SUICA LIBRARY 1.12

A. Functions and variables 3

Initialization 3

Version of the library (version) 3
Total elapsed time (time) 3
Elapsed time (dTime) 3
Initialization of the graphics area (Suica) 3
Background colour (background) 3

Viewpoint 4

Orthographic projection (orthographic) 4
Perspective projection (perspective) 4
Viewpoint (lookAt) 4
Screen position (getPosition) 5

Help functions 5

Coordinate system (oxyz) 5
Automatic rotation (demo) 5
Object cloning (sameAs) 6
Random number (random) 6
Degrees to radians (radians) 6

Functions for vectors 6

Unit vector (unitVector) 6
Subtraction of vectors or points (vectorPoints) 6
Scalar product (scalarProduct) 7
Vector product (vectorProduct) 7

B. General graphical properties 8

Position 8

Center (center) 8
Initial offset (origin) 8
Screen position (getPosition) 8

Size 8

Size of point (pointSize) 8
Size of object (size) 9
Sizes of object (sizes) 9
Radius of object (radius) 9
Radii of object (radii) 9

Orientation 9

Direction (focus) 9
Rotation (spin) 9

Structure 10

Number of sides or faces (count) 10
Hollowness (hollow) 10

Object clipping (clipPlanes) ... 10
Colour and lights ... 10
Colour (color) ... 10
Object visibility (visible) ... 10
Drawing mode (mode) ... 11
Illumination (light) ... 11
Miscallaneous ... 11
Style (custom) ... 11
Merged objects (merge) ... 11
Merged colour (mergeColor) ... 11
Animation loop (nextFrame) ... 12
Interactivity (interactive) ... 12

C. GraphicAL objects ... 13

Zero-dimensional ... 13
Point (point) ... 13
One-dimensional ... 13
Line (line) ... 13
Ray (ray) ... 13
Segment (segment) ... 14
Two-dimensional ... 14
Square (square) ... 14
Rectangle (rectangle) ... 14
Regular polygon (polygon) ... 14
Circle (circle) ... 14
Ellipse (ellipse) ... 15
Three-dimensional ... 15
Cube (cube) ... 15
Rectangular parallelepiped (cuboid) ... 15
Sphere (sphere) ... 15
Spheroid (spheroid) ... 15
Regular prism (prism) ... 16
Regular pyramid (pyramid) ... 16
Cylinder (cylinder) ... 16
Elliptical cylinder (cylindroid) ... 16
Cone (cone) ... 16
Elliptical cone (conoid) ... 17

A. FUNCTIONS AND VARIABLES

INITIALIZATION

VERSION OF THE LIBRARY (VERSION)

*Suica.**version***

Variable. Contains string with the Suica version number. The variable is in the *Suica* class itself, not in the instance.

TOTAL ELAPSED TIME (TIME)

*Suica.**time***

Variable. Contains elapsed time in seconds since program startup. The variable is in the *Suica* class itself, not in the instance.

ELAPSED TIME (DTIME)

*Suica.**dTime***

Variable. Contains the elapsed time in seconds since the previously generated frame. The variable is in the *Suica* class itself, not in the instance.

INITIALIZATION OF THE GRAPHICS AREA (SUICA)

*new **Suica** ( )*
*new **Suica** (identifier)*

Class. Creates an instance of the *Suica* class. If an *identifier* parameter is specified, the instance associates with the canvas HTML element with the same value of its id attribute. If no parameter is specified, the first canvas element is used. If no canvas exists, a new canvas is automatically created.

The instance may not be stored in a variable unless it has to be reused later.

BACKGROUND COLOUR (BACKGROUND)

***background** (colour)*
*suica.**background** (colour)*

Procedure. Determines the background colour using the *colour* parameter. The colour is an array of 3 elements from 0 to 1.

If an instance *suica* is not provided, then the latest created instance of the class Suica is used.

VIEWPOINT

ORTHOGRAPHIC PROJECTION (ORTHOGRAPHIC)

orthographic *(from, to)*
*suica.**orthographic** (from, to)*

Procedure. Activates orthographic projection – the sizes of the graphic objects do not depend on the distance to them. The parameters *from* and *to* determine the distance to the near and far planes of the projection. Only the objects between these two planes are drawn.

If an instance *suica* is not provided, then the latest created instance of the class Suica is used.

PERSPECTIVE PROJECTION (PERSPECTIVE)

perspective *(angle,from,to)*
*suica.**perspective** (angle,from,to)*

Procedure. Enables perspective projection –the dimensions of the graphical objects depend on the distance to them. The parameters *from* and *to* determine the distance to the near and far planes of the projection. Only the objects between these two planes are drawn. The *angle* determines the horizontal viewing angle in degrees.

If an instance *suica* is not provided, then the latest created instance of the class Suica is used.

VIEWPOINT (LOOKAT)

lookAt *(position,target,up)*
*suica.**lookAt** (position,target,up)*

Procedure. Defines a current viewpoint from a 3D *position* to a specific 3D point *target* and with a given *up* direction, which is a vector. The scene is drawn so that the eye of the viewer is in *position*, the *target* is in the middle of the canvas and vector *up* points vertically up the screen.

If an instance *suica* is not provided, then the latest created instance of the class Suica is used.

SCREEN POSITION (GETPOSITION)

getPosition *(position)*
*suica.**getPosition*** *(position)*

Function . Calculates a screen position of a 3D *position* in the graphic scene. The result is an array with two coordinates – x and y of a pixel. This function is used to find the screen coordinates of a 3D position in order there to put HTML element (eg. label).

If an instance *suica* is not provided, then the latest created instance of the class Suica is used.

Graphical objects have their own *getPosition* method, which uses object's own centre instead of a *position*.

HELP FUNCTIONS

COORDINATE SYSTEM (OXYZ)

oxyz *(size)*
*suica.**oxyz*** *(size)*

Procedure. Creates anonymous point at (0,0,0) and three anonymous segments for the three axes of the coordinate system. The length of the segments is *size*.

If there is no *size* or if it is 0, then it is assumed to be 30. If an instance *suica* is not provided, then the latest created instance of the class Suica is used.

AUTOMATIC ROTATION (DEMO)

demo *( )*
demo *(distance)*
demo *(distance, speed)*
demo *(distance, speed, height)*
demo *(distance, speed, height, target)*
*suica.**demo*** *( )*
*suica.**demo*** *(distance)*
*suica.**demo*** *(distance, speed)*
*suica.**demo*** *(distance, speed, height)*
*suica.**demo*** *(distance, speed, height, target)*

Procedure. Automatically activated rotation of the viewpoint of a horizontal *distance* from the Z axis, with a certain *speed*, *height* of the viewpoint and the height of the *target*. The speed unit is 18°/s (full orbit in 20 seconds). Both heights are defined as the ratio in respect to *distance*.

The scene is drawn so that *target* appears in the middle of the canvas.

All parameters are optional. By default their values are *distance* 100, *speed* 1 (18°/s), *height* 0.3 (30 units) and *target* height 0.1 (10 units). If an instance *suica* is not provided, then the latest created instance of the class Suica is used.

OBJECT CLONING (SAMEAS)

*Suica.**sameAs** (object)*
***sameAs** (object)*

Function. Copies an object with all its properties.

RANDOM NUMBER (RANDOM)

*Suica.**random** (from,to)*
***random** (from,to)*

Function. Returns a random fractional number in the range [*from*,*to*).

DEGREES TO RADIANS (RADIANS)

*Suica.**radians** (degrees)*
***radians** (degrees)*

Function. Converts *degrees* to radians.

FUNCTIONS FOR VECTORS

UNIT VECTOR (UNITVECTOR)

***unitVector** (vector)*
*Suica.**unitVector** (vector)*

Function. Calculates a unit vector of a *vector*. The result is a vector that is parallel to *vector* but has a unit length.

The function is defined not only in the Suica class but also as a standalone function.

SUBTRACTION OF VECTORS OR POINTS (VECTORPOINTS)

***vectorPoints** (vector, vector)*
*Suica.**vectorPoints** (vector, vector)*

Function. Calculates the subtraction of two vectors or vectors between two points. The result is a vector that starts from the end of the second vector and reaches the end of the first vector.

The function is defined not only in the Suica class but also as a standalone function.

SCALAR PRODUCT (SCALARPRODUCT)

scalarProduct *(vector,vector)*
*Suica.**scalarProduct** (vector,vector)*

Function. Calculates the scalar product of two vectors. The result is a number. If it is 0, both vectors are perpendicular.

The function is defined not only in the Suica class but also as a standalone function.

VECTOR PRODUCT (VECTORPRODUCT)

vectorProduct *(vector,vector)*
*Suica.**vectorProduct** (vector,vector)*

Function. Calculates the vector product of two vectors. The result is a vector that is perpendicular to both vectors at the same time.

The function is defined not only in the Suica class but also as a standalone function.

B. GENERAL GRAPHICAL PROPERTIES

Suica graphical objects share common properties – these are properties that are identical or similar to all the objects and are used in the same way.

POSITION

CENTER (CENTER)

*object.**center***

Variable. Contains an array of coordinates of the centre of a graphic object. It is set when the object is constructed. For objects that do not have a natural centre, the *center* is used as a reference point.

INITIAL OFFSET (ORIGIN)

*object.**origin***

Variable. Contains an array of three numbers – the coordinates of the new object's centre. These coordinates are local, do not depend on the size of the object and are relative to its geometric centre.

SCREEN POSITION (GETPOSITION)

*object.**getPosition** ( )*

Function. Calculates the screen position of the centre of a graphic object. The result is an array with two coordinates – x and y of a pixel. The function is used to find where an object is displayed in order to place an HTML element there (for example, an object name tag).

Each instance of the Suica class has a *getPosition* method with a perimeter the 3D point coordinates and calculating the corresponding screen coordinates.

SIZE

SIZE OF POINT (POINTSIZE)

*object.**pointSize***

Variable. It contains a number specifying the visual radius (in pixels) of the image of a point. The default is 3 pixels. It is used when drawing a point object, as well as when drawing other graphic objects in point mode.

SIZE OF OBJECT (SIZE)

*object.**size***

Variable. Contains a number – the size of an object. Applicable to objects that have dimensions described by a single number (eg. square and cube).

SIZES OF OBJECT (SIZES)

*object.**sizes***

Variable. Contains an array of two or three numbers –object sizes. Applicable to objects that have several sizes (eg. rectangle and cuboid).

RADIUS OF OBJECT (RADIUS)

*object.**radius***

Variable. Contains a number – the radius of an object or the circular part of an object. Applicable to objects that have a circular or spherical shape (eg. a regular polygon, circle, sphere).

RADII OF OBJECT (RADII)

*object.**radii***

Variable. Contains an array of two or three numbers –the radii of an object. Applicable to objects that have several radii (eg. ellipse and spheroid).

ORIENTATION

DIRECTION (FOCUS)

*object.**focus***

Variable. Contains an array of three numbers – a vector that determines the direction of the local Z axis of an object.

ROTATION (SPIN)

*object.**spin***

Variable. Contains number – the angle in radians for rotation of an object around its local Z axis.

STRUCTURE

NUMBER OF SIDES OR FACES (COUNT)

*object.**count***

Variable. Contains an integer greater than or equal to 3. Determines the number of sides or faces for objects with a polygonal structure (eg. polygon, prism, pyramid).

HOLLOWNESS (HOLLOW)

*object.**hollow***

Variable. Contains a boolean value, false by default. If *hollow* is true, then the object bases (if any) are not drawn. If it is false, the bases are drawn.

OBJECT CLIPPING (CLIPPLANES)

*object.**clipPlanes***

Variable. Contains an array of 1 to 4 subarrays. Each subarray has 4 numbers a, b, c, and d for a plane coefficients. When drawing an object, only the part for which ax+by+cz+d>0 is drawn.

COLOUR AND LIGHTS

COLOUR (COLOR)

*object.**color***

Variable. Contains the colour of a graphic object – an array of three numbers between 0 and 1 for the three colour components – red, green and blue. Suica objects have different default colours.

OBJECT VISIBILITY (VISIBLE)

*object.**visible***

Variable. Contains a boolean value that determines whether or not an object is to be drawn. The default is true.

DRAWING MODE (MODE)

*object.**mode***

Variable. Contains a number that defines the drawing mode of a graphic object, given by one of the built-in constants:

- *Suica.**POINT*** the vertices of the object are drawn as points
- *Suica.**LINE*** the edges of the object are drawn as segments
- *Suica.**SOLID*** the faces of the object are drawn as solid polygons (default)

ILLUMINATION (LIGHT)

*object.**light***

Variable. Contains the boolean value true or false. Determines whether light is used to shade oblique walls, or the light is ignored and all walls are not shaded. The default is true.

MISCALLANEOUS

STYLE (CUSTOM)

*object.**custom** ({name:value, name:value, …})*

Method. Defines a set of properties of an object. The properties are described as a JavaScript object – a list of name-value pairs.

MERGED OBJECTS (MERGE)

*group.**merge**( )*

Group object method. Makes all current elements in a group indistinguishable by the *objectAtPoint* method. When using *objctAtPoint*, the entire group is selected instead of the specific group element.

MERGED COLOUR (MERGECOLOR)

*group.**mergeColor**( )*

Group object method. Removes all current individual elements' colours. When drawing, the colour of the group itself will be used.

ANIMATION LOOP (NEXTFRAME)

*suica.**nextFrame** = function*

Variable in the Suica class instance. Contains or indicates the function to be executed when generating a frame during animation.

INTERACTIVITY (INTERACTIVE)

*object.**interactive***

Variable. Contains a boolean value. If it is true, the object is processed by *objectAtPoint*. The default is false, i.e. the object is ignored by *objectAtPoint*.

C. GRAPHICAL OBJECTS

ZERO-DIMENSIONAL

POINT (POINT)

point *(coordinates)*
*new Suica.**Point** (coordinates)*

Function and class. Creates a geometric object point. The *coordinate* parameter is an array of x, y, and z coordinates of the point. A point is drawn as a small circle at the corresponding coordinates. Its size does not depend on the distance of the point of view.

The following common properties are maintained for a point: *center*, *color*, *visible* and *pointSize*. By default, the point *color* is red [1,0.2,0.2].

ONE-DIMENSIONAL

LINE (LINE)

line *(point,point)*
*new Suica.**Line** (point,point)*

Function and class. Creates a geometric object line that passes through the two points given as parameters. The line is drawn as a long segment.

After construction, the points through which the line passes are accessible through the *to* and *from* properties. In addition, the following common properties are maintained for a line: *color* and *visible*. By default, the line *color* is green [0,0 .5,0].

RAY (RAY)

ray *(point,point)*
*new Suica.**Ray** (point,point)*

Function and class. Creates a geometric object ray that starts from the first point passed as a parameter and passes through the second. The properties of a ray are the same as those of a line .

SEGMENT (SEGMENT)

segment *(point,point)*
*new Suica.**Segment** (point,point)*

Function and class. Creates a geometric object segment that starts from the first point given as a parameter and ends with the second. The properties of a segment are the same as those of a line.

TWO-DIMENSIONAL

SQUARE (SQUARE)

square *(center,size)*
*new Suica.**Square** (center,size)*

Function and class. Creates a geometric object square, centered on a point given as parameter *center*, and a side length equal to *size* .

RECTANGLE (RECTANGLE)

rectangle *(center,sizes)*
*new Suica.**Rectangle** (center,sizes)*

Function and class. Creates a geometric object rectangle, centered on a point given as parameter *center*, and sides lengths set in an array *sizes* of two numbers.

REGULAR POLYGON (POLYGON)

polygon *(center,radius,count)*
*new Suica.**Polygon** (center,radius,count)*

Function and class. Creates a geometric object regular polygon, with a given point for the *center*, a number for the *radius* of the circumscribed circle, and an integer for the *count* of sides.

CIRCLE (CIRCLE)

circle *(center,radius)*
*new Suica.**Circle** (center,radius)*

Function and class. Creates a geometric object circle, with a point of *center* and given *radius*.

ELLIPSE (ELLIPSE)

ellipse *(center,radii)*
*new Suica.**Ellipse** (center,radii)*

Function and class. Creates a geometric object ellipse, with a point of *center* and an array of two numbers for *radii* in local X and Y axes.

THREE-DIMENSIONAL

CUBE (CUBE)

cube *(center,size)*
*new Suica.**Cube** (center,size)*

Function and class. Creates a geometric object cube, centered on a point given as parameter *center*, and a side length equal to *size*.

RECTANGULAR PARALLELEPIPED (CUBOID)

cuboid *(center,sizes)*
*new Suica.**Cuboid** (center,sizes)*

Function and class. Creates a geometric object rectangular parallelepiped, centered on a point given as parameter *center*, and sides lengths set in the array *sizes* of three numbers.

SPHERE (SPHERE)

sphere *(center,radius)*
*new Suica.**Sphere** (center,radius)*

Function and class. Creates a geometric object sphere, with a given point for *center* and number for *radius*.

SPHEROID (SPHEROID)

spheroid *(center,radii)*
*new Suica.**Spheroid** (center,radii)*

Function and class. Creates a geometric object spheroid, with a given point for *center* and an array of three numbers of the *radii* along the local X, Y and Z axes.

REGULAR PRISM (PRISM)

prism *(center,radius,height,count)*
*new Suica.**Prism** (center,radius,height,count)*

Function and class. Creates a geometric object regular prism, with a given point for the *center* of the bottom base, a number for the *radius* of the circumscribed circle, the *height* of the prism and an integer for the *count* of sides.

REGULAR PYRAMID (PYRAMID)

pyramid *(center,radius,height,count)*
*new Suica.**Pyramid** (center,radius,height,count)*

Function and class. Creates a geometric object regular pyramid, with a given point for the *center* of the base, a number for the *radius* of the circumscribed circle, the *height* of the pyramid and an integer for the *count* of sides.

CYLINDER (CYLINDER)

cylinder *(center,radius,height)*
*new Suica.**Cylinder** (center,radius,height)*

Function and class. Creates a geometric object cylinder, with a given point for the *center* of the bottom base, a number for the *radius* of the base and the *height* of the cylinder.

ELLIPTICAL CYLINDER (CYLINDROID)

cylindroid *(center,radii,height)*
*new Suica.**Cylindroid** (center,radii,height)*

Function and class. Creates a geometric object an elliptical cylinder, with a given point for the *center* of the bottom base, an array of two numbers for the *radii* of the base and the *height* of the cylinder.

CONE (CONE)

cone *(center,radius,height)*
*new Suica.**cone** (center,radius,height)*

Function and class. Creates a geometric object cone, with a given point for the *center* of the base, a number for the *radius* of the base and the *height* of the cone.

ELLIPTICAL CONE (CONOID)

conoid *(center,radii,height)*
*new Suica.**Conoid** (center,radii,height)*

Function and class. Creates a geometric object an elliptical cone, with a given point for the *center* of the base, an array of two numbers for the *radii* of the base and the *height* of the cone.