Open Command-oriented Geometric Graphics Generator

OpenCG^3 Spec Version 0.2.6

Dong Nai-Jia [1] Lin Yong-Hsiang [2]

[1] National Chiao Tung University
Department of Computer Science

[2] National 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]+` $\}$ $\Rightarrow \mathbb{R} \supset \mathbb{N}$

$\mathbb{S} := \{\ \alpha \mid \alpha \in$ `'(.*?)'|[.0-9A-Za-z+\-]+` $\}$ $\Rightarrow \mathbb{S} \supset \mathbb{R}$

$\mathbb{W} := \{\ \alpha \mid \alpha \in$ `[ \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

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

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

$$\left.\begin{array}{l} \mathbf{T}(\Pi) \equiv \Pi : n\,) \rightarrow \texttt{(}\ \Sigma(\Pi, n)\ \texttt{)} \\ \mathbf{V}(\Pi) \equiv \Pi : n\rangle \rightarrow \texttt{<}\ \Sigma(\Pi, n)\ \texttt{>} \\ \mathbf{S}(\Pi) \equiv \Pi : n\} \rightarrow \texttt{\{}\ \Sigma(\Pi, n)\ \texttt{\}} \end{array}\right\| \quad \Sigma(\Pi, n) \rightarrow \overbrace{\Pi\ \cdots\ \Pi}^{n \text{ items}} \quad \text{(identical)}$$

$$\mathbf{L}(\Pi) \equiv \mathbb{L}\,[\,\Pi : n\,] \rightarrow \texttt{[}\ \Sigma(\Pi, n)\ \texttt{]}$$

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

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 arguments 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` $\mathbb{S}$ label $\mathbb{R}:3)$ coord $\mathbb{R}:3\rangle:3)$ dirct (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.
- dirct: the window width $\vec{v_w}$, height $\vec{v_h}$, and the camera view $\vec{v_c}$.

Examples

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

Delete a Window

Command

`delete window` $\mathbb{S}$ message (2)

Parametres

- message: the text string printed right after exiting

Examples

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

Create Points

Command

`create point` $\underline{\mathbb{S}\ \underline{\text{label}} : \}}\ \underline{\mathbb{R} : 3)\ \underline{\text{coord}}}$ (3)

`create point` $\underline{\mathbb{S}\ \underline{\text{label}} :\geqslant n)}\ \underline{\mathbb{R} : 3)\ \underline{\text{coord}} : n)}$ (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` $\mathbb{S}$ 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 desc : } 𝕃[ 𝕃[ 𝕊 t-class 𝕊 key A value ] : ]   (6)
create attrib 𝕊 desc : ) 𝕃[ 𝕃[ 𝕊 t-class 𝕊 key A value ] : ]   (7)
```

Parametres

- desc: the object name of the class attrib
- t-class: the name of one of the top classes
- key: the property of the object of class t-class
- 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` $\underline{\mathbb{S}\ \text{desc}\ :\ )}\ \underline{\mathbb{S}\ \text{label}\ :\ \}}$ (8)

`attach attrib` $\underline{\mathbb{S}\ \text{desc}\ :\ )}\ \underline{\mathbb{S}\ \text{label}\ :\ )}$ (9)

Parametres

- desc: the name of the object of the class attrib
- label: the name of the object derived from the top classes

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` $\underline{\mathbb{S}\ \text{action}}$ $\underline{\mathbb{S}\ \text{class}}$ $\underline{\mathbb{N}\ \text{repeat}\ [=\infty]}$ (10)

Parametres

- action: the name of the action
- class: the name of one of the classes
- 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
```