RenderScript Object Types

Overview

The types below are used to manipulate RenderScript objects like allocations, samplers, elements, and scripts. Most of these object are created using the Java RenderScript APIs.

Summary

Types	
rs_allocation	Handle to an allocation
rs_allocation_cubemap_face	Enum for selecting cube map faces
rs_allocation_usage_type	Bitfield to specify how an allocation is used
rs_data_kind	Element data kind
rs_data_type	Element basic data type
rs_element	Handle to an element
rs_sampler	Handle to a Sampler
rs_sampler_value	Sampler wrap T value
rs_script	Handle to a Script
rs_type	Handle to a Type
rs_yuv_format	YUV format

Types

rs_allocation : Handle to an allocation

An opaque handle to a RenderScript allocation.

See android.renderscript.Allocation.

rs_allocation_cubemap_face : Enum for selecting cube map faces

An enum with the following values: Added in API level 14

RS_ALLOCATION_CUBEMAP_FACE_POSITIVE_X = 0

RS_ALLOCATION_CUBEMAP_FACE_NEGATIVE_X = 1

RS_ALLOCATION_CUBEMAP_FACE_POSITIVE_Y = 2

RS_ALLOCATION_CUBEMAP_FACE_NEGATIVE_Y = 3

RS_ALLOCATION_CUBEMAP_FACE_POSITIVE_Z = 4

RS_ALLOCATION_CUBEMAP_FACE_NEGATIVE_Z = 5

An enum used to specify one the six faces of a cubemap.

rs_allocation_usage_type : Bitfield to specify how an allocation is used

An enum with the following values: Added in API level 14

RS_ALLOCATION_USAGE_SCRIPT = 0x0001	Allocation is bound to and accessed by scripts.
RS_ALLOCATION_USAGE_GRAPHICS_TEXTURE = 0x0002	Allocation is used as a texture source.
RS_ALLOCATION_USAGE_GRAPHICS_VERTEX = 0x0004	Deprecated.
RS_ALLOCATION_USAGE_GRAPHICS_CONSTANTS = 0x0008	Deprecated.
RS_ALLOCATION_USAGE_GRAPHICS_RENDER_TARGET = 0x0010	Deprecated.
RS_ALLOCATION_USAGE_IO_INPUT = 0x0020	Allocation is used as a Surface consumer.
RS_ALLOCATION_USAGE_IO_OUTPUT = 0x0040	Allocation is used as a Surface producer.
RS_ALLOCATION_USAGE_SHARED = 0x0080	Allocation's backing store is shared with another object (usually a Bitmap). Copying to or from the original source Bitmap will cause a synchronization rather than a full copy.

These values are ORed together to specify which usages or memory spaces are relevant to an allocation or an operation on an allocation.

rs_data_kind : Element data kind

An enum with the following values: Added in API level 16

RS_KIND_USER = 0	No special interpretation.
RS_KIND_PIXEL_L = 7	Luminance.
RS_KIND_PIXEL_A = 8	Alpha.
RS_KIND_PIXEL_LA = 9	Luminance and Alpha.
RS_KIND_PIXEL_RGB = 10	Red, Green, Blue.
RS_KIND_PIXEL_RGBA = 11	Red, Green, Blue, and Alpha.
RS_KIND_PIXEL_DEPTH = 12	Depth for a depth texture.
RS_KIND_PIXEL_YUV = 13	Luminance and chrominance.
RS_KIND_INVALID = 100	

This enumeration is primarly useful for graphical data. It provides additional information to help interpret the rs_data_type.

RS_KIND_USER indicates no special interpretation is expected.

The RS_KIND_PIXEL_* values are used in conjunction with the standard data types for representing texture formats.

See the Element.createPixel() method.

rs_data_type : Element basic data type

An enum with the following values: Added in API level 16

RS_TYPE_NONE = 0	Element is a complex type, i.e. a struct.
RS_TYPE_FLOAT_16 = 1	A 16 bit floating point value.
RS_TYPE_FLOAT_32 = 2	A 32 bit floating point value.
RS_TYPE_FLOAT_64 = 3	A 64 bit floating point value.
RS_TYPE_SIGNED_8 = 4	An 8 bit signed integer.
RS_TYPE_SIGNED_16 = 5	A 16 bit signed integer.
RS_TYPE_SIGNED_32 = 6	A 32 bit signed integer.
RS_TYPE_SIGNED_64 = 7	A 64 bit signed integer.
RS_TYPE_UNSIGNED_8 = 8	An 8 bit unsigned integer.

RS_TYPE_UNSIGNED_16 = 9	A 16 bit unsigned integer.
RS_TYPE_UNSIGNED_32 = 10	A 32 bit unsigned integer.
RS_TYPE_UNSIGNED_64 = 11	A 64 bit unsigned integer.
RS_TYPE_BOOLEAN = 12	0 or 1 (false or true) stored in an 8 bit container.
RS_TYPE_UNSIGNED_5_6_5 = 13	A 16 bit unsigned integer packing graphical data in 5, 6, and 5 bit sections.
RS_TYPE_UNSIGNED_5_5_5_1 = 14	A 16 bit unsigned integer packing graphical data in 5, 5, 5, and 1 bit sections.
RS_TYPE_UNSIGNED_4_4_4_4 = 15	A 16 bit unsigned integer packing graphical data in 4, 4, 4, and 4 bit sections.
RS_TYPE_MATRIX_4X4 = 16	A 4x4 matrix of 32 bit floats, aligned on a 32 bit boundary.
RS_TYPE_MATRIX_3X3 = 17	A 3x3 matrix of 32 bit floats, aligned on a 32 bit boundary.
RS_TYPE_MATRIX_2X2 = 18	A 2x2 matrix of 32 bit floats, aligned on a 32 bit boundary.
RS_TYPE_ELEMENT = 1000	A handle to an Element.
RS_TYPE_TYPE = 1001	A handle to a Type.
RS_TYPE_ALLOCATION = 1002	A handle to an Allocation.
RS_TYPE_SAMPLER = 1003	A handle to a Sampler.
RS_TYPE_SCRIPT = 1004	A handle to a Script.
RS_TYPE_MESH = 1005	Deprecated.
RS_TYPE_PROGRAM_FRAGMENT = 1006	Deprecated.
RS_TYPE_PROGRAM_VERTEX = 1007	Deprecated.
RS_TYPE_PROGRAM_RASTER = 1008	Deprecated.
RS_TYPE_PROGRAM_STORE = 1009	Deprecated.
RS_TYPE_FONT = 1010	Deprecated.
RS_TYPE_INVALID = 10000	

rs_data_type is used to encode the type information of a basic element.

RS_TYPE_UNSIGNED_5_6_5, RS_TYPE_UNSIGNED_5_5_5_1, RS_TYPE_UNSIGNED_4_4_4_4 are for packed graphical data formats and represent vectors with per vector member sizes which are treated as a single unit for packing and alignment purposes.

rs_element : Handle to an element

An opaque handle to a RenderScript element.

See android.renderscript.Element.

rs_sampler : Handle to a Sampler

An opaque handle to a RenderScript sampler object.

See android.renderscript.Sampler.

rs_sampler_value : Sampler wrap T value

An enum with the following values: Added in API level 16

RS_SAMPLER_NEAREST = 0

RS_SAMPLER_LINEAR = 1

RS_SAMPLER_LINEAR_MIP_LINEAR = 2

RS_SAMPLER_WRAP = 3

RS_SAMPLER_CLAMP = 4

RS_SAMPLER_LINEAR_MIP_NEAREST = 5

RS_SAMPLER_MIRRORED_REPEAT = 6

RS_SAMPLER_INVALID = 100

rs_script : Handle to a Script

An opaque handle to a RenderScript script object.

See android.renderscript.ScriptC.

rs_type : Handle to a Type

An opaque handle to a RenderScript type.

See android.renderscript.Type.

rs_yuv_format : YUV format

An enum with the following values: Added in API level 24

RS_YUV_NONE = 0

RS_YUV_YV12 = 0x32315659

RS_YUV_NV21 = 0x11

RS_YUV_420_888 = 0x23

Android YUV formats that can be associated with a RenderScript Type.

See android.graphics.ImageFormat for a description of each format.