Scripting Manual for TurnTool2011

1. Preface

This manual is intended to describe which functions and properties that can be accessed at runtime when using TurnTool. Read this manual or the TurnToolBox manual related to your 3D application for help. You can also use the TurnTool Forum at www.turntool.com/forum for more help and discussions with other users.

2. Integrating TurnTool

The TurnTool Viewer is an ActiveX Control, and as such can be integrated with many kinds of documents. As a few examples, amongst others, can be mentioned Adobe Director projector files (exe) and websites (html).

How an ActiveX Control is inserted into a specific type of document varies from program to program and you should refer to the documentation for your particular program. The document or program in which the control is inserted is referred to as the container or client. The control is also known as the server. Inserting the control is often done by selecting it from a list of available controls. Select ”TurnTool Scene” to insert the TurnTool control. In some containers (html being one), it may be necessary to refer to the control by its Class ID which is a numeric identifier uniquely identifying the TurnTool control. This ID is ”4a8a5cc4-d0f5-4e7b-aaea-bfc70446bfda”.

Common to all containers are their ability to affect the control's behavior by setting the control’s properties and/or calling the control's methods. The TurnTool control exposes a number of properties and methods, which can be manipulated by the container, and some must be initialized for the control to become active. These are:

2.1. Properties

-backColor: This property represents the background color of the control before the scene starts rendering e.g. when it is loading. The color is formatted as colors are in html: #RRGGBB, where RR is the red intensity in hexadecimal notation, GG is green and finally BB is the blue intensity. Default value is ”#FFFFFF” (white).

-videoDevice: The videoDevice property is '0' as default and with this value the TurnTool Viewer will allocate hardware accelerated videoDevice. This property is set to -1 the TurnTool Viewer will allocate a software emulated videoDevice, which is slow to reliable.

-realtimeAntialias: The realtimeAntialias property is '1' as default and with this value the entire screen area will be rendered with 2x2 pixel realtime

antialiasing. If this property is set to '0' the screen will be rendered without realtime antialiasing.

-antialiasPasses: The antialiasPasses property is '2' as default and with this value the TurnTool Viewer will perform a 4x4 progressive antialiasing when the scene has not changed for a while. If the antialiasPasses property is set to '1' then 2x2 progressive antialiasing will enabled. Finally when set to '0' progressive antialiasing will be disabled.

-tnt_id: This property is a unique identifier of the TurnTool control. Normally defined as "TNTCtrl". If multiple TurnTool controls are used on the same HTML page, the idenfier of the first TurnTool control should be set to TNTCtrl1 and the second TurnTool control should be set to TNTCtrl2 and so on.

Methods

-doTNTCommand: This method is used by the script to communicate with the TurnTool control. (For more information, see chapter 3: "Scripting the TurnTool Control".)

2.2.Events

-TNTEvent: Advanced feature used for communicating with the active script. Its use depends on the TNT script used. (For more information, see chapter 3: "Scripting the TurnTool Control".)

2.3.Inserting TurnTool in an HTML document

Here is an example of how the control could be implemented in an html document:

```
<object id="TNTCtrl" width="800" height="600" classid="CLSID:4a8a5cc4-d0f5-
4e7b-aaea-bfc70446bfda"
codebase="http://tt11.turntool.com/TurnTool2011.exe#version=1,0,1,2"></object>
```

Note: This will create an empty TurnTool control in your HTML page.
Where possible it is recommended to link to
http://www.turntool.com/TurnTool2011.exe both for automatic download as above (the CODEBASE parameter) and as a separate link.

3. Scripting the TurnTool Control

TurnTool offers the feature of advanced scripting. Through scripting, true interactivity can be applied to your scene. Imagine that you have modeled a machine and created nice animations showing the functionality of the machine. Without scripting the animations would play automatically, either continuously or just once. With scripting, the scene can be programmed to respond to user interaction. The user clicks a button on the machine or in the web page, and the machine starts to operate. Or the user can move the object and turn it around by clicking a button or arrow key. The object can actually be seen from all angles by means of the user's control and interaction.

This advanced scripting is available through the TurnTool control's interface. It is comprised of a simple method (or function) and an equally simple event: doTNTCommand and TNTEvent. These are both briefly described in the "Integrating TurnTool" section. Here is a more in depth description:

3.1. TNTEvent

Syntax: `OnTNTEvent(string event);`
The syntax of the event handler must be adapted to the scripting language of your choice.
The event allows you to respond to certain events happening in the TurnTool scene. The string parameter event is a string representing the event that has occurred. This string may have to be dissected by the script, to find out exactly what happened. Here is a list of the possible events:

3.1.1. OnInitialize

Syntax: TNTEvent('OnInitialize(string::tntCtrlID)')
This event is generated when the TurnTool control has been initialized. Once the TurnTool control is initialized the scene is empty except for one node called "Scene Root". The argument of the OnInitialize function is a string called tntCtrlID. The tntCtrlID string contains the content of the property called 'tnt_id' of the TurnTool control that is being initialized. When using multiple TurnTool controls in the same HTML page, the 'tnt_id' property of each TurnTool control should be set to a unique value. Because the value of tnt_id property is passed to the OnInitialize function as an argument it becomes possible to initialize each TurnTool control differently.

3.1.2. OnSceneLoad

Syntax: TNTEvent('OnSceneLoad(string::fileName)')
This event is generated when scene is fully loaded. Once a TNT file is downloaded the TurnTool Viewer begins to load the information from the TNT file into memory. When the scene is ready this event will be triggered. Loading a scene into memory is done by calling this command.
'Viewer().LoadScene(+pathToTNTfile+)'

The argument of the OnSceneLoad function is a string called fileName. The fileName string contains the name of the file that has been loaded as a scene.

3.1.3. OnMouseKeyPress

Syntax: TNTEvent('OnMouseKeyPress(integer::keyIndex)')
This event is generated when a mouse key is pressed.
The argument is a integer called keyIndex. The keyIndex integer can be used to identify which mouse key that has been pressed. The keyIndex will be 1 if the left mouse button is pressed and it will be 2 if the right mouse button is pressed.

3.1.4. OnMouseKeyRelease

Syntax: TNTEvent('OnMouseKeyRelease(integer::keyIndex)')
This event is generated when a mouse key is released.
The argument is a integer called keyIndex. The keyIndex integer can be used to identify which mouse key that has been released. The keyIndex will be 1 if the left mouse button is released and it will be 2 if the right mouse button is released.

3.1.5. OnKeyPress

Syntax: TNTEvent('OnKeyPress(integer::keyIndex)')
This event is generated when a keyboard key is pressed.
The argument is a integer called keyIndex. The keyIndex integer can be used to identify which keyboard key that has been pressed.

3.1.6.OnKeyRelease

Syntax: TNTEvent('OnKeyRelease(integer::keyIndex)')
This event is generated when a keyboard key is released.
The argument is a integer called keyIndex. The keyIndex integer can be used to identify which keyboard key that has been released.

3.1.7. OnMouseEnter

Syntax: TNTEvent('OnMouseEnter(string::nodeName, integer::nodeIndex)')
This event is generated when the mouse cursor is over an object on the screen. In other words when the mouse cursor enters the shape of the object on the screen this event will be triggered. The first argument is a string called nodeName. The nodeName string contains the name of the object that the mouse cursor is moved over. The second argument is an integer called nodeIndex. The nodeIndex integer contains the index of the node in the scene. The nodeIndex integer is useful when there are two or more nodes with the same name in the scene.

3.1.8. OnMouseExit

Syntax: TNTEvent('OnMouseExit(string::nodeName, integer::nodeIndex)')
This event is only generated after a OnMouseEnter event and when the mouse cursor is moved away from an object on the screen.

The first argument is a string called nodeName. The nodeName string contains the name of the object that the mouse cursor is moved over. The second argument is an integer called nodeIndex. The nodeIndex integer contains the index of the node in the scene. The nodeIndex integer is useful when there are two or more nodes with the same name in the scene.

3.1.9. OnNodeDistanceEventEnter

Syntax: TNTEvent('OnNodeDistanceEventEnter(integer::eventIndex)')
This event is generated when two nodes gets within a certain distance between each other. To enable this event for any two nodes a NodeDistEvent object must first be created and then the two nodes must be assigned to the event. A radius must be defined in the NodeDistEvent object and the NodeDistEvent must be enabled. When this is done the NodeDistEvent will for each frame calculate the distance between the two nodes. If distance value is lower than the radius value the event will be triggered. The argument is an integer called eventIndex. The eventIndex integer contains the index of the NodeDistEvent object that has been triggered.

3.1.10. OnNodeDistanceEventExit

Syntax: TNTEvent('OnNodeDistanceEventExit(integer::eventIndex)')
This event is only generated after a OnNodeDistanceEventEnter event and when the distance between the two nodes are greater than the radius value define in the NodeDistEvent object. The argument is an integer called eventIndex. The eventIndex integer contains the index of the NodeDistEvent object that has been triggered.

3.1.11. OnExternalResourceMerge

Syntax: TNTEvent('OnExternalResourceMerge(string::uniqueNodeName, integer::nodeIndex, string::fileName)')
This event is generated when an external resource is merged into the scene. When the external resource (a TNT file) has been downloaded then the content of the TNT file is then added to the existing scene. The argument called uniqueNodeName is the a uniqueNodeName of the node that was created when the external resource was merged into the scene. The node with the unique name is the root node from the TNT file. The argument nodeIndex is an integer that holds the nodeIndex of unique node. The argument fileName contains the name of the file that has been merged.

3.1.12. OnLODMerge

Syntax: TNTEvent('OnLODMerge(string::uniqueNodeName, integer::nodeIndex, string::fileName)')
This event is generated when an Level Of Detail object is merged into the scene. As the distance from the active to the LOD container object changes the new level of detail replaces the previous level of detail.
The argument called uniqueNodeName is the a uniqueNodeName of the node that was created when the LOD was merged into the scene. The node with the

unique name is the root node from the TNT file. The argument nodeIndex is an integer that holds the nodeIndex of the unique node.

3.1.13. OnFileDownloadProgress

Syntax:
TNTEvent('OnFileDownloadProgress(string::fileName,integer::downloadFactor)')
This event is generated while a download is in progress when a new piece of the file is downloaded. The string fileName contains the name of the file that is being downloaded. The integer called downloadFactor is a value from 0 to 10000. A downloadFactor of 10000 means that the file is fully downloaded.

3.1.14. OnFileDownloadComplete

Syntax: TNTEvent('OnFileDownloadComplete(string::fileName)')
The event is generated when a file is fully downloaded. The string fileName contains the name of the file that has been downloaded.

3.1.15. OnScriptError

Syntax: TNTEvent('OnScriptError(string::text)')
This event is generated if the command string that is passed as argument to doTNTCommand method contain one or more errors. The argument text is a string that contains information regarding the nature of the error. This event should only be implemented when you are debugging your application using TurnTool.

3.1.16. OnError

Syntax: TNTEvent('OnError(string::fileName,integer::status)')
This event is generated if an error occurs in TurnTool. The integer called 'status' holds an error code that will indicate which kind of error that has occurred.
The errorcode -1 indicates that the download of a file failed. The string called fileName contains the name of the file that failed.

3.1.17. OnPlayerCreate

Syntax: TNTEvent('OnPlayerCreate(string::nodeName, string::playerName)')
This event is generated when a player is created. The concept of a player is introduced when a person using another computer is visible in your scene.
To get this behavior it is required that multi-player license component must be available. For the multi-player feature to work it is necessary that a server is available to exchange data between client computers. The argument nodeName is a string that contains the name of the node which position and rotation is being controlled from another computer. The argument playerName is a string that contains the a unique name of the player that has been created.

3.1.18. OnPlayerDestroy

Syntax: TNTEvent('OnPlayerDestroy(string::nodeName, string::playerName)')
This event is only generated after a OnPlayerCreate event. The argument

nodeName is a string that contains the name of the node which position and rotation is being controlled from another computer. The argument playerName is a string that contains the a unique name of the player that has been created.

3.2.DoTNTCommand

Syntax: `string doTNTCommand(string command);`
This function executes a TurnTool command define as string 'command'.
The format of the string command is defined below.
`'rootselector.selector.method(arg1,arg2,...,argn)'.`
The rootselector can be omitted if the a scene selector is used. See the full list of scene selectors in section 3.4 below.
Method is the method of the object in question being called arg1, arg2 and so on are the parameters of method, the number and types of these depends on the method in question.
The parameters in cmd can have the following types:
string: Sequence of characters enclosed by double quotes (””) or single quotes (').
integer: Signed whole number.
float: Signed number with a floating decimal point.
boolean: Flag, can be true or false.
Some methods, such as the many ”Get” functions, return a value when called. This is always a string. The string should be interpreted according to the function being called and can contain values of any of the aforementioned types.

3.3. TurnTool Command List

The rest of this manual contain the full list of selectors, set, get and execute methods, which can be combined into a command. The sections below contain the full list of methods which are available in the TurnTool scripting language.

3.4. Scene Selectors

3.4.1.1. Node

Syntax: Node(integer|string)

With the node selector method you can select one or more nodes in the scene with ether an index or a string. We recommend using a unique search string as indices might change when external resources are added to the scene.

Example1: `doTNTCommand('Node("RootNode*")');`

Example1 creates a selection containing all nodes which have a name that starts with "RootNode".

Example2: `doTNTCommand('Node(4).GetName()');`

Example2 creates a selection containing the node with index 4 in the scene and returns the name of the first entry in the selection.

3.4.1.2. Viewport

Syntax: Viewport(integer|search)

With the viewport selector method you can select one or more viewports in the scene with ether an index or a string. We recommend using a unique search string as indices might change when external resources are added to the scene.

Example1: `doTNTCommand('Viewport("Viewport1").GetName()');`

Example1 creates a selection containing the viewport with the name "Viewport1" and returns the name of the first entry in the selection.

Example2: `doTNTCommand('Viewport(1)');`

Example2 creates a selection containing the viewport at index 1.

3.4.1.3. Camera

Syntax: Camera(integer|string)

With the camera selector method you can select one or more cameras in the scene with ether an index or a string. We recommend using a unique search string as indices might change when external resources are added to the scene.

Example: `doTNTCommand('Camera("Camera01").GetName()');`

This example creates a selection containing the camera called "Camera01" and returns the name of the first entry in the selection.

3.4.1.4. Light

Syntax: Light(integer|string)

With the light selector method you can select one or more lights in the scene with ether an index or a string. We recommend using a unique search string as indices might change when external resources are added to the scene.

Example: `doTNTCommand('Light("Omni01").GetName()');`

This example creates a selection containing the light source called "Omni01" and returns the name of the first entry in the selection.

3.4.1.5. Mesh

Syntax: Mesh(integer|string)

With this mesh selector method you can select one or more meshes in the scene with ether an index or a string. We recommend using a unique search string as indices might change when external resources are added to the scene.
Example: `doTNTCommand('Mesh("Torus01").GetName()');`
This example creates a selection containing the mesh called "Torus01" and returns the name of the first entry in the selection.

3.4.1.6. LOD

Syntax: LOD(integer|string)

With this LOD selector method you can select one or more LOD objects in the scene with either an index or a string. We recommend using a unique search string as indices might change when external resources are added to the scene.
Example: `doTNTCommand('LOD("MyLOD01").GetName()');`
This example creates a selection containing the LOD called "MyLOD01" and returns the name of the first entry in the selection.

3.4.1.7. Bitmap

Syntax: Bitmap(integer|string)

With this Bitmap selector method you can select one or more Bitmaps in the scene with either an index or a string. We recommend using a unique search string as indices might change when external resources are added to the scene.
Example: `doTNTCommand('Bitmap("MyBitmap")');`
This example creates a selection containing the bitmap called "MyBitmap".

3.4.1.8. NodeDistEvent

Syntax: NodeDistEvent(integer)

With this NodeDistEvent selector method you can select a NodeDistEvent objects in the scene using an index. The NodeDistEvent only purpose see to generate the OnNodeDistanceEventEnter and OnNodeDistanceEventExit events. See these two events for more details.
Example: `doTNTCommand('NodeDistEvent(2).GetIndex()');`
This example creates a selection containing the NodeDistEvent object at index 2 in the scene and return the index of this object.

3.4.1.9. PlanTool

Syntax: PlanTool(integer|string)

With this PlanTool selector method you can select one or more PlanTool objects in the scene with either an index or a string. We recommend using a unique search string as indices might change when external resources are added to the scene.
Example: `doTNTCommand('PlanTool("MyHouse1")');`

This example creates a selection containing the PlanTool object called "MyHouse1".

3.4.1.10. Player

Syntax: Player(integer|string)

With this Player selector method you can select one or more Player objects in the scene with either an index or a string. We recommend using a unique search string as indices might change when external resources are added to the scene.

Example: `doTNTCommand('Player("Player01")');`

This example creates a selection containing the Player object called "Player01".

3.4.1.11. Animation

Syntax: Animation(integer|string)

With this Animation selector method you can select one or more Animation objects in the scene with either an index or a string. We recommend using a unique search string as indices might change when external resources are added to the scene.

Example: `doTNTCommand('Animation("Global")');`

This example creates a selection containing the Animation object called "Global".

3.4.1.12. CurrentCamera

Syntax: CurrentCamera()

The CurrentCamera selector will select the camera defined as current in the scene. The current camera is the active camera in the scene that is used in the process of rendering the image for the screen.

Example: `doTNTCommand('CurrentCamera()');`

This example creates a selection containing the current camera.

3.4.1.13. CurrentViewport

Syntax: CurrentViewport()

The CurrentViewport selector will select the viewport defined as current in the scene. The current viewport is the active viewport in the scene that is used in the process of rendering the image for the screen.

Example: `doTNTCommand('CurrentViewport()');`

This example creates a selection containing the current viewport.

3.4.1.14. CurrentPhysics

Syntax: CurrentPhysics()

The CurrentPhysics selector will select the physics defined as current in the scene. The current physics is the active physics sphere in the scene that the user can control using the mouse and keyboard. Physics calculation are carried out each frame using only the physics sphere that is set as current.

Example: `doTNTCommand('CurrentPhysics()');`

This example creates a selection containing the current physics object.

3.4.1.15. CameraTarget

Syntax: CameraTarget()

If there is a camera in the object selection then the CameraTarget selector will replace the selection to contain the target node of this camera.
Example:

```
doTNTCommand('Camera("Camera01").CameraTarget().GetLocalPositionX()');
```

This example will return the X position of the target node of the "Camera01" camera.

3.4.1.16. SceneRoot

Syntax: SceneRoot()

The SceneRoot selector will create a selection and add the root node of the scene to the selection. The root node of the scene is called "Scene Root". The scene root node is a empty node which all other nodes are linked to either directly or indirectly. Other nodes have a parent or a grand-parent node that is linked to scene root.
The node of the scene basically creates unbalanced branching structure or tree that contain all nodes in the scene.
Example: `doTNTCommand('SceneRoot()');`
This example creates a selection containing the SceneRoot Node.

3.4.1.17. Destroy

Syntax: Destroy()

The destroy method will destroy all objects in the selection. If the object being destroy has dependencies to other objects these objects will be destroyed as well.
Example: `doTNTCommand('Node("DummyNode").Destroy()');`
This example will destroy the node called "DummyNode".

3.4.1.18. SelectByRay

Syntax:
SelectByRay(nameFilter:string,posx:float,posy:float,posz:float,dirx:float,diry:float,dirz:float)

The SelectByRay method allow you add an object to the selection that intersects with a ray that you define by specifying a start position and a direction vector. The arguments posx, posy, posz is used as the x,y,z coordinates of the start position vector. The arguments dirx, diry, dirz is used as the x,y,z coordinates direction vector. The string called nameFilter can be used as a name filter to ignore all objects that do not match with the search string.
Example: `doTNTCommand('SelectByRay("*Wall*",0.0,0.0,0.0,1.0,0.0,0.0');`
This example returns the first object that is hit by the defined ray and have a name that contain the word "Wall".

3.4.1.19. SelectAnimation

Syntax: SelectAnimation()

If there is a node in the object selection then the SelectAnimatin selector will replace the selection to contain the Animation object assigned to this node.

Example: `doTNTCommand('Node("Teapot01").SelectAnimation()');`

This example selects the animation object that is assigned to the node called "Teapot01".

3.5.Narrowing

3.5.1.1. Index

Syntax: Index(index:integer)

The Index selector method is a narrowing selector which can be used for limiting a selection to one object. The index selector will remove all other objects from the selection except the object at the defined index.

Example: `doTNTCommand('Node("ExternNode*").Index(2)');`

In the first part of this example all nodes that have a name that starts with "ExternNode" is added to the selection then the Index selector method limits the selection only to contain the object at index 2.

3.5.1.2. Range

Syntax: Range(start:integer, end:integer)

The range selector method is a narrowing selector which can be used to select objects in the selection with a certain range. Objects that are outside the defined range will be removed from the selection.

Example: `doTNTCommand('Node("*").Range(10,20)');`

This example selects all nodes in the scene and then limits the selection the nodes with in the range of 10 to 20.

3.5.1.3. Name

Syntax: Name(name:string)

The Name selector method is a narrowing selector which can be used to select objects which have a name that matches a certain search string.

Example: `doTNTCommand('Node("ExternNode4").ChildNodes().Name("Door*")');`

This example will first select ExternNode4 in the scene and then all it's child nodes and finally the selection will be limited to contain only objects that have a name that starts with "Door".

3.5.1.4. Type

Syntax: Type(type:string)

The Type selector method is a narrowing selector which can be used to limit the selection to objects of a certain type. Objects that are a different type then the one specified will be removed from the selection.

Example: `doTNTCommand('Node("*").Type("Light")');`

This example selects all nodes in the scene and then removes all nodes that do not have the type "Light".

3.5.1.5. Cast
Syntax: Cast(type:string)
The Cast selector method can be used to convert an object in the selector to another type of object if possible. A mesh can be converted into a node and vice versa.
Example: `doTNTCommand('Mesh("Teapot01").Cast("Node").Cast("Mesh")');`
This example selects the mesh called "Teapot01" and casts it to a node and then casts it back to a mesh.

3.6.Creational

3.6.1.1. CreateNode
Syntax: CreateNode(name:string)
The CreateNode method can be used to create a Node with have the name that you specify argument. The node will automatically have the Scene Root node as parent.
Example: `doTNTCommand('CreateNode("MyDummyNode").GetName()');`
This example creates a node called "MyDummyNode".

3.6.1.2. CreateViewport
Syntax: CreateViewport(name:string)
The CreateViewport method can be used to create a Viewport with have the name that you specify as argument.
Example: `doTNTCommand('CreateViewport("Viewport2")');`
This example creates a viewport called "Viewport2".

3.6.1.3. CreateCamera
Syntax: CreateCamera(name:string)
The CreateCamera method can be used to create a Camera with have the name that you specify as argument.
Example: `doTNTCommand('CreateCamera("Camera01")');`
This example creates a Camera called "Camera01".

3.6.1.4. CreateLight
Syntax: CreateLight(name:string)
The CreateLight method can be used to create a Light with have the name that you specify as argument.
Example: `doTNTCommand('CreateLight("Omni01")');`
This example creates a Light called "Omni01".

3.6.1.5. CreateMesh
Syntax: CreateMesh(name:string)
The CreateMesh method can be used to create a Mesh with have the name that you specify as argument.
Example: `doTNTCommand('CreateMesh("MyMesh01")');`
This example creates a Mesh called "MyMesh01".

3.6.1.6. CreateLOD
Syntax: CreateLOD(name:string)
The CreateLOD method can be used to create a LOD with have the name that you

specify as argument.
Example: `doTNTCommand('CreateLOD("MyLOD")');`
This example creates a LOD object called "MyLOD".

3.6.1.7. CreatePlanTool
Syntax: CreatePlanTool(name:string)
The CreatePlanTool method can be used to create a PlanTool object with have the name that you specify as argument.
Example: `doTNTCommand('CreatePlanTool("MyHouse1")');`
This example creates a PlanTool object called "MyHouse1".

3.6.1.8. CreatePlayer
Syntax: CreatePlayer(name:string)
The CreatePlayer method can be used to create a Player object with have the name that you specify as argument.
Example: `doTNTCommand('CreatePlayer("Player01")');`
This example creates a Player object called "Player01".

3.6.1.9. CreateNodeDistEvent
Syntax: CreateNodeDistEvent()
The CreateNodeDistEvent method can be used to create a NodeDistEvent object.
Example: `doTNTCommand('CreateNodeDistEvent().GetIndex()');`
This example creates a NodeDistEvent and return the unique index of this object.

3.6.1.10. CreateAnimation
Syntax: CreateAnimation(name:string)
The CreateAnimation method can be used to create an Animation object with have the name that you specify as argument.
Example: `doTNTCommand('CreateAnimation("Global")');`
This example creates an Animation object called "Global".

3.7.Filtering Selectors

3.7.1.1. NodeFrame
Syntax: NodeFrame(index:integer)
If the selection already contains one or more Nodes, then the NodeFrame selector method will select index argument number of NodeFrames from each Node. If the index argument is omitted, then all NodeFrames of the Node(s) will be added to the selection.
Example: `doTNTCommand('Node("Torus01").NodeFrame()');`
This example will select the node called "Torus01" and then all of it's node frames.

3.7.1.2. ViewportFrame
Syntax: ViewportFrame(index:integer)
If the selection already contains one or more Viewports, then the ViewportFrame selector method will select index argument number of ViewportFrames from each

Viewport. If the index argument is omitted, then all ViewportFrames of the Viewport(s) will be added to the selection.
Example: `doTNTCommand('CurrentViewport().ViewportFrames()');`
This example will select all the viewport frames of the current Viewport.

3.7.1.3. CameraFrame
Syntax: CameraFrame(index:integer)
If the selection already contains one or more Cameras, then the CameraFrame selector method will select index argument number of CameraFrames from each Camera. If the index argument is omitted, then all CameraFrames of the Camera(s) will be added to the selection.
Example: `doTNTCommand('CurrentCamera().CameraFrame()');`
This example will select all the camera frames of the current Camera.

3.7.1.4. LightFrame
Syntax: LightFrame(index:integer)
If the selection already contains one or more Lights, then the LightFrame selector method will select index argument numberof LightFrames from each Light. If the index argument is omitted, then all LightFrames of the Light(s) will be added to the selection.
Example: `doTNTCommand('Light("Omni01").LightFrame()');`
This example will select all the light frames of the light source called "Omni01".

3.7.1.5. MeshFrame
Syntax: MeshFrame(index:integer)
If the selection already contains one or more meshes, then the MeshFrame selector method will select index argument number of MeshFrames from each Mesh. If the index argument is omitted, then all MeshFrames of the Mesh(es) will be added to the selection.
Example: `doTNTCommand('Mesh("Torus01").MeshFrame()');`
This example will select all mesh frames of mesh called "Torus01".

3.7.1.6. FaceSegment
Syntax: FaceSegment(index:integer)
If the selection already contains one or more MeshFrames, then the FaceSegment selector method method will select index argument number of FaceSegments from each MeshFrame. If the index argument is omitted, then all FaceSegments of the MeshFrame(s) will be added to the selection.
Example: `doTNTCommand('Mesh("Floor*").MeshFrame().FaceSegment()');`
This example will select all meshes that have a name that starts with "Floor". Then all meshframes are selected and then all face-segments is selected.

3.7.1.7. TextureFragment
Syntax: TextureFragment(index:integer)
If the selection already contains one or more FaceSegments, then the TextureFragment selector method will select index argument number of TextureFragments from each FaceSegments. If the index argument is omitted,

then all TextureFragments of the FaceSegment(s) will be added to the selection.
Example:
```
doTNTCommand('Mesh("MyObj").MeshFrame().FaceSegment().TextureFragment(0)');
```
This example will select the Mesh called "MyObj". All MeshFrames of the Mesh are then selected and then all FaceSegments of these MeshFrames are selected and finally all the TextureFragments at index 0 is selected.

3.7.1.8. Texture
Syntax: Texture(index:integer)
If the selection already contains one or more TextureFragments, then the Texture selector method will select index argument number of Textures from each TextureFragments. If the index argument is omitted, then all Textures of the TextureFragment(s) will be added to the selection.
Example:
```
doTNTCommand('Mesh("MyObj").MeshFrame().FaceSegment().TextureFragment().Textur
e()');
```
This example will select the Mesh called "MyObj". All MeshFrames of the Mesh are then selected and then all FaceSegments of these MeshFrames are selected and then, all the TextureFragments of these FaceSegments are selected. Finally all Textures of the TextureFragments are then selected.

3.7.1.9. LODRangeInfo
Syntax: LODRangeInfo(index:integer)
If the selection already contains one or more LOD object, then the LODRangeInfo selector method will select index argument number of LODRangeInfo from eachj LODs. If the index argument is omitted, then all LODRangeInfo objects of the LOD(s) will be added to the selection.
Example: `doTNTCommand('LOD("MyLOD").LODRangeInfo(0)');`
This example will select the LOD object called "MyLOD" and then the LODRangeInfo at index 0 of the LOD object.

3.8.Selection Storage

3.8.1.1. Load
Syntax: Load(storeID:string)
The load method can be used to load a selection that you have previously stored under a certain name.
Example: `doTNTCommand('Load("MySelection1")');`
The example loads a selection that has previously been stored with the name "MySelection1".

3.8.1.2. Store
Syntax: Store(storeID:string)
The Store method can be used to store any selection that you have made.
Example:
```
doTNTCommand('SceneRoot().ChildNodes(5).Name("Wall*").Store("MySelection1")');
```

The example stores a selection, which have been generated by a series of selectors, under the name "MySelection1".

3.8.1.3. ChildNodes

Syntax: ChildNodes(childDepth:integer)

If the selection already contains one or more Nodes, then the ChildNodes selector method can be used to select the nodes that are linked to each of these nodes. A node that is linked to another node is called a child and a node that have one or more child nodes is called a parent node. The childDepth integer speficies the depth of the selection from the parent node.
A childDepth value of zero will only select the nodes in the first level below the parent node. A childDepth values above zero indicates how many levels of nodes below the parent node to include. A childDepth value of -1 will select the entire node hierachy below the parent node(s).
Example: `doTNTCommand('Node("MyDummyNode").ChildNodes(-1)');`
This example will select all childnodes below the node called "MyDummyNode".

3.8.1.4. ParentNode

Syntax: ParentNode(parentDepth:integer)

If the selection already contains a node, then the ParentNode selector method can be used to select the parent node of the node in the selection. If parentDepth integer argument is 1 then the parent node will be selected. If parentDepth is 2 then the grand-parent node will be selected and so on.
Example: `doTNTCommand('Node("Window15").ParentNode(2)');`
The example will select the grand-parent node of node "Window15".

3.9.General

3.9.1.1. SetName

Syntax: SetName(name:string)

The SetName method will set the name of the object(s) in the selection to the name argument specifyed.
Example: `doTNTCommand('Node("Node44").SetName("Node23")');`
This example will select the node called "Node44" and rename it to "Node23".

3.9.1.2. GetName

Syntax: GetName()

The GetName method will return the name of the first object in the selection.
Example: `doTNTCommand('Light(0).GetName()');`
This example will select the light at index 0 and return the name of it.

3.9.1.3. GetIndex

Syntax: GetIndex()

The GetIndex method can be used to return the index of any object in the selection. E.g if there are 5 light sources in the scene then each of them will have an index ranging from 0 to 4.
Example: `doTNTCommand('SceneRoot().GetIndex()');`

This example will return the index of the scene root node.

3.9.1.4. GetScaledColor

Syntax: GetScaledColor(color:color, scalar:float)

The GetScaledColor method will return the color that has been scaled. The color argument must contain the color that you wish to scale and the scalar floating point argument must contain the scale factor. The reg, green, blue components of the color will each be scaled.

Example:

```
doTNTCommand('Mesh("Wall4").SetMaterialDiffuseColor(GetScaledColor(#73AE92,0.4
)');
```

This example will scale the color #73AE92 with factor 0.4 and return the resulting color.

3.10.Scene

3.10.1.1. GetNodeCount

Syntax: GetNodeCount()

The GetNodeCount method will return the total number of nodes in the scene.

Example: `doTNTCommand('GetNodeCount()');`

This example will return the total number of nodes in the scene.

3.10.1.2. GetViewportCount

Syntax: GetViewportCount()

The GetViewportCount method will return the total number of viewports in the scene.

Example: `doTNTCommand('GetViewportCount()');`

This example will return the total number of viewports in the scene.

3.10.1.3. GetCameraCount

Syntax: GetCameraCount()

The GetCameraCount method will return the total number of cameras in the scene.

Example: `doTNTCommand('GetCameraCount()');`

This example will return the total number of cameras in the scene.

3.10.1.4. GetLightCount

Syntax: GetLightCount()

The GetLightCount method will return the total number of lights in the scene.

Example: `doTNTCommand('GetLightCount()');`

This example will return the total number of lights in the scene.

3.10.1.5. GetMeshCount

Syntax: GetMeshCount()

The GetMeshCount method will return the total number of meshes in the scene.

Example: `doTNTCommand('GetMeshCount()');`

This example will return the total number of meshes in the scene.

3.10.1.6. GetLODCount

Syntax: GetLODCount()

The GetLODCount method will return the total number of LOD objects in the scene.
Example: `doTNTCommand('GetLODCount()');`
This example will return the total number of LOD objects in the scene.

3.10.1.7. GetPlanToolCount
Syntax: GetPlanToolCount()
The GetPlanToolCount method will return the total number of PlanTool objects in the scene.
Example: `doTNTCommand('GetPlanToolCount()');`
This example will return the total number of PlanTool objects in the scene.

3.10.1.8. GetBitmapCount
Syntax: GetBitmapCount()
The GetBitmapCount method will return the total number of Bitmaps in the scene.
Example: `doTNTCommand('GetBitmapCount()');`
This example will return the total number of Bitmaps in the scene.

3.10.1.9. GetNodeDistEventCount
Syntax: GetNodeDistEventCount()
The GetNodeDistEventCount will return the total number of NodeDistEvents in the scene.
Example: `doTNTCommand('GetNodeDistEventCount()');`
This example will return the total number of NodeDistEvents in the scene.

3.10.1.10. GetAnimationCount
Syntax: GetAnimationCount()
The GetAnimationCount method will return the total number of Animations in the scene.
Example: `doTNTCommand('GetAnimationCount()');`
This example will return the total number of Animations in the scene.

3.10.1.11. GetSceneCount
Syntax: GetSceneCount(type:string, name:string)
The SceneCount method can be used to count the number of objects of a defined type that have a certain name.The type string argument is the type of object that you wish to check and the name string argument is name string that compared to the name of the objects.
Example: `doTNTCommand('GetSceneCount("Animation","Global"');`
This will return the number of Animations in the scene that have the name "Global".

3.10.1.12. GetRealtimeAntialias
Syntax: GetRealtimeAntialias()
The GetRealtimeAntialias method will return current state of realtime antialiasing.
See SetRealtimeAntialias for more details.
Example: `doTNTCommand('GetRealtimeAntialias()');`

This example will return the current state of realtime antialasing.

3.10.1.13. SetRealtimeAntialias
Syntax: SetRealtimeAntialias(rtaa:integer)
The SetRealtimeAntialias method can be used to enable or disable realtime antialias on the rendered screen image. The rtaa value is '1' as default and with this value the entire screen area will be rendered with 2x2 pixel realtime antialiasing. If the rtaa value is set to '0' then the screen will be rendered without realtime antialiasing.
Example: `doTNTCommand('SetRealtimeAntialias(1)');`
This example will enable Realtime Antialias on the rendered screen.

3.10.1.14. GetAntialiasPasses
Syntax: GetAntialiasPasses()
The GetAntialiasPasses method will return the number of antialiasing passes that are set for the scene. As default the AntialiasPasses is set to '2'
See SetAntialiasPasses for more details.
Example: `doTNTCommand('GetAntialiasPasses()');`
This example will return the number of AntialiasPasses.

3.10.1.15. SetAntialiasPasses
Syntax: SetAntialiasPasses(passes:integer)
The AntialiasPasses method can be used to control how many passes of progressive antialiasing that should be performed. When set to '2' the TurnTool Viewer will perform a 4x4 progressive antialiasing after the scene has been unchanged for a while. If set to '1' 2x2 progressive antialiasing will performed and finally when set to '0' progressive antialiasing will not be performed.
Example: `doTNTCommand('SetAntialiasPasses(2)');`
This example will set the number of AntialiasPasses to 2.

3.10.1.16. GetTimeBeforeIdle
Syntax: GetTimeBeforeIdle()
The GetTimeBeforeIdle method will return the value of the TimeBeforeIdle integer in the scene. See SetTimeBeforeIdle for more details.
Example: `doTNTCommand('GetTimeBeforeIdle()');`
This example returns the TimeBeforeIdle value.

3.10.1.17. SetTimeBeforeIdle
Syntax: SetTimeBeforeIdle(miliseconds:integer)
The SetTimeBeforeIdle method can be used to set the value of the TimeBeforeIdle integer in the scene. The TimeBeforeIdle value is the number of milliseconds to pass since the last activity in the scene, before antialiasing begins and idle mode is entered.
Example: `doTNTCommand('SetTimeBeforeIdle(1000)');`
This example sets the SetTimeBeforeIdle value to 1000 miliseconds.

3.11.Node

3.11.1.1. GetVisible
Syntax: GetVisible()

The GetVisible method will return the Visible value of the first node in the selection. See the SetVisible method for more details.
Example: `doTNTCommand('Node("Sphere01").GetVisible()');`
This example returns the Visible value of the Node called "Sphere01".

3.11.1.2. SetVisible
Syntax: SetVisible(integer)
The SetVisible method can be used to change the Visible value of one or more nodes in the selection. A mesh attached to a node, which has a Visible value of 0 will not be rendered on the screen. The default value of Visible is 1, which means that any Mesh using such a Node will be rendered on the screen.
Example: `doTNTCommand('Node("Torus*").SetVisible(0)');`
This example selects all nodes which have a name that starts with "Torus" and sets their Visible value to 0.

3.11.1.3. GetEnabled
Syntax: GetEnabled()
The GetEnabled method will return the Enabled value of the first node in the selection. See the SetEnabled method for more details.
Example: `doTNTCommand('Node("PhysicsSphere").GetEnabled()');`
This example returns the Enabled value of the Node called "PhysicsSphere".

3.11.1.4. SetEnabled
Syntax: SetEnabled(integer)
The SetEnabled method can be used to change the Enabled value of one or more nodes in the selection. The defailt value of the Enabled value is 1. If the Enabled value is set to 0, then it will be like the object using the Node, do not exist in the scene. Here are a few examples:
A Mesh using a Node with a Enabled value of 0 will not be rendered.
A Light using a Node with a Enabled value of 0 will not illuminate.
A Physics Sphere using a Node with a Enabled value of 0 will not calculate physics.
Example: `doTNTCommand('Node("PhysicsSphere").SetEnabled(0)');`
This example will set the Enabled value to '0' of the node called "PhysicsSphere".

3.11.1.5. GetLocked
Syntax: GetLocked()
The GetLocked method will return the Locked value of the first node in the selection. The default Locked value is 0. If the Locked value is set to 1 then the node will not be affected by any animation, but instead it will be locked at the position on the time that it had at the time when the lock was enabled.
Example: `doTNTCommand('Node("Teapot01").GetLocked()');`
This example will return the Locked value of the Node called "Teapot01".

3.11.1.6. SetLocked
Syntax: SetLocked(integer)
The SetLocked method change the Locked value of one or more nodes in the selection.

The default Locked value is 0. If the Locked value is set to 1, then the node will not be affected by animation, but instead it will be locked at the position on the time-line that it had at the time of locking.
Example: `doTNTCommand('Node("Teapot01").SetLocked(1)');`
This example will lock the Node called "Teapot01".

3.11.1.7. SetParent
Syntax: SetParent(name:string)
The SetParent method can be used to change the node that the node in the selection is parent of. The name string is the name of the parent. To avoid misunderstandings it is recommend that the name of the parent is unique.
Example: `doTNTCommand('Node("Window154").SetParent("House5")');`
This example will set the parent of "Window154" to "House5".

3.11.1.8. GetNodeFrameCount
Syntax: GetNodeFrameCount()
The GetNodeFrameCount method will return the number of NodeFrame of a Node in the selection.
Example: `doTNTCommand('Node("Sphere07").GetNodeFrameCount()');`
This example will return the number of NodeFrames of the Node called "Sphere07".

3.11.1.9. AppendNodeFrames
Syntax: AppendNodeFrames(count:integer)
The AppendNodeFrame method can be used to append NodeFrame to a Node in the selection. The count argument is used to define how many frames that you wish to add.
Example: `doTNTCommand('Node("Sphere07").AppendNodeFrames(100)');`
This example will append 100 NodeFrames to the node called "Sphere07".

3.11.1.10. RemoveNodeFrames
Syntax: RemoveNodeFrames(index:integer, count:integer)
The RemoveNodeFrames method can be used to remove NodeFrames of a Node in the selection. The index argument is the start frame of the removal and the count argument is how many frames that you wish to remove.
Example: `doTNTCommand('Node(“Sphere07”).RemoveNodeFrames(25,50)');`
This example will remove the node frames from 25 to 75 of the node called “Sphere07”.

3.11.1.11. LinkToNodeDistEvent
Syntax: LinkToNodeDistEvent(eventIndex:integer,nodeIndex:integer)
The LinkToNodeDistEvent method can be used to link a node in the selection to a NodeDistEvent. The eventIndex argument is the index of the NodeDistEvent that you wish to assign the node in the selection to. The nodeIndex argument is either 0 or 1. Before the NodeDistEvent object can generate events a node must be assigned to both index 0 and index 1.
Example: `doTNTCommand('CurrentCamera().LinkToNodeDistEvent(0,1)');`

This example links the CurrentCamera node to NodeDistEvent 0 as node index 1.

3.11.1.12. CreateBoundingBox

Syntax: CreateBoundingBox(ignoreHidden:integer)

The CreateBoundingBox method can be used to calculate an Axis Aligned Bounding Box of the entire node hierarchy from the node in the selection. The ignoreHidden argument will if set to 1 ignore all hidden objects in the node hierarchy.

Example: `doTNTCommand('Node("House17").CreateBoundingBox(1)');`

This example will create an axis aligned bounding box for the node called "House17".

3.11.1.13. AddToSelection

Syntax: AddToSelection()

The AddToSelection method will add one or more nodes in the selection to Selection class. See the selection class for more details.

Example: `doTNTCommand('Node("Torus01").AddToSelection()');`

This example will add the node called "Torus01" to the Selection class.

3.11.1.14. RemoveFromSelection

Syntax: RemoveFromSelection()

The RemoveFromSelection method will remove one or more nodes in the selection from the Selection Class. See the selection class for more details.

Example: `doTNTCommand('Node("Torus01").RemoveFromSelection()');`

This example will add the node called "Torus01" from the selection class.

3.11.1.15. GetLODIndex

Syntax: GetLODIndex()

The GetLODIndex() will return the LOD index of an node in the selection.

Example: `doTNTCommand('Node("MyLOD").GetLODIndex()');`

This example returns the LOD index of the node called "MyLOD".

3.11.1.16. SetAnimation

Syntax: SetAnimation(animationName:string)

The SetAnimation method can be used to assign an Animation to one or more nodes in the selection. The argument animationName is a search string to find the animation.

Example: `doTNTCommand('Node("*Anim").SetAnimation("MyAnimation2")');`

This example will assign the animation called "MyAnimation2" to all nodes that have a name that ends with "Anim".

3.11.1.17. Match

Syntax: Match(nodeName:string, duration:integer)

The Match method can be used for match the position and rotation of a node in the selection to another node in the scene. The nodeName argument is the name of the node in the scene that you wish to match. The duration argument is the number of milliseconds that the match will take to complete. Slowly matching of one node to another creates a good visual effect.

Example: `doTNTCommand('Node("Pyramid01").Match("Pyramid02",2000)');`
This example will match the position and rotation of the of "Pyramid01" to "Pyramid02" during a period of 2000 milliseconds.

3.11.1.18. GetExternalResourcePath
Syntax: GetExternalResourcePath()
The GetExternalResourcePath method will return the value of the ExternalResourcePath string for the node in the selection. See SetExternalResourcePath for more details.
Example: `doTNTCommand('Node('ExternNode').GetExternalResourcePath()');`
This example will return the ExternalResourcePath value of the node called "ExternNode".

3.11.1.19. SetExternalResourcePath
Syntax: SetExternalResourcePath(relativePath:string)
The SetExternalResourcePath method can be used to load another TNT file onto a node in the selection. When value of the ExternResourcePath changed a request will be added to the download queue. When the file is downloaded and loaded and added to the scene the OnExternalResourceMerge event will be generated.
Example:
```
doTNTCommand('CreateNode("ExternNode5").SetExternalResourcePath("teapot.tnt")'
);
```
This example will create a node called "ExternNode5" and set it's ExternalResourcePath to point to a TNT file called "teapot.tnt".

3.11.1.20. GetExternalResourceInstanceFlags
Syntax: GetExternalResourceInstanceFlags()
The GetExternalResourceInstanceFlags will return the value of the ExternalResourceInstanceFlags of the node in the selection.
Example:
`doTNTCommand('Node("ExternNode").GetExternalResourceInstanceFlags()');`
This example will return the ExternalResourceInstanceFlags of the node called "ExternNode".

3.11.1.21. SetExternalResourceInstanceFlags
Syntax: SetExternalResourceInstanceFlags(value:integer)
The SetExternalResourceInstanceFlags method can be used to change the value of the ExternalResourceInstanceFlags value of the node in the selection. If the value is zero then all resources in the external resource will be shared between nodes pointing their ExternalResourcePath to the same file. If the value of the ExternalResourceInstanceFlags is set to -1 then all resources in the external resource will not be shared between nodes pointing their ExternalResourcePath to the same file.
Example: `doTNTCommand('Node("ExternNode").SetExternalResourceInstanceFlags(-1)');`
This example will set the ExternalResourceInstanceFlags value of the node called "ExternNode".

3.11.1.22. GetLocalPositionX
Syntax: GetLocalPositionX()

The GetLocalPositionX method will return the X Position of the node in the selection. The position is in local space which means that to node coordinates are relative to the parent.
Example: `doTNTCommand('Node("Teapot01").GetLocalPositionX()');`
This example returns the Local X Position of the node called "Teapot01".

3.11.1.23. GetLocalPositionY
Syntax: GetLocalPositionY()
The GetLocalPositionY method will return the Y Position of the node in the selection. The position is in local space which means that to node coordinates are relative to the parent.
Example: `doTNTCommand('Node("Teapot01").GetLocalPositionY()');`
This example returns the Local Y Position of the node called "Teapot01".

3.11.1.24. GetLocalPositionZ
Syntax: GetLocalPositionZ()
The GetLocalPositionY method will return the Y Position of the node in the selection. The position is in local space which means that to coordinates are relative to the parent.
Example: `doTNTCommand('Node("Teapot01").GetLocalPositionZ()');`
This example returns the Local Z Position of the node called "Teapot01".

3.11.1.25. SetLocalPosition
Syntax: SetLocalPosition(x:float,y:float,z:float)
The SetLocalPosition method will set the Local Position of the node in the selection. The position is in local space which means that to coordinates are relative to the parent. The x,y,z arguments are the x,y,z coordiates of the position vector.
Example: `doTNTCommand('Node("Teapot01").SetLocalPosition(5.0,-7.0,2.0)');`
This example sets the Local Position of the node called "Teapot01".

3.11.1.26. GetLocalRotationX
Syntax: GetLocalRotationX()
The GetLocalRotationX method will return the X Rotation of the node in the selection. The position is in local space which means that to coordinates are relative to the parent.
Example: `doTNTCommand('Node("Teapot01").GetLocalRotationX()');`
This example returns the Local X Rotation of the node called "Teapot01".

3.11.1.27. GetLocalRotationY
Syntax: GetLocalRotationY()
The GetLocalRotationY method will return the Y Rotation of the node in the selection. The position is in local space which means that to coordinates are relative to the parent.
Example: `doTNTCommand('Node("Teapot01").GetLocalRotationY()');`
This example returns the Local Y Rotation of the node called "Teapot01".

3.11.1.28. GetLocalRotationZ
Syntax: GetLocalRotationZ()
The GetLocalRotationZ method will return the Z Rotation of the node in the selection. The position is in local space which means that to node coordinates is relative to the

parent.
Example: `doTNTCommand('Node("Teapot01").GetLocalRotationZ()');`
This example returns the Local Z Rotation of the node called "Teapot01".

3.11.1.29. SetLocalRotation
Syntax: SetLocalRotation(x:float,y:float,z:float)
The SetLocalRotation method will set the Local Rotation of the node in the selection.
The rotation is in local space which means that to coordinates are relative to the parent.
The x,y,z arguments are the x,y,z coordiates of the rotation vector.
Example: `doTNTCommand('Node("Teapot01").SetLocalRotation(0.0,0.0,90.0)');`
This example sets the Local Rotation of the node called "Teapot01".

3.11.1.30. GetLocalScaleX
Syntax: GetLocalScaleX()
The GetLocalScaleX method will return the X Scale of the node in the selection.
The scale is in local space which means that to angles are relative to the parent.
Example: `doTNTCommand('Node("Teapot01").GetLocalScaleX()');`
This example returns the Local X Scale of the node called "Teapot01".

3.11.1.31. GetLocalScaleY
Syntax: GetLocalScaleY()
The GetLocalScaleY method will return the Y Scale of the node in the selection.
The scale is in local space which means that to angles are relative to the parent.
Example: `doTNTCommand('Node("Teapot01").GetLocalScaleY()');`
This example returns the Local Y Scale of the node called "Teapot01".

3.11.1.32. GetLocalScaleZ
Syntax: GetLocalScaleZ()
The GetLocalScaleZ method will return the Z Scale of the node in the selection.
The scale is in local space which means that to angles are relative to the parent.
Example: `doTNTCommand('Node("Teapot01").GetLocalScaleZ()');`
This example returns the Local Z Scale of the node called "Teapot01".

3.11.1.33. SetLocalScale
Syntax: SetLocalScale(x:float,y:float,z:float)
The SetLocalScale method will set the Local Scale of the node in the selection.
The scale is in local space which means that to coordinates are relative to the parent.
The x,y,z arguments are the x,y,z coordiates of the scale vector.
Example: `doTNTCommand('Node("Teapot01").SetLocalScale(2.0,2.0,2.0)');`
This example sets the Local Scale of the node called "Teapot01".

3.11.1.34. GetWorldPositionX
Syntax: GetWorldPositionX()
The GetWorldPositionX method will return the X Position in WorldSpace of the node in the selection.
Example: `doTNTCommand('Node("Teapot01").GetWorldPositionX()');`

This example returns the World X Position of the node called "Teapot01".

3.11.1.35. GetWorldPositionY
Syntax: GetWorldPositionY()
The GetWorldPositionY method will return the Y Position in WorldSpace of the node in the selection.
Example: `doTNTCommand('Node("Teapot01").GetWorldPositionY()');`
This example returns the World Y Position of the node called "Teapot01".

3.11.1.36. GetWorldPositionZ
Syntax: GetWorldPositionZ()
The GetWorldPositionZ method will return the Z Position in WorldSpace of the node in the selection.
Example: `doTNTCommand('Node("Teapot01").GetWorldPositionZ()');`
This example returns the World Z Position of the node called "Teapot01".

3.11.1.37. SetWorldPosition
Syntax: SetWorldPosition(x:float,y:float,z:float)
The SetLocalPosition method will set the Local Position of the node in the selection. The position is in local space which means that to coordinates are relative to the parent. The x,y,z arguments are the x,y,z coordiates of the position vector.
Example: `doTNTCommand('Node("Teapot01").SetWorldPosition(100.0,200.0,300.0)');`
This example sets the WorldSpace Position of the node called "Teapot01".

3.11.1.38. GetWorldRotationX
Syntax: GetWorldRotationX()
The GetWorldRotationX method will return the WorldSpace X Rotation of the node in the selection.
Example: `doTNTCommand('Node("Teapot01").GetWorldRotationX()');`
This example returns the WorldSpace X Rotation of the node called "Teapot01".

3.11.1.39. GetWorldRotationY
Syntax: GetWorldRotationY()
The GetWorldRotationY method will return the WorldSpace Y Rotation of the node in the selection.
Example: `doTNTCommand('Node("Teapot01").GetWorldRotationY()');`
This example returns the WorldSpace Y Rotation of the node called "Teapot01".

3.11.1.40. GetWorldRotationZ
Syntax: GetWorldRotationZ()
The GetWorldRotationZ method will return the WorldSpace Z Rotation of the node in the selection.
Example: `doTNTCommand('Node("Teapot01").GetWorldRotationZ()');`
This example returns the WorldSpace Z Rotation of the node called "Teapot01".

3.11.1.41. SetWorldRotation
Syntax: SetWorldRotation(x:float,y:float,z:float)

The SetLocalRotation method will set the Local Rotation of the node in the selection. The x,y,z arguments are the x,y,z coordiates of the rotation vector.
Example: `doTNTCommand('Node("Teapot01").SetWorldRotation(90.0,0.0,0.0)');`
This example sets the Local Rotation of the node called "Teapot01".

3.11.1.42. GetWorldScaleX
Syntax: GetWorldScaleX()
The GetWorldScaleX method will return the WorldSpace X Scale of the node in the selection.
Example: `doTNTCommand('Node("Teapot01").GetWorldScaleX()');`
This example returns the WorldSpace X Scale of the node called "Teapot01".

3.11.1.43. GetWorldScaleY
Syntax: GetWorldScaleY()
The GetWorldScaleY method will return the WorldSpace Y Scale of the node in the selection.
Example: `doTNTCommand('Node("Teapot01").GetWorldScaleY()');`
This example returns the WorldSpace Y Scale of the node called "Teapot01".

3.11.1.44. GetWorldScaleZ
Syntax: GetWorldScaleZ()
The GetWorldScaleZ method will return the WorldSpace Z Scale of the node in the selection.
Example: `doTNTCommand('Node("Teapot01").GetWorldScaleY()');`
This example returns the WorldSpace Z Scale of the node called "Teapot01".

3.11.1.45. SetWorldScale
Syntax: SetWorldScale(x:float,y:float,z:float)
The SetLocalScale method will set the WorldSpace Scale of the node in the selection. The x,y,z arguments are the x,y,z coordiates of the scale vector.
Example: `doTNTCommand('Node("Teapot01").SetWorldScale(2.0,2.0,2.0)');`
This example sets the WorldSpace Scale of the node called "Teapot01".

3.11.1.46. GenerateBox
Syntax: GenerateBox(x:float,y:float,z:float)
The GenerateBox method will generate a box mesh using the node in the selection. The x,y,z arguments will become that x,y,z size lengths of generated box.
Example: `doTNTCommand('CreateNode("Box5").GenerateBox(2.0,3.0,4.0)');`
This example will create a node called "Box5" and generate a box with 2.0, 3.0, 4.0 as side lengths.

3.11.1.47. GeneratePyramid
Syntax: GeneratePyramid(fov:float,height:float)
The GeneratePyramid method will generate a pyramid mesh using the node in the selection. The fov argument will define the angle of the pyramid. The lenght argument will be define the height of the pyramid.
Example: `doTNTCommand('CreateNode('Pyramid4').GeneratePyramid(30.0,100.0)');`

This example will createa a node called "Pyramid4" and generate a pyramid mesh with a angle of 30.0 and a height of 100.0.

3.12.Viewport

3.12.1.1. SetCurrentViewport
Syntax: SetCurrentViewport()
The SetCurrentViewport method will set the viewport in the selection to be the current viewport. The current viewport is the viewport that is used in the rendering process.
Example:
```
doTNTCommand('CreateViewport("MyViewport").SetViewportBackgroundColor(#00007f)
.SetViewportAmbientColor(#303030).SetViewportMinLight(#000000).SetViewportMaxL
ight(#ffffff).SetCurrentViewport()');
```
This example creates a viewport called "MyViewport" the viewport is then configured and finally set as current viewport.

3.12.1.2. GetCurrentViewport
Syntax: GetCurrentViewport()
The GetCurrentViewport method will return 1 if the viewport in the selection is the current viewport and 0 if it is not.
Example: `doTNTCommand('Viewport(0).GetCurrentViewport()');`
This example returns whether or not the viewport at index 0 is the current viewport.

3.12.1.3. GetViewportFrameCount
Syntax: GetViewportFrameCount()
The GetViewportFrameCount method will return how many ViewportFrames the Viewport in the selection have.
Example: `doTNTCommand('Viewport(0).GetViewportFrameCount()');`
This example will return how many ViewportFrames the viewport at index 0 have.

3.12.1.4. AppendViewportFrames
Syntax: AppendViewportFrames(count:integer)
The AppendViewportFrames method can be used to append ViewportFrames on one or more Viewports in the selection. The argument count is the number of ViewportFrames that will be added at the end of the animation timeline.
Example: `doTNTCommand('Viewport("*").AppendViewportFrames(100)');`
This example will append 100 ViewportFrames to all viewports in the scene.

3.12.1.5. RemoveViewportFrames
Syntax: RemoveViewportFrames(index:integer,count:integer)
The RemoveViewportFrames method can be used to remove ViewportFrames from one or more viewports in the selection. The index argument is the start frame of the removal and the count argument is how many frames that you wish to remove.
Example: `doTNTCommand('Viewport(1).RemoveViewportFrames(12,20)');`
This example will remove the ViewportFrames from 12 to 32 of the viewport at index 1.

3.13.ViewportFrame

3.13.1.1. SetViewportBackgroundColor

Syntax: SetViewportBackgroundColor(color:color)

The SetViewportBackgroundColor method can be used to set the Background color of one or more ViewportFrames in the selection. The Background color is the inital color of the screen before rendering of the meshes begins.
Example:

```
doTNTCommand('CurrentViewport().SetViewportBackgroundColor(#006080)');
```

This example sets the Background color to #006080 on all ViewportFrames of the current viewport.

3.13.1.2. GetViewportBackgroundColor

Syntax: GetViewportBackgroundColor()

The GetViewportBackgroundColor returns the background color of the first ViewportFrame in the selection.
Example: `doTNTCommand('CurrentViewport().GetViewportBackgroundColor()');`
This examples returns the Background color of the current viewport.

3.13.1.3. SetViewportMinLight

Syntax: SetViewportMinLight(color:color)

The SetViewportMinLight method can be used to set the Minimum color of one or more ViewportFrames in the selection. The Minimum color of the Viewport is the minimum color in the scene. It can be used to make an overall impression of all objects in the scene being illuminated from every direction.
Example: `doTNTCommand('CurrentViewport().SetViewportMinColor(#303030)');`
This example set the ViewportMinColor to #303030 on all ViewportFrames of the current viewport.

3.13.1.4. GetViewportMinLight

Syntax: GetViewportMinLight()

The GetViewportMinLight method return the Minimum Color of the first ViewportFrame in the selection.
Example: `doTNTCommand('CurrentViewport().GetViewportMinColor()');`
This examples returns the Minimum color of the current viewport.

3.13.1.5. SetViewportMaxLight

Syntax: SetViewportMaxLight(color:color)

The SetViewportMaxLight method can be used to set the Maximum color of one or more ViewportFrames in the selection. The Maximum color of the Viewport is the maximum color in the scene. It can be used to lower the overall illumination of all objects in the scene.
Example: `doTNTCommand('CurrentViewport().SetViewportMaxColor(#FFFFFF)');`
This example set the ViewportMaxColor to #FFFFFF on all ViewportFrames of the current viewport.

3.13.1.6. GetViewportMaxLight

Syntax: GetViewportMaxLight()

The GetViewportBackgroundColor returns the background color of the first ViewportFrame in the selection.
Example: `doTNTCommand('CurrentViewport().GetViewportMaxColor()');`
This examples returns the Maximum color of the current viewport.

3.13.1.7. SetViewportFogMode
Syntax: SetViewportFogMode(mode:integer)
The SetViewportFogMode method can be used to set the Fog mode of one or more ViewportFrames in the selection. An overview of the Fog mode values can be seen below.
0 = No Fog.
1 = Exponential Fog
2 = Exponential Squared Fog
3 = Linear Fog
Example:
```
doTNTCommand('Viewport(0).SetViewportBackgroundColor(0xA8C5D7).SetViewportFogColor(0xA8C5D7).SetViewportFogStart(100).SetViewportFogEnd(5000).SetViewportFogMode(3)');
```
This example enables Linear Fog using the color #A8C5D7 with starts with 100 and reaches the full itensity at 5000 distance from the camera.

3.13.1.8. GetViewportFogMode
Syntax: GetViewportFogMode()
The GetViewportFogMode method return the FogMode of the first ViewportFrame in the selection.
Example: `doTNTCommand('CurrentViewport().GetViewportFogMode()');`
This example returns the FogMode of the currentViewport.

3.13.1.9. SetViewportFogColor
Syntax: SetViewportFogColor(col:integer)
The SetViewportFogColor method can be used to set the Fog Color of one or more ViewportFrames in the selection.
Example:
```
doTNTCommand('Viewport(0).SetViewportBackgroundColor(0xA8C5D7).SetViewportFogColor(0xA8C5D7).SetViewportFogStart(100).SetViewportFogEnd(5000).SetViewportFogMode(3)');
```
This example enables linear fog with using the color #A8C5D7 with starts with 100 and reaches the full itensity at 5000 distance from the camera.

3.13.1.10. GetViewportFogColor
Syntax: GetViewportFogColor()
The GetViewportFogMode method returns the FogMode of the first ViewportFrame in the selection.
Example: `doTNTCommand('CurrentViewport().GetViewportFogColor()');`
This example returns the ViewportFogColor of the current Viewport.

3.13.1.11. SetViewportFogStart
Syntax: SetViewportFogStart(start:float)
The SetViewportFogStart method can be used to set the Fog Start of one or more

ViewportFrames in the selection.
Example:
```
doTNTCommand('Viewport(0).SetViewportBackgroundColor(0xA8C5D7).SetViewportFogC
olor(0xA8C5D7).SetViewportFogStart(100).SetViewportFogEnd(5000).SetViewportFog
Mode(3)');
```
This example enables linear fog with using the color #A8C5D7 with starts with 100 and reaches the full itensity at 5000 distance from the camera.

3.13.1.12. GetViewportFogStart
Syntax: GetViewportFogStart()
The GetViewportFogStart method returns the FogStart of the first ViewportFrame in the selection.
Example: `doTNTCommand('CurrentViewport().GetViewportFogStart()');`
This example returns the Fog Start value of the CurrentViewport.

3.13.1.13. SetViewportFogEnd
Syntax: SetViewportFogEnd(end:float)
The SetViewportFogEnd method can be used to set the Fog End of one or more ViewportFrames in the selection. The Fog End value determines where the Fog reaches maximum intensity.
Example:
```
doTNTCommand('Viewport(0).SetViewportBackgroundColor(0xA8C5D7).SetViewportFogC
olor(0xA8C5D7).SetViewportFogStart(100).SetViewportFogEnd(5000).SetViewportFog
Mode(3)');
```
This example enables linear fog with using the color #A8C5D7 with starts with 100 and reaches the full itensity at 5000 distance from the camera.

3.13.1.14. GetViewportFogEnd
Syntax: GetViewportFogEnd()
The GetViewportFogMode method returns the Fog End value of the first ViewportFrame in the selection.
Example: `doTNTCommand('CurrentViewport().GetViewportFogEnd()');`
This example returns the Fog End value of the CurrentViewport.

3.13.1.15. CreateBackground
Syntax: CreateBackground()
The CreateBackground method creates an empty black background image on one or more ViewportFrames in the selection. If a background image exists on a ViewportFrame then the Background color will not be used in the rendering process.
Example:
```
doTNTCommand('Viewport(0).ViewportFrame().CreateBackground().LoadBitmap("backg
round.png")');
```
This example creates a background image and loads an image called "background.png" onto the background. The background is created on all ViewportFrames of the Viewport at index 0.

3.13.1.16. DestroyBackground
Syntax: DestroyBackground()
The DestroyBackground method can be used to destroy the background on one or more

ViewportFrames in the selection.
Example: `doTNTCommand('CurrentViewport().DestroyBackground()');`
This example destroys the background on all ViewportFrames associated with the Current Viewport.

3.13.1.17. HasBackground
Syntax: HasBackground()
The HasBackground method return a value that define whether or not the first ViewportFrame in the selection has a background. If the return value is 0 then no background is available. If the return value is 1 then a background exists.
Example: `doTNTCommand('Viewport(0).HasBackground()');`
This example will return whether or not the first ViewportFrame of Viewport at index 0 has a background.

3.14.Camera

3.14.1.1. SetCurrentCamera
Syntax: SetCurrentCamera()
The SetCurrentCamera method sets the camera in the selection to be the current camera. The current camera is the camera used for the rendering process.
Example: `doTNTCommand('Camera("Camera01").SetCurrentCamera()');`
This example sets the camera called "Camera01" to be the current camera.

3.14.1.2. GetCurrentCamera
Syntax: GetCurrentCamera()
The GetCurrentCamera method returns a value defining whether or not the camera in the selection is the current camera.
Example: `doTNTCommand('Camera("Camera01").GetCurrentCamera()');`
This example return a value defining whether or not the camera called "Camera01" is the current camera.

3.14.1.3. SetCameraMovableValue
Syntax: SetCameraMovableValue(integer)
The SetCameraMovableValue method sets the MovableValue of one or more camera in the selection. An overview of the Movable values can be seen below.
0 = No.
1 = Yes.
2 = Auto.
If the Movable value is 'No' then the user will not be able to move the camera using mouse and keyboard.
If the Movable value is 'Yes' then the user will be able to move the camera using mouse and keyboard.
Finally if the Movable value is 'Auto' then the camera will not be movable if it contains a timeline animation, but it will be Movable if it only have no animation.
Example: `doTNTCommand('Camera("Camera01").SetCameraMovableValue(1)');`
This example sets the MovableValue of the Camera called "Camera01" to 'Yes'.

3.14.1.4. GetCameraMovableValue
Syntax: GetCameraMovableValue()
The GetCameraMovableValue returns the MovableValue of the first camera in the selection.
Example: `doTNTCommand('Camera("Camera01").GetCameraMovableValue()');`
This example returns the MovableValue of the Camera called "Camera01".

3.14.1.5. SetCameraIgnoreInput
Syntax: SetCameraIgnoreInput(integer)
The SetCameraIgnoreInput method sets the IgnoreInput value of one or more camera in the selection. When the IgnoreInput value if 1 then this camera will ignore all input from mouse and keyboard.
Example: `doTNTCommand('CurrentCamera().SetCameraIgnoreInput(1)');`
This example sets the IgnoreInput value of the current camera to 1.

3.14.1.6. GetCameraIgnoreInput
Syntax: GetCameraIgnoreInput()
The GetCameraIgnoreInput method returns the IgnoreInput value of the first Camera in the selection.
Example: `doTNTCommand('Camera("Camera02").GetCameraIgnoreInput()');`
This example returns IgnoreInput value of the camera called "Camera02".

3.14.1.7. SetCameraTarget
Syntax: SetCameraTarget(nodeName:string)
The SetCameraTarget method sets Target of the camera in the selection. The camera in the selection will become a Target Camera which means that the camera always will look at the Target node. The function will locate the new Target node by searching the scene using the nodeName argument.
Example: `doTNTCommand('CreateCamera("Camera1").SetCameraTarget("Target1")');`
This example creates a camera called "Camera1" and sets the target of the camera to be a node called "Target1".

3.14.1.8. SetCameraMinDistance
Syntax: SetCameraMinDistance(distance:float)
The SetCameraMinDistance method sets the MinDistance value of the camera in the selection to the value of the distance argument. The MinDistance value is the Minimum Distance allowed between the camera and the camera target.
Example: `doTNTCommand('Camera("Camera01").SetCameraMinDistance(100.0)');`
This examples sets the MinDistance of the camera called "Camera01" to 100.0.

3.14.1.9. GetCameraMinDistance
Syntax: GetCameraMinDistance()
The GetCameraMinDistance method returns the MinDistance value of the first camera in the selection.
Example: `doTNTCommand('Camera("Camera02").GetCameraMinDistance()');`
This example returns the MinDistance value of the camera called "Camera02".

3.14.1.10. SetCameraMaxDistance
Syntax: SetCameraMaxDistance(distance:float)
The SetCameraMaxDistance method sets the MaxDistance value of the camera in the selection to the value of the distance argument. The MaxDistance value is the Maximum Distance allowed between the camera and the camera target.
Example: `doTNTCommand('Camera("Camera02").SetCameraMaxDistance(5000.0)');`
This examples sets the MaxDistance of the camera called "Camera02" to 5000.0.

3.14.1.11. GetCameraMaxDistance
Syntax: GetCameraMaxDistance()
The GetCameraMaxDistance method returns the MaxDistance value of the first camera in the selection.
Example: `doTNTCommand('Camera("Camera02").GetCameraMaxDistance()');`
This example returns the MaxDistance value of the camera called "Camera02".

3.14.1.12. SetCameraMoveSpeed
Syntax: SetCameraMoveSpeed(velocity:float)
The SetCameraMoveSpeed method sets the MoveSpeed value of the camera in the selection to the value of the distance argument. The MoveSpeed value is the speed of which the camera moves when pressing the arrow keys or when holding down both mouse buttons and moving mouse up, down, left or right.
Example: `doTNTCommand('Camera("Camera02").SetCameraMoveSpeed(0.8)');`
This examples sets the MoveSpeed of the camera called "Camera02" to 0.8.

3.14.1.13. GetCameraMoveSpeed
Syntax: GetCameraMoveSpeed()
The GetCameraMoveSpeed method returns the MoveSpeed value of the first camera in the selection.
Example: `doTNTCommand('Camera("Camera02").GetCameraMoveSpeed()');`
This example returns the MoveSpeed value of the camera called "Camera02".

3.14.1.14. SetCameraRotationSpeedU
Syntax: SetCameraRotationSpeedU(speed:float)
The SetCameraRotationSpeedU method sets the RotationSpeedU value of the camera in the selection to the value of the speed argument. As the RotationSpeedU value increase then the camera will rotate faster when moving left and right.
Example: `doTNTCommand('Camera("Camera02").SetCameraRotationSpeedU(0.009)');`
This example sets the RotationSpeedU to 0.009 of the camera called "Camera02".

3.14.1.15. GetCameraRotationSpeedU
Syntax: GetCameraRotationSpeedU()
The GetCameraRotationSpeedU method returns the RotationSpeedU value of the first camera in the selection.
Example: `doTNTCommand('Camera("Camera02").GetCameraRotationSpeedU()');`

This example return the RotationSpeedU value of the camera called "Camara02".

3.14.1.16. SetCameraRotationSpeedV

Syntax: SetCameraRotationSpeedV(speed:float)

The SetCameraRotationSpeedV method sets the RotationSpeedV value of the camera in the selection to the value of the speed argument. As the RotationSpeedV value increase then the camera will rotate faster when moving up and down.

Example: `doTNTCommand('Camera("Camera02").SetCameraRotationSpeedV(0.009)');`

This example sets the RotationSpeedV to 0.009 of the camera called "Camera02".

3.14.1.17. GetCameraRotationSpeedV

Syntax: GetCameraRotationSpeedV()

The GetCameraRotationSpeedV method returns the RotationSpeedV value of the first camera in the selection.

Example: `doTNTCommand('Camera("Camera02").GetCameraRotationSpeedV()');`

This example return the RotationSpeedV value of the camera called "Camara02".

3.14.1.18. SetCameraMaxUAngle

Syntax: SetCameraMaxUAngle(degrees:float)

The SetCameraMaxUAngle method sets the MaxUAngle value of the camera in the selection to the value of the degrees argument. The MaxUAngle will be the Maximum angle allowed between the x & y unit vector of the camera direction.

Example: `doTNTCommand('Camera("Camera02").SetCameraMaxUAngle(180)');`

This example sets the MaxUAngle to 180 of the camera called "Camera02".

3.14.1.19. GetCameraMaxUAngle

Syntax: GetCameraMaxUAngle()

The GetCameraMaxUAngle method returns the MaxUAngle value of the first camera in the selection.

Example: `doTNTCommand('Camera("Camera02").GetCameraMaxUAngle()');`

This example return the MaxUAngle value of the camera called "Camera02".

3.14.1.20. SetCameraMinUAngle

Syntax: SetCameraMinUAngle(degrees:float)

The SetCameraMinUAngle method sets the MinUAngle value of the camera in the selection to the value of the degrees argument.The MinUAngle will be the Minimum angle allowed between the x & y unit vector of the camera direction.

Example: `doTNTCommand('Camera("Camera02").SetCameraMaxUAngle(0.0)');`

This example sets the MinUAngle to 0.0 of the camera called "Camera02".

3.14.1.21. GetCameraMinUAngle

Syntax: GetCameraMinUAngle()

The GetCameraMinUAngle method returns the MinUAngle value of the first camera in the selection.

Example: `doTNTCommand('Camera("Camera02").GetCameraMinUAngle()');`
This example returns the MinUAngle value of the camera called "Camera02".

3.14.1.22. SetCameraMaxVAngle
Syntax: SetCameraMaxVAngle(degrees:float)
The SetCameraMaxUAngle method sets the MaxVAngle value of the camera in the selection to the value of the degrees argument. The MaxVAngle will be the Maximum allow angle between the y & z unit vector of the camera.
Example: `doTNTCommand('Camera("Camera02").SetCameraMaxVAngle(180)');`
This example sets the MaxVAngle to 180 of the camera called "Camera02".

3.14.1.23. GetCameraMaxVAngle
Syntax: GetCameraMaxVAngle()
The GetCameraMaxVAngle method returns the MaxVAngle value of the first camera in the selection.
Example: `doTNTCommand('Camera("Camera02").GetCameraMaxVAngle()');`
This example return the MaxVAngle value of the camera called "Camera02".

3.14.1.24. SetCameraMinVAngle
Syntax: SetCameraMinVAngle(degrees:float)
The SetCameraMinVAngle method sets the MinVAngle value of the camera in the selection to the value of the degrees argument.The MinVAngle will be the Minimum angle allowed between the y & z unit vector of the camera direction.
Example: `doTNTCommand('Camera("Camera02").SetCameraMinVAngle(0.0)');`
This example sets the MinVAngle to 0.0 of the camera called "Camera02".

3.14.1.25. GetCameraMinVAngle
Syntax: GetCameraMinVAngle()
The GetCameraMinVAngle method returns the MinVAngle value of the first camera in the selection.
Example: `doTNTCommand('Camera("Camera02").GetCameraMinVAngle()');`
This example returns the MinVAngle value of the camera called "Camera02".

3.14.1.26. GetCameraFrameCount
Syntax: GetCameraFrameCount()
The GetCameraFrameCount method will return how many CameraFrames the first Camera in the selection have.
Example: `doTNTCommand('CurrentCamera().GetCameraFrameCount()');`
This example will return how many CameraFrames the current camera have.

3.14.1.27. AppendCameraFrames
Syntax: AppendCameraFrames(count:integer)
The AppendCameraFrames method can be used to append CameraFrames on one or more Cameras in the selection. The count argument is the number of CameraFrames that will be added at the end of the animation timeline.
Example: `doTNTCommand('Camera("Camera01").AppendCameraFrames(100)');`

This example will append 100 CameraFrames to the camera called "Camera01.

3.14.1.28. RemoveCameraFrames

Syntax: RemoveCameraFrames(index:integer,count:integer)

The RemoveCameraFrames method can be used to remove CameraFrames from one or more cameras in the selection. The index argument is the start frame of the removal and the count argument is how many frames that you wish to remove.

Example: `doTNTCommand('Camera("Camera02").RemoveCameraFrames(12,20)');`

This example will remove the CameraFrames from 12 to 32 of the camera called "Camera02".

3.15.CameraFrame

3.15.1.1. SetCameraFOV

Syntax: SetCameraFOV(degrees:float)

The SetCameraFOV method sets the FOV value on one or more CameraFrames in the selection. The FOV stands for "Field Of View" and determines the view angle of the camera.

Example: `doTNTCommand('Camera("Camera01").SetCameraFOV(90)');`

This example sets the FOV value to 90 degrees on all CameraFrames of the camera called ""Camera01".

3.15.1.2. GetCameraFOV

Syntax: GetCameraFOV()

The GetCameraFOV method returns the FOV value of the first CameraFrame in the selection.

Example: `doTNTCommand('CurrentCamera().GetCameraFOV()');`

This example will return the FOV of the current camera.

3.15.1.3. SetCameraNearClipPlane

Syntax: SetCameraNearClipPlane(nearClip:float)

The SetCameraNearClipPlane method sets the NearClipPlane value on one or more CameraFrames in the selection. The NearClipPlane is the shortest distance between the camera and a polygon before camera clipping occur.

Example: `doTNTCommand('Camera("Camera01").SetCameraNearClipPlane(1.0)');`

This example sets the NearClipPlane of the camera called "Camera01" to 1.0.

3.15.1.4. GetCameraNearClipPlane

Syntax: GetCameraNearClipPlane()

The GetCameraNearClipPlane method returns the NearClipPlane value of the first CameraFrame in the selection.

Example: `doTNTCommand('Camera("Camera01").GetCameraNearClipPlane()');`

This example returns the NearClipPlane value of camera called "Camera01".

3.15.1.5. SetCameraFarClipPlane

Syntax: SetCameraFarClipPlane(farClip:float)

The SetCameraFarClipPlane method sets the FarClipPlane value on one or more CameraFrames in the selection. The default value of farClipPlane is -1. When

farClip value is -1 then TurnTool automatically fits clipping planes to the scene in real time.
Example: `doTNTCommand('Camera("Camera01").SetCameraNearClipPlane(8000.0)');`
This example sets the FarClipPlane of the camera called "Camera01" to 8000.0.

3.15.1.6. GetCameraFarClipPlane
Syntax: GetCameraFarClipPlane()
The GetCameraFarClipPlane method returns the FarClipPlane value of the first CameraFrame in the selection.
Example: `doTNTCommand('Camera("Camera01").GetCameraFarClipPlane()');`
This example returns the FarClipPlane value of camera called "Camera01".

3.16.Light

3.16.1.1. GetLightFrameCount
Syntax: GetLightFrameCount()
The GetLightFrameCount method will return how many LightFrames the first Light in the selection have.
Example: `doTNTCommand('Light(0).GetLightFrameCount()');`
This example will return how many LightFrames the Light at index 0 have.

3.16.1.2. AppendLightFrames
Syntax: AppendLightFrames(count:integer)
The AppendLightFrames method can be used to append LightFrames to one or more Lights in the selection. The argument count is the number of LightFrames that will be added at the end of the animation time line.
Example: `doTNTCommand('Light("Omni01").AppendLightFrames(100)');`
This example will append 100 LightFrames to the "Omni01" in the scene.

3.16.1.3. RemoveLightFrames
Syntax: RemoveLightFrames(start:integer,end:integer)
The RemoveViewportFrames method can be used to remove ViewportFrames from one or more viewports in the selection. The index argument is the start frame of the removal and the count argument is how many frames that you wish to remove.
Example: `doTNTCommand('Light("Omni02").RemoveLightFrames(10,20)');`
This example will remove the LightFrames from index 10 to 30 on the light called "Omni02".

3.17.LightFrame

3.17.1.1. SetLightRange
Syntax: SetLightRange(range:float)
The SetLightRange method sets the LightRange value on one or more LightFrames in the selection.
Example: `doTNTCommand('Light("*").SetLightRange(10000)');`
This example sets the LightRange of all lights in the scene to 1000.

3.17.1.2. GetLightRange

Syntax: GetLightRange()

The GetLightRange method returns the Range value of the first LightFrame in the selection.
The Range of the light is the distance from the light to a vertex where the light has an effect.
Example: `doTNTCommand('Light(0).GetLightRange()');`
This example returns the Range of the light at index 0.

3.17.1.3. SetLightAttenuation

Syntax: SetLightAttenuation(c:float, b:float, a:float)

The SetLightAttenuation method sets the LightAttenuation values on one or more LightFrames in the selection. The Light Attenuation values are used to define how the light intensity decrease as the distance from the light source to a vertex increase.
The arguments c, b and a are variables in the following quadratic equation that can be used to calculate the Light Intensity 'i' at distance 'x' from the light.

$$i = a*x*x + b*x + c$$

Example: `doTNTCommand('Light(0).SetLightAttenuation(1.5,0,0)');`
This example sets the Light Attenuation on all LightFrames of the Light at index 0 to these values c = 1.5 and b = 0 and a = 0.

3.17.1.4. GetLightAttenuation

Syntax: GetLightAttenuation(index:integer)

The GetLightAttenuation method returns the LightAttenuation of the first LightFrame in the selection. The argument index is used to define which variables a, b, c that will be returned.
A index argument of 0 will return the c variable.
A index argument of 1 will return the b variable.
A index argument of 2 will return the a variable.
Example: `doTNTCommand('Light(0).GetLightAttenuation(0)');`
This example returns the c variable in the light attenuation equation of the the first LightFrame of the Light at index 0.

3.17.1.5. SetLightDiffuseColor

Syntax: SetLightDiffuseColor(color:integer)

The SetLightDiffuseColor method sets the DiffuseColor of one or more LightFrames in the selection. The LightDiffuseColor is the diffuse color illuminated by the light-source.
Example: `doTNTCommand('Light("Omni01").SetLightDiffuseColor(#FF0000)');`
This example sets the DiffuseColor of all LightFrames in the light called "Omni01" to red.

3.17.1.6. GetLightDiffuseColor

Syntax: GetLightDiffuseColor()

The GetLightDiffuseColor method return the DiffuseColor of the first light source in the selection.

Example: `doTNTCommand('Light("Omni01").GetLightDiffuseColor()');`
This example returns the DiffuseColor of the light called "Omni01".

3.17.1.7. SetLightSpecularColor
Syntax: SetLightSpecularColor(color:integer)
The SetLightSpecularColor method sets the SpecularColor of one or more LightFrames in the selection. The LightSpecularColor is the specular color illuminated by the light-source.
Example: `doTNTCommand('Light("Omni01").SetLightSpecularColor(#00FF00)');`
This example sets the SpecularColor of all LightFrames in the light called "Omni01" to green.

3.17.1.8. GetLightSpecularColor
Syntax: GetLightSpecularColor()
The GetLightSpecularColor method return the SpecularColor of the first light source in the selection.
Example: `doTNTCommand('Light("Omni01").GetLightSpecularColor()');`
This example returns the SpecularColor of the light called "Omni01".

3.17.1.9. SetLightAmbientColor
Syntax: SetLightAmbientColor(color:integer)
The SetLightAmbientColor method sets the AmbientColor of one or more LightFrames in the selection. The LightAmbientColor is the ambient color illuminated by the light-source.
Example: `doTNTCommand('Light("Omni01").SetLightAmbientColor(#0000FF)');`
This example sets the AmbientColor of all LightFrames in the light called "Omni01" to blue.

3.17.1.10. GetLightAmbientColor
Syntax: GetLightAmbientColor()
The GetLightAmbientColor method returns the AmbientColor of the first light source in the selection.
Example: `doTNTCommand('Light("Omni01").GetLightAmbientColor()');`
This example returns the AmbientColor from the light called "Omni01".

3.18.Mesh

3.18.1.1. SetMeshCollision
Syntax: SetMeshCollision(enable:integer)
The SetMeshCollision method sets the Collision value of one or more meshes in the selection. If Collision is enable the physics object will collide with this object. The ground and walls are good examples of objects, which should have collision enabled.
Example: `doTNTCommand('Mesh("Floor").SetMeshCollision(1)');`
This example enables Collision for the mesh called "Floor".

3.18.1.2. GetMeshCollision
Syntax: GetMeshCollision()
The GetMeshCollision method returns the Collision value from the first Mesh in the

selection.
Example: `doTNTCommand('Mesh("Floor01").GetMeshCollision()');`
This example returns the Collision value of the mesh called "Floor01".

3.18.1.3. SetMeshOcclusion
Syntax: SetMeshOcclusion(enable:integer)
The SetMeshOcclusion method sets the Occlusion value of one or more meshes in the selection. When occlusion for an object is enabled, each polygon of the object will be checked for intersection with a ray that has the position of the mouse-pointer as starting point. This process is processor time expensive, so only give objects the Occlusion flag when it is absolute necessary.
Occlusion is a part of the Mouse Over Event system in TurnTool.
Objects that need to generate the mouse-over event must have the Occlusion flag as well. When an object only has the Occlusion enabled and not the MouseOver Event the the object can cover or occlude other objects so the "Mouse Over Event" only will be generated when looking at the object from a certain angle or position.
Example: `doTNTCommand('Mesh("Teapot01").SetMeshOcclusion(1)');`
This example enables Occlusion for the mesh called "Teapot01".

3.18.1.4. GetMeshOcclusion
Syntax: GetMeshOcclusion()
The GetMeshOcclusion method returns the Occlusion value from the first mesh in the selection.
Example: `doTNTCommand('Mesh("Teapot01").GetMeshOcclusion()');`
This example returns the Occlusion value from the mesh called "Teapot01".

3.18.1.5. SetMeshMouseOverEvent
Syntax: SetMeshMouseOverEvent(enable:integer)
The SetMeshMouseOverEvent method sets the MouseOverEvent value of one or more meshes in the selection. Enabling the MouseOverEvent value only has an effect if the Occlusion value is enabled too. Objects that have the MouseOverEvent will generate a JavaScript event called "OnMouseEnter" when the mouse pointer is over the object and another JavaScript event called "OnMouseExit" when the mouse pointer leaves the object.
Example: `doTNTCommand('Mesh("Teapot01").SetMeshMouseOverEvent(1)');`
This example enables the MouseOverEvent for the mesh called "Teapot01".

3.18.1.6. GetMeshMouseOverEvent
Syntax: GetMeshMouseOverEvent()
The GetMeshMouseOverEvent method returns the MouseOverEvent value from the first mesh in the selection.
Example: `doTNTCommand('Mesh("Torus05").GetMeshMouseOverEvent()');`
This example returns the MouseOverEvent for the mesh called Torus05.

3.18.1.7. SetMeshValue
Syntax: SetMeshValue(value:integer)
The SetMeshValue method sets the integer MeshValue of one or more meshes in the selection. The concept of a MeshValue is simply a integer value in each of the Mesh in

the scene that you can use to store what ever you want. The MeshValue for each Mesh is also saved when saving the scene as a TNT file.
Example: `doTNTCommand('Mesh("MyMesh").SetMeshValue(37)');`
This example sets the MeshValue of the mesh called "MyMesh" to 37.

3.18.1.8. GetMeshValue
Syntax: GetMeshValue()
The GetMeshValue method returns the integer MeshValue from the first mesh in the selection.
Example: `doTNTCommand('Mesh("MyMesh").GetMeshValue()');`
This example returns the integer MeshValue of the mesh called "MyMesh".

3.18.1.9. SetMeshLightMode
Syntax: SetMeshLightMode(integer)
The SetMeshLightMode method sets the LightMode value of one or more meshes in the selection.
LightMode = 1 means "Dynamic Lighting". Lighting is calculated dynamically when viewing the scene. This allows lights and objects to move relative to each other.
LightMode = 2 means "Static Lighting". Lighting is calculated once when the scene loads. This has the benefit of much better performance. Drawbacks are slightly longer loading time and lighting may look wrong if an object is moved. It is ideal for large static scenes. No vertex colors are exported because the calculated light would replace them.
LightMode = 3 means "Vertex Colors". TurnTool performs no lighting calculations but exports a color for each vertex instead. This can be used for pre-generating more advanced static lighting vertex color "baking".
LightMode = 4 means "Baked Textures (No Lighting)". TurnTool performs no lighting calculations. This is useful when using textures "baked" with pre-calculated lighting.
LightMode = 0 is the same as LightMode = 1.
Example: `doTNTCommand('Mesh("Mesh54").SetMeshLightMode(1)');`
This example sets the lightmode of the mesh called "Mesh54" to Dynamic Lighting.

3.18.1.10. GetMeshLightMode
Syntax: GetMeshLightMode()
The GetMeshLightMode method returns the LightMode value from the first mesh in the selection. See SetMeshLightMode for more details.
Example: `doTNTCommand('Mesh("Mesh54").GetMeshLightMode()');`
This example returns the lightmode of the mesh called "Mesh54".

3.18.1.11. SetMeshRenderPass
Syntax: SetMeshRenderPass(integer)
The SetMeshRenderPass method sets the RenderPass value of one or more meshes in the selection. The RenderPass indicates in which order objects should be rendered. A lower numbered object is rendered before a higher numbered one. The range of the RenderPass value range is +32767 to -32767.
Example: `doTNTCommand('Mesh("LogoMesh").SetMeshRenderPass(5)');`

This example sets the RenderPass value of the mesh called "LogoMesh" to 5.

3.18.1.12. GetMeshRenderPass

Syntax: GetMeshRenderPass()

The GetMeshRenderPass method returns the RenderPass value from the first mesh in the selection.

Example: `doTNTCommand('Mesh("LogoMesh").GetMeshRenderPass()');`

Explain example text.

3.18.1.13. SetMeshFaceSort

Syntax: SetMeshFaceSort(integer)

The SetMeshFaceSort method sets the FaceSort value of one or more meshes in the selection.When rendering, objects with transparency are sorted by their pivot points and rendered back to front. For objects with a lot of overlapping transparent faces, this may not look right. This can be corrected by enabling FaceSort. This means the object's faces are also sorted by depth and rendered back to front. This face sort process can be very processor time expensive, so only enable SortFace on objects when it is absolute necessary.

Example: `doTNTCommand('Mesh("TorusTwist").SetMeshFaceSort(1)');`

This example enables FaceSort for the mesh called "TorusTwist".

3.18.1.14. GetMeshFaceSort

Syntax: GetMeshFaceSort()

The GetMeshFaceSort method returns the FaceSort value from the first mesh in the selection.

Example: `doTNTCommand('Mesh("TorusTwist").GetMeshFaceSort()');`

This example returns the FaceSort value of the mesh called "TorusTwist".

3.18.1.15. SetMeshLookAtU

Syntax: SetMeshLookAtU(integer)

The SetMeshLookAtU method sets the LookAtU value of one or more meshes in the selection. If the LookAtU value is a non zero value then the LookAt process will allow U rotation (in XY plane).

Example: `doTNTCommand('Mesh("Billboard").SetMeshLookAtU(1)');`

This example enables LookAtU for the mesh called "Billboard".

3.18.1.16. GetMeshLookAtU

Syntax: GetMeshLookAtU()

The GetMeshLookAtU method returns the LookAtU value from the first mesh in the selection.

Example: `doTNTCommand('Mesh("Billboard").GetMeshLookAtU()');`

This example returns the MeshLookAtU value of the mesh called "Billboard".

3.18.1.17. SetMeshLookAtV

Syntax: SetMeshLookAtV(integer)

The SetMeshLookAtV method sets the LookAtV value of one or more meshes in the selection. If the LookAtV value is a non zero value then the LookAt process will allow

V rotation (in YZ plane).
Example: `doTNTCommand('Mesh("Billboard").SetMeshLookAtV(1)');`
This example enables LookAtV for the mesh called "Billboard".

3.18.1.18. GetMeshLookAtV
Syntax: GetMeshLookAtV()
The GetMeshLookAtV method returns the LookAtV value from the first mesh in the selection.
Example: `doTNTCommand('Mesh("Billboard").GetMeshLookAtV()');`
This example returns the MeshLookAtV value of the mesh called "Billboard".

3.18.1.19. SetMeshUAngleImageCount
Syntax: SetMeshUAngleImageCount(integer)
The SetMeshUAngleImageCount method sets the UAngleImageCount value of one or more meshes in the selection. The MeshUAngleImageCount value can a non zero value if the object has a sequence of different textures defined on the timeline as an animated texture. The U Count parameter define how many different angle images that exist for this object. If the U Count parameter is set to a value higher than 1 then the Angle Image process will be enabled for this object. To determine which angle image that will be shown the x and y position of the camera and the object used to calculate the angle. If U Count is set to 32 then there will be 360 / 32 = 11.25 degrees between each image. In other words: when the angle between the camera and the object is in the range 0.0 to 11.25 then the first image will be shown. If the angle is between 11.25 and 22.50 then the second image will be shown and so on. This technique of changing the texture of an object according the angle between the object and the camea is sometimes refered to as RPC.
Example: `doTNTCommand('Mesh("Billboard").SetMeshUAngleImageCount(32)');`
This example sets the MeshUAngleImageCount of the mesh called "Billboard" to 32.

3.18.1.20. GetMeshUAngleImageCount
Syntax: GetMeshUAngleImageCount()
The GetMeshUAngleImageCount method returns the UAngleImageCount value from the first mesh in the selection.
Example: `doTNTCommand('Mesh("Billboard").GetMeshUAngleImageCount()');`
This example returns the MeshUAngleImageCount of the mesh called "Billboard".

3.18.1.21. SetMeshDepthTesting
Syntax: SetMeshDepthTesting(integer)
The SetMeshDepthTesting method sets the DepthTesting value of one or more meshes in the selection. The default value of the DepthTesting value is 1.
When an object is rendering on the screen the depth of each visible pixel is tested against the depth buffer (Z buffer) to determine whether or not the pixel is in front of the pixel that already has been rendered. A new pixel will only be drawn on the screen, if this pixel is in front of the pixel that already exist at this position on the screen. This process is the normal behavior when rendering each pixel of an object. If you wish all pixels of the object to be rendered regardless of the depth then you can disable the

DepthTest by settings the DepthTesting to zero.
Example: `doTNTCommand('Mesh("Menu").SetMeshDepthTesting(0)');`
This example disables DepthTesting for the mesh called "Menu".

3.18.1.22. GetMeshDepthTesting
Syntax: GetMeshDepthTesting()
The GetMeshDepthTesting method returns the DepthTesting value from the first mesh in the selection.
Example: `doTNTCommand('Mesh("Menu").GetMeshDepthTesting()');`
This example return the DepthTesting value of the mesh called "Menu".

3.18.1.23. SetMeshIs2D
Syntax: SetMeshIs2D(integer)
The SetMeshIs2D method sets the Is2D value of one or more meshes in the selection. When Is2D vale is '1' then object will be rendered as if it was a 2D object. An object having a Is2D value of 1 must be inside a worldspace bounding box with these coordinates lower left corner (-1,-1,-1) Upper right corner (1,1,1). Object with a Is2D value of 1 will always be at the same position on the screen relative to the size of the screen. Setting a mesh to have a Is2D value of 1 is very useful when making menu systems inside TurnTool.
Example: `doTNTCommand('Mesh("My2DObj").SetMeshIs2D(1)');`
This example sets the Is2D value of the mesh called "My2DObj" to 1.

3.18.1.24. GetMeshIs2D
Syntax: GetMeshIs2D()
The GetMeshIs2D method returns the Is2D value from the first mesh in the selection.
Example: `doTNTCommand('Mesh("My2DObj").GetMeshIs2D()');`
This example returns the Is2D value of the mesh called "My2DObj".

3.18.1.25. GetMeshSphereRadius
Syntax: GetMeshSphereRadius()
The GetMeshSphereRadius method will return the radius of the sphere primitive. This method will only give a result if the mesh in the selection is a sphere.
Example: `doTNTCommand('Mesh("Sphere01").GetMeshSphereRadius()');`
This example returns the radius of the mesh-sphere called "Sphere01".

3.18.1.26. GetLocalGeometryMinX
Syntax: GetLocalGeometryMinX()
The GetLocalGeometryMinX method returns the minimum X value of a axis aligned bounding box calculated in local space of the first mesh in the selection.
Example: `doTNTCommand('Node('AABB').GetLocalGeometryMinX()');`
This example returns the LocalGeometryMinX value of the node called "AABB".

3.18.1.27. GetLocalGeometryMaxX
Syntax: GetLocalGeometryMaxX()
The GetLocalGeometryMinX method returns the maximum X value of a axis aligned bounding box calculated in local space of the first mesh in the selection.

Example: `doTNTCommand('Node('AABB').GetLocalGeometryMaxX()');`
This example returns the LocalGeometryMaxX value of the node called "AABB".

3.18.1.28. GetLocalGeometryMinY
Syntax: GetLocalGeometryMinY()
The GetLocalGeometryMinY method returns the minimum Y value of a axis aligned bounding box calculated in local space of the first mesh in the selection.
Example: `doTNTCommand('Node('AABB').GetLocalGeometryMinY()');`
This example returns the LocalGeometryMinY value of the node called "AABB".

3.18.1.29. GetLocalGeometryMaxY
Syntax: GetLocalGeometryMaxY()
The GetLocalGeometryMaxY method returns the maximum Y value of a axis aligned bounding box calculated in local space of the first mesh in the selection.
Example: `doTNTCommand('Node('AABB').GetLocalGeometryMaxY()');`
This example returns the LocalGeometryMaxY value of the node called "AABB".

3.18.1.30. GetLocalGeometryMinZ
Syntax: GetLocalGeometryMinZ()
The GetLocalGeometryMinZ method returns the minimum Z value of a axis aligned bounding box calculated in local space of the first mesh in the selection.
Example: `doTNTCommand('Node('AABB').GetLocalGeometryMinZ()');`
This example returns the LocalGeometryMinZ value of the node called "AABB".

3.18.1.31. GetLocalGeometryMaxZ
Syntax: GetLocalGeometryMaxZ()
The GetLocalGeometryMaxZ method returns the maximum Z value of a axis aligned bounding box calculated in local space of the first mesh in the selection.
Example: `doTNTCommand('Node('CSABB').GetLocalGeometryMaxZ()');`
This example returns the LocalGeometryMaxZ value of the node called "AABB".

3.18.1.32. GetWorldGeometryMinX
Syntax: GetWorldGeometryMinX()
The GetWorldGeometryMinX method returns the minimum X value of a axis aligned bounding box calculated in world space of the first mesh in the selection.
Example: `doTNTCommand('Node('CSABB').GetWorldGeometryMinX');`
This example returns the WorldGeometryMinX value of the node called "AABB".

3.18.1.33. GetWorldGeometryMaxX
Syntax: GetWorldGeometryMaxX()
The GetWorldGeometryMaxX method returns the maximum X value of a axis aligned bounding box calculated in world space of the first mesh in the selection.
Example: `doTNTCommand('Node('CSABB').GetWorldGeometryMaxX');`
This example returns the WorldGeometryMaxX value of the node called "AABB".

3.18.1.34. GetWorldGeometryMinY
Syntax: GetWorldGeometryMinY()
The GetWorldGeometryMinY method returns the minimum Y value of a axis aligned bounding box calculated in world space of the first mesh in the selection.
Example: `doTNTCommand('Node('CSABB').GetWorldGeometryMinY');`
This example returns the WorldGeometryMinY value of the node called "AABB".

3.18.1.35. GetWorldGeometryMaxY
Syntax: GetWorldGeometryMaxY()
The GetWorldGeometryMaxY method returns the maximum Y value of a axis aligned bounding box calculated in world space of the first mesh in the selection.
Example: `doTNTCommand('Node('CSABB').GetWorldGeometryMaxY');`
This example returns the WorldGeometryMaxY value of the node called "AABB".

3.18.1.36. GetWorldGeometryMinZ
Syntax: GetWorldGeometryMinZ()
The GetWorldGeometryMinZ method returns the minimum Z value of a axis aligned bounding box calculated in world space of the first mesh in the selection.
Example: `doTNTCommand('Node('CSABB').GetWorldGeometryMinZ');`
This example returns the WorldGeometryMinZ value of the node called "AABB".

3.18.1.37. GetWorldGeometryMaxZ
Syntax: GetWorldGeometryMaxZ()
The GetWorldGeometryMaxZ method returns the maximum Z value of a axis aligned bounding box calculated in world space of the first mesh in the selection.
Example: `doTNTCommand('Node('CSABB').GetWorldGeometryMaxZ');`
This example returns the WorldGeometryMaxZ value of the node called "AABB".

3.18.1.38. GetMeshFrameCount
Syntax: GetMeshFrameCount()
The GetMeshFrameCount method will return the number of MesheFrames of the first Mesh in the selection.
Example: `doTNTCommand('Mesh("Mesh423").GetMeshFrameCount()');`
This example returns the number of MeshFrames of the mesh called "Mesh423".

3.18.1.39. CreateMeshFrame
Syntax: CreateMeshFrame(count:integer)
The CreateMeshFrame method creates MeshFrames on one or more Meshes in the selection. The count argument is the number of MeshFrames that will be created on the mesh(es).
Example: `doTNTCommand('Mesh("Mesh423").CreateMeshFrame(100)');`
This example creates 100 MeshFrames on the mesh called "Mesh423".

3.18.1.40. GenerateUVBoxCoords
Syntax: GenerateUVBoxCoords(x:float,y:float,z:float)
The GenerateUVBoxCoords method generates UV texture coordinates using box

mapping on one or more meshes in the selection. The arguments x,y,z are tiling factors for the texture in the x,y,z direction.
Example: `doTNTCommand('Mesh("Box01").GenerateUVBoxCoords(1.0,2.0,3.0)');`
This example generate UV texture coordinates for the mesh called "Box01".

3.19.Physics

3.19.1.1. SetPhysicsCurrent
Syntax: SetPhysicsCurrent(value:float)
The SetPhysicsCurrent method will set the mesh in the selection to be the current Physics. The current physics is the only object for which physics will be calculated.
Example: `doTNTCommand('Mesh("PhysicsObject").SetPhysicsCurrent()');`
This example will set the mesh called "PhysicsObject" to be the current physics.

3.19.1.2. GetPhysicsCurrent
Syntax: GetPhysicsCurrent()
The GetPhysicsCurrent method will return the value '1' if the first mesh in the selection is the current physics. If the mesh in the selection is not the current physics then '0' will be returned.
Example: `doTNTCommand('Mesh("PhysicsObject").GetPhysicsCurrent()');`
This example will return a value define whether the mesh called "PhysicsObject" is the current physics.

3.19.1.3. GetPhysicsGravityForce
Syntax: GetPhysicsGravityForce()
The GetPhysicsGravityForce method returns the GravityForce of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsGravityForce()');`
This example returns the GravityForce of the current physics.

3.19.1.4. SetPhysicsGravityForce
Syntax: SetPhysicsGravityForce(value:float)
The SetPhysicsGravityForce method sets the GravityForce of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsGravityForce(100.0)');`
This example sets the GravityForce of the current physics to 100.

3.19.1.5. GetPhysicsMoveSpeed
Syntax: GetPhysicsMoveSpeed()
The GetPhysicsMoveSpeed method returns the MoveSpeed of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.Example: `doTNTCommand('CurrentPhysics().GetPhysicsMoveSpeed()');`
This example returns the MoveSpeed of the current physics.

3.19.1.6. SetPhysicsMoveSpeed
Syntax: SetPhysicsMoveSpeed(value:float)

The SetPhysicsMoveSpeed method sets the MoveSpeed of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsMoveSpeed(3879.69)');`
This example sets the MoveSpeed of the current physics to 3879.69.

3.19.1.7. GetPhysicsFriction
Syntax: GetPhysicsFriction()
The GetPhysicsFriction method returns the Friction of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsFriction()');`
This example returns the Friction of the current physics.

3.19.1.8. SetPhysicsFriction
Syntax: SetPhysicsFriction(value:float)
The SetPhysicsFriction method sets the Friction of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsFriction(0.0502)');`
This example sets the Friction of the current physics to 0.0502.

3.19.1.9. GetPhysicsMass
Syntax: GetPhysicsMass()
The GetPhysicsMass method returns the Mass of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsMass()');`
This example returns the Mass of the current physics.

3.19.1.10. SetPhysicsMass
Syntax: SetPhysicsMass(value:float)
The SetPhysicsMass method sets the Mass of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsMass(1.0)');`
This example sets the Mass of the current physics to 1.0.

3.19.1.11. GetPhysicsBounce
Syntax: GetPhysicsBounce()
The GetPhysicsBounce method returns the Bounce of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsBounce()');`
This example returns the Bounce of the current physics.

3.19.1.12. SetPhysicsBounce
Syntax: SetPhysicsBounce(value:float)

The SetPhysicsBounce method sets the Bounce of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsBounce(0.029)');`
This example sets the Bounce of the current physics to 0.029.

3.19.1.13. GetPhysicsAirResistance
Syntax: GetPhysicsAirResistance()
The GetPhysicsAirResistance method returns the AirResistance of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsAirResistance()');`
This example returns the AirResistance of the current physics.

3.19.1.14. SetPhysicsAirResistance
Syntax: SetPhysicsAirResistance(value:float)
The SetPhysicsAirResistance method sets the AirResistance of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsAirResistance(0.5)');`
This example sets the AirResistance of the current physics to 0.5.

3.19.1.15. GetPhysicsGripThreshold
Syntax: GetPhysicsGripThreshold()
The GetPhysicsGravityGripThreshold method returns the GripThreshold of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsGripThreshold()');`
This example returns the GripThreshold of the current physics.

3.19.1.16. SetPhysicsGripThreshold
Syntax: SetPhysicsGripThreshold(value:float)
The SetPhysicsGripThreshold method sets the GripThreshold of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsGripThreshold(25.0)');`
This example sets the GripThreshold of the current physics to 25.0.

3.19.1.17. GetPhysicsGripFaceAngle
Syntax: GetPhysicsGripFaceAngle()
The GetPhysicsGripFaceAngle method returns the GripFaceAngle of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsGripFaceAngle()');`
This example returns the GripFaceAngle of the current physics.

3.19.1.18. SetPhysicsGripFaceAngle
Syntax: SetPhysicsGripFaceAngle(value:float)
The SetPhysicsGripFaceAngle method sets the GripFaceAngle of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsGripFaceAngle(-0.684)');`
This example sets the GripFaceAngle of the current physics to -0.684.

3.19.1.19. GetPhysicsAngleToFace
Syntax: GetPhysicsAngleToFace()
The GetPhysicsAngleToFace method returns the AngleToFace of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsAngleToFace()');`
This example returns the AngleToFace of the current physics.

3.19.1.20. SetPhysicsAngleToFace
Syntax: SetPhysicsAngleToFace(value:float)
The SetPhysicsAngleToFace method sets the AngleToFace of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsAngleToFace(0.0841)');`
This example sets the AngleToFace of the current physics to 0.0841.

3.19.1.21. GetPhysicsAngleToGrip
Syntax: GetPhysicsAngleToGrip()
The GetPhysicsAngleToGrip method returns the AngleToGrip of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsAngleToGrip()');`
This example returns the AngleToGrip of the current physics.

3.19.1.22. SetPhysicsAngleToGrip
Syntax: SetPhysicsAngleToGrip(value:float)
The SetPhysicsAngleToGrip method sets the AngleToGrip of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsAngleToGrip(1.0)');`
This example sets the AngleToGrip of the current physics to 1.0.

3.19.1.23. GetPhysicsMoveFree
Syntax: GetPhysicsMoveFree()
The GetPhysicsMoveFree method returns the MoveFree of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsMoveFree()');`

This example returns the MoveFree of the current physics.

3.19.1.24. SetPhysicsMoveFree
Syntax: SetPhysicsMoveFree(value:integer)
The SetPhysicsMoveFree method sets the MoveFree of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsMoveFree(1)');`
This example sets the MoveFree of the current physics to 1.

3.19.1.25. GetPhysicsJumpUpForce
Syntax: GetPhysicsJumpUpForce()
The GetPhysicsJumpUpForce method returns the JumpUpForce of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsJumpUpForce()');`
This example returns the JumpToForce of the current physics.

3.19.1.26. SetPhysicsJumpUpForce
Syntax: SetPhysicsJumpUpForce(value:float)
The SetPhysicsJumpUpForce method sets the JumpUpForce of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsJumpUpForce(350.77)');`
This example sets the JumpUpForce of the current physics to 350.77.

3.19.1.27. GetPhysicsJumpScalar
Syntax: GetPhysicsJumpScalar()
The GetPhysicsJumpScalar method returns the JumpScalar of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsJumpScalar()');`
This example returns the JumpScalar of the current physics.

3.19.1.28. SetPhysicsJumpScalar
Syntax: SetPhysicsJumpScalar(value:float)
The SetPhysicsJumpScalar method sets the JumpScalar of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsJumpScalar(60.7)');`
This example sets the JumpScalar of the current physics to 60.7.

3.19.1.29. GetPhysicsAirControlFactor
Syntax: GetPhysicsAirControlFactor()
The GetPhysicsAirControlFactor method returns the AirControlFactor of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.

Example: `doTNTCommand('CurrentPhysics().GetPhysicsAirControlFactor()');`
This example returns the AirControlFactor of the current physics.

3.19.1.30. SetPhysicsAirControlFactor
Syntax: SetPhysicsAirControlFactor(value:float)
The SetPhysicsAirControlFactor method sets the AirControlFactor of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsAirControlFactor(0.0936)');`
This example sets the AirControlFactor of the current physics to 0.0936.

3.19.1.31. GetPhysicsJumpEnableAngle
Syntax: GetPhysicsJumpEnableAngle()
The GetPhysicsJumpEnableAngle method returns the JumpEnableAngle of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsJumpEnableAngle()');`
This example returns the JumpEnableAngle of the current physics.

3.19.1.32. SetPhysicsJumpEnableAngle
Syntax: SetPhysicsJumpEnableAngle(value:float)
The SetPhysicsJumpEnableAngle method sets the JumpEnableAngle of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsJumpEnableAngle(-0.684)');`
This example sets the JumpEnableAngle of the current physics to -0.684.

3.19.1.33. GetPhysicsMoveEnableAngle
Syntax: GetPhysicsMoveEnableAngle()
The GetPhysicsMoveEnableAngle method returns the MoveEnableAngle of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsMoveEnableAngle()');`
This example returns the MoveEnableAngle of the current physics.

3.19.1.34. SetPhysicsMoveEnableAngle
Syntax: SetPhysicsMoveEnableAngle(value:float)
The SetPhysicsMoveEnableAngle method sets the MoveEnableAngle of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsMoveEnableAngle(-0.56)');`
This example sets the MoveEnableAngle of the current physics to -0.56.

3.19.1.35. GetPhysicsBorderSize
Syntax: GetPhysicsBorderSize()
The GetPhysicsBorderSize method returns the BorderSize of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsBorderSize()');`
This example returns the BorderSize of the current physics.

3.19.1.36. SetPhysicsBorderSize
Syntax: SetPhysicsBorderSize(value:float)
The SetPhysicsBorderSize method sets the BorderSize of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsBorderSize(0.1)');`
This example sets the BorderSize of the current physics to 0.1.

3.19.1.37. GetPhysicsCacheFactor
Syntax: GetPhysicsCacheFactor()
The GetPhysicsCacheFactor method returns the CacheFactor of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsCacheFactor()');`
This example returns the CacheFactor of the current physics.

3.19.1.38. SetPhysicsCacheFactor
Syntax: SetPhysicsCacheFactor(value:float)
The SetPhysicsCacheFactor method sets the CacheFactor of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsCacheFactor(2.0)');`
This example sets the CacheFactor of the current physics to 2.0.

3.19.1.39. GetPhysicsRotateMaxSpeed
Syntax: GetPhysicsRotateMaxSpeed()
The GetPhysicsRotateMaxSpeed method returns the RotateMaxSpeed of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsRotateMaxSpeed()');`
This example returns the RotateMaxSpeed of the current physics.

3.19.1.40. SetPhysicsRotateMaxSpeed
Syntax: SetPhysicsRotateMaxSpeed(value:float)
The SetPhysicsRotateMaxSpeed method sets the RotateMaxSpeed of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsRotateMaxSpeed(0.003)');`

This example sets the RotateMaxSpeed of the current physics to 0.003.

3.19.1.41. GetPhysicsRotateQuadratic
Syntax: GetPhysicsRotateQuadratic()
The GetPhysicsRotateQuadratic method returns the RotateQuadratic of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsRotateQuadratic()');`
This example returns the RotateQuadratic of the current physics.

3.19.1.42. SetPhysicsRotateQuadratic
Syntax: SetPhysicsRotateQuadratic(value:float)
The SetPhysicsRotateQuadratic method sets the RotateQuadratic of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsRotateQuadratic(0.05)');`
This example sets the RotateQuadratic of the current physics to 0.05.

3.19.1.43. GetPhysicsRotateLinear
Syntax: GetPhysicsRotateLinear()
The GetPhysicsRotateLinear method returns the RotateLinear of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsRotateLinear()');`
This example returns the RotateLinear of the current physics.

3.19.1.44. SetPhysicsRotateLinear
Syntax: SetPhysicsRotateLinear(value:float)
The SetPhysicsRotateLinear method sets the RotateLinear of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsRotateLinear(0.13)');`
This example sets the RotateLinear of the current physics to 0.13.

3.19.1.45. GetPhysicsRotateDamping
Syntax: GetPhysicsRotateDamping()
The GetPhysicsRotateDamping method returns the RotateDamping of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsRotateDamping()');`
This example returns the RotateDamping of the current physics.

3.19.1.46. SetPhysicsRotateDamping
Syntax: SetPhysicsRotateDamping(value:float)
The SetPhysicsRotateDamping method sets the RotateDamping of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.

Example: `doTNTCommand('CurrentPhysics().SetPhysicsRotateDamping(0.975)');`
This example sets the RotateDamping of the current physics to 0.975.

3.19.1.47. GetPhysicsRotXDeltaMouse
Syntax: GetPhysicsRotXDeltaMouse()
The GetPhysicsRotXDeltaMouse method returns the RotXDeltaMouse of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsRotXDeltaMouse()');`
This example returns the RotXDeltaMouse of the current physics.

3.19.1.48. SetPhysicsRotXDeltaMouse
Syntax: SetPhysicsRotXDeltaMouse(value:float)
The SetPhysicsRotXDeltaMouse method sets the RotXDeltaMouse of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsRotYDeltaMouse(0.007)');`
This example sets the RotXDeltaMouse of the current physics to 0.007.

3.19.1.49. GetPhysicsRotYDeltaMouse
Syntax: GetPhysicsRotYDeltaMouse()
The GetPhysicsRotYDeltaMouse method returns the RotYDeltaMouse of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsRotYDeltaMouse()');`
This example returns the RotYDeltaMouse of the current physics.

3.19.1.50. SetPhysicsRotYDeltaMouse
Syntax: SetPhysicsRotYDeltaMouse(value:float)
The SetPhysicsRotYDeltaMouse method sets the RotYDeltaMouse of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsRotYDeltaMouse(0.007)');`
This example sets the RotYDeltaMouse of the current physics to 0.007.

3.19.1.51. GetPhysicsJumpDelay
Syntax: GetPhysicsJumpDelay()
The GetPhysicsJumpDelay method returns the JumpDelay of the first mesh in the selection. This method will only return a result if the mesh is a Physics Sphere.
Example: `doTNTCommand('CurrentPhysics().GetPhysicsJumpDelay()');`
This example returns the JumpDelay of the current physics.

3.19.1.52. SetPhysicsJumpDelay
Syntax: SetPhysicsJumpDelay(value:integer)
The SetPhysicsJumpDelay method sets the JumpDelay of one or more Meshes in the selection. This method will only have an effect on the meshes in the selection that are physics spheres. Read the section about the Physics panel in the TurnToolBox manual

for further details.
Example: `doTNTCommand('CurrentPhysics().SetPhysicsJumpDelay(500)');`
This example sets the JumpDelay of the current physics to 500.

3.20.MeshFrame

3.20.1.1. GetFaceSegmentCount
Syntax: GetFaceSegmentCount()

The GetFaceSegmentCount method will returns the number of FaceSegments of the first MeshFrame in the selection.
Example: `doTNTCommand('Mesh("TestMesh").MeshFrame().GetFaceSegmentCount()');`
This example returns the number of FaceSegments of the first MeshFrame of the mesh called "TestMesh".

3.20.1.2. CreateFaceSegment
Syntax: CreateFaceSegment(count:integer)

The CreateFaceSegment method creates a FaceSegment on one or more MeshFrames in the selection.
Example: `doTNTCommand('Mesh("TestMesh").MeshFrame().CreateFaceSegment(1)');`
This example creates one FaceSegment on all MeshFrames of the mesh called "TestMesh".

3.21.FaceSegment

3.21.1.1. GetFaceCount
Syntax: GetFaceCount()

The GetFaceCount method will returns the FaceCount of the first FaceSegment in the selection.
Example:
`doTNTCommand('Mesh("TestMesh").MeshFrame().FaceSegment().GetFaceCount()');`
This example returns the face count of the first FaceSegment of the first MeshFrame of the mesh called "TestMesh".

3.21.1.2. AppendFace
Syntax: AppendFace(index0:integer, index1:integer, index2:integer)

The AppendFace method appends a face to one or more FaceSegments in the selection. The arguments index0, index1 and index2 are the indices into the position vertices which then define a triangle. All indices must be lower than the value returned from the GetPositionCount() method for the same FaceSegment.
Example:
`doTNTCommand('Mesh("TestMesh").MeshFrame().FaceSegment().AppendFace(0,1,2)');`
This example appends a Face to all the FaceSegments of all the MeshFrames of the mesh called "TestMesh".

3.21.1.3. GetColorCount
Syntax: GetColorCount()

The GetColorCount method returns the number of Colors of the first FaceSegment in the selection.
Example:
`doTNTCommand('Mesh("TestMesh").MeshFrame().FaceSegment().GetColorCount()');`

This example returns the number of colors in the first FaceSegment of the first MeshFrame of the mesh called "TestMesh".

3.21.1.4. AppendColor

Syntax: AppendColor(color:integer)

The AppendColor method appends a color to one or more FaceSegments in the selection The color argument is the actual color that will be appended to the color array.
Example:

```
doTNTCommand('Mesh("TestMesh").MeshFrame().FaceSegment().AppendColor(#00FF00)'
);
```

This example appends a color on all the FaceSegments of all the MeshFrames of the mesh called "TestMesh".

3.21.1.5. GetVertexSegmentCount

Syntax: GetVertexSegmentCount()

The GetVertexSegmentCount method returns the number of VertexSegments of the first FaceSegment in the selection.
Example:

```
doTNTCommand('Mesh("Box01").MeshFrame().FaceSegment().GetVertexSegmentCount()'
);
```

This example returns the VertexSegment count of all the FaceSegment of all the MeshFrames of the mesh called "Box01".

3.21.1.6. CreateVertexSegment

Syntax: CreateVertexSegment(count:integer)

The CreateVertexSegment method will create one or more VertexSegments on one or more FaceSegments in the selection. The count argument is the the number of VertexSegments that will be created.
Example:

```
doTNTCommand('Mesh("Box01").MeshFrame().FaceSegment().CreateVertexSegment(1)')
;
```

This example creates one VertexSegment of the all the FaceSegments of all the MeshFrames of the mesh called "Box01".

3.21.1.7. GetTextureFragmentCount

Syntax: GetTextureFragmentCount()

The GetTextureFragmentCount method returns the number of TextureFragments of the first FaceSegment in the selection.
Example:

```
doTNTCommand('Mesh("Box01").MeshFrame().FaceSegment().GetTextureFragmentCount(
)');
```

This example returns the number of TextureFragments of the first FaceSegment of the first MeshFrame of the mesh called "Box01".

3.21.1.8. CreateTextureFragment

Syntax: CreateTextureFragment(count:integer)

The CreateTextureFragment method creates a TextureFragment on one or more FaceSegment in the selection.
Example:

```
doTNTCommand('Mesh("Box01").MeshFrame().FaceSegment().CreateTextureFragment(1)
```

```
');
```
This example creates one TextureFragment on all the FaceSegments of all the MeshFrames of the mesh called "Box01".

3.21.1.9. GetSrcBlend
Syntax: GetSrcBlend()
The GetSrcBlend method returns the SrcBlend value of the first FaceSegment in the selection.
Example:
```
doTNTCommand('Mesh("Box01").MeshFrame().FaceSegment().GetSrcBlend()');
```
This example returns the SrcBlend value of the first FaceSegment of the first MeshFrame of the mesh called "Box01".

3.21.1.10. SetSrcBlend
Syntax: SetSrcBlend(blendmode:integer)
The SetSrcBlend method sets the SrcBlend value of one or more FaceSegment in the selection.
Example:
```
doTNTCommand('Mesh("Box01").MeshFrame().FaceSegment().SetSrcBlend(2)');
```
This example sets the SrcBlend to 2 of all the FaceSegment of all the MeshFrames of the mesh called "Box01.

3.21.1.11. GetDestBlend
Syntax: GetDestBlend()
The GetDestBlend method returns the DestBlend value of the first FaceSegment in the selection.
Example:
```
doTNTCommand('Mesh("Box01").MeshFrame().FaceSegment().GetDestBlend()');
```
This example returns the DestBlend value of the first FaceSegment of the first MeshFrame of the mesh called "Box01".

3.21.1.12. SetDestBlend
Syntax: SetDestBlend(blendmode:integer)
The SetDestBlend method sets the DestBlend value of one or more FaceSegment in the selection.
Example:
```
doTNTCommand('Mesh("Box01").MeshFrame().FaceSegment().SetDestBlend(2)');
```
This example sets the DestBlend to 2 of all the FaceSegment of all the MeshFrames of the mesh called "Box01.

3.21.1.13. GetColorOperation
Syntax: GetColorOperation()
The GetColorOperation method returns the ColorOperation value of the first FaceSegment in the selection.
Example:
```
doTNTCommand('Mesh("Box01").MeshFrame().FaceSegment().GetColorOperation()');
```
This example returns the ColorOperation value of the first FaceSegment of the first MeshFrame of the mesh called "Box01".

3.21.1.14. SetColorOperation

Syntax: SetColorOperation(colorop:integer)

The SetColorOperation method sets the ColorOperation value of one or more FaceSegment in the selection.

Example:

```
doTNTCommand('Mesh("Box01").MeshFrame().FaceSegment().SetColorOperation(2)');
```

This example sets the ColorOperation to 2 of all the FaceSegment of all the MeshFrames of the mesh called "Box01.

3.22.VertexSegment

3.22.1.1. GetPositionCount

Syntax: GetPositionCount()

The GetPositionCount method returns the number of Position Vectors of the first VertexSegment in the selection. The position vectors are a part of the vertices used for rendering the 3D object. The position vectors are 100% necessary for the vertices to work in the rendering process.

Example: `doTNTCommand('Mesh(0).GetPositionCount()');`

This example returns the number of position vectors of the mesh at index 0.

3.22.1.2. AppendPosition

Syntax: AppendPosition(x:float, y:float, z:float)

The AppendPosition method will append a position vector to one of more VertexSegments in the selection. The arguments x, y, z are used as the x,y,z part of a 3D dimensional position vector that will be added to the end of the position array.

Example: `doTNTCommand('Mesh("MyMesh").AppendPosition(1.0,2.0,3.0)');`

This example will add a position vector to the end of the position array of the mesh called "MyMesh".

3.22.1.3. GetNormalCount

Syntax: GetNormalCount()

The GetNormalCount method returns the number of Normal Vectors of the first VertexSegment in the selection. The normal vectors can be a part of the vertices used for rendering the 3D object.

Example: `doTNTCommand('Mesh("MyMesh").GetNormalCount()');`

This example returns the number of normal vectors of the mesh called "MyMesh".

3.22.1.4. AppendNormal

Syntax: AppendNormal(x:float, y:float, z:float)

The AppendNormal method will append a normal vector to one of more VertexSegments in the selection. The arguments x, y, z are used as the x,y,z part of a 3D dimensional normal vector that will be added to the end of the normal array. The lenght of the normal vector should always be one.

Example: `doTNTCommand('Mesh("MyMesh").AppendNormal(0.0,1.0,0.0)');`

This example will add a normal vector to the end of the normal array of the mesh called "MyMesh".

3.23.Material

3.23.1.1. SetMaterialDiffuseColor

Syntax: SetMaterialDiffuseColor(color:integer)

The SetMaterialDiffuseColor method set the DiffuseColor of the material in the selection.

Example: `doTNTCommand('Mesh("Teapot01").SetMaterialDiffuseColor(#FF0000)');`

This example sets the DiffuseColor of the material to red. The material changes is the material used by the mesh called "Teapot01".

3.23.1.2. GetMaterialDiffuseColor

Syntax: GetMaterialDiffuseColor()

The GetMaterialDiffuseColor method returns the DiffuseColor of the material in the selection.

Example: `doTNTCommand('Mesh("Teapot01").GetMaterialDiffuseColor()');`

This example returns the DiffuseColor of the material used by the mesh called "Teapot01".

3.23.1.3. SetMaterialAmbientColor

Syntax: SetMaterialAmbientColor(color:integer)

The SetMaterialAmbientColor method set the AmbientColor of the material in the selection.

Example: `doTNTCommand('Mesh("Teapot01").SetMaterialAmbientColor(#00FF00)');`

This example sets the AmbientColor of the material to green. The material changes is the material used by the mesh called "Teapot01".

3.23.1.4. GetMaterialAmbientColor

Syntax: GetMaterialAmbientColor()

The GetMaterialAmbientColor method returns the AmbientColor of the material in the selection.

Example: `doTNTCommand('Mesh("Teapot01").GetMaterialAmbientColor()');`

This example returns the AmbientColor of the material used by the mesh called "Teapot01".

3.23.1.5. SetMaterialSpecularColor

Syntax: SetMaterialSpecularColor(color:integer)

The SetMaterialSpecularColor method set the SpecularColor of the material in the selection.

Example: `doTNTCommand('Mesh("Teapot01").SetMaterialSpecularColor(#0000FF)');`

This example sets the SpecularColor of the material to blue. The material changes is the material used by the mesh called "Teapot01".

3.23.1.6. GetMaterialSpecularColor

Syntax: GetMaterialSpecularColor()

The GetMaterialSpecularColor method returns the SpecularColor of the material in the selection.

Example: `doTNTCommand('Mesh("Teapot01").GetMaterialSpecularColor()');`

This example returns the SpecularColor of the material used by the mesh called "Teapot01".

3.23.1.7. SetMaterialEmissiveColor

Syntax: SetMaterialEmissiveColor(color:integer)

The SetMaterialEmissiveColor method set the EmissiveColor of the material in the selection.

Example: `doTNTCommand('Mesh("Teapot01").SetMaterialEmissiveColor(#FFFFFF)');`

This example sets the EmissiveColor of the material to white. The material changes is the material used by the mesh called "Teapot01".

3.23.1.8. GetMaterialEmissiveColor

Syntax: GetMaterialEmissiveColor()

The GetMaterialEmissiveColor method returns the EmissiveColor of the material in the selection.

Example: `doTNTCommand('Mesh("Teapot01").GetMaterialEmissiveColor()');`

This example returns the EmissiveColor of the material used by the mesh called "Teapot01".

3.23.1.9. SetMaterialPower

Syntax: SetMaterialPower(power:float)

The SetMaterialPower method sets the power value of the material in the selection.

Example: `doTNTCommand('Mesh("Teapot01").SetMaterialPower(1.2)');`

This example sets the power value of the material to 1.2. The material changes is the material used by the mesh called "Teapot01".

3.23.1.10. GetMaterialPower

Syntax: GetMaterialPower()

The GetMaterialPower method returns the power value of the material in the selection.

Example: `doTNTCommand('Mesh("Teapot01").GetMaterialPower()');`

This example returns the Power value of the material used by the mesh called "Teapot01".

3.23.1.11. SetMaterialTransparency

Syntax: SetMaterialTransparency(transparency:float)

The SetMaterialTransparency method sets the transparency value of the material in the selection. A transparency value of 0.0 means 100% transparent. A transparency value of 0.5 means 50% transparent. A transparency value of 1.0 means 0% transparent or in other words 100% opaque.

Example: `doTNTCommand('Mesh("Torus01").SetMaterialTransparency(0.5)');`

This example sets the transparency value of the material to 0.5. The material changes is the material used by the mesh called "Torus01".

3.23.1.12. GetMaterialTransparency

Syntax: GetMaterialTransparency()

The GetMaterialTransparency method returns the transparency value of the material in the selection.

Example: `doTNTCommand('Mesh("Torus01").GetMaterialTransparency()');`
This example returns the Transparency value of the material used by the mesh called "Torus01".

3.23.1.13. SetMaterialTwoSided
Syntax: SetMaterialTwoSided(value:integer)
The SetMaterialTwoSided method sets the TwoSided value of the material in the selection. If the TwoSided value is a non zero value then the polygons rendered with this material will be visible from both sides.
Example: `doTNTCommand('Mesh("Torus01").SetMaterialTwoSided(1)');`
This example sets the TwoSided value of the material used by the mesh called "Torus01".

3.23.1.14. GetMaterialTwoSided
Syntax: GetMaterialTwoSided()
The GetMaterialTwoSided method returns the TwoSided value of the material in the selection.
Example: `doTNTCommand('Mesh("Torus01").GetMaterialTwoSided()');`
This example returns the TwoSided value of the material used by the mesh called "Torus01".

3.23.1.15. SetMaterialWireframe
Syntax: SetMaterialWireframe(integer)
The SetMaterialWireframe method sets the Wireframe value of the material in the selection.
Example: `doTNTCommand('Mesh("Torus01").SetMaterialWireframe(1)');`
This example sets the Wireframe value of the material used by the mesh called "Torus01".

3.23.1.16. GetMaterialWireframe
Syntax: GetMaterialWireframe()
The GetMaterialWireframe method returns the Wireframe value of the material in the selection.
Example: `doTNTCommand('Mesh("Torus01").GetMaterialWireframe()');`
This example returns the Wireframe value of the material used by the mesh called "Torus01".

3.23.1.17. SetMaterialMinEmissive
Syntax: SetMaterialMinEmissive(color:integer)
The SetMaterialMinEmissive method set the MinEmissive color of the material in the selection. The MinEmissive color can be used to set a minimum light received by a mesh giving it the appearance of being illuminated by global ambient light.
Example: `doTNTCommand('Mesh("Torus01").SetMinEmissive(#606060)');`
This example sets the MinEmissive color of the material used by the mesh called "Torus01".

3.24.TextureFragment

3.24.1.1. GetUVCount

Syntax: GetUVCount()

The GetUVCount method returns the number of UV Coordinates of the first TextureFragment in the selection. The UV coordinates are an array of two dimensional vectors that are used as indices into the texture for each vertex.
Example: `doTNTCommand('Mesh("MyMesh").GetUVCount()');`
This example returns the UV coordinate count of first TextureFragment of the mesh called "MyMesh".

3.24.1.2. AppendUV

Syntax: AppendUV(u:float, v:float)

The AppendUV method will append a UV coordinate to one of more TextureFragment in the selection. The arguments u,v are used as the u,v part of a 2D dimensional vector that will be added to the end of the uv array.
Example: `doTNTCommand('Mesh("MyMesh").AppendUV(0.0,1.0)');`
This example will add a UV vector to the end of the UV array of the mesh called "MyMesh".

3.24.1.3. SetUVOffset

Syntax: SetUVOffset(u:float, v:float)

The SetUVOffset method will set the UV offset in the UV matrix used for transforming the UV coordinates when rendering the 3D object.
Example:

```
doTNTCommand('Mesh("Floor").MeshFrame().FaceSegment().TextureFragment(0).SetUV
Offset(2.0,3.0)');
```

This example set the UV offset of the TextureFragment at index 0 of all the FaceSegments of all the MeshFrames of the mesh called "Floor".

3.24.1.4. GetUOffset

Syntax: GetUOffset()

The GetUOffset method returns the UOffset of the first TextureFragment in the selection.
Example:

```
doTNTCommand('Mesh("Floor").MeshFrame().FaceSegment().TextureFragment(0).GetUO
ffset()');
```

This example returns the U offset of the TextureFragment at index 0 of the mesh called "Floor".

3.24.1.5. GetVOffset

Syntax: GetVOffset()

The GetVOffset method returns the VOffset of the first TextureFragment in the selection.
Example:

```
doTNTCommand('Mesh("Floor").MeshFrame().FaceSegment().TextureFragment(0).GetVO
ffset()');
```

This example returns the V offset of the TextureFragment at index 0 of the mesh called "Floor".

3.24.1.6. SetUVTiling

Syntax: SetUVTiling(u:float, v:float)

The SetUVTiling method will set the UV tiling in the UV matrix used for transforming the UV coordinates when rendering the 3D object.

Example:

```
doTNTCommand('Mesh("Floor").MeshFrame().FaceSegment().TextureFragment(0).SetUV
Offset(2.0,2.0)');
```

This example set the U and V tiling of the TextureFragment at index 0 of all the FaceSegments of all the MeshFrames of the mesh called "Floor".

3.24.1.7. GetUTiling

Syntax: GetUTiling()

The GetUTiling method returns the UTiling of the first TextureFragment in the selection.

Example:

```
doTNTCommand('Mesh("Floor").MeshFrame().FaceSegment().TextureFragment(0).GetUT
iling()');
```

This example returns the U Tiling of the TextureFragment at index 0 of the mesh called "Floor".

3.24.1.8. GetVTiling

Syntax: GetVTiling()

The GetVTiling method returns the VTiling of the first TextureFragment in the selection.

Example:

```
doTNTCommand('Mesh("Floor").MeshFrame().FaceSegment().TextureFragment(0).GetVT
iling()');
```

This example returns the V Tiling of the TextureFragment at index 0 of the mesh called "Floor".

3.24.1.9. SetUVMatrix

Syntax: SetUVMatrix(tilingU:tilingV,offsetU,offsetV,rotationW)

The SetUVMatrix method will set the UVTiling,UVOffset and W rotation in the UV matrix used for transforming the UV coordinates when rendering the 3D object.

Example:

```
doTNTCommand('Mesh("Floor").MeshFrame().FaceSegment().TextureFragment(0).SetUV
Matrix(2.0,1.0,0,0,0)');
```

This example sets the UVTiling,UVOffset and W rotation of the TextureFragment at index 0 of the mesh called "Floor".

3.24.1.10. GetWRotation

Syntax: GetWRotation()

The GetWRotation method returns the WRotation of the first TextureFragment in the selection.

Example:

```
doTNTCommand('Mesh("Floor").MeshFrame().FaceSegment().TextureFragment(0).GetWR
otation()');
```

This example returns the W rotation of the TextureFragment at index 0 of the mesh called "Floor".

3.24.1.11. GetTextureFactor

Syntax: GetTextureFactor()

The GetTextureFactor method return the TextureFactor of the first TextureFragment in the selection.

Example:

```
doTNTCommand('Mesh("Floor").MeshFrame().FaceSegment().TextureFragment(1).GetTextureFactor()');
```

This example returns the TextureFactor of the texture fragment at index 1 of the mesh called "Floor".

3.24.1.12. SetTextureFactor

Syntax: SetTextureFactor(float)

The SetTextureFactor method sets the TextureFactor of one or more TextureFragment in the selection.

Example:

```
doTNTCommand('Mesh("Floor").MeshFrame().FaceSegment().TextureFragment(2).SetTextureFactor(0.5)');
```

This example sets the TextureFactor of the TextureFragment at index 2 of the mesh called "Floor".

3.24.1.13. CreateTexture

Syntax: CreateTexture()

The CreateTexture method creates a Texture object on one or more TextureFragments in the selection. Only one texture can exists for each TextureFragment, so if this method is called on a TextureFragment that already has a Texture then the command will have no effect.

Example:

```
doTNTCommand('Mesh("Floor").MeshFrame().FaceSegment().TextureFragment().CreateTexture()');
```

This example creates a texture on all TextureFragments of all the FaceSegments of all the MeshFrames of the Mesh called "Floor".

3.25.Texture

3.25.1.1. CreateBitmap

Syntax: CreateBitmap(count:integer)

The CreateBitmap method creates one or more Bitmap objects on the Texture in the selection.

Example:

```
doTNTCommand('CreateNode("UVBox").GenerateBox(20.0,20.0,20.0).SetMaterialDiffuseColor(#FFFFFF).SetMaterialEmissiveColor(#808080).GenerateUVBoxCoords(1.0,1.0,1.0).CreateTexture().CreateBitmap().LoadBitmap("drusty.png")');
```

This example creates a box mesh using a node called "UVBox", UV coordinates are generated a texture and bitmap is created and finally the image drusty.png is loaded onto the created bitmap.

3.26.Bitmap

3.26.1.1. SetBitmapFilename

Syntax: SetBitmapFilename(name:string)

The SetBitmapFilename method sets the FileName of one or more Bitmaps in the

selection.
Example: `doTNTCommand('Bitmap(0).SetBitmapFilename("MyTexture")');`
This example sets the filename of the bitmap at index 0 to "MyTexture".

3.26.1.2. GetBitmapFilename
Syntax: GetBitmapFilename()
The GetBitmapFilename method returns the Filename of the first Bitmap in the selection.
Example: `doTNTCommand('Bitmap(0).GetBitmapFilename()');`
This example returns the Filename of the Bitmap at index 0.

3.26.1.3. GetBitmapWidth
Syntax: GetBitmapWidth()
The GetBitmapWidth method returns the width of the first Bitmap in the selection.
Example: `doTNTCommand('Bitmap(0).GetBitmapWidth()');`
This example returns the width of the bitmap at index 0.

3.26.1.4. GetBitmapHeight
Syntax: GetBitmapHeight()
The GetBitmapHeight method returns the height of the first Bitmap in the selection.
Example: `doTNTCommand('Bitmap("MyBitmap").GetBitmapWidth()');`
This example returns the width of the bitmap called "MyBitmap".

3.26.1.5. LoadBitmap
Syntax: LoadBitmap(url:string)
The LoadBitmap method loads or downloads a bitmap from an absolute or relative path onto one or more bitmaps in the selection. If the filename already have been downloaded the the file in the temporary folder will be used.
Example:
```
doTNTCommand('CreateNode("UVBox").GenerateBox(20.0,20.0,20.0).SetMaterialDiffu
seColor(#FFFFFF).SetMaterialEmissiveColor(#808080).GenerateUVBoxCoords(1.0,1.0
,1.0).CreateTexture().CreateBitmap().LoadBitmap("drusty.png")');
```
This example creates a box mesh using a node called "UVBox", UV coordinates are generated a texture and bitmap is created and finally the image drusty.png is loaded onto the created bitmap.

3.26.1.6. UnloadBitmap
Syntax: UnloadBitmap()
The UnloadBitmap method will Unload the bitmap on one or more Bitmap objects in the selection.
Example: `doTNTCommand('Bitmap("drusty").UnloadBitmap()');`
This example will unload the bitmap data from the bitmap object called "drusty".

3.26.1.7. SetBitmapCompression
Syntax: SetBitmapCompression(compression:float)
The SetBitmapCompression method will set the BitmapCompression value of one or more bitmaps in the selection. The compression argument is a floating point value

ranging from 0.0 to 1.0. The higher the compression value the higher the bitmap will be compressed when the scene is saved.
Example: `doTNTCommand('Bitmap("*").SetBitmapCompression(0.50)');`
This example sets the compression of all bitmap to 50%

3.26.1.8. GetBitmapCompression
Syntax: GetBitmapCompression()
The GetBitmapCompression method returns the BitmapCompression value of the first bitmap in the selection.
Example: `doTNTCommand('Bitmap("MyBitmap").GetBitmapCompression()');`
This example returns the compression of the bitmap called "MyBitmap".

3.26.1.9. SetBitmapLightMap
Syntax: SetBitmapLightMap(switch:integer)
The SetBitmapLightMap method sets the LightMap value of one or more bitmaps in the selection. The LightMap is a non zero value then
Example: `doTNTCommand('Bitmap("bakedmap").SetBitmapLightMap(1)');`
This example sets the LightMap value of the Bitmap called "bakedmap" to 1.

3.26.1.10. GetBitmapLightMap
Syntax: GetBitmapLightMap()
The GetBitmapLightMap method returns the LightMap value of the first Bitmap in the selection.
Example: `doTNTCommand('Bitmap("bakedmap").GetBitmapLightMap()');`
This example returns the LightMap value of the Bitmap called "bakedmap".

3.26.1.11. SetLightmapOffset
Syntax: SetLightmapOffset(offset:float)
The SetLightmapOffset method sets the LightmapOffset value of one or more bitmaps in the selection.
Example:
```
doTNTCommand('Mesh("LightMesh").MeshFrame().FaceSegment().TextureFragment(1).SetLightmapOffset(-1.0)');
```
This example sets the Lightmap Offset of the bitmap of the TextureFragment at index 1 of all the FaceSegments of all the MeshFrame of the mesh called "LightMesh" to -1.0.

3.26.1.12. GetLightmapOffset
Syntax: GetLightmapOffset()
The GetLightmapOffset method returns the LightmapOffset value of the first bitmap in the selection.
Example:
```
doTNTCommand('Mesh("LightMesh").MeshFrame().FaceSegment().TextureFragment(1).GetLightmapOffset()');
```
This example returns the Lightmap Offset of the bitmap of the TextureFragment at index 1 of first FaceSegment of the first MeshFrame of the mesh called "LightMesh".

3.26.1.13. SetBitmapExternal
Syntax: SetBitmapExternal(value:integer)
The SetBitmapExternal method sets the External value of one or more Bitmaps in the selection. If the External value is '0' then the bitmap will be included in the TNT file. If the External value is '1' then the bitmap will be saved externally beside the TNT file. Bitmaps that have alpha channel will be saved as PNG and bitmaps without alpha will be saved in the JPG fileformat.
Example: `doTNTCommand('Bitmap("*").SetBitmapExternal(1)');`
This example sets all bitmaps in the scene to be external.

3.26.1.14. GetBitmapExternal
Syntax: GetBitmapExternal()
The GetBitmapExternal method returns the External value of the first Bitmap in the selection.
Example: `doTNTCommand('Bitmap("bakedmap").GetBitmapExternal()');`
This example returns the External value of the Bitmap called "bakedmap".

3.26.1.15. SetBitmapMipMap
Syntax: SetBitmapMipMap(value:integer)
The SetBitmapMipMap method sets the MipMap value of one or more Bitmaps in the selection. Then the MipMap value is '1' then bitmap will use an advanced rendering technique called mipmapping. Briefly explained, this means that textures use multiple levels of detail. This can improve both performance and rendering quality, the expense being an increase in the usage of video memory (33% increase). Best used with high detail bitmaps that are seen from both near and far distances.
Example: `doTNTCommand('Bitmap("Wood").SetBitmapMipMap(1)');`
This example enables MipMapping on the Bitmap called "Wood".

3.26.1.16. GetBitmapMipMap
Syntax: GetBitmapMipMap()
The GetBitmapMipMap method returns the MipMap value of the first Bitmap in the selection.
Example: `doTNTCommand('Bitmap("Wood").GetBitmapMipMap()');`
This example returns the MipMap vaue of the bitmap called "Wood".

3.26.1.17. BitmapUpdate
Syntax: BitmapUpdate()
The BitmapUpdate method will recreate or update the Bitmap on the graphicscard.
Example: `doTNTCommand('Bitmap("Wood").BitmapUpdate()');`
This example will update the bitmap called "Wood".

3.27.LOD

3.27.1.1. CreateLODRangeInfo
Syntax: CreateLODRangeInfo(count:integer)
The CreateLODRangeInfo method creates a number of LODRangeInfo objects and assignes them to the LOD object in the selection. LODRangeInfo objects are used to define when the LOD object will switch between the different Level Of Detail that each

are created as a TNT file. Each LODRangeInfo object contain information regarding each Level Of Detail.
Example: `doTNTCommand('LOD("MyLOD").CreateLODRangeInfo(3)');`
This example creates 3 LODRangeInfo objects on the LOD object called "MyLOD".

3.27.1.2. GetLODRangeInfoCount
Syntax: GetLODRangeInfoCount()
The GetLODRangeInfoCount method return the number of LODRangeInfo object for the first LOD object in the selection.
Example: `doTNTCommand('LOD("MyLOD").GetLODRangeInfoCount()');`
This example returns the number of LODRangeInfo objects

3.28.LODRangeInfo

3.28.1.1. SetURL
Syntax: SetURL(url:string)
The SetURL method sets the URL string of an LODRangeInfo object in the selection.
Example: `doTNTCommand('LOD("MyLOD").LODRangeInfo(0).SetURL("object0.tnt")');`
This example sets the URL of the LODRangeInfo object at index 0 to "object0.tnt".

3.28.1.2. GetURL
Syntax: GetURL()
The GetURL method returns the URL of the first LODRangeInfo object in the selection.
Example: `doTNTCommand('LOD("MyLOD").LODRangeInfo(0).GetURL()');`
This example returns the URL of the LODRangeInfo at index 0 of the LOD called "MyLOD".

3.28.1.3. SetMaxValue
Syntax: SetMaxValue(range:float)
The SetMaxValue method sets the Max value of one or more LODRangeInfo objects in the selection. The MaxValue determines when an object will be replaced with the object define in the next LODRangeInfo object. The MaxValue is the maximum distance an object can be away from the current camera before it swapped out of the scene.
Example: `doTNTCommand('LOD("MyLOD").LODRangeInfo(0).SetMaxValue(1300)');`
This example sets the MaxValue of the LODRangeInfo object at index 0 of the LOD called "MyLOD" to 1300.

3.28.1.4. GetMaxValue
Syntax: GetMaxValue()
The GetMaxValue method return the MaxValue of the first LODRangeInfo object in the selection.
Example: `doTNTCommand('LOD("MyLOD").LODRangeInfo(0).GetMaxValue()');`
This example returns the MaxValue of the LODRangeInfo object at index 0 of the LOD object called "MyLOD".

3.29.NodeDistanceEvent

3.29.1.1. SetNodeDistEventEnabled

Syntax: SetNodeDistEventEnabled(switch:integer)

The SetNodeDistEventEnabled method sets the enabled value of the NodeDistEvent object in the selection. When the Enabled value is zero then the NodeDistEvent object will not be active at all. However when the Enabled is '1' then the NodeDistEvent object will be active.

Example: `doTNTCommand('NodeDistEvent(5).SetNodeDistEventEnabled(1)');`

This example sets the NodeDistEventEnabled value of the of the NodeDistEvent at index 5 to 1.

3.29.1.2. GetNodeDistEventEnabled

Syntax: GetNodeDistEventEnabled()

The GetNodeDistEventEnabled method returns the NodeDistEventEnabled value of the NodeDistEvent object in the selection.

Example: `doTNTCommand('NodeDistEvent(5).GetNodeDistEventEnabled()');`

This example returns the NodeDistEventEnabled value of the NodeDistEvent object at index 5.

3.29.1.3. SetNodeDistEventRadius

Syntax: SetNodeDistEventRadius(radius:float)

The SetNodeDistEventRadius method sets the radius of the NodeDistEvent object in the selection.

Example: `doTNTCommand('NodeDistEvent(2).SetNodeDistEventRadius(45.0)');`

This example sets the NodeDistEventRadius of the NodeDistEvent object at index 2 to 45.0.

3.29.1.4. GetNodeDistEventRadius

Syntax: GetNodeDistEventRadius()

The GetNodeDistEventRadius method returns the NodeDistEventRadius of the first NodeDistEvent object in the selection.

Example: `doTNTCommand('NodeDistEvent(2).GetNodeDistEventRadius()');`

This example returns the NodeDistEventRadius of the NodeDistEvent at index 2.

3.29.1.5. SetNodeDistEventThreshold

Syntax: SetNodeDistEventThreshold(threshold:float)

The SetNodeDistEventThreshold method set the NodeDistEventThreshold of the NodeDistEvent object in the selection.

Example: `doTNTCommand('NodeDistEvent(2).SetNodeDistEventThreshold(1.0)');`

This example sets the SetNodeDistEventThreshold of the NodeDistEvent at index 2 to 1.0.

3.29.1.6. GetNodeDistEventThreshold

Syntax: GetNodeDistEventThreshold()

The GetNodeDistEventThreshold returns the NodeDistEventThreshold value of the NodeDistEvent object in the selection.

Example: `doTNTCommand('NodeDistEvent(2).GetNodeDistEventThreshold()');`

This example returns the NodeDistEventThreshold value of NodeDistEvent object at index 2.

3.29.1.7. GetNodeDistEventNode
Syntax: GetNodeDistEventNode(index:integer)
The GetNodeDistEventNode method selects the Node assigned to an index of the NodeDistEventObject.
Example: `doTNTCommand('NodeDistEvent(2).GetNodeDistEventNode(1)');`
This example puts the Node assigned at index 1 of the NodeDistEvent object at index 2 into the selection.

Selection
3.29.1.8. Selection
Syntax: Selection()
The Selection class is a container object that can hold the objects currently selected by the user. Like in normal 3D application the user selects one or more object and move them around in the scene using the move tool. Object that are added to the section
Example: `doTNTCommand('Selection()');`
This example creates a selection and adds the Selection class to this selection.

3.29.1.9. SetSelectionMinX
Syntax: SetSelectionMinX(float)
The SetSelectionMinX method set the MinX value of the selection class. The MinX value is the minimum allowed X coordinate of an object that is being moved in the selection.
Example: `doTNTCommand('Selection().SetSelectionMinX(-400.0)');`
This example sets the MinX value in the selection class to be -400.0

3.29.1.10. GetSelectionMinX
Syntax: GetSelectionMinX()
The GetSelectionMinX method returns the MinX value of the Selection class.
Example: `doTNTCommand('Selection().GetSelectionMinX()');`
This example returns the MinX value of the Selection class.

3.29.1.11. SetSelectionMaxX
Syntax: SetSelectionMaxX(float)
The SetSelectionMaxX method set the MaxX value of the selection class. The MaxX value is the maximum allowed X coordinate of an object that is being moved in the selection.
Example: `doTNTCommand('Selection().SetSelectionMaxX(400.0)');`
This example sets the MaxX value in the selection class to be 400.0

3.29.1.12. GetSelectionMaxX
Syntax: GetSelectionMaxX()
The GetSelectionMaxX method returns the MaxX value of the Selection class.
Example: `doTNTCommand('Selection().GetSelectionMaxX()');`

This example returns the MaxX value of the Selection class.

3.29.1.13. SetSelectionMinY
Syntax: SetSelectionMinY(float)
The SetSelectionMinY method set the MinY value of the selection class. The MinY value is the minimum allowed Y coordinate of an object that is being moved in the selection.
Example: `doTNTCommand('Selection().SetSelectionMinY(-400.0)');`
This example sets the MinY value in the selection class to be -400.0

3.29.1.14. GetSelectionMinY
Syntax: GetSelectionMinY()
The GetSelectionMinY method returns the MinY value of the Selection class.
Example: `doTNTCommand('Selection().GetSelectionMinY()');`
This example returns the MinY value of the Selection class.

3.29.1.15. SetSelectionMaxY
Syntax: SetSelectionMaxY(float)
The SetSelectionMaxY method set the MaxY value of the selection class. The MaxY value is the maximum allowed Y coordinate of an object that is being moved in the selection.
Example: `doTNTCommand('Selection().SetSelectionMaxY(400.0)');`
This example sets the MaxY value in the selection class to be 400.0

3.29.1.16. GetSelectionMaxY
Syntax: GetSelectionMaxY()
The GetSelectionMaxX method returns the MaxX value of the Selection class.
Example: `doTNTCommand('Selection().GetSelectionMaxY()');`
This example returns the MaxY value of the Selection class.

3.29.1.17. SetSelectionMinZ
Syntax: SetSelectionMinZ(float)
The SetSelectionMinZ method set the MinZ value of the selection class. The MinZ value is the minimum allowed Z coordinate of an object that is being moved in the selection.
Example: `doTNTCommand('Selection().SetSelectionMinZ(-400.0)');`
This example sets the MinZ value in the selection class to be -400.0

3.29.1.18. GetSelectionMinZ
Syntax: GetSelectionMinZ()
The GetSelectionMinZ method returns the MinZ value of the Selection class.
Example: `doTNTCommand('Selection().GetSelectionMinZ()');`
This example returns the MinZ value of the Selection class.

3.29.1.19. SetSelectionMaxZ
Syntax: SetSelectionMaxZ(float)
The SetSelectionMaxZ method set the MaxZ value of the selection class. The MaxZ

value is the maximum allowed Z coordinate of an object that is being moved in the selection.
Example: `doTNTCommand('Selection().SetSelectionMaxZ(400.0)');`
This example sets the MaxZ value in the selection class to be 400.0

3.29.1.20. GetSelectionMaxZ
Syntax: GetSelectionMaxZ()
The GetSelectionMaxZ method returns the MinX value of the Selection class.
Example: `doTNTCommand('Selection().GetSelectionMaxZ()');`
This example returns the MaxZ value of the Selection class.

3.29.1.21. SetSelectionMoveDirectionX
Syntax: SetSelectionMoveDirectionX(switch:integer)
The SetSelectionMoveDirectX method sets the SelectionMoveDirectionX value of the selection class. When the SelectionMoveDirectionX is '1' then it is possible to move objects in the selection class in the X direction.
Example: `doTNTCommand('Selection().SetSelectionMoveDirectionX(1)');`

3.29.1.22. GetSelectionMoveDirectionX
Syntax: GetSelectionMoveDirectionX()
The GetSelectionMoveDirectionX method returns the SelectionMoveDirectionX value of the selection class.
Example: `doTNTCommand('Selection().GetSelectionMoveDirectionX()');`

3.29.1.23. SetSelectionMoveDirectionY
Syntax: SetSelectionMoveDirectionY(switch:integer)
The SetSelectionMoveDirectY method sets the SelectionMoveDirectionY value of the selection class. When the SelectionMoveDirectionY is '1' then it is possible to move objects in the selection class in the Y direction.
Example: `doTNTCommand('Selection().SetSelectionMoveDirectionY(1)');`

3.29.1.24. GetSelectionMoveDirectionY
Syntax: GetSelectionMoveDirectionY()
The GetSelectionMoveDirectionY method returns the SelectionMoveDirectionY value of the selection class.
Example: `doTNTCommand('Selection().GetSelectionMoveDirectionY()');`

3.29.1.25. SetSelectionMoveDirectionZ
Syntax: SetSelectionMoveDirectionZ(switch:integer)
The SetSelectionMoveDirectZ method sets the SelectionMoveDirectionZ value of the selection class. When the SelectionMoveDirectionZ is '1' then it is possible to move objects in the selection class in the Z direction.
Example: `doTNTCommand('Selection().SetSelectionMoveDirectionZ(1)');`

3.29.1.26. GetSelectionMoveDirectionZ
Syntax: GetSelectionMoveDirectionZ()
The GetSelectionMoveDirectionZ method returns the SelectionMoveDirectionZ value of the selection class.

Example: `doTNTCommand('Selection().GetSelectionMoveDirectionZ()');`

3.29.1.27. SetSelectionMouseKeyMode

Syntax: SetSelectionMouseKeyMode(mouseKey:integer, moveMode:integer)

The SetSelectionMouseKeyMode method sets the move mode that will be used when a certain mouse key is pressed.
An overview of the Mouse Key values can be seen below.
1 = Left Mouse Button.
2 = Right Mouse Button.
3 = Both Mouse Buttons.
An overview of the Move Mode values can be seen below.
0 = Void - No Reaction.
1 = Position movement.
2 = XYZ Rotation.
3 = X Rotation only
4 = Y Rotation only.
5 = Z Rotation only.
Example1: `doTNTCommand('Selection().SetSelectionMouseKeyMode(1,1)');`
Example1 sets the left mouse-key to use the move mode 1, which is position movement.
Example2: `doTNTCommand('Selection().SetSelectionMouseKeyMode(2,5)');`
Example2 sets the right mouse-key to use the move mode 5, which is Z rotation.
Example3: `doTNTCommand('Selection().SetSelectionMouseKeyMode(3,2)');`
Example3 sets the both mouse-keys to use the move mode, which is any axis rotation.

3.29.1.28. GetSelectionMouseKeyMode

Syntax: GetSelectionMouseKeyMode(mouseKey:integer)

The GetSelectionMouseKeyMode returns the Move Mode of a giving MouseKey.
Example: `doTNTCommand('Selection().GetSelectionMouseKeyMode(1)');`
This example returns the selection move mode of the left mouse button.

3.29.1.29. GetSelectionSize

Syntax: GetSelectionSize()

The GetSelectionSize method returns number of objects in the selecion.
Example: `doTNTCommand('Selection().GetSelectionSize()');`
This example returns the size of the selection.

3.29.1.30. SelectionEntry

Syntax: SelectionEntry(index:integer)

The SelectionEntry method is a selector method that can be used to selection one of the objects in the selection. The index argument define the index of the object in the selection. The range of the index should be between 0 and the size of the selection.
Example: `doTNTCommand('Selection().SelectionEntry(5)');`

This example selects the object at index 5 in the selection.

3.29.1.31. EmptySelection
Syntax: EmptySelection()
The EmptySelection method will empty the selection. In other words when this method has been executed there are no objects in the selection.
Example: `doTNTCommand('Selection().EmptySelection()');`

3.30.Player

3.30.1.1. MovePlayerTo
Syntax:
MovePlayerTo(posx:float,posy:float,posz:float,rotx:float,roty:float,rotz:float)
The MovePlayerTo method can be used to change the position and rotation of a player in the scene. The arguments posx, posy, posz are the X,Y,Z elements of the position vector.
Example: `doTNTCommand('Player(0).MovePlayerTo(5.0,7.0,10.0,0.0,0.0,0.0)');`
This example change the position and rotation of the player at index 0.

3.31.NetClient

3.31.1.1. NetClient
Syntax: NetClient()
The NetClient method will put the NetClient object into the selection. The NetClient objects is a communication object that can be used to make post requests to a server. The NetClient object can also be used to communicate the position and rotation of a 'player' in the scene in realtime.
Example: `doTNTCommand('NetClient()');`

3.31.1.2. SetPlayerName
Syntax: SetPlayerName(name:string)
The SetPlayerName method sets the name of the player at your computer.
Example: `doTNTCommand('NetClient().SetPlayerName("Player1")');`

3.31.1.3. LinkToNetClient
Syntax: LinkToNetClient()
The LinkToNetClient method links the node in the selection to NetClient object as the player on your computer.
Example: `doTNTCommand('Node("PlayerNode").LinkToNetClient()');`
This example links the Node called "PayerNode" to the NetClient object.

3.31.1.4. ConnectToServer
Syntax: ConnectToServer()
The ConnectToServer method will connect the NetClient object to the specified server at a given port.
Example: `doTNTCommand('NetClient().ConnectToServer("www.myserver.com",9911)');`
This example connects to the server at www.myserver.com using port 9111.

3.31.1.5. DisconnectFromServer
Syntax: DisconnectFromServer()
The DisconnectFromServer method disconnects the NetClient object from the server.
Example: `doTNTCommand('NetClient().DisconnectFromServer()');`

3.32.Animation

3.32.1.1. SetLoopStartFrame
Syntax: SetLoopStartFrame(integer)
The SetLoopStartFrame method sets the LoopStartFrame value of one or more Animations in the selection. The LoopStartFrame is the position on the time-line to start the loop of the animation.
Example: `doTNTCommand('Animation("Anim05").SetLoopStartFrame(0)');`
This example sets the LoopStartFrame of the Animation called "Anim05" to 0.

3.32.1.2. GetLoopStartFrame
Syntax: GetLoopStartFrame()
The GetLoopStartFrame method returns the LoopStartFrame value of the first Animation in the selection.
Example: `doTNTCommand('Animation("Anim05").GetLoopStartFrame()');`
This example returns the LoopStartFrame of the Animation called "Anim05".

3.32.1.3. SetLoopStopFrame
Syntax: SetLoopStopFrame(integer)
The SetLoopStopFrame method sets the LoopStopFrame value of one or more Animations in the selection.
Example: `doTNTCommand('Animation("Anim05").SetLoopStopFrame(100)');`
This example sets the LoopStopFrame of the Animation called "Anim05" to 100.

3.32.1.4. GetLoopStopFrame
Syntax: GetLoopStopFrame()
The GetLoopStopFrame method returns the LoopStopFrame value of the first Animation in the selection.
Example: `doTNTCommand('Animation("Anim05").GetLoopStopFrame()');`
This example returns the LoopStopFrame of the Animation called "Anim05".

3.32.1.5. SetFrameRate
Syntax: SetFrameRate(integer)
The SetFrameRate method sets the FrameRate value of one or more animations in the selection. The FrameRate is the number of frame per second that will be played by the Animation.
Example: `doTNTCommand('Animation("MyAnim").SetFrameRate(60)');`
This example sets the FrameRate of the Animation called "MyAnim" to 60.

3.32.1.6. GetFrameRate
Syntax: GetFrameRate()
The GetFrameRate method returns the FrameRate value of the first Animation in the selection.

Example: `doTNTCommand('Animation("MyAnim").GetFrameRate()');`
This example returns the FrameRate of the Animation called "MyAnim".

3.32.1.7. SetLoopCount
Syntax: SetLoopCount(integer)
The SetLoopCount method sets the LoopCount value of one or more Animations in the selection. The LoopCount value determines the how many times the animation will move between the LoopStart and LoopStop value before stopping.
Example: `doTNTCommand('Animation("MyAnim").SetLoopCount(10)');`
This example sets the LoopCount of the Animation called "MyAnim" to 10.

3.32.1.8. GetLoopCount
Syntax: GetLoopCount()
The GetLoopCount method returns the LoopCount value of the first Animation in the selection.
Example: `doTNTCommand('Animation("MyAnim").GetLoopCount()');`
This example returns the LoopCount value of the Animation called "MyAnim".

3.32.1.9. SetFrame
Syntax: SetFrame(integer)
The SetFrame method sets the Frame value of one or more Animations in the selection. The Frame is the position on the timeline that the Animation points to.
Example: `doTNTCommand('Animation("*").SetFrame(57)');`
This example sets the Frame of all Animations to 57.

3.32.1.10. GetFrame
Syntax: GetFrame()
The GetFrame method returns the Frame value of the first Animation in the selection.
Example: `doTNTCommand('Animation("MyAnim").GetFrame()');`
This example returns the Frame of the Animation called "MyAnim".

3.32.1.11. IsPlaying
Syntax: IsPlaying()
The IsPlaying method return the IsPlaying value of the first Animation in the selection. The IsPlaying value is '1' if the animation is playing.
Example: `doTNTCommand('Animation("MyAnim").IsPlaying()');`
This example returns the IsPlaying value of the Animation called "MyAnim".

3.32.1.12. Play
Syntax: Play()
The Play method will when called start to play of one or more Animation in the selection.
Example: `doTNTCommand('Animation("*").Play()');`
This example will begin start playing all Animations in the scene.

3.32.1.13. Stop
Syntax: Stop()

The Stop method will when called stop the playing of one or more Animations in the selection.
Example: `doTNTCommand('Animation("MyAnim").Stop()');`
This example will stop the playing of the Animation called "MyAnim".

3.32.1.14. Reset
Syntax: Reset()
The Reset method will when called reset one or more Animations in the selection.
Example: `doTNTCommand('Animation("MyAnim").Reset()');`
This example will reset the Animation called "MyAnim".

3.32.1.15. PlayAnimation
Syntax:
PlayAnimation(loopStartFrame:integer,loopStopFrame:integer,startFrame:integer,loopCount:integer,frameRate:integer)
The PlayAnimation method will start an animation using the given parameters. The following arguments LoopStartFrame, LoopStopFrame, StartFrame, LoopCount and FrameRate will animation values of with the same name before playing the animation.
Example: `doTNTCommand('Animation("MyAnim").PlayAnimation(0,100,50,5,30)');`
This example starts the animation called "MyAnim", which a loopStartFrame value of 0 and a loopStopFrame value of 100 and a startFrame of 50 and a loopCount of 5, finally the animation is playing with a frameRate of 30.

3.33.Renderer

3.33.1.1. Renderer
Syntax: Renderer()
The Renderer method put the Renderer object into the selection. The Renderer object is where the screen image is generated from the SceneGraph data.
Example: `doTNTCommand('Renderer()');`

3.33.1.2. SaveImage
Syntax:
SaveImage(filename:string,expSize:integer,compression:float)
The SaveImage method will save a rendering as a image of the current viewport to the file specified by filename. The Fileformats available are JPG,PNG,BMP. The parameter exp is the exponent of the size factor, which is always a power of 2. This means the image dimensions will be multiplied with 2exp, e.g. if exp=0 the image saved will have the same dimensions as the viewport, if exp=1 it will be twice as big, if exp=2 it will be four times as big etc.
TECHNICAL NOTE: Because of internal logics the saved image will be clamped to dimensions that are divisible by: 2exp+2 * 2exp. If the dimensions are clamped, the rendering will loose some of its right and bottom borders. This can be helped by making the viewport dimensions divisible by 2expmax+2, where expmax is the highest value of exp to be used. E.g. if exp is 0, the image is clamped to be divisible by 4, and the viewport should ideally also be divisible by 4. If exp is 2, the image is clamped to be divisible by 64 and the viewport should ideally be divisible by 16. The exp parameter can be omitted in which case it is considered to be 0. It is not possible to make a

rendering if 2exp+2 is greater than the viewport size (width OR height), system resource limit are met long before this however. A viewport of 640x480 rendered with an exponent of 3, will generate an image 5120x3840 and will require 78643200 bytes (75 MB) of additional RAM to render. If the exponent is 4 the picture would be twice as big and require 4 times as much RAM (300 MB).
Example1: `doTNTCommand('Renderer().SaveImage('<DESKTOP>MyImage.jpg',1,0.10)');`
This example saves an image on the desktop called "MyImage.jpg". The image is saved in the JPG fileformat and will be compressed 10%.
Example2:
`doTNTCommand('Renderer().SaveImage('<TEMPDIR>MyImage2.png',0,0.0)');`
This example saves an image on the temporary directory called "MyImage2.png". The image is saved in the PNG fileformat and will not be compressed.

3.34.FileManager

3.34.1.1. FileManager

Syntax: FileManager()

The FileManager method put the FileManager object into the selection. The FileManager object can be used to handle the TNT files that are saved from the scene.
The files saved using the FileManager are files that are saved in the unique FileManager project folder.
Example: `doTNTCommand('FileManager()');`
This example puts the FileManager object into the selection.

3.34.1.2. SetCompanyName

Syntax: SetCompanyName(companyName:string)

The SetCompanyName method sets the name of the company. The name of the company should be the name of the company that uses the 3D solution. The company name is a necessary part of a unique FileManager project path.
Example: `doTNTCommand('FileManager().SetCompanyName("MyCompany")');`
This example sets the name of the Company to "MyCompany".

3.34.1.3. GetCompanyName

Syntax: GetCompanyName()

The GetCompanyName method returns the name of the company.
Example: `doTNTCommand('FileManager().GetCompanyName()');`
This example returns the name of the company.

3.34.1.4. SetProjectName

Syntax: SetProjectName(projectName:string)

The SetProjectName method sets the name of the project. The name of the project should be what you call this 3D solution. The project name is a necessary part of a unique FileManager project path.
Example:
`doTNTCommand('FileManager().SetProjectName("MySuperDuper3DProject")');`

This example sets the project name to be "MySuperDuper3DProject".

3.34.1.5. GetProjectName
Syntax: GetProjectName()
The GetProjectName method returns the name of the project.
Example: `doTNTCommand('FileManager().GetProjectName()');`
This example returns the name of the project.

3.34.1.6. GetFileCount
Syntax: GetFileCount()
The GetFileCount method returns the number of files in the unique FileManager project folder.
Example: `doTNTCommand('FileManager().GetFileCount()');`
This example returns the number of files in the unique FileManager project folder.

3.34.1.7. GetFileNameAtIndex
Syntax: GetFileNameAtIndex(index:integer)
The GetFileNameAtIndex method returns the name of the file at a given index. The index should be between 0 and the number return by the GetFileCount method.
Example: `doTNTCommand('FileManager().GetFileNameAtIndex(4)');`
This example returns the name of the file at index 4.

3.34.1.8. DeleteFile
Syntax: DeleteFile(name:string)
The DeleteFile method deletes the file with the given argument from the FileManager project folder.
Example: `doTNTCommand('FileManager().DeleteFile("MyScene3")');`
This example deletes the file called "MyScene3" from the unique Filemanager project folder.

3.34.1.9. Update
Syntax: Update()
The Update method will update the file list of FileManager.
Example: `doTNTCommand('FileManager().Update()');`

3.35.PlanTool

3.35.1.1. Build
Syntax: Build()
The Build method will build the floors and inner and outer walls and create all the necessary holes in the walls for doors and windows. These objects are generated was a result of the points, walls and hole objects that are directly or indirectly linked to the PlanTool object.
Example: `doTNTCommand('PlanTool("MyHouse").Build()');`
This example will build the PlanTool object called "MyHouse".

3.35.1.2. Unbuild
Syntax: Unbuild()
The Unbuild method will remove all the objects that where generated when the PlanTool object was build.
Example: `doTNTCommand('PlanTool("MyHouse").Unbuild()');`
This example will unbuild the PlanTool object called "MyHouse".

3.35.1.3. Empty
Syntax: Empty()
The Empty method will remove all points, walls and hole objects that are linked to the PlanTool object. After the Empty has been called then there will no points, walls or hole objects linked to the PlanTool object.
Example: `doTNTCommand('PlanTool("MyHouse").Empty()');`
This example will empty the PlanTool object called "MyHouse".

3.35.1.4. CreateFloor
Syntax: CreateFloor(count:integer)
The CreateFloor method will create defined number Floor objects on the PlanTool object in selection. The count argument will determine how many floor objects that will be created.
Example: `doTNTCommand('PlanTool("MyHouse1").CreateFloor(2)');`
This example will create two Floor objects on the PlanTool object called "MyHouse1".

3.35.1.5. GetFloorCount
Syntax: GetFloorCount()
The GetFloorCount method will return the number of floors of the first PlanTool object in the selection.
Example: `doTNTCommand('PlanTool("MyHouse1").GetFloorCount()');`
This example returns the number of floors of the PlanTool object called "MyHouse1".

3.35.1.6. Floor
Syntax: Floor(index:integer)
With Floor selector method you can select one or the Floor object from the PlanTool obejct in the selection.
Example: `doTNTCommand('PlanTool("MyHouse1").Floor(1)');`
This example selects the Floor at index 1 of the PlanTool object called "MyHouse1".

3.36.Floor

3.36.1.1. SetFloorHeight
Syntax: SetFloorHeight(height:float)
The SetFloorHeight method will set the Height of the Floor object in the selection.
Example: `doTNTCommand('PlanTool("MyHouse1").Floor(1).SetFloorHeight(230)');`

This example sets the FloorHeight on the Floor at index 1 of the PlanTool object called "MyHouse1".

3.36.1.2. GetFloorHeight
Syntax: GetFloorHeight()
The GetFloorHeight method returns the Height of the first Floor object in the selection.
Example: `doTNTCommand('PlanTool("MyHouse1").Floor(1).GetFloorHeight()');`
This example returns the FloorHeight of the Floor at index 1 of the PlanTool object called "MyHouse1".

3.36.1.3. CreatePoint
Syntax: CreatePoint(count:integer)
The CreatePoint method will create the defined number Point objects on the Floor object in selection. The count argument will determine how many point objects that will be created.
Example: `doTNTCommand('PlanTool("House0").Floor(0).CreatePoint(4)');`
This example will create 4 point objects on the Floor at index 0 of the PlanTool object called "House0".

3.36.1.4. GetPointCount
Syntax: GetPointCount()
The GetPointCount method will return the number of Point objects that are available on the first Floor object in the selection.
Example: `doTNTCommand('PlanTool("Tower").Floor(2).GetPointCount()');`
This example will return the number of Point objects that are available at the Floor at index 2 of the PlanTool object called "Tower".

3.36.1.5. Point
Syntax: Point(index:integer)
With the Point selector method you can select a point using an index of the Floor object in the selection.
Example: `doTNTCommand('PlanTool("MyHouse2").Floor(1).Point(5)');`
This example selects the Point at index 5 of the Floor at index 1 of the PlanTool object called "MyHouse2".

3.36.1.6. CreateWall
Syntax: CreateWall(count:integer)
The CreateWall method will create the defined number of Walls on the Floor object in the selection. The count argument will determine how many wall objects that will be created.
Example: `doTNTCommand('PlanTool("MyHouse5").Floor(0).CreateWall(4)');`
This example will create 4 wall object on the floor at index 0 of the PlanTool object called "MyHouse5".

3.36.1.7. GetWallCount
Syntax: GetWallCount()
The GetWallCount method returns the number of Wall objects that are available on the

first Floor object in the selection.
Example: `doTNTCommand('PlanTool("MyHouse5").Floor(0).GetWallCount()');`
This example will return the number of Walls object for the Floor at index 0 of the PlanTool object called "MyHouse5".

3.36.1.8. Wall
Syntax: Wall(index:integer)
With the Wall selection method a wall can be selected using an index.
Example: `doTNTCommand('PlanTool("MyHouse").Floor(0).Wall(3)');`
This example selects the Wall at index 3 of the Floor at index 0 of the PlanTool object called "MyHouse".

3.37.Point

3.37.1.1. SetPointX
Syntax: SetPointX(position:float)
The SetPointX method sets the X coordinate of the point in the selection.
Example: `doTNTCommand('PlanTool("MyHouse2").Floor(1).Point(5).SetPointX(-100.0)');`
This example sets X coordinate of the point at index 5 of the floor at index 1 of the PlanTool object called "MyHouse2" to -100.0

3.37.1.2. GetPointX
Syntax: GetPointX()
The GetPointX method returns the X coordinate of the point in the selection.
Example: `doTNTCommand('PlanTool("MyHouse2").Floor(1).Point(5).GetPointX()');`
This example returns the X coordinate of the point at index 5 of the floor at index 1 of the PlanTool object called "MyHouse2".

3.37.1.3. SetPointY
Syntax: SetPointY(position:float)
The SetPointY method sets the Y coordinate of the point in the selection.
Example:
`doTNTCommand('PlanTool("MyHouse2").Floor(1).Point(5).SetPointY(100.0)');`
This example sets Y coordinate of the point at index 5 of the floor at index 1 of the PlanTool object called "MyHouse2" to 100.0

3.37.1.4. GetPointY
Syntax: GetPointY()
The GetPointY method returns the Y coordinate of the point in the selection.
Example: `doTNTCommand('PlanTool("MyHouse2").Floor(1).Point(5).GetPointY()');`
This example returns the Y coordinate of the point at index 5 of the floor at index 1 of the PlanTool object called "MyHouse2".

3.38.Wall

3.38.1.1. SetSrcPoint
Syntax: SetSrcPoint(pointIndex:integer)
The SetSrcPoint method sets the Source point index of the wall in the selection.

Each wall must point to two different points using both a source and a destination point index.
Example: `doTNTCommand('PlanTool("MyHouse2").Floor(1).Wall(3).SetSrcPoint(2)');`
This example sets the SrcPoint index of the wall at index 3 of the floor at index 1 of the PlanTool called "MyHouse2".

3.38.1.2. GetSrcPoint
Syntax: GetSrcPoint()
The GetSrcPoint returns the Source point index of the wall in the selection.
Example: `doTNTCommand('PlanTool("TestHouse").Floor(1).Wall(3).GetSrcPoint()');`
This example returns the Source point of the Wall at index 3 of the Floor at index 1 of the PlanTool called "TestHouse".

3.38.1.3. SetDesPoint
Syntax: SetDesPoint(pointIndex:integer)
The SetDesPoint method sets the Destination point index of the wall in the selection. Each wall must point to two different points using both a source and a destination point index.
Example: `doTNTCommand('PlanTool("MyHouse2").Floor(1).Wall(3).SetDesPoint(3)');`
This example sets the Destination point index of the wall at index 3 of the floor at index 1 of the PlanTool called "MyHouse2".

3.38.1.4. GetDesPoint
Syntax: GetDesPoint()
The GetDesPoint returns the Destination point index of the wall in the selection.
Example: `doTNTCommand('PlanTool("TestHouse").Floor(1).Wall(3).GetDesPoint()');`
This example returns the Destination point of the Wall at index 3 of the Floor at index 1 of the PlanTool called "TestHouse".

3.38.1.5. SetHeight
Syntax: SetHeight(height:float)
The SetHeight method sets the Height of the Wall in the selection.
Example:
`doTNTCommand('PlanTool("TestHouse").Floor(1).Wall(3).SetHeight(230)');`
This example sets the height of the Wall at index 3 of the Floor at index 1 of the PlanTool called "TestHouse" to 230.

3.38.1.6. GetHeight
Syntax: GetHeight()
The GetHeight method returns the Height of the Wall in the selection.
Example: `doTNTCommand('PlanTool("TestHouse").Floor(1).Wall(3).GetHeight()');`
This example returns the Height of the Wall at index 3 of the Floor at index 1 of the PlanTool called "TestHouse".

3.38.1.7. CreateHole
Syntax: CreateHole(count:integer)
The CreateHole method will create the defined number of Hole object on the Wall in

the selection. The count argument will determine how many Hole objects that will be created.
Example: `doTNTCommand('PlanTool("MyHouse").Floor(0).Wall(2).CreateHole(1)');`
This example will create 1 Hole object on the Wall at index 2 on the Floor at index 0 of the PlanTool object called "MyHouse".

3.38.1.8. GetHoleCount
Syntax: GetHoleCount()
The GetHoleCount method returns the number of Hole objects of the Wall in the selection.
Example: `doTNTCommand('PlanTool("MyHouse").Floor(0).Wall(2).GetHoleCount()');`
This example returns the number of Hole objects available of the Wall at index 2 of the Floor at index 0 of the PlanTool object called "MyHouse".

3.38.1.9. Hole
Syntax: Hole(index:integer)
With the Hole selector method a Hole object can be selected using an index.
Example: `doTNTCommand('PlanTool("MyHouse").Floor(0).Wall(2).Hole(1)');`
This example will select the Hole at index 1 of the Wall at index 2 of the floor at index 0 of the PlanTool object called "MyHouse".

3.39.Hole

3.39.1.1. GetOffset
Syntax: GetOffset()
The GetOffset method returns the offset value of the Hole object in the selection.
Example:
`doTNTCommand('PlanTool("MyHouse").Floor(0).Wall(2).Hole(1).GetOffset()');`
This example returns the Offset value of the Hole object at index 1 of the Wall at index 2 of the Floor at index 0 of the PlanTool object called "MyHouse".

3.39.1.2. SetOffset
Syntax: SetOffset(offset:float)
The SetOffset method sets the offset value of the Hole object in the selection.
Example:
`doTNTCommand('PlanTool("MyHouse").Floor(0).Wall(2).Hole(1).SetOffset(0.5)');`
This example will set the offset value of the Hole object at index 1 of the Wall at index 2 of the Floor at index 0 of the PlanTool object called "MyHouse" to 0.5.

3.39.1.3. GetWidth
Syntax: GetWidth()
The GetWidth method return the Width value of the Hole object in the selection.
See SetWidth for more details.
Example:
`doTNTCommand('PlanTool("MyHouse").Floor(0).Wall(2).Hole(1).GetWidth()');`
This example returns the GetWidth value of the Hole at index 1 of the Wall at index 2 of the Floor at index 0 of the PlanTool object called "MyHouse".

3.39.1.4. SetWidth
Syntax: SetWidth(width:float)
The SetWidth method sets the Width value of the Hole object in the selection. The Width value of a Hole object determines how wide the Hole is.
Example:
```
doTNTCommand('PlanTool("MyHouse").Floor(0).Wall(2).Hole(0).SetWidth(100.0)');
```
This example will sets the Width value of the Hole at index 0 of the Wall at index 2 of the Floor at index 0 of the PlanTool object called "MyHouse".

3.39.1.5. GetStartHeight
Syntax: GetStartHeight()
The GetStartHeight method returns the StartHeight value of the Hole object in the selection. See SetStartHeight for more details.
Example:
```
doTNTCommand('PlanTool("MyHouse").Floor(1).Wall(4).Hole(0).GetStartHeight()');
```
This example returns the StartHeight value of the Hole object at index 0 of the Wall at index 4 of the Floor at index 1 of the PlanTool object called "MyHouse".

3.39.1.6. SetStartHeight
Syntax: SetStartHeight(startHeight:float)
The SetStartHeight method sets the StartHeight value of the Hole object in the selection. The StartHeight value determines the Start Height of the Hole in the wall. The StartHeight value should always be smaller than the Stop Height value.
Example:
```
doTNTCommand('PlanTool("MyHouse").Floor(1).Wall(4).Hole(0).SetStartHeight(30.0
)');
```
This example sets the StartHeight value of the Hole object at index 0 of the Wall at index 4 of the Floor at index 1 of the PlanTool object called "MyHouse".

3.39.1.7. GetStopHeight
Syntax: GetStopHeight()
The GetStopHeight method returns the StopHeight value of the Hole object in the selection. See SetStopHeight for more details.
Example:
```
doTNTCommand('PlanTool("MyHouse").Floor(1).Wall(4).Hole(0).GetStopHeight()');
```
This example returns the StopHeight value of the Hole object at index 0 of the Wall at index 4 of the Floor at index 1 of the PlanTool object called "MyHouse".

3.39.1.8. SetStopHeight
Syntax: SetStopHeight(stopHeight:float)
The SetStopHeight method sets the StopHeight value of the Hole object in the selection. The StopHeight value determines the Stop Height of the Hole in the wall.The Stop Height value should always be greater than the StartHeight value.
Example:
```
doTNTCommand('PlanTool("MyHouse").Floor(1).Wall(4).Hole(0).SetStopHeight(130.0
)');
```
This example sets the StopHeight value of the Hole object at index 0 of the Wall at index 4 of the Floor at index 1 of the PlanTool object called "MyHouse" to 130.

3.40.SceneData

3.40.1.1. SceneData

Syntax: SceneData()

The SceneData selector method will put the SceneData object into the selection.
The SceneData object can be used to put set any number of integers, floating point values and strings which then will saved and reloaded when the scene is saved and reloaded.
Example: `doTNTCommand('SceneData()');`

3.40.1.2. GetStringCount

Syntax: GetStringCount()

The GetStringCount method returns the number of strings in the SceneData object.
Example: `doTNTCommand('SceneData().GetStringCount()');`

3.40.1.3. GetIntegerCount

Syntax: GetIntegerCount()

The GetIntegerCount method returns the number of integer in the SceneData object.
Example: `doTNTCommand('SceneData().GetIntegerCount()');`

3.40.1.4. GetFloatCount

Syntax: GetFloatCount()

The GetFloatCount method returns the number of floating point values in the SceneData object.
Example: `doTNTCommand('SceneData().GetFloatCount()');`

3.40.1.5. CreateString

Syntax: CreateString(count:integer)

The CreateString method creates the defined number of Strings in the SceneData object. A string is an object that is able to contain any text.
Example: `doTNTCommand('SceneData().CreateString(17)');`
This example will create 14 strings in the SceneData object.

3.40.1.6. CreateInteger

Syntax: CreateInteger(count:integer)

The CreateInteger method creates the defined number of Integers in the SceneData object.
Example: `doTNTCommand('SceneData().CreateInteger(12)');`
This example will create 12 integers in the SceneData object.

3.40.1.7. CreateFloat

Syntax: CreateFloat(count:integer)

The CreateFloat method creates the defined floating point in the SceneData.
Example: `doTNTCommand('SceneData().CreateFloat(517)');`
This example will create 517 floating point values in the SceneData object.

3.40.1.8. SetStringAtIndex

Syntax: SetStringAtIndex(index:integer,text:string)

The SetStringAtIndex method set the text on the string at the defined index.
This string can be returned using the GetStringAtIndex method.
Example: `doTNTCommand('SceneData().SetString(5,"This can be any text")');`
This example sets the string at index 5 to contain "This can be any text".

3.40.1.9. GetStringAtIndex
Syntax: GetStringAtIndex(index:integer)
The GetStringAtIndex method will return the text that the is contained by the string at the defined index. This string can be changed using the SetStringAtIndex method
Example: `doTNTCommand('SceneData().GetStringAtIndex(13)');`
This example will return the text that is contained by the string at index 13.

3.40.1.10. SetIntegerAtIndex
Syntax: SetIntegerAtIndex(index:integer,value:integer)
The SetIntegerAtIndex method will set the integer at the defined index in the integer array. This integer value can be returned using the GetIntegerAtIndex method.
Example: `doTNTCommand('SceneData().SetIntegerAtIndex(17,32)');`
This example sets the integer at index 17 to 32.

3.40.1.11. GetIntegerAtIndex
Syntax: GetIntegerAtIndex(index:integer)
The GetIntegerAtIndex method will return the integer at the defined index in the integer array. This integer value can be changed using the SetIntegerAtIndex method.
Example: `doTNTCommand('SceneData().GetIndexAtIndex(17)');`
This example will return the value of the integer at index 17.

3.40.1.12. SetFloatAtIndex
Syntax: SetFloatAtIndex(index:integer,value:float)
The SetFloatAtIndex method will set the floating point at the defined index in the floating point array. This floating point value can be returned using the GetFloatAtIndex method.
Example: `doTNTCommand('SceneData().SetFloatAtIndex(17,300)');`
This example sets the floating point value at index 17 to 300.

3.40.1.13. GetFloatAtIndex
Syntax: GetFloatAtIndex(index:integer)
The GetFloatAtIndex method will return the floating point value at the defined index in the floating point array. This floating point value can be changed using the SetFloatAtIndex method.
Example: `doTNTCommand('SceneData().GetFloatAtIndex(17)');`
This example will return the value in floating point array at index 17.

3.41.EventManager
3.41.1.1. EventManager
Syntax: EventManager()
The EventManager selector method will put the EventManager object into the selection. The EventManager is the manager of events generated in TurnTool. It is possible to get

the number of events available in the EventManager. With the EventManager each event can be enabled and disable individually.
Example: `doTNTCommand('EventManager()');`

3.41.1.2. GetEventCount
Syntax: GetEventCount()
The GetEventCount method returns the number of events available in the EventManager.
Example: `doTNTCommand('EventManager().GetEventCount()');`

3.41.1.3. GetEventEnabled
Syntax: GetEventEnabled(eventIndex:integer)
The GetEventEnabled method returns the EnabledState of the event index defined by the eventIndex argument. The returned value will be "0" if the event is disabled and "1" if the event is enabled. The eventIndex argument must be lower that the value returned by the GetEventCount.
Example: `doTNTCommand('EventManager().GetEventEnabled(5)');`
This example will return a integer value that can be used to determine whether the event at index 5 is enabled or disabled.

3.41.1.4. SetEventEnabled
Syntax: SetEventEnabled(eventIndex:integer,enabledState:integer)
The SetEventEnabled method sets the EnabledState of the event index defined by the eventIndex argument. If the EnabledState value is sets to "0" then the event is disabled and if sets to "1" then is event is enabled.
Example1: `doTNTCommand('EventManager().GetEventEnabled(3,1)');`
Example1 will enable the event at index 3.
Example2: `doTNTCommand('EventManager().GetEventEnabled(4,0)');`
Example2 will disable the event at index 4.

3.41.1.5. GetEventName
Syntax: GetEventName(eventIndex:integer)
The GetEventName method returns the name of the event defined by the eventIndex argument.
Example: `doTNTCommand('EventManager().GetEventName(4)');`
This example returns the name of the event at index 4.

3.41.1.6. SetEventName
Syntax: SetEventName(eventIndex:integer,newEventName:string)
The SetEventName method sets the name of the event defined by the eventIndex argument. The name of the event will be set to the value of the newEventName argument.
Example1: `doTNTCommand('EventManager().SetEventName(0,"OnMySceneLoad")');`
This example sets the name of the event at index 0 to "OnMySceneLoad".

3.42.CursorIconHandler

3.42.1.1. CursorIconHandler

Syntax: CursorIconHandler()

The CursorIconHandler selector method will put the CursorIconHandler object into the selection. The CursorIconHandler can be used to changed the image of the Cursor by script when certain events occur in your TurnTool solution.
Example: `doTNTCommand('CursorIconHandler()');`

3.42.1.2. SetCursorIconMode

Syntax: SetCursorIconMode(cursorMode:integer)

The SetCursorIconMode method sets the cursorMode value in the CursorIconHandler object. The icon-image of the Cursor can be changed when change the cursorMode value. Which icon-image that will be disable in each mode and either none,left,right or both mouse button are pressed can be controlled using the SetCursorIcon method.
Example: `doTNTCommand('CursorIconHandler().SetCursorIconMode(1)');`
This example sets the CursorIconMode to 1.

3.42.1.3. GetCursorIconMode

Syntax: GetCursorIconMode()

The GetCursorIconMode method returns the cursorMode value for the CursorIconHandler.
Example: `doTNTCommand('CursorIconHandler().GetCursorIconMode()');`

3.42.1.4. SetCursorIcon

Syntax: SetCursorIcon(cursorMode:integer,state:string,icon:string)

The SetCursorIcon method sets which icon-image to display for a given mouse key combination when a certain CursorMode is active.
An overview of the valid State argument strings can be seen below.
"None", "Left","Right", "Both".
An overview of the valid Icon argument strings can be seen below.
"None","Arrow","Finger","FingerClick","Hand","HandGrab","Help","HourGlass","Magnify".
Example1:
`doTNTCommand('CursorIconHandler().SetCursorIcon(1,"None","Finger")');`
Example1 will display the "Finger" icon-image when "None" mouse button are pressed while CursorIcon is "1".
Example2:
`doTNTCommand('CursorIconHandler().SetCursorIcon(1,"Left","FingerClick")');`
Example2 will display the "FingerClick" icon-image when "Left" mouse button is pressed while CursorIcon is "1".

3.42.1.5. GetCursorIcon

Syntax: GetCursorIcon(cursorMode:integer,state:string)

The GetCursorIcon method will return which icon-image will be used for a given mouse-key combination.when a certain CursorMode is active.
Example: `doTNTCommand('CursorIconHandler().GetCursorIcon(1,"Left")');`
This example will return which icon-image will be used when the left mouse key is pressed while the cursorIcon mode is 1.

3.43.Viewer

3.43.1.1. LoadScene

Syntax: LoadScene(filePath:string)

The LoadScene method will when called load the scene using the filePath argument. The filePath argument is a relative path from the location of the HTML document to a file in the TurnTool format. All the scene objects that where saved into the file will be re-created and re-stored to the same state as they were in prior to saving them as a TNT file.

Example: `doTNTCommand('Viewer().LoadScene("testfile.tnt")');`

This example will load the scene using a file called "testfile.tnt".

3.43.1.2. SaveScene

Syntax: SaveScene(filePath:string)

The SaveScene method will when called save the scene using the filePath argument. The filePath argument is a relative path from the location of the HTML document to a file in the TurnTool format. All the scene objects, and their settings, which exist at the time of calling the SaveScene method will be saved into the TNT file.

Example: `doTNTCommand('Viewer().SaveScene("testfile.tnt")');`

This example will save the scene using a file called "testfile.tnt".

3.43.1.3. ClearScene

Syntax: ClearScene()

The ClearScene method will when called erase all scene objects leaving the scene empty.

Example: `doTNTCommand('Viewer().ClearScene()');`

This example will empty the scene completely for all scene objects.

3.43.1.4. GetScreenUpdateFrequency

Syntax: GetScreenUpdateFrequency()

The GetSceneUpdateFrequency method will return the UpdateFrequency of screen. See SetScreenUpdateFrequency method for more details.

Example: `doTNTCommand('Viewer().GetScreenUpdateFrequency()');`

3.43.1.5. SetScreenUpdateFrequency

Syntax: SetScreenUpdateFrequency(frequencyValue:integer)

The SetScreenUpdateFrequency method will sets the UpdateFrequency value in the Viewer. The UpdateFrequency value will determine the number of screen update per second. As default the UpdateFrequency value -1 when means that the number of updates by the screen will be the same as the screen refresh rate. If the UpdateFrequency is set to zero then the screen will not update at all with a time interval. Settings the UpdateFrequency to zero can be used full if you wish to suspend the TurnTool Viewer. Doing this will make sure that the TurnTool Viewer only use veyr little processor time and this releases more processor time for other things.

Example: `doTNTCommand('Viewer().SetScreenUpdateFrequency(0)');`

This example will set the UpdateFrequency of the screen to 0.

3.44.Downloader

3.44.1.1. Enqueue

Syntax: Enqueue(sourceURL:string)

The Enqueue method will when call create a downloadItem and put it in the download queue. The sourceURL must point to the information that you wish to download. The downloaded file will when fully download be located in the temporary folder in your harddrive. When the information at this sourceURL is fully downloaded the OnFileDownloadComplete event will be called.
Example:

```
doTNTCommand('Downloader().Enqueue("http://www.turntool.com/TurnTool2011/download_test/index.tnt")');
```

This example puts a downloadItem which downloades from "http://www.turntool.com/TurnTool2011/download_test/index.tnt" into the download queue.

3.44.1.2. Cancel

Syntax: Cancel()

The Cancel method will Cancel all downloads that are already in the download queue.
Example: `doTNTCommand('Downloader().Cancel()');`

3.44.1.3. ShowProgressBar

Syntax: ShowProgressBar(show:integer)

The ShowProgressBar method will set the value of the ShowProgressBar integer in the Downloader. If the ShowProgressBar integer is zero then the Progress bar will be hidden. If the ShowProgressBar integer is '1' then the Progress bar will shown while downloading the files in the download queue.
Example: `doTNTCommand('Downloader().ShowProgressBar(0)');`
This example will hide the progressbar when downloading.

3.44.1.4. SetDownloadText

Syntax: SetDownloadText(text:string)

The SetDownloadText method sets the download text, which is displayed on the progressbar when downloading a file.
Example: `doTNTCommand('Downloader().SetDownloadText("Downloading File")');`
This example will set the text displayed while downloading to "Downloading File".

3.44.1.5. ClearCacheForFile

Syntax: ClearCacheForFile(fileName:string)

The ClearCacheForFile method will clear the cache for a file that has already been downloaded. The value of the fileName string is the filename of the file that is cleared in the temporary folder.
Example: `doTNTCommand('Downloader().ClearCacheForFile("turntool.tnt")');`
This example clears the cache for the file called "turntool.tnt".