# COLLADA 1.4 Quick Reference Card - Page 1

COLLADA™ defines an XML-based schema to allow transport of 3D assets between applications, enabling diverse 3D authoring and content processing tools to be combined into a production pipeline.

#### All elements on this card apply to the COMMON profile unless otherwise noted.

- [n] refers to chapters in COLLADA 1.4 Specification: www.khronos.org/collada
- Attributes are green. Optional Attributes are italic.
- Elements are blue. [Placeholder elements] are in brackets.
- + element expanded elsewhere on card.
- element expanded in specification.
- indicates sequence.
- indicates choice.
- xs:\* types are defined in the XML Schema language specification.
- The default cardinality is 1.

- · <any> may contain any well-formed XML data.
- Type TargetableFloat is a floating point value that has a sid attribute
- Type TargetableFloat3 is a floating point vector value that has an sid attribute.
  - Color model is RGB for float3, and RGBA for float4 values
  - Spatial coordinates are Cartesian for float (X), float2 (XY), and float3 (XYZ) values.
  - Texture coordinates are Cartesian for float (S), float2 (ST), and float3 (STP) values; and homogenous for float4 (STPQ) values.

#### The parent of all library\_\* elements is COLLADA

Declares a module of <animation> elements

library_animations	
id	xs:ID
name	xs:NCName
	[01] +
_ + animation	[1*] +
L extra	[0*] +

Declares a module of <camera> elements.

library_cameras	
id	xs:ID
name	xs:NCName
_ asset	[01] +
_ — camera	[1*] +
L extra	[0*] +

Declares a module of <controller> elements

library_controllers	
id	xs:ID
name	xs:NCName
_ asset	[01] +
_ + controller	[1*] +
L extra	[0*] +

Declares a module of <effect> elements.

library_effects	
id	xs:ID
name	xs:NCName
_ asset	[01] +
_ + effect	[1*] +
L extra	[0*] +

Declares a module of <geometry> elements.

xs:ID
xs:NCName
[01] +
[1*] +
[0*] +

Declares a module of <image> elements

Deciales a module of \minage> elements.	
library_images	
id	xs:ID
name	xs:NCName
_ asset	[01] +
_ image	[1*] +
- extra	[0*] +

Declares a module of light> elements

library_lights	
id	xs:ID
name	xs:NCName
_ asset	[01] +
_ — light	[1*] +
L extra	[0*] +

Declares a module of <material> elements

library_materials	
id	xs:ID
name	xs:NCName
_ asset	[01] +
_ — material	[1*] +
extra	[0*] +

Declares a module of <node> elements

library_nodes	
id	xs:ID
name	xs:NCName
_ asset	[01] +
_ — node	[1*] +
L extra	[0*] +

Defines unit of distance for COLLADA elements and objects.

unit	
meter	float
name	xs:NMTOKEN
D	

#### Parent@sset

#### Scene Elements [5]

Describes the entire set of information that can be visualized from the contents of a COLLADA resource.

scene		
instance_physics_scene	[0*]	+ InstanceWithExtra
_ instance_visual_scene	[01]	+ InstanceWithExtra
L extra	[0*]	+

Parent@COLLADA

Declares an environment in which physical objects are instantiated and simulated

physics_scene		
id		xs:ID
name		xs:NCName
☐ asset	[01]	+
instance_force_field	[0*]	+ InstanceWithExtra
instance_physics_model	[0*]	+
_ technique_common		
L gravity	[01]	TargetableFloat3
time_step	[01]	TargetableFloat
technique (core)	[0*]	+
∟ <sub>extra</sub>	[0*]	+
Daront dibrary physics scopes		

Parent@ibrary\_physics\_scenes

Describes the entire set of information that can be visualized from the contents of a COLLADA resource

visual_scene		
id		xs:ID
name		xs:NCName
_ asset	[01]	+
– node	[1*]	+
<ul> <li>– evaluate_scene</li> </ul>	[0*]	+
L extra	[0*]	+

Parent@ibrary\_visual\_scenes

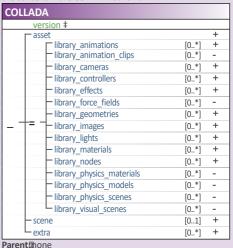
Allows the instantiation of a physics model within another physics model, or in a physics scene.

	xs:anyURI
	xs:NCName
	xs:anyURI
[0*]	+ InstanceWithExtra
[0*]	+
[0*]	-
[0*]	+
	[0*]

Parents physics scene, physics model

#### Metadata Elements [5]

Declares the root of the document that contains some of the content in the COLLADA schema



Defines asset-management information.

‡ version: 1.4.0, 1.4.1

Defines asset management informatio		
asset		
r contributor	[0*]	
author	[01]	xs:string
authoring_tool	[01]	xs:string
└_ + comments	[01]	xs:string
- copyright	[01]	xs:string
└ source_data	[01]	xs:anyURI
- created		dateTime
<ul><li>keywords</li></ul>	[01]	xs:string
_ modified		dateTime
revision	[01]	xs:string
<ul><li>subject</li></ul>	[01]	xs:string
– title	[01]	xs:string
– unit	[01]	+
up_axis ‡	[01]	-

Parents@amera, COLLADA, light, material, source, geometry, image, animation, animation\_clip, controller, extra, node, visual\_scene, library\_\*, effect, force\_field, physics\_{material, scene, model}, profile\_\*, profile\_{CG, COMMON, GLES}/technique (FX)

‡ up axis: X UP, Y UP, Z UP. Default = Y UP

#### Instantiates a COLLADA resource.

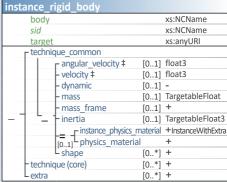
instance_animation, instance_{camera, light, instance_{visual, physics_instance_physics_materi	} scene.
instance_physics_materi instance_force_field	InstanceWithExtra
url	xs:anyURI
sid	xs:NCName
name	xs:NCName
— extra	[0*] +

#### Parents 2

instance\_animation: animation\_clip instance {camera, light, node}: node; instance {visual, physics} scene: scene; instance physics material: {instance} rigid\_body, shape;

instance\_force\_field: physics\_scene, instance\_physics\_model

Instantiates < rigid\_body> within an <instance\_physics\_model>.



Parent@nstance\_physics\_model ‡ angular\_velocity, velocity: Default = 0 0 0

Scene Elements Continued >

#### Scene Elements (continued)

Declares instantiation of a COLLADA < geometry> resource.

instance_geometry		
bind_material	[01] +	
− ∟ <sub>extra</sub>	[0*] +	
Parents@node, shape		

Binds a specific material to a piece of geometry, binding varying and uniform parameters at the same time.

bind_material		
┌ param (core)	[0*]	
name		xs:NCName
sid -		xs:NCName
- semantic		xs:NMTOKEN
L type		xs:NMTOKEN
technique_common		
instance_material	[1*]	
- symbol		xs:NCName
target		xs:anyURI
sid		xs:NCName
name		xs:NCName
bind (material)	[0*]	-
L_ + bind_vertex_input	[0*]	-
L extra	[0*]	+
technique (core)	[0*]	+
L extra	[0*]	+

Parents@nstance\_geometry, instance\_controller

Declares instantiation of a COLLADA <controller> resource.

instance_controller		
— skeleton	[0*]	xs:anyURI
_ bind_material	[01]	+
– extra	[0*]	+
Parent@hode		

Describes an afternative way to ev	araate a	VISUAL_SCETICE.
evaluate_scene		
name		xs:NCName
render	[1*]	
camera_node		xs:anyURI
layerlayer	[0*]	xs:NCName
instance_effect	[01]	+

Parent@visual scene

Describes hierarchical relationship of elements in a scene.

node	
id	xs:ID
name	xs:NCName
sid	xs:NCName
type ‡	NodeType
layer	ListOfNames
☐ asset	[01] +
□lookat	[0*] +
matrix	[0*] +
rotate	[0*] +
_ <del>  =   scale</del>	[0*] +
skew	[0*] -
└─ translate	[0*] +
<ul><li>instance_camera</li></ul>	[0*] + InstanceWithExtra
<ul><li>instance_controller</li></ul>	[0*] +
<ul><li>instance_geometry</li></ul>	[0*] +
instance_light	[0*] + InstanceWithExtra
instance_node	[0*] + InstanceWithExtra
node -	[0*] +
∟ <sub>extra</sub>	[0*] +

Parents@ibrary\_nodes, node, visual\_scene ‡ type: JOINT, NODE. Default = NODE

## **Animation Elements [5]**

Declares interpolation sampling function for an animation.



Parent@nimation

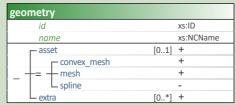
‡ semantic: see note for input (shared) on page 3

Declares an output channel of an animation

channel	
source	xs:URIFragmentType
target	xs:token
Parent@nimation	

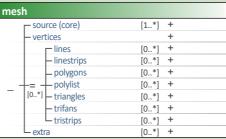
#### **Geometry Elements [5]**

Describes visual shape and appearance of object in scene.



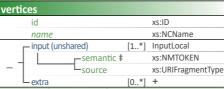
Parent@ibrary\_geometries

Describes basic geometric meshes using vertex and primitive information.



**Parent**@eometry

Declares the attributes and identity of mesh vertices.



Parents@mesh.convex mesh

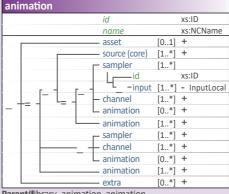
‡ semantic: see note for input (shared) on page 3

Declares the binding of geometric primitives and vertex attributes for a mesh element to produce individual triangles.

triangles	
name	xs:NCName
count	uint
material	xs:NCName
input (shared)	[0*] + InputLocalOffset
_ +p	[01] ListOfUInts
extra	[0*] +

Parents@mesh. convex mesh

Declares animation information



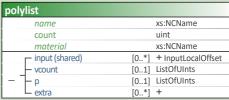
Parent@ibrary animation, animation

Describes a section of the animation curves to be used together as an animation clip.

animation_clip	
id	xs:ID
name	xs:NCName
start ‡	xs:double
end	xs:double
r asset	[01] +
<ul><li>instance_animation</li></ul>	[1*] + InstanceWithExtra
url	xs:anyURI
	[0*] +
L extra	[0*] +
Parent@ihrany animation cline	t start: Default - 0.0

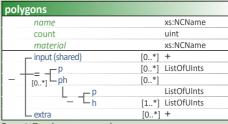
Parent@ibrary\_animation\_clips # start: Default = 0.0

Declares the binding of geometric primitives and vertex attributes for a mesh element to produce polylists.



Parent@mesh

Declares the binding of geometric primitives and vertex attributes for a mesh element to produce polygons.



Parents@mesh, convex mesh

Declares the binding of geometric primitives and vertex attributes for a mesh element to produce lines.

lines		
name	xs:NCN	ame
count	uint	
material	xs:NCN	ame
input (shared)	[0*] + Inpu	tLocalOffset
_ + p	[01] ListOfU	lints
L extra	[0*] +	

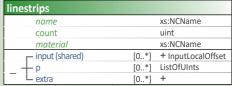
Parents@mesh, convex\_mesh

Declares the binding of geometric primitives and vertex attributes for a mesh element to produce connected triangles.

trifans, tristrips		
name		xs:NCName
count		uint
material		xs:NCName
input (shared)	[0*]	+ InputLocalOffset
<b>—</b> p	[0*]	ListOfUInts
extra	[0*]	+

Parents@mesh, convex mesh

Declares the binding of geometric primitives and vertex attributes for a mesh element to produce linestrips.



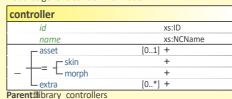
Parents@mesh, convex\_mesh

# Using to represent assembly of mesh primitive

The first index in a element refers to all inputs with an offset attribute value of 0. The second index refers to all inputs with an offset of 1. There is an index value for each unique input offset attribute value. Each vertex of the primitive is assembled using the value(s) read from indexed inputs. After each input is sampled, producing a primitive vertex, the next index in the element again refers to the inputs with offset of 0.

#### Controller Elements [5]

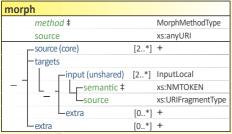
Declares generic control information



Controller Elements Continued >

# **Controller Elements (continued)**

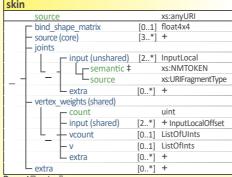
Describes the data required to blend between sets of static meshes.



Parent@controller

‡ method: NORMALIZED, RELATIVE. Default = NORMALIZED
semantic: see note for input (shared)

Declares vertex and primitive information sufficient to describe blend-weight skinning.

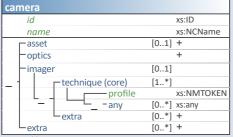


Parent@controller

‡ semantic: see note for input (shared)

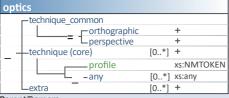
#### Camera Elements [5]

Declares a view into scene hierarchy or graph. Contains elements that describe the camera's optics and imager.



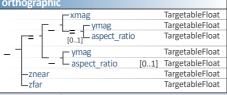
Parentdibrary\_cameras

Describes the apparatus on a camera that projects the image onto the image sensor.



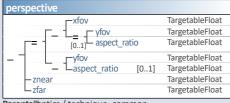
Parent@tamera

Describes the field of view of an orthographic camera.



Parents@ptics / technique common

Describes the field of view of a perspective camera. <xfov> and <yfov> values are in Euler degrees.



Parents apptics / technique\_common

#### **Extensibility Element [5]**

Declares information used to describe some portion of the content. Each technique applies to an associated profile.

technique (core)	
profile	xs:NMTOKEN
— any	[0*] xs:any

Parents Eextra, source (core), light, optics, imager, force\_field, physics\_material, physics\_scene, rigid\_body, rigid\_constraint, instance\_rigid\_body, bind\_material

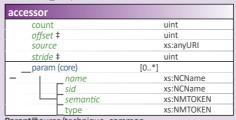
# Data Flow Elements [5]

Declares a data repository that provides values according to the semantics of an <input> element that refers to it.

source (core)	
id	xs:ID
name	xs:NCName
— asset	[01] +
☐ IDREF_array	[01] +
<ul><li>Name_array</li></ul>	[01] +
_ <del>  =   bool_array</del>	[01] +
[01] — float_array	[01] +
└ int_array	[01] +
tęchnique_common	[01]
	+
technique (core)	[0*] +

Parents@morph, animation, mesh, convex\_mesh, skin, spline

Declares an access pattern to one of the array elements: <float\_array>, <int\_array>, <Name\_array>, <bool\_array>, and <IDREF\_array>.



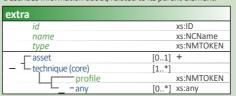
Parentsource/technique\_common ‡ Defaults: offset = 0, stride = 1

Declares storage for a homogenous array. <bool\_array> uses type ListOfBools, an xs:list of type xs:boolean. <\text{Name\_array} uses type ListOfNames, an xs:list of type xs:Name.

bool_array, Name_array	
id	xs:ID
name	xs:NCName
count	uint

Parentsource (core)

Describes information about/related to its parent element.



Parents@nimation, animation\_clip, attachment, box, camera, bind\_material, capsule, COLLADA, controller, cylinder, control\_vertices, convex\_mesh, effect, force\_field, format\_hint, geometry, image, imager, instance\_\*, joints, library\_\*, light, lines, linestrips, material, mesh, morph, node, optics, pass, plane, physics\_material, physics\_model, physics\_scene, polygons, polylist, profile\_CG, profile\_COMMON, profile\_GLES, profile\_GLSL, ref\_attachment, rigid\_body, rigid\_constraint, sampler\_\*, scene, shape, skin, sphere, spline, surface, targets, tapered\_capsule, tapered\_cylinder, triangles, trifans, tristrips, texture\_pipeline, texture\_unit, vertex\_weights, vertices, visual\_scene, and technique (FX) (in profile\_CG, profile\_COMMON, profile\_GLES, and profile\_GLSL)

#### Transform Elements [5]

Declare local coordinate system transformations.

<rotate> specifies an axis (XYZ) and rotation (Euler angle) about it as a float4.

<translate> specifies a translation (XYZ) as a float3.

# rotate, translate sid xs:NCName

Parents@hode, instance\_rigid\_body, {ref\_}attachment, shape, technique\_common/mass\_frame in rigid\_body

<scale> specifies a change in proportions (XYZ) of the axes as a float3.

<lookat> describes a position/orientation transformation as a float3x3, organized as three vectors in order: eye position, interest point, up-axis direction.

<matrix> describes a homogeneous transformation as a float4x4, organized in column-major order.



Declares the storage for a homogenous array of ID reference values of type xs:IDREFS.

IDREF_array	
id	xs:ID
name	xs:NCName
count	uint

Parentsource (core)

Declares the storage for a homogenous array of type ListOfInts, which is an xs:list of type xs:long.

int_array	
id	xs:ID
name	xs:NCName
count	uint
minInclusive ‡	xs:integer
maxInclusive ‡	xs:integer

Parentsource (core)

‡ Defaults: minInclusive = -2147483648, maxInclusive = 2147483647

Declares the storage for a homogenous array of type ListOfFloats, which is an xs:list of type xs:double.

float_array	
id	xs:ID
name	xs:NCName
count	uint
digits ‡	xs:short
magnitude ‡	xs:short

Parentsource (core)

‡ Defaults: digits = 6, magnitude = 38

Declares the input semantics of a data source and connects a consumer to that source.

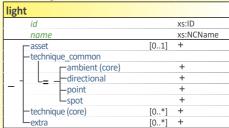
input (shared)	InputLocalOffset		
offset	uint		
semantic ‡	xs:NMTOKEN		
source	xs:URIFragmentType		
set	uint		

Parents dines, linestrips, polygons, polylist, triangles, trifans, tristrips, vertex\_weights

‡ semantic: The common semantic attribute values are: {TEX}BINORMAL, CONTINUITY, IMAGE, INPUT, WEIGHT, INTERPOLATION, INV\_BIND\_MATRIX, UV, VERTEX, JOINT, LINEAR\_STEPS, NORMAL, OUTPUT, TEXCOORD, POSITION, MORPH {TARGET, WEIGHT}, {TEX}TANGENT, {IN, OUT}\_TANGENT

## Lighting Elements [5]

Declares a light source that illuminates a scene.



Parent@ibrary\_lights

Lighting Elements Continued >

# Lighting Elements (continued)

Describes an ambient light source.

ambient (core), directional		
<ul><li>color</li></ul>	TargetableFloat3	
_ L sid	xs:NCName	

Parentdight/technique\_common

Describes a spot light source.

spot		
_ color		TargetableFloat3
<ul><li>constant_attenuation</li></ul>	[01]	TargetableFloat
<ul> <li>linear_attenuation</li> </ul>	[01]	TargetableFloat
<ul> <li>quadratic_attenuation</li> </ul>	[01]	TargetableFloat
<ul><li>falloff_angle</li></ul>	[01]	TargetableFloat
falloff_exponent	[01]	TargetableFloat

Parent dight/technique\_common

Describes a point light source.

point		
		TargetableFloat3
— constant_attenuation ‡	[01]	TargetableFloat
linear_attenuation ‡	[01]	TargetableFloat
uadratic attenuation ‡	[01]	TargetableFloat
Parent@ight/technique_common		

‡ Defaults: constant\_attenuation = 1.0, linear\_attenuation = 0.0, quadratic\_attenuation = 0.0

# **Physics Material Element [6]**

Describes the physical properties of an object.

physics_material	
id	xs:ID
name	xs:NCName
r asset	[01] +
technique common	
dynamic_friction #	[01] TargetableFloat
restitution ‡	[01] TargetableFloat
static_friction ‡	[01] TargetableFloat
technique (core)	[0*] +
- extra	[0*] +

Parentsdibrary\_physics\_materials, shape, {instance\_}rigid\_body/technique\_common

# {dynamic, static}\_friction, restitution: Default = 0

# FX2Rendering Elements (COMMON) [8]

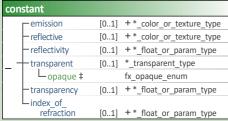
Describes a specularly shaded surface where the specular reflection is shaded according to the Blinn BRDF approximation. In the diagram, \* = common.

blinn, phong		
emission	[01]	+ *_color_or_texture_type
- ambient (FX)	[01]	+ *_color_or_texture_type
- diffuse	[01]	+ *_color_or_texture_type
specular	[01]	+ *_color_or_texture_type
_ shininess	[01]	+ *_float_or_param_type
- reflective	[01]	+ *_color_or_texture_type
reflectivity	[01]	+ *_float_or_param_type
transparent	[01]	*_transparent_type
Lopaque ‡		fx_opaque_enum
- transparency	[01]	+ *_float_or_param_type
index_of_ refraction	[01]	+ *_float_or_param_type

Parents dechnique (FX) in profile\_COMMON

‡ opaque: A\_ONE, RGB\_ZERO. Default = A\_ONE

Describes a constantly shaded surface that is independent of lighting. In the diagram, \* = common



Parent dechnique (FX) in profile\_COMMON ‡ opaque: A ONE, RGB ZERO. Default = A ONE

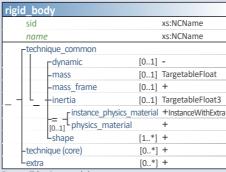
#### Physics Model Elements [6]

Allows for building complex combinations of rigid bodies and constraints that may be instantiated multiple times.

physics_model		
id		xs:ID
name		xs:NCName
_ asset	[01]	+
rigid_body	[0*]	+
rigid_constraint	[0*]	+
instance_physics_model	[0*]	+
Lextra	[0*]	+

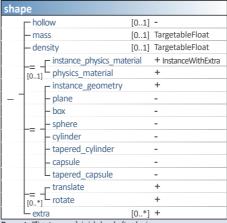
Parent@ibrary\_physics\_models

Describes simulated bodies that do not deform.



Parent@physics model

Describes components of a < rigid body>



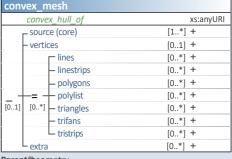
Parents@instance\_}rigid\_body/technique\_common

Defines the center and orientation of the rigid body.



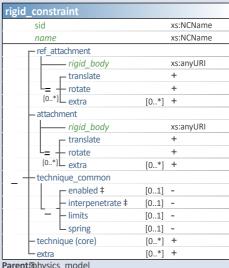
Parent@rigid\_body/technique\_common

Contains or refers to information that describes basic geometric meshes



**Parent** regeometry

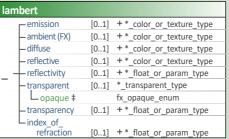
Connects components, such as <rigid\_body>, into complex physics models with moveable parts.



Parent@physics\_model

‡ Defaults: enabled = True, interpenetrate = False

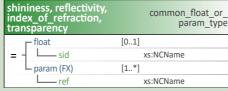
Describes a diffuse shaded surface that is independent of lighting. In the diagram, \* = common



Parent@echnique (FX) in profile\_COMMON

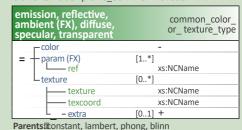
‡ opaque: A\_ONE, RGB\_ZERO. Default = A\_ONE

Describes scalar attributes of fixed-function shader elements inside cpmofile COMMON> effects.



Parents tonstant, lambert, phong, blinn

Describes color attributes of fixed-function shader elements inside <profile COMMON> effects.



<newparam> creates a new, named param object in the FX Runtime, and assigns it a type, an initial value, and additional attributes at declaration time.

newparam		comr	non_newparam_type
sid	-		xs:NCName
- semantic		[01]	xs:NCName
_ floa	at		float
- I - floa	at2		float2
	at3		float3
- floa	at4		float4
– sur	face (FX)		+ fx_surface_common
L san	npler2D		+ fx_sampler2D_common
L san		. /	

Parents@profile\_COMMON/technique (FX)

# FX2Texturing Elements (All Profiles) [8]

Declares the storage for the graphical representation of

image		
id		xs:ID
name		xs:NCName
format		xs:token
height		uint
width		uint
depth		uint
_ asset	[01]	+
data		ListOfHexBinary
- T- Linit_from		xs:anyURI
L extra	[0*]	+

Profiles@COMMON, CG, GLSL, GLES

Parents@library\_images, effect, profile\_CG, profile\_GLSL, profile\_COMMON, profile\_GLES; technique (FX) in profile\_CG, profile\_COMMON, profile\_GLES, profile\_GLSL

Declares a two-dimensional texture sampler.

sampler2D	fx_sampler2D_common gl_sampler_2d
_source	xs:NCName
_wrap_s ‡	[01] fx_sampler_wrap_common
_wrap_t ‡	[01] fx_sampler_wrap_common
_minfilter ‡	[01] fx_sampler_filter_common
-magfilter ‡	[01] fx_sampler_filter_common
_mipfilter ‡	[01] fx_sampler_filter_common
_border_color	[01] - fx_color_common
_mipmap_maxlevel ‡	[01] xs:unsignedByte
_mipmap_bias ‡	[01] float

Profiles COMMON, CG, GLSL, External, Effect

Parents@newparam, setparam, usertype, array, shader/bind

‡ wrap\_s, wrap\_t: NONE, WRAP, MIRROR, CLAMP, BORDER. Default = WRAP

minfilter, magfilter, mipfilter: NONE, NEAREST, LINEAR, {NEAREST, LINEAR}\_MIPMAP\_NEAREST {NEAREST, LINEAR}\_MIPMAP\_LINEAR, Default = NONE

Defaults: mipmap\_maxlevel = 255, mipmap\_bias = 0

Declares a resource that can be used both as the source for texture samples and as the target of a rendering pass. Child elements differ depending on the profile used in the diagram. \* = common

in the diagram, *= commo	''
surface	fx_surface_common
type ‡	fx_surface_type_enum
rinit_as_null	xs:anyType
-init_as_target	xs:anyType
init_cube	<ul><li>fx_surface_init_cube_*</li></ul>
[01] -init_volume	<ul><li>fx_surface_init_volume_*</li></ul>
-init_planar	<ul><li>fx_surface_init_planar_*</li></ul>
Linit_from	[1*] fx_surface_init_from_*
_mip ‡	xs:unsignedInt
slice ‡	xs:unsignedInt
	fx_surface_face_enum
- format	[01] xs:token
- format_hint	[01] - fx_surface_format_hint_*
_size ‡	int3
[01] viewport_ratio ‡	float2
- mip_levels ‡	[01] xs:unsignedInt
- mipmap_generate ‡	[01] xs:boolean
- extra	[0*] +

Profiles COMMON, CG, GLES, GLSL, External, Effect

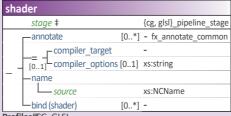
Parents COMMON - newparam, setparam; CG - newparam, setparam, array, shader/bind, usertype; GLES - newparam, setparam, texture\_unit; GLSL - newparam, setparam, array, shader/bind

‡ type: UNTYPED, 1D, 2D, 3D, RECT, CUBE, DEPTH init\_from/face: POSITIVE\_{X,Y,Z}, NEGATIVE\_{X,Y,Z}. Default = POSITIVE\_X

Defaults: size = 0 0 0, viewport\_ratio = 1 1 mip\_levels = 0, mipmap\_generate = False, init\_from/mip, init\_from/slice = 0

#### FXShader Elements (Other Profiles) [8]

Declares and prepares a shader for execution in the rendering pipeline of a <pass> element



Profiles CG. GLSL

Parent@profile\_{CG,GLSL}/technique/pass,

‡ stage: CG: VERTEX, FRAGMENT

GLSL: VERTEXPROGRAM, FRAGMENTPROGRAM

Declares all the render states, shaders, and settings for one rendering pipeline.

pass	
sid	xs:NCName
annotate	[0*] - fx_annotate_common
-color_target	[01] - fx_colortarget_common
-depth_target	$[01] \ \hbox{-} \ \ fx\_depthtarget\_common$
-stencil_target	[01] - fx_stenciltarget_common
_ color_clear	[01] - fx_clearcolor_common
-depth_clear	[01] - fx_cleardepth_common
-stencil_clear	[01] - fx_clearstencil_common
-draw	[01] - fx_draw_common
render_states]	† -
[1*] - shader	+
Lextra	[0*] +

Profiles@CG, GLES, GLSL

Parents®profile\_CG/technique (FX) and profile\_GLSL/technique (FX). <pass> is also a child of profile\_GLES/technique (FX), in which case it excludes the child element <shader>

† [render\_states]: Refer to the Render States subsection in the description of <pass> in the specification. The schema indicates use of group gl\_pipeline\_settings for profiles GLSL or CG, and gles\_pipeline\_settings for GLES. <newparam> creates a new, named param object in the FX Runtime, and assigns it a type, an initial value, and additional attributes at declaration time.

newparam	fx_newparam_common gles_newparam
sid	xs:NCName
- annotate	[0*] - fx_annotate_common
_ + semantic	[01] xs:NCName
– modifier ‡	[01] fx_modifier_enum_common
L= [values] †	- fx_basic_type_common

Profile@Effect, GLES

Parent@For fx\_newparam\_common: effect; For gles\_newparam: profile\_GLES, profile\_GLES/technique

- ‡ modifier: CONST, UNIFORM, VARYING, STATIC, VOLATILE, EXTERN, SHARED
- † [values]: Includes elements from the following list, where n is 1, 2, 3, or 4: bool, booln, int, intn, float, floatn, floatnxm, surface (FX), and enum.

For fx newparam common the list includes sampler{1D, 2D, 3D, CUBE, RECT, DEPTH}.

For gles\_newparam the list includes sampler\_state and texture\_{pipeline, unit}.

nev	vparam	{glsl, cg}_newparam
	sid	{glsl, cg}_identifier
	annotate	[0*] - fx_annotate_common
-	_semantic	[01] xs:NCName
1	-modifier ‡	[01] fx_modifier_enum_common
	r= -[values]	] + - {glsl, cg}_param_type
	L= +array	<ul> <li>{glsl, cg}_newarray_type</li> </ul>
	Lusertype	<ul><li>cg_setuser_type</li></ul>

Profile@CG. GLSL

Parents@profile\_{GLSL, CG}, profile\_{GLSL, CG}/technique (FX)

Child <usertype> excluded from glsl newparam

- ‡ modifier: CONST, UNIFORM, VARYING, STATIC, VOLATILE, EXTERN. SHARED
- † [values]: Includes elements from the following list, where n is 1, 2, 3, or 4: bool, booln, int, intn, float, floatn, string, sampler{1D, 2D, 3D, CUBE, RECT, DEPTH}, and enum.

For glsl\_newparam the list includes float2x2, float3x3, float4x4, and surface (GLSL),

For cg\_newparam the list includes boolnxm, intnxm, half, halfn, halfnxm, fixed, fixedn, fixednxm, floatnxm, and surface.

# FX: Texturing Elements (Other Profiles) [8]

Declares a two-dimensional texture sampler state for element <profile\_GLES>

sampler_state	gles_sampler_state
sid	xs:NCName
_wrap_s ‡	[01] gles_sampler_wrap
_ wrap_t ‡	[01] gles_sampler_wrap
_ minfilter ‡	[01] fx_sampler_filter_common
_ magfilter ‡	[01] fx_sampler_filter_common
mipfilter ‡	[01] fx_sampler_filter_common
_ mipmap_maxlevel ‡	[01] xs:unsignedByte
_ mipmap_bias ‡	[01] float
L extra	[0*] +
Profile@GLES	

Parents newparam, setparam

wrap\_s, wrap\_t: REPEAT, CLAMP, CLAMP\_TO\_EDGE, MIRRORED\_REPEAT. Default = REPEAT

minfilter, magfilter, mipfilter: NONE, NEAREST, LINEAR, {NEAREST, LINEAR}\_MIPMAP\_NEAREST, {NEAREST, LINEAR}\_MIPMAP\_LINEAR, Default = NONE

Default: mipmap\_maxlevel = 255, mipmap\_bias = 0

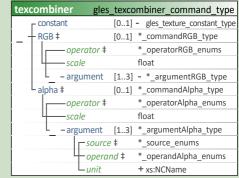
Defines a texture unit that will be mapped to hardware texture units based on its usage in <texture\_pipeline>

texture_unit	gles_texture_unit
sid	xs:NCName
surface	[01] xs:NCName
sampler_state	[01] xs:NCName
[1*] texcoord	[01]
semantic	xs:NCName

**Profile** GLES

Parents Betparam, newparam

Defines a <texture\_pipeline> command for combiner-mode texturing. In the diagram, \* = gles\_texcombiner.



Profile GLES

Parents@newparam/texture\_pipeline, setparam/texture\_pipeline, pass/texture\_pipeline/value

‡ RGB, RGB/operator: REPLACE, MODULATE, ADD, ADD SIGNED, INTERPOLATE, SUBTRACT, DOT3\_{RGB, RGBA}

alpha, alpha/operator: REPLACE, MODULATE, ADD, ADD\_SIGNED, INTERPOLATE, SUBTRACT

alpha/argument/source: TEXTURE, CONSTANT, PRIMARY,

alpha/argument/operand: ONE\_MINUS\_SRC\_ALPHA, SRC\_ALPHA. Default = SRC\_ALPHA

Texturing Elements (Other Profiles) Continued >

#### Texturing Elements (Other Profiles) (cont'd)

Declares a resource that can be used both as the source for texture samples and as the target of a rendering pass. This element inherits the elements from <surface> (FX) and adds the following:

surface	cg_surface_type glsl_surface_type
type ‡	fx_surface_type_enum
— generator	[01]
annotate  annotate  code include name	[0*] - fx_annotate_common - fx_code_profile - fx_include_common +
L setparam ‡	[0*]

#### Profile CG, GLSL, GLES

Parents: COMMON - newparam, setparam; CG - newparam, setparam, array, shader/bind, usertype; GLES - newparam, setparam, texture unit; GLSL - newparam, setparam, array, shader/bind

‡ type: UNTYPED, 1D, 2D, 3D, CUBE, DEPTH, RECT setparam: for surface (CG), type is cg\_setparam\_simple, for surface (GLSL), type is glsl\_setparam\_simple

Defines a set of texturing commands that will be converted into multitexturing operations using glTexEnv in regular and combiner mode.

texture_pipeline, texture_pipeline/v	/alue gles_texture_pipeline
sid	xs:NCName
_ texcombiner	+ gles_texcombiner_command_type
= texenv	+ gles_texenv_command_type
[1*] ∟ extra	+
Profile@GLES	

Parents@newparam, setparam, pass/render\_state

Defines a texture\_pipeline command for simple, noncombiner-mode texturing.

texenv	gles_texenv_command_type	
operator ‡	gles_texenv_mode_enums	
unit	+ xs:NCName	
constant (combine	er)[01] - gles_texture_constant_type	

# Profile@GLES

Parents@newparam/texture\_pipeline, setparam/texture\_pipeline, pass/texture\_pipeline/value ‡ operator: REPLACE, MODULATE, DECAL, BLEND, ADD

#### FX@Materials Elements [8]

Describes the visual appearance of a geometric object.

material	
id	xs:ID
name	xs:NCName
rasset	[01] +
_ instance_effect	+
extra	[0*] +

Parent@ibrary\_material

Instantiates a COLLADA effect.

inst	ance_effect		
	url		xs:anyURI
	sid		xs:NCName
	name		xs:NCName
	rechnique_hint	[01]	
	<ul><li>platform</li></ul>		xs:NCName
	– profile		xs:NCName
	⊢ref		xs:NCName
	– setparam	[0*]	-
	L extra	[0*]	+

Parents@material, render

#### FXÆffects Elements [8]

Declares a self-contained description of a COLLADA effect.

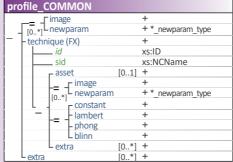
effect	
id	xs:ID
name	xs:NCName
r asset	[01] +
– annotate	[0*] - fx_annotate_common
_ image	[0*] +
— newparam	[0*] + fx_newparam_common
_ [fx_profile_abstract] ‡	[1*] xs:anyType
∟ extra	[0*] +

#### **Profile** Effect

#### Parent dibrary\_effects

‡ [fx\_profile\_abstract]: Exactly one of profile\_{CG, GLES, GLSL,COMMON}

Opens a block of platform-independent declarations for the common, fixed-function shader. \* = common



Profile@COMMON
Parent@effect

Declares platform-specific data types and techniques for the GLES language.

profile_GLES				
	id		xs:ID	
	platform		xs:NCName	
	asset	[01]	+	
_	image		+	
-	[0*] newparam		+ gles_newparam	
	technique (FX)	[1*]	+	
	Lextra	[0*]	+	

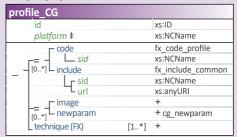
Profile@GLES
Parent@effect



Visit www.collada.org for more on COLLADA, including a forum, a model bank, directories of extensions and conditioners, and more.

Get your copy of COLLADA: Sailing the Gulf of 3d Digital Content Creation from your technical bookstore or www.amazon.com.

Declares platform-specific data types and techniques for the Cg language.

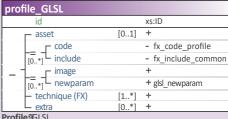


# Profile CG

#### Parent@effect

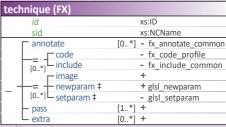
‡ platform: Default = "PC"

Declares platform-specific data types and techniques for the GLSL language.



Profile GLSL
Parent Deffect

Declares information to process content. Each technique applies to an associated profile. Child elements differ depending on parent. Refer to parent descriptions for list of children.



Profiles@GLSL, CG, GLES

Parents@profile\_GLSL, profile\_CG, profile\_GLES

- ‡ The type for child elements <newparam> and <setparam> differ depending on parent of <technique> (FX), as follows:
  - profile\_GLSL/technique (FX): types are glsl\_\*
  - profile\_CG/technique (FX): types are cg\_\*
  - profile\_GLES/technique (FX): types are gles\_\*

#### **Extending COLLADA**

COLLADA allows you to extend its data model and add functionality to your documents. These extensions take the form of alternative <technique>, additive <extra>, and scalable <input> elements. For more information and a list of published extensions, see https://collada.org/mediawiki/index.php/Portal:Extensions\_directory.

#### <technique> profiles

Declares alternative techniques to <technique\_common> that provide a better description for a specific profile.

#### <extra> types

Declares new techniques that add descriptions to existing ones. This extra information can represent additional real data or semantic (meta) data to the application.

#### <input> semantics

Declares new streams that add to data flows

# Example <extra type="MY\_TYPE"> <technique profile="PROFILE-A"> < ... </technique> <technique profile="PROFILE-B"> < ... </technique> </technique> </technique> </technique> </technique>

## COLLADA Certification Logos Baseline, Superior, and Exemplary

Certification logos are earned by a COLLADA product that has passed conformance testing and indicate the degree of feature support you can expect from the software. For more information about conformance, please visit http://www.khronos.org/collada/adopters/









The Khronos Group is an industry consortium creating open standards for the authoring and acceleration of parallel computing, graphics and dynamic media on a wide variety of platforms and devices. See www.khronos.org to learn more about the Khronos Group.

COLLADA is a trademark of Khronos Group Inc.