

Titel: FX Erstellung, die Grundbegriffe

Schwierigkeit: Grundkenntnisse

Typ: Allgemeine Einführung

Programm: Final BIG

Alien aka Infiltrator

## Wieder einmal: Willkommen zum Tutorial

Heute behandeln wir das gemiedene Thema GFX.

Sämtliche Inhalte dieses Tutorials habe ich mir selbst beigebracht, indem ich stundenlang probiert habe, es gab damals kein einziges FX-Tutorial und auch heute sind diese spärlich vorhanden und nicht wirklich aufschlussreich, ich hoffe das hiermit zu ändern.

Zuerst muss man wissen wo wir unsere GFX finden, dies ist zum einen die „FXList.ini“ und zum anderen die „FXParticleSystem.ini“. Die Pfade lauten wie folgt:

data\ini\fxlist.ini

data\ini\fxparticlesystem.ini

Beide Dateien findet ihr in der „ini.big“.

Ich würde euch raten diese Dateien nicht umzuschreiben, stattdessen könnt ihr euch einen neuen FXPath anlegen oder eine Included-Datei erstellen(dies könnt ihr in meinem Tutorial „Erstellen einer Mod-BIG“ nachlesen).

Vorerst werde ich nur die einzelnen Grundbegriffe erklären und ein paar Textbeispiele dazu schildern, in kommenden Tutorials werde ich dann darauf eingehen wie ihr speziellere Effekte mit Zusatzzeilen erzeugen könnt und das alles mit Projectilen und Weapons verknüpft.

Beginn auf der nächsten Seite.

## **Inhaltsverzeichnis:**

Einführung	Seite 1
Die FXList und ihre Bestandteile	Seite 2 - 3
- ParticleSystem-Box	Seite 2
- Sound- und ViewShake-Box	Seite 3
Die FXParticleSystems	Seite 4 - 10
- Übersicht, kurze Begriffserklärung	Seite 4
- Das Beispielsystem	Seite 5
- System-Box	Seite 6
- Color- und Update-Box	Seite 7
- Physics-Box	Seite 8
- EmissionVelocity	Seite 8 - 9
- EmissionVolume	Seite 9 - 10
Hinweise und Informationen	Seite 10 - 11

Wir starten damit das ich euch die FXList erkläre, diese ist recht einfach und grundlegend für FX, allerdings wird diese oft auch umgangen (es wird öfter mal direkt das Particlesystem abgefragt statt der List).

Ich habe hier eine veränderte Version von Gandalfs „Wort der Macht“ (speziell für das Tutorial um die grundlegenden Zeilen in einer FXList zu haben):

```
; This is Gandalf's Shockwave Blast Attack (version)
FXList FX_TutorialBlast
  ParticleSystem
    Name = TutorialFX1
    Offset = X:0.0 Y:0.0 Z:60 ;Z:8.0
    OrientToObject = Yes
  End
  ParticleSystem
    Name = TutorialBlastShockwave
  End
  Sound
    Name = Tutorialsound
  End
  ViewShake
    Type = CINE_EXTREME
  End
End
```

Also das ganze ist einfacher als es aussieht (wie eigentlich alles bei EA Games Modding^^)

Ich werde euch einfach eine Zeile nach der anderen Erklären:

**FXList FX\_TutorialBlast**

Diese Zeile gibt an das es sich um eine FXList mit dem Namen „FX\_TutorialBlast“ handelt, theoretisch könnte die Zeile auch „FXList Tutorial“ heißen, es gibt also keine Bedingungen für das benennen der List.

**Particlesystem**

Diese Zeile sagt dem Spiel – Hier kommt jetzt ein Verweis auf ein Particlesystem. Das heißt, es wird eine Art Box geöffnet die alles über das System enthält, diese Box wird mit „End“ wieder zu gemacht (wie alle Boxen).

Sollte nur ein Sound gespielt werden könnt ihr die Particle weglassen und nur eine Soundbox einbauen.

**Name = TutorialBlastFinalLight**

Der Name der FX, jedes System hat einen Namen und dieser muss hier angegeben werden, denn die FXList ruft sämtliche Systeme auf die in ihr genannt werden, und es kommt zu Fehlern wenn ein Tippfehler vorhanden ist.

**Offset = X:0.0 Y:0.0 Z:20.0**

Dies ist die Position an der das System gespawnt/erzeugt werden soll, gemessen von dem Punkt an dem eure FXList entstanden ist. Z:20 entspricht der Höhe etwas über dem Kopf eines Helden, genau kann ich euch das allerdings nicht sagen, daher: Einfach ausprobieren ;-)

Offset kann auch weggelassen werden wenn die Position 0 0 0 gewünscht ist.

**OrientToObject = Yes**

Das System wird selbst nach dem Erzeugen noch von dem Object erzeugt, das heißt, bewegt sich euer Held während der FX läuft, wird der FX ihm Folgen, bzw. die Particel werden immer an der Position des Helden erzeugt. In Kombination mit Offset kann dann z. B. permanent vor einer Figur ein Effekt erzeugt werden.

OrientToObject kann weggelassen werden, es wird dann einfach „No“ angenommen, ich persönlich lasse sie aber immer in der List falls ich das nachträglich ändern möchte.

## **Sound**

Eine Box für Sound wird geöffnet.

### **Name = Tutorialsound**

Hier gilt selbiges wie bei der Systembox, hier wird einfach nur der Name des gewünschten Sounds angegeben. Der Sound wird zu Beginn der FXList gespielt.

Sollte kein Sound gebraucht werden könnt ihr die Box einfach weglassen.

### **ViewShake**

Eine ViewShake-Box wird geöffnet, diese Box beinhaltet ob und wie stark die Kamera des Spielers vibriert während er die FX ansieht, ich rate euch damit sparsam umzugehen.

Sollte keine Kamerabewegung gebraucht werden könnt ihr die Box einfach weglassen.

### **Type = CINE\_EXTREME**

Hier wird angegeben wie stark die Kamera vibrieren soll, dies kann man leider nicht genau einstellen sondern man muss von EA Games vordefinierte Profile benutzen.

Es gibt also folgende Möglichkeiten:

**SUBTLE, NORMAL, STRONG, SEVERE, CINE\_EXTREME, CINE\_INSANE**

Es gibt auch noch die Dynamic Decals, das sind Bilddateien die auf dem Boden angezeigt werden, so wie bei der Fähigkeit Elbenwald, doch diese werde ich aus Gründen der Übersichtlichkeit in einem anderem Tutorial erklären.

Wie man sieht ist die FXList nur eine Art Zusammenfassung welche Sound, Kamerabewegung und ein (bzw. mehrere) FXSysteme zusammenführt.

Weiter geht's auf der nächsten Seite, mit den Particlesystemen (fürchtet euch^^).

Jetzt kommen wir zu den wahren FX und damit zum schwierigsten Teil dieses Tutorials (zum einen für euch und zum anderen für mich es zu erklären :p )

Ich habe wieder eine exemplarische System genommen um euch die wichtigsten Funktionen zu erklären, die Schwierigeren sind hier noch nicht inbegriffen. Zuerst werde ich euch die bloße Bedeutung und später dann die Zusammenhänge erklären.

## System-Box

Diese Box beinhaltet alles Grundlegende, die Lebenszeiten, das Particlebild sowie Grafikeinstellungen.

## Color-Box

Diese Box enthält Farbeinstellungen der Particle.

## Update-Box

Diese Box gibt an ob die Particle beim erzeugen gedreht sind, und ob sie sich im Verlauf ihres Life drehen (und wie schnell).

## Physics-Box

Die Physics-Box enthält Parameter welche die Geschwindigkeit und Richtung der Particle beeinflussen.

## EmissionVelocity-Box

Diese Box gibt an wie und wie schnell die Partikel sich vom/zum Ursprungsort bewegen.

## EmissionVolume-Box

Die EmissionVolume-Box gibt an auf welche Art und Weise die Particle gespawnt werden. (z.B. in einem gewissen Radius um den Ursprungsort)

## Draw-Box

Gibt die Art der FX an, dies ist in diesem Tutorial nicht von Bedeutung.

```

FXParticleSystem TutorialFX1
System
    Priority = ALWAYS_RENDER
    ParticleName = EXTutorial.tga
    Lifetime = 35 75
    SystemLifetime = 2500 2600
    Size = 43 55
    BurstDelay = 5 11
    BurstCount = 1 3
    IsGroundAligned = Yes
End
Color = DefaultColor
    Color1 = R:100 G:100 B:200 0
    Color2 = R:255 G:255 B:255 25
    Color3 = R:0 G:0 B:0 75
End
Update = DefaultUpdate
    SizeRate = 2 3
    SizeRateDamping = 0.8 0.85
    AngleZ = -1 1
    AngularRateZ = 0.05 0.15
    AngularDamping = 1 1
End
Physics = DefaultPhysics
    VelocityDamping = 0.85 0.95
    Gravitiy = 0.02 0.03
    DriftVelocity = X:0 Y:0 Z:0.5
End
EmissionVelocity = OrthoEmissionVelocity
    X = 0 0.5
    Y = 0 0.5
    Z = 0 0.5
End
EmissionVolume = LineEmissionVolume
    StartPoint = X:0 Y:0 Z:5
    EndPoint = X:0 Y:0 Z:15
End
Draw = DefaultDraw
End
End

```

FXParticleSystem TutorialFX1

Befehl das eine FXParticleSystem-Box mit dem Namen TutorialFX1 geöffnet wird (bzw. hier beginnt)

## System

Beginn der System-Box, diese Box ist IMMER anzugeben.

Priority = ALWAYS\_RENDER

Dies gibt an wie wichtig der Effekt ist, damit könnt ihr einstellen ab welcher Grafikstufe der FX angezeigt wird. Weitere verwendete Bezeichnungen:

NONE, VERY\_LOW\_OR\_ABOVE, LOW\_OR\_ABOVE, MEDIUM\_OR\_ABOVE, HIGH\_OR\_ABOVE

ParticleName = EXTutorial.tga

Dies gibt an auf welchem Bild die FX beruht, eine FX besteht aus etlichen kleinen Bildern, daher ist es sehr wichtig das Richtige zu wählen. Es ist üblich eine FX-Textur mit „EXname“ zu benennen, also das die ersten beiden Buchstaben EX sind, dadurch findet man sich in der Textur-BIG besser zurecht.

Lifetime = 35 75

und SystemLifetime = 2500 2600

Diese zwei Faktoren sind sehr wichtig, eine FX besteht im Prinzip aus einer Quelle, das System, und den Eigenschaften was nach erzeugen an dieser Quelle mit den Particles passiert. Die SystemLifetime gibt uns also an wie lange unsere Quelle „lebt“, und die Lifetime gibt an wie lange die einzelnen Particel leben. Diese Zeitangaben erfolgen in Frames, wobei gilt 30 Frames sind eine Sekunde. Es werden im Normalfall bei beiden Werten zwei Ziffern angegeben, das Spiel wählt dann einen Zufallswert aus, der  $\geq$ Wert1 und  $\leq$ Wert2 ist. Bei der SystemLifetime geschieht das beim erzeugen des Systems, bei der Lifetime wird der Wert für JEDEN Particle einzeln neu per Zufall ermittelt.

ACHTUNG: Die SystemLifetime kann weggelassen werden, dann dauert der FX ewig an, dies wird unter anderem für Auren benutzt die Helden umgeben (da deren FX nur bei der Fähigkeit aktiv ist, und daher nicht durch eine SystemLifetime ausgeschaltet werden muss).

Size = 43 55

Gibt die Startgröße der Particle an. Wieder wird für JEDEN Particle einzeln, per Zufall, ein Wert bestimmt der  $\geq$ Wert1 und  $\leq$ Wert2 ist. (Wie groß dies im Spiel wirkt sollte jeder selbst herausfinden)

BurstDelay = 5 11

und BurstCount = 1 3

Zwei weitere Werte welche am besten zusammen erklärt werden. Ein Burst ist das einmalige erzeugen von Particeln innerhalb eines Systems. Dies geschieht sooft bis das System endet (also die SystemLifetime vorüber ist).

BurstDelay beschreibt dabei die Abstände zwischen den Bursts (in Frames) und BurstCount die Anzahl der Particle die pro Burst erzeugt werden.

IsGroundAligned = Yes

Bei „Yes“ bedeutet dies das die Particle parallel zum Boden ausgerichtet werden (wie bei Wort der Macht), bei „No“ (bzw. bei weglassen der Zeile) sind die Particle zur Kamera ausgerichtet (wie z. B. bei allen Explosionen)

### System-Box Zusammenfassung

Im Beispiel werden also 1 bis 3 Particle (siehe BurstCount) alle 5 bis 11 Frames (siehe BurstDelay) erzeugt und das zwischen 2500 bis 2600 Frames lang (Siehe Systemlifetime), die Particle haben anfangs jeweils eine Größe von 43 bis 55 Frames und leben zwischen 35 und 75 Frames. Die Particle sind parallel zum Boden ausgerichtet.

Color = DefaultColor

Dies öffnet eine Box namens „DefaultColor“, andere Farbboxen als DefaultColor sind nicht bekannt. (ACHTUNG: Dieser Name ist eine feste Konstante und kann nicht frei gewählt werden.) Diese Box beinhaltet die Farbinformationen über unsere Particle.

Color1 = R:100 G:100 B:200 0

Diese Zeile gibt den Particeln eine Farbe nach dem RGB System. Wobei zu beachten ist das Schwarz durchsichtig ist. R gibt den Rotanteil G den Grünanteil und B den Blauanteil an. Die Letzte Zahl gibt den Zeitpunkt (in Frames) an, an dem die Particle die Farbe erhalten, gemessen an der Lifetime des Particles nicht der SystemLifetime.

Man kann die Farbe nun wechseln lassen, indem man einfach eine Zeile namens „Color2 =...“ mit den gewünschten Farbwerten und Zeitpunkt angibt (selber Aufbau wie Color1, siehe Beispiel oben). Dadurch werden Farbwechsel ganze einfach möglich.

Update = DefaultUpdate

Dies öffnet eine Box namens DefaultUpdate, andere UpdateBoxen sind nicht bekannt (ACHTUNG: dieser Name ist ebenfalls eine feste Konstante und kann nicht frei gewählt werden).

SizeRate = 2 3

und SizeRateDamping = 0.8 0.85

SizeRate gibt euch an wie schnell sich die Particle ausbreiten, nicht wie groß sie am Ende ihrer Lifetime sind.

SizeRateDamping, dies ist gedacht um eine bessere Kontrolle über „SizeRate“ zu haben, da bei SizeRate nur ganze Zahlen benutzt werden können ist es Relativ schwer einen passenden Wert zu finden. SizeRateDamping ist ein Multiplikator von SizeRate, es gilt also:

SizeRate \* SizeRateDamping = „Wahre Vergrößerungsgeschwindigkeit“

Da ihr bei beiden Zufallswerte angeben können innerhalb eines Systems Particle mit sehr unterschiedlichen Vergrößerungsverhalten erzeugt werden. Negative Werte lassen eure Particle schrumpfen. Meine Theorie zur Ausbreitung der Particle ist wie folgt:

SizeRate = 2 2 und Damping = 1 1, bewirken das sich der Durchmesser des Particles um 20 20 in einer Sekunde (also 30 Frames) erweitert.

AngleZ = -1 1

Hier wird angegeben um wie viele Grad die Particle gedreht sind, wenn sie erzeugt werden.

Das sagt nichts über die Drehung während des Lifes aus. Es sind Werte von 0 bis 360 möglich, -1 würde also 359 bedeuten. Mit den Werten -1 1 wird also erreicht das die Particle um 0 bis 359 Grad gedreht, erzeugt werden (also die größtmögliche Anzahl an Möglichkeiten), ich würde daher für die meisten Effekt diese Werte empfehlen.

AngularRateZ = 0.05 0.15

und AngularDamping = 1 1

Dies gibt an wie schnell sich die Particle während ihres Life drehen. Eine positive Zahl veranlasst sie Links herum zu drehen eine negative rechts herum zu drehen. Wieder wird für jeden einzelnen Particle eine Zufällige Zahl zwischen den beiden angegeben ausgewählt.

AngularDamping ist ein Multiplikator zu AngularRateZ, so wie SizeRateDamping zu SizeRate, es gilt also Folgendes:

AngularRateZ \* AngularDamping = „Wahre Drehgeschwindigkeit+Richtung“

## Update-Box Zusammenfassung

Im Beispiel werden die Particle also in einem beliebigem Winkel erzeugt, drehen sich mit zwischen 0.05 und 0.15 schnell und breiten sich mit einer Geschwindigkeit zwischen 1.6 und 2.55 aus.

Physics = DefaultPhysics

Öffnet eine Physics-Box (darf nicht weggelassen werden, muss allerdings nichts beinhalten). Der Name ist auch hier eine feste Konstante und kann nicht gewählt werden (es sind auch keine anderen bekannt)

VelocityDamping = 0.85 0.95

Dieser Parameter gibt die Beschleunigung der Particle an. 1 wäre gleich bleibend, eine Zahl größer 1 wäre stetige Beschleunigung und eine unter 1 ein stetiges abbremmen.

Gravity = 0.02 0.03

Dies ist die Gravitation des Partikels, eine Positive Zahl lässt den Partikel permanent steigen, eine negative permanent sinken. Es ist ratsam sehr kleine Zahlen zu verwenden, da dieser Parameter dazu neigt sehr schnell zu sein.

DriftVelocity = X:0 Y:0 Z:0.5

Hiermit wird angegeben in welche Richtung sich die Particle bewegen, dies gilt für ALLE Particle GLEICH, das heißt alle Particle würden sich jetzt auf der Z-Achse gleich schnell bewegen (dies geschieht zusätzlich aber unabhängig von den anderen Beschleunigungsfaktoren).

### Physics-Box Zusammenfassung

Im Beispiel werden also alle Particle permanent um 0.85 oder 0.95 (bzw. ein Wert dazwischen) abgebremst, das heißt immer langsamer. Die Partikel steigen ständig in der Höhe (Gravity ist 0.02 oder 0.03). Außerdem bewegen sich ALLE Partikel auf der Z-Achse mit der Geschwindigkeit (+)0.5

EmissionVelocity = OrthoEmissionVelocity

Eröffnet eine EmissionVelocity-Box mit dem Wert „OrthoEmissionVelocity“. Es gibt allerdings auch noch andere EmissionVelocity-Boxen. Daher Folgen jetzt die Erklärungen der einzelnen Typen nacheinander (es kann immer nur eine EmissionVelocity-Box in einem System sein).

OrthoEmissionVelocity

Dieser Typ enthält schlichte Richtungsangaben allerdings mit Random-Zahlen, das Schema ist wie folgt:

*EmissionVelocity = OrthoEmissionVelocity*

*X = 0.1 0.5*

*Y = 0.1 0.5*

*Z = 0.1 0.5*

*End*

Es können alle möglichen Werte hergenommen werden, dies müssen nicht die selben auf den drei Achsen sein. Es können auch negative Zahlen verwendet werden.

CylindricalEmissionVelocity

Hier werden die Partikel vom Zentrum aus Radial und Normal bewegt. Versucht euch das so vorzustellen: Die Quelle der Partikel ist das Zentrum von vielen konzentrischen Kreisen.

Radial = -1 1

Sagt den Partikeln dann das sie auf diesen Kreisen links bzw. rechts entlang wandern sollen.

Normal = 1 1

sagt den Partikeln das sie sich vom Zentrum entfernen sollen (auch negative Zahlen möglich)

*EmissionVelocity = CylindricalEmissionVelocity*

*Radial = -1 1*

*Normal = 1 1*

*End*



## OutwardEmissionVelocity

Diese (meine Lieblings-)Variante sendet die Partikel einfach vom Zentrum weg, stellt es euch einfach so vor als würden viele Strahlen vom Zentrum weggehen auf denen die Particle entlangwandern, Speed gibt dann nur noch die Geschwindigkeit an.

EmissionVelocity = OutwardEmissionVelocity

Speed = 1 2

End

Das waren alle bekannten Möglichkeiten des EmissionVelocity-Blocks.

## EmissionVolume = LineEmissionVolume

Diese Zeile öffnet einen EmissionVolume-Block mit „LineEmissionVolume“ als Eigenschaft. Der EmissionVelocity-Block gibt an in welchem Gebiet die Particle gespawnt werden. Auch hier gibt es mehrere fest definierte Möglichkeiten, die ich nun auflisten und erklären werde.

### PointEmissionVolume

Die Particle werden schlicht und einfach an dem Ursprungspunkt gespawnt. Es gibt den Parameter IsHollow = Yes/No, dieser ist allerdings nutzlos (es wäre mir noch kein Unterschied aufgefallen), die Box kann daher einfach leer gelassen werden.

EmissionVolume = PointEmissionVolume

End

### LineEmissionVolume

Die Particle werden auf einer Linie gespawnt (findet z.B. als Effekt für Waffen Anwendung). Die Linie wird mithilfe von einem Startpunkt (StartPoint) und einem Endpunkt (EndPoint) definiert.

EmissionVolume = LineEmissionVolume

StartPoint = X: 1 Y:1 Z:1

EndPoint = X:1.2 Y:2 Z:3

End

### BoxEmissionVolume

Hier werden die Particle, orientiert an einem Quader, gespawnt. Neben der Angabe der Größe über „Halfsize“, kann man auch den Parameter „IsHollow“ verwenden. WICHTIG: „Halfsize“ gibt wie der Name schon sagt nur die HÄLFTE der Breite/Höhe/Tiefe an, nicht die Vollständige.

Wird „IsHollow“ als Yes definiert, werden die Particle statt IN dem Quader, AN DER „HÜLLE“ des Quaders gespawnt. Wird „IsHollow“ weggelassen wird es als „No“ definiert.

EmissionVolume = BoxEmissionVolume

Halfsize = X:4 Y:15 Z:1

IsHollow = Yes

End

### CylinderEmissionVolume

Hier werden die Particle, orientiert an einem Zylinder/einer Röhre, gespawnt. Neben der Höhe (Length) und dem Radius sind auch noch die drei Parameter „IsHollow“, „Offset“ und „RadiusRate“ verfügbar. „IsHollow“ bewirkt wieder das die Particle an der Hülle entlang gespawnt werden. Mit „Offset“ lässt sich der Mittelpunkt des Zylinders vom Ursprungsort aus angeben. „RadiusRate“ ist eine außerordentlich nützliche Funktion mit der der Radius stetig erhöht wird während das System läuft (der Radius wird also immer größer).

Für Beispiel siehe nächste Seite.

*EmissionVolume = CylinderEmissionVolume*

*Length = 80*

*Radius = 40*

*RadiusRate = 1 2*

*End*

*SphereEmissionVolume*

Hier werden die Particle, orientiert an einer Sphäre/Kugel, gespawnt. Es gibt zwei Werte: „Radius“ und „IsHollow“. „IsHollow“ bewirkt das die Partikel an der Hülle entlang gespawnt werden, anstatt innerhalb der Kugel.

*EmissionVolume = SphereEmissionVolume*

*Radius = 35*

*IsHollow = No*

*End*

Dies waren alle EmissionVolume-Typen.

Draw = DefaultDraw

Diese Zeile ist noch nicht relevant, da sie nur übernommen werden muss, wir benötigen keinerlei Änderungen für Particle-FX.

## ***Hinweise und Informationen:***

### **Zufallswerte**

Immer wenn zwei Werte für eine Sache angegeben werden können wird ein Zufallswert für jeden einzelnen Partikel ermittelt.

Wobei gilt:

Zufallswert >= Kleinere Zahl

und zugleich:

Zufallswert <= Größere Zahl

### **Spielabstürze**

Solltet ihr eine FX (um)geschrieben haben, und das Spiel lässt sich nicht mehr starten („Funktioniert nicht mehr“) dann habt ihr höchstwahrscheinlich einen Tippfehler in den FX, denn FX-Fehler bekommen zu 90% keine Fehlermeldung. Häufig ist auch der Fehler, das statt zwei Zahlen (wie z.B. bei SystemLifetime) nur einer angegeben wird.

### **Fehlende/Ausgelassene Elemente und Parameter**

In diesem Tutorial wurden bei weitem nicht alle Teile der FX-Erstellung behandelt, ich habe mehrere Teile ausgelassen, da ich denke ihr solltet zuerst das hier lernen bevor ich euch in tieferes Gewässer wagen könnt.

### **Wie geht's weiter?**

Ich werde noch ein paar FX-Tutorials veröffentlichen, über Alphakanal, Wind und Erweiterte Zusammenhänge.

Außerdem werde ich noch Übungsaufgaben zum Download bereitstellen, mit ausführlichen Hilfestellungen und langsamen antasten an das Thema FX.

**Du kannst keine FX aufgrund dieser Erklärung erstellen?**

Es gibt separat hierzu noch Übungsaufgaben, die dir das erstellen der FX Schritt für Schritt beibringen. Dies hier ist nur eine pure aber wichtige Begriffserklärung, damit auch der Grundstein für die Übungsaufgaben gelegt ist.

**Vorgehensweise beim Schreiben**

Am besten fängst du an mit deinem EmissionVolume, wie willst du die Particle am Anfang haben? Dann weiter mit dem EmissionVelocity-Block, dann Physics und System.

Ich gebe keine Garantie für meine Erklärungen, ich habe das alles aus dem Kopf geschrieben und daher meine Vermutungen zur Funktionsweise der Parameter dokumentiert, doch im Normalfall sollte das alles seine Richtigkeit haben. Bei Fragen zu dem Thema, weiteren Tutorials oder weil ihr Hilfe bei speziellen FX benötigt, könnt ihr euch gerne an mich wenden.

Grüße  
Alien aka Infiltrator

FX-Pionier und Schöpfer von Stylize.de.tc