Open Command-oriented Geometric Graphics Generator

OpenCG[3] Specification Version 0.2.12

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 19, 2017

Regular Expressions

$\mathbb{N} := \{\ \alpha\ |\ \alpha \in \texttt{[0-9]+}\ \}$

$\mathbb{R} := \{\ \alpha\ |\ \alpha \in \texttt{[+\textbackslash-]?([0-9]*[.])?[0-9]+}\ \}$ \hfill $\Rightarrow \mathbb{R} \supset \mathbb{N}$

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

$\mathbb{W} := \{\ \alpha\ |\ \alpha \in \texttt{[\textbackslash t]}\ \}$ \hfill 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

$\mathbf{C} \to \mathbf{A}\,\mathbf{C} \mid \texttt{;} \mid \texttt{EOL}$

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

$\mathbf{T}(\Pi) \equiv \Pi : n\,) \to (\ \Sigma(\Pi, n) \ )$

$\mathbf{V}(\Pi) \equiv \Pi : n\rangle \to \texttt{<} \ \Sigma(\Pi, n) \ \texttt{>}$ $\quad \left\| \quad \Sigma(\Pi, n) \to \overbrace{\Pi \ \cdots \ \Pi}^{n \text{ items}} \quad \text{(identical)} \right.$

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

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

Descriptions

- Each command starts from $\mathbf{C}$ and ends with a ; or an EOL.
- Non-terminal symbol expansions are prior than function expansions.

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.

Fields, Classes, Objects and References

Definitions

- The whole system are divided into four fields and several classes:
  1. field e-(nviron.):   includes class window and class camera.
  2. field p-(rimitive):   includes class point, class circle, etc.
  3. field c-(ompound): includes class line, class triangle, class polygon, etc.
  4. field a-(uxiliary):   includes class attrib and class group.

Notations

- $class^x$ denotes the name of the class in the field x.

- $label^x$ denotes the unique name of the object from the class in the field x.

Prototypes

- Argument prototypes are written in a mixture of types and names with underlines.

- Each type with an asterisk indicates that the brackets are used for cross-referencing.

- Cross-reference is a feature for manipulating multitple objects in a single command.

- Each name with a plus/minus/ampersand implies that the given name is used for creating new objects/deleting existed objects/cross-referencing among objects, etc.

Figure: Projection in Euclidean $\mathbb{R}^3$ Space
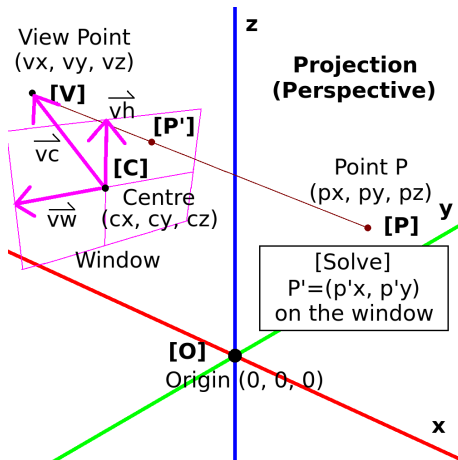
Create a Window

Command

`create window` $\underline{\mathbb{S}}$ $\underline{+ label^e}$ (1)

Parametres

- $\underline{label^e}$: the name of the object instantiated from the class window

Examples

`create window main`

Delete a Window

Command

`delete window` $\underline{\mathbb{S} - \text{label}^e}$ $\mathbb{S}$ $\underline{\text{string}}$ (2)

Parametres

- $\underline{\text{label}^e}$: the name of the object instantiated from the class window
- $\underline{\text{string}}$: the text printed right after exiting the session

Examples

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

Create a Camera

Command

```
create camera  𝕊 +label^e  ℝ : 3)  centre  ℝ : 3⟩ : 2)  plane  ℝ : 3⟩  sight        (3)
```

Parametres

- label[e] : the name of the object instantiated from the class camera
- centre : the world coordinate (c_x, c_y, c_z) of the centre of the viewport
- plane : the horizontal and the vertical vertors $(\vec{v_w}, \vec{v_h})$ of the viewport
- sight : the reverse line of sight $\vec{v_c}$ from centre to the camera

Examples

```
create camera z-top (0 0 1) (<1 0 0> <0 1 0>) <0 0 1>
```

Select a Camera

Command

`select camera` $\underline{\mathbb{S} \ \& \, \text{label}_1^e}$ $\underline{\mathbb{S} \ \& \, \text{label}_2^e}$ (4)

Parametres

- $\underline{\text{label}_1^e}$: the name of the object instantiated from the class camera
- $\underline{\text{label}_2^e}$: the name of the object instantiated from the class window

Examples

```
select camera z-top main
```

Create Points

Command

```
create point   𝕊 + labelᵖ : }⋆     ℝ : 3) coord                    (5)
create point   𝕊 + labelᵖ :⩾ n)⋆   ℝ : 3) coord : n)⋆             (6)
```

Parametres

- $\underline{label^p}$: the name of the object instantiated from the class point
- $\underline{coord}$: the world coordinate (p_x, p_y, p_z) of the object named $\underline{label^p}$

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   𝕊 – label^p : }*                                              (7)
```

Parametres

- label^p   : the name of the object instantiated from the class point

Examples

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

Create Attributes

Command

```
create attrib  S + label^a : }^*  L [ L [ S class^{p,c} L [ S prop S value : ] ] : ]^*   (8)
create attrib  S + label^a : )^*  L [ L [ S class^{p,c} L [ S prop S value : ] ] : ]^*   (9)
```

Parametres

- $\underline{label^a}$: the name of the object instantiated from the class attrib
- $\underline{class^{p,c}}$: the name of a class in the field primitive or compound
- $\underline{prop}$: the property of the object of $\underline{class^{p,c}}$
- $\underline{value}$: the value of $\underline{prop}$ in designated format

Examples

```
create attrib (magenta dashed-and-translucent-line) \
[[point fill-hsv '(300 1.0 1.0)'] \
 [line [style dashed] [fill-rbga '[(0 255 0) 0.5]']]]
```

Attach Attributes

Command

```
attach attrib  S & label^a : )*   S & label^{p,c} : }*                    (10)
attach attrib  S & label^a : )*   S & label^{p,c} : )*                    (11)
```

$$\texttt{attach attrib } \underline{\mathbb{S} \ \& \, \text{label}^a} : \,)^\star \quad \underline{\mathbb{S} \ \& \, \text{label}^{p,c}} : \, \}^\star \tag{10}$$

$$\texttt{attach attrib } \underline{\mathbb{S} \ \& \, \text{label}^a} : \,)^\star \quad \underline{\mathbb{S} \ \& \, \text{label}^{p,c}} : \,)^\star \tag{11}$$

Parametres

- $\underline{\text{label}^a}$: the name of the object instantiated from the class attrib
- $\underline{\text{label}^{p,c}}$: the name of the object instantiated from a class in the field p or c

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 triangle-0)
```

Assign Operations

Command

```
assign operat  𝕊 action  𝕊 class  ℕ repeat [= ∞]                    (12)
```

Parametres

- **action** : the name of the corresponding action of class
- **class** : the name of a class
- **repeat** : the amount of the commands emitting operation names

Examples

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