Contents

Glossary

`[name]` is case sensitive. It include the whole path of the object (for example: `tn/si/bd/ro/rk`)
`[color]` is a hexadecimal code (*ffffff*)

Variables

Set a variable:

```
.var:[name]=[value]
```

Use a variable:

```
${[name]} or $[name] (in that case the longest identifier is used as [name])
```

Comments

You can put comments in an .ocli file with the `//` indicator.

```
// This is a comment
+tn:example@ffffff // This is another comment
```

Loading commands

Load commands from a text file

`[path]` path of the file

```
.cmds:[path]
```

Load template from JSON

`[path]` path of the json file

```
.template:[path]
```

Hierarchy commands

Select an object

If `[name]` is empty, go back to root

```
=[name]
```

Select child / children object

Select one or several children of current selected object.
`[relativeName]` is the hierarchy name without the selected object part

```
={[relativeName]}
={[relativeName],[relativeName],...}
```

Select parent object

```
..
```

Delete object

Works with single or multi selection.

```
-[name]
-selection
```

Focus an object

If `[name]` is empty, unfocus all items

```
>[name]
```

Create commands

Create a Tenant

Tenant will be created as a new root.

```
+tenant:[name]@[color]
+tn:[name]@[color]
```

Create a Site

Site must be child of a Tenant.

```
+site:[name]
+si:[name]
```

Create a Building

Building must be child of a Site.
`[pos]` is a Vector2 [x,y] (m,m)
`[rotation]` is the rotation of the building around its lower left corner, in degree
`[size]` is a Vector3 [width,length,height] (m,m,m)
`[template]` is the name (slug) of the building template

```
+building:[name]@[pos]@[rotation]@[size]
+building:[name]@[pos]@[rotation]@[template]
+bd:[name]@[pos]@[rotation]@[size]
+bd:[name]@[pos]@[rotation]@[template]
```

Create a Room

Room must be child of a building.
Its name will be displayed in the center of the room in its local coordinates system.
`[pos]` is a Vector2 [x,y] (m,m)
`[rotation]` is the rotation of the building around its lower left corner, in degree
`[size]` is a Vector3 [width,length,height] (m,m,m)
`[axisOrientation]` defines the orientation of the rows and columns. It can be any combinason of [+/-]x[+/-]y. eg: +x+y or -x+y
`[template]` is the name of the room template
`[floorUnit]` is optionnal: by default set to "t" (tiles), can also be m (meters) or f (feet)

```
+room:[name]@[pos]@[rotation]@[size]@[axisOrientation]@[floorUnit]
+room:[name]@[pos]@[rotation]@[template]
+ro:[name]@[pos]@[rotation]@[size]@[axisOrientation]@[floorUnit]
+ro:[name]@[pos]@[rotation]@[template]
```

Create a Rack

Rack must be child of a room.
`[pos]` is a Vector2 [x,y] (tile,tile) or a Vector3 [x,y,z] (tile,tile,cm) if the rack is wall mounted. It can be decimal or fraction. Can also be negative
`[unit]` is t(tiles), m(meters) or f(feet)
`[rotation]` is a Vector3 of angles or one of following keywords :
"front": [0, 0, 180]
"rear": [0, 0, 0]
"left": [0, 90, 0]
"right": [0, -90, 0]
"top": [90, 0, 0]
"bottom": [-90, 0, 0]
`[size]` is a Vector3 [width,length,height] (cm,cm,u)
`[template]` is the name of the rack template

```
+rack:[name]@[pos]@[unit]@[rotation]@[size]
+rack:[name]@[pos]@[unit]@[rotation]@[template
+rk:[name]@[pos]@[unit]@[rotation]@[size]
+rk:[name]@[pos]@[unit]@[rotation]@[template]
```

Create a Device

A chassis is a *parent* device racked at a defined U position.
`[posU]` is the position in U in a rack
`[sizeU]` is the height in U in a rack
`[slot]` is the name of the slot in which you want to place the device
`[template]` is the name of the device template
`[side]` is from which side you can see the device if not "fullsize". This value is for overriding the one defined in the template. It can be front | rear | frontflipped | rearflipped
If the parent rack doesn't have slots:

```
+device:[name]@[posU]@[sizeU]
+device:[name]@[posU]@[template]
```

If the parent rack has slots:

```
+device:[name]@[slot]@[sizeU]
+device:[name]@[slot]@[template]
```

All other devices (blades / components like processor, memory, adapters, disks...) have to be declared with a parent's slot and a template.

```
+device:[name]@[slot]@[template]
+device:[name]@[slot]@[template]@[side]
+dv:[name]@[slot]@[template]
+dv:[name]@[slot]@[template]@[side]
```

Create a Group

Group must be child of a room or a rack A group is a box containing all given children.

- If the group is a child of a room, it can contain racks and corridors.
- If the group is a child of a rack, it can contain devices.

`c1,c2,...,cN` are the short names (eg. A01 instead of tn.si.bd.ro.A01)

```
+group:[name]@{c1,c2,...,cN}
+gr:[name]@{c1,c2,...,cN}
```

Create a Corridor

Corridor must be child of a room A corridor is a cold or warm corridor.
`[pos]` is a Vector2 [x,y] (tile,tile) or a Vector3 [x,y,z] (tile,tile,cm) if the corridor is wall mounted. It can be decimal or fraction. Can also be negative
`[unit]` is t(tiles), m(meters) or f(feet)
`[rotation]` is a Vector3 of angles or one of following keywords :
"front": [0, 0, 180]
"rear": [0, 0, 0]
"left": [0, 90, 0]
"right": [0, -90, 0]
"top": [90, 0, 0]
"bottom": [-90, 0, 0]
`[size]` is a Vector3 [width,length,height] (cm,cm,u)
`[temperature]` is cold or warm.

```
+corridor:[name]@[pos]@[unit]@[rotation]@[size]@[temperature]
+co:[name]@[pos]@[unit]@[rotation]@[size]@[temperature]
```

Set commands

Set colors for zones of all rooms in a datacenter

```
[datacenter]:usableColor=[color]
[datacenter]:reservedColor=[color]
```

```
[datacenter]:technicalColor=[color]
```

Set reserved and technical zones of a room

Enables tiles edges display.
You can modify areas only if the room has no racks in it.
Technical area : typically a restricted zone where power panels and AC systems are installed. separated from "IT space" with either a wall or a wire mesh
Reserved area : some tiles around the room that must be kept free to move racks and walk (usually 2 or 3 tiles)

`[reserved]` is a vector4: [front,back,right,left] (tile,tile,tile,tile)
`[technical]` is a vector4: [front,back,right,left] (tile,tile,tile,tile)

```
[room]:areas=[reserved]@[technical]
```

Add a separator to a room

Add a separator (wired or plain wall) inside a room.
`[name]` is an identifier for the separator
`[startPos]` is a vector2: [x,y] (m,m)
`[endPos]` is a vector2: [x,y] (m,m)
`[type]` is the type of wall: wireframe or plain

```
[room]:separator=[name]@[startPos]@[endPos]@[type]
```

It will add the given coordinates to `[room].attributes["separators"]` witch is a list of all its separators parameters

Add a pillar to a room

Add a pillar inside a room.
`[name]` is an identifier for the pillar
`[centerXY]` is a vector2: [x,y] (m,m)
`[sizeXY]` is a vector2: [x,y] (m,m)
`[rotation]` is the angle of the pillar, in degree

```
[room]:pillar=[name]@[centerXY]@[sizeXY]@[rotation]
```

It will add the given coordinates to `[room].attributes["pillars"]` witch is a list of all its pillars parameters

Modify object's attribute

Works with single or multi selection.
`[name]` can be `selection` or `_` for modifying selected objects attributes

```
[name]:[attribute]=[value]

selection:[attribute]=[value]
_:[attribute]=[value]
```

- Object's domain can be changed recursively for changing all it's children's domains at the same time.

```
[name]:domain=[value]@recursive
```

- Object's description attribute is a list: you have to use an index to fill one.

```
[name]:description1=[value]
[name]:description[N]=[value] where [N] is an index, starting at 1
```

Labels

Some objects have a label displayed in there 3D model: racks, devices, rack groups and corridors.
The default label is the object's name.

Choose Label

You can change the label by a string or with a choosen attribute:
`#[attribute]` is one of the attribute of the object. If `description`, it will display all descriptions. To display a specific description, use `description[N]` where N is the index of the wanted description.

```
[name]:label=#[attribute]
[name]:label=[string]
```

Modify label's font

You can make the font bold, italic or change its color.

```
[name]:labelFont=bold           //will toggle bold
[name]:labelFont=italic         //will toggle bold
[name]:labelFont=color@[color]
```

Modify label's background color

You can change the label's background color when it is hovering over the object.

```
[name]:labelBackground=[color]
```

Interact with objects

The same way you can modify object's attributes, you can interact with them through specific commands.

Room

- Display or hide tiles name

```
[name]:tilesName=[true|false]
```

- Display or hide colors and textures

```
[name]:tilesColor=[true|false]
```

Rack

- Display or hide rack's box. This will also affect its label

```
[name]:alpha=[true|false]
```

- Display or hide rack's U helpers to simply identify objects in a rack.

```
[name]:U=[true|false]
```

- Display or hide rack's slots

```
[name]:slots=[true|false]
```

- Display or hide rack's local coordinate system

```
[name]:localCS=[true|false]
```

Device

- Display or hide device's box. This will also affect its label

```
[name]:alpha=[true|false]
```

- Display or hide device's slots

```
[name]:slots=[true|false]
```

- Display or hide device's local coordinate system

```
[name]:localCS=[true|false]
```

Group

- Display or hide contained racks/devices

```
[name]:content=[true|false]
```

Manipulate UI

Delay commands

You can put delay before each command: up to 2 seconds.

```
ui.delay=[time]
```

Display infos panel

```
ui.infos=[true|false]
```

Display debug panel

```
ui.debug=[true|false]
```

Highlight object

This is a "toggle" command: use it to turn on/off the highlighting of an object.
If given object is hidden in its parent, the parent will be highlighted.

```
ui.highlight=[name]
ui.hl=[name]
```

Manipulate camera

Move camera

Move the camera to the given point.
`[position]` *is a Vector3: the new position of the camera*
`[rotation]` *is a Vector2: the rotation of the camera*

```
camera.move=[position]@[rotation]
```

Translate camera

Move the camera to the given destination. You can stack several destinations, the camera will move to each point in the given order.
`[position]` *is a Vector3: the position of the camera's destination*
`[rotation]` *is a Vector2: the rotation of the camera's destination*

```
camera.translate=[position]@[rotation]
```

Wait between two translations

You can define a delay between two camera translations.
`[time]` *is the time to wait in seconds*

```
camera.wait=[time]
```

Examples

```
+tn:DEMO@ffffff
    DEMO.mainContact=Ced
    DEMO.mainPhone=0612345678
    DEMO.mainEmail=ced@ogree3D.com
```

```
+tn:Marcus@42ff42
    Marcus.mainContact=Marcus Pandora
    Marcus.mainPhone=0666666666
    Marcus.mainEmail=marcus@pandora.com

+tenant:Billy@F0C300

+si:DEMO.ALPHA@NW
    DEMO.ALPHA.description=This is a demo...
    DEMO.ALPHA.address=1 rue bidule
    DEMO.ALPHA.zipcode=42000
    DEMO.ALPHA.city=Truc
    DEMO.ALPHA.country=FRANCE
    DEMO.ALPHA.gps=[1,2,0]
    DEMO.ALPHA.usableColor=5BDCFF
    DEMO.ALPHA.reservedColor=AAAAAA
    DEMO.ALPHA.technicalColor=D0FF78

// Building A

+bd:DEMO.ALPHA.A@[0,0,0]@[12,12,5]
    DEMO.ALPHA.A.description=Building A
    DEMO.ALPHA.A.nbFloors=1
+ro:DEMO.ALPHA.A.R0_EN@[6,6,0]@[4.2,5.4,1]@EN
+ro:DEMO.ALPHA.A.R0_NW@[6,6,0]@[4.2,5.4,1]@NW
+ro:DEMO.ALPHA.A.R0_WS@[6,6,0]@[4.2,5.4,1]@WS
+ro:DEMO.ALPHA.A.R0_SE@[6,6,0]@[4.2,5.4,1]@SE

+rk:DEMO.ALPHA.A.R0_EN.TEST_EN@[ 1,1]@[60,120,42]@front
+rk:DEMO.ALPHA.A.R0_NW.TEST_NW@[1 ,1]@[60,120,42]@front
+rk:DEMO.ALPHA.A.R0_WS.TEST_WS@[1, 1]@[60,120,42]@front
+rk:DEMO.ALPHA.A.R0_SE.TEST_SE@[1,1 ]@[60,120,42]@front

// Building B

+bd:DEMO.ALPHA.B@[-30,10,0]@[25,29.4,5]
    DEMO.ALPHA.B.description=Building B
    DEMO.ALPHA.B.nbFloors=1

+ro:DEMO.ALPHA.B.R1@[0,0,0]@[22.8,19.8,4]@NW
    DEMO.ALPHA.B.R1.areas=[2,1,5,2]@[3,3,1,1]
    DEMO.ALPHA.B.R1.description=First room

+ro:DEMO.ALPHA.B.R2@[22.8,19.8,0]@[9.6,22.8,3]@WS
    DEMO.ALPHA.B.R2.areas=[3,1,1,3]@[5,0,0,0]
    DEMO.ALPHA.B.R2.description=Second room, owned by Marcus
    DEMO.ALPHA.B.R2.tenant=Marcus

// Racks for R1

+rk:DEMO.ALPHA.B.R1.A01@[1,1]@[60,120,42]@front
    DEMO.ALPHA.B.R1.A01.description=Rack A01
    DEMO.ALPHA.B.R1.A01.vendor=someVendor
    DEMO.ALPHA.B.R1.A01.type=someType
```

```
    DEMO.ALPHA.B.R1.A01.model=someModel
    DEMO.ALPHA.B.R1.A01.serial=someSerial

+rk:DEMO.ALPHA.B.R1.A02@[2,1]@[60,120,42]@front
+rk:DEMO.ALPHA.B.R1.A03@[3,1]@[60,120,42]@front
+rk:DEMO.ALPHA.B.R1.A04@[4,1]@[60,120,42]@front
+rk:DEMO.ALPHA.B.R1.A05@[5,1]@[60,120,42]@front
    DEMO.ALPHA.B.R1.A05.tenant=Billy

+rk:DEMO.ALPHA.B.R1.B05 @[8,6] @[60,120,42]@rear
+rk:DEMO.ALPHA.B.R1.B09 @[9,6] @[60,120,42]@rear
+rk:DEMO.ALPHA.B.R1.B010@[10,6]@[60,120,42]@rear
+rk:DEMO.ALPHA.B.R1.B011@[11,6]@[60,120,42]@rear
+rk:DEMO.ALPHA.B.R1.B012@[12,6]@[60,120,42]@rear

+rk:DEMO.ALPHA.B.R1.C08 @[8,9] @[60,120,42]@front
+rk:DEMO.ALPHA.B.R1.C09 @[9,9] @[60,120,42]@front
+rk:DEMO.ALPHA.B.R1.C010@[10,9]@[60,120,42]@front
+rk:DEMO.ALPHA.B.R1.C011@[11,9]@[60,120,42]@front
+rk:DEMO.ALPHA.B.R1.C012@[12,9]@[60,120,42]@front

+rk:DEMO.ALPHA.B.R1.D01@[20,5]@[60,120,42]@left
    DEMO.ALPHA.B.R1.D01.tenant=Marcus
+rk:DEMO.ALPHA.B.R1.D02@[20,6]@[60,120,42]@left
    DEMO.ALPHA.B.R1.D02.tenant=Marcus
+rk:DEMO.ALPHA.B.R1.D03@[20,7]@[60,120,42]@left
    DEMO.ALPHA.B.R1.D03.tenant=Marcus

+rk:DEMO.ALPHA.B.R1.E01@[23,5]@[60,120,42]@right
    DEMO.ALPHA.B.R1.E01.tenant=Marcus
+rk:DEMO.ALPHA.B.R1.E02@[23,6]@[60,120,42]@right
    DEMO.ALPHA.B.R1.E02.tenant=Marcus
+rk:DEMO.ALPHA.B.R1.E03@[23,7]@[60,120,42]@right
    DEMO.ALPHA.B.R1.E03.tenant=Marcus

// Racks for R2

+rk:DEMO.ALPHA.B.R2.A01@[1,3]@[60,120,42]@rear
+rk:DEMO.ALPHA.B.R2.A02@[2,3]@[60,120,42]@rear
+rk:DEMO.ALPHA.B.R2.A03@[3,3]@[60,120,42]@rear
+rk:DEMO.ALPHA.B.R2.A04@[4,3]@[60,120,42]@rear
+rk:DEMO.ALPHA.B.R2.A05@[5,3]@[60,120,42]@rear

+rk:DEMO.ALPHA.B.R2.B01@[1,5]@[60,120,42]@front
    DEMO.ALPHA.B.R2.B01.tenant=Billy
    DEMO.ALPHA.B.R2.B01.alpha=50

// Edit description of several racks in R1
={B05,B09,B10,B11,B12}
selection.description=Row B
```