uBudiler API 2.0 Documentation

Release 0.1

uinnova inc.

May 10, 2017

Table of Contents:

uBuilder API is a set of javascript-like, executed-at-runtime scripts runs inside Momoda system. uBuilder API provides operations on object, UI, behavior and effect in 3D scene. uBuilder API could be used to make customized, industries specific apps such as storage management, emergency drilling, archive management, intelligent buildings etc.

uBuilder API supports fetch and interact with external data source. Through uBuilder API's data interface, 3rd party system could push realtime data to 3D scene to drive realtime object behavior and/or information display.

CHAPTER 1

Concept Guide

model and Scene

There are two basic and import concepts in Momoda: model and scene.

Model

Model is *polygon representation* of object. 3D object could presents something from real world, concept or just your imagination.

Model in Momoda shares the same of idea of 3D object in general, those 3D object could be organized and put into 3D scene like Lego.

Momoda also provides 'official' model library and could be used for free during 3D scene creation. User could also leverage uinnova's profession service to create customized models. Besides, Momoda also provide a 3DSMax plugin which could upload 3D object created in 3DSMax to Momoda's model library, and user could create and use their own model using this plugin, which make Momoda very extensible in term of model creation.

Scene

Scene this one of the basic concept in Momoda. By use momoda's free drawing studio, user can place , combine and layout 3D models/objects into a scene in drag and drop fashion. Typical scene could be campus, a factory, a harbor, and any another things you may think of. We call them "Momoda Scene".

Momoda scene are stored in Momoda Cloud, which could be opened by own later on for editing, preview, or API debugging. Momoda scene can also be download and running locally as "Momoda Offline Edition" (may need purchase license issued by uinnova inc.).

Model Library

There are three common objects in right navigation bar of momoda studio: model, texture and function.

- **Model**

Objects like building, car, people, desk, etc. User could also upload their custom objects through Momoda 3DSMax plugin.

- **Texture**

Textures to cover the surface of object, also supports use custom texture upload by user.

- **Function**

Effects such as fire, rain, flash, or assistant function such as drawing line, pipeline, or text. Function are system build-in and do not supports user upload at this time.

Scene Levels

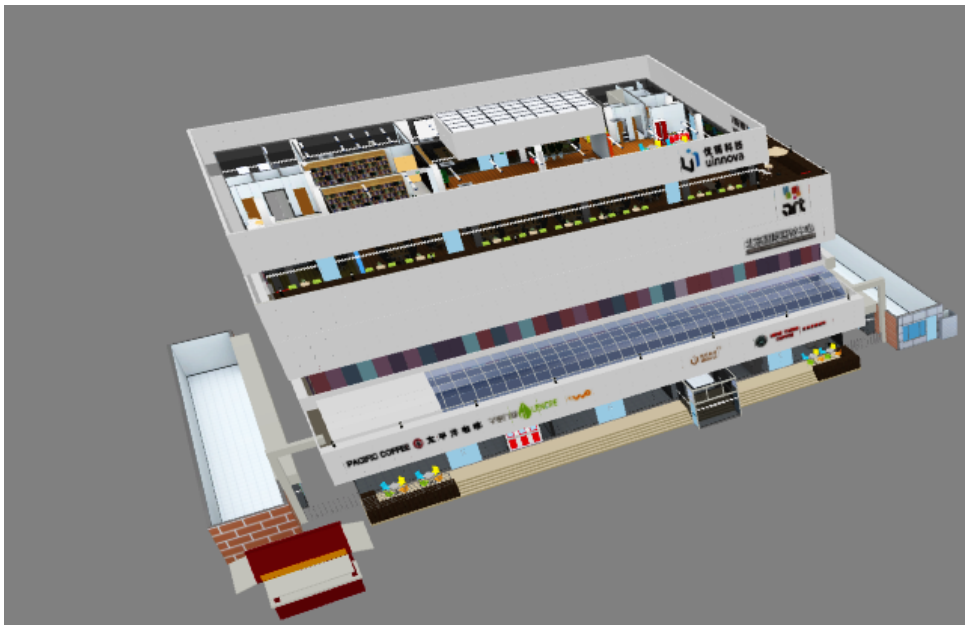
There are four level in Momoda scene: campus, building, stories and rooms.

- **Campus**

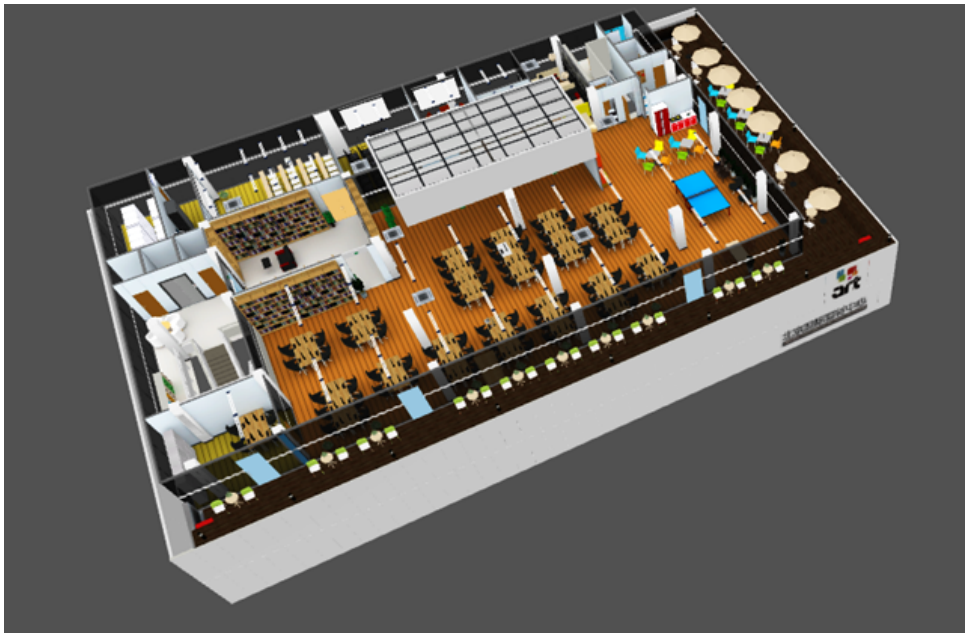
Upmost level of a scene.

- **Building**

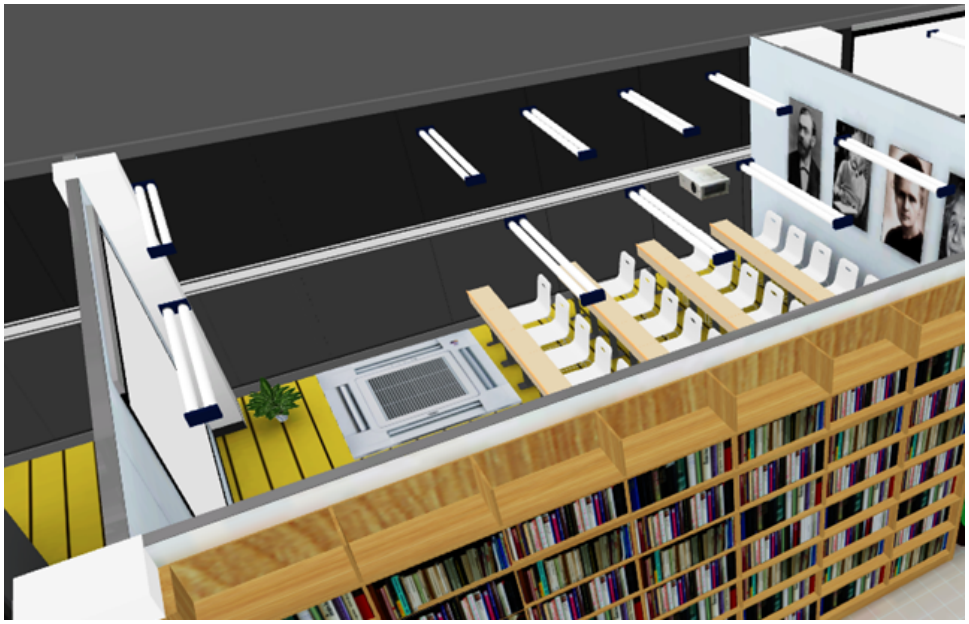
Buildings (do not includes building facade)



- **Stories**



- **Room**



API Developement

Momoda supports online development via **uBuilder API developing web page**.

Create New API Project

There a two pane in developing web page, left pane is for script editing, right pane is for running scripts in 3D scene. To begin a new API development project, visit the uBuilder API developing web page, input the ID of target 3D scene, then click `New` button.

Deploy API Project to Scene

To includes API script into your own scene, simply copy all the script source code and paste to the text box of `My Projects` —> `Target Scene` —> `Configure Scene`.

Keyboard Shortcuts

Below are keyboard shortcuts in *uBuilder API debug page*:

- `Ctrl+Enter` Run Script
- `Ctrl+R` Reset Scene
- `Ctrl+/` Comment/Uncomment source code

API Namespace

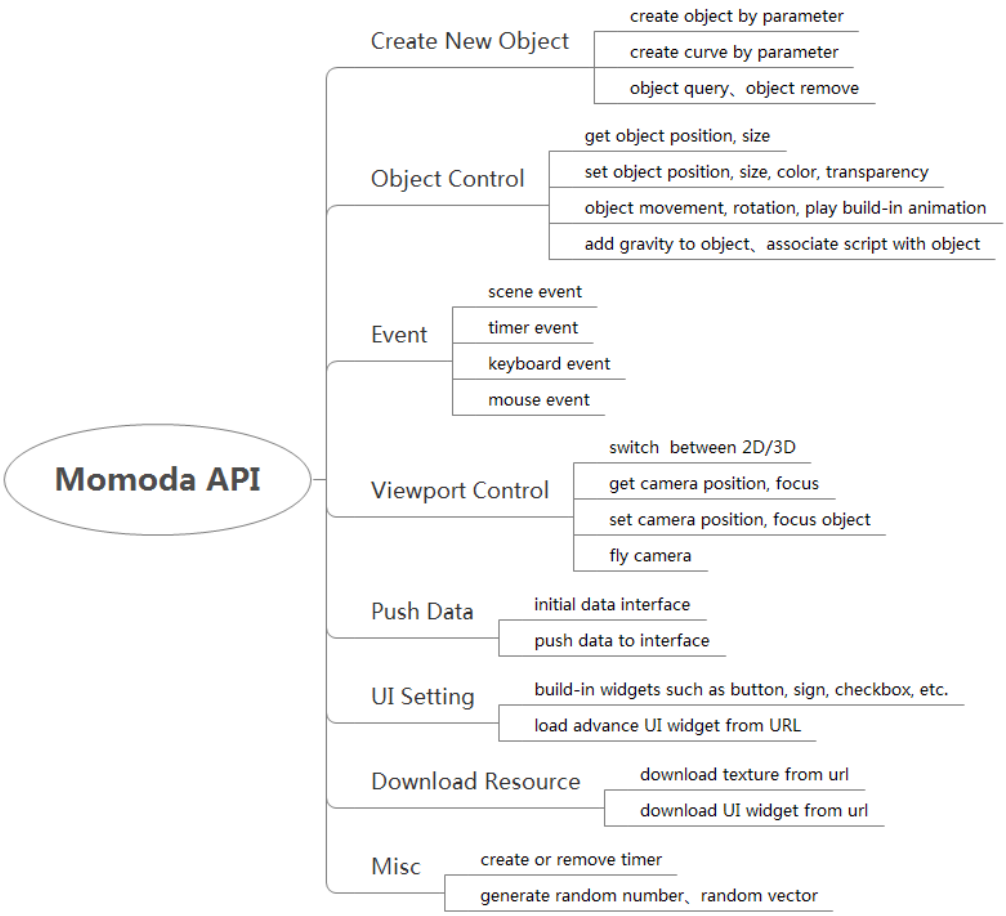
uBuilder API functions are grouped in following namespaces:

Namespace	Description
camera	camera control
object	create, search or modify objects in 3D scene
gui	manage user interface in 3D scene
BaseObject	control existing objects
ScriptObject	associate scripts on existing object
util	common functions
input	keyboard and mouse events
console	management console

CHAPTER 2

Overview

uBuilder API provides following functions:



CHAPTER 3

API Referance

camera

Controls camera objects within a scene.

Overview

Name	Description	Returns	Parameters
changeTo2D	Change scene mode to 2D	none	`camera.changeTo2D()`
changeTo3D	Change the Scene mode to 3D	none	`camera.changeTo3D()`
getEyePos	Get positions of all camera	Vector3	`camera.getEyePos()`
getTargetPos	Get the position of the fixation point of the camera(s)	Vector3	`camera.getTargetPos()`
fit	Focus all cameras on specified object	none	`camera.fit(obj)`
flyTo	Move the camera to new position specified by the function	none	`camera.flyTo({json})`
lookAt	Set the Point of view of a camera	none	`camera.lookAt(pos)`
setPosition	Set the position of a camera	none	`camera.setPosition(pos)`
stopFlying	Stops the camera at the position specified by the function. Often used with the function `FlyTo`	none	`camera.stopFlying()`

camera.changeTo2D

Change scene mode to 2D.

Parameters

none

Example

```
// Changes the scene mode to '2D'.
// If the current scene mode is already in 2D, the scene does not change
camera.changeTo2D()
```

camera.changeTo3D

Change scene mode to 3D.

Parameters

none

Example

```
// Changes the scene mode to '3D'.
// If the current scene mode is already in 3D, the scene does not change
camera.changeTo3D()
```

camera.getEyePos

Get the current position of the camera.

Parameters

none

Example

```
// print the position of the camera
print(camera.getEyePos());
```

camera.getTargetPos

Get the fixation point position of the camera

Parameters

none

Example

```
// print fixation point position of the camera
print(camera.getTargetPos());
```

camera.fit

Focus camera on specified object

Parameters

Name	Description
obj	object referance, which will be focued by camera

Example

```
/** create box object and have the camera focus on the object.
The focus point is the center point of the object.
The position of the camera is based on size of the object */

var object.create("AB052B5B646E4A48B9C045096FF9B088");
camera.fit(obj);
```

camera.flyTo

Move the position and the fixation point of the camera within a certain time, then, execute a function.

Parameters

Name	Description
{json}	`json message`, including position, fixation point ,time, execute function

Example

```
/** move camera to position (2,3,4) and change the fixation point to (3,4,5 )
within 2 seconds, then print "OK".*/

camera.flyTo({
 "eye":Vector3(2,3,4),
 "target":Vector3(3,4,5),
 "time":2.0,
 "complete":function(){print("OK!")}
 })
```

camera.lookAt

Set camera's fixation point as given Vector

Parameters

Name	Description
pos	Vector3 variable; camera's fixation point

Example

```
// set camera's fixation point to be the center point of the object 'obj'.
camera.lookAt(obj.center);
```

camera.setPosition

Set camera's position as given Vector

Parameters

Name	Description
pos	`Vector3` camera's position

Example

```
// set camera's position to (0,1,2)
camera.setPosition(Vector3(0,1,2));
```

camera.stopFlying

Set camera's position as given Vector

Parameters

none

Example

```
// Create a button named'Execute'.
// Clicking on the button will stop moving the position or fixation point of the camera.

camera.setPosition(Vector3(0,1,2));

camera.flyTo({

 "eye":Vector3(2,3,4),

 "target":Vector3(3,4,5),

 "time":2.0,

 "complete":function(){print("OK!")}})

 gui.createButton("Execute", Rect(10, 50, 200, 50), function() {camera.stopFlying();})
```

object

Overview

Control the objects within the scene.

Name	Description	Returns
create	Create an object	Returns the object type of the created object
createArrowLine	Create an arrow line	Returns the object type of the created arrow line
createCurveLine	Create a curved line	Returns the object type of the curved line
destroyAll	Delete all the objects created by a script	none

object.create

Create object.

Parameters

Name	Description
bundleId	`string` , the id of the object
parentObj	`BaseObject`, parent of the object
callback	`function`, Callback function. After loading the object, execute this callback
pos	`Vector3` ; the position of the object
scale	`Vector3`; the size of the object

Example

```
/** Create object'obj1'and set its position to (1,0,1)
Create a second object 'obj2'and set its parent to be obj1 and its position to (2,0,1)
scale (1,2,3), after loading obj2, execute callback, rotate the obj1 45 degree of Y
↪(this time obj1 is obj2's parent,
so obj1 and obj2 will rotate together.) */

var obj1 = object.create("AB052B5B646E4A48B9C045096FF9B088",Vector3(1,0,1));
var obj2 = object.create("AB052B5B646E4A48B9C045096FF9B088",obj1,function(){obj1.
↪yaw(45)},Vector3(2,0,1),Vector3(1,2,3));
```

object.createArrowLine

Create arrowed line.

Parameters

Name	Description
vertices	`array` or `vector3List` ; the set of points on the arrowed line
{json}	json message; includes the color of the arrow and the color of the line.

Example

```
//define a Vector array
var vecArray2 = [Vector3(0, 1, 20), Vector3(10, 1, 20)];


//Create an arrowed line. Set the start position to (0,1,20) and the end position to (10,1,20).
//The color of the line is set to red, the color of the arrow is set to green.

object.createArrowLine(vecArray2, {

"color": Color.red,

"arrowColor": Color.green});
```

object.createCurveLine

Create curve line.

Parameters

Name	Description
vertices	`array` or `Vector3List`The set of the points on the curve line
bundleOrColorOrMat	`string` or `color`
parentObj	`BaseObject`; the parent of the curve line
width	`float`; the width of the curve line
textiling	repeatability of the material
texOffSet	the offset of the material

Example

```
// create a curve line named curveLine1
var vecList = Vector3List();

vecList.Add(Vector3(0,1,0));

vecList.Add(Vector3(10,1,0));

vecList.Add(Vector3(10,1,5));

var curveLine1=object.createCurveLine(vecList, Color.green);


// Create a curve line named 'curveLine2'. Set the material of the curve line to specified material.
// Set the parent of'curveLine2'to'curveLine1'.
//Set the repeatability of'curveLine2's' material to (1,2) and the offset of it's material to (0,0)

var vecArray = [Vector3(0,1,5), Vector3(0,2,15), Vector3(10,4,15), Vector3(10,6,5)];

var curveLine2 = object.createCurveLine(vecArray, "1D2702801708453680664DCABE70890B", curveLine1,2,Vector2(1,2),Vector2(0,0))
```

object.destroyAll

Destoy all the objects created by a script

Parameters

None

Example

```
//create obj
var obj  = object.create("AB052B5B646E4A48B9C045096FF9B088");


//create curveLine
var vecArray = [Vector3(0,1,5), Vector3(0,2,15), Vector3(10,4,15), Vector3(10,6,5)];

var curveLine1=object.createCurveLine(vecArray, Color.green);


//Create a button called 'Delete'. Clicking on the button'Delete'will destroy all␣
↪objects created by this script.
gui.createButton("Delete", Rect(100, 100, 100, 30), function() {object.destroyAll()})
```

object.find

Find object by object ID.

Parameters

Name	Description
uid	`string` object uid

Example

```
// Find Object with an Uid equal to'Object01',
// then rotate this object around the Y-axis at a 45 degree angle.
object.find("Object01");
obj.yaw(-45);
```

gui

Overview

Control graphical user interface within the scene.

Name	Description	Returns	Parameters
createBox	Create a box	`object`	`gui.createbox(text,rect)`
createButton	Create a button	`object`	`gui.createButton(text,rect,callback)`
createLabel	Create a label	`object`	`gui.createLabel(text, rect)`
createToggle	Create a toggle button	`object`	`gui.createToggle(checked,text,rect, callback)`
load	Load an external GUI resource	none	`gui.load(url,callback)`

gui.createBox

Creat a button.

Parameters

Name	Description
text	`string` text to display on the button
rect	`rect` a rectangle on the screen to use as the button
callback	`function` callback function on mouse click

Example

```
/** Create a button. Set the left coordinate to 100 pixels and the top coordinate to 200 pixels.
The width and height of the button is set to 80 pixels and 50 pixels respectively.
Clicking on this button will print the string'You Clicked Button 1' */

var buttont1 =  gui.createButton("Button 1", Rect(100, 200, 80, 50), function() {
print ("You clicked Button 1");});
```

gui.createLabel

Make a text or texture label.

Parameters

Name	Description
text	`string` text to display on the label
rect	`rect` a rectangle on the screen to use as the label

Example

```
/** Create a Label. Set the left coordinate to 100 pixels and the top coordinate to 200 pixels.
 The width and height of the label is set to 80 pixels and 50 pixels respectively.*/

gui.createLabel("Label 1", Rect(100, 200, 80, 50));
```

gui.createToggle

Make an on/off toggle button

Parameters

Name	Description
checked	`Boolean` set initial status to 'On' or 'Off'
text	`string` text to display on the toggle
rect	`rect` a rectangle on the screen to use as the toggle button
callback	`function` callback function on mouse click

Example

```
/** Create a Toggle Button. Set the left coordinate to 100 pixels and the top coordinate to 200 pixels.
 The width and height of the GUI Box is set to 80 pixels and 50 pixels respectively.
 If the value of the Toggle Button is changed, print the string'You clicked Toggle 1'*/

 gui.createToggle ("Toggle 1", Rect(100, 200, 80, 50), function() {
     print("You changed the state of Toggle 1");}
 );
```

gui.load

Load GUI resource.

Parameters

Name	Description
url	`string` resource url
callback	`function` callback function on mouse click

Example

```
/** GUI resource from the specified URL. After the GUI has loaded,
  print 'Successfully loading resource from url! */

var url = "http://www.3dmomoda.com/mmdclient/script/examples/demos/scifi_ui.bundle"
gui.load(url, function(){print("Successfully loading resource from url")});
```

BaseObject

Overview

Provides actions on object.

Name	Description	Returns
addGravity	Add a gravity value to a specified object	none
addScript	Add a script to a specified object,Returns the type of script added to the object	`script`
addTail	Add a tail to a specified object. Commonly used with the function movePath	none
clone	Copy an existing object	`BaseObject`
destroy	Remove an existing object	`BaseObject`
getPosition	Acquire the position of a specified object	`Vector3`
getScale	Acquire the scale of a specified object,‘‘Vector3‘‘ movePath,Move object by a specified parameter	none
moveTo	Move object to a position within a given time	none
pitch	Angel of rotation(in degree) of the object around its pivot point along X axis	none
removeScript	Remove a script from the specified object	none
roll	Angel of rotation (in degree) of the object around its pivot point along the Z axis	none
setAnimSpeed	Set the animation speed of a specified object	none
setColor	Set the color of a specified object	none
setColorFlash	Set the flash state, flash color and flash time of a specified object	none
setPickEnabled	Set the selectable state of the specified object. Commonly used with *mouse event* functions	none
setPosition	Set the object’s position	none
setPositionXZ	Set the object’s horizontal position	none
setPositionY	Set the object’s position along the Y axis	none
setScale	Set the scale of the object	none
setTransparent	Set the transparency of a specified object	none
stopAnim	Stop a specified object’s animation	none
stopMoving	Stop the object moving	none
transformPoint	Change specified coordinates of object from relative coordinates to absolute coordinates	`Vector3`
translate	Move a specified object in a specified direction and distance	`Vector3`
yaw	Angel of rotation (in degree) of the object around its pivot point along the Y axis	none

BaseObject.addGravity

Add gravitational value to a object

Parameters

Name	Description
mass	`float` the weight of the object

Example

```
//add gravity to a 3.5 KG weight object
var obj = object.create("AB052B5B646E4A48B9C045096FF9B088");
obj.addGravity(3.5)
```

BaseObject.addScript

Add script to a object

Parameters

Name	Description
script	`script` to associate with the object
name	`string` name of the script

Example

```
/** Create a script named'AutoRtate'used to define the speed of object'objOption'.
A script may include a'Start'and'Update' function which is automatically recognized by the system.
The'Start'function will be called only once while the'Update'function will be called on repeatedly */

AutoRotate = {

 speed : 0,

 objOption : null,

 function Start() {this.speed = util.randomFloat(1, 8);}

 function Update() {this.objOption.yaw(this.speed); }

 }

 //create object
 var obj = object.create("AB052B5B646E4A48B9C045096FF9B088", Vector3(2.5, 0, 0));
```

```
19
20  //add script "rotation" to object
21  var script = obj.addScript(AutoRtate,"rotation");
22
23  //set script's attribute objOption to be object; now function Update can rotate obj␣
    ↪by its Y axis in a random speed
24  script.objOption = obj;
```

BaseObject.addTail

Add trail to a specified object. Most Commonly used with the function movePath. Often used to increase the visual effects of an object.

Parameters

Name	Description
{json}	json format includes start width, end width ,end color and time

Example

```
1   //create object
2   var obj =object.create("AB052B5B646E4A48B9C045096FF9B088");
3   var path = Vector3List();
4
5   //generate 36 vector3 point, add them to a Vector3List, this could be seen as a␣
    ↪circle, its radius is 10
6   for (var degree = 0; degree < 360; degree += 10)
7   {
8       path.Add(Vec3(Math.Cos(degree*Math.Deg2Rad)*10,0.5,Math.Sin(degree*Math.
    ↪Deg2Rad)*10));
9   }
10
11  // move an object along the path in 10 seconds repeatly, whilst moving,
12  // the object will always look at (0,0,0)
13  obj.movePath({
14
15      "path": path,
16
17      "time": 10,
18
19      "lookPos": Vector3.zero,
20
21      "loopType": "loop"
22
23  });
24
25
26  //add a tail, start width 0.6, end width 0, color is red, lasting 5 seconds
27  obj.addTail({
28
```

```
 "startWidth": 0.6,

 "endWidth":0,

 "color":Color.red,

 "time": 5

 });
```

BaseObject.clone

Clone a object.

Parameters

None.

Example

```
//Create an object, copy it and name is'obj2'. Then rotate'obj2'along the Y-Axis at a␣
↪45 degree angle

var obj = object.create("AB052B5B646E4A48B9C045096FF9B088");

var obj2=obj.clone();

obj2.yaw(45)
```

BaseObject.destroy

Remove a object.

Parameters

None.

Example

```
//Create an object named 'Obj'. Create a button named'Delete'with the size (10,100,
↪100,20).
//click on this button to remove object'obj'from the scene

var obj = object.create("AB052B5B646E4A48B9C045096FF9B088");

gui.createButton("Delete",Rect(10,100,100,20),function(){obj.destroy()})
```

BaseObject.getPosition

Get position of a object.

Parameters

None.

Example

```
//Create an object and print its position

var obj = object.create("AB052B5B646E4A48B9C045096FF9B088");
print(obj.getPosition())
```

BaseObject.getScale

Get scale information of a object.

Parameters

None.

Example

```
//Create an object and print its scale

var obj = object.create("AB052B5B646E4A48B9C045096FF9B088");
print(obj.getScale())
```

BaseObject.movePath

Move object along pre-defined path.

Parameters

Name	Description
{json}	json format, specify path, time, target point, if loop, etc.

Example

```
//Create object
var obj =  object.create("AB052B5B646E4A48B9C045096FF9B088");

//Generate 36 Vector3 point, add them in a Vector3List, you could see this Vector3List as a circle

var path = Vector3List();

for (var degree = 0; degree < 360; degree += 10)

path.Add(Vec3(Math.Cos(degree*Math.Deg2Rad)*10,0.5,Math.Sin(degree*Math.Deg2Rad)*10));


//Move an object along a path in 10 seconds , whilst moving ,the object will always face the vector (0,0,0) ,
//after the object completes the movement , loop this function

obj.movePath({

"path": path,

"time": 10,

"lookPos": Vector3.zero,

"loopType": "loop"

});
```

BaseObject.moveTo

Move object to target position within a certain time

Parameters

Name	Description
pos	`Vector3` destination position
time	`float` moving time

Example

```
// Create object, move it to (10,0,0) in 5 seconds
var obj = object.create("AB052B5B646E4A48B9C045096FF9B088");
obj.moveTo(Vector3(10, 0, 0), 5.0)
```

BaseObject.pitch

Rotate object on the X-axis by degree

Parameters

Name	Description
degree	`float` degree of rotation

Example

```
//Create an object and rotate the object on the X-axis at a 45 degree angle.
var obj = object.create("AB052B5B646E4A48B9C045096FF9B088");
obj.pitch(45)
```

BaseObject.playAnim

Play object's animation

Parameters

Name	Description
animName	`string` name of the animation

Example

```
// Create an object named'obj'. Create a button with size (100,100,100,30).
// Pressing on this button will execute the "Run" animation of the object.
var obj = object.create("0bcba8ca78734b64a3dae3eb699a913c");
gui.createButton("Run", Rect(100, 100, 100, 30), function() {obj.playAnim("Run");});
```

BaseObject.removeScript

Move script associate with object

Parameters

Name	Description
name	`string` name of the script

Example

```
AutoRtate = {

speed : 0,

objOption : null,

function Start() {this.speed = util.randomFloat(1, 8);}

function Update() {this.objOption.yaw(this.speed); }}

var obj = object.create("AB052B5B646E4A48B9C045096FF9B088", Vector3(2.5, 0, 0));

var script = obj.addScript(AutoRtate,"rotation");

script.objOption = obj;

//Create a button. Pressing on this button will remove the "rotation" script from␣
↪the object'obj'.

gui.createButton("Remove Script", Rect(100, 100, 100, 30),function(){obj.
↪removeScript("rotation")})
```

BaseObject.roll

Rotate object a specified degree around the Z-axis (clockwise)

Parameters

Name	Description
degree	`float` degree of rotation

Example

```
//Create an object and rotate the object on the X-axis at a 45 degree angle.
var obj = object.create("AB052B5B646E4A48B9C045096FF9B088");
obj.roll(45)
```

BaseObject.setAnimSpeed

Set the animation speed of a specified object

Parameters

Name	Description
speed	`float` speed of animation playing

Example

```
var obj = object.create("0bcba8ca78734b64a3dae3eb699a913c");

gui.createButton("Run", Rect(100, 100, 100, 30), function() {obj.playAnim("Run");});

//Create a button. Pressing on this button will set the objects animation speed to'4.
↪5'
gui.createButton("Accelerate", Rect(100, 150, 100, 30), function() {obj.
↪setAnimSpeed(4.5)})
```

BaseObject.setColor

Set object color

Parameters

Name	Description
color	`color`

Example

```
//Set the object color to blue

var obj = object.create("FF2A3E364B1E4B928891E05A9279C7A7", Vector3(0, 0, 0));

obj.setColor(Color.blue);
```

BaseObject.setColorFlash

Set flash state, flash color and flash interval of a specified object

Parameters

Name	Description
enable	`boolen` turn flash 'On' or' Off'
color	`color` flash color
time	`float` flash interval

Example

```
// Create object and set flash to'On', flash color to 'Green' and Flash interval to 2.5 seconds.

var obj = object.create("FF2A3E364B1E4B928891E05A9279C7A7", Vector3(4, 0, 0));

obj.setColorFlash(true, Color.green,2.5);
```

BaseObject.setPickEnabled

Set selectable state of the specified object. Commonly use with the 'Mouse event' functions

Parameters

Name	Description
enable	`boolen` turn pickable 'On' or' Off'

Example

```
var obj = object.create("AB052B5B646E4A48B9C045096FF9B088");

var dragObj = null;

util.addEventListener("dragstart", function(event) {

if (event.obj && event.button == 0) {

dragObj = event.obj;

dragObj.setPickEnabled(false)

camera.enableRot = false; }});

// Add a "dragstart" event to an object named'obj'.
// Left clicking and dragging object'obj'will change the selectable state to false
↪(Prevents object from being repeatedly dragged).

util.addEventListener("drag", function(event) {

if (dragObj && event.button == 0)

dragObj.pos = event.pos;});

util.addEventListener("dragend", function(event) {

if (dragObj && event.button == 0) {

dragObj.setPickEnabled(true);

dragObj = null;

camera.enableRot = true;}});
```

BaseObject.setPosition

Set object position

Parameters

Name	Description
x	`float` X-Axis value
y	`float` Y-Axis value
z	`float` Z-Axis value

Example

```
//Set the object's position to (0,5,0)

```

```
var obj = object.create("AB052B5B646E4A48B9C045096FF9B088");

obj.setPosition(0, 5, 0);
```

BaseObject.setPositionXZ

Set object horizontal position

Parameters

Name	Description
x	`float` X-Axis value
z	`float` Z-Axis value

Example

```
// Set the object's horizontal position to (1,1)

var obj = object.create("AB052B5B646E4A48B9C045096FF9B088");

obj.setPositionXZ(1,1);
```

BaseObject.setPositionY

Set object Y-Axis coordinate position

Parameters

Name	Description
y	`float` Y-Axis value

Example

```
// Set the Y-Axis coordinate of the object to 5

var obj = object.create("AB052B5B646E4A48B9C045096FF9B088");

obj.setPositionY(5);
```

BaseObject.setScale

Set object scale

Parameters

Name	Description
x	`float` X-Axis value
y	`float` Y-Axis value
z	`float` Z-Axis value

Example

```
// Set the scale of the object to be (1,2,3)

var obj = object.create("AB052B5B646E4A48B9C045096FF9B088");

obj.setScale(1,2,3);
```

BaseObject.setTransparent

Set object transparency.

Parameters

Name	Description
trans	`float` transparency value range 0~1

Example

```
// Set the object's transparency to'0.5'

object.create("AB052B5B646E4A48B9C045096FF9B088");

obj.setTransparent(0.5);
```

BaseObject.stopAnim

Stop play object animation.

Parameters

None

Example

```
var obj =  object.create("0bcba8ca78734b64a3dae3eb699a913c");

gui.createButton("Run", Rect(100, 100, 100, 30), function() {obj.playAnim("Run");});

// Create a button named 'Stop'and set its size to (100,150,100,30).
// Pressing this button will stop object'obj'from playing the animation'Run'.

gui.createButton("Stop", Rect(100, 150, 100, 30), function() {obj.stopAnim()});
```

BaseObject.stopMoving

Stop object moving.

Parameters

None

Example

```
var obj =  = object.create("AB052B5B646E4A48B9C045096FF9B088");

obj.moveTo(Vector3(10, 0, 0), 5.0)

// Create a button named 'Stop'and set its size to (100,150,100,30).
// Pressing this button will stop object'obj'from moving.

gui.createButton("Stop", Rect(100, 150, 100, 30), function() {obj.stopMoving()});
```

BaseObject.transformPoint

Convert coordinates of object from relative coordinates to absolute coordinates

Parameters

Name	Description
pos	`Vector3` relative coordinates of the object

Example

```
// Create object'obj1, set its position (1,2,3)
var obj  = object.create("AB052B5B646E4A48B9C045096FF9B088",Vector3(1,2,3));

// Create a second object named'obj2'whose parent is'obj1'
var obj2 = object.create("AB052B5B646E4A48B9C045096FF9B088",obj1,Vector3(4,5,6));

print(obj2.transformPoint(Vector3(7,8,9)));
```

Note:

- The relative coordinates of'obj2'is (4,5,6)Thus the absolute coordinates of'obj2'is `(1,2,3)+(4,5,6)=(5,7,9)`
- Printing the transformPoint of Vector3(7,8,9) will show the value (12,15,18) (Converting the relative coordinates of Vector3 to absolute coordinates is `(5,7,9)+(7,8,9)=(12,15,18))`

BaseObject.translate

Move a specified object to a specified direction and distance .

Parameters

Name	Description
pos	`Vector3`

Example

```
// Create object'obj1, set its position (1,2,3)
var obj  =  object.create("81807868C78141BFB2E93275AC3ABB39");

// Create button *Button1*, If press this button, object obj's position add
↪Vector3(1,0,1)

var Button1= gui.createButton("translate", Rect(100, 200, 80, 50), function() {

obj.translate(Vector3(1, 0, 1))});



// Create a button named *Button2* Pressing on this button will move object'obj'
↪position by a vector of (1,0,1)

var Button2= gui.createButton("setPosition", Rect(100, 300, 80, 50), function() {

obj.setPosition(Vector3(1, 0, 1))});
```

BaseObject.yaw

Rotate an object a specified degree around the Y-axis (clockwise)

Parameters

Name	Description
degree	`float` rotation degree

Example

```
//Create an object named'obj'and rotate object'obj'a 45 degree angle around the Y-axis(clockwise).

var obj  =  object.create("AB052B5B646E4A48B9C045096FF9B088");
obj.yaw(45)
```

ScriptObject

Overview

Control all the scripts associated with objects. script may implements two object interface: `Start` and/or `Update`

Name	Description	Returns	Parameters
Start	called before any object updates, just once	none	`Start({script})`
Update	called per frame	none	`Update({script})`

Start

Parameters

None

Exmaple

```
//Create a Start function, this function defines the initial speed of an object
//to a random float between 1 to 8
AutoRtate = {
    speed : 0,
    objOption : null,
    function Start() {this.speed = util.randomFloat(1, 8);}

    // Create an Update function,
    //this function rotates the object a random degree along the Y-Axis every frame.
    function Update() {this.objOption.yaw(this.speed); }
}
```

```

var obj = object.create("AB052B5B646E4A48B9C045096FF9B088", Vector3(2.5, 0, 0));

var script = obj.addScript(AutoRtate,"rotation");

script.objOption = obj;
```

Update

Parameters

None.

Exmaple

```
//Create a Start function, this function defines the initial speed of an object
//to a random float between 1 to 8
AutoRtate = {
    speed : 0,
    objOption : null,
    function Start() {this.speed = util.randomFloat(1, 8);}

    // Create an Update function,
    //this function rotates the object a random degree along the Y-Axis every frame.
    function Update() {this.objOption.yaw(this.speed); }
}

var obj = object.create("AB052B5B646E4A48B9C045096FF9B088", Vector3(2.5, 0, 0));

var script = obj.addScript(AutoRtate,"rotation");

script.objOption = obj;
```

util

Overview

Common functions.

Name	Description	Returns	Parameters
addEventListener	Add an event	none	`util.addEventListener(eventType, callback)`
clearAllEvents	Clear all events	none	`util.clearAllEvents()`
clearAllTimers	Clear all timers and unnecessary Timer IDs	none	`util.clearAllTimers()`
clearInterval	Clear the timer generated by setInterval, need timerID	none	util."`util.clearInterval(intervalID)`"
clearScriptObjects	Clear all objects created by scripts, including objects and GUIs	none	`util.clearScriptObjects()`
clearTimeout	Clear timers generated by setTimeout. The Timer's ID needs to be provided.	none	`util.clearTimeout(timeoutID)`
downloadTexture	Download a texture from an external URL	none	`util.downloadTexture({json}`
downloadTextures	Download multiple textures form an external URL	none	`util.downloadTextures({json}`
randomColor	Generate a random RGBA type color	RGBA	`util.randomColor()`
randomFloat	Generate a random float number between two specified values	float	`util.randomFloat(a,b)`
randomInt	Generate a random integer between two specified values	int	`util.randomInt(a,b)`
randomVector3	Generate a random Vector 3	Vector3	`util.randomVector3(randius)`
setInterval	Call a function after a defined time	Int	`util.setInterval(callback, tickTime)`
setRenderCallback	Call a callback function every frame	none	`util.setRenderCallback(callback)`
setTimeout	call a callback function when timeout, return timer ID	int	`util.setTimeout(callback, delayTime)`

util.addEventListener

Add event listener.

Parameters

Name	Description
eventType	`string` event type, e.g. click,dbclick,mouseup,mousedown,mousemove,drag,dragstart,dragend,keydown,keyup,resize
callback	`function` callback function

Example

```
//Add a listener to the event'click', this will create an object on a mouse click
util.addEventListener("click", function(event) {object.create(
↪"FF2A3E364B1E4B928891E05A9279C7A7", event.pos);});
```

util.clearAllEvents

Parameters

None.

Example

```
//Clear all the events within the scene
util.clearAllEvents()
```

util.clearAllTimers

Parameters

None.

Example

```
//Clear all the timers within the scene
util.clearAllTimers()
```

util.clearInterval

Remove timer created by `setInterval`.

Parameters

Name	Description
timerID	`number` timer ID

Example

```
//remove timer with ID 2
util.clearInterval(2)
```

util.clearScriptObjects

Parameters

None.

Example

```
//Clear all the script associate with object
util.clearScriptObjects()
```

util.downloadTexture

Download texture from an external URL

Parameters

Table 3.1: :header: Name, Description :widths: 5, 15

{json}	json message ; including url , callback function

Example

```
/** Create a cube, download a texture from url,
if the download is successful, set the cube's texture to earMat, and earMat to be earth's material
(the texture is provided by uinnova, details on creating and using custom texture , please contact uinnova inc.) */

var earth = object.create("B723E9E1B279467EBC9433D30D35F683", Vec3(0, 5, 0));

util.downloadTexture({

    "url": " http://img1.juimg.com/141102/330507-141102164G965.jpg ",

    "success": function(texture) {

```

```
    var earthMat = util.createMaterial(texture);

    earth.setMaterial(earthMat); }});
```

util.clearTimeout

Remove timer created by `setTimeout`.

Parameters

Name	Description
timerID	`number` timer ID

Example

```
//remove timer with ID 2
util.clearTimeout(2)
```

util.downloadTextures

Download multiple texture from an external URL

Parameters

Table 3.2: :header: Name, Description :widths: 5, 15

{json}	json message ; including url , callback function

Example

```
/** Download textures from an external URL, if the download issuccessful ,
set "Earth.jpg" to earthMat and earthMat to be earth's material;
set texture "Moon.jpg" to moonMat, and MoonMat to be moon's material.
(the texture is provided by uinnova, details on creating and using custom texture ,
please contact uinnova inc.) */

var earth = object.create("9f5681fe55674ce9b617f9fa23d9729b", Vec3(0, 5, 0));

var moon = object.create("9f5681fe55674ce9b617f9fa23d9729b",Vec3(0, 7, 0),Vec3(0.2,
↪0.2, 0.2));

```

```
util.downloadTextures({

    "url": "http://www.3dmomoda.com/mmdclient/script/examples/demos/earth_moon.zip",

    "success": function(textures) {

    var earthMat = util.createMaterial(textures["Earth.jpg"]);

    earth.setMaterial(earthMat);

    var moonMat = util.createMaterial(textures["Moon.jpg"]);

    moon.setMaterial(moonMat);}});
```

util.randomColor

Generate a random RGHA color.

Parameters

None.

Example

```
//Generate a random RGBA type color and apply it as the color of the object'obj'
obj.setColor(util.randomColor())
```

util.randomFloat

Generate a random float number between two specified values

Parameters

Name	Description
a	`float` upper bound value
b	`float` lower bound value

Example

```
// Generate a random number between 1(included) and 3(included)
var d = util.randomFloat(1.0,3.0)
```

util.randomInt

Generate a random integer number between two specified values

Parameters

Name	Description
a	`int` upper bound value
b	`in` lower bound value

Example

```
// Generate a random number between 1(included) and 10(included)
var d = util.randomInt(1,10)
```

util.randomVector3

Generate a random Vector3

Parameters

Name	Description
radius	`number` vector radius

Example

```
// Generate a random vector between ([1, -1], 1, -1])
var d = util.randomVector3(1)
```

util.setRenderCallback

Create callback function run every frame

Parameters

Name	Description
callback	`function`

Example

```
//Create an object and add a gravitational weight of 3KG every frame.
util.setRenderCallback(function(){

var obj = object.create("AB052B5B646E4A48B9C045096FF9B088");

obj.addGravity(3);

})
```

util.setTimeout

Create a callback function with timeout, return the timer's ID

Parameters

Name	Description
callback	`function`
timeout	`number`

Example

```
// Print'time over!'when timer is equal to 3
var a=util.setTimeout(function() {print("time over!")}, 3000)
```

input

Overview

Inputs from keyboards and mouses.

Name	Description	Returns	Parameters
getKey	Get statue of holding down appointed ke	boolen	`getKey(keyCode)`
getKeyDown	Get statue of pressing appointed key	boolen	`getKeyDown(keyCode)`
getKeyUp	Get statue of releasing appointed key	boolen	`getKeyUp(keyCode)`
getMouseButton	Get statue of pressing a mouse button	boolen	`getMouseButton(MouseCode)`
getMouseButton-Down	Get statue of pressing a mouse button	boolen	`getMouseButtonDown(MouseCode)`
getMouseButtonUp	Get statue of releasing the mouse button	boolen	`getMouseButtonUp(MouseCode)`

input.getKey

Check if user press and holds specific key.

Parameters

Name	Description
keyCode	`KeyCode` raw key code for keyboard events.

Example

```
Player = {
    obj : null,
    function Update() {
             //if user holds down key A, rotate obj -5 degree around its Y axis
            if (input.getKey(KeyCode.A))  this.obj.yaw(-5);

            //if user holds down key D, rotate obj 5 degree around its Y axis
            if (input.getKey(KeyCode.D))  this.obj.yaw(5);

            //if user holds down key R, move obj to Vector3(3,0,3) in 2 seconds
            if (input.getKeyDown(KeyCode.R))  this.obj.moveTo(Vector3(3,0,3),2);


            //if user holds down key R, move obj to Vector3(-3,0,-3) in 2 second
            if (input.getKeyUp(KeyCode.R))  this.obj.moveTo(Vector3(-3,0,-3),2);


            //if user clicks the left mouse button, print ' Pressed left click'
            if (input.getMouseButtonDown(0)) print("Pressed left click ");

            //if user clicks the right mouse button,print'Pressed right click'
            if (input.getMouseButtonDown(1)) print("Pressed right click ");
        }
    }

var obj = object.create("0bcba8ca78734b64a3dae3eb699a913c");

var script = obj.addScript(Player);

script.obj = obj;

camera.enableMove = false;input.getKeyDown(keyCode);
```

input.getKeyDown

Check if user press specific key.

Parameters

Name	Description
keyCode	`KeyCode` raw key code for keyboard events.

Example

See *getKey*

input.getKeyUp

Check if user release specific key.

Parameters

Name	Description
keyCode	`KeyCode` raw key code for keyboard events.

Example

See *getKey*

input.getMouseButton

Check if user click and hold mouse button.

Parameters

Name	Description
MouseCode	`int` raw key code for mouse events.

Example

See *getKey*

input.getMouseButtonDown

Check if user click mouse button.

Parameters

Name	Description
MouseCode	`int` raw key code for mouse events.

Example

See *getKey*

input.getMouseButtonUp

Check if user release mouse button.

Parameters

Name	Description
MouseCode	`int` raw key code for mouse events.

Example

See *getKey*

console

Overview

Manage control panel

Name	Description	Returns	Parameters
clear	Clear text content	none	`console.clear()`
log	Print message on console platform	none	`console.log(obj)`
show	Hide or show console platform	none	`console.show(show)`

console.clear

Parameters

None.

Example

```
// Clear all text content on the console platform
//Clear all text content on the control panel

var obj = object.create("AB052B5B646E4A48B9C045096FF9B088");

console.log(obj)

console.clear()
```

console.log

Print output on **Control Panel**

Parameters

Name	Description
obj	object

Example

```
//Print the scale of the object'obj'on the control panel
var obj = object.create("AB052B5B646E4A48B9C045096FF9B088");
console.log(obj.getScale())
```

console.show

Show or hide **Control Panel**

Parameters

Name	Description
show	boolean

Example

```
print("show/hide control panel")

gui.createButton("Platform shows:", Rect(100, 100, 200, 50), function() {console.
↪show(true)});
```

```
4
5  gui.createButton("Close platform", Rect(100, 200, 200, 50), function() {console.
↪show(false)});
```

Data Interface

Summary

Momoda data infterface connect Momoda to third party systems, it can be used to scene initialization, push alarm, control objects, such as create, move, transform, remove, etc., in realtime.

There are three catelogies in Momoda data infterface, namely:

- I Interface
- M Interface
- R Interface

I Interface

I (short for Initialize) interface is used for scene initialization. Data push through *I Interface* will loaded by Momoda client automatically.

Usage

url	http://*{Your Momoda Server IP}*:8080/goods/save
http method	GET or POST
parameters	• `g.sid`: scene ID • `g.oid`: object ID • `g.props`: object initialization data

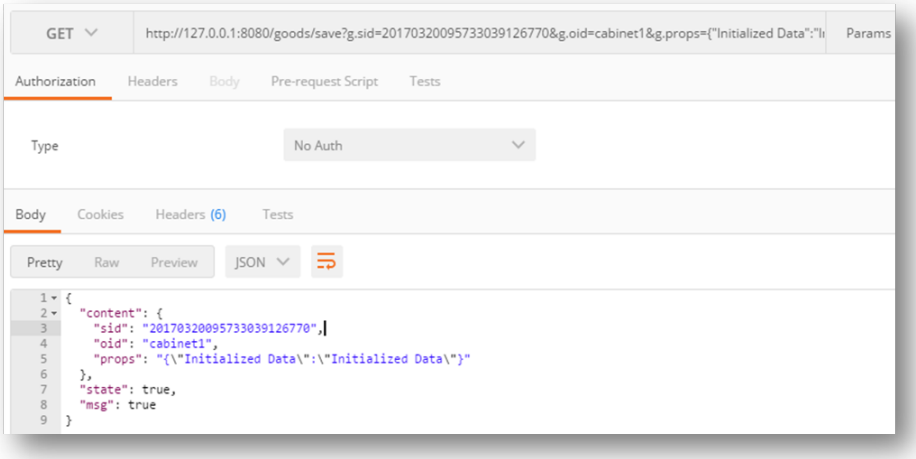
HTTP GET

- request

```
1     http://127.0.0.1:8080/goods/save?g.sid=20170320095733039126770&g.oid=cabinet1&g.
↪props={"Initialized Data":"Initialized Data"}
```

Note: request will push init data in JSON data {"Initialized Data":"Initialized Data"} to object `cabinet1` in scene `20170320095733039126770`

- server response



- sample request in html/javascript

```
<!DOCTYPE html>

<html>

<head>

<script src="jquery-1.11.1.min.js">

</script>

<script>

$(document).ready(function(){

$("button").click(function(){

$.get('http://127.0.0.1:8080/goods/save?g.sid=20170320095733039126770&g.oid=cabinet1&
↪g.props={"Initialized Data":"Initialized Data"}',

    function(data) { alert("Data:" + data);}

        );

});

});

</script>

</head>

<body>

```

```
<button>Submit data</button>

</body>

</html>
```

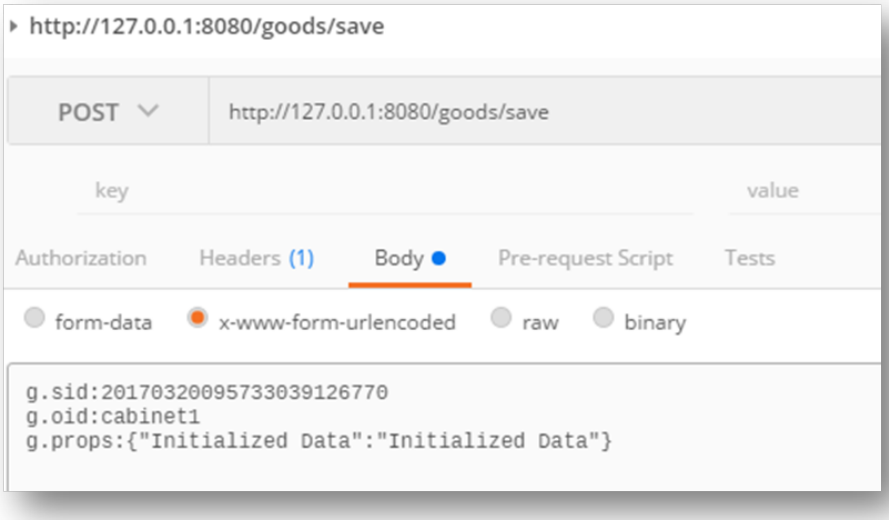
HTTP POST

- **request**
  - url, `http://*{Your Momoda Server IP}*:8080/goods/save`
  - POST message:

```
{

        g.sid:201703200957330391267 70

        g.oid:cabinet1

        g.props:{"Initialized Data":"Initialized Data"}

}
```

Note: request will push init data in JSON format {"Initialized Data":"Initialized Data"} to object `cabinet1` in scene `201703200957330391267 70`

- server response



- sample request in html/javascript

```
<!DOCTYPE html>

<html>

<head>

<script src="jquery-1.11.1.min.js">

</script>

<script>

$(document).ready(function(){

$("button").click(function(){

    $.post("http://127.0.0.1:8080/goods/save",

    {

    "g.sid":"20170320095733039126770",

    "g.oid":"cabinet1",

    "g.props":'{"Initialized Data":"Initialized Data"}'    },

    function(data) { alert("Data:" + data);}

        );

});

});

</script>

</head>

<body>

<button>Submit data</button>

</body>

</html>
```

M Interface

use the'M interface'to push *realtime* data to the scene, typical user case could be showing *realtime* information upon sensor objects. For instance, shows realtime alarms of fire/gas sensor, realtime location of cargo, current reading of temperature, etc.

Warning: Data pushed to M Interface is stored in Momoda server's message queue, and there are no *message replay* for the queue, so if a Momoda client is **newly connected** to server, it only shows the **current** message/data in queue.

Usage

url	http://*{Your Momoda Server IP}*:8080/data/putdata
http method	GET or POST
parameters	{JSON Message}

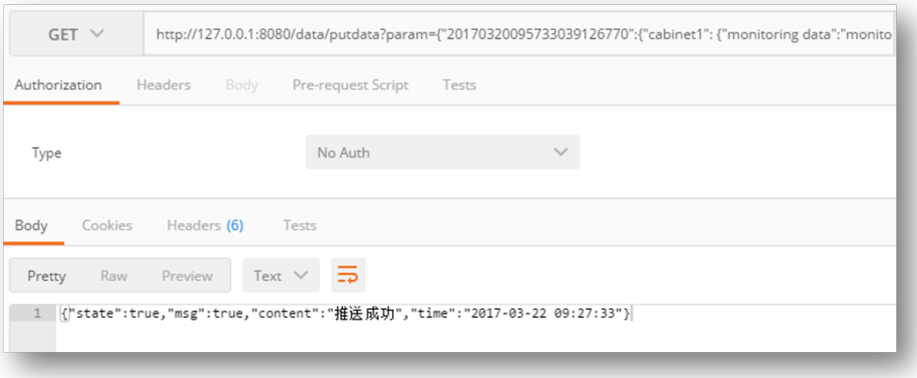
HTTP GET

- request

```
http://127.0.0.1:8080/data/putdata?param={"20170320095733039126770":{"cabinet1": {
↪"monitoring data":"monitoring data"}}}
```

Note: request will push init data in JSON data {"monitoring data":"monitoring data"} to object `cabinet1` in scene `20170320095733039126770`

- server response



- sample request in html/javascript

```
<!DOCTYPE html>

<html>

<head>

<script src="jquery-1.11.1.min.js">

</script>

<script>

$(document).ready(function(){

```

```
$("button").click(function(){

    $.get('http://127.0.0.1:8080/data/putdata?param={"201703200957330391267 70":{"cabinet1": {"monitoring data":"monitoring data"}}}',

    function(data) { alert("Data:" + data);}

        );

});

});

</script>

</head>

<body>

<button>Submit data</button>

</body>

</html>
```

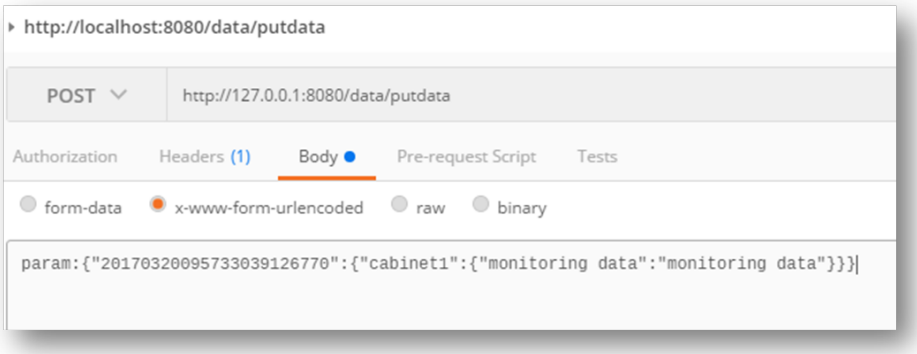
HTTP POST

- **request**
  - url, `http://*{Your Momoda Server IP}*:8080/data/putdata`
  - POST message:

```
{

   param:{"20170320095733039126770":{"cabinet1":{"monitoring data":"monitoring data"}
}
```

Note: request will push init data in JSON format {"monitoring data":"monitoring data"} to object `cabinet1` in scene `20170320095733039126770`

- sample request



- sample request in html/javascript

```
<!DOCTYPE html>

<html>

<head>

<script src="jquery-1.11.1.min.js">

</script>

<script>

$(document).ready(function(){

$("button").click(function(){

    $.post("http://127.0.0.1:8080/data/putdata",

    {

param:'{"20170320095733039126770":{"cabinet1":{"monitoring data":"monitoring data"}}}
'    },

    function(data) { alert("Data:" + data);}

        );

});

});

</script>

</head>

<body>

<button>Submit data</button>
```

```
</body>

</html>
```

R Interface

Remove any initialization data in a scene.

Usage

url	http:///*{Your Momoda Server IP}*:8080/goods/remove
http method	GET
parameters	• `sid` scene ID

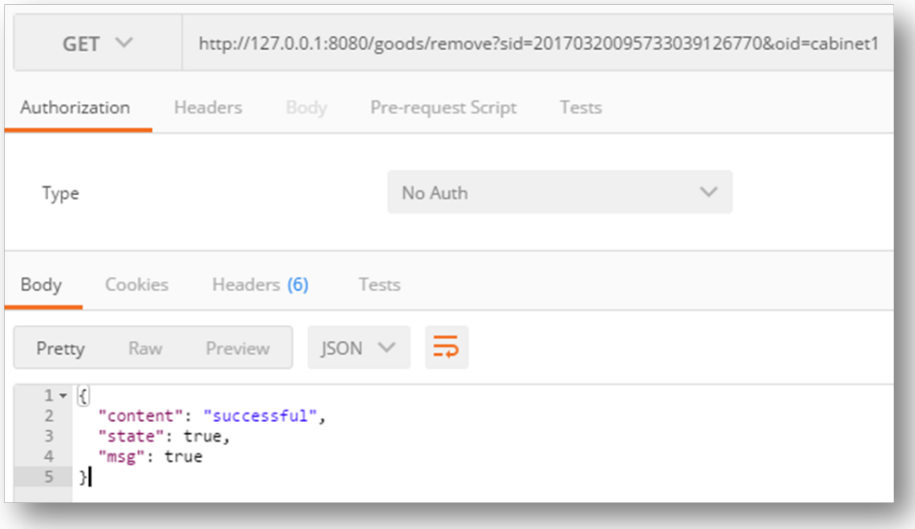
HTTP GET

- request

```
http://127.0.0.1:8080/goods/remove?sid=20170320095733039126770&oid=cabinet1
```

Note: request will remove all initialization data from object `cabinet1` in scene `20170320095733039126770`

- server response



- sample request in html/javascript

```
<!DOCTYPE html>

<html>

<head>

<script src="jquery-1.11.1.min.js">

</script>

<script>

$(document).ready(function(){

$("button").click(function(){

    $.get('http://127.0.0.1:8080/goods/remove?sid=20170320095733039126770&
↪oid=cabinet1',

    function(data) { alert("Data:" + data);}

        );

});

});

</script>

</head>

<body>
```

```
32
33  <button>Submit data</button>
34
35  </body>
36
37  </html>
```

CHAPTER 4

Try API Online

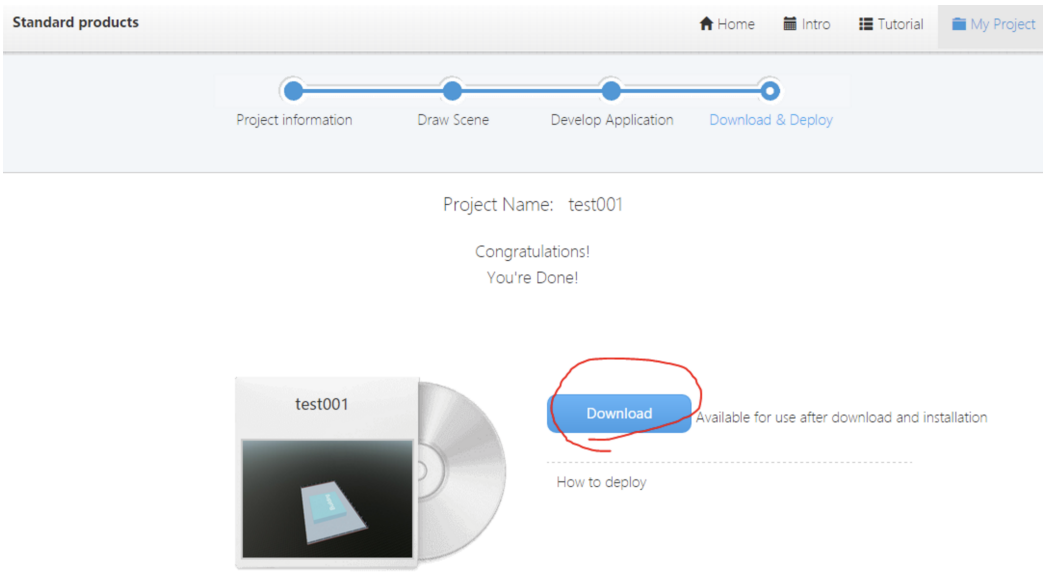
You can try uBuilder API online by click http://uinnova.com:9010/mmdapi

> **Warning:** The online API debugger supports **Internet Explorer 11** ONLY.

CHAPTER 5

Deploy

Step 1: Click ont the button shown below to download scene and API code to your PC



Step 2: Decompress the zip file

Step 3: Run start.bat

Step 4: Apply for license if needed.

CHAPTER 6

Q & A

- Does uBuilder API using the same syntax/library as Javascript

uBuilder API use javascript-like syntax, which is NOT exactly the same as javascript. For instance, there is no document object or windows object in uBuilder API.

- How to load my 3D scene into API developing UI?

In uBuilder API online development page, input your scene ID, then click "NEW" button.

- How to associate my API scripts to existing scene

Locate your scene in Momoda, click [Configure], then paste script source code into popup text box.

- Why I cannot play animation on custom object

Object animation is created during the model process, so object do not support play animation function if it do not associate any animation during the model process.

- Why browser do not response during API debugging

uBuilder API debugging requires 3D scene, which is resource consuming. Open more than one scenes at the same may slow the computer or even crash your browser. it is recommended that *simultaneously* opened 3D scene is *less than 3*.

- Why some object do not response to click event

While creating the scene, make sure to "selectable" checkbox is in checked status in object property settings.

CHAPTER 7

Revise History

2017-4-1API2.0 Release
• Total 72 API2.0 functions • 99 parameters in those functions • Offline version supports data initial interface, as well as realtime alarm interface • Support api call to data interface. Developer can save custom scene settings; • Bug fixes;