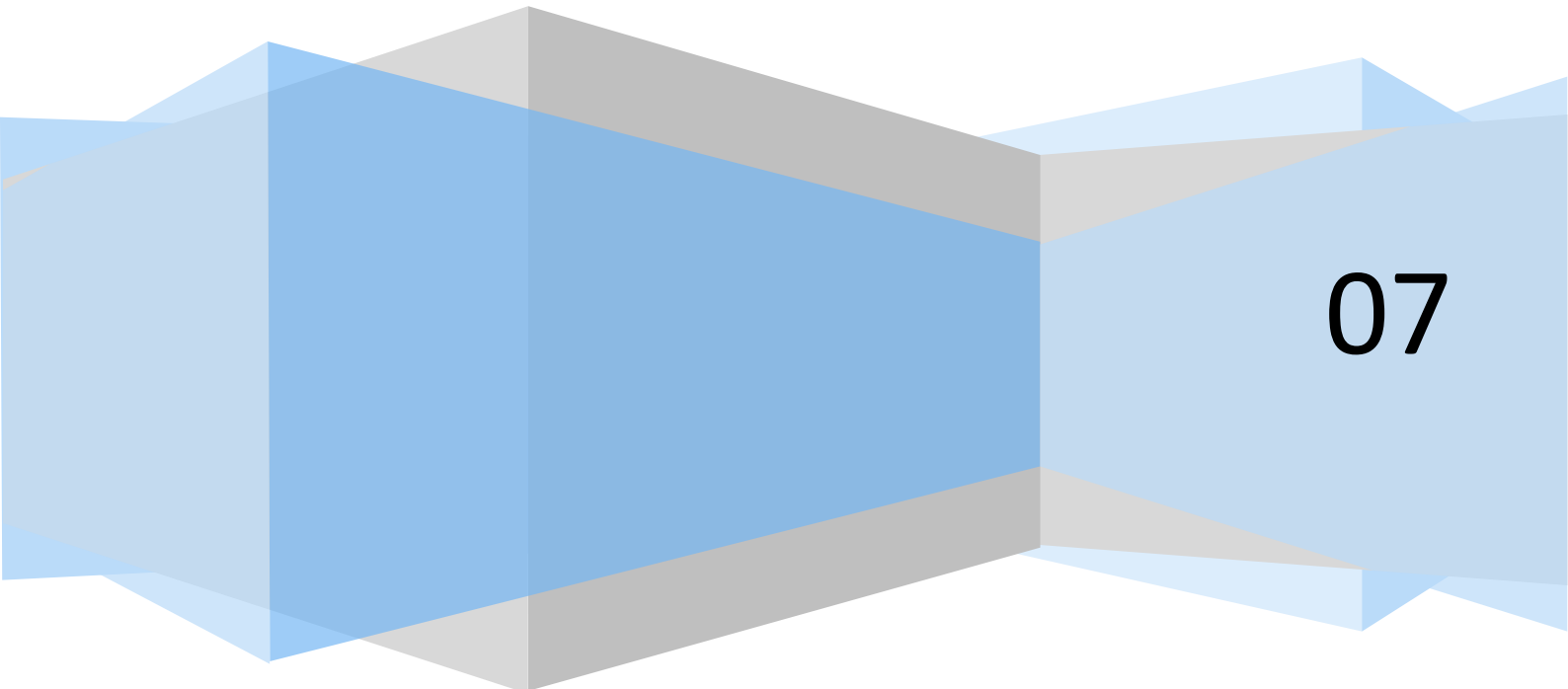


Corona Bytes .NET

CCI-Tutorial

How to use the FxLibrary

Raven



07

How to use the FxLibrary

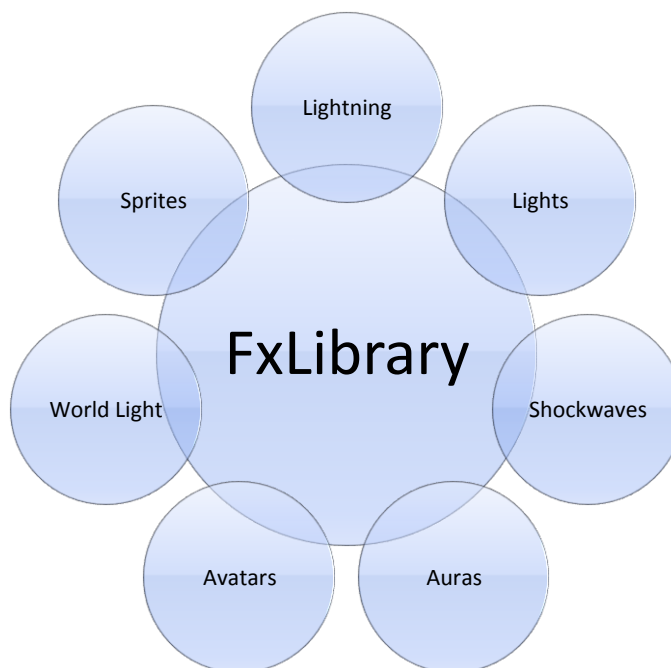
Introduction:

This tutorial will focus on the FxLibrary that ECX brings you. First I will try to give you an overview. Then I will try to give you a perspective on the syntax and at least we will go through all the Effects and I will try to explain them in detail. However there are endless possibilities so I won't be able to give you everything in this tutorial. But if you understood the basics I'm sure you can come up with something new on your own.

Another important thing is: the FxLib hasn't been created to be used by N00bs so some of you might as well fail to understand or fail to use it all together. I'm sorry but it was meant to be a developer tool and meant to be used by developers only. There might be crashes and all that, you can overuse it and you can slow down every machine and you can cause crashes if you don't handle it the right way.

What is the FxLib?

The FxLib is a tool; it is as the name states a Library of effects that can be called by different functions. It is meant to be used at transformations or special effects on characters for ECX.



Hello Shockwave

Hello Shockwave? Yes our most basic effect is a simple shockwave.

```
AddFx( Client, "FxBlow" );
```

AddFx & RemFx

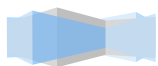
AddFx activates your Effect:

AddFx		
Client	Effect Name	additional Parameters

RemFx removes your Effect:

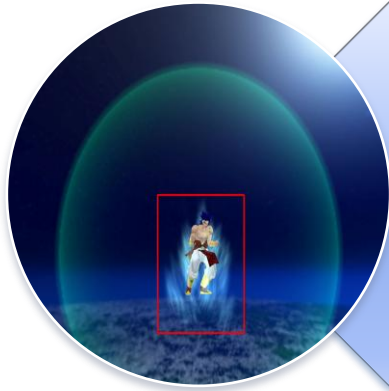
RemFx		
Client	Effect Name	additional Parameters

Sry I will explain them more detailed when I have the time.



Effects

fxAvatar



fxAvatar

- Client
- Effect[]
- command[]
- sequence
- float:framerate

```
AddFx( Client, "fxAvatar", "create", 102, 1.0 );
```

```
RemFx( Client, "fxAvatar", 0 );
```

This Effect is a rather complex one, although I will only explain this simple form of it. You see the two example lines. AddFx creates a new avatar at your origin coordinated and let it play sequence 102 at a framerate of 1.0.

fxBlow

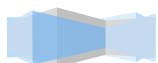


fxBlow

- Client
- Effect[]

```
AddFx( Client, "fxBlow" );
```

This is even simpler, it just calls a shockwave; no need to remove the effect.



fxCallSpirits



fxCallSpirits

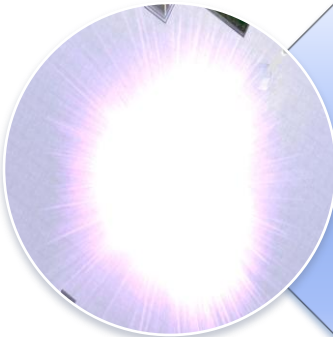
- Client
- Effect[]
- int:unknown
- z-offset

```
AddFx( Client, "fxCallSpirits", 1 , 50 );
```

```
RemFx( Client, "fxCallSpirits", 0 );
```

This is the Effect that ESF used for the SpiritBomb Charge. The only important parameter is actually the z-offset which is self-explaining as well.

fxExplosion

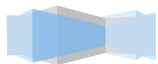


fxExplosion

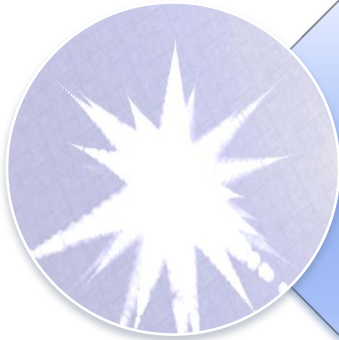
- Client
- Effect[]
- size
- color
- z-offset

```
AddFx( Client, "fxExplosion", 500 , 2, 0 );
```

This is the Message that calls ESF's default explosion effect.



fxExplosionsmoke



fxExplosionsmoke

- Client
- Effect[]
- size
- z-offset

```
AddFx( Client, "fxExplosion", 500 , 0 );
```

This is the Message that calls ESF's default explosion smoke effect.

fxLgtField



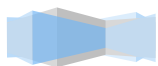
fxLgtField

- Client
- Effect[]
- float:effect rate
- sprite size
- brightness
- fieldsize
- z-fieldsize

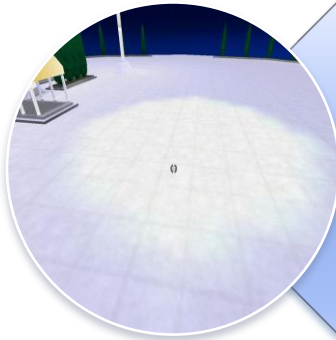
```
AddFx( Client, "fxLgtField", 0.4 , 1, 255, 20, 20 );
```

```
RemFx( Client, "fxLgtField", 0 );
```

This is a Lightning Field around your character.



fxWorldLight



fxWorldLight

- Client
- Effect[]
- radius
- r
- g
- b
- life
- decayrate

```
AddFx( Client, "fxWorldLight", 20, 100, 100, 100, 10, 1 );
```

This is a Light Field around your character.

fxEntityLight

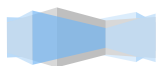


fxEntitiyLight

- Client
- Effect[]
- radius
- r
- g
- b
- life
- decayrate

```
AddFx( Client, "fxEntityLight", 20, 100, 100, 100, 10, 1 );
```

This is a Light on your character.



fxBSPLight



fxBSPLight

- Client
- Effect[]
- strength

```
AddFx( Client, "fxBSPLight", 5 );
```

This sets the BSPLight. Don't mess around with is or you'll crash the server.

fxLightning



fxLightning

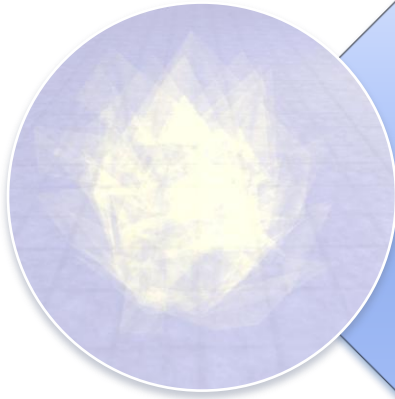
- Client
- Effect[]
- Sprite[]
- life
- width
- noise
- r
- g
- b
- brightness
- amount
- start range
- target range

```
AddFx( Client, "fxLightning", "sprites/lgtning.spr", 2, 50, 150, 200, 10, 10, 250, 10, 100, 0 );
```

```
RemFx( Client, "fxLightning", 0 );
```

This sets a Lightning effect similar as seen in Frieza's transformation.

fxModelEntity



fxModelEntity

- Client
- Effect[]
- Model[]
- attachment flag
- rendermode
- renderamt
- float:r
- float:g
- float:b
- framerate
- scale
- z-offset
- skin

```
AddFx( Client, "fxModelEntity", "models/evolution/Auras/default.mdl", 0, 0, 10.0, 0, 0, 0, 1.0, 0.3, 0, 4 );
```

```
RemFx( Client, "fxModelEntity", 0 );
```

This function allows you to spawn a model entity with lots of possible properties.

fxPowerup



fxPowerup

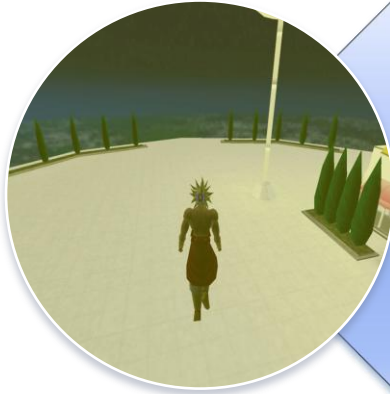
- Client
- Effect[]
- r
- g
- b

```
AddFx( Client, "fxPowerup", 150, 50, 200 );
```

```
RemFx( Client, "fxPowerup", 0 );
```

This created the Powerup Effect we know from the recharging.

fxScreenFade



fxScreenFade

- Client
- Effect[]
- float: Duration
- float: HoldTime
- float: Flags
- r
- g
- b
- alpha

```
AddFx( Client, "fxScreenFade", 3.0, 1.0, 0, 200, 200, 10, 100 );
```

This is the default ScreenFade.

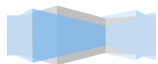
fxScreenShake



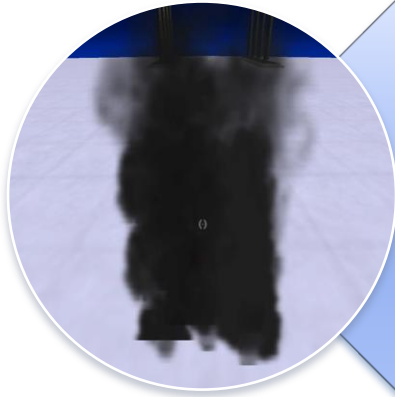
fxScreenShake

- Client
- Effect[]
- float: amplitude
- float: duration
- float: frequency

```
AddFx( Client, "fxScreenShake", 50.0, 3.0, 5.0 );
```



fxSmokeField



fxSmokeField

- Client
- Effect[]
- Sprite[]
- float: time
- sprite size
- framerate
- particle amount
- fieldsize

```
AddFx( Client, "fxSmokeField", "sprites/steam1.spr", 0.3, 6, 10, 10, 10);
```

```
RemFx( Client, "fxSmokeField", 0);
```

This creates a field of smoke around your character as seen in Frieza's transformation.

fxSprite



fxSprite

- Client
- Effect[]
- Sprite[]
- scale
- brightness
- x-offset
- y-offset
- z-offset

```
AddFx( Client, "fxSprite", "sprites/xflare1.spr", 15, 255, 0, 0, 0 );
```

This spawns a sprite that plays one circle.

fxSpriteBall



fxSpriteBall

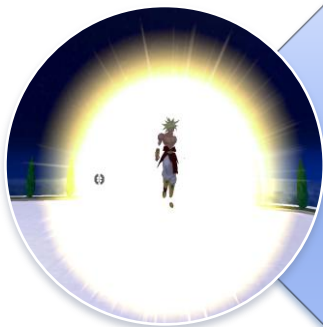
- Client
- Effect[]
- Sprite[]
- float: timer
- particle size
- brightness
- x-size
- y-size
- z-size

```
AddFx( Client, "fxSpriteBall", "sprites/ecx.BuuSmoke.spr", 0.5, 25, 255, 20, 20, 20 );
```

```
RemFx( Client, "fxSpriteBall", "sprites/ecx.BuuSmoke.spr", 0 );
```

This is a SpriteBall / Spherical Particle Effect which was seen in Buu's transformation.

fxSpriteEntity



fxSpriteEntity

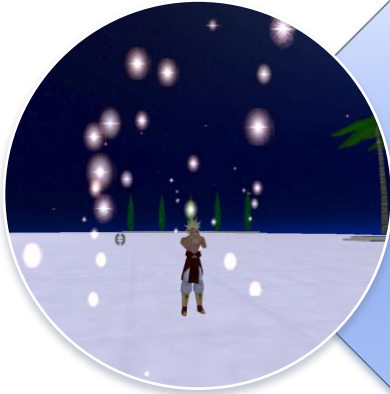
- Client
- Effect[]
- Model[]
- attachment flag
- rendermode
- renderamt
- float:r
- float:g
- float:b
- framerate
- scale
- z-offset
- skin

```
AddFx( Client, "fxSpriteEntity", "sprites/ecx.aura.ssj3.spr", 0, 0, 255.0, 10.0, 1.0, 0.4
```

```
RemFx( Client, "fxSpriteEntity", 0 );
```

This function allows you to spawn a sprite entity with lots of possible properties.

fxSpriteField



fxSpriteField

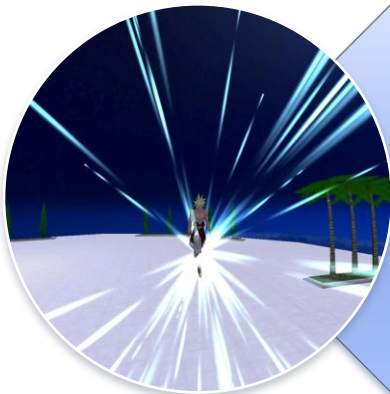
- Client
- Effect[]
- Sprite[]
- float: timer
- particle size
- brightness
- particle amount
- fieldsize

```
AddFx( Client, "fxSpriteField ", "sprites/xflare1.spr", 0.1, 1, 255, 5, 50 );
```

```
RemFx( Client, "fxSpritefield", 0 );
```

This is a sprite particle field.

fxSpriteRays



fxSpriteRays

- Client
- Effect[]
- Sprite[]
- life
- width
- noise
- r
- g
- b
- brightness
- speed
- fieldsize
- direction
- amount

```
AddFx( Client, "fxSpriteRays ", "sprites/xsmoke1.spr", 5, 15, 0, 150, 255, 255, 255, 25, 350, 0, 25);
```

This creates some Sprite Rays. Multiple configurations are possible ☺

fxCSpriteRays



fxCSpriteRays

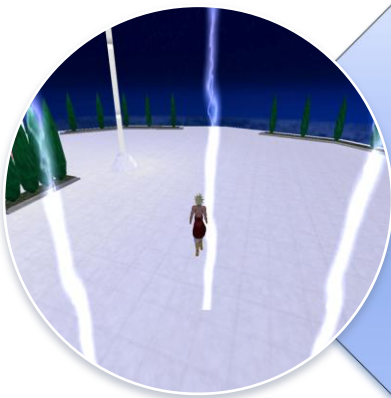
- Client
- Effect[]
- Sprite[]
- float:timer
- life
- width
- noise
- r
- g
- b
- brightness
- speed
- fieldsize
- direction
- amount

```
AddFx( Client, "fxSpriteRays "sprites/stmbal1.spr", 0.1, 30, 255, 0, 10, 10, 10, 250, 20, 700, 0, 3);
```

```
RemFx( Client, "fxSpriteRays ", 0);
```

This creates some **constant** Sprite Rays. Multiple configurations are possible ☺
Our example creates the moving clouds seen in Goku-TS's ssj3 transformation.

fxVerticalSpriteField



fxVerticalSpriteField

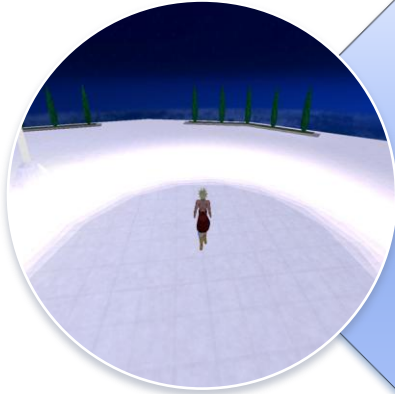
- Client
- Effect[]
- Sprite[]
- float: timer
- life
- width
- noise
- r
- g
- b
- brightness
- speed
- framerate
- fieldsize

```
AddFx( Client, "fxVerticalSpriteField ", "sprites/ecx.lightning.spr", 1.7, 12, 100, 0, 200, 180, 255, 255, 5, 150, 600 );
```

```
RemFx( Client, "fxVerticalSpritefield ", 0 );
```

This is a sprite vertical sprite particle field. E.g. like in Goku's ssj1 transformation.

fxPowerWave

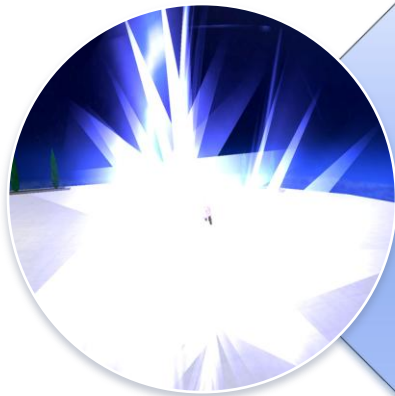


fxPowerWave

- Client
- Effect[]
- Sprite[]
- r
- g
- b
- brightness
- spread

```
AddFx( Client, "fxPowerWave", "sprites/white.spr", 180, 145, 120, 30, 250 );
```

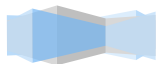
fxBeamTorus

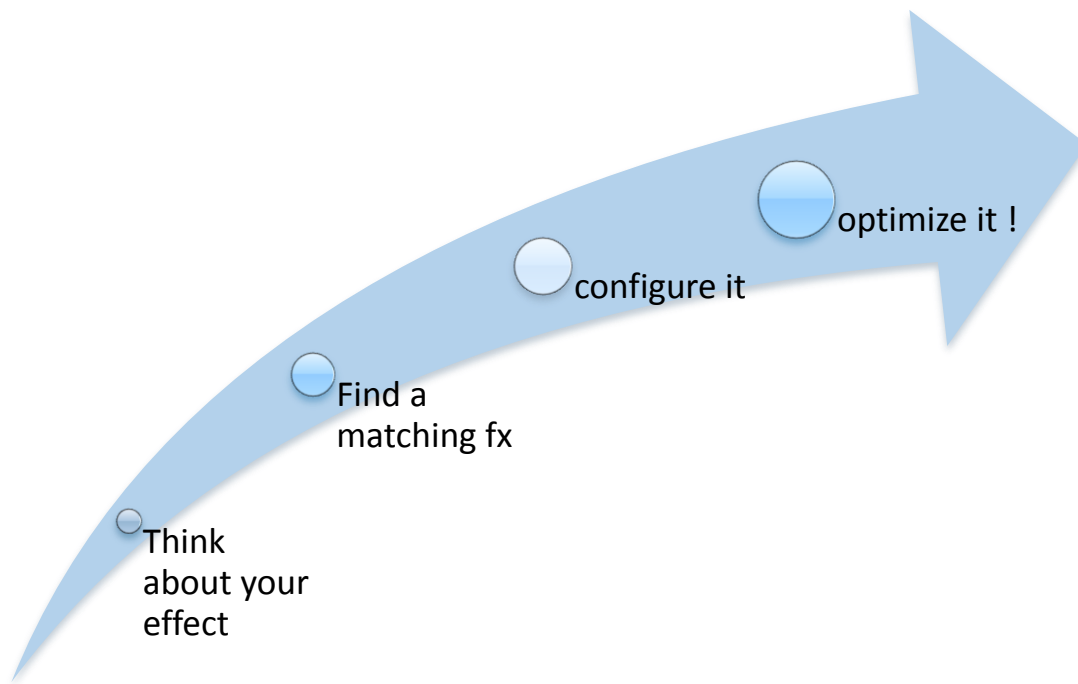


fxBeamTorus

- Client
- Effect[]
- Sprite[]
- r
- g
- b
- brightness
- spread
- life
- width
- amplitude

```
AddFx( Client, "fxBeamTorus", "sprites/lgtning.spr", 255, 255, 255, 255, 250, 8, 96, 0 );
```





Warnings

At the End of this rather strange tutorial I would like to warn you guys. Don't mess around too much with the fx effect. Since all of them are created as calculated server-side you will have to send these information to every client which will take away valuable network performance. In ECX I tried very hard to make them as best performing as possible.

I advice everybody to turn on your "netgraph" while testing those effects:

```
net_graph 2
```

In your console.

But now have fun

Regards Raven

& the Corona-Bytes Team

