)
multmatrix	*oglcontext:multmatrix (modelid + 16 integer);*	*DEBUG ONLY*
popmodelmatrix	*oglcontext:popmodelmatrix (modelid:integer);*	*DEBUG ONLY*
pushimage	*oglcontext:pushimage ();*	*DEBUG ONLY*
pushmodelmatrix	*oglcontext:pushmodelmatrix (modelid:integer);*	*DEBUG ONLY*

release	*oglcontext:release ();*	*DEBUG ONLY: Releases the current oglcontext (allowing to create a new one)*
releaserawbuffer	*oglcontext:releaserawbuffer ();*	*DEBUG ONLY*
render	oglcontext:Render ();	Renders the scene
rotate	oglcontext:Rotate (Modelid:integer, angle: integer);	Rotates the model across the camera vector (needs to be set up first with Lookatmodelfromsurroundingsphere)
savetobmp	oglcontext:Savetobmp (filename:string, threaded: boolean);	Saves current scene to BMP file as 'filename'. Threaded: optional parameter, when true the export runs in a separate thread
savetojpeg	oglcontext:savetojpeg (filename:string, quality: integer, threaded: boolean);	Saves current scene to JPEG file as 'filename'. Quality: JPEG quality, default: 90 Threaded: optional parameter, when true the export runs in a separate thread
savetopng	oglcontext:savetopng (filename:string, threaded: boolean);	Saves current scene to PNG file as 'filename'. Threaded: optional parameter, when true the export runs in a separate thread
scale	oglcontext:Scale (x,y,z:all float);	Scales the context
setBackgroundColor	oglcontext:SetBackgroundColor (r, g, b: all integer);	Sets background colour to RGB value

setBackgroundGradient	oglcontext:SetBackgroundGradient (ar, ag, ab, br, bg, bb, cr, cg, cb, dr, dg, db: all integer);	Background gradient map: RBG values for the points a, b, c and d.
		A = bottom left
		B = bottom right
		C = top right
		D = top left

setenvironmentmodel	oglcontext:Setenvironmentmodel (modelid:integer);	Sets the environment model
setModelColor	oglcontext:setmodelcolor(modelid: integer, r, g, b, a: float)	Changes the uniform color of a model. In order to usethis method vertex colors need to be disabled in the createmodel call.
setModelTextureEnabled	oglcontext:setmodeltextureenabled(textureenabled: boolean)	Changes the textureendabled flag

setlightpos	oglcontext:Setlightpos (x,y,z,type: all integer);	Sets the position of lightsource, type 1 = point lightsource, otherwise unidirectional
setreflectivity	oglcontext:Setreflectivity (modelid: integer, reflectivity: float);	Sets the reflectivity factor (e.g. "0.6") for the model
swapbuffers	oglcontext:swapbuffers ();	OpenGL glutswapbuffers function
removemodelfromscene	oglcontext:Removemodelfromscene (modelid:integer);	Removes model from scene
translate	*oglcontext:translate (x,y,z:all float);*	*DEBUG ONLY: OpenGL translate function*

2.11.2 Image Processing

[Desktop Automation]

The Lua Image Processing Interface provides functionality for simple image processing and algorithms.

2.11.2.1 Properties

None.

2.11.2.2 Method Overview

Name	Syntax	Description
loadimage	Imageobject = Ip:loadimage(file: string)	Loads image from specified file and returns a lua-image-object

2.11.3 Image Object

[Desktop Automation]

All colors are encoded as Integer 0xBBGGRR

2.11.3.1 Properties

Property	Read / Write	Type	Description
height	read only	Number	Height of the image
width	read only	Number	Width of the image

2.11.3.2 Method Overview

Name	Syntax	Description
colortotransparent	img:colortotransparent(color: number)	Converts given color to transparency.

compareto	img:compareto(img: object)	Compares the image to another specified image pixel by pixel. The result ist the percentage of equality (50% = random image, 100% = equal image).
comparebyfiltercolor	Img:comparebyfiltercolor(img: object; color: number)	Compares the image to another specified image pixel by pixel. Just respect pixels with given mask-color The result ist the percentage of equality
deltato	img:deltato(img: object)	Calculates the difference between the two images. Same areas are black, difrerences are higlighted white
detectedges	img:detectedges()	Detects edges of the image by the canny-algorithm. The resulting image is black with white one pixel wide edges.
gausssmoothing	img:gausssmoothing(sigmar: number)	Performs the gausssmoothing filter with specified sigmar value
colormask	img:colormask(mask: object; maskcolor: color; newcolor: number)	Applies a mask operation on the image where every pixel of the image is replaced by "*newcolor*" where the color of the "*mask*" image is "*maskcolor*" [Desktop Automation]
imagemask	Img:imagemask(mask: object; maskcolor: number)	Applies a mask operation on the image where every pixel of the image is replaced by mask-pixel if the input-pixel equals maskcolor.
invert	img:invert()	Inverts colors of the image
laplace	img:laplace()	Performs the laplace filter on the image
paint	img:paint(img: object)	Draw a image over another image. Transparent areas in the top image will be ignored.
saveto	img:saveto(file: string)	Saves image to specified file
setcolor	img:setcolor(color: number)	Override the color of the entire image. Transparency will stay the same.
tograyscale	img:tograyscale()	Converts colored image to grayscale

2.12 CAD Import related Objects

2.12.1 CadImporter Object

[Desktop Automation]

2.12.1.1 Properties

None.

2.12.1.2 Method Overview

Name	Syntax	Description
loadmodel	cadmodel = importer:loadmodel('handle.step', 0.1, 20, 5)	Syntax 1: Load the model "handel.step" with a maximum surface deviation of 0.1 and a angle tolerance and a maximal edge length of 5 mm Syntax 2: loadmodel(filename, detaillevel) where detaillevel is a number 1-5 corresponding to the 5 Detail settings of the UI in Netfabb: 1 = extra low 2 = low 3 = medium 4 = high 5 = Extra high

2.12.2 CADImportModel Object

[Desktop Automation]

Sometimes a CAD File consists of several single files. You can access the number of single entities by the property *entitycount* and get an entity's name by the method *getentityname(Index).* You can create a mesh with all entities using the method *createmesh* or you can create a mesh with a single entity using the property *createsinglemesh.*

2.12.2.1 Properties

Name	Syntax	Description
entitycount	entities = cadmodel.entitycount	Holds the number of entities of the cadmodel
trianglecount	triangles = cadmodel.trianglecount	Holds the number of triangles of the cadmodel

2.12.2.2 Method Overview

Name	Syntax	Description
createsinglemesh	mesh = cadmodel:createsinglemesh (EntityIndex: Number);	Creates a mesh with a single entity
createmesh	mesh = cadmodel:createmesh()	Creates a mesh with all entities

getentityname	name = cadmodel:getentityname(EntityIndex: Number);	Returns the entity's name

2.13 Utility Data Objects

[Desktop Automation]

2.13.1 LUAVector3

The LUA representation of a vector with 3 components (x, y, z)

2.13.1.1 Properties

Property	Read / Write	Type	Description
x	Read / write	Number	X-Component of the vector
y	Read / write	Number	Y-Component of the vector
z	Read / write	Number	Z-Component of the vector

2.13.1.2 Methods

None.

2.13.2 LUAArray

2.13.2.1 Properties

Property	Read / Write	Type	Description
length	Read	Integer	Length of the array

2.13.2.2 Methods

Name	Syntax	Description
get	array:get(index: integer)	Get the array value at index
set	array:set(index: integer; value: number)	Set the array value at index

2.13.3 LUAStringMap

The LUA representation of a map of strings (key / value pairs)

2.13.3.1 Properties

Property	Read / Write	Type	Description
count	Read	Integer	Number of items in the map

2.13.3.2 Methods

Name	Syntax	Description
setitem	Map:setitem(AKey: String; Avalue: String)	Set an Item (add or replace)
getitem	Item = Map:getitem(AKey: String)	Get an Item by key
deleteitem	Map:deleteitem(AKey: String)	Delete an item with a key
getitembyindex	KeyItem = Map:getitembyindex(AIndex: Integer)	Get an item with key by index (Format: “Key”=”Value”)

2.13.4 LUAMatrix4f

The Lua representation of a 4x4 Matrix.

2.13.4.1 Properties

None

2.13.4.2 Methods

Name	Syntax	Description
get	matrix:get(x: integer; y: integer)	Get the matrix value at x and y
set	matrix:set(x: integer; y: integer; value: number)	Set the matrix value at x and y

2.13.5 LUAAlignment

2.13.5.1 Properties

Property	Read / Write	Type	Description
firstaxis	Read	TLUAVector3f	First principal axis designating the shortest axis of the reference mesh. Can be used to set up the camera for rendering
newmodeltoworldmatrix	Read	TLUAMatrix4f	A new model to world matrix for the mesh/model to be transformed
secondaxis	Read	TLUAVector3f	Second principal axis designating the longest axis of the reference mesh. Can be used to set up the camera for rendering
thirdaxis	Read	TLUAVector3f	Third principal axis designating the remaining axis of the reference mesh. Can be used to set up the camera for rendering
transformationmatrix	Read	TLUAMatrix4f	The matrix that needs to be multiplied with the model to world matrix to align the mesh/model to be aligned with the reference mesh/model

2.13.5.2 Methods

None

2.13.6 Outbox Object

Represents an outbox

2.13.6.1 Properties

Property	Read / Write	Type	Description
minx	Read/write	Number	Returns the min_x (in mm)
miny	Read/write	Number	Returns the min_y (in mm)
minz	Read/write	Number	Returns the min_z (in mm)
maxx	Read/write	Number	Returns the max_x (in mm)
maxy	Read/write	Number	Returns the max_y (in mm)
maxz	Read/write	Number	Returns the max_z (in mm)

2.13.6.2 Method Overview

None.

2.14 Webservices related Objects

2.14.1 Netfabb Taskserver Object

[Desktop Automation]

The Netfabb Taskserver is an Addon service program, allowing to distribute tasks to Netfab Ultimate Clients. The object allows to connect to this service and to act as a client as well as a worker instance to a complete system. For details please refer to the according separate documentation.

2.14.1.1 Properties

Property	Read / Write	Type	Description
lasterror	Read	String	The description of the last error, that has occured

2.14.1.2 Method Overview

Name	Syntax	Description
authenticate	Ret:boolean = Taskhandler:authenticate(userid:String, serverkey: String);	Authenticates the taskhandller with a userid and the serverkey (shared secret/passphrase)

checktask	Task:Taskobject = Taskhandler:checktask(uuidstring:string);	Returns task object by “uuidstring”. Use to find “your” task.
createtask	Task:taskobject = Taskhandler:createtask(taskname:String)	Creates a new task with the name “taskname”
retrievetask	Task:taskobject = taskhandler:retrievetask(taskname:String)	Retrieves a task object by name of “taskname”. To use by a “worker” to find a task of a certain “type” (name).

2.14.2 Netfabb Task Object

[Desktop Automation]

2.14.2.1 Properties

Property	Read / Write	Type	Description
lasterror	Read	String	The description of the last error, that has occured
parametercount	Read	Number	Number of parameters of the task
resultcount	Read	Number	Number of results for the task
status	Read	String	A status string. One of: SUCCESS ERROR RETURNED CANCELED NEW INPROCESS “NEW” and “INPROCESS” can’t be set by the Netfabb Client API, this is only done by the Taskserver itself.
taskname	Read	String	Name of the task
uuid	Read	String	UUID of the task

2.14.2.2 Method Overview

Name	Syntax	Description
addparameter	Task:addparameter(key:String; value: String)	Adds a parameter to the task as a key/value pair.
addresult	Task:addresult(key:String; value: String)	Adds a result to the task as a key/value pair.
cancel	Result:Boolean = Task: cancel ()	Sets the status of the task to “CANCELED”
error	Result:Boolean = Task: error()	Sets the status of the task to “ERROR”
getparameterbyindex	param:String = Task:getparameterbyindex(index:Number)	Gets a parameter value from the task by the index number of the parameter
getparameterbyname	param:String = task:getparameterbyname(keyname:String)	Gets a parameter value from the task by the name of the key of the parameter

getparametername	param:String = task:getparametername(index:Number)	Get the key name of a parameter by its index number
getresultbyindex	Result:String = task:getresultbyindex(index:Number)	Gets a result value from the task by the index number of the result
getresultbyname	Result:String = task:getresultbyname(keyname:String)	Gets a result value from the task by the name of the key of the result
getresultname	result:String = task:getresultname(index:Number)	Get the key name of a result by its index number
giveback	Result:Boolean = Task: giveback()	Sets the status of the task to "RETURNED"
submit	Result:Boolean = Task:submit()	Submits the task to the Taskserver. Returns TRUE if successful.
success	Result:Boolean = Task: success ()	Sets the status of the task to "SUCCESS"
updatestatus	Result:Boolean = Task: updatestatus ()	Poll task for updated status

2.14.3 Hublist Object

[Desktop Automation]

Hublist objects are created from these methods of the Application object:

- loadappserverhublist and
- loadforgehublist

Both calls return a list of hubs from the respective context, either the Autodesk Forge Cloud system or the Netfabb Application Server local storage definition.

2.14.3.1 Properties

Property	Read / Write	Type	Description
count	Read	String	Number of hubs in this list

2.14.3.2 Method Overview

Name	Syntax	Description
gethub	Hub = Hublist:gethub(index:Integer)	Returns hub object by "index". Index list starts with 0.

2.14.4 Hub Object

[Desktop Automation]

A hub is a collection of projects, either on Autodesk Forge or in the Netfabb Application Server.

2.14.4.1 Properties

Property	Read / Write	Type	Description
id	Read	String	ID of this hub
name	Read	String	Name of this hub

2.14.4.2 Method Overview

Name	Syntax	Description
createproject	Project = Hub:createproject(name:String);	Create a new project with “name” on the hub
getprojectlist	Projectlist = hub:getprojectlist();	Returns the list of projects in this hub.

2.14.5 Projectlist Object

[Desktop Automation]

A projectlist is a list of projects from a hub, either from Autodesk Forge or from the Netfabb Application Server.

2.14.5.1 Properties

Property	Read / Write	Type	Description
count	Read	String	Number of projects in this list

2.14.5.2 Method Overview

Name	Syntax	Description
getproject	Project = projectlist:getproject(index:Integer);	Returns a project by its index number.

2.14.6 Project Object

[Desktop Automation]

A project is a collection of folders from a hub, either from Autodesk Forge or from the Netfabb Application Server.

2.14.6.1 Properties

Property	Read / Write	Type	Description

hasavatar	Read	Boolean	Whether this project has an avatar image
Id	Read	String	Id of the project
name	Read	String	Name of the project

2.14.6.2 Method Overview

Name	Syntax	Description
getrootfolderlist	folderlist = project:getrootfolderlist();	Returns the root folderlist of the project, nil if empty.

2.14.7 Folderlist Object

[Desktop Automation]

A folderlist is a list of folders from a project, either from Autodesk Forge or from the Netfabb Application Server.

2.14.7.1 Properties

Property	Read / Write	Type	Description
count	Read	Integer	Number of folders in this list

2.14.7.2 Method Overview

Name	Syntax	Description
getfolder	Folder = folderlist:getfolder(index:Integer);	Gets a folder from a folderlist by index number. Numbers start at 0.

2.14.8 Folder Object

[Desktop Automation]

A folder is an object from a project, either from Autodesk Forge or from the Netfabb Application Server.

2.14.8.1 Properties

Property	Read / Write	Type	Description
contenthasbeenreceived	Read	Boolean	Has content been received?
forcehidden	Read	Boolean	Hide the folder
id	Read	String	Id of folder
ishidden	Read	Boolean	Is the folder hidden?
name	Read	String	Name of folder
projectid	Read	String	ProjectID to this folder

2.14.8.2 Method Overview

Name	Syntax	Description
createsubfolder	Folder:createsubfolder(foldername:String);	Creates foldername as subfolder in folder.
getitemlist	Itemlist = Folder:getitemlist();	Returns list of items from folder object, otherwise returns NIL
getsubfolders	Folderlist = Folder:getsubfolders();	Returns list of subfolders from folder object, otherwise returns NIL.
receivecontent	Folder:receivecontent();	Scans the folder for items and subfolders. Has to be called once before “getitemlist” or “getsubfolders”

2.14.9 Itemlist Object

[Desktop Automation]

An itemlist is a list of items from a projectfolder, either from Autodesk Forge or from the Netfabb Application Server.

2.14.9.1 Properties

Property	Read / Write	Type	Description
count	Read	Integer	Number of items in list

2.14.9.2 Method Overview

Name	Syntax	Description
finditembyname	Item = Itemlist:finditembyname(name:String, [Casesenitive:Boolean]);	Get item by name, optional casesensitive.
getitem	Item = Itemlist:getitem(index:Integer);	Get an item by index number.

2.14.10 Item Object

[Desktop Automation]

An item is a content element from a projectfolder, either from Autodesk Forge or from the Netfabb Application Server.

2.14.10.1 Properties

Property	Read / Write	Type	Description
createuser	Read	String	User, who has created the item
folderid	Read	String	Id of the folder
id	Read	String	Id of the item
lastmodifier	Read	String	User, who has last modified the item

name	Read	String	Name of item
projectid	Read	String	Project id of item

2.14.10.2 Method Overview

None.

2.15 Miscellaneous Objects

2.15.1 LUAStamper

[Desktop Automation]

This Lua objects allows to label meshes. A text is stamped on the meshes. The position of the text, the estimated plane of the text, and the direction where is up needs to be given. Please note that the example script Script17_LabelMeshes.lua has some helper functions for setting the position, normal, and upvector.

2.15.1.1 Properties

Property	Read / Write	Type	Description
depth	Read /write	number	The depth of the label.
height	Read /write	number	The height of the label in mm. The width is scaled to match the original ratio.
isinverted	Read /write	boolean	If true the text is inverted
issubtracted	Read /write	boolean	If false, the label is added to meshes, otherwise is subtracted
pos	Read	TLUAVector3f	The position of the label.
normal	Read	TLUAVector3f	The normal of the plane of the label.
upvector	Read	TLUAVector3f	The defines, what is up and down.

2.15.1.2 Methods

Name	Syntax	Description
stamp	newmesh = stamper:stamp(Mesh: LUAMesh, text : string)	Labels the text on the mesh. Creates a new mesh as an output.
setpos	stamper:setpos(APos1: number, APos2: number, APos3: number)	Sets the positon. This is the center of the text.
setnormal	stamper:setnormal(APos1: number, APos2: number, APos3: number)	Sets the normal of the plane of the label. For example Vec (0,0,1) would be the x,y plane.

setupvector	stamper:setupvector(APos1: number, APos2: number, APos3: number)	If a plane is given for the label, one needs to define what angle the text should have in the plane, i.e. in what direction the text is printed. 'setupvector' sets the upvector, which defines what direction is defined as "up" for a text. The Vector should to be perdendicular to the normal vector.

2.15.2 LUAReportgenerator

[Desktop Automation]

This Lua objects allows to generate reports.

2.15.2.1 Properties

None

2.15.2.2 Methods

Name	Syntax	Description
createreportformesh	reportgenerator:createreportformesh (Mesh: LUATrayMesh, template : string; reportname : string)	Takes a TrayMesh and generates a report based on the given template. The report is saved in file called reportname. For optimal output the template type should correspond to a tray.
createreportfortray	stamper:setpos(APos1: number, APos2: number, APos2: number)	Takes a tray and generates a report based on the given template. The report is saved in file called reportname. For optimal output the template type should correspond to a tray.

2.15.3 Test framework interface

[Desktop Automation]

The Lua test framework provides functionality for uniform test suites and convenient result file formats.

2.15.3.1 Properties

Property	Read / Write	Type	Description
childcount	read only	Number	Number of direct sub test suites
errormessage	read / write	String	Errormessage of this test on failure

duration	read / write	Number	Duration of this test in seconds
failurecount	read	Number	Number of failed sub tests
name	read / write	String	Name of the test suite
success	read / write	Boolean	True for success, false for failure
testcount	read	Number	Number of all sub tests

2.15.3.2 Method overview

Name	Syntax	Description
asserttrue	suite:asserttrue(emassage: string; value: boolean)	Asserts for value = true. For false value, the test fails and sets the errormessage
assertequalsmeshgeometry	suite:asserequalsmeshgeometry(emassage: string; mesh1, mesh2: MeshObject; accuracy: Number)	Asserts for equal geometry of the two meshes with meshcompare considering to the optional accuracy. For not equal meshes, the test fails and sets the errormessage.
assertequalsmeshproperties	suite:assertequalsmeshproperties(emassage: string; mesh1, mesh2: MeshObject)	Asserts for equal properties of the two meshes: NodeCount, EdgeCont and FaceCount. For not equal meshes, the test fails and sets the errormessage.
assertequalsnumber	suite:assertequalsnumber(emassage: string; n1, n2, accuracy: Number)	Asserts for n1 = n2 with optional accuracy. For not equal numbers, the test fails and sets the errormessage.
createtestsuite	subsuite = suite:createtestsuite(name: string)	Creates a new sub test suite for the current suite.
fails	suite:fails(emassage: string)	The test fails. The parameter specifies the error message of the failed test.
finishtest	suite:finishtest()	Finishes the current test. This call calculates the duration of the test.
savetocsv	suite:savetocsv(file: string)	Saves the test result into specified file. The format is a comma separated format.
savetojunitxml	suite:savetojunitxml(file: string)	Saves the test result into specified file. The format is a junit convenient xml format.

2.15.4 ZipObject

[Desktop Automation]

This object is used for creating/managing ZIP files.

2.15.4.1 Properties

Property	Read / Write	Type	Description
count	read only	Number	Number of files in the open ZIP archive. Only available in archives opened for reading
canread	read only	Boolean	Flag indicating wether or not the archive can be read from. Only functions marked in green are available if this flag is set
canwrite	read only	Boolean	Flag indicating wether or not the archive can be written to. Only functions marked in yellow are available if this flag is set
password	read / write	String	Password of the ZIP archive.

2.15.4.2 Method Overview

Please not that the “add*” and the “exportfile” methods are only available in ZIP files opened for writing using “system:createzip” whereas the “load*” and “get*” methods are only available when the file was opened for reading using either “system:openzip”.

Availability of the functions is marked by colors for archives open for reading and writing.

Name	Syntax	Description
addstring	Zipobject:addstring(stringtoadd:string, zipname:string);	Adds stringtoadd to zipobject as zipname
addfile	Zipobject:addfile(filetoadd:string, zipname:string);	Adds filetoadd to zipobject as zipname
addtext	Zipobject:addtext(texttoadd:Lua Textfile object, zipname:string);	Adds texttoadd (a Lua textfile object) to zipobject as zipname
addxml	Zipobject:addxml(xmltoadd:Lua XML object, zipname:string);	Adds xmltoadd (a Lua XML object) to zipobject as zipname
exportfile	Zipobject:exportfile(filename:string);	Exports the ZIPfile as filename. File cannot be changed afterwards by API. This function is only available if a ZIP is created using the LUA API
extractfile	Zipobject:extractfile(filename: String; targetname: String)	Saves the file “filename” from the archive to the file “targetname” on the file system
getfilename	Zipobject:getfilename(index: number)	Returns the name of the file with the index <index> of the open zip file
getfileindex	Zipobject:getfileindex(filename: String)	Returns the index of the file with the name <filename> of the zip file

loadimage	Zipobject:loadimage(filename: String)	Loads an image from an open ZIP file. Returns an instance of Image
loadjson	Zipobject:loadjson(filename: String)	Loads a json file from an open ZIP file. Returns an instance of Json file Object
loadtextfile	Zipobject:loadtextfile(filename: String)	Loads a text file from an open ZIP file. Returns an instance of Text File Object
loadxml	Zipobject:loadxml(filename: String)	Loads a XML file from an open ZIP file. Returns an instance of XML file Object

2.15.5 PartOrienter

[Desktop Automation]

This object allows to orient a part with several options. The orienter calculates several solutions with its criteria values.

2.15.5.1 Properties

Property	Read / Write	Type	Description
cutoff_radian	Read/ Write	Number	Defines the angle where the part needs support structures in radian
cutoff_degree	Write	Number	Defines the angle where the part needs support structures in degree
distance_from_platform	Read/ Write	Number	Resulting distance to platform
rotation_axis	Read/ Write	String	Defines freedom of rotations. Valid values are 'arbitrary', 'x', 'y'
smallest_distance_between_minima_degree	Write	Number	Defines the minimal rotation distance between to calculated solutions. The smaller the angle the more solution will be calculated
smallest_distance_between_minima_radian	Read/ Write	Number	Defines the minimal rotation distance between to calculated solutions. The smaller the angle the more solution will be calculated
solutioncount	Read	Number	Number of calculated solution
support_bottom_surface	Read/ Write	Boolean	Defines wherer the buttom surfaces needs support or not. That influences the support-area critiria.

2.15.5.2 Method Overview

Name	Syntax	Description
get_best_solution_for	s = orienter:get_best_solution_for(criteria: String);	Returns the best orientation for the specified critiria. The following criteria are allowed: 'outbox_volume', 'part_height', 'support_area' ,'support_volume' and ‘center_of_gravity_height’
get_matrix_from_solution	matrix = orienter:get_matrix_from_solution(solution: lua_json)	Creates a matrix from a solution json object. This matrix can be applied by a mesh object. The mesh is rotated around its center of gravity to reflect the best orientation.
get_solution	s = orienter:get_solution(Index: Number);	Return the orientation solution for specific index
search_orientation	orienter:search_orientation();	Calculates part orientations.
search_orientation_with_progress	orienter:search_orientation_with_progress (start_progress: Number; end_progress: Number);	Calculates part orientations and shows a progress bar. Start and end progress numbers are optional percentage values. This is usefull to show the progress-bar in a specific range (e.g. 20% to 40%)

An orientation solution is a lua_json object. The json object has the the following attributes:

Attribute	Description
outbox_volume	Outbox volume of the part
part_height	Part height
support_area	Support area of the part
support_volume	Support volume of the part
center_of_gravity_x	Center of Gravity x
center_of_gravity_y	Center of Gravity y
center_of_gravity_z	Center of Gravity z
rotation_axis_x	Rotation axis in x for this orientation
rotation_axis_y	Rotation axis in y for this orientation
rotation_axis_z	Rotation axis in z for this orientation
rotation_radian	Rotation angle in radian this orientation
rotation_degree	Rotation angle in degree this orientation
rotation_matrix	Rotation 4*4 matrix as string. The matrix is colum-major.

2.15.6 PartAnalysis

[Desktop Automation]

This object is used to run analyses on the underlying mesh. Those include the default, center of gravity, wall thickness, support volume, shadow area and up-skin/downskin analyses. The available methods run the analyses, the properties can then be used to get results, with each analysis having a property to indicate successful calculation. The object is created from a mesh object by:

TLUAPartAnalysis analysis = mesh:createanalyzer ().

2.15.6.1 Properties

Property	Read / Write	Type	Description	Analysis
averagewallthickness	Read	Number	Returns the average wall thickness	Wallthickness Analysis
badedges	Read	Number	Returns the number of invalid edges	Default Analysis
boundaryedges	Read	Number	Returns the number of edges making up holes	Default Analysis
boundarylength	Read	Number	Returns the total length of all boundary edges	Default Analysis
centerofgravityx	Read	Number	Returns the x value of the mesh's center of gravity	Center of Gravity Analysis
centerofgravityy	Read	Number	Returns the y value of the mesh's center of gravity	Center of Gravity Analysis
centerofgravityz	Read	Number	Returns the z value of the mesh's center of gravity	Center of Gravity Analysis
coganalysiswassuccessful	Read	Boolean	Returns true if no errors occurred during the calculation of the analysis	Center of Gravity Analysis
defaultanalysiswassuccessful	Read	Boolean	Returns true if no errors occurred during the calculation of the analysis	Default Analysis
downskinangle	Read	Number	Returns the entered downskin angle threshold between the z plane and the triangle	Upskin Downskin Analysis
downskinarea	Read	Number	Returns the total area of all downskin triangles	Upskin Downskin Analysis
downskincomponentcount	Read	Number	Returns the number of downskin components	Upskin Downskin Analysis

edgecount	Read	Number	Returns the number of edges	Default Analysis
facecount	Read	Number	Returns the number of triangles	Default Analysis
flippedtrianglecount	Read	Number	Returns the number of flipped triangles	Default Analysis
holecount	Read	Number	Returns the number of holes	Default Analysis
isorientable	Read	Boolean	Returns true if the meshes triangles can be flipped to form a closed mesh	Default Analysis
mesharea	Read	Number	Returns the total area of the mesh's trianles	Default Analysis
meshisclosed	Read	Boolean	Returns true if the mesh has no holes	Default Analysis
meshisok	Read	Boolean	Returns true if the mesh is orientable and closed	Default Analysis
meshvolume	Read	Number	Returns the volume of the closed mesh or 0 if the mesh contains flipped triangles or is open	Default Analysis
nodecoount	Read	Number	Returns the number of nodes	Default Analysis
outboxmaxx	Read	Number	Returns the max_x of the outbox (in mm)	Default Analysis
outboxmaxy	Read	Number	Returns the max_y of the outbox (in mm)	Default Analysis
outboxmaxz	Read	Number	Returns the max_z of the outbox (in mm)	Default Analysis
outboxminx	Read	Number	Returns the min_x of the outbox (in mm)	Default Analysis
outboxminy	Read	Number	Returns the min_y of the outbox (in mm)	Default Analysis
outboxminz	Read	Number	Returns the min_z of the outbox (in mm)	Default Analysis
outboxsizex	Read	Number	Returns the length of the outbox in x direction	Default Analysis
outboxsizey	Read	Number	Returns the length of the outbox in y direction	Default Analysis
outboxsizez	Read	Number	Returns the length of the outbox in x direction	Default Analysis
shadowarea	Read	Number	Returns the area of the mesh's shadow at z = 0	Shadow Area Analysis
shadowareaanalysiswassuccessful	Read	Boolean	Returns true if no errors occurred during the calculation of the analysis	Shadow Area Analysis

supportangle	Read	Number	Returns the angle used to calculate support clusters used as basis for the support shells	Support Volume Analysis
supportvolume	Read	Number	Returns the volume of the support shells	Support Volume Analysis
supportvolumeanalysiswassuccessful	Read	Boolean	Returns true if no errors occurred during the calculation of the analysis	Support Volume Analysis
testcriticaldistance	Read	Number	Returns 1 the test passed, 0 otherwise	Wallthickness Analysis
updownskinanalysiswassuccessful	Read	Boolean	Returns true if no errors occurred during the calculation of the analysis	Upskin Downskin Analysis
upskinangle	Read	Number	Returns the entered upskin angle threshold between the z plane and the triangle	Upskin Downskin Analysis
upskinarea	Read	Number	Returns the total area of all upskin triangles	Upskin Downskin Analysis
upskincomponentcount	Read	Number	Returns the number of upskin components	Upskin Downskin Analysis
wallthicknessanalysiswassuccessful	Read	Boolean	Returns true if no errors occurred during the calculation of the analysis	Wallthickness Analysis
wallthicknessareabelowthreshold	Read	Number	Returns the area below the wallthicknesscriticaldistance threshold	Wallthickness Analysis
wallthicknessclustercount	Read	Number	Returns the number of detected clusters	Wallthickness Analysis
wallthicknesscriticaldistance	Read	Number	Returns the entered failing threshold in mm below which the cluster's area counts towards the failed area	Wallthickness Analysis
wallthicknesscriticalsurface	Read	Number	Returns the entered percentage threshold of the surface below which the test fails	Wallthickness Analysis
wallthicknesslargestclusterarea	Read	Number	Returns the area of the largest cluster	Wallthickness Analysis

2.15.6.2 Method Overview

Name	Syntax	Description
createdefaultanalysis	partanalyser:createdefaultanalysis()	Runs a default analysis on the given part
createcenterofgravityanalysis	partanalyser: createcenterofgravityanalysis()	Runs a center of gravity analysis on the given part

createshadowareaanalysis	partanalyser:createshadowareaanalysis()	Runs a shadow area analysis on the given part
createsupportvolumeanalysis	partanalyser:createsupportvolumeanalysis (AAngle: number)	Runs a shadow area analysis on the given part AAngle: threshold used for support cluster detection
createwallthicknessanalysis	partanalyser: createwallthicknessanalysis(ADistance, AAreaPercentage: number, ACancelOnFail: boolean)	Runs a wall thickness analysis on the given part ADistance: minimal passing wall thickness AAreaPercentage: Area below minimal passing thickness still allowed to pass ACancelOnFail: cancel if test already failing without waiting for end result
createupskindownskinanalysis	partanalyser: createupskindownskinanalysis(AUpskinAngle, ADownskinAngle, AMinAreaSize: number, AFilterSmallTriangles: boolean)	Runs a upskin downskin analysis on the given part AUpskinAngle: minimal angle between triangle and z plane to be counted as upskin ADownskinAngle: minimal angle between triangle and z plane to be counted as downskin AMinAreaSize: minimal area a cluster needs to have to count towards the upskin or downskin AFilterSmallTriangles: flag to exclude very small triangles

2.15.7 Primitive List

[Desktop Automation]

This object is used for creating meshes. Created by system (list = system:createprimitivelist)

2.15.7.1 Properties

Property	Read / Write	Type	Description
count	read only	Number	Number of primitives in the list

2.15.7.2 Method Overview

Name	Syntax	Description
createprimitive	List:createprimitive(name: string)	Creates a primitive-object by identifier
createprimitivebyindex	List:createprimitivebyindex(index: number)	Creates a primitive-object by index

getname	List:getname(index: number)	Get name (identifier) of primitive at given index

2.15.8 Primitive Object

[Desktop Automation]

This object is used for creating meshes.

2.15.8.1 Properties

Property	Read / Write	Type	Description
settingcount	read only	Number	Number of settings of this primitive
[generic setting name]	Read / write	[Number, String]	Read and write any setting of the primitive by the specific name. The list of available properties depends on the primitive.

2.15.8.2 Method Overview

Name	Syntax	Description
Generatemesh	Primitive:generatemesh()	Generates and returns a mesh object.
Getsettingname	Primitive:getsettingname(index: number)	Get setting-name at given index
Getsettingvalue	Primitive:getsettingvalue(index: number)	Get value of setting at given index
Setsettingvalue	Primitive:setsettingvalue(index: number; value number|string)	Set value of setting at given index
setsettingvaluebyname	Primitive:setsettingvaluebyname(key: string; value: number|string)	Set the value of a setting with the given name

2.16 3S Script Objects

[3S]

The 3S Lua implementation is accessible as the 3S “script” function and its API only contain the functions documented in this chapter. No other Netfabb API calls are available in this implementation. Vice versa, the 3S API functionality is only available in the 3S module and nowhere else.

2.16.1 LUAStructure

Commands for the Lua Structure class

2.16.1.1 Properties

Property	Read / Write	Type	Description
cellcount	read only	number	cell.cellcount
simulateafterexecution	read / write	boolean	structure.simulateafterexecution=true;
surfacedatacount	read only	number	surface.surfacedatacount
volumedatacount	read only	number	volume.volumedatacount

2.16.1.2 Method Overview

Name	Syntax	Description
executescript	structure:executescript(name: string);	Executes script, no return value
findcell	Cell:LUABaseCell = structure:findcell(name:string);	
findvolumedata	Volumedata:LUAVolumeData = structure:findvolumedata(name:string);	
findsurfacedata	Surface: LUASurfaceData = structure:findsurfacedata(“surface”);	
getcell	Cell:LUABaseCell structure:getcell(index:number);	
getsurfacedata	Surfacedata: LUASurfaceData = structure:getsurfacedata(index:Number);	
getvolumedata	Volumedata: LUAVolumeData = structure:getvolumedata(index:Number);	

2.16.2 LUAVolume Data

Commands for the LUAVolume class

2.16.2.1 Properties

Property	Read / Write	Type	Description
fragmentcount	read only	number	volume.fragment
groupcount	read only	number	volume.groupcount
name	read only	string	volume.name=”volume”;
originx	read only	number	volume.originx
originy	read only	number	volume.originy
originz	read only	number	volume.originz
rastersizex	read only	number	volume.rastersizex
rastersizey	read only	number	volume.rastersizey
rastersizez	read only	number	volume.rastersizez
sizex	read only	number	volume.sizex
sizey	read only	number	volume.sizey
sizez	read only	number	volume.sizez
visible	read / write	boolean	volume.visible=true/false;

2.16.2.2 Method Overview

Name	Syntax	Description
addgroup	Group: LUAVolumeDataGroup = volumedata:addgroup(name:String);	
addmeshtoraster	Part: LUAVolumeDataFragment = volumedata:addmeshtoraster();	
addsurfacedata	Volume: LUAVolumeDataFragment = volumedata:addsurfacedata(surfacefragment: SurfaceFragment, [xyoffset:Number, zoffset:Number]);	
cleanup	volumedata:cleanup();	No return value
findfragment	Part: LUAVolumeDataFragment = volumedata:findfragment(name:String);	
findgroup	Group: LUAVolumeDataGroup = volumedata:findgroup(name:String);	
getfragment	Part: LUAVolumeDataFragment = volumedata:getfragment(name:String);	
getgroup	Group: LUAVolumeDataGroup = volumedata:getgroup(index:Number);	
merge	Part: LUAVolumeDataFragment = volumedata:merge(part1: LUAVolumeDataGroup| LUAVolumeDataFragment, [...]);	
reset	volumedata:reset();	No return value
resize	Part: LUAVolumeDataFragment = volumedata:resize(plusx, plusy, plusz, minusx, minusy, minusz:all Number);	

2.16.3 LUAVolumeDataFragment

Commands for the LUAVolumeDataFragment class

2.16.3.1 Properties

Property	Read / Write	Type	Description
cell	read / write	object	part.cell;
color	read / write	number	49906=part.color; part.color=49906;
name	read / write	string	“part1”=part.name; part.name=”part1”;
rastercount	read only	number	part.rastercount;

2.16.3.2 Method Overview

Name	Syntax	Description
createexpansion	part1: LUAVolumeDataFragment = part2:createexpansion(minusx:number, minuxy:number, minuxz:number, plusx:number, plusy:number, plusz:number, celloverride:boolean);	
createhull	Skin: LUAVolumeDataFragment = part:createhull(minusx:number, minuxy:number, minuxz:number, plusx:number, plusy:number, plusz:number);	

createprojection	part1: LUAVolumeDataFragment = part1:createprojection(type:number, celloverwrite:boolean);	projection(number) 1 = +x , 2 = -x , 3 = +y , 4 = -y 5 = +z ,6 = -z
divideblocks	Blocks: LUAVolumeDataFragment = part:divideblocks(blocksizex:Number, blocksizey:Number, blocksizez:Number, translationx: Number, translation: Number, translationz:Number, fillblocks:boolean);	
generatechessboard	part1: LUAVolumeDataFragment = part2:generatechessboard(blocksizex:Number, blocksizey:Number, blocksizez:Number, translationx: Number, translation: Number, translationz:Number)	
movetogroup	Ret:Boolean = volume:movetogroup(group1:LUAVolumeDataGroup);	
randomize	part1: LUAVolumeDataFragment = part2:randomize(blocksizex:Number, blocksizey:Number, blocksizez:Number, translationx: Number, translation: Number, translationz:Number, probability:Number)	
remove	part:remove();	

2.16.4 LUAVolumeDataGroup

Commands for the LUAVolumeDataGroup class.

2.16.4.1 Properties

Property	Read / Write	Type	Description
fragmentcount	read only	number	volume.fragmentcount

name	read / write	string	"volume1"=volume.name;
			volume.name="volume1";

subgroupcount	read only	number	Volume.subgroupcount

2.16.4.2 Method Overview

Name	Syntax	Description
addgroup	Group: LUAVolumeDataGroup = volumedata:addgroupname:String);	
findfragment	Fragment: LUAVolumeDataFragment = volumedata:findfragment(name:string);	
findgroup	Group: LUAVolumeDataGroup = volumedata:findgroup(name:string);	
getfragment	Fragment: LUAVolumeDataFragment = volumedata:getfragment(index:number);	
getgroup	Group: LUAVolumeDataGroup = volumedata:getgroup(number:index);	
movetogroup	Ret:Boolean = part:movetogroup(group1: LUAVolumeDataGroup);	

2.16.5 LUASurfaceData

Commands for the LUASurfaceData class.

2.16.5.1 Properties

Property	Read / Write	Type	Description
fragmentcount	read only	number	surface.fragmentcount
groupcount	read only	number	surface.groupcount
name	read only	string	"surface"=surface.name;
visible	read / write	boolean	surface.visible=true/false

2.16.5.2 Method Overview

Name	Syntax	Description
addgroup	Group: LUASurfaceDataGroup = surfacedata:addgroup(group1:String);	
addmesh	Surface: LUASurfaceDataFragment = surfacedata:addmesh();	
classifydownsides	part=surfacedata:classifydownsides(angle:Number, minsizearea:Number, samplecomponents: boolean);	
classifyupsides	part=surfacedata:classifyupsides(angle:Number, minsizearea:Number, samplecomponents: boolean);	
cleanup	surfacedata:cleanup();	
findfragment	Surface: LUASurfaceDataFragment = surfacedata:findfragment("surface1:String");	
findgroup	Group: LUASurfaceDataGroup = surfacedata:findgroup("group1":String);	
getfragment	Surface: LUASurfaceDataFragment = surfacedata:getfragment(index:Number);	
getgroup	Group: LUASurfaceDataGroup = surfacedata:getgroup(index:Number);	
reset	surfacedata:reset();	

2.16.6 LUASurfaceDataFragment

Commands for the LUASurfaceDataFragment class

2.16.6.1 Properties

Property	Read / Write	Type	Description
color	read / write	number	49906=surface.color; surface.color=49906;
name	read / write	string	"Surface1"=surface.name; surface.name="Surface1";

2.16.6.2 Method Overview

Name	Syntax	Description
movetogroup	Ret:Boolean = surface:movetogroup(group1: LUASurfaceDataGroup);	

2.16.7 LUASurfaceDataGroup

Commands for the LUASurfaceDataGroup class

2.16.7.1 Properties

Property	Read / Write	Type	Description
fragmentcount	read only	number	surface.fragmentcount
name	read / write	string	"surface1"=surface.name; surface.name="surface1";
subgroupcount	read only	number	surface.subgroupcount

2.16.7.2 Method Overview

Name	Syntax	Description
addgroup	Group: LUASurfaceDataGroup = surfacedata:addgroup("group1":string);	
findfragment	Surface: LUASurfaceDataFragment = surfacedata:findfragment("fragment":String);	
findgroup	Group: LUASurfaceDataGroup = surfacedata:findgroup("group1":String);	
getfragment	Surface: LUASurfaceDataFragment = surfacedata:getfragment(index:Number);	
getgroup	Group: LUASurfaceDataGroup = surfacedata:getgroup(index:Number);	
movetogroup	Ret:Boolean = surface:movetogroup(group1: LUASurfaceDataGroup);	

2.16.8 LUABaseCell

Commands for the LUABaseCell class

2.16.8.1 Properties

Property	Read / Write	Type	Description
name	read only	string	"cell"=cell.name;
sizex	read only	number	cell.sizex
sizey	read only	number	cell.sizey
sizez	read only	number	cell.sizez

2.16.8.2 Method Overview

None.