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

Declares a module of <animation> elements.

library	_animations	
	id	xs:ID
	name	xs:NCName
Г	asset	[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]	\oplus
	controller	[1*]	\pm
_	extra	[0*]	H

Declares a module of <effect> elements.

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

Declares a module of <geometry> elements.

library_geometries	
id	xs:ID
name	xs:NCName
_ asset	[01] ⊞
	[1*] ⊞
∟ _{extra}	[0*] ⊞

Declares a module of <image> elements.

libra	ry_images		
	id	xs:ID	
	name	xs:NCName	
-	– asset	[01] ⊞	
+	– image	[1*] ⊞	
	– extra	[0*] ⊞	

Declares a module of light> elements.

		8	
library	_lights		
	id		xs:ID
	name		xs:NCName
	asset	[01]	⊞
+	light	[1*]	\pm
LL	extra	[0*]	Œ

Declares a module of <material> elements.

libra	ry_materials	
	id	xs:ID
name		xs:NCName
	- asset	[01] ⊞
	— material	[1*] ⊞
	L extra	[0*] ⊞

Declares a module of <node> elements

librar	y_nodes		
	id		xs:ID
	name		xs:NCName
	_ asset	[01]	±
	– node	[1*]	\oplus
	- extra	[0*]	Œ

Defines unit of distance for COLLADA elements and objects.

unit	
meter	float
name	xs:NMTOKEN
Parent: asset	

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*]	
instance_visual_scene	[01]	$\scriptstyle \boxplus$ InstanceWithExtra
└ extra	[0*]	⊞
Parent: COLLADA		

Declares an environment in which physical objects are instantiated and simulated.

physics_scene		
id		xs:ID
name		xs:NCName
r asset	[01]	±
instance_force_field	[0*]	
instance_physics_model	[0*]	±
gravity	[01]	TargetableFloat3
time_step	[01]	TargetableFloat
technique (core)	[0*]	±
∟ extra	[0*]	±

Parent: library_physics_scenes

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

visual_s	cene		
ia			xs:ID
n	ате		xs:NCName
r as	set	[01]	\pm
	ode	[1*]	\pm
- e	aluate_scene	[0*]	\pm
	tra	[0*]	±

Parent: library_visual_scenes

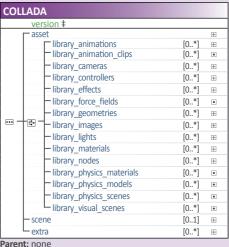
Allows the instantiation of a physics model within another physics model, or in a physics scene.

instance_physics_model		
url		xs:anyURI
sid		xs:NCName
parent		xs:anyURI
instance_force_field	[0*]	$\boxplus \ InstanceWithExtra$
instance_rigid_body	[0*]	\oplus
instance_rigid_constraint	[0*]	•
L extra	[0*]	⊞

Parents: physics_scene, physics_model

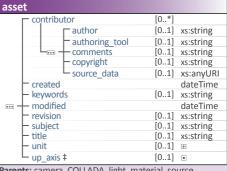
Metadata Elements [5]

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



‡ version: 1.4.0, 1.4.1

Defines asset-management information.



Parents: camera, COLLADA, light, material, source, geometry, image, animation, animation_clip, controller, extra, node, visual_scene, library_*, effect, force_field, physics_fmaterial, scene, model}, profile_*, profile_{CG, COMMON, GLES}/technique (FX)

‡ up_axis: X_UP, Y_UP, Z_UP. Default = Y_UP

Instantiates a COLLADA resource.

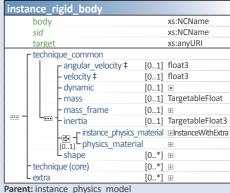
instance_animation, instance_{camera, light, instance_{visual, physics instance_physics_mater	s} scene.
instance_force_field	InstanceWithExtra
url	xs:anyURI
sid	xs:NCName
name	xs:NCName
— extra	[0*] ⊞

Parents:

instance_animation: animation_clip; instance_{camera, light, node}: node; instance_{visual, physics}_scene: scene; instance_physics_material: {instance}_rigid_body, shape;

instance_physics_material: {instance}_rigid_body, shape; instance_force_field: physics_scene, instance_physics_model

Instantiates < rigid_body> within an <instance_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] ⊞	
extra	[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 (cor	re)	[0*]	
name			xs:NCName
- sid			xs:NCName
- seman	tic		xs:NMTOKEN
L type			xs:NMTOKEN
technique_	common		
instanc	e_material	[1*]	
	- symbol		xs:NCName
	- target		xs:anyURI
	- sid		xs:NCName
	- name		xs:NCName
	bind (material)	[0*]	•
│	bind_vertex_input	[0*]	•
	- extra	[0*]	\pm
- technique	(core)	[0*]	\pm
L _{extra}		[0*]	⊞

Parents: instance_geometry, instance_controller

Declares instantiation of a COLLADA <controller> resource.

instance_controller		
□ skeleton	[0*]	xs:anyURI
	[01]	±
L extra	[0*]	\pm
Parent: node		

Describes an alternative way to evaluate a <visual scene>

evaluate_scene		
name		xs:NCName
- render - render	[1*]	
camera_node		xs:anyURI
layer	[0*]	xs:NCName
instance_effect	[01]	Ħ

Parent: visual scene

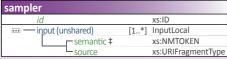
Describes hierarchical relationship of elements in a scene.

node		
id		xs:ID
name		xs:NCName
sid		xs:NCName
type ‡		NodeType
layer		ListOfNames
asset	[01]	H
□lookat	[0*]	⊞
matrix	[0*]	\pm
rotate	[0*]	\pm
scale	[0*]	\pm
[0*] -skew	[0*]	0
∟translate	[0*]	⊞
instance_camera	[0*]	
instance_controller	[0*]	\boxplus
instance_geometry	[0*]	\boxplus
instance_light	[0*]	
instance_node	[0*]	
node	[0*]	H
extra	[0*]	⊞

Parents: library nodes, node, visual scene ‡ type: JOINT, NODE. Default = NODE

Animation Elements [5]

Declares interpolation sampling function for an animation.



Parent: animation

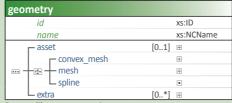
‡ semantic: see note for input (shared) on page 3

Declares an output channel of an animation

channel	Tor arranmation.
source	xs:URIFragmentType
target	xs:token
Parent: animation	

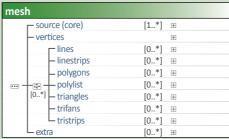
Geometry Elements [5]

Describes visual shape and appearance of object in scene.



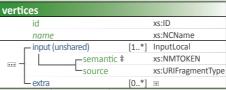
Parent: library_geometries

Describes basic geometric meshes using vertex and primitive information.



Parent: geometry

Declares the attributes and identity of mesh vertices.



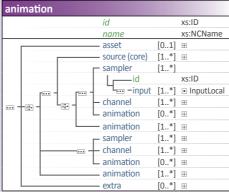
Parents: mesh, convex mesh ‡ semantic: see note for input (shared) on page 3

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: library_animation, animation

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

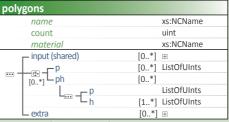
animation_clip	
id	xs:ID
name	xs:NCName
start ‡	xs:double
end	xs:double
r asset	[01] ⊞
instance_animation	[1*] ⊞ InstanceWithExtra
url	xs:anyURI
extra	[0*] ⊞
L extra	[0*] ⊞
Parantelibrane animation cline	t start: Default - 0.0

Parent: library animation clips ‡ start: Default = 0.0

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



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:NCName
count	uint
material	xs:NCName
input (shared)	[0*] ⊞ InputLocalOffset
	[01] ListOfUInts
L extra	[0*] ⊞

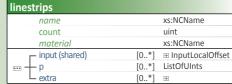
Parents: mesh, convex_mesh

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

triangics.		
trifans, tristrips		
name		xs:NCName
count		uint
material		xs:NCName
input (shared)	[0*]	
+ 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.



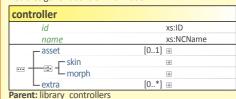
Parents: mesh, convex_mesh

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]

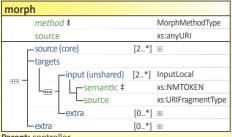
Declares generic control information.



Controller Elements Continued >

Controller Elements (continued)

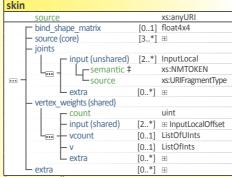
Describes the data required to blend between sets of static meshes



Parent: controller

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

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

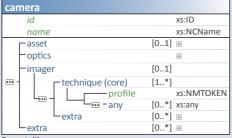


Parent: controller

‡ semantic: see note for input (shared)

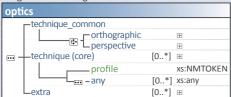
Camera Elements [5]

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



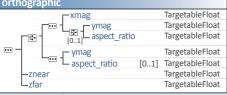
Parent: library_cameras

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



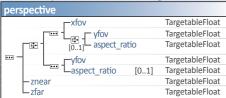
Parent: camera

Describes the field of view of an orthographic camera.



Parents: optics / technique common

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



Parents: optics / 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: extra, source (core), light, optics, imager, force_field, physics_material, physics_scene, rigid_body, rigid_constraint, instance_rigid_body, bind_material

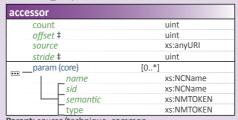
Data Flow Elements [5]

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
r asset	[01] ⊞
☐ IDREF_array	[01] ⊞
─ Name_array	[01] ⊞
□ + ⊕ + bool_array	[01] ⊞
[01] — float_array	[01] ⊞
☐ int_array	[01] ⊞
tęchnique_common	[01]
- accessor	⊞
Lechnique (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>.



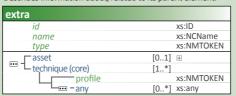
Parent: source/technique_common ‡ Defaults: offset = 0, stride = 1

Declares storage for a homogenous array. <bool_array> uses type ListOfBools, an xs:list of type xs:boolean. <Name_array> uses type ListOfNames, an xs:list of type xs:Name.

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

Parent: source (core)

Describes information about/related to its parent element.



Parents: animation, 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)

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

rotate, translate xs:NCName

Parents: node, 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

Parent: source (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

Parent: source (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

Parent: source (core)

‡ Defaults: digits = 6, magnitude = 38

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

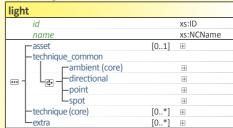
input (shared)	InputLocalOffset
offset	uint
semantic ‡	xs:NMTOKEN
source	xs:URIFragmentType
set	uint

Parents: lines, 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: library_lights

Lighting Elements Continued >

Lighting Elements (continued)

Describes an ambient light source.

ambient (core), directional			
└ sid xs:NCName			

Parent: light/technique_common

Describes a spot light source.

spo	t		
	r color		TargetableFloat3
	_ constant_attenuation	[01]	TargetableFloat
_	 linear_attenuation 	[01]	TargetableFloat
•••	quadratic_attenuation	[01]	TargetableFloat
	falloff_angle	[01]	TargetableFloat
	L falloff_exponent	[01]	TargetableFloat

Parent: light/technique_common

Describes a point light source.

point			
_ color		TargetableFloat3	
constant_attenuation ‡	[01]	TargetableFloat	
linear_attenuation ‡	[01]	TargetableFloat	
quadratic attenuation ‡	[01]	TargetableFloat	

Parent: light/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_mat	erial		
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*]	±
L extra		[0*]	±

Parents: library_physics_materials, shape, {instance_}rigid_body/technique_common

‡ {dynamic, static}_friction, restitution: Default = 0

FX: Rendering Elements (COMMON) [8]

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

blir	n, phong		
	emission	[01]	
	- ambient (FX)	[01]	
	- diffuse	[01]	
	- specular	[01]	
	shininess	[01]	
	- reflective	[01]	
	- reflectivity	[01]	
	- transparent	[01]	*_transparent_type
	Lopaque ‡		fx_opaque_enum
	- transparency	[01]	
	index_of_ refraction	[01]	⊞ *_float_or_param_type

Parents: technique (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

con	stant		
	emission	[01] # *_color_or_texture_	type
	- reflective	[01] # *_color_or_texture_	type
	reflectivity	[01] # *_float_or_param_ty	/ре
	transparent	[01] *_transparent_type	
	Lopaque ‡	fx_opaque_enum	
	- transparency	[01] # *_float_or_param_ty	/ре
	index_of_ refraction	[01] \boxplus *_float_or_param_ty	/pe

Parent: technique (FX) in profile_COMMON ‡ opaque: A_ONE, RGB_ZERO. Default = A_ONE

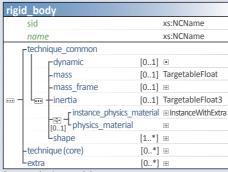
Physics Model Elements [6]

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*] ⊞
	[0*] ⊞
instance_physics_model	[0*] ⊞
Lextra	[0*] ⊞

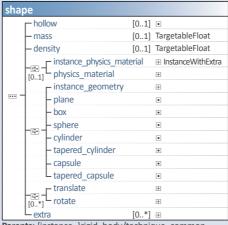
Parent: library_physics_models

Describes simulated bodies that do not deform



Parent: physics model

Describes components of a < rigid body>



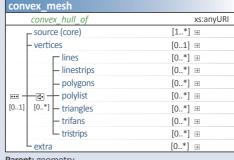
Parents: {instance_}rigid_body/technique_common

Defines the center and orientation of the rigid body.

mass	_frame	
	translate	±
[1*]	· rotate	Œ

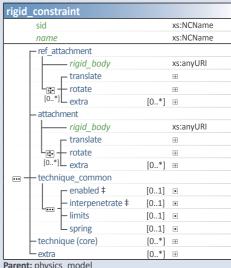
Parent: rigid_body/technique_common

Contains or refers to information that describes basic geometric meshes



Parent: geometry

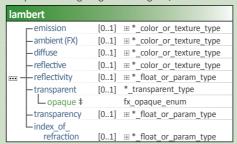
Connects components, such as <rigid_body>, into complex physics models with moveable parts.



Parent: physics_model

‡ Defaults: enabled = True, interpenetrate = False

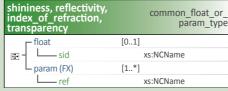
Describes a diffuse shaded surface that is independent of lighting. In the diagram, * = common



Parent: technique (FX) in profile_COMMON

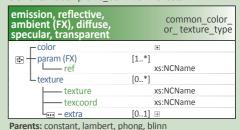
‡ opaque: A_ONE, RGB_ZERO. Default = A_ONE

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

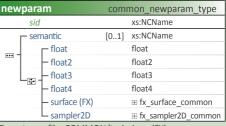


Parents: constant, lambert, phong, blinn

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



Parents: profile_COMMON/technique (FX)

FX: Texturing Elements (All Profiles) [8]

Declares the storage for the graphical representation of

imag	e		
	id		xs:ID
	name		xs:NCName
	format		xs:token
	height		uint
	width		uint
	depth		uint
	- asset	[01]	\blacksquare
	data		ListOfHexBinary
	-⊞ -L init_from		xs:anyURI
	– 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
	Lmipmap_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, * = common			
sur	face		fx_surface_common	
	type ‡		fx_surface_type_enum	
	rinit_as_null		xs:anyType	
	-init_as_target		xs:anyType	
	init_cube		fx_surface_init_cube_*	
	[01] init_volume		fx_surface_init_volume_*	
	-init_planar		fx_surface_init_planar_*	
	Linit_from	[1*]	fx_surface_init_from_*	
	_mip ‡		xs:unsignedInt	
	slice ‡		xs:unsignedInt	
	∟face ‡		fx_surface_face_enum	
	- format	[01]	xs:token	
	- format_hint	[01]	<pre>fx_surface_format_hint_*</pre>	
	rsize ‡		int3	
	viewport_ratio ‡		float2	
	- mip_levels ‡	[01]	xs:unsignedInt	
	- mipmap_generate ‡	[01]	xs:boolean	
	L _{extra}	[0*]	⊞	

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

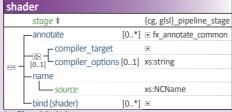
Parents: COMMON - newparam, setparam; CG - newparam, setparam, array, shader/bind, usertype; GLES - newparam, setparam, texture_unit, GLSL - newparam, setparam, array, shader/bind

‡ 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

FX: Shader Elements (Other Profiles) [8]

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



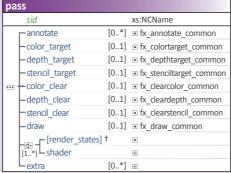
Profiles: CG. GLSL

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.



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		
5	id		xs:NCName	
⊢ a	nnotate	[0*]	■ fx_annotate_common	
	emantic	[01]	xs:NCName	
— n	nodifier‡	[01]	$fx_modifier_enum_common$	
L	— [values] †		■ fx_basic_type_common	
D (1) /				

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}.

newparam			{glsl, cg}_newparam
sid		{glsl, cg}_identifier	
	-annotate	[0*]	■ fx_annotate_common
	semantic	[01]	xs:NCName
	-modifier ‡	[01]	fx_modifier_enum_common
	r ⊕ - [value	<u>[</u> s] †	■ {glsl, cg}_param_type
	array usertype		■ {glsl, cg}_newarray_type
			■ cg_setuser_type

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>

san	npler_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	
	Lextra	[0*] ⊞	
Prof	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

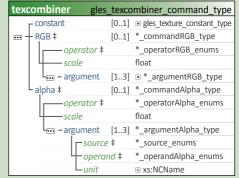
Defines a texture unit that will be mapped to hardware texture units based on its usage in <texture_pipeline>

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

Profile: GLES

Parents: setparam, 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]	
r annotate	[0*]	■ fx_annotate_common
r code		■ fx_code_profile
include		■ fx_include_common
name		\oplus
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/va	alue gles_texture_pipeline
sid	xs:NCName
	⊞ gles_texcombiner_command_type
texenv 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		
constant (combiner) [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	田
extra	[0*] ⊞

Parent: library material

Instantiates a COLLADA effect.

instance_effect	
url	xs:anyURI
sid	xs:NCName
name	xs:NCName
rtechnique_hint	[01]
– platform	xs:NCName
□ □ profile	xs:NCName
∟ref	xs:NCName
- setparam	[0*]
L extra	[0*] ⊞

Parents: material, render

FX: Effects Elements [8]

Declares a self-contained description of a COLLADA effect.

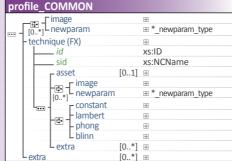
effe	ect		
	id		xs:ID
	name		xs:NCName
	r asset	[01]	\pm
	annotate	[0*]	■ fx_annotate_common
l	_ image	[0*]	\pm
	_ newparam	[0*]	⊞ fx_newparam_common
	_ [fx_profile_abstract] ‡	[1*]	xs:anyType
	L extra	[0*]	H

Profile: Effect

Parent: library_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.

pro	file_GLES		
id			xs:ID
platform			xs:NCName
	_asset	[01]	H
-	image		H
	[0*] newparam		⊞ gles_newparam
	technique (FX)	[1*]	Œ
	∟ _{extra}	[0*]	⊞

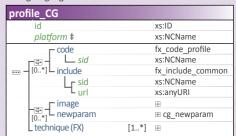
Profile: GLES Parent: effect



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

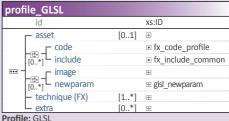
Declares platform-specific data types and techniques for the Cg language.



Profile: CG Parent: effect

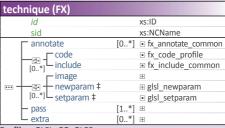
‡ platform: Default = "PC"

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



Profile: GLSL Parent: effect

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.

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.

<extra type="MY TYPE"> <technique profile="PROFILE-A"> </technique> <technique profile="PROFILE-B"> </technique> </extra>

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.