

N&C RT Light

Map	Entity	Face
Key	Value	
_color	0.5 0.5 0.5	
_nacCustom		
_nacCustomEnabled	0	
_nacCustomLoopEnabled	1	
_nacCustomToggleEnabled	0	
_nacHz	30	
_nacdirection	0 0 1	
_naclightmax	5	
_naclightpow	1	
_naclighttype	0	
_nacpenumbraangle	3	
_nacumbraangle	4	
classname	light	
count		
light	10	
nacname		
origin	-166 -488 234	
spawnflags	0	
style	2	
_angfade	1	
_cone	10	
_fade	1	
_falloff	0	
_focus	1	
_spotangle		
_spotpoint		
_spotvector		
movewith		
target		
targetname		

Things to note:

The original Light and its properties still exist just extended for N&C, If you have a Light and Material tech that tech will use the lights as you do for that using the old fields only.

For all lights you place you can not use targetname if you do you will lose LightStyles from the bsp output completely for the light so when a target name is needed for a light we use nacname which is also required for a moving light to be spawned in-game.

Light Styles process by default @ 30hz but each light can be set with a different hz,
 Light Tracking (Spot, Dir only) is at FPS hz but is paced by the 20hz game tick mover
 Light Movers is at FPS hz and is also is paced by the 20hz game tick mover

Lights can move with any in-game entity with a targetname entered into movewith

Light Types:

ShpereLight:

Irradiance based omni light

// Sphere

vec3 c = light_center_radius.xyz - p;

float dist = length(c);

float rdist = 1.0 / dist;

vec3 L = c * rdist;

float irradiance = 2 * (1 - sqrt(max(0, 1 - square(light_parms2.z * rdist))));

irradiance = min(irradiance, max_solid_angle);

irradiance *= float(global_ubo.num_sphereLights); // 1 / pdf

light_color = light_parms.w * light_color * irradiance;

onb = construct_ONB_frisvad(L);

position_light = light_center_radius.xyz + (onb[0] * diskpt.x + onb[2] * diskpt.y - L * diskpt.z) * sphere_radius;

SpotLight:

Irradiance based spot light

// Spot

vec3 lightVector = (light_center_radius.xyz - p);

float lightDistance = length(lightVector);

vec3 lightDirection = normalize(lightVector);

float rdist = 1.0 / lightDistance;

float irradiance = 2 * (1 - sqrt(max(0, 1 - square(light_parms2.z * rdist))));

irradiance = min(irradiance, max_solid_angle);

irradiance *= float(global_ubo.num_sphereLights); // 1 / pdf

vec3 spotDirection = -normalize(light_parms.xyz);

float attenuation = SpotAttenuation(spotDirection, -lightDirection, light_parms2.x * (3.14159265358979323846 / 180.0), light_parms2.y * (3.14159265358979323846 / 180.0));

float falloff = LightFalloff(lightDistance);

float window = LightWindowing(lightDistance, light_parms2.z);

light_color = light_parms.w * light_color * irradiance * attenuation * falloff * window;

onb = construct_ONB_frisvad(lightDirection);

position_light = light_center_radius.xyz + (onb[0] * diskpt.x + onb[2] * diskpt.y - lightDirection * diskpt.z) *

sphere_radius;

Directional:

NoL based directional light

// Directional

vec3 lightDirection = -normalize(light_parms.xyz);

float nol = dot(n, lightDirection);

if (nol < 0.0) nol = dot(-n, lightDirection);

light_color = light_parms.w * light_color * nol;

onb = construct_ONB_frisvad(lightDirection);

position_light = light_center_radius.xyz + (onb[0] * diskpt.x + onb[2] * diskpt.y - lightDirection * diskpt.z) *

sphere_radius;

`_COLOR`

The color in RGB

The two mode exists **FLOAT** or **CHAR**

Note:

In RT it is known by pathtracers to use values for color red like 8000 0 0 not 0 - 1
For this reason it is recommended to always use the float type.

`_NACCUSTOM`

The custom LightStyle string

Supports: a-z 26 phases of light power

a = no power

z = full power

Note:

Phases control the power field and are not fixed to any set value

If your light has the power of 10 the light style of 'a' would be 0 and 'm' would be 5 with 'z' being 10

This will override the built in style automatically

`_NACCUSTOMENABLED`

The custom LightStyle active flag

Value 0 the light will ignore the light style and stay at full power of 'z'

Value 1 the light will playback the style set

Note:

This has no effect on the built in LightStyle presets via the style field

`_NACCUSTOMLOOPENABLED`

The custom LightStyle looping flag

Value 0 the light will playback one time

Value 1 the light will loop the playback endlessly

Note:

This will work with the built in LightStyle presets via the style field

`_NACCUSTOMTOGGLEENABLED`

The custom LightStyle pingpong flag

Value 0 the light do nothing

Value 1 the light will ping-pong the playback

Note:

This will work with the built in LightStyle presets via the style field

Ping-Pong can rewind a play once ramp via trigger

`_NACHZ`

The custom LightStyle playback speed

Value 0 the light will playback nothing

Value 1 to 1000 the light will tick the animation frame

Note:

This will work with the built in LightStyle presets via the style field

30 hz is the default because it looks the best and not overdone

_NACDIRECTION

The custom Light direction (Spots, Directional only)

Vector based and not angles

X Y Z

For easy directions these presets values

Down = 0 0 1

Up = 0 0 -1

Forward = 1 0 0

Backward = -1 0 0

Left = 0 1 0

Right = 0 -1 0

Note:

TARGET_LIGHTTRACKER was created for the RT light, simply add one in the map; place it and then set the light's target to the light trackers name and it will show the line and you are all set the light will look at the tracker and if the tracker moves the light updates to track the new position.

`_NACLIGHTMAX`

The custom Light max distance for all light types in world units

Recommend defaults:

SphereLight 1

SpotLight None just pick distance you want like 100 to reach across a large room

Directional 1

Note:

You can have bright short reaching lights

You can have dim long reaching lights

`_NACLIGHTPOW`

The custom Light power for all light types

Power affects the color of the light

Recommend defaults:

SphereLight 1

SpotLight 1

Directional 1

Note:

Use of non 0 - 1 values support

`_NACLIGHTTYPE`

The custom Light type

Values:

SphereLight 0

SpotLight 1

Directional 2

Note:

Dynamic lights default to 0 and their processing is the same original Q2RTX

`_NACUMBRAANGLE`

The custom Light Spotlight outer radius angle

Values: 0 - 90

90 is mirrored to get the 180 fov

Note:

This is the outer cone of the spotlight

`_NACPENUMBRAANGLE`

The custom Light Spotlight inner radius angle

Values: 0 - (Minus 10% of the outer radius)

Ex: if my outer cone is 90 then my inner cone needs to be 80 this give off the soft spot edges

Note:

This is the inner cone of the spotlight

`_LIGHT`

The original field used by N&C as a Irradiance root

It needs to set to 10 the default and left alone unless you need more noise

Note:

The engine still uses this field but not for TB light setup

nacname

The replacement field for targetName on lights only

The name for spawning in game is required for moving lights and:
target_light, trigger_light, target_lightLS, trigger_lightLS

Note:

TargetName when set forces LightStyle 32 which is an old way of having lights spawn in game with turn on or off toggleable logic

movewith

The movewith target

Move with the named target, uses the spawned XYZ offset of the two

Note:

target

The lookat target name

This has to be a TARGET_LIGHTTRACKER

Note:

TARGET_LIGHTTRACKER

Map	Entity	Face
Key	Value	
classname	target_lightracker	
delay	1	
ltAttachOffset	0.0 0.0 0.0	
ltMode	0	
ltTracker	Light1 Tracker	
movewith	trainModel	
origin	-296 -248 168	
targetname	testLookAt	
count		
spawnflags	0	

ItMode

The tracking mode name

Value 0 will use TB placement for the offset of the tracker

Value 1 will use the tracker location and add a offset given by the mapper

Note:

If tracker is moving via train:

Tracking 0 works with func_train

Tracking 0 and 1 works with model_train

ItAttachOffset

The tracking mode 1's XYZ attachment offset

Note:

movewith

The movewith target

Move with the named target

Note:

targetname

The named target

This is used to assign the light to the tracker

Note:
