RenderScript Numerical Types

Overview

Scalars:

RenderScript supports the following scalar numerical types:

	8 bits	16 bits	32 bits	64 bits
Integer:	char, int8_t	short, int16_t	int32_t	long, long long, int64_t
Unsigned integer:	uchar, uint8_t	ushort, uint16_t	uint, uint32_t	ulong, uint64_t
Floating point:		half	float	double

Vectors:

RenderScript supports fixed size vectors of length 2, 3, and 4. Vectors are declared using the common type name followed by a 2, 3, or 4. E.g. float4, int3, double2, ulong4.

To create vector literals, use the vector type followed by the values enclosed between curly braces, e.g. `(float3){1.0f, 2.0f, 3.0f}`.

Entries of a vector can be accessed using different naming styles.

Single entries can be accessed by following the variable name with a dot and:

- The letters x, y, z, and w,
- The letters r, g, b, and a,
- The letter s or S, followed by a zero based index.

For example, with `int4 myVar;` the following are equivalent:

```
myVar.x == myVar.r == myVar.s0 == myVar.S0
myVar.y == myVar.g == myVar.s1 == myVar.S1
myVar.z == myVar.b == myVar.s2 == myVar.S2
myVar.w == myVar.a == myVar.s3 == myVar.S3
```

Multiple entries of a vector can be accessed at once by using an identifier that is the concatenation of multiple letters or indices. The resulting vector has a size equal to the number of entries named.

With the example above, the middle two entries can be accessed using `myVar.yz`, `myVar.gb`, `myVar.s12`, and `myVar.S12`.

The entries don't have to be contiguous or in increasing order. Entries can even be repeated, as long as we're not trying to assign to it. You also can't mix the naming styles.

Here are examples of what can or can't be done:

```
float4 v4;
float3 v3;
float2 v2;
v2 = v4.xx; // Valid
v3 = v4.zxw; // Valid
v3 = v4.bba; // Valid
v3 = v4.s032; // Valid
v3.s120 = v4.S233; // Valid
```

```
v4.yz = v3.rg; // Valid
v4.yzx = v3.rg; // Invalid: mismatched sizes
v4.yzz = v3; // Invalid: z appears twice in an assignment
v3 = v3.xas0; // Invalid: can't mix xyzw with rgba nor s0...
v3 = v4.s034; // Invalid: the digit can only be 0, 1, 2, or 3
```

Matrices and Quaternions:

RenderScript supports fixed size square matrices of floats of size 2x2, 3x3, and 4x4. The types are named rs_matrix2x2, rs_matrix3x3, and rs_matrix4x4. See Matrix Functions for the list of operations.

Quaternions are also supported via rs_quaternion. See Quaterion Functions for the list of operations.

Summary

Types	
char2	Two 8 bit signed integers
char3	Three 8 bit signed integers
char4	Four 8 bit signed integers
double2	Two 64 bit floats
double3	Three 64 bit floats
double4	Four 64 bit floats
float2	Two 32 bit floats
float3	Three 32 bit floats
float4	Four 32 bit floats
half	16 bit floating point value
half2	Two 16 bit floats
half3	Three 16 bit floats
half4	Four 16 bit floats
int16_t	16 bit signed integer
int2	Two 32 bit signed integers
int3	Three 32 bit signed integers
int32_t	32 bit signed integer
int4	Four 32 bit signed integers
int64_t	64 bit signed integer
int8_t	8 bit signed integer
long2	Two 64 bit signed integers
long3	Three 64 bit signed integers
long4	Four 64 bit signed integers
rs_matrix2x2	2x2 matrix of 32 bit floats
rs_matrix3x3	3x3 matrix of 32 bit floats
rs_matrix4x4	4x4 matrix of 32 bit floats
rs_quaternion	Quaternion
short2	Two 16 bit signed integers

short3	Three 16 bit signed integers
short4	Four 16 bit signed integers
size_t	Unsigned size type
ssize_t	Signed size type
uchar	8 bit unsigned integer
uchar2	Two 8 bit unsigned integers
uchar3	Three 8 bit unsigned integers
uchar4	Four 8 bit unsigned integers
uint	32 bit unsigned integer
uint16_t	16 bit unsigned integer
uint2	Two 32 bit unsigned integers
uint3	Three 32 bit unsigned integers
uint32_t	32 bit unsigned integer
uint4	Four 32 bit unsigned integers
uint64_t	64 bit unsigned integer
uint8_t	8 bit unsigned integer
ulong	64 bit unsigned integer
ulong2	Two 64 bit unsigned integers
ulong3	Three 64 bit unsigned integers
ulong4	Four 64 bit unsigned integers
ushort	16 bit unsigned integer
ushort2	Two 16 bit unsigned integers
ushort3	Three 16 bit unsigned integers
ushort4	Four 16 bit unsigned integers

Types

char2 : Two 8 bit signed integers

A typedef of: char __attribute__((ext_vector_type(2)))

A vector of two chars. These two chars are packed into a single 16 bit field with a 16 bit alignment.

char3 : Three 8 bit signed integers

A typedef of: char __attribute__((ext_vector_type(3)))

A vector of three chars. These three chars are packed into a single 32 bit field with a 32 bit alignment.

char4 : Four 8 bit signed integers

A typedef of: char __attribute__((ext_vector_type(4)))

A vector of four chars. These four chars are packed into a single 32 bit field with a 32 bit alignment.

double2 : Two 64 bit floats

A typedef of: double __attribute__((ext_vector_type(2)))

A vector of two doubles. These two double fields packed into a single 128 bit field with a 128 bit alignment.

double3 : Three 64 bit floats

A typedef of: double __attribute__((ext_vector_type(3)))

A vector of three doubles. These three double fields packed into a single 256 bit field with a 256 bit alignment.

double4 : Four 64 bit floats

A typedef of: double __attribute__((ext_vector_type(4)))

A vector of four doubles. These four double fields packed into a single 256 bit field with a 256 bit alignment.

float2 : Two 32 bit floats

A typedef of: float __attribute__((ext_vector_type(2)))

A vector of two floats. These two floats are packed into a single 64 bit field with a 64 bit alignment.

A vector of two floats. These two floats are packed into a single 64 bit field with a 64 bit alignment.

float3 : Three 32 bit floats

A typedef of: float __attribute__((ext_vector_type(3)))

A vector of three floats. These three floats are packed into a single 128 bit field with a 128 bit alignment.

float4 : Four 32 bit floats

A typedef of: float __attribute__((ext_vector_type(4)))

A vector of four floats type. These four floats are packed into a single 128 bit field with a 128 bit alignment.

half : 16 bit floating point value

A typedef of: __fp16   Added in API level 23

A 16 bit floating point value.

half2 : Two 16 bit floats

A typedef of: half __attribute__((ext_vector_type(2)))   Added in API level 23

Vector version of the half float type. Provides two half fields packed into a single 32 bit field with 32 bit alignment.

half3 : Three 16 bit floats

A typedef of: half __attribute__((ext_vector_type(3)))   Added in API level 23

Vector version of the half float type. Provides three half fields packed into a single 64 bit field with 64 bit alignment.

half4 : Four 16 bit floats

A typedef of: half __attribute__((ext_vector_type(4)))   Added in API level 23

Vector version of the half float type. Provides four half fields packed into a single 64 bit field with 64 bit alignment.

int16_t : 16 bit signed integer

A typedef of: short

A 16 bit signed integer type.

int2 : Two 32 bit signed integers

A typedef of: int __attribute__((ext_vector_type(2)))

A vector of two ints. These two ints are packed into a single 64 bit field with a 64 bit alignment.

int3 : Three 32 bit signed integers

A typedef of: int __attribute__((ext_vector_type(3)))

A vector of three ints. These three ints are packed into a single 128 bit field with a 128 bit alignment.

int32_t : 32 bit signed integer

A typedef of: int

A 32 bit signed integer type.

int4 : Four 32 bit signed integers

A typedef of: int __attribute__((ext_vector_type(4)))

A vector of four ints. These two fours are packed into a single 128 bit field with a 128 bit alignment.

int64_t : 64 bit signed integer

A typedef of: long long   Removed from API level 21 and higher

A typedef of: long   Added in API level 21

A 64 bit signed integer type.

int8_t : 8 bit signed integer

A typedef of: char

8 bit signed integer type.

long2 : Two 64 bit signed integers

A typedef of: long __attribute__((ext_vector_type(2)))

A vector of two longs. These two longs are packed into a single 128 bit field with a 128 bit alignment.

long3 : Three 64 bit signed integers

A typedef of: long __attribute__((ext_vector_type(3)))

A vector of three longs. These three longs are packed into a single 256 bit field with a 256 bit alignment.

long4 : Four 64 bit signed integers

A typedef of: long __attribute__((ext_vector_type(4)))

A vector of four longs. These four longs are packed into a single 256 bit field with a 256 bit alignment.

rs_matrix2x2 : 2x2 matrix of 32 bit floats

A structure with the following fields:

float m[4]

A square 2x2 matrix of floats. The entries are stored in the array at the location [row*2 + col].

See Matrix Functions.

rs_matrix3x3 : 3x3 matrix of 32 bit floats

A structure with the following fields:

float m[9]

A square 3x3 matrix of floats. The entries are stored in the array at the location [row*3 + col].

See Matrix Functions.

rs_matrix4x4 : 4x4 matrix of 32 bit floats

A structure with the following fields:

float m[16]

A square 4x4 matrix of floats. The entries are stored in the array at the location [row*4 + col].

See Matrix Functions.

rs_quaternion : Quaternion

A typedef of: float4

A square 4x4 matrix of floats that represents a quaternion.

See Quaternion Functions.

short2 : Two 16 bit signed integers

A typedef of: short __attribute__((ext_vector_type(2)))

A vector of two shorts. These two shorts are packed into a single 32 bit field with a 32 bit alignment.

short3 : Three 16 bit signed integers

A typedef of: short __attribute__((ext_vector_type(3)))

A vector of three shorts. These three short fields packed into a single 64 bit field with a 64 bit alignment.

short4 : Four 16 bit signed integers

A typedef of: short __attribute__((ext_vector_type(4)))

A vector of four shorts. These four short fields packed into a single 64 bit field with a 64 bit alignment.

size_t : Unsigned size type

A typedef of: uint64_t	When compiling for 64 bits.
A typedef of: uint32_t	When compiling for 32 bits.

Unsigned size type. The number of bits depend on the compilation flags.

ssize_t : Signed size type

A typedef of: int64_t	When compiling for 64 bits.
A typedef of: int32_t	When compiling for 32 bits.

Signed size type. The number of bits depend on the compilation flags.

uchar : 8 bit unsigned integer

A typedef of: uint8_t

8 bit unsigned integer type.

uchar2 : Two 8 bit unsigned integers

A typedef of: uchar __attribute__((ext_vector_type(2)))

A vector of two uchars. These two uchar fields packed into a single 16 bit field with a 16 bit alignment.

uchar3 : Three 8 bit unsigned integers

A typedef of: uchar __attribute__((ext_vector_type(3)))

A vector of three uchars. These three uchar fields packed into a single 32 bit field with a 32 bit alignment.

uchar4 : Four 8 bit unsigned integers

A typedef of: uchar __attribute__((ext_vector_type(4)))

A vector of four uchars. These four uchar fields packed into a single 32 bit field with a 32 bit alignment.

uint : 32 bit unsigned integer

A typedef of: uint32_t

A 32 bit unsigned integer type.

uint16_t : 16 bit unsigned integer

A typedef of: unsigned short

A 16 bit unsigned integer type.

uint2 : Two 32 bit unsigned integers

A typedef of: uint __attribute__((ext_vector_type(2)))

A vector of two uints. These two uints are packed into a single 64 bit field with a 64 bit alignment.

uint3 : Three 32 bit unsigned integers

A typedef of: uint __attribute__((ext_vector_type(3)))

A vector of three uints. These three uints are packed into a single 128 bit field with a 128 bit alignment.

uint32_t : 32 bit unsigned integer

A typedef of: unsigned int

A 32 bit unsigned integer type.

uint4 : Four 32 bit unsigned integers

A typedef of: uint __attribute__((ext_vector_type(4)))

A vector of four uints. These four uints are packed into a single 128 bit field with a 128 bit alignment.

uint64_t : 64 bit unsigned integer

A typedef of: unsigned long long   Removed from API level 21 and higher

A typedef of: unsigned long   Added in API level 21

A 64 bit unsigned integer type.

uint8_t : 8 bit unsigned integer

A typedef of: unsigned char

8 bit unsigned integer type.

ulong : 64 bit unsigned integer

A typedef of: uint64_t

A 64 bit unsigned integer type.

ulong2 : Two 64 bit unsigned integers

A typedef of: ulong __attribute__((ext_vector_type(2)))

A vector of two ulongs. These two ulongs are packed into a single 128 bit field with a 128 bit alignment.

ulong3 : Three 64 bit unsigned integers

A typedef of: ulong __attribute__((ext_vector_type(3)))

A vector of three ulongs. These three ulong fields packed into a single 256 bit field with a 256 bit alignment.

ulong4 : Four 64 bit unsigned integers

A typedef of: ulong __attribute__((ext_vector_type(4)))

A vector of four ulongs. These four ulong fields packed into a single 256 bit field with a 256 bit alignment.

ushort : 16 bit unsigned integer

A typedef of: uint16_t

A 16 bit unsigned integer type.

ushort2 : Two 16 bit unsigned integers

A typedef of: ushort __attribute__((ext_vector_type(2)))

A vector of two ushorts. These two ushort fields packed into a single 32 bit field with a 32 bit alignment.

ushort3 : Three 16 bit unsigned integers

A typedef of: ushort __attribute__((ext_vector_type(3)))

A vector of three ushorts. These three ushort fields packed into a single 64 bit field with a 64 bit alignment.

ushort4 : Four 16 bit unsigned integers

A typedef of: ushort __attribute__((ext_vector_type(4)))

A vector of four ushorts. These four ushort fields packed into a single 64 bit field with a 64 bit alignment.