Commands

SItoA provides custom, script-able commands that allow you to manually execute some of the features of the plug-in.

Details of each command are given below. All the listed examples are in JScript.

SITOA_ExportScene

With this command, users can export the Softimage scene into Arnold's native scene definition format. One .ass file will be exported for each frame. These .ass files can later be rendered outside Softimage with the kick command-line executable (available on multiple platforms), or with any other custom standalone Arnold Renderer..

Syntax:

```
SITOA_ExportScene(frameStart, frameEnd, frameStep, createStandIn, selectionOnly,
filename);
```

- frameStart: start frame of the sequence to be exported.
- frameEnd: end frame.
- frameStep: frame step.
- createStandIn: if true, the options and camera are not exported, and a .asstoc file is also created, hosting the bounding box information of the exported objects. This is usually the option you want to use if you plan to use the exported ass file a a standin later.
- selectionOnly: if true, only the selected objects are exported.
- filename: the output filename.

Note: filename supports tokens. When exporting a sequence, you can use the [Frame] token in the filename, to have it resolved correctly at each frame. **Example**:

```
SITOA_ExportScene(1, 10, 1, false, false, "/usr/tmp/Test.[Frame #4].ass);
```

Exports 10 .ass files, named /usr/tmp/Test.0001.ass, /usr/tmp/Test.0002.ass, etc.

SITOA_ExportObjects

Same as above, but allowing to export an objects collection.

Syntax:

```
SITOA_ExportObjects(frameStart, frameEnd, frameStep, createStandIn, objects,
recurse, filename);
```

- frameStart, frameEnd, frameStep, createStandIn: same as above
- objects: collection of objects to be exported (defaulting to selection if void).
- recurse: if true, recursively export also the children of each item in the objects collection.
- filename: the output filename (supporting tokens from 2.6)

SITOA_DestroyScene

This command flushes the current scene loaded in Arnold, which is used for IPR rendering. Because it's difficult for SItoA to detect every single possible user-interface event, users can manually flush the scene, which will force Softimage to resend all the information to Arnold, thus picking up all the latest UI changes. If you find one of these cases, please report it to the developers and we will do our best to fix it.

Hint: we recommend assigning this command to one of the available hotkeys from *File / Keyboard Mapping*. F6 is a popular choice.

SITOA_FlushTextures

This command invalidates any texture maps in memory, forcing them to be reloaded from disk the next time they are accessed. This is very useful in interactive IPR sessions where the textures are being modified by you or somebody else on the network, and avoids an expensive full scene reload via SITOA_DestroyScene.

SITOA_ShowVersion

This command prints both the SItoA and Arnold versions of the add-on. This is useful when reporting bugs or suggestions.

Syntax:

```
SITOA_ShowVersion(in_log)
```

- in_log: if true, logs the message in the script editor.
  Default value: true

Returned value: a string, containing the SItoA and Arnold versions, separated by a ';' character.

SITOA_CreateRenderChannels

This command creates the following Arnold AOV core layers as Render Channels in the scene, if not there yet:

- Arnold_Alpha
- Arnold_Opacity
- Arnold_CPU Time
- Arnold_Ray_Count
- Arnold_Point
- Arnold_Pref
- Arnold_Motion_Vector
- Arnold_Shadow_Matte
- Arnold_Direct_Diffuse
- Arnold_Direct_Specular
- Arnold_Indirect_Diffuse
- Arnold_Indirect_Specular
- Arnold_Emission
- Arnold_Refraction
- Arnold_Reflection
- Arnold_Refraction_Opacity
- Arnold_SSS
- Arnold_Specular
- Arnold_Diffuse_Albedo
- Arnold_Sheen
- Arnold_Direct_Transmission
- Arnold_Volume
- Arnold_Volume_Opacity

SITOA_RegenerateRenderOptions

Regenerates the Arnold Render Options panel, updating the layout, adding new parameters, removing old or non-existent parameters. It will assign automatically to new Render Options all previous values the parameters had. Users will have to execute it in old scenes to update Render Options parameters. Take into account that this command doesn't refresh the arnold properties already applied to objects.

SITOA_GetMotionBlurKeys

Returns the motion blur keys time samples as a float array

Syntax:

```
SITOA_GetMotionBlurKeys(objName, deformation, frame);
```

- objName: the object to get the motion blur keys from (they can override the global values).
  Default value: selection
- deformation: if true, get the deformation motion blur keys, otherwise the transformation ones.
  Default value: false
- frame: the frame where the computation is done.
  Default value: the current frame

Example: log the deformation key times for object "torus" at frame 1

```
var keys = SITOA_GetMotionBlurKeys("torus", true, 1);
var arr = keys.toArray();
for (var i=0; i<arr.length; i++)
   logMessage(arr[i]);
```

SITOA_OpenVdbGrids

Returns the names of the grids contained in a VDB file, as a single string.

Example: log the grids in "C:\temp\fireball.vdb"

```
var names = SITOA_OpenVdbGrids("C:\\Temp\\fireball.vdb");
LogMessage(names);
```

SITOA_AddVisibilityProperties

Adds the visibility properties to the selected (or input) objects.

SITOA_AddSidednessProperties

Adds the sidedness properties to the selected (or input) objects.

SITOA_AddParametersProperties

Adds the parameters properties to the selected (or input) objects.

SITOA_AddStandinProperties

Adds the standin properties to the selected (or input) objects.

SITOA_AddUserOptionsProperties

Adds the user options property to the selected (or input) objects.

SITOA_AddMatteProperties

Adds the matte property to the selected (or input) objects.

SITOA_AddTextureOptionsProperties

Adds the texture options property to the selected (or input) images.

SITOA_AddCameraOptionsProperties

Adds the camera options property to the selected (or input) cameras.

SITOA_AddVolumeProperties

Adds the volume property to the selected (or input) objects.

The commands for applying properties share the same **syntax** and behavior. Taking the visibility case,

```
SITOA_AddVisibilityProperties(in_collection, in_inspect)
```

- in_collection: the object or collection of objects you want to apply the property to.
  Default value: selection
- in_inspect : true if the property must be inspected after being applied.
  Default value: false

Returned value: the collection of applied properties, so you can further edit their parameters in your custom script.

Example: assign the visibility property to cube and sphere, and set "shadow" off, without inspecting the ppgs

```
var props = SITOA_AddVisibilityProperties("cube,sphere", false);
for (var i=0; i<props.count; i++)
{
   var prop = props.Item(i);
   prop.Parameters("shadows").Value = false;
}
```

SITOA_AddPointLight

Adds an Arnold point light to the scene.

SITOA_AddDistantLight

Adds an Arnold distant light to the scene.

SITOA_AddSpotLight

Adds an Arnold spot light to the scene.

SITOA_AddSkydomeLight

Adds an Arnold skydome light to the scene.

SITOA_AddDiskLight

Adds an Arnold disk light to the scene.

SITOA_AddQuadLight

Adds an Arnold quad light to the scene.

SITOA_AddMeshLight

Adds an Arnold mesh light to the scene

SITOA_AddPhotometricLight

Adds an Arnold photometric light to the scene

The commands for creating lights, except SITOA_AddMeshLight, share the same **syntax** and behavior. Taking the point light case,

```
SITOA_AddPointLight(in_name)
```

- in_name: the name assigned to the light.
  Default value: "Point"

Returned value: the created light.

SITOA_AddMeshLight has an extra mandatory parameter

```
SITOA_AddMeshLight(in_name, in_obj)
```

- in_obj: the object to be used as shape.

Example: create an Arnold point light, set its intensity to 2, and inspect the light

```
var light = SITOA_AddPointLight("Nice_Light");
SetValue(light + ".light.arnold_point_light.intensity", 2.0, null);
InspectObj(light);
```

SITOA_AddLightDecayFilter

Adds the light decay filter shader to the selected (or input) lights.

SITOA_AddBarndoorFilter

Adds the barndoor filter shader to the selected (or input) lights.

SITOA_AddGoboFilter

Adds the gobo filter shader to the selected (or input) lights.

SITOA_AddLightBlockerFilter

Adds the light blocker filter shader to the selected (or input) lights.

SITOA_AddPassthroughFilter

Adds the sta_passthrough filter shader to the selected (or input) lights.

The commands for applying filters share the same **syntax** and behavior. Taking the barndoor case,

```
SITOA_AddBarndoorFilter(in_collection, in_inspect)
```

- in_collection: the light or collection of lights you want to apply the filter to.
  Default value: selection
- in_inspect : true if the filter shader must be inspected after being applied.
  Default value: false

Returned value: the collection of applied filters, so you can further edit their parameters in your custom script.

Example: apply a decay filter to Point and Spot, enable the Near Attenuation flag, and set the Near Start/End values to 5 and 10, without inspecting the ppgs

```
var filters = SITOA_AddLightDecayFilter("Point,Spot", false);
for (var i=0; i<filters.count; i++)
{
   var filter = filters.Item(i);
   filter.Parameters("use_near_atten").Value = true;
   filter.Parameters("near_start").Value = 5;
   filter.Parameters("near_end").Value = 10;
}
```