CHANGELOG Python API

1.5.8 to 2018.1

PiXYZ Software
Python API changelog 1.5.8 to 2018.1
contact@pixyz-software.com

Table des matières

A - NEW FUNCTIONS

Module: Algo

copyUV(...)

copyUV(scenePaths[, sourceChannel[, destinationChannel]])

Copy an UV channel to another UV channel

Parameters:

scenePaths (ScenePathList) : scene paths of part to process

sourceChannel (Int) : the source uv channel to copy [optional] (default: 0)

destinationChannel (Int) : the destination uv channel to copy into [optional] (default: 0)

bakeMaps(...)

bakeMaps(destinationOccurrences, sourceOccurrences, wantedMaps[, channel[, resolution]]) -> bakedMaps

Experimental :

Parameters:

destinationOccurrences (ScenePathList) : Scene path of the mesh where to store the baked map

sourceOccurrences (ScenePathList) : Scene paths of components from which to bake maps (if empty use destination)

wantedMaps (StringList) : List of map name to generate (Normal, Diffuse)

channel (Int) : UV channel of destOccurrence to use for the map generation [optional] (default: 0)

resolution (Int) : Map resolution [optional] (default: 2048)

Return value:

bakedMaps (ImageList) : Baked map list

bakeUV(...)

bakeUV(source, destination[, sourceChannel[, destinationChannel[, tolerance]]])

Bake UV from a mesh to another mesh

Parameters:

source (ScenePath) : Scene path to the source mesh

destination (ScenePath) : Scene path to the destination mesh

sourceChannel (Int) : Source UV channel to bake [optional] (default: 0)

destinationChannel (Int) : Destination UV channel to bake to [optional] (default: 0)

tolerance (Distance) : Tolerance when point is projected on seam (if the model come from a decimation it is recommanded to use the lineic tolerance here) [optional] (default: 0.001)

crackMoebiusStrips(...)

crackMoebiusStrips(scenePaths[, maxEdgeCount])

Remove moebius strip by topologically cracking them (make it orientable)

Parameters:

scenePaths (ScenePathList) : ScenePath of components to repair

maxEdgeCount (Int) : Maximum number of edges to crack to remove one moebius strip [optional] (default: 3)

createIdentifiedPatchesFromPatches(...)

createIdentifiedPatchesFromPatches(scenePaths)

Create identified patch from existing patch (this is usefull before cloning for baking)

Parameters:

scenePaths (ScenePathList) : Scene paths of components to process

//!\\: experimental

equilateralize(...)

equilateralize(scenePaths[, maxIterations])

Sswap edges to make triangles more equilateral

Parameters:

scenePaths (ScenePathList) : Scene paths of components to process

maxIterations (Int) : Maximum number of swapping iteration [optional] (default: 1)

hiddenSelection(...)

hiddenSelection(scenePaths, resolution, sphereCount[, fovX[, screenSizePerView]])

Select parts not viewed from a sphere around the scene

Parameters:

scenePaths (ScenePathList) : Scene paths of components to process

resolution (Int) : Resolution of the visibility viewer

sphereCount (Int) : Segmentation of the sphere sphereCount x sphereCount

fovX (Double) : Horizontal field of view (in degree) [optional] (default: 90)

screenSizePerView (Coeff) : Pixel ratio by view to set an element as viewed (if ratio is null or negative, only one pixel is needed) [optional] (default: -1)

mapUvOnCubicAABB(...)

mapUvOnCubicAABB(scenePaths, uv3dSize[, channel[, overideExistingUvs]])

Generate texture coordinates using the projection on object AABB, with same scale on each axis

Parameters:

scenePaths (ScenePathList) : Scene paths of part to process

uv3dSize (Distance) : 3D size of the UV space [0-1]

channel (Int) : The UV channel which will contains the texture coordinates [optional] (default: 0)

overideExistingUvs (Boolean) : If True, overide existing UVs on channel [optional] (default: true)

normalizeUV(...)

normalizeUV(scenePaths, srcUVChannel[, dstUVChannel[, uniform[, sharedUVSpace]]])

Normalize UVs to fit in the [0-1] uv space

Parameters:

scenePaths (ScenePathList) : Scene paths of part to process

srcUVChannel (Int) : UV Channel to normalize

dstUVChannel (Int) : UV channel to store the normalized UV (if -1, srcUVChannel will be replaced) [optional] (default: -1)

uniform (Boolean) : If true, the scale will be uniform. Else UV can be deformed with a non-uniform scale [optional] (default: true)

sharedUVSpace (Boolean) : If true, all parts will be processed as if they were merged to avoid overlapping of their UVs [optional] (default: true)

optimizeSubMeshes(...)

optimizeSubMeshes(scenePaths)

Sort sub meshes by materials

Parameters:

scenePaths (ScenePathList) : ScenePath of parts to process

proxyMesh(...)

proxyMesh(scenePaths, voxelSize[, projectPoints]) -> proxyMeshPart

Replace the tessellations of the selected parts by a proxy mesh based on a voxelization

Parameters:

scenePaths (ScenePathList) : Scene paths of part to process

voxelSize (Distance) : Size of voxels

projectPoints (Boolean) : Project voxels corners to geometry [optional] (default: true)

Return value:

proxyMeshPart (ScenePath) : Resulting part occurrence

removeDegeneratedPolygons(...)

removeDegeneratedPolygons(scenePaths, tolerance)

Remove some kinds of degenerated polygons

Parameters:

scenePaths (ScenePathList) : Scene paths of components to process

tolerance (Distance) : Degenerated tolerance

removeUV(...)

removeUV(scenePaths[, channel])

Remove one or all UV channel(s)

Parameters:

scenePaths (ScenePathList) : Scene paths of part to process

channel (Int) : The UV channel to remove (all if channel=-1) [optional] (default: -1)

smartHiddenCreateVisibilityInformation(...)

smartHiddenCreateVisibilityInformation(scenePaths, voxelSize, minimumCavityVolume, resolution[, onlyOutter])

Create visilibity information on parts viewed from a set of camera automatically generated

Parameters:

scenePaths (ScenePathList) : Scene paths of components to process

voxelSize (Distance) : Size of the voxels in mm (smaller it is, more viewpoints there are)

minimumCavityVolume (Volume) : Size of the voxels in cubic meter (smaller it is, more viewpoints there are)

resolution (Int) : Resolution of the visibility viewer

onlyOutter (Boolean) : Only add camera on the outter (not in cavities) [optional] (default: false)

smartHiddenSelection(...)

smartHiddenSelection(scenePaths, voxelSize, minimumCavityVolume, resolution[, onlyOutter])

Select parts not viewed from a set of camera automatically generated

Parameters:

scenePaths (ScenePathList) : Scene paths of components to process

voxelSize (Distance) : Size of the voxels in mm (smaller it is, more viewpoints there are)

minimumCavityVolume (Volume) : Size of the voxels in cubic meter (smaller it is, more viewpoints there are)

resolution (Int) : Resolution of the visibility viewer

onlyOutter (Boolean) : Only add camera on the outter (not in cavities) [optional] (default: false)

smoothUV(...)

smoothUV(scenePaths[, iterations[, channel]])

Smooth texture coordinates

Parameters:

scenePaths (ScenePathList) : Scene paths of part to process

iterations (Int) : Number of smooth iterations [optional] (default: 1)

channel (Int) : The UV channel which will contains the texture coordinates to smooth [optional] (default: 0)

voxelize(...)

voxelize(scenePaths, voxelSize) -> voxelizedPart

Replace the tessellations of the selected parts by a voxelization of the external skin

Parameters:

scenePaths (ScenePathList) : Scene paths of part to process

voxelSize (Distance) : Size of voxels

Return value:

voxelizedPart (ScenePath) : Resulting part occurrence

Module: Scene

exportImage(...)

exportImage(image, filename)

Export an image

Parameters:

image (Image) : identifier of the image to export

filename (OutputFilePath) : filename of the file to export

importImage(...)

importImage(filename) -> image

Import an image

Parameters:

filename (FilePath) : filename of the image to import

Return value:

image (Image) : identifier of the imported image

getAllImages(...)

getAllImages() -> images

Returns all the images loaded in the current session

Return value:

images (ImageList) : A list containing all images identifiers

getMaterialPattern(...)

getMaterialPattern(material) -> pattern

Returns the pattern used by the given material

Parameters:

material (Material) : a material

Return value:

pattern (MaterialPattern) : the material pattern used by the material

Module: CAD

createCylinderSurface(...)

createCylinderSurface(radius[, matrix]) -> cylinderSurface

Create a new cylinder surface

Parameters:

radius (Distance) : Radius of the cylinder

matrix (Matrix4) : Positionning matrix of the cylinder [optional] (default: geom.IdentityMatrix4)

Return value:

cylinderSurface (Surface) : The new cylinder surface

Module: Core

checkForUpdates(...)

checkForUpdates() -> newVersionAvailable

check for software update

Return value:

newVersionAvailable (Bool) : True if there is a new version available of this product

Module: Geom

Adding constant value:

IdentityMatrix4 = [[1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0,1]]

Module: IO

exportSelection(...)

exportSelection(filename)

Export the selection to a file

Parameters:

filename (OutputFilePath) : Path of the file to export

Module: Material

INFO : New module named “Material”. Regroups all the material function previously available in module “Scene”

convertAllMaterialsToColors(...)

convertAllMaterialsToColors()

Converts all the material in the scene to color materials

createMaterial(...)

createMaterial(name, pattern) -> material

Create a new material from pattern

Parameters:

name (String) : Name of the material

pattern (String) : Name of the pattern

Return value:

material (Material) : The created material

exportImage(...)

exportImage(image, filename)

Export an image

Parameters:

image (Image) : Identifier of the image to export

filename (OutputFilePath) : Filename of the file to export

findCustomMaterialPatternByName(...)

findCustomMaterialPatternByName(name) -> pattern

Returns the material pattern which has the given name

Parameters:

name (String) : The name of the material pattern

Return value:

pattern (CustomMaterialPattern) : The material pattern

findMaterialsByPattern(...)

findMaterialsByPattern(pattern) -> materials

Returns all materials using the given pattern

Parameters:

pattern (String) : A material pattern

Return value:

materials (MaterialList) : Materials using the pattern

findMaterialsByProperty(...)

findMaterialsByProperty(propertyName, propertyValue) -> materials

Returns all materials which match a given property value

Parameters:

propertyName (String) : Name of the property to match

propertyValue (Regex) : Regular expression to match for the property value

Return value:

materials (MaterialList) : Materials matching the property value

getAllImages(...)

getAllImages() -> images

Returns all the images loaded in the current session

Return value:

images (ImageList) : A list containing all images identifiers

getAllMaterialPatterns(...)

getAllMaterialPatterns() -> shaders

Returns all the material patterns in the current session

Return value:

shaders (StringList) : A list containing all material patterns

getAllMaterials(...)

getAllMaterials() -> materials

Retrieve the list of all the materials in the material library

Return value:

materials (MaterialList) : List of materials in the material library

importImage(...)

importImage(filename) -> image

Import an image

Parameters:

filename (FilePath) : Filename of the image to import

Return value:

image (Image) : Identifier of the imported image

importSubstanceMaterials(...)

importSubstanceMaterials(directory)

Import Substance PBR materials from a directory and assign them to parts if possible

Parameters:

directory (FilePath) : Directory path

Module: Polygonal

createMeshFromDefinition(...)

createMeshFromDefinition(meshDefinition) -> tessellation

Parameters:

meshDefinition (MeshDefinition) : Mesh definition

Return value:

tessellation (Tessellation) : The new mesh

Module: Scenarios

INFO : New module named “Scenarios”. Contains specific data preparation processes and workflows.

generateLODChain(...)

generateLODChain(scenePaths)

Automatically create LODs for game engines (Unity and Unreal Engine only)

Parameters:

scenePaths (ScenePathList) : Scene paths of components to process

generatePhantomMesh(...)

generatePhantomMesh()

Copyright FB.

generateProxyMesh(...)

generateProxyMesh(scenePaths, precision[, keepOriginalMesh])

Parameters:

scenePaths (ScenePathList) : Scene paths of components to process

precision (Distance) :

keepOriginalMesh (Boolean) : If True, initial shapes will be kept [optional] (default: true)

runProcessForGameEngines(...)

runProcessForGameEngines(overrideExistingUVs, repackUVs)

Automatic preparation process for games engines. Ready for FBX export

Parameters:

overrideExistingUVs (Boolean) :

repackUVs (Boolean) :

selectIdentical(...)

selectIdentical(scenePaths, volumeRatio, polyRatio)

Parameters:

scenePaths (ScenePathList) : Scene path of component to process

volumeRatio (Double) :

polyRatio (Double) :

Module: SceneGraph

createMotionGroupAtPosition(...)

createMotionGroupAtPosition(name, position) -> group

Create a new Motion Group at the given position

Parameters:

name (String) : the name of the group

position (Point3) : The position of the new motion group

Return value:

group (MotionGroup) : The created Motion Group

Module: Scene

cleanUnusedMaterials(...)

cleanUnusedMaterials()

Remove unused materials from material library

createSymmetry(...)

createSymmetry(paths, plane)

Create symmetries from selection

Parameters:

paths (ScenePathList) : Selection of occurrences

plane (Plane) : Symmetry plane

deleteEmptyAssemblies(...)

deleteEmptyAssemblies()

Delete all empty essemblies

ScenePaths are now represented internally by Occurences entities. New functions are thus available to handle ScenePaths. See « DEPRECATED" section for deleted features.

getOccurrenceChildren(...)

getOccurrenceChildren(occurrence) -> children

Returns the children of the given occurrence

Parameters:

occurrence (Occurrence) : The parent occurrence

Return value:

children (OccurrenceList) : The children occurrences

getOccurrenceFromScenePath(...)

getOccurrenceFromScenePath(path) -> occurrence

Returns the complete scene path of a given occurrence

Parameters:

path (ScenePath) : The scene path to retrieve

Return value:

occurrence (Occurrence) : The occurrence of the given scene path

getOccurrenceParent(...)

getOccurrenceParent(occurrence) -> parent

Returns the parent of the given occurrence

Parameters:

occurrence (Occurrence) : The child occurrence

Return value:

parent (Occurrence) : The parent occurrence

getOccurrencesOnSceneNode(...)

getOccurrencesOnSceneNode(node) -> occurrences

Returns all occurrences of the given scene node

Parameters:

node (SceneNode) : A Scene node

Return value:

occurrences (OccurrenceList) : All occurrences of the given scene node

getSceneNodeOfOccurrence(...)

getSceneNodeOfOccurrence(occurrence) -> node

Return the scene node instanciated by the given occurrence

Parameters:

occurrence (Occurrence) : The occurrence to retrieve

Return value:

node (SceneNode) : The scene node of the given occurrence

getScenePathFromOccurrence(...)

getScenePathFromOccurrence(occurrence) -> path

Returns the complete scene path of a given occurrence

Parameters:

occurrence (Occurrence) : The occurrence to retrieve

Return value:

path (ScenePath) : The scene path of the given occurrence

identifyMultipleOccurrences(...)

identifyMultipleOccurrences(minOccurrenceCount)

Identify parts with more than one occurrence on the scene

Parameters:

minOccurrenceCount (Int) : Min occurrence count

mergePartsByMaterials(...)

mergePartsByMaterials(partPaths) -> mergedAssembly

Merge a set of parts

Parameters:

partPaths (ScenePathList) : Scene path of the parts to merge

Return value:

mergedAssembly (Assembly) : Resulting assembly of the merged parts

renameLongPartName(...)

renameLongPartName(maxLength)

truncate names of part with too long names

Parameters:

maxLength (Int) : Maximum name length

selectDuplicatedParts(...)

selectDuplicatedParts([acceptVolumeRatio[, acceptPolycountRatio[, acceptAABBAxisRatio[, acceptAABBCenterDistance]]]])

Select duplicated parts

Parameters:

acceptVolumeRatio (Real) : If the ratio of volumes of two part is lower than acceptVolumeRatio, they will be considered duplicated [optional] (default: 0.01)

acceptPolycountRatio (Real) : If the ratio of polygon counts of two part is lower than acceptPolycountRatio, they will be considered duplicated [optional] (default: 0.1)

acceptAABBAxisRatio (Real) : If the ratio of AABB axis of two part is lower than acceptAABBAxisRatio, they will be considered duplicated [optional] (default: 0.01)

acceptAABBCenterDistance (Distance) : If the ratio of AABB centers of two part is lower than acceptAABBCenterRatio, they will be considered duplicated [optional] (default: 0.1)

selectPartsFromNoShow(...)

selectPartsFromNoShow()

Select hidden parts

selectSmallParts(...)

selectSmallParts(maxDiagLength[, maxLength])

Select small parts

Parameters:

maxDiagLength (Distance) : If the diagonal axis of the bouding box is less than maxDiagLength, part will be selected. -1 to ignore

maxLength (Distance) : If the longer axis of the box is less than maxLength, part will be selected. -1 to ignore [optional] (default: -1)

Module: View

createMotionGroupOnCameraTarget(...)

createMotionGroupOnCameraTarget(name) -> motionGroup

Create a motion group on the target position of the camera (mid-click)

Parameters:

name (String) : Name of the new group

Return value:

motionGroup (MotionGroup) : The created motion group

B - DEPRECATED / DELETED FUNCTIONS

All functions related to ScenePaths properties are now deprecated. Please consider using the function "getOccurrenceFromScenePath » to retrieve the Occurrence entity related to a ScenePath, then use the function core.getProperty to access a specific property.

getOccurrenceProperties(...)

getOccurrenceProperties(occurrence) -> properties

Deprecated :Return all the properties set to a given occurrence

Parameters:

occurrence (ScenePath) : The occurrence scene path

Return value:

properties (StringList) : Names of the properties set to this occurrence

getOccurrenceProperty(...)

getOccurrenceProperty(occurrence, property) -> value

Deprecated :Return the value of an occurrence property

Parameters:

occurrence (ScenePath) : The occurrence scene path

property (String) : The name of the property

Return value:

value (String) : The value of the property

hasOccurrenceProperty(...)

hasOccurrenceProperty(occurrence, property) -> propertyExists

Deprecated :Return true if a given property is set to an occurrence

Parameters:

occurrence (ScenePath) : The occurrence scene path

property (String) : The name of the property

Return value:

propertyExists (Boolean) : True if the property exists, else False

Module: Algo

deleteUnfoldWeightAttribute(...)

deleteUnfoldWeightAttribute(scenePaths)

Delete Unfold Weight attributes on tessellations

Parameters:

scenePaths (ScenePathList) : scenePath of components to delete attributes

stitchUV(...)

stitchUV(scenePaths[, channel])

Try to stitch existing uv islands

Parameters:

scenePaths (ScenePathList) : scene paths of part to process

channel (Int) : the uv channel to repack [optional] (default: 0)

transferVisibilityToUnfoldWeight(...)

transferVisibilityToUnfoldWeight(scenePaths)

Set Unfold Weight Attribute from Visibility Attribute

Parameters:

scenePaths (ScenePathList) : scene paths of part to process

Module: Scene

All functions related to materials management are now in the Module:Material.
WARNING : their names and attributes may have changed.

createMaterial(...)

createMaterial(name, pattern) -> material

Create a new material from pattern

Parameters:

name (String) : name of the material

pattern (String) : name of the pattern

Return value:

material (Material) : the created material

exportImage(...)

exportImage(image, filename)

Export an image

Parameters:

image (Image) : identifier of the image to export

filename (OutputFilePath) : filename of the file to export

findMaterialsByPattern(...)

findMaterialsByPattern(pattern) -> materials

Returns all materials using the given pattern

Parameters:

pattern (MaterialPattern) : a material pattern

Return value:

materials (MaterialList) : materials using the pattern

findMaterialsByProperty(...)

findMaterialsByProperty(propertyName, propertyValue) -> materials

Returns all materials which match a given property value

Parameters:

propertyName (String) : name of the property to match

propertyValue (Regex) : regular expression to match for the property value

Return value:

materials (MaterialList) : materials matching the property value

getAllImages(...)

getAllImages() -> images

Returns all the images loaded in the current session

Return value:

images (ImageList) : A list containing all images identifiers

getAllMaterialPatterns(...)

getAllMaterialPatterns() -> shaders

Returns all the material patterns in the current session

Return value:

shaders (MaterialPatternList) : A list containing all material patterns

getAllMaterials(...)

getAllMaterials() -> materials

Retrieve the list of all the materials in the material library

Return value:

materials (MaterialList) : list of materials in the material library

importImage(...)

importImage(filename) -> image

Import an image

Parameters:

filename (FilePath) : filename of the image to import

Return value:

image (Image) : identifier of the imported image

importSubstanceMaterials(...)

importSubstanceMaterials(directory)

Import Substance PBR materials from a directory and assign them to parts if possible

Parameters:

directory (FilePath) : directory path

findMaterialPatternByName(...)

findMaterialPatternByName(name) -> pattern

Returns the material pattern which has the given name

Parameters:

name (String) : the name of the material pattern

Return value:

pattern (MaterialPattern) : the material pattern

C - CHANGES – UPDATES

Module: Core

Adding the parameter "installForAllUsers" to the function installPlugin

installPlugin(...)

installPlugin(pluginFile[, installForAllUsers])

Install a new plugin

Parameters:

pluginFile (FilePath) : path to the plugin to be installed

=> installForAllUsers (Boolean) : if false only the current user will see the plugin installed [optional] (default: true)

Module: Algo

Adding optionnal parameter "channel" to function applyUvTransform

applyUvTransform(...)

applyUvTransform(scenePaths, matrix[, channel])

Apply a transformation matrix on texture coordinates

Parameters:

scenePaths (ScenePathList) : scene paths of part to process

matrix (Matrix4) : transformationmatrix

=> channel (Int) : uv channel to transform [optional] (default: 0)

Adding optionnal parameter "channel" to function automaticUVMapping

//!\\ WARNING : Order of parameters has changed

automaticUVMapping(...)

automaticUVMapping(scenePaths[, channel[, maxAngleDistorsion[, maxAreaDistorsion[, sharpToSeam[, forbidOverlapping[, scaleToOne]]]]]])

Experimental :Generates the texture coordinates and automatically cut

Parameters:

=> scenePaths (ScenePathList) : scene paths of part to process

channel (Int) : the uv channel which will contains the texture coordinates [optional] (default: 0)

maxAngleDistorsion (Double) : maximum angle distorsion |2PI-SumVtxAng|/2PI [optional] (default: 0.5)

maxAreaDistorsion (Double) : maximum area distorsion before scale to 1. |2DArea-3DArea|/3DArea [optional] (default: -1)

sharpToSeam (Bool) : If enabled, sharp edges are automatically considered as UV seams [optional] (default: true)

forbidOverlapping (Bool) : If enabled, uv cannot overlap [optional] (default: true)

scaleToOne (Bool) : Scale UV to fit between [0-1] [optional] (default: true)

Function « createTangents » now uses an optional uvChannel argument

createTangents(...)

createTangents(scenePaths[, sharpEdge[, uvChannel]])

Create tangent attributes on tessellations

Parameters:

scenePaths (ScenePathList) : ScenePath of components to create attributes

sharpEdge (Angle) : Edges with an angle between their polygons greater than sharpEdge will be considered sharp (default use the PiXYZ sharpAngle parameter) [optional] (default: -1)

uvChannel (Int) : UV channel to use for the tangents creation [optional] (default: 0)

//!\\: Obsolete => prefer bakeMaps

2 optionnal parameters added : tangentSpace et uvChannel

generateNormalMap(...)

generateNormalMap(scenePaths, resolution[, shareMap[, tangentSpace[, uvChannel]]]) -> normalMaps

Experimental :

Parameters:

scenePaths (ScenePathList) : ScenePath of components for map generation

resolution (Int) : Map resolution

shareMap (Boolean) : Map is shared by all given scene paths. If UV overlaps, the result is undefined [optional] (default: false)

tangentSpace (Boolean) : If true, normal map is generated in tangent space, else it is generated in object space [optional] (default: true)

uvChannel (Int) : Uv channel to use for the map generation, the UVs must be normalized (fit in [0-1]) [optional] (default: 0)

Return value:

normalMaps (ImageList) : List of normal maps generated for the given scene paths, if shareMap=true, the list contains only one image

New ScenePaths parameter added for all « hidden removal » related functions. Allows to target specific assemblies to be processed during the hidden removal process. Unselect entities will be ignored but used for occlusion purposes.

getHiddenOccurrences(...)

getHiddenOccurrences(scenePaths, resolution, sphereCount[, fovX[, screenSizePerView]]) -> hiddenOccurrences

Return parts path not viewed from a sphere around the scene

Parameters:

=> scenePaths (ScenePathList) : Scene paths of components to process

resolution (Int) : Resolution of the visibility viewer

sphereCount (Int) : Segmentation of the sphere sphereCount x sphereCount

fovX (Double) : Horizontal field of view (in degree) [optional] (default: 90)

screenSizePerView (Coeff) : Pixel ratio by view to set an element as viewed (if ratio is null or negative, only one pixel is needed) [optional] (default: -1)

Return value:

hiddenOccurrences (ScenePathList) : Hidden occurrences

hiddenRemoval(...)

hiddenRemoval(scenePaths, level, resolution, sphereCount[, fovX[, screenSizePerView]])

Delete parts, patches or polygons not viewed from a sphere around the scene

Parameters:

=> scenePaths (ScenePathList) : Scene paths of components to process

level (SelectionLevel) : Level of parts to remove : Parts, Patches or Polygons
resolution (Int) : Resolution of the visibility viewer
sphereCount (Int) : Segmentation of the sphere sphereCount x sphereCount
fovX (Double) : Horizontal field of view (in degree) [optional] (default: 90)
screenSizePerView (Coeff) : Pixel ratio by view to set an element as viewed (if ratio is null, only one pixel is needed) [optional] (default: 0)

hiddenRemovalCamera(...)

hiddenRemovalCamera(scenePaths, level, cameraPositions, resolution, sphereCount[, fovX])

Delete parts, patches or polygons not viewed from spheres generated with a set of camera position

Parameters:

=> scenePaths (ScenePathList) : Scene paths of components to process
level (SelectionLevel) : Level of parts to remove : Parts, Patches or Polygons
cameraPositions (Point3List) : List of camera positions
resolution (Int) : Resolution of the visibility viewer
sphereCount (Int) : Segmentation of the sphere sphereCount x sphereCount
fovX (Double) : Horizontal field of view (in degree) [optional] (default: 90)

hiddenRemovalViewPoints(...)

hiddenRemovalViewPoints(scenePaths, level, cameraPositions, cameraDirections, cameraUps, resolution[, fovX])

Delete parts, patches or polygons not viewed from a set of camera position/orientation

Parameters:

=> scenePaths (ScenePathList) : Scene paths of components to process
level (SelectionLevel) : Level of parts to remove : Parts, Patches or Polygons
cameraPositions (Point3List) : List of camera positions
cameraDirections (Point3List) : List of camera directions
cameraUps (Point3List) : List of camera up vectors
resolution (Int) : Resolution of the visibility viewer
fovX (Double) : Horizontal field of view (in degree) [optional] (default: 90)

smartHiddenRemoval(...)

smartHiddenRemoval(scenePaths, level, voxelSize, minimumCavityVolume, resolution[, onlyOutter])

Delete parts, patches or polygons not viewed from a set of camera automatically generated

Parameters:

=> scenePaths (ScenePathList) : Scene paths of components to process
level (SelectionLevel) : Level of parts to remove : Parts, Patches or Polygons
voxelSize (Distance) : Size of the voxels in mm (smaller it is, more viewpoints there are)
minimumCavityVolume (Volume) : Size of the voxels in cubic meter (smaller it is, more viewpoints there are)
resolution (Int) : Resolution of the visibility viewer
onlyOutter (Boolean) : Only add camera on the outter (not in cavities) [optional] (default: false)

Adding the optional parameter « overrideExistingUvs » to the functions mapUvOnAABB and mapUvOnCustomAABB

mapUvOnAABB(...)

mapUvOnAABB(scenePaths, useLocalAABB, uv3dSize[, channel[, overideExistingUvs]])

Generate texture coordinates using the projection on object AABB

Parameters:

scenePaths (ScenePathList) : Scene paths of part to process

useLocalAABB (Bool) : If enabled, uses part own bounding box, else use global one

uv3dSize (Distance) : 3D size of the UV space [0-1]

channel (Int) : The UV channel which will contains the texture coordinates [optional] (default: 0)

=> overideExistingUvs (Boolean) : If True, overide existing UVs on channel [optional] (default: true)

mapUvOnCustomAABB(...)

mapUvOnCustomAABB(scenePaths, aabb, uv3dSize[, channel[, overideExistingUVs]])

Generate texture coordinates using the projection on custom AABB

Parameters:

scenePaths (ScenePathList) : Scene paths of part to process

aabb (AABB) : Axis aligned bounding box to project on

uv3dSize (Distance) : 3D size of the UV space [0-1]

channel (Int) : The UV channel which will contains the texture coordinates [optional] (default: 0)

=> overideExistingUVs (Boolean) : If True, overide existing UVs on channel [optional] (default: true)

Adding optionnal parameter makeOrientable to the function orient (c.f: crackMoebiusStrip)

orient(...)

orient(scenePaths[, makeOrientable])

Orient tessellation elements

Parameters:

scenePaths (ScenePathList) : Scene paths of components to process

=> makeOrientable (Boolean) : Crack moebius strips to make the model orientable [optional] (default: true)

Deleting the experimental paramater maxNormalDistortion in the fucntion tessellate

tessellate(...)

tessellate(scenePaths, maxSag, maxLength, maxAngle[, createNormals[, uvMode[, uvChannel[, uvPadding[, createTangents[, createBinormals[, createFreeEdges]]]]]]])

Create a tessellated representation from a CAD representation for each given part

Parameters:

scenePaths (ScenePathList) : ScenePath of components to tessellate

maxSag (Distance) : Maximum distance between the geometry and the tessellation

maxLength (Distance) : Maximum length of elements

maxAngle (Angle) : Maximum angle between normals of two adjacent elements

=>

createNormals (Boolean) : If true, normals will be generated [optional] (default: true)

uvMode (UVGenerationMode) : Select the texture coordinates generation mode [optional] (default:)

uvChannel (Int) : The UV channel of the generated texture coordinates (if any) [optional] (default: 1)

uvPadding (Double) : The UV padding between UV island in UV coordinate space (between 0-1). This parameter is handled as an heuristic so it might not be respected [optional] (default: 0.0)

createTangents (Boolean) : If true, tangents will be generated [optional] (default: false)

createBinormals (Boolean) : If true, binormals will be generated [optional] (default: false)

createFreeEdges (Boolean) : If true, free edges will be created for each patch borders [optional] (default: false)

Module: CAD

Parameter « Matrix » is now optionnal in function createConeSurface

createConeSurface(...)
createConeSurface(radius, semiAngle[, matrix]) -> coneSurface
Create a new cone surface
Parameters:
radius (Distance) : Radius of the cone at origin
semiAngle (Angle) : Semi-angle of the cone
matrix (Matrix4) : Positionning matrix of the cone [optional] (default: geom.IdentityMatrix4)
Return value:
coneSurface (Surface) : The new cone surface

Parameter « Matrix » is now optionnal in function createPlaneSurface

createPlaneSurface(...)
createPlaneSurface([matrix]) -> planeSurface
Create a new plane surface
Parameters:
matrix (Matrix4) : Positionning matrix of the plane [optional] (default: geom.IdentityMatrix4)

Le parametre matrix est desormais optionnel dans la fonction createSphereSurface
createSphereSurface(...)
createSphereSurface(radius[, matrix]) -> sphereSurface
Create a new sphere surface
Parameters:
radius (Distance) : Radius of the sphere
matrix (Matrix4) : Positionning matrix of the sphere [optional] (default: geom.IdentityMatrix4)
Return value:
sphereSurface (Surface) : The new sphere surface

Parameter « Matrix » is now optionnal in function createTorusSurface

createTorusSurface(...)
createTorusSurface(radiusMax, radiusMin[, matrix]) -> torusSurface
Create a new torus surface
Parameters:
radiusMax (Distance) : Major radius
radiusMin (Distance) : Minor radius
matrix (Matrix4) : Positionning matrix of the sphere [optional] (default: geom.IdentityMatrix4)
Return value:
torusSurface (Surface) : The new torus surface

Module: Scene

Renaming of identifySubTree in resetTransform

identifySubTree(...)

identifySubTree([assembly])

Set all transformation matrices to identity in a sub-tree (Instances will be singularized)

Parameters:

assembly (Assembly) : root assembly for the process [optional] (default: 0)

resetTransform(...)

resetTransform([assembly])

Set all transformation matrices to identity in a sub-tree (Instances will be singularized)

Parameters:

assembly (Assembly) : Root assembly for the process [optional] (default: 0)

Function mergeParts is now merging all entities is a single assembly being returned by the python function mergeParts

Behavior of this function has changed, materials are now more used for merging. See function mergePartsByMaterials

mergeParts(...)

mergeParts(partPaths) -> mergedAssembly

Merge a set of parts according to their materials

Parameters:

partPaths (ScenePathList) : Scene path of the parts to merge

Return value:

=> mergedAssembly (Assembly) : Resulting assembly of the merged parts