

Open Command-oriented Geometric Graphics Generator

OpenCG³ Spec Version 0.2.5

Dong Nai-Jia ¹ Lin Yong-Hsiang ²

¹National Chiao Tung University
Department of Computer Science

²National Taiwan University
Department of Agricultural Chemistry

August 16, 2017

Command Tokens

Regular Expressions

$$\mathbb{N} := \{ \alpha \mid \alpha \in [0-9]^+ \}$$
$$\mathbb{R} := \{ \alpha \mid \alpha \in [+|-]?([0-9]*[.]?)?[0-9]^+ \}$$
 $\Rightarrow \mathbb{R} \supset \mathbb{N}$
$$\mathbb{S} := \{ \alpha \mid \alpha \in '(. * ?)' \mid [.0-9A-Za-z+|-]^+ \}$$
 $\Rightarrow \mathbb{S} \supset \mathbb{R}$
$$\mathbb{W} := \{ \alpha \mid \alpha \in [\backslash t] \}$$

whitespace

Descriptions

- The matching mechanism abides by the maximal munch rule.
- Each command is whitespace-insensitive except being quoted by a pair of single quotation marks (').

Command Grammars

Context-Free Expansions

$\mathbf{C} \rightarrow \mathbf{AC} \mid ; \mid \text{EOL}$

$\mathbf{A} \rightarrow \mathbf{T(A)} \mid \mathbf{V(A)} \mid \mathbf{S(A)} \mid \mathbf{L(A)} \mid \mathbf{L(A, A, \dots, A)} \mid \mathbf{N} \mid \mathbf{R} \mid \mathbf{S}$

$\mathbf{T(\Pi)} \equiv \Pi : n \rangle \rightarrow (\Sigma(\Pi, n)) \quad \left\| \quad \Sigma(\Pi, n) \rightarrow \underbrace{\Pi \dots \Pi}_{n \text{ items}} \quad (\text{identical})$

$\mathbf{V(\Pi)} \equiv \Pi : n \rangle \rightarrow < \Sigma(\Pi, n) >$

$\mathbf{S(\Pi)} \equiv \Pi : n \} \rightarrow \{ \Sigma(\Pi, n) \} \quad \left\| \quad \mathbf{L(\Pi)} \equiv \mathbb{L} [\Pi : n] \rightarrow [\Sigma(\Pi, n)]$

$\mathbf{L(\Pi_1, \Pi_2, \dots, \Pi_{n-1}, \Pi_n)} \equiv \mathbb{L} [\Pi_1 \Pi_2 \dots \Pi_{n-1} \Pi_n] \rightarrow [\Pi_1 \dots \Pi_n]$

Descriptions

- Each command starts from \mathbf{C} and ends with a $;$ or an EOL.
- Non-terminal symbol expansions are prior than function expansions except that symbols are used for describing argument types of a command.

Command Parsing

Escape Sequence

- `\x` is an escape sequence.
- If `x` is `\`, then it is treated as a single backslash.
- If `x` is EOL which may vary from platforms, then the sequence is omitted.
- Otherwise, the sequence is ignored and triggers a warning by default.

Error Handling

- Physical lines are separated by an EOL.
- Logical lines are separated by either a semicolon or an unescaped EOL.
- If the command cannot be parsed by the grammar, then all the characters on the same logical line will be discarded.

Class and Object System

Classes

- Classes are split into two categories, top and bottom.
- Top classes consist of window, camera, point, line, surface, etc.
- Bottom classes consist of attrib(ute) and group.

Objects

- An object is derived from a class aforementioned.
- An object has an unique name throughout its class category derivations.

Relations

- Objects derived from the same class category cannot form a relation.
- Relations are bidirectional and can be created or deleted via commands.

Create a Window

Command

```
create window S label  $\mathbb{R} : 3$  coord  $\mathbb{R} : 3$   $\rangle : 3$  direct (1)
```

Parametres

- label: the object name of the class window
- coord: the coordinate (c_x, c_y, c_z) of the centre of the window.
- direct: the window width v_w^{\rightarrow} , height v_h^{\rightarrow} , and the camera view v_c^{\rightarrow} .

Examples

```
create window main (0 0 1) (<1 0 0> <0 1 0> <0 0 1>)
```

Delete a Window

Command

`delete window $ message` (2)

Parametres

- message: the text string printed right after exit

Examples

```
delete window  
delete window 'Have a nice day.'
```

Create Points

Command

```
create point  S label : }   $\mathbb{R} : 3$  coord (3)
create point  S label :  $\geq n$   $\mathbb{R} : 3$  coord : n (4)
```

Parametres

- label: the object name of the class point
- coord: the coordinate (p_x, p_y, p_z) of the point

Examples

```
create point 'origin' (0 0 0)
create point {X-1 X-2} (1 0 0)
create point (Y-1 Z-1) ((0 1 0)(0 0 1))
```


Delete Points

Command

```
delete point  $ label : } (5)
```

Parametres

- label: the object name of the class point

Examples

```
delete point  origin  
delete point {origin 'random-point'}
```

Create Attributes

Command

```
create attrib S palette : } L [ L [ S type S key A value ] : ] (6)
```

```
create attrib S palette : ) L [ L [ S type S key A value ] : ] (7)
```

Parametres

- palette: the object name of the class attrib
- type: the name of the class in the top class category
- key: the property of the object of class type
- value: the appropriate value of the property key

Examples

```
create attrib (magenta dashed-and-traslucent-green) \
[[point fill-hsv (300 1.0 1.0)] \
[line style dashed] [line fill-rgba [(0 255 0) .5]]]
```

Attach Attributes

Command

```
attach attrib $ palette : $ label : } (8)
```

```
attach attrib $ palette : $ label : } (9)
```

Parametres

- palette: the object name of the class attrib
- label: the name of the object derived from the top class category

Examples

```
attach attrib red point-0
attach attrib (red large) point-1
attach attrib blue {point-2 rect-0}
attach attrib (5px black) {point-3 circ-0}
attach attrib (red thick) (point-4 line-0 trianle-0)
```

Assign an Operation Name

Command

`assign opname $ action $ type N repeat [= ∞]` (10)

Parametres

- action: the name of the action
- type: the object of the class type applying the action
- repeat: the amount of the commands emitting operation names

Examples

```
assign instr create point 2
x-axis (1 0 0); y-axis (0 1 0)
// Back To Normal
```