Grundlagen des UnrealDevelopmentKit

Einleitung und Editor
Geometry und Lighting
StaticMeshes und Materials
Kismet
Particles

UNIVERSITÄT KOBLENZ · LANDAU

Wintersemester 13/14
Tutorium von Raphael Menges

Mittwochs, 18-20 Uhr in **F230**. Vom **06.11**.13 bis **04.12**.13

Vorwort

- Grundlagenveranstaltung, also nur geringe Vorkenntnisse notwendig
- Content wurde erstellt mit Blender, Gimp, MapZone2 und UDK
- Zu dieser Veranstaltung und anderen Projekten von mir:

  http://userpages.uni-koblenz.de/~raphaelmenges/

Präsentiert von…

- **Arbeitsbereich Digitale Medien**
  - Leitung: Dr. Markus Lohoff

Inhalt

1. Einleitung und Editor
2. Geometry und Lighting
3. StaticMeshes und Materials
4. Kismet
5. Particles

Einleitung

- Geschichte der Unreal-Engine
- Was ist das UDK?
- Vor- und Nachteile des UDK
- Was lerne ich in diesem Tutorium?
- Wie läuft das Ganze ab?
- Kann der da vorne das?
- Woher bekomme ich das UDK?

Geschichte der Unreal-Engine

Version	Editoren (für den End-User)	Spiele (nur einige)
Unreal-Engine 1 (~1998)	UED 1 (Unreal) UED 2 (Unreal Tournament)	Unreal Unreal Tournament Deus Ex
Unreal-Engine 2.x (~2002)	UED 3 (Unreal Tournament 2003/2004)	Unreal2 Unreal Tournament 2003/2004 XIII Duke Nukem Forever
Unreal-Engine 3.x (~2005)	UED 4 (Unreal Tournament 3) UDK-Editor (eigenständig)	Unreal Tournament 3 BioShock MassEffect Gears of War Borderlands …
Unreal-Engine 4 (2012)	UDK ?	Fortnite

Geschichte der Unreal-Engine

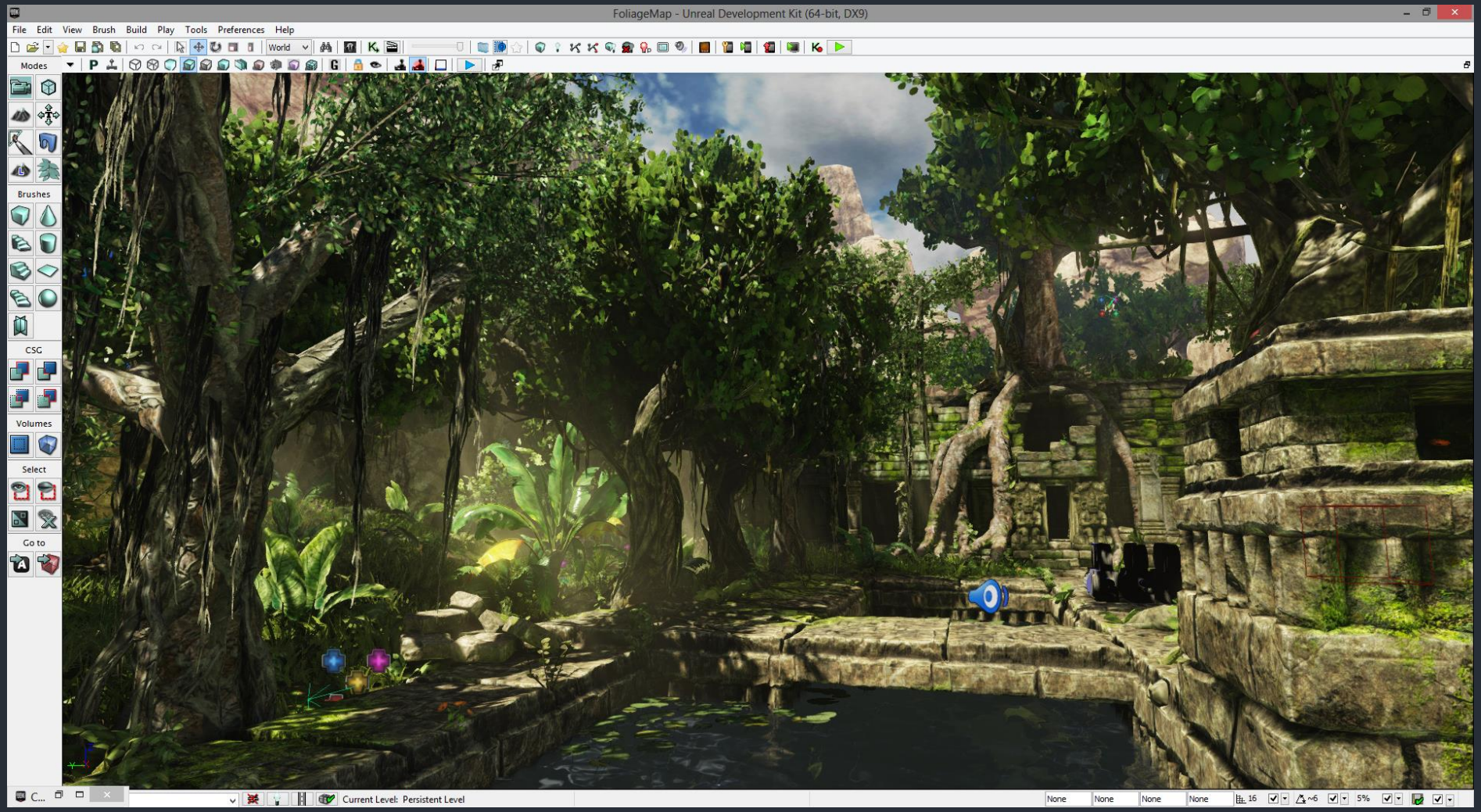
Was ist das UDK?

- Das **UDK** ist eine **Sammlung von Werkzeugen** zum **Erstellen** und **Zusammensetzen** eines virtuellen Raumes und der **Definition von Interaktion** mit diesem
- Setzt auf die **Unreal-Engine 3.5**
- Wird von **Epic Games** entwickelt


UNREAL
TECHNOLOGY


EPIC
GAMES


UDK
UNREAL
DEVELOPMENT KIT


FoliageMap - Unreal Development Kit (64-bit, DX9)
File Edit View Brush Build Play Tools Preferences Help
Modes
Brushes
CSG
Volumes
Select
Go to
Current Level: Persistent Level

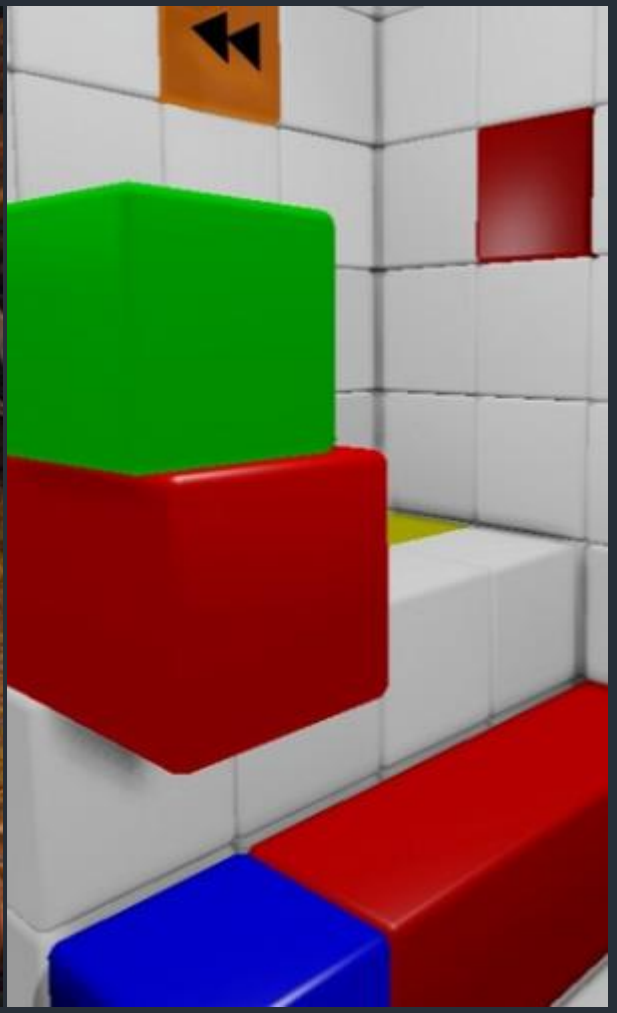
Takenaka

Samaritan

The Ball

Hawken

Chivalry



Q.U.B.E.

Material

Kismet

StaticMesh

SkeletalMesh

Cascade

K.I.

Terrain

Landscape

Lightmass

ActorClassBrowser

Material

Kismet

StaticMesh

SkeletalMesh

Cascade

K.I.

~~Terrain~~

Landscape

Lightmass

ActorClassBrowser

Vor- und Nachteile des UDK

Vorteile	Nachteile
▪ Aktuelle Rendering-Engine ▪ Einfache aber mächtige Werkzeuge ▪ Relative gute Dokumentation ▪ Große Community ▪ Direkte Anbindung an Autodesk-Produkte ▪ Läuft auf nahezu allem (zumindest die Unreal-Engine an sich) ▪ Bringt auch Team-Werkzeuge mit ▪ Billig (Frei für non-commercial, 99$ + ab 50.000$ 25%)	▪ UDK-Editor selbst läuft nur auf Windows-PC; Runtimes können für PC, MacOS und mit richtiger Lizenz für iOS gepackt werden ▪ Einige Funktionen nur mit professioneller Software nutzbar, wie z.B. UI ▪ Einige Funktionen doch eher lasch dokumentiert

Was lerne ich in diesem Tutorium?

- Grundlegenden Einblick wie alles zusammenarbeitet
- Erstellen einer Welt mit internen Geometry-Tools
- Ausleuchtung der Welt
- Import von Texturen und Modellen
- Erstellen von Materialien und Partikelsystemen
- Visuelles Scripting mit Kismet

Wie läuft das Ganze ab?

- Die Werkzeuge und Techniken werden Schritt für Schritt erst im Vortrag erklärt und dann aktiv im UDK benutzt
- Alle Werkzeuge und Techniken werden an einem großen Beispiel gezeigt, sie sind passend zum Workflow geordnet
- Für jede Session gibt es eine fertige Welt, sodass ihr ein mögliches Endprodukt mit allen relevanten Inhalten anschauen könnt

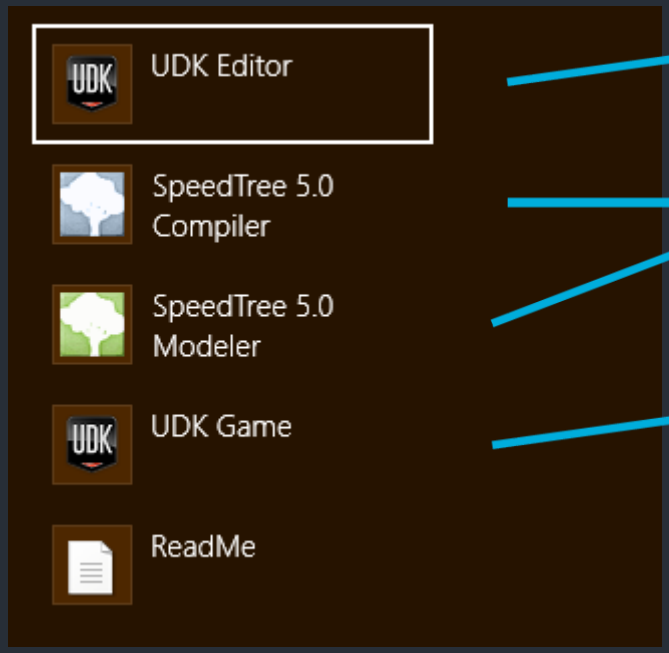
Kann der da vorne das?

- Bis 2008 einige (eher schlechte) Maps für Unreal Tournament 2003/2004/3
- 2008 erste Total Conversion für Unreal Tournament 3 (Team-Position: Artist)
- 2009 Fortsetzung (Team-Position: Artist) → Einzug ins Finale des „Make Something Unreal"-Contest

http://www.moddb.com/mods/tre-last-life

Woher bekomme ich das UDK?

- http://www.unrealengine.com/en/udk/



Editor zum Erstellen/Bearbeiten der Welten (den Benutzen wir)

Software für Pflanzenmodelle (irrelevant)

Hier mit kann man Welten laden und „begehen". Alternative zum Erstellen einer extra Runtime

Editor

- Grundlegende Datenstruktur
- Das Editor-GUI
- Viewports
- Der Content Browser
- Die Steuerung im 3D-Raum
- Das Widget
- Die Steuerung im 2D-Raum
- Unreal Units
- Koordinatensystem
- World Properties

Grundlegende Datenstruktur

- Daten werden in zwei verschiedenen Formaten gespeichert: .udk und .upk
- .upk: Alle Arten von erstelltem (Materials, ParticleSystems…) und importiertem (Textures, Meshes..) Content
- .udk: Die Welten an sich (Positionen von Objekten, Geometry…) und eng verknüpfter Content (Lights, Kismet, Landscape…)

BTW: Im UDK heißt eine Welt einfach „Map“

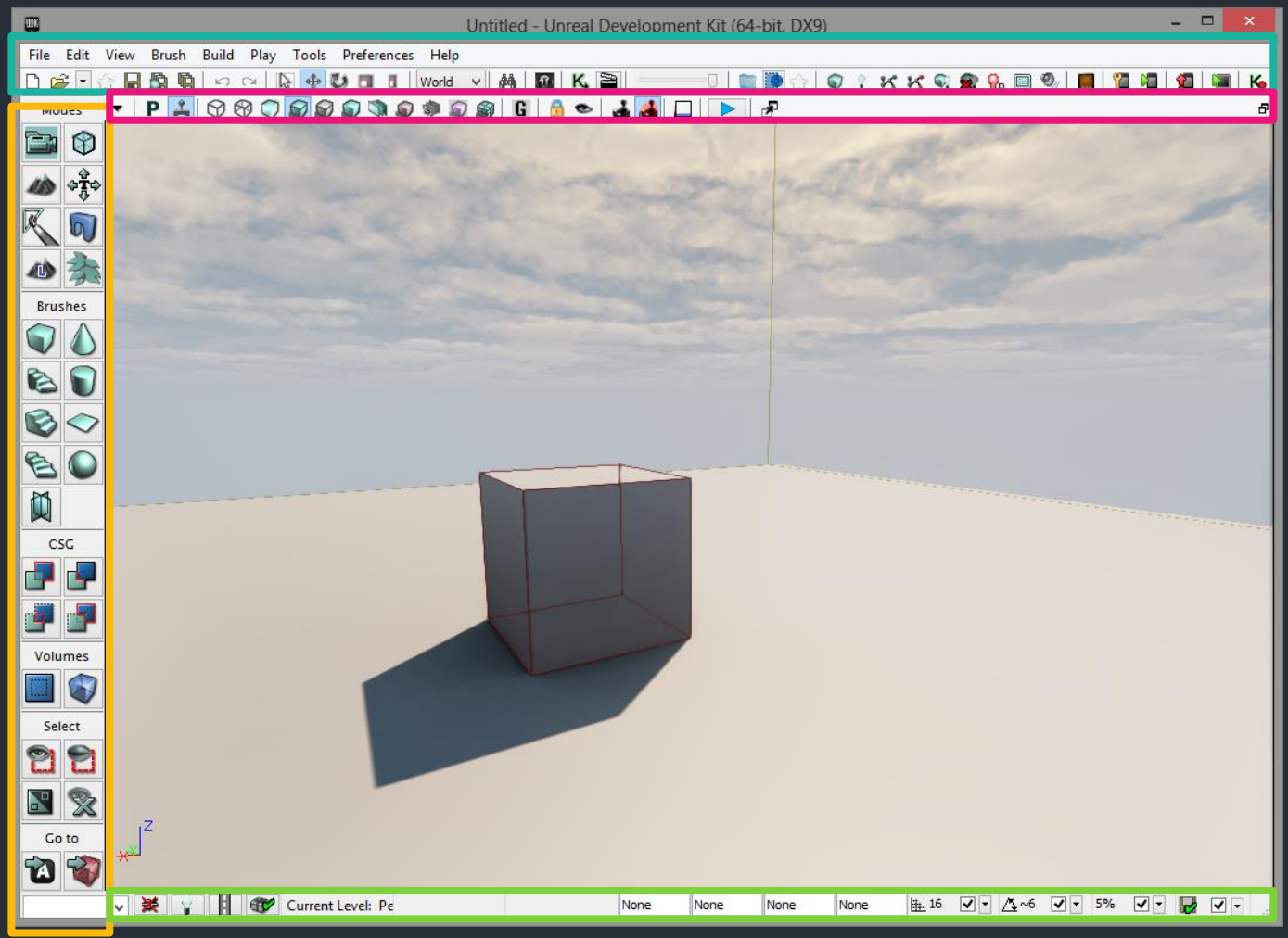
Das Editor-GUI

Menübar
→ alles, was die Welt als ganzes betrifft. Außerdem noch Einstellungen für das Widget

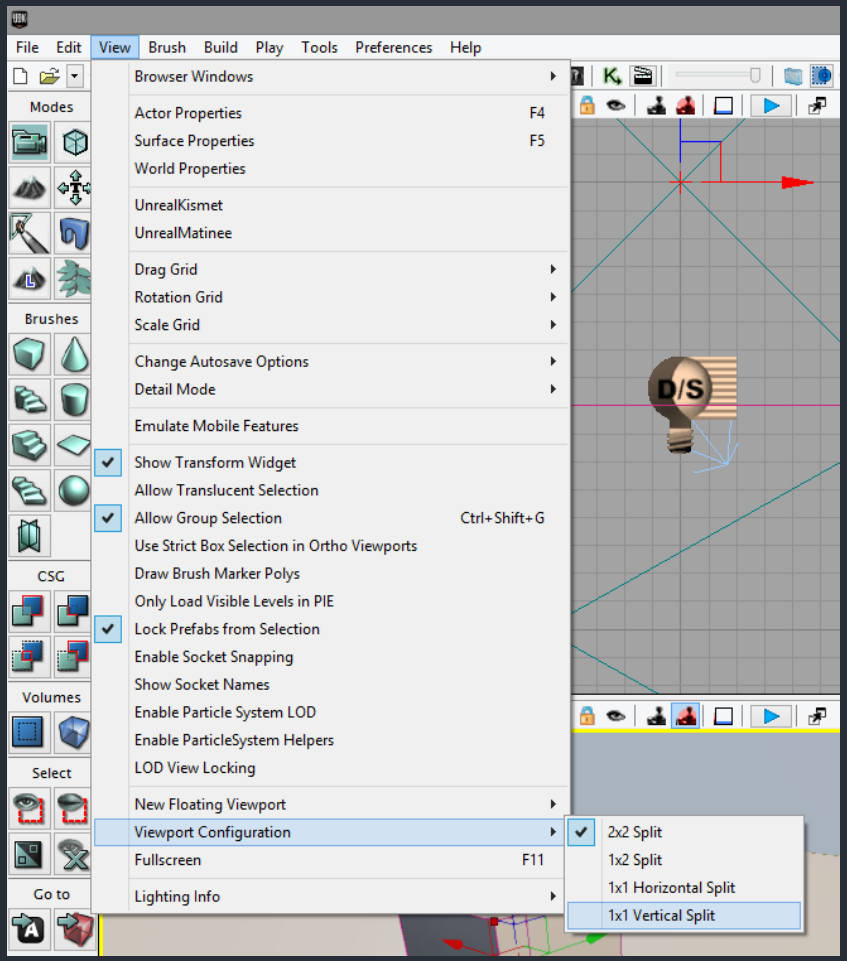
Toolbar
→ Aktuellen Modus wechseln, Geometry erstellen, Objekte Auswählen

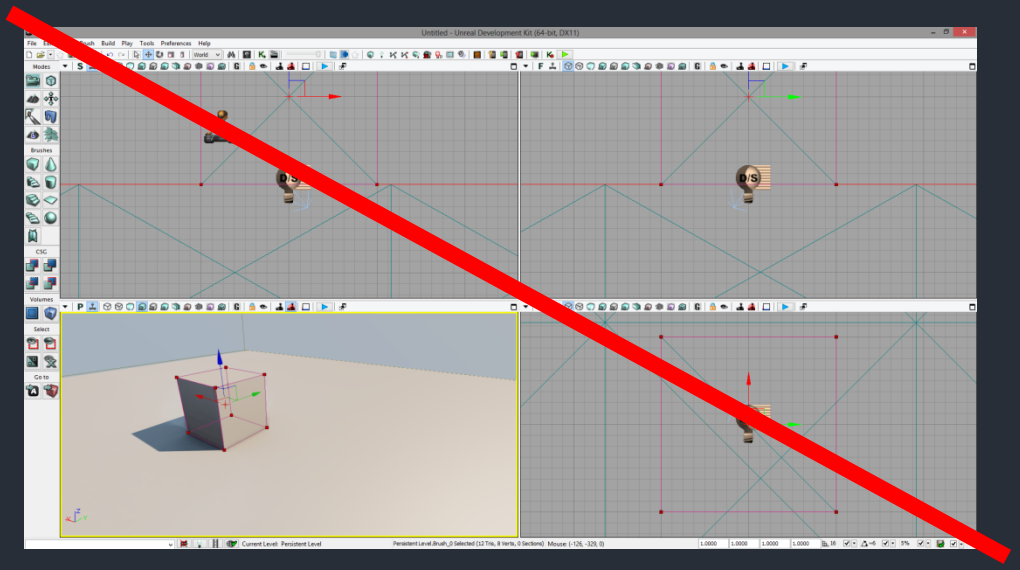
Viewportbar
→ Shading usw.

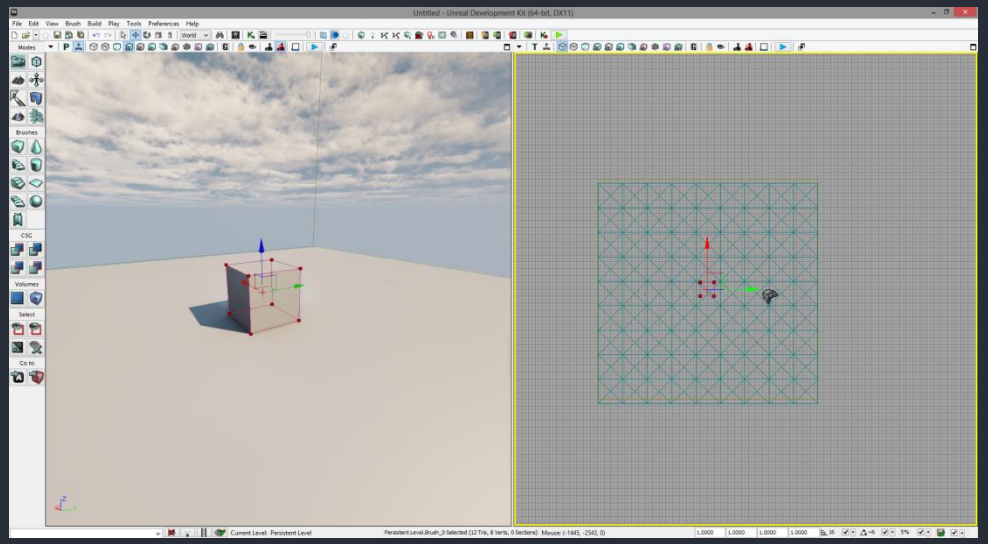
Statusbar ?
→ Grideinstellung usw.



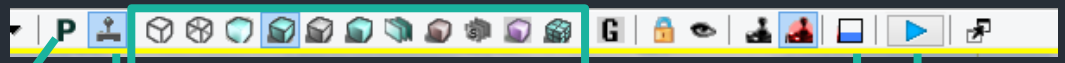
Viewports


File
Edit
View
Brush
Build
Play
Tools
Preferences
Help
Modes
Brushes
CSG
Volumes
Select
Go to
Browser Windows
Actor Properties F4
Surface Properties F5
World Properties
UnrealKismet
UnrealMatinee
Drag Grid
Rotation Grid
Scale Grid
Change Autosave Options
Detail Mode
Emulate Mobile Features
Show Transform Widget
Allow Translucent Selection
Allow Group Selection Ctrl+Shift+G
Use Strict Box Selection in Ortho Viewports
Draw Brush Marker Polys
Only Load Visible Levels in PIE
Lock Prefabs from Selection
Enable Socket Snapping
Show Socket Names
Enable Particle System LOD
Enable ParticleSystem Helpers
LOD View Locking
New Floating Viewport
Viewport Configuration
Fullscreen F11
Lighting Info
2x2 Split
1x2 Split
1x1 Horizontal Split
1x1 Vertical Split
D/S





Viewports



Viewport-Type (hier gerade Perspective, also 3D-Raum. Bei allen anderen Ansichten handelt es sich um 2D-Ansichten)

Realtime (Partikel werden emittiert usw.)

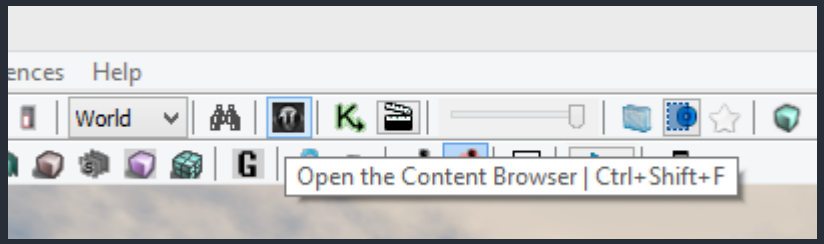
Shading im Viewport

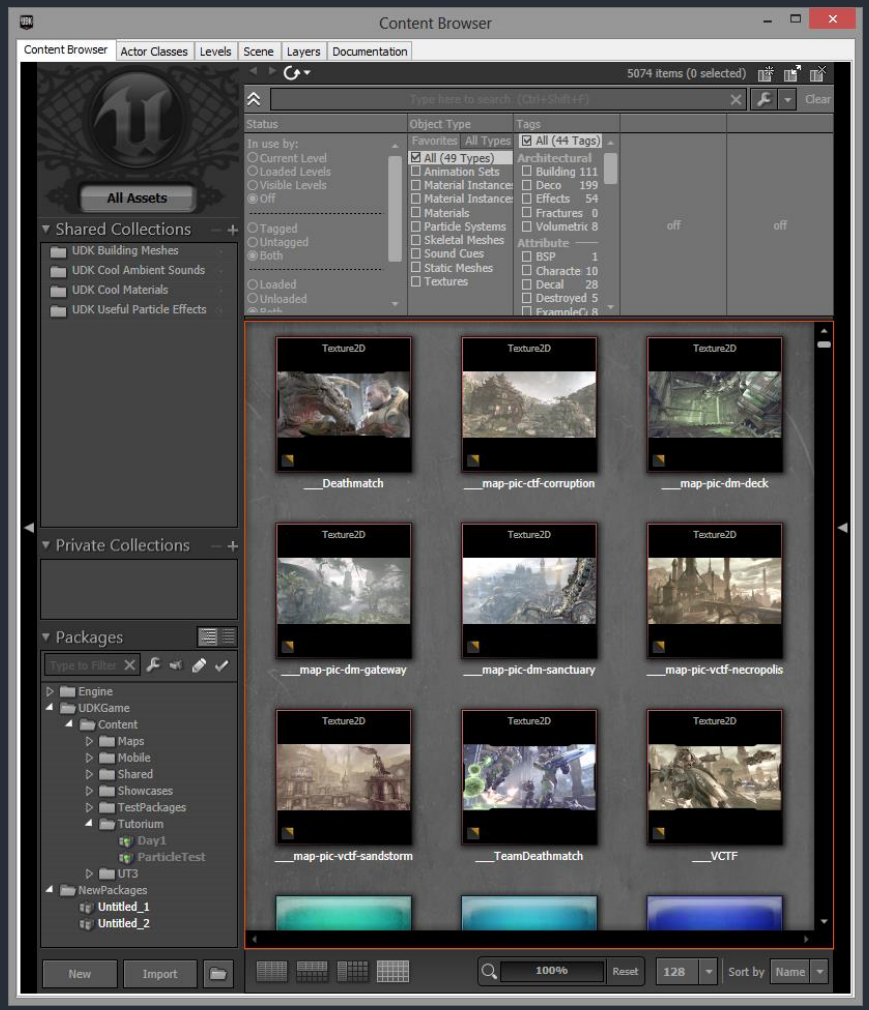
Camera Speed

Play here!

Der Content Browser

- Dient zur Anzeige von Content aus Packages und deren Verwaltung (mehr dazu bei Benutzung)



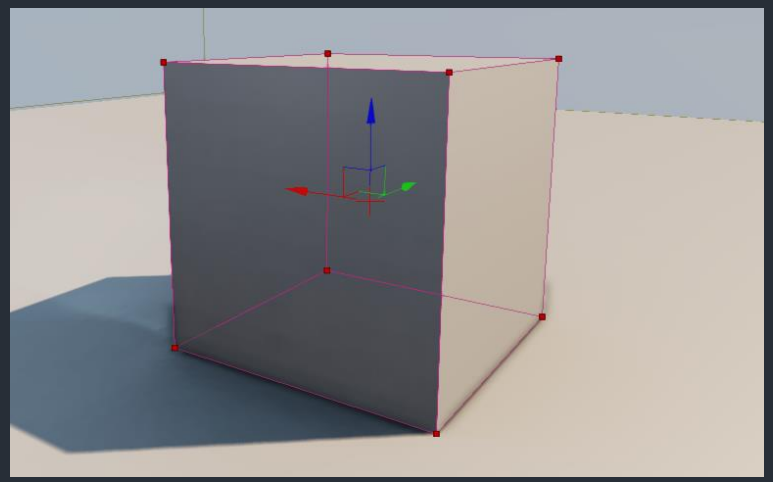


Die Steuerung im 3D-Raum

- Für Anfänger
  - Viewport mit „Reinklicken“ markieren (gelber Rahmen) und WASD für Bewegung
  - Rechte Maustaste gedrückt halten und Maus bewegen für Drehen
- Für Fortgeschrittene
  - Linke Maustaste halten für Vor- , Zurückbewegen und Drehen
  - Rechte Maustaste halten für Drehen
  - Beide Maustasten für Bewegung nach links/rechts/oben/unten

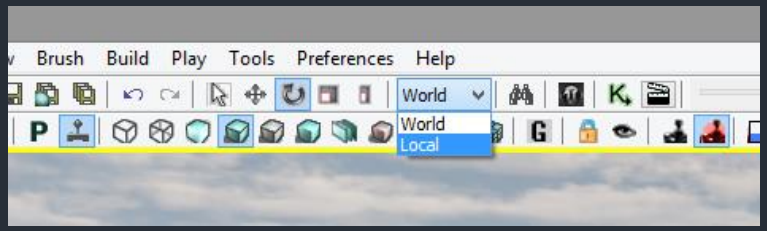
Das Widget

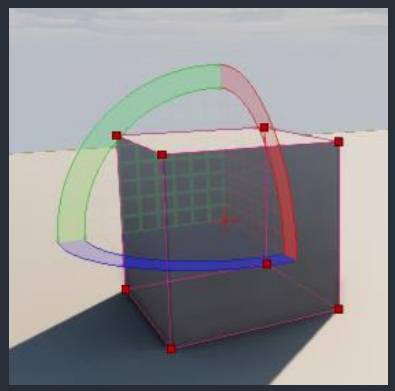
- Objekte (Actors) in der Welt werden mit Linksklick markiert
- Es erscheint das Widget, sodass man dieses mit Linker Maustaste hält und zieht und damit den Actor bewegt

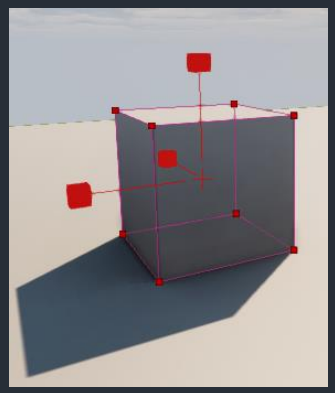


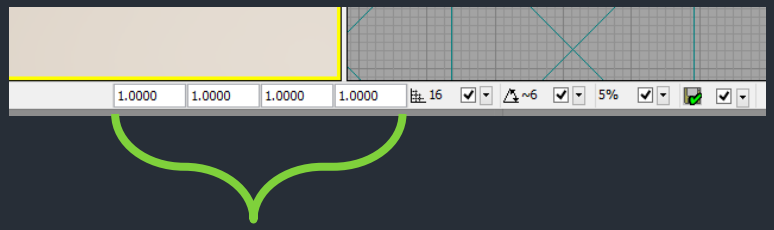
Das Widget

- Neben Bewegen kann das Widget noch rotieren und skalieren. Dazu entweder in der Menüleiste oben eine Auswahl treffen oder mit Leertaste durchschalten
- Unten rechts in der Statusbar kann das Grid, in dem das Widget einrastet, eingestellt werden









Skalierung des markierten Objektes

Die Steuerung im 2D-Raum

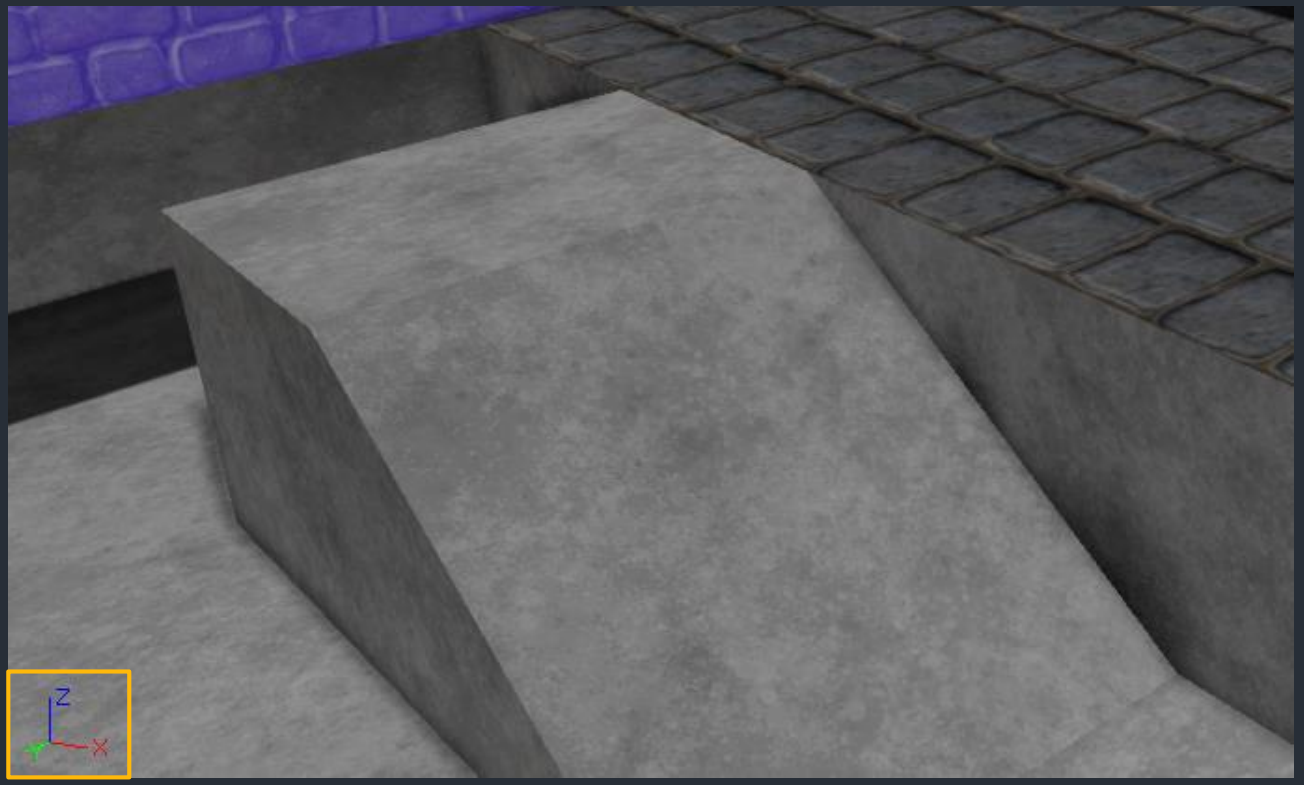
- Linke oder Rechte Maustaste halten zum Bewegen
- Beide Maustasten halten oder Mausrad für Zoomen

Unreal Units

- Die Position von Objekten wird in UU gespeichert
- Laut UDN
  - Bei Unreal Tournament 3 ist 1 UU == 2 cm
  - Bei Gears of War sind 2 UU == 1 Inch == 2.54 cm
  - Die meisten Lizenzbenutzer machen 1 UU == 1 cm
- Am besten immer in 2er-Potenzen denken (vor allem später bei der Geometry), da Werte welche aus 2er Potenzen zusammengesetzt sind gut gespeichert werden können und ans Grid passen

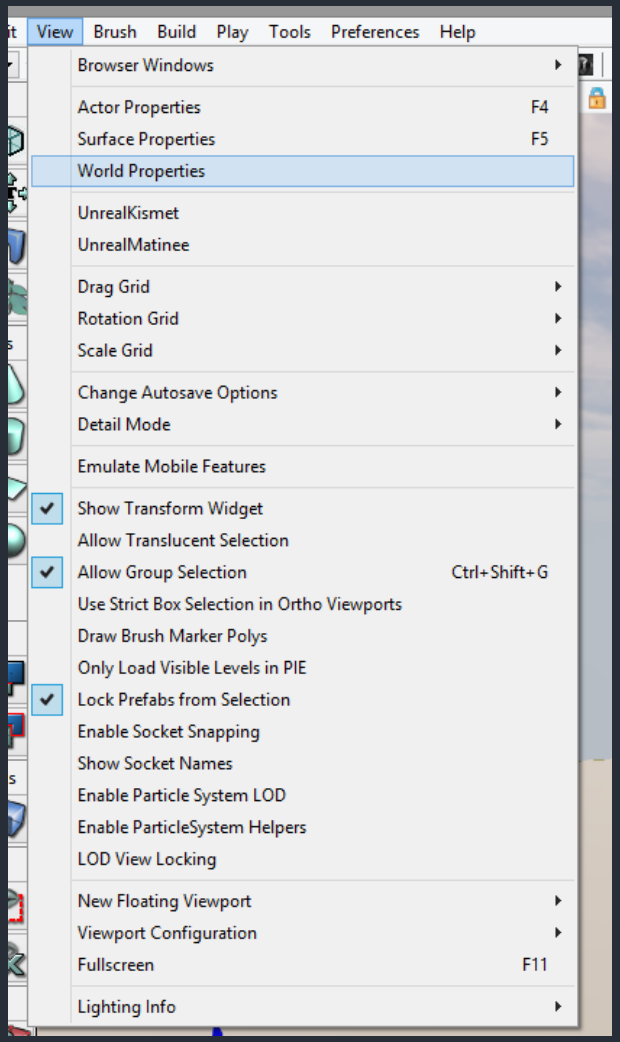
Koordinatensystem

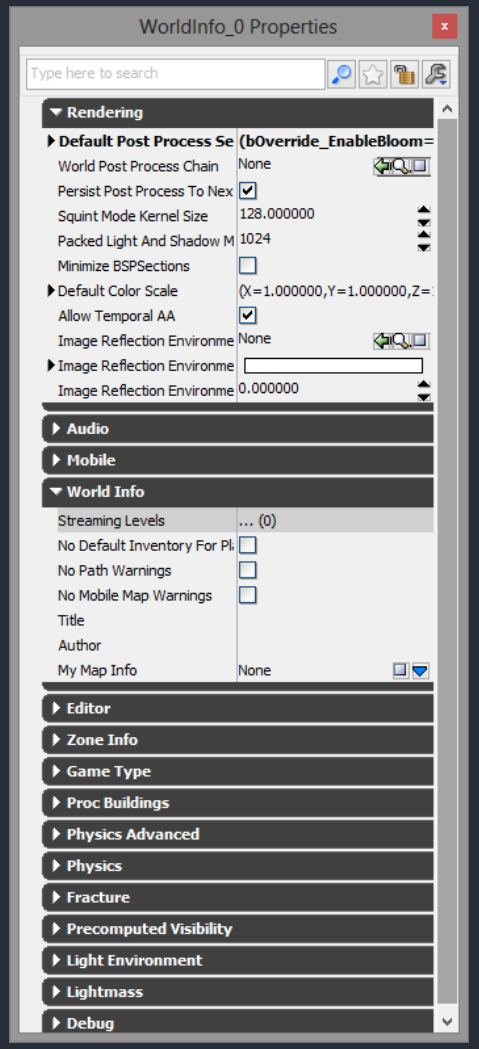
- In der Unreal-Engine zeigt die Z-Achse nach oben und die Y-Achse nach vorne



World Properties

- Anzeigename
- PostProcessing
- Game Type
- Autor





Hausaufgabe

Was stellt dieses Icon dar?

Bemerkungen

- Bei Benutzung bestimmter Tools stürzt das UDK gerne mal ab. Ist halt so
- Content/Packages/Maps niemals gleich benennen. Auch nicht einem Package den gleichen Namen wir einer Map geben
- Welten werden in der Unreal-Engine als „Map“ bezeichnet
- Alle Objekte in der Welt sind „Actors“, also programmiertechnisch eine Spezialisierung von Actor

Resources

- Offizielle Dokumentation: http://udn.epicgames.com/Three/WebHome.html
- Offizielles Forum: http://forums.epicgames.com/
- Gute Tutorials: http://www.hourences.com/
- Größtes deutsches Forum: http://www.unrealed.info/forum/

Inhalt

1. Einleitung und Editor
2. Geometry und Lighting
3. StaticMeshes und Materials
4. Kismet
5. Particles

In „AlphaMap.udk“ ist Geometry und Lighting fertig und kann als Referenz genommen werden.

Installation der Beispiele

- Aus der .zip den Ordner „Tutorium“ in das UDK Verzeichnis verschieben (…\UDK-2012-07\UDKGame\Content\Tutorium)
- Wenn ihr auf den lokalen Rechnern arbeitet, wird alles nur lokal dort gespeichert. Wenn ihr eure Daten „retten“ wollt, bitte auf USB-Stick kopieren oder sonstiges

Geometry (aka **B**inary **S**pace **P**artition aka **C**onstructive **S**olid **G**eometry)

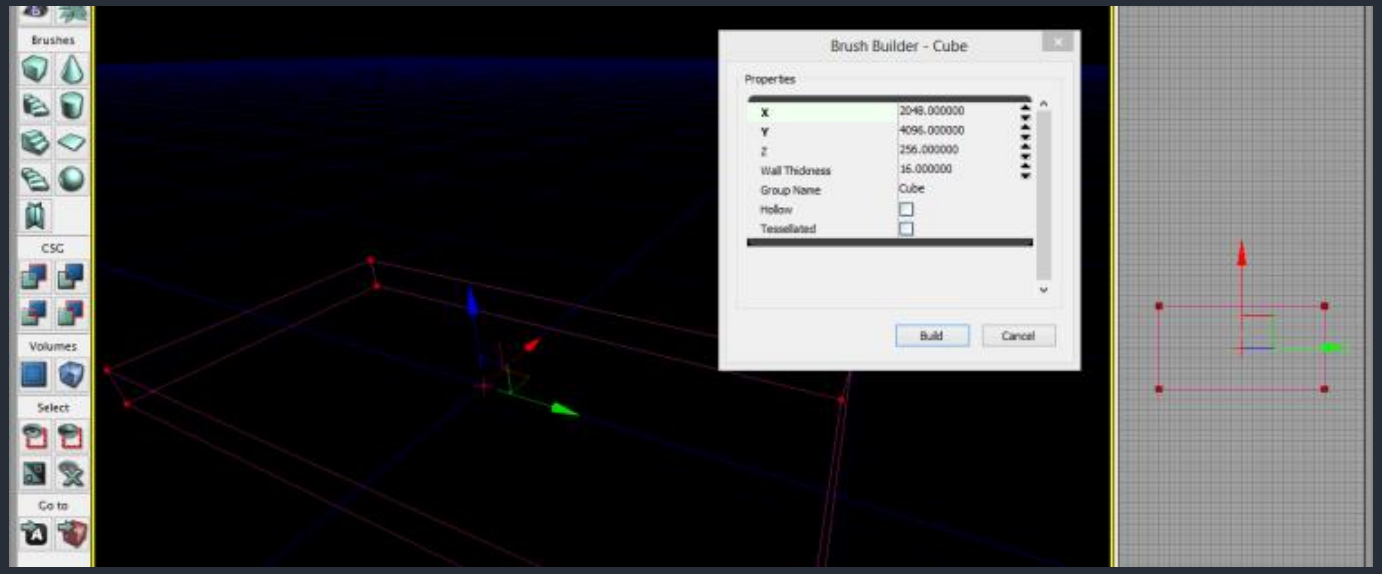
- Geometry in der Unreal-Engine
- Builder Brush
- Add/Subtract
- Geometry Tools
- Surface Properties
- Materialzuweisung

Geometry in der Unreal-Engine

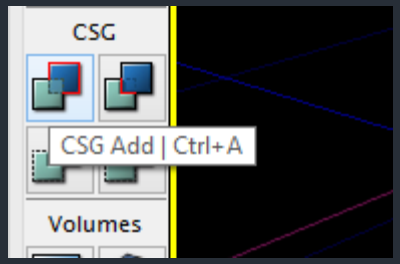
- Seit dem ersten Unreal-Editor bietet dieser die Möglichkeit mit internen Werkzeugen einfache geometrische Formen zu einer Welt zusammenzusetzen
- In der Unreal-Engine 3 ist der Raum dabei am Anfang leer
- Für die Erstellung wird ein „Builder Brush“ benutzt, der sozusagen als Form dient
- Alternative zu StaticMeshes, welche ungefähr die gleiche Funktion haben, aber extern erstellt werden

Builder Brush

- Der Builder Brush (die rote Form) kann per Linksklick markiert und mit dem Widget bewegt werden
- In der Toolbar unter „Brushes“ können ihm neue Formen zugewiesen werden
- Um die neue Form zu definieren, Rechtsklick auf eine Form in der Toolbar



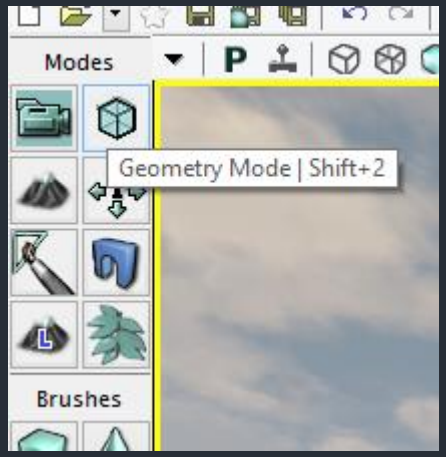
Add/Subtract

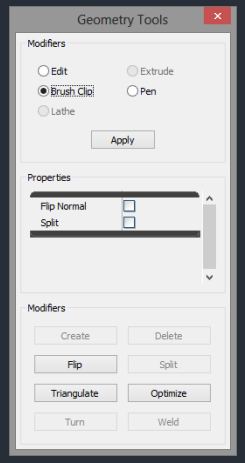


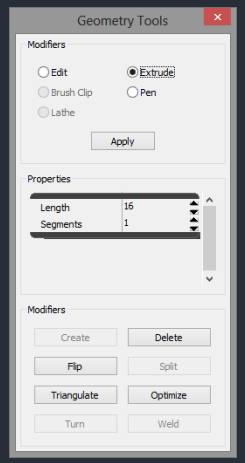
- Mit dem Builder Brush kann nun Materie erschaffen (Add) oder abgezogen (Subtract) werden
- Dadurch entsteht ein Brush Actor, den man nur im Brush Wireframe-Modus sehen kann. Dabei sind additive Brushes blau und subtraktive orange.

- Bestehende Brushes können im Wireframe-Modus selektiert und transformiert werden
- Wenn ein gesetzter Brush nachträglich verschoben wird, wird die Materie erst nach einem „Rebuild“ sichtbar (dazu später mehr)

Geometry Tools

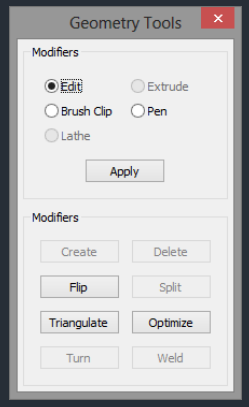


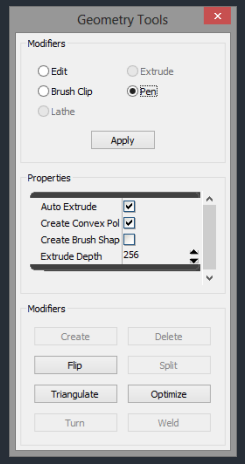




- Brush Clip
  - Sozusagen ein Schneidewerkzeug für Brushes
  - Wieder per STRG+Rechtsklick in nicht-perspektivischem Viewport
  - Den Builder Brush kann man zurechtschneiden
  - Bestehende Brushes können zusätzlich noch geteilt werden
- Extrude
  - Für Extrude muss erst via Edit eine Fläche markiert werden. Diese wird mit Extrude extrudiert

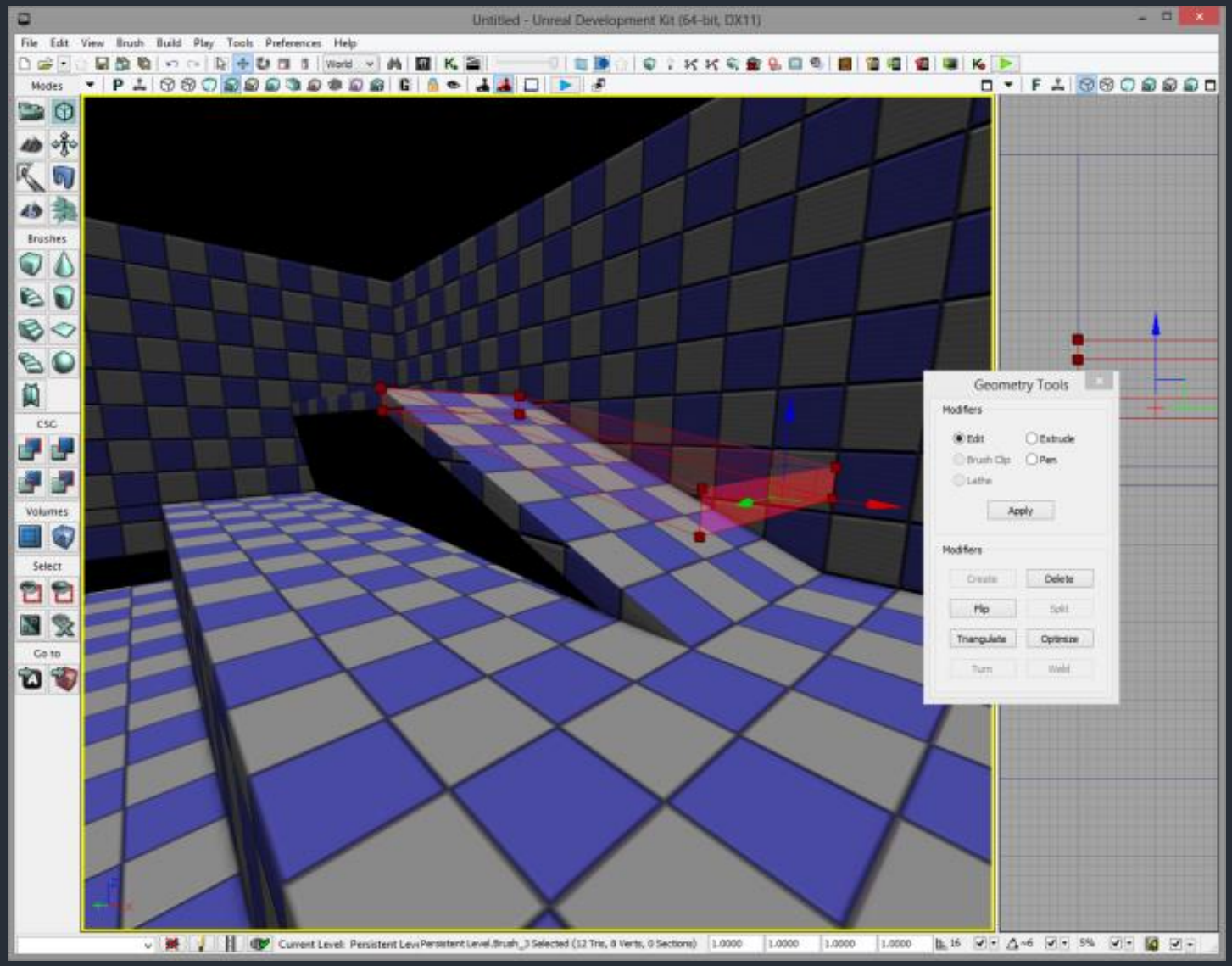
Geometry Tools





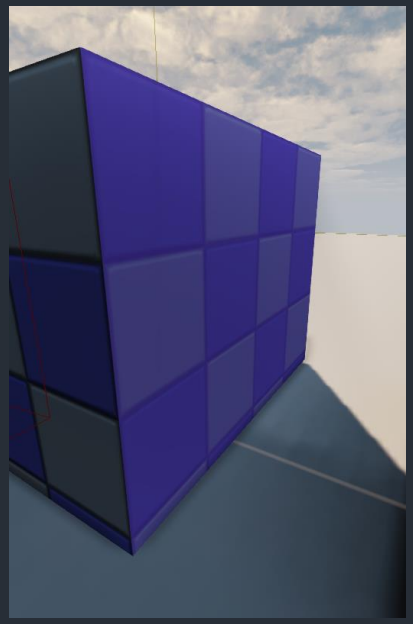
- Edit (das Einzige was man wirklich braucht)
  - Damit kann man Ecken, Kanten und ganze Flächen eines Brushes verschieben
  - Auf Builder Brush und bestehende Brushes anwendbar
- Pen
  - In einer nicht perspektivischen Viewport-Ansicht kann man mit STRG+Rechter Maustaste einen Builderbrush zusammenklicken

Geometry Tools



Surface Properties

- Brushes erzeugen Oberflächen, welche z.B. in den Viewport-Modi „Unlit“ oder „Lit“ mit Linksklick markiert werden können. Die Fläche wird daraufhin blau gefärbt dargestellt
- Per Doppelklick oder Rechtsklick → Surface Properties werden die Einstellungen für diese Oberfläche geöffnet
- Mit gedrücktem STRG können mehrere Oberflächen auf einmal selektiert und deren Einstellungen verändert werden



Surface Properties

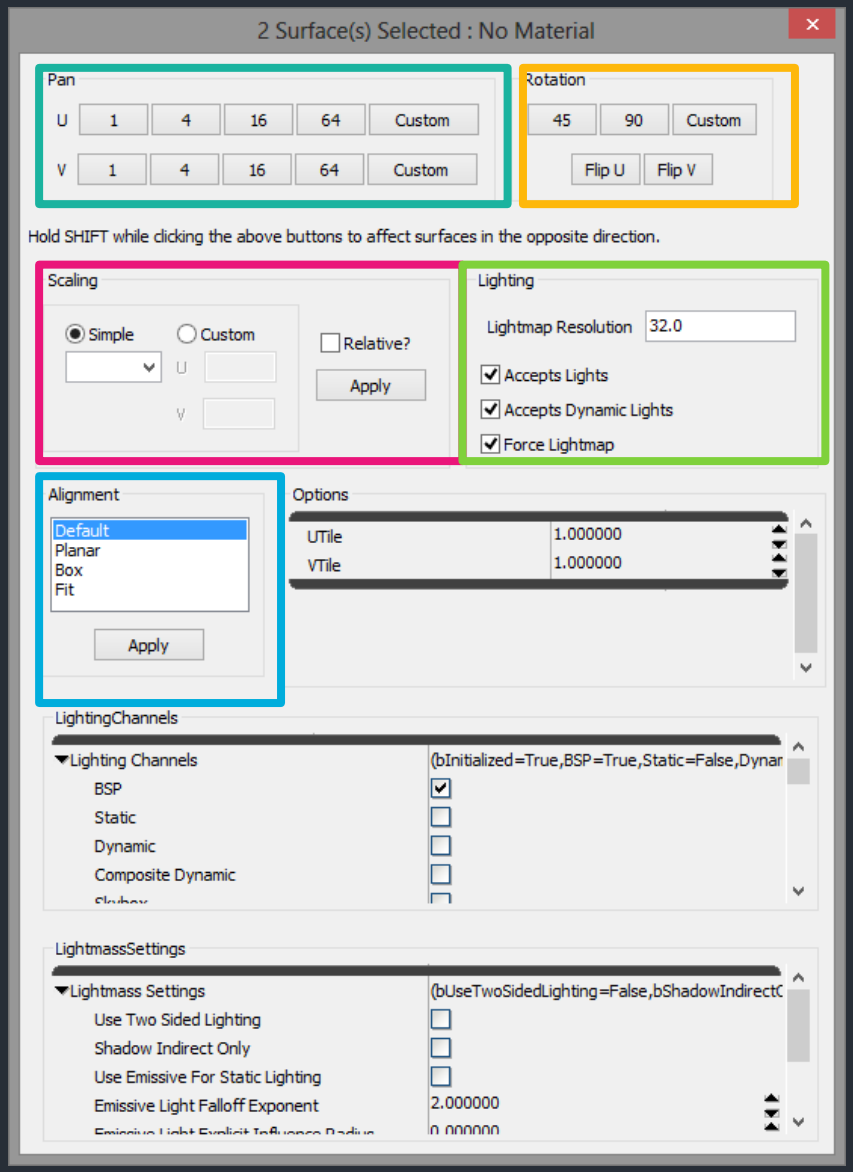
Verschieben des Materials

Rotieren des Materials

Skalieren des Materials

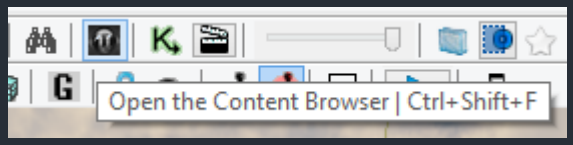
Einstellung zum Lighting

Automatisches Ausrichten des Materials, vor allem interessant wenn mehrere Oberflächen ausgewählt sind

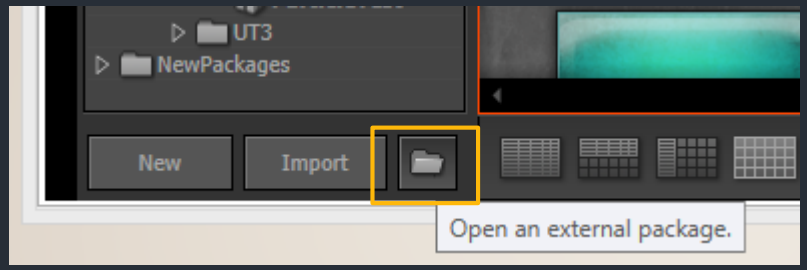


Materialzuweisung

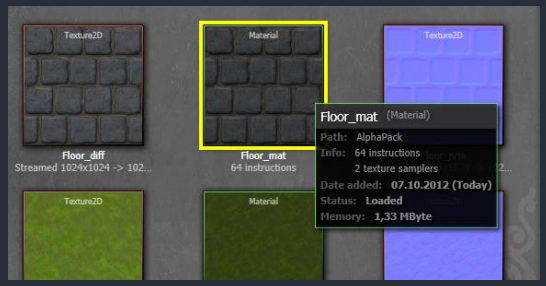
- Vorspiel (hier in Bezug zum Tutorium)
  - Content Browser öffnen



  - Fertiges Package „AlphaPack.upk“ öffnen

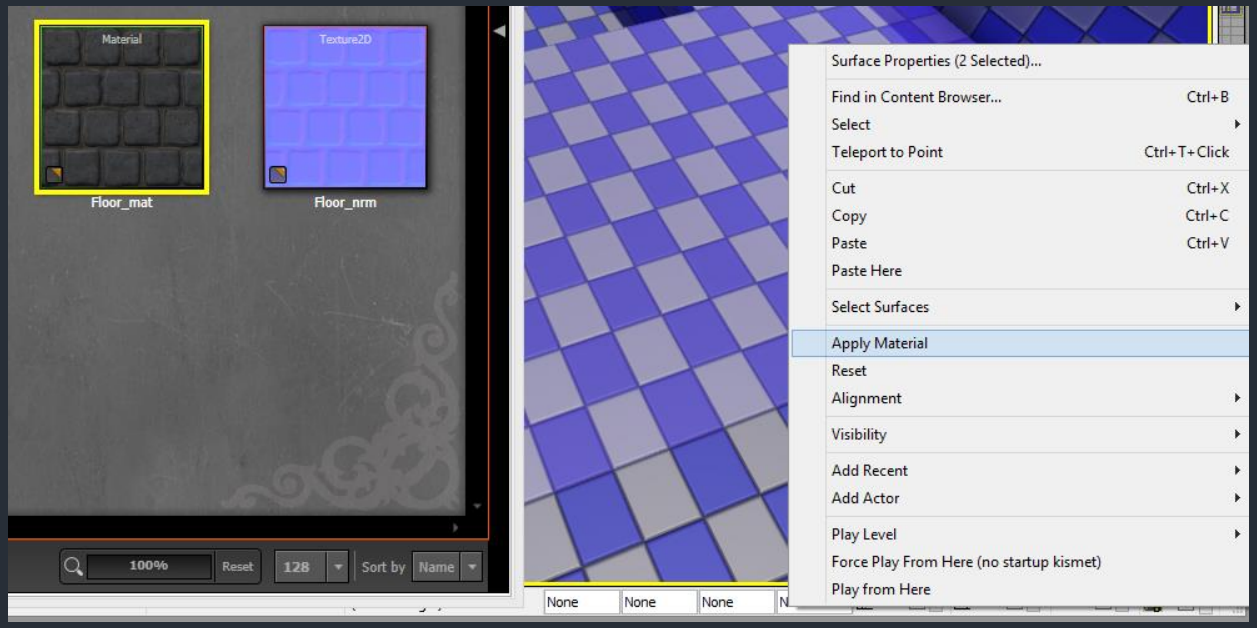


  - Gewünschtes Material mit Linksklick auswählen



Materialzuweisung

- Brushoberfläche bzw. Surface mit rechter Maustaste anklicken und „Apply Material“ zuweisen. Alternativ funktioniert auch Drag‘n‘Drop



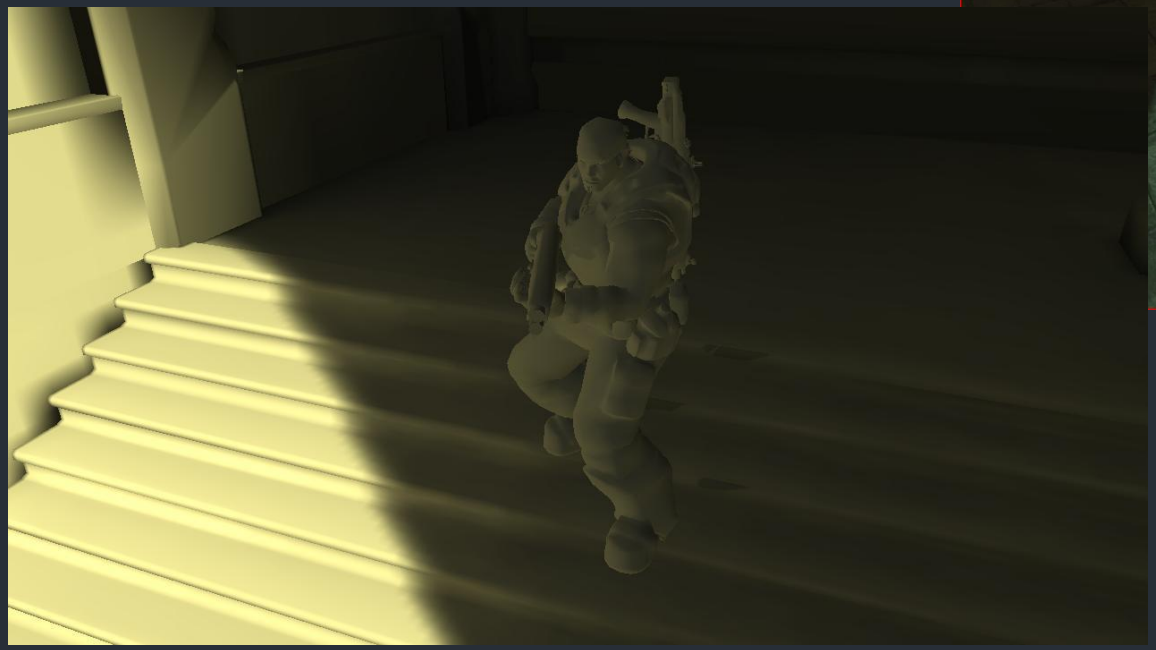
Lighting

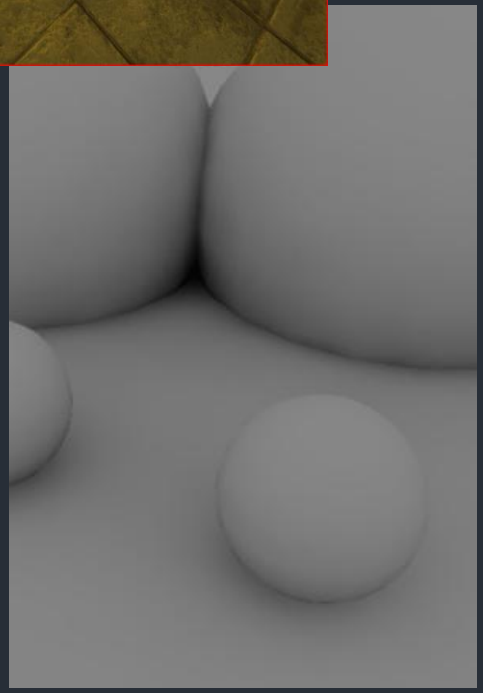
- Lightmass
- Was ist eine Lightmap?
- Directional Light
- Point Light
- (Actor) Properties
- Grouping
- Volumes
- PlayerStart
- Building

Lightmass

- „Lightmass“ wurde mit UDK eingeführt und bezeichnet eine neue Beleuchtungstechnik, welche für die statische Beleuchtung benutzt wird. Es handelt sich dabei um Global Illumination, d.h. Lichtquellen (Spezielle Actors oder Materialien) senden Photonen aus und das Lightmass Modul berechnet die Lightmaps für alle statischen Actors
- Für dynamische Objekte gibt es ein 3D-Grid mit Informationen über indirekte Beleuchtung

Lightmass

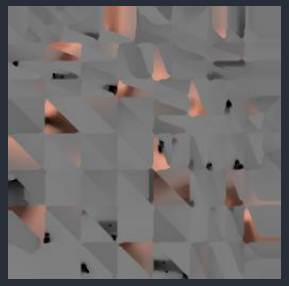




http://udn.epicgames.com/Three/Lightmass.html

Was ist eine Lightmap?

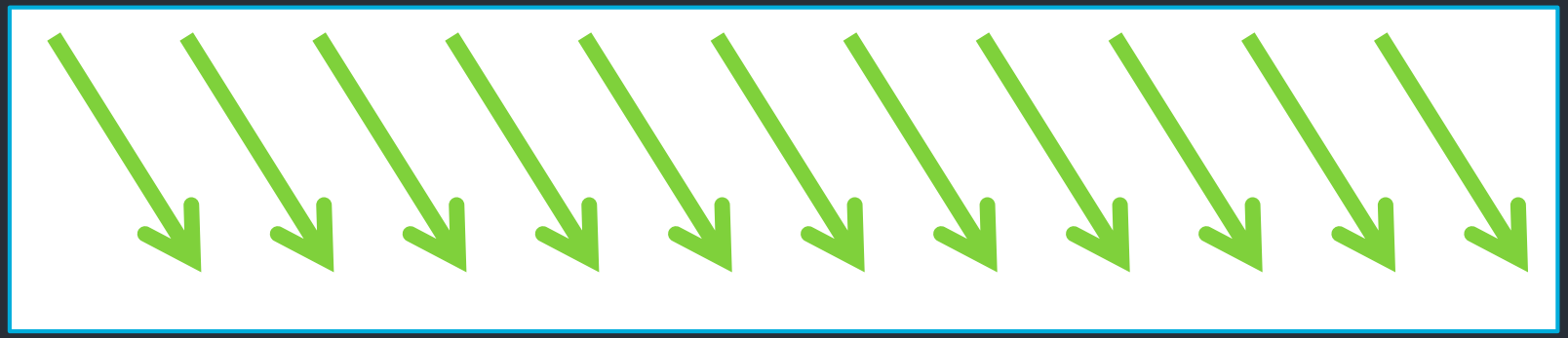
- Für statische Objekte (Geometry, StaticMeshes…) wird die statische Beleuchtung mithilfe von Lightmass vorberechnet und in sog. Lightmaps gespeichert. Dabei handelt es sich im Endeffekt nur um Texturen



- Bei Geometry werden die Voraussetzungen für Lightmaps automatisch hergestellt, bei StaticMeshes sieht das anders aus

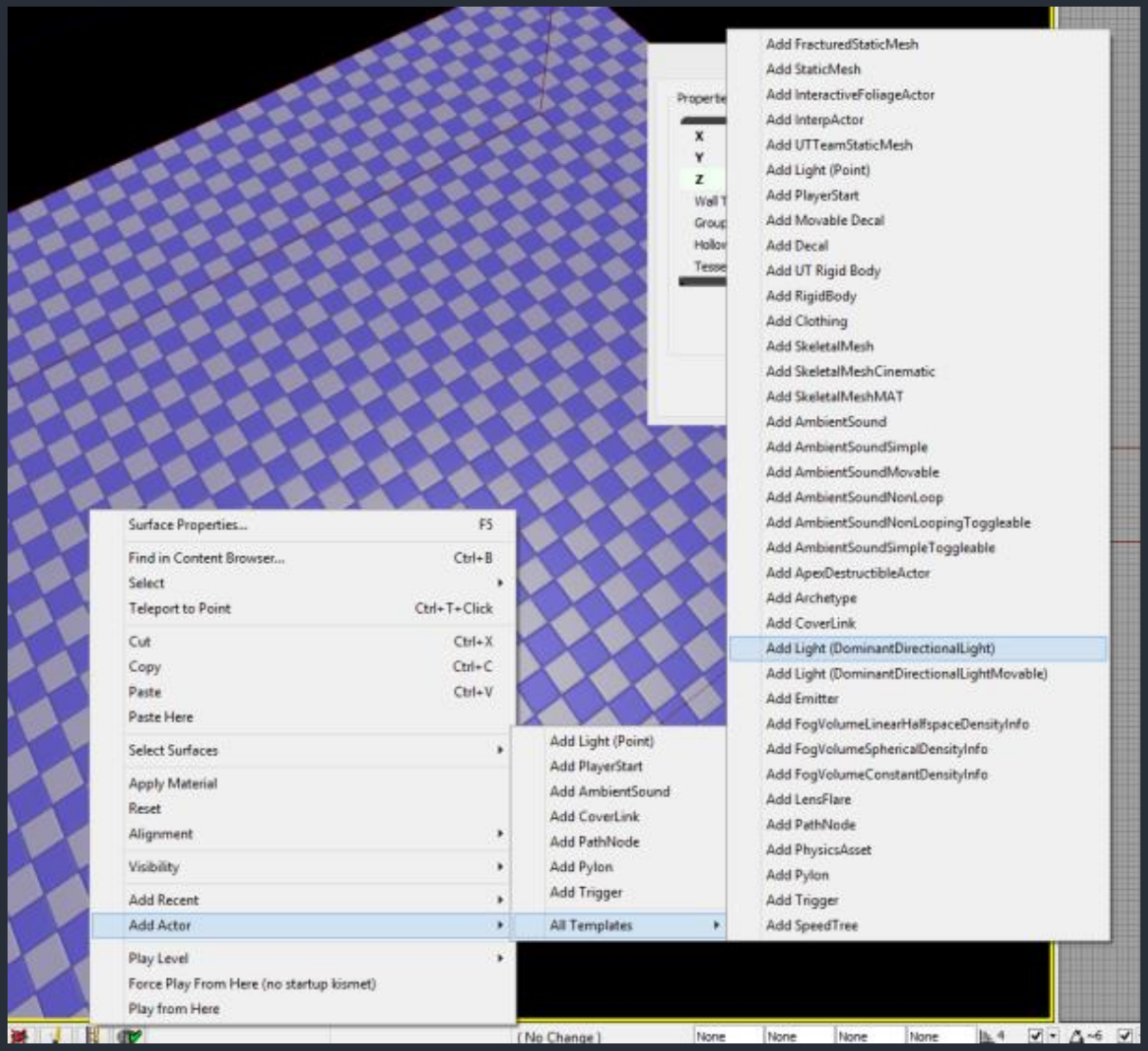
Directional Light

- Alle Lichtstrahlen kommen parallel über die ganze Welt. Daher ist nur die Rotation des Actors wichtig, Position und Skalierung sind irrelevant. Wird unter anderem für Sonnenlicht eingesetzt

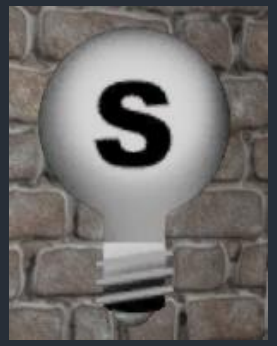


- Kann über Pop-Up Menü bei Rechtsklick in die Welt gesetzt werden

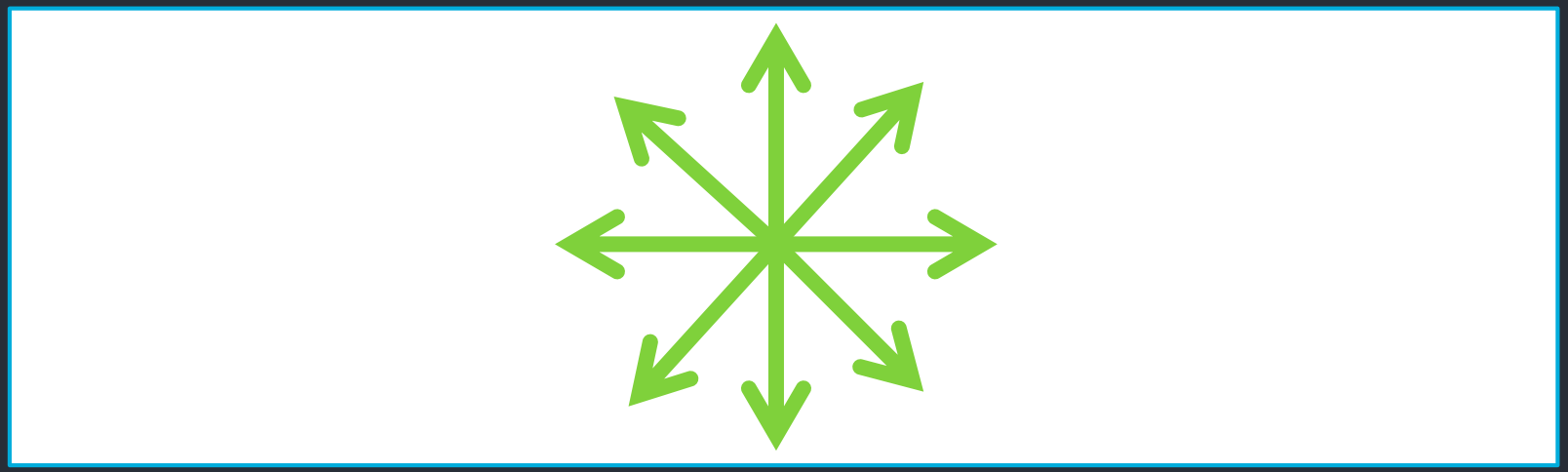
Directional Light



Point Light

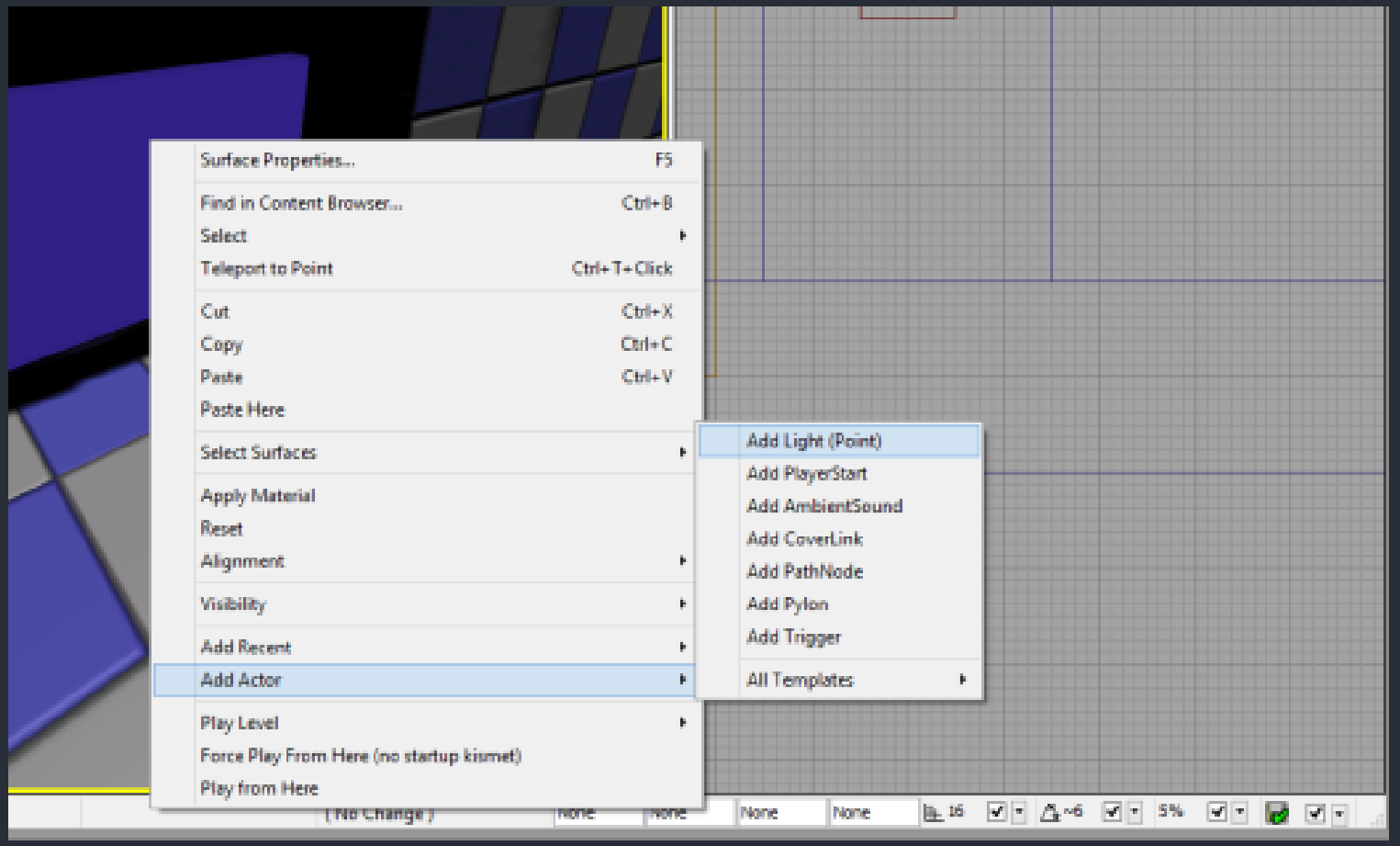


- Simuliert eine Punktlichtquelle, z.B. eine Glühlampe. Rotation des Actors ist dabei irrelevant aber Position und Skalierung werden beachtet



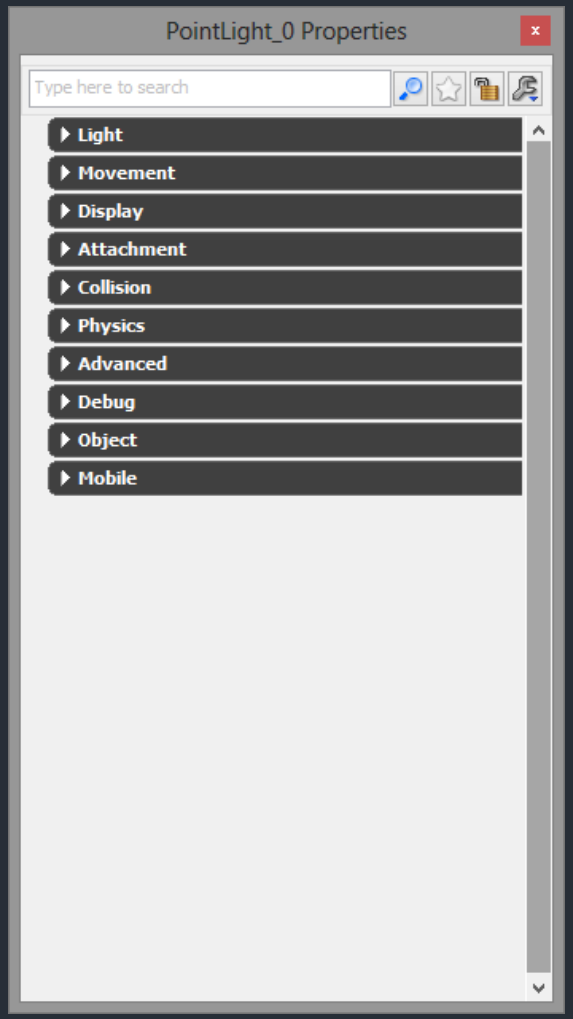
- Kann über Pop-Up Menü bei Rechtsklick in die Welt gesetzt werden

Point Light


Surface Properties... F5
Find in Content Browser... Ctrl+B
Select
Teleport to Point Ctrl+T+Click
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Paste Here
Select Surfaces
Apply Material
Reset
Alignment
Visibility
Add Recent
Add Actor
Play Level
Force Play From Here (no startup kismet)
Play from Here
Add Light (Point)
Add PlayerStart
Add AmbientSound
Add CoverLink
Add PathNode
Add Pylon
Add Trigger
All Templates
None
None
None
None
16
5%

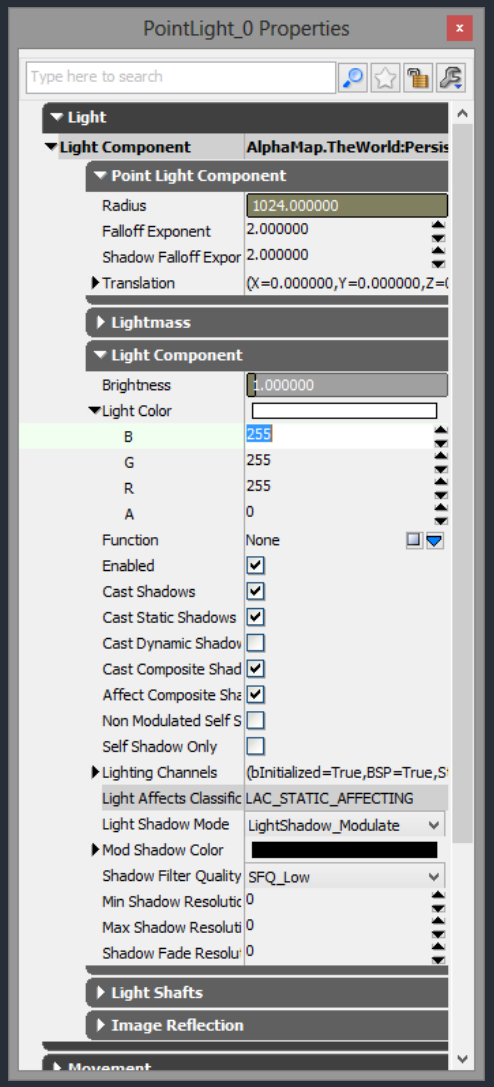
(Actor) Properties

- Doppelklick oder Rechtsklick auf den Actor (hier das Glühlampen-Icon) → Properties öffnet die Einstellungen zu diesem Actor. Jeder Actortyp hat spezialisierte Properties, sodass einige Reiter alle haben und manche Reiter nur bestimmten Typen vorkommen

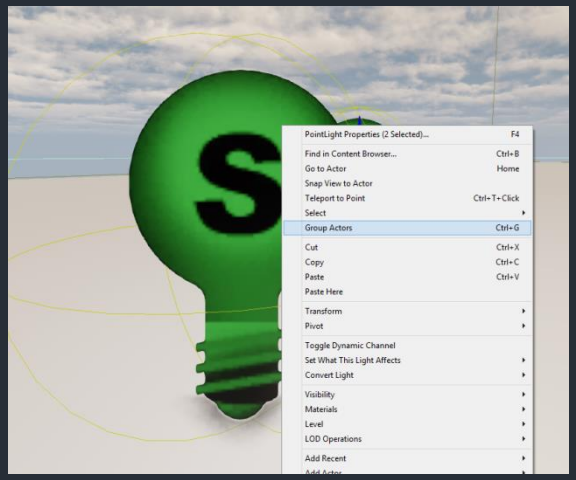


(Actor) Properties

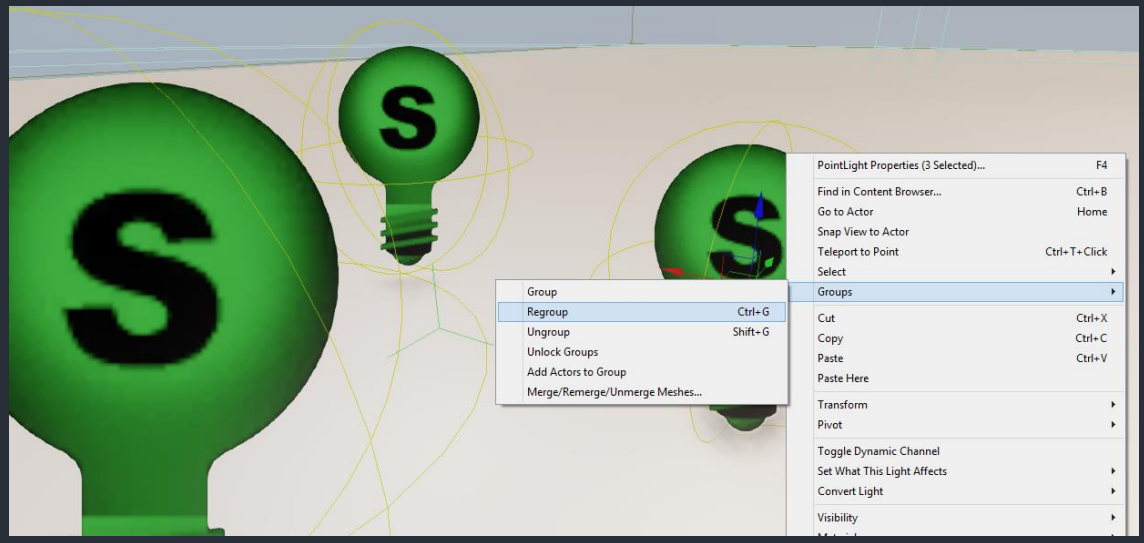
- Für Lichter ist der „Light"-Reiter sehr interessant, da in diesem die Lichtfarbe angegeben werden kann
- Immer dabei ist „Movement" und „Display", wo Position, Rotation und Skalierung manuell verändert werden können. Skalierung betrifft hier aber nicht den Radius sondern nur die Icon-Größe. Das Scale-Widget wiederum verändert im Light Actor den Radius des Lichtes und nicht die Skalierung unter Display



Grouping



Actors zu Gruppe zusammenfassen

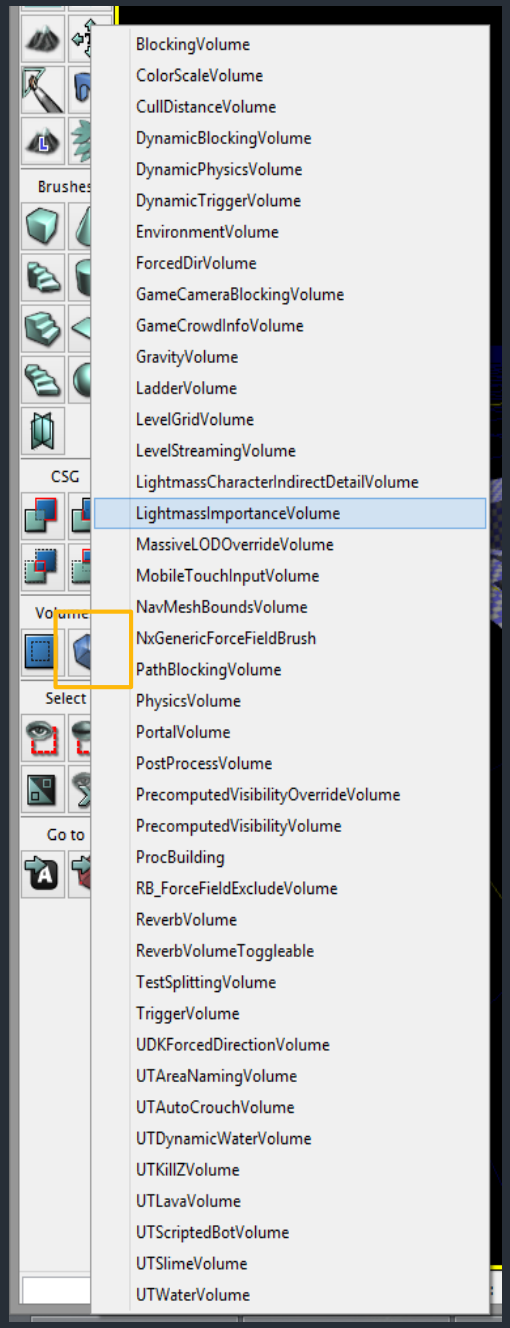


Bestehende Gruppe erweitern

- Desweiteren gibt es noch Prefabs, damit kann man Gruppierungen von Objekten als Objekt im Package speichern und immer wieder verwenden.

Volumes

- Mithilfe des Builder Brushes lässt sich nicht nur Geometry erzeugen, sondern auch Volumes
- Diese sind meist im Spiel unsichtbar, können aber vielfältige Aufgaben vollrichten
- Rechtsklick auf den Volumes-Button in der linken Toolbar

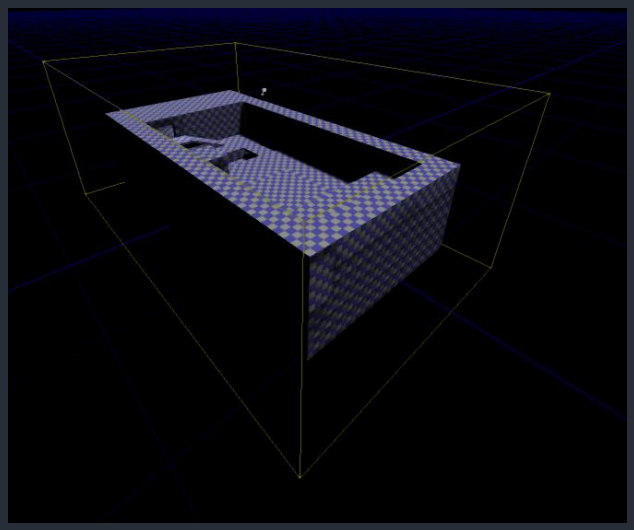


Volumes

- Blocking Volume
  - Blockiert den Spieler bzw. ausgewählte Collisionsklassen
- Physics Volume
  - Für den Bereich des Volumes können abweichende Umwelteigenschaften wie Beschleunigung definiert werden
  - Kann z.B. für Flüsse oder Lava eingesetzt werden

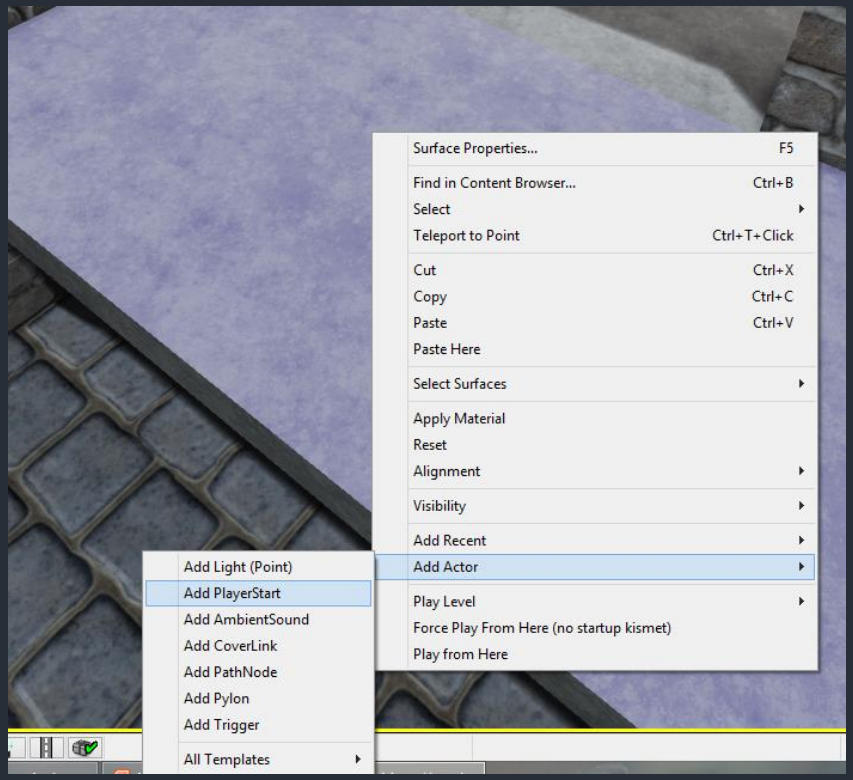
Volumes

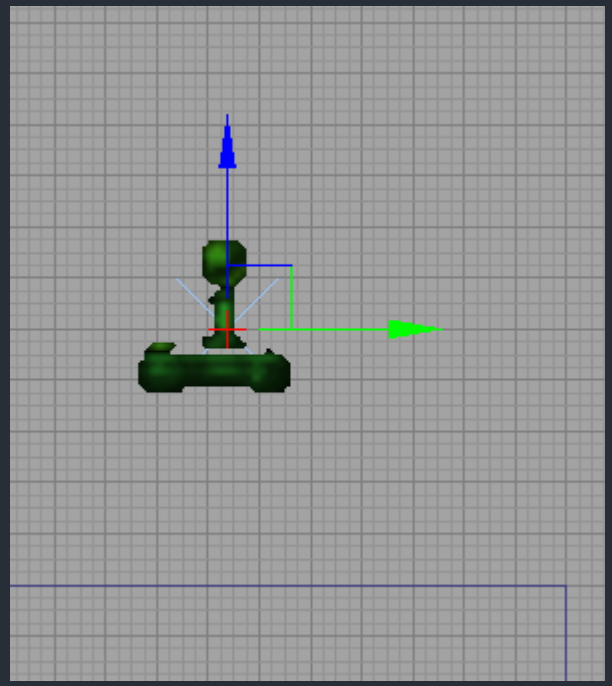
- Lightmass Importance Volume
  - Definiert den „wichtigen“ Bereich für die Lichtberechnung der „Dominant“ Lights, sodass außerhalb weniger genau und damit schneller berechnet werden kann
  - Nur innerhalb dieses Volumes bekommen dynamische Objekte indirekte Beleuchtung durch die Umgebung



PlayerStart

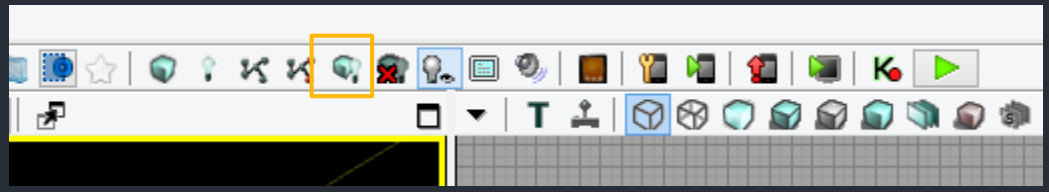
- Es fehlt noch ein Actor um den Startpunkt des Spielers zu definieren





Building

- Aus den Brushes muss eine „echte“ polygongestützte Geometrie errechnet werden. Lightmass muss ausgeführt werden und eventuell müssen auch Pfadberechnung für die KI gemacht werden. Dazu gibt es einen „Build All“-Button in der Kopfleiste, der alle diese Funktionen startet und automatisch ausführt.



- Für die Beleuchtung gibt es desweiteren noch verschiedene Qualitätsstufen, die sich mit Rechtsklick auf das Icon einstellen lassen

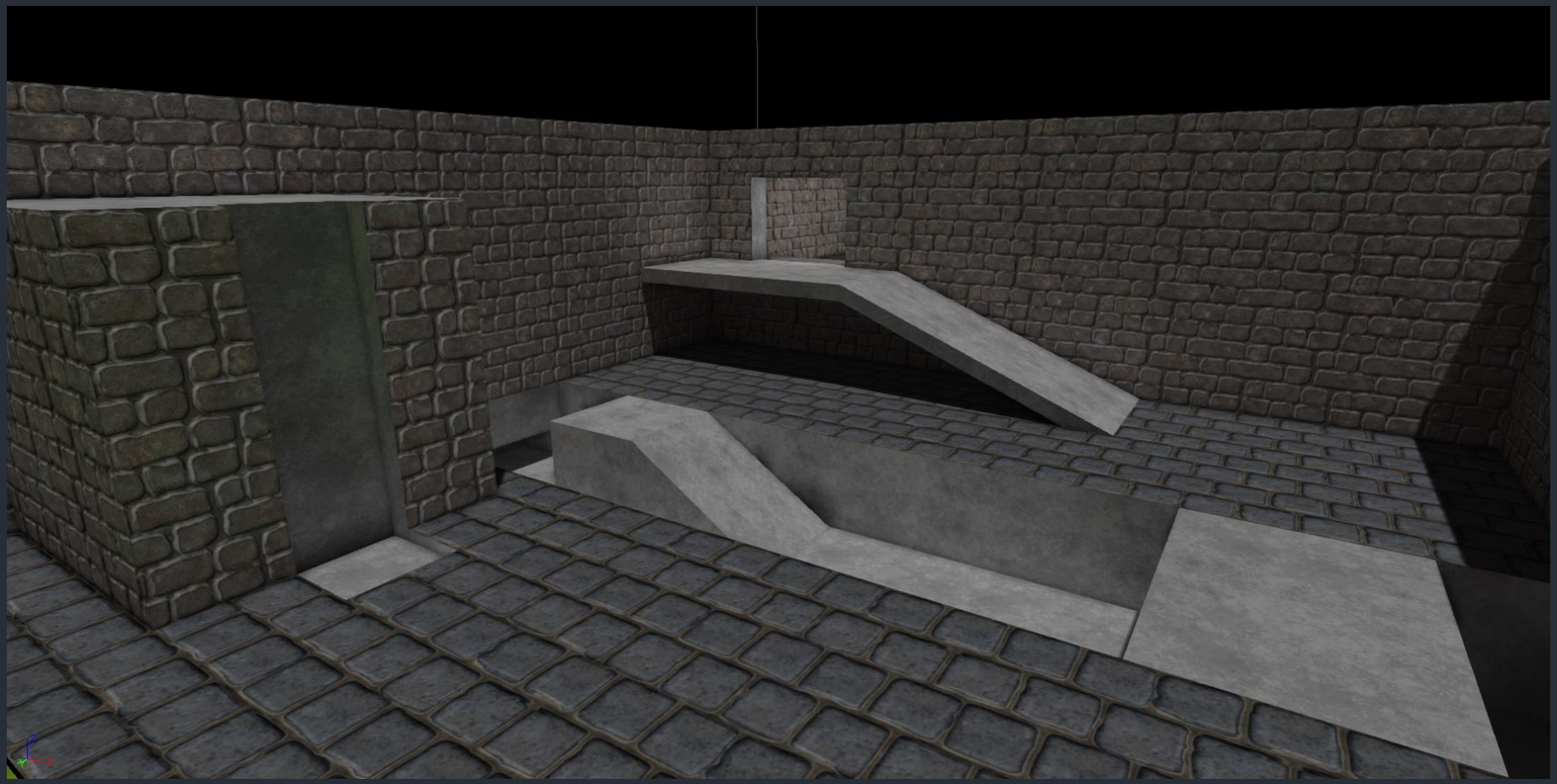
Bemerkungen

- BSP-Holes sind Löcher in der Geometriedarstellung. Dies passiert oft, wenn stark verzerrte Brushes verwendet werden oder nicht am Grid gearbeitet wurde
- Allen Surfaces von einem Brush wird automatisch das aktuell ausgewählte Material im Browser beim Erschaffen (Add/Subtract) zugewiesen
- Bei den Geometry Tools habe ich „Lathe" unterschlagen. Solche Sachen sollte man auf jeden Fall extern machen und nicht mit Geometrie
- BSP kann keine weichen Rundungen und braucht generell mehr Leistung als StaticMeshes
- Jede Map wird auch als Pack gleichen Namens angezeigt. Content der da erzeugt wurde wird in der .udk gespeichert
- Light Actors und andere Icons sind automatisch in deren Properties als „Hidden" markiert, sodass die Icons im Spiel unsichtbar sind

Resources

- Geometry Mode:
  http://udn.epicgames.com/Three/GettingStartedWithGeometryMode.html
- Dominant Lights:
  http://udn.epicgames.com/Three/DominantLights.html

Hausaufgabe



Eigene Welt bauen.

(Ihr könnt das bestimmt besser als ich ;)

Inhalt

1. Einleitung und Editor
2. Geometry und Lighting
3. StaticMeshes und Materials
4. Kismet
5. Particles

Bitte „myBetaMap.udk“ im UDK laden und Content in „myBetaPack“ importieren

Zum Content-Managment…

- Wie im letzten Tutorium findet ihr fertigen Content in „BetaPack.upk“ und eine fertige Map in Form von „BetaMap.udk“ vor
- Zum Üben ladet bitte „MyBetaMap.udk“ und erstellt im Laufe des Tutoriums das Pack „MyBetaPack.upk“, um selbst Assets zu importieren und Materialien zu erstellen

StaticMeshes

- Was ist der Unterschied zu Geometry?
- Exkurs: UV?
- Import ins UDK
- Einstellungen des globalen Objektes im Browser
- Einfügen in die Welt
- Einstellungen des lokalen Actors in der Welt

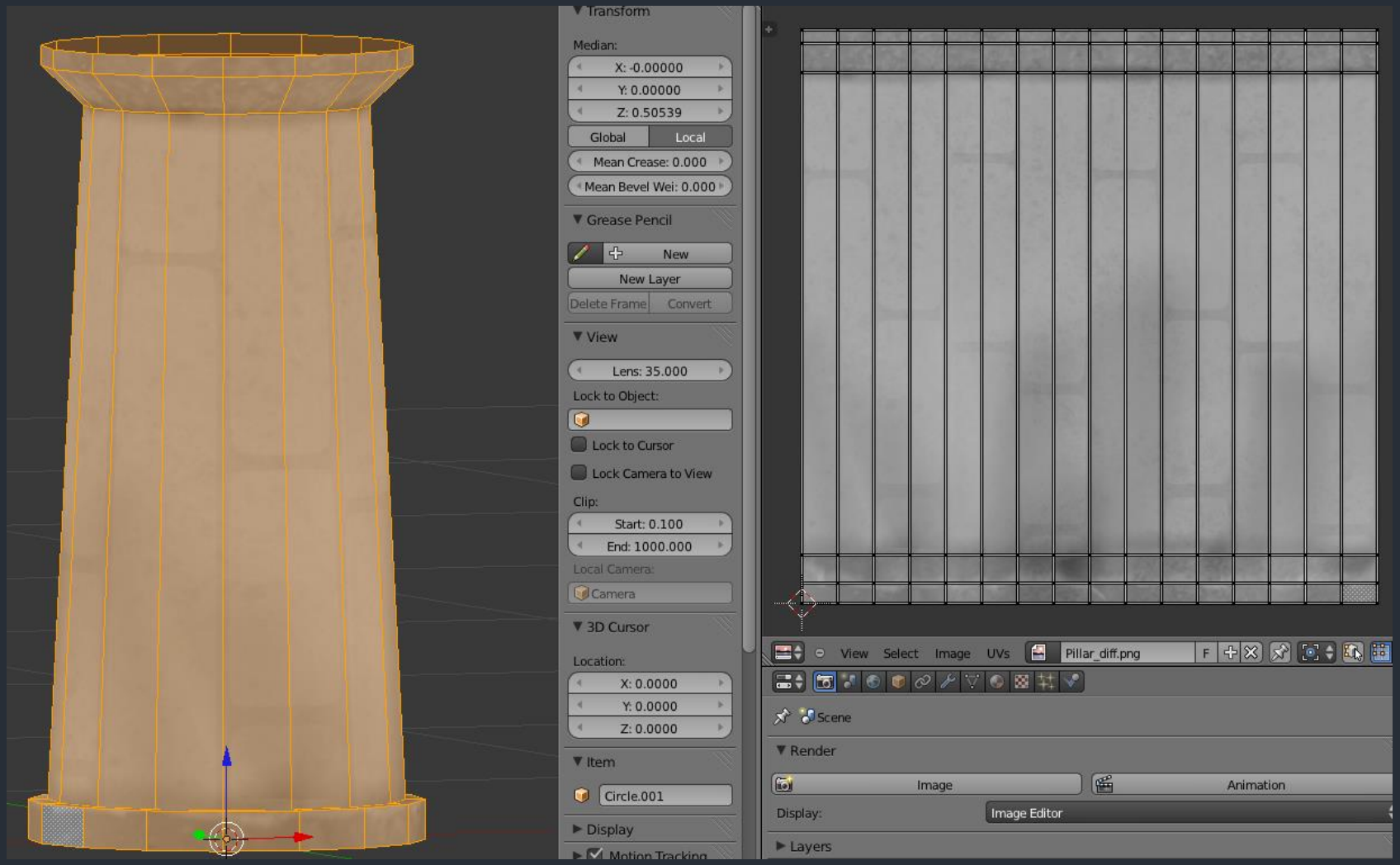
Was ist der Unterschied zu Geometry?

- Allgemein
  - StaticMeshes werde importiert, sprich sie werden nicht mit dem UDK erstellt sondern extern (z.B. in 3Ds Max oder Blender)
  - Der Artist hat hier die Kontrolle über jedes Polygon, Geometrie wird im Gegensatz dazu generiert
  - Weiche Kanten werden ermöglicht (Smooth Shaded)
- Performance
  - Die Berechnung von StaticMeshes ist vor allem bei vielen Polygonen viel schneller als die von Geometrie
  - Es können Techniken wie Instancing angewendet werden
  - Ermöglichen LevelOfDetail (LoD)
- UV-Layer
  - StaticMeshes können bis zu 4 UV-Layer haben, wobei einer für die Lightmap-Berechnung ausgewählt werden sollte
- Weiteres
  - StaticMeshes können via Fracture Tool zerlegt werden

Exkurs: UV?

- StaticMeshes bestehen (wie auch erzeugte Geometrie) aus zu Dreiecken verbundenen Punkten
- Pro Punkt wir auch eine „UV-Koordinate“ gespeichert, sodass im Material jedem Punkt und dazwischen (durch Interpolation) ein Farbwert aus den Texturen zugewiesen werden kann

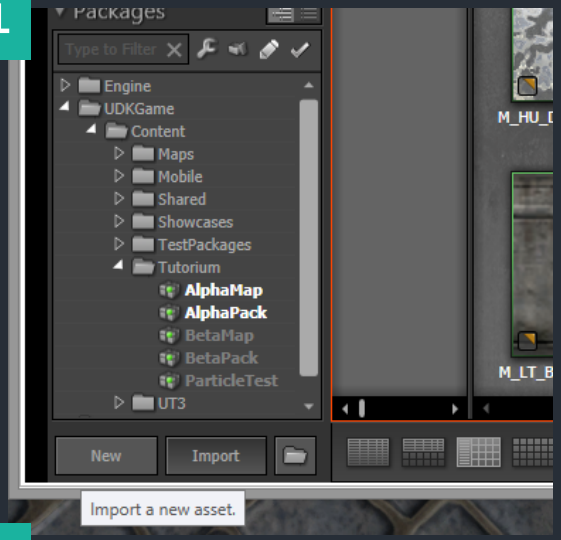
Exkurs: UV?



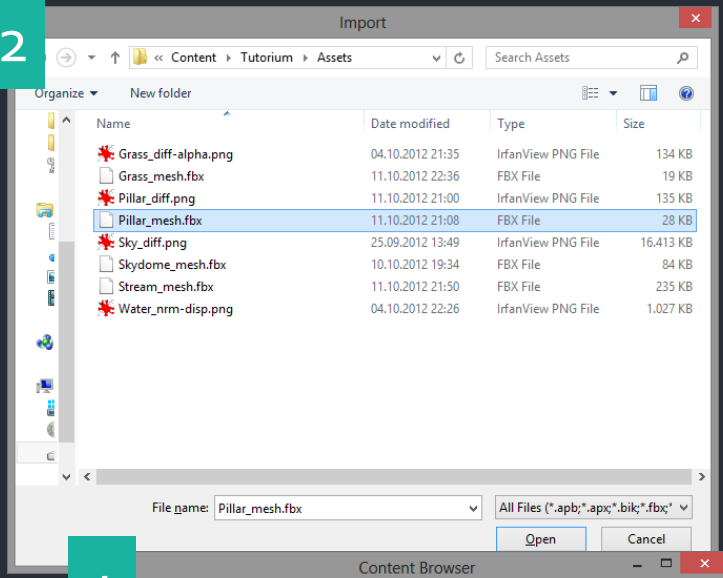
UV-Editor in Blender (rechts oben)

Import ins UDK

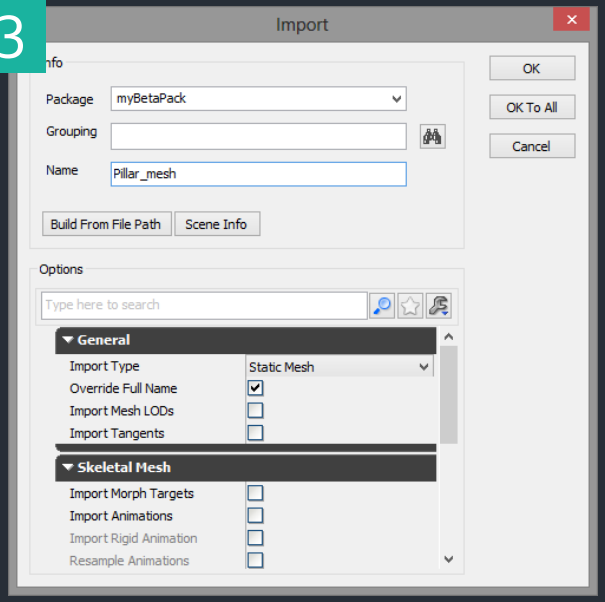
1



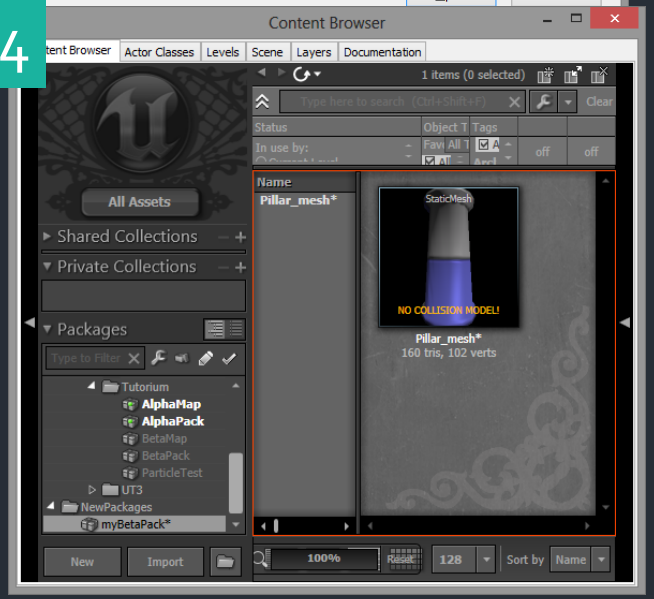
2



3

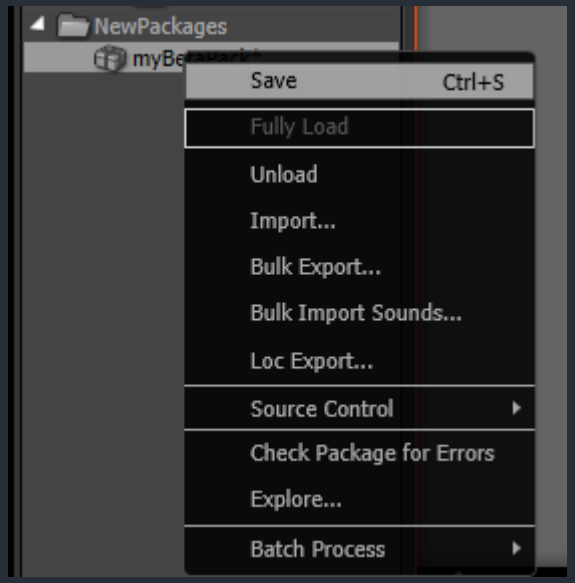


4



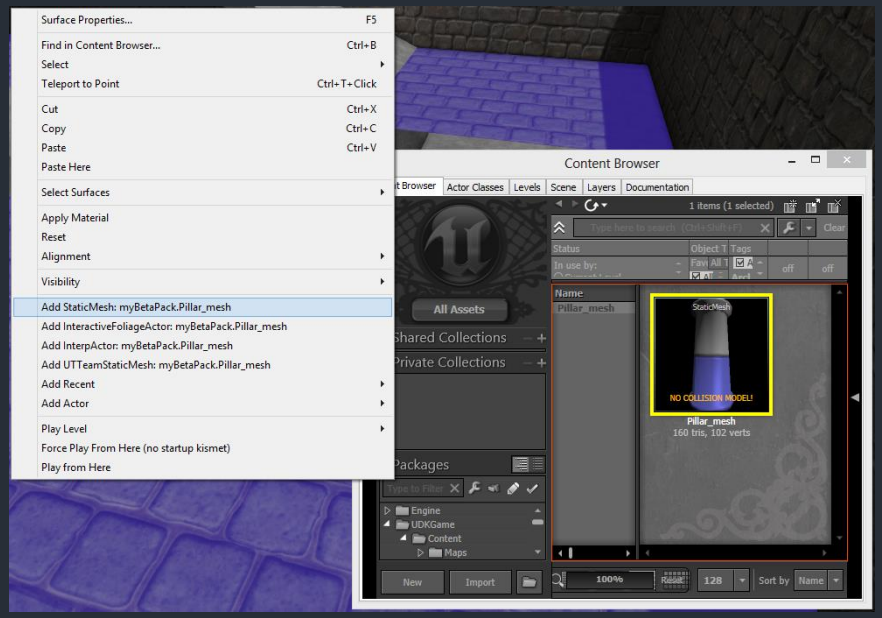
Import ins UDK

- Da mit dem Import ein neues Pack entstanden ist (myBetaPack.upk), sollte dieses auch im Tutorium-Ordner gespeichert werden, damit es auch unter „Tutorium“ angezeigt wird



Einfügen in die Welt

- Entweder per Drag'n'Drop vom Browser in die Welt oder..
- Per PopUp-Menü in der Welt bei selektiertem StaticMesh im Browser



Einstellungen des globalen Objektes im Browser

- Doppelklick auf den StaticMesh im Browser öffnet die Einstellungen (Navigation im 3D-Vorschaufenster via linker und rechter Maustaste)

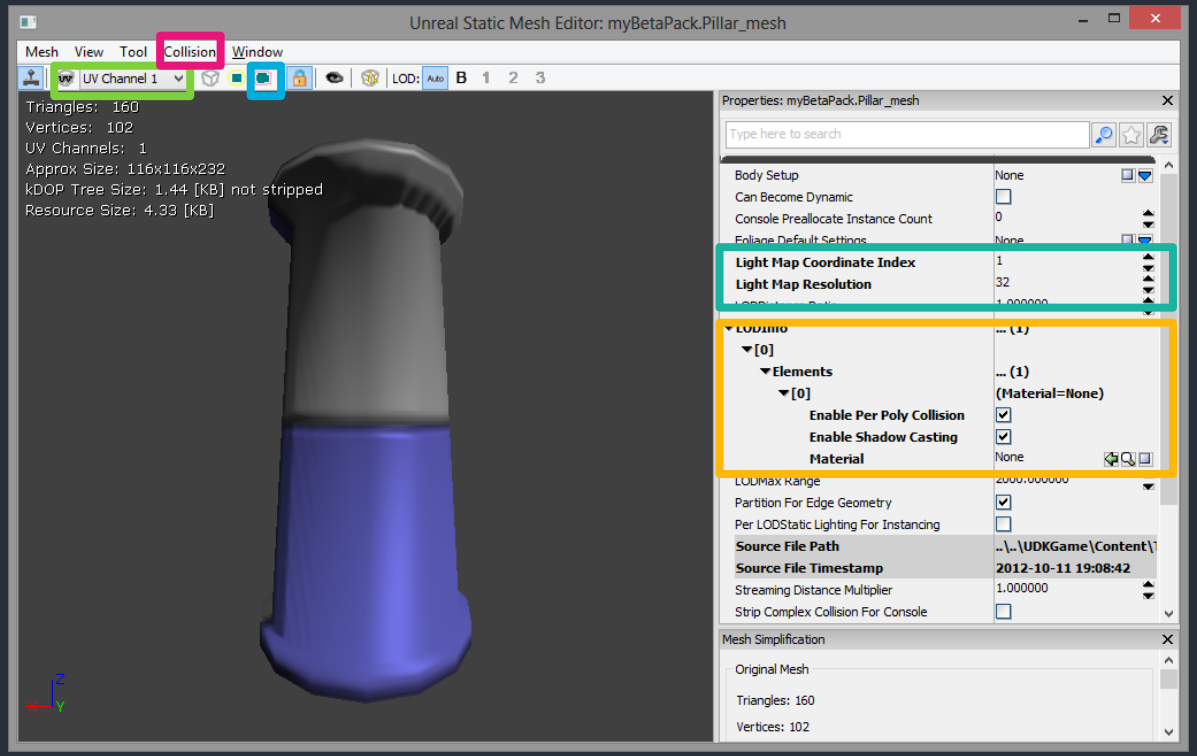
Lightmap-Einstellungen (beim Pillar ist die UV-Map für Lightmapping die gleiche wie für das Material, also 0 statt 1)

Materialzuweisung

Collision erstellen

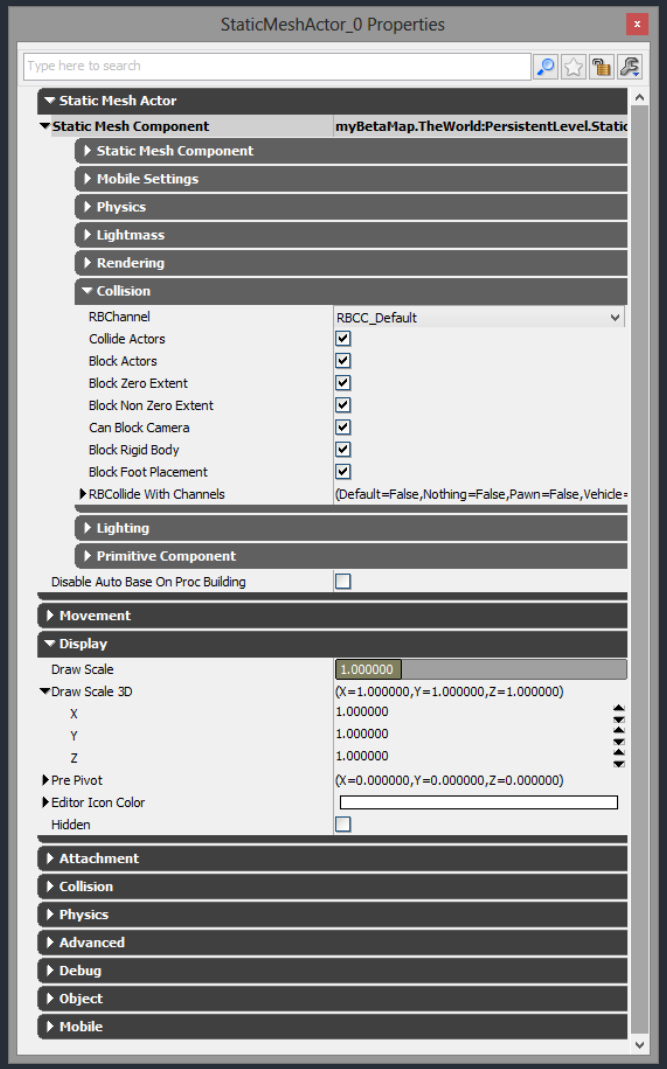
UV-Kanäle

Collisionshüllenvorschau



Einstellungen des lokalen Actors in der Welt

- Doppelklick auf das StaticMesh in der Welt öffnet dessen (lokale) Actor Propterties. Diese existieren wie bei den Lights pro Instanz des StaticMeshes in der Welt
- Es können Pos/Rot/Scale, Collision und vieles mehr eingestellt werden



Bermerkungen

- Beleuchtung von StaticMeshes ist in der Welt immer nur Vorschau, erst nach einem Rebuild ist sie final

Materials

- Was sind Texturen?
- Was ist ein Material?
- Import von Texturen
- Textureinstellungen
- Erstellung eines Materials
- Material Editor
- Material Output
- BlendModes
- Wichtige Nodes
- Beispiel: Pillar Material
- Beispiel: Sky Material
- Beispiel: Water Material
- Reflections (grobe Übersicht)

Was sind Texturen?

- Texturen bilden einen Eingabetypus für Materialien
- Im Endeffekt handelt es sich dabei um „normale“ Bilder, welche in einer „Power of Two“ Größe vorliegen müssen, zB. 256x256 oder 512x1024
- Importiert werden nahezu alle Formate, jedoch werden sie vom UDK intern zu .DDS konvertiert um Platz und Leistung zu sparen (ja, kann man gezielt abschalten)

Was sind Materialien?

- Für komplexe Oberflächen ist die Kombination verschiedener Texturen (Diffuse, Normal, Specular, Detail, Height…) im PixelShader notwendig, d.h. für jeden sichtbaren Pixel auf dem Bildschirm wird das passende Material angewendet und der Farbwert berechnet
- Materialien im UDK sind eine Abstraktion der bekannten „Shader", sodass im UDK für alle Plattformen Materialien erstellt werden können und sie intern für verschiedene Shader Modelle (DirectX/OpenGL) kompilierbar sind. Daher werden sie nicht geschrieben sondern mit Nodes zusammengebaut

Was sind Materialien?

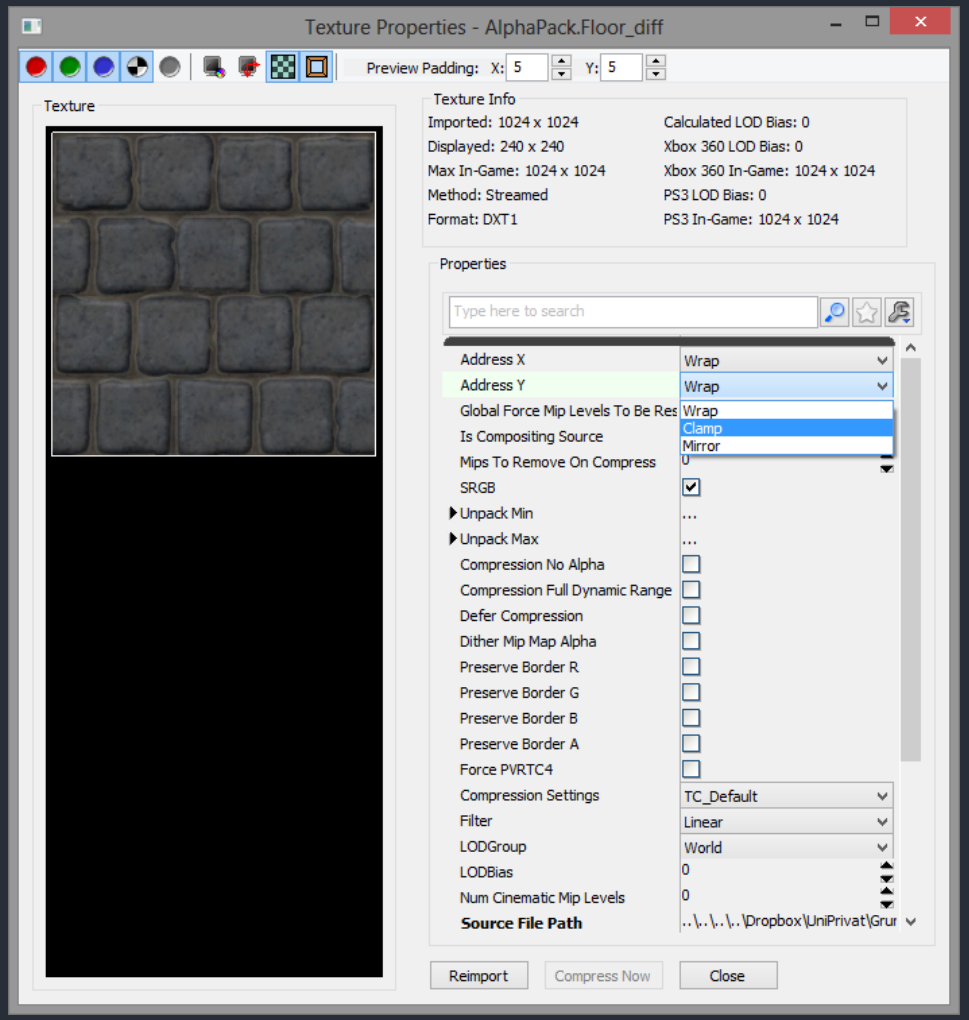
- Input für Materialien sind…
  - Texturen, deren Farbinformationen (RGBA) ausgelesen werden können
  - TextureCoordinates (UV-Layer), welche vom StaticMesh abhängen dem das Material zugewiesen wurde
  - WorldPosition, Time, Parameter…

Import von Texturen

- Funktioniert genauso wie für StaticMeshes
- Importierte Texturen werden auch im Browser angezeigt
- Erst beim Speichern des Packages werden Texturen zu .DDS konvertiert

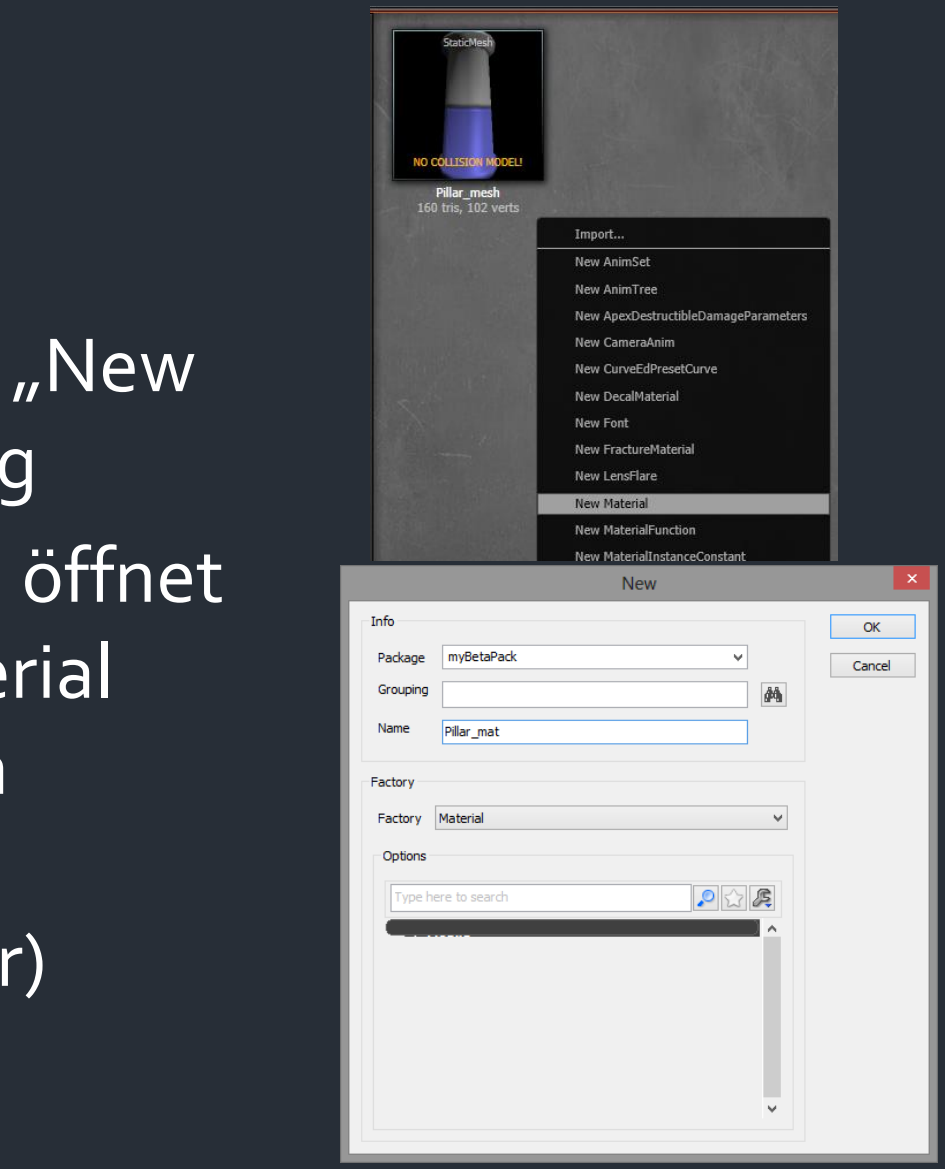
Textureinstellungen

- Für uns ist hier nur „Adress X“ und „Adress Y“ wichtig
  - Wrap: Die Textur wird immer weiter gekachelt, auch bei UV-Koordinaten >1 oder < 0
  - Clamp: Die Textur wird nur zwischen den UV-Koordinaten 0 und 1 dargestellt



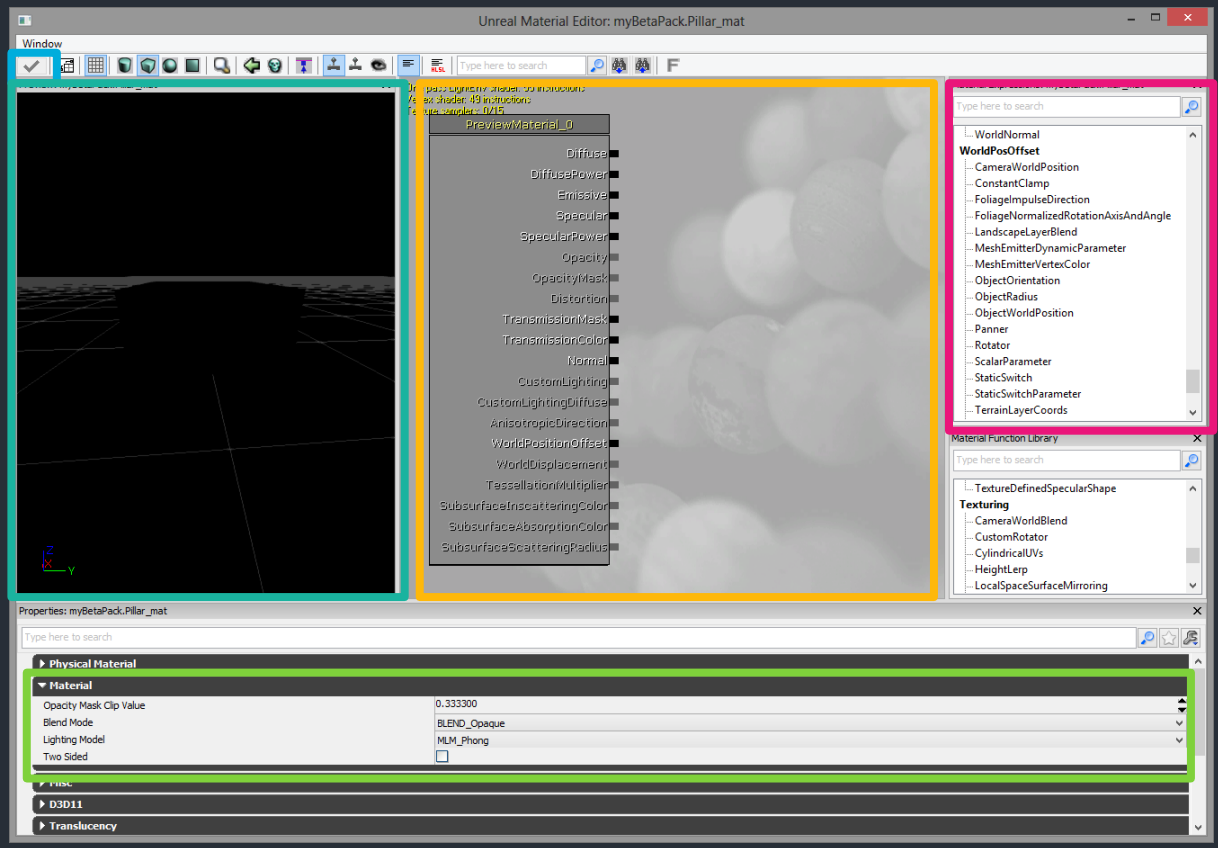
Erstellung eines Materials

- Rechtsklick in die Anzeigefläche des Packages und dann „New Material“ ist ein Weg
- Nach der Erstellung öffnet sich direkt der Material Editor (oder einfach Doppelklick auf das Material im Browser)

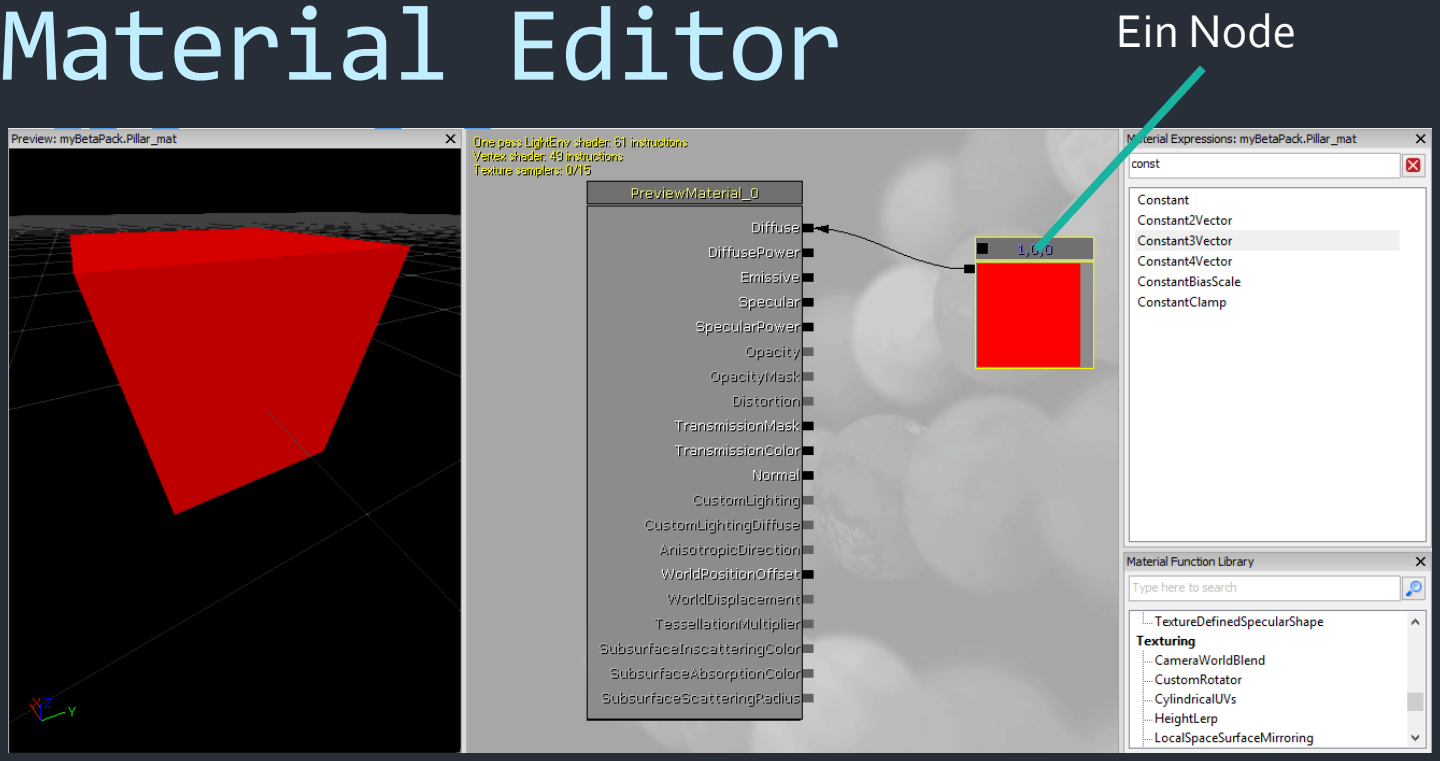


Material Editor

- 3D-Vorschaufenster
- Bereich für die Nodes
- Node Bibliothek
- Materialeigenschaften
- Material kompilieren



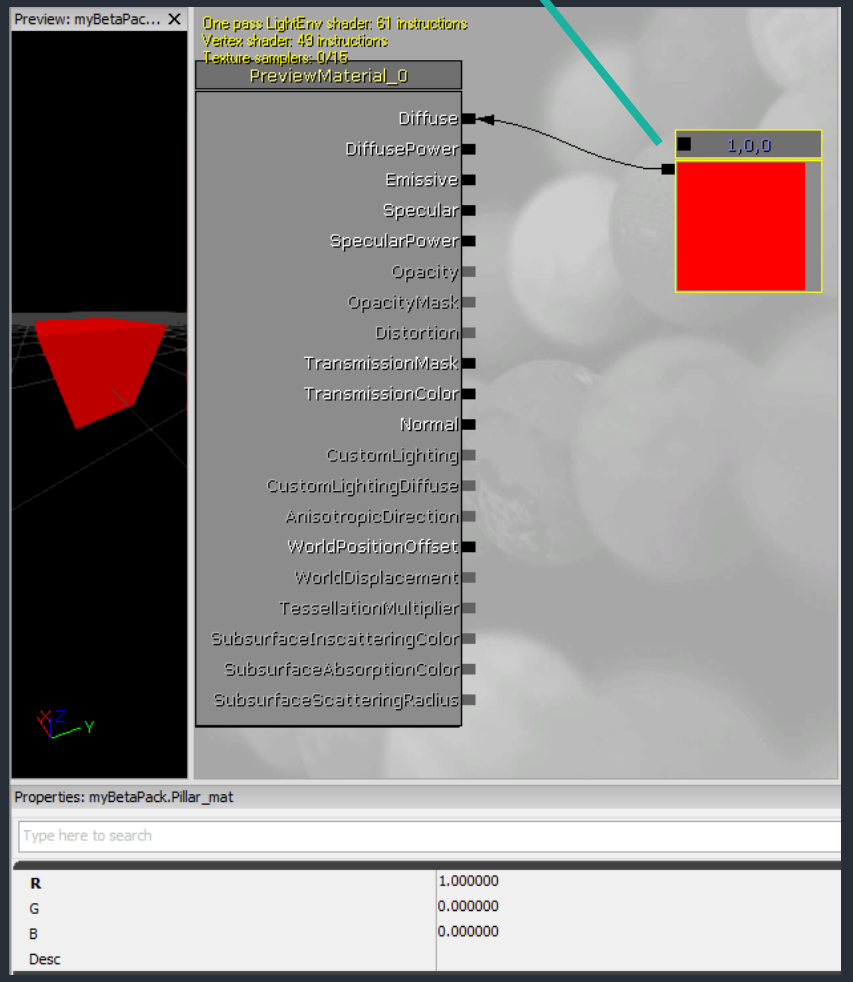
Material Editor



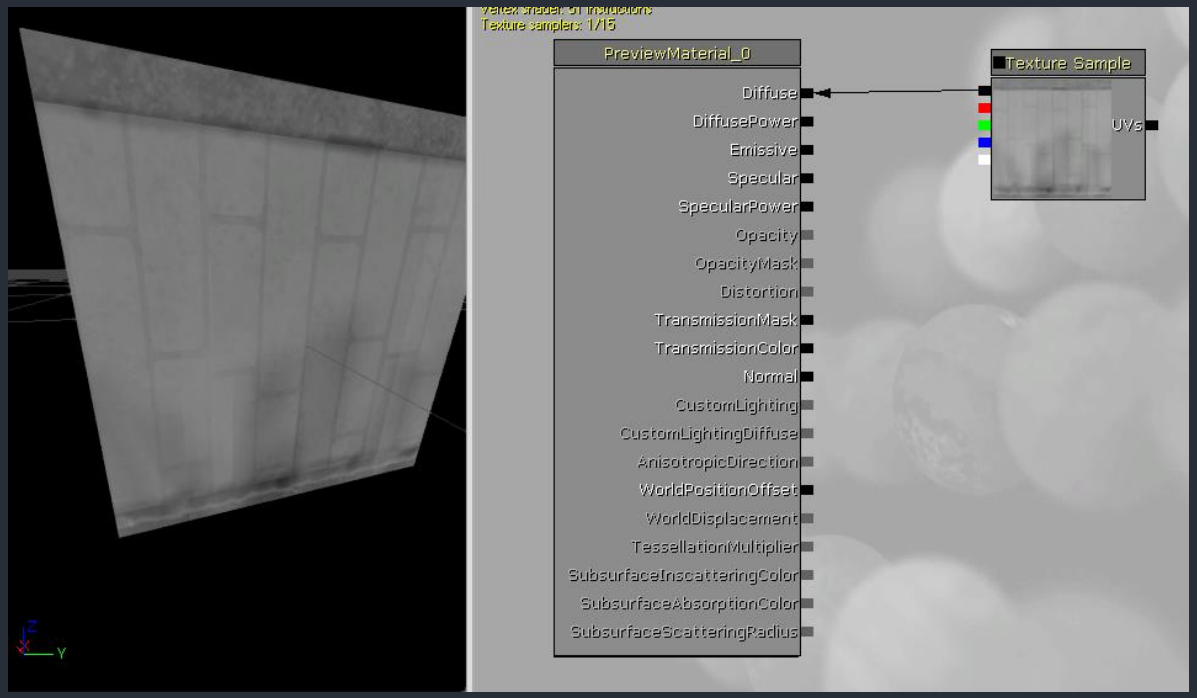
- Nodes können gesucht und per „Drag'n'Drop" eingefügt werden, das funktioniert auch mit Texturen direkt aus dem Browser
- Navigation in diesem Feld erfolgt wie bei 2D-Viewports

Material Editor

- Für selektierte Nodes werden unten die Properties eingeblendet
- Nodes haben manchmal Input (rechte Pins) und immer Output (linke Pins), wobei dieser mit anderen Inputs verbunden wird (linke Maustaste und ziehen)
- Per Rechtsklick auf die Pins können Verbindungen wieder gelöst werden



Material Editor



- TextureSamples (also Texturen im Material) haben 5 Output-Pins (RGB, Rot, Grün, Blau, Alpha)

Material Output

- Alles was später angezeigt wird muss der großen Ouput-Node zugeführt werden

Farbwert ohne Beleuchtung (die wird aufmultipliziert)

Farbwert, der die Beleuchtung ignoriert

Glanz des Materials

Größe des Glanzpunktes

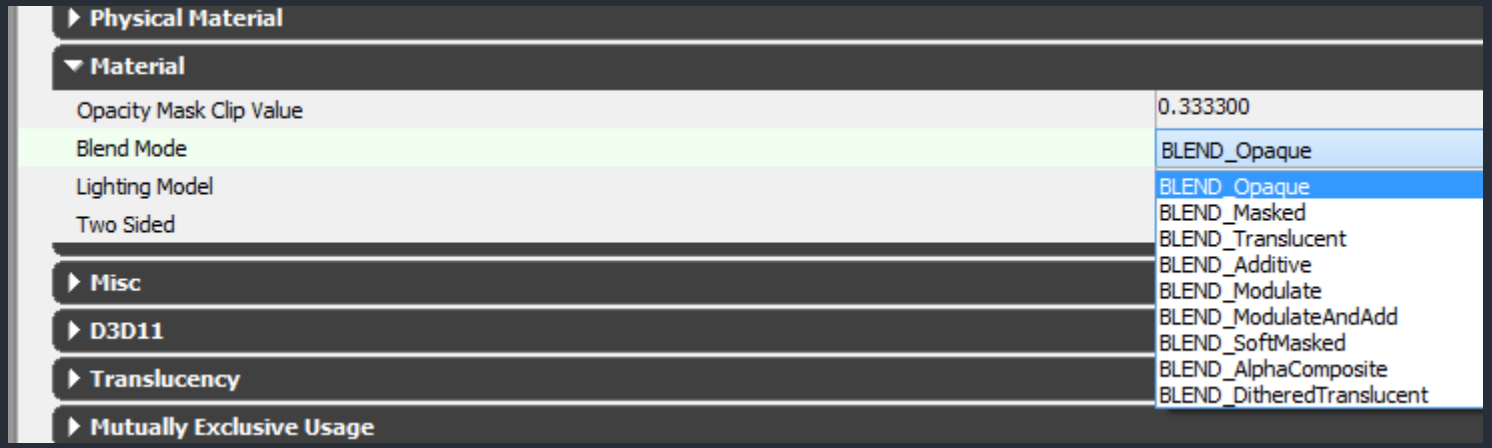
Transparenz (abhängig vom BlendMode)

Verzerrung des Hintergrunds bei Transparenz

Normal Map Input

Relative Vertex Position

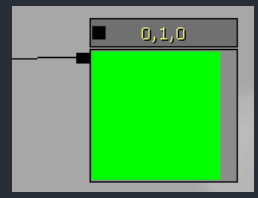
BlendModes

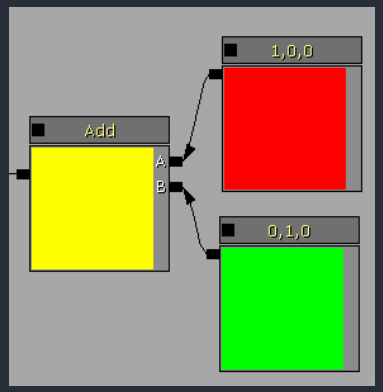


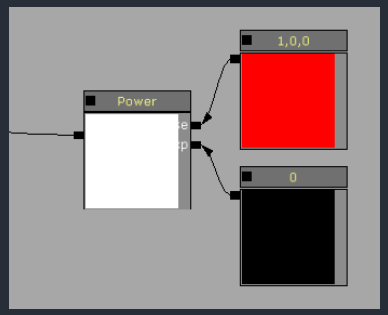
- Opaque
  - Undurchsichtig
- Masked
  - Binäre Transparenz (ja/nein pro Pixel)
  - OpacityMask Input-Pin für die Transparenz
- Translucent (Additive/Modulate ähnlich)
  - Richtige Transparenz (255 Werte pro Pixel)
  - Opacity Input-Pin für den Wert der Transparenz
  - Keine dynamische Beleuchtung möglich

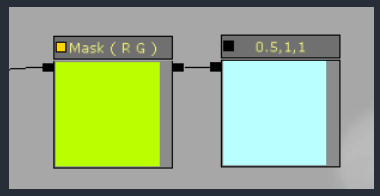
Wichtige Nodes

- Constant (-2Vector/-3Vector/-4Vector)
  - Konstante Float Werte
- Add/Multiply
  - Addition/Multiplikation
- Power
  - Basis hoch Exponent
  - Kann für Kontrastjustierung verwendet werden
- ComponentMask
  - Blendet Werte aus



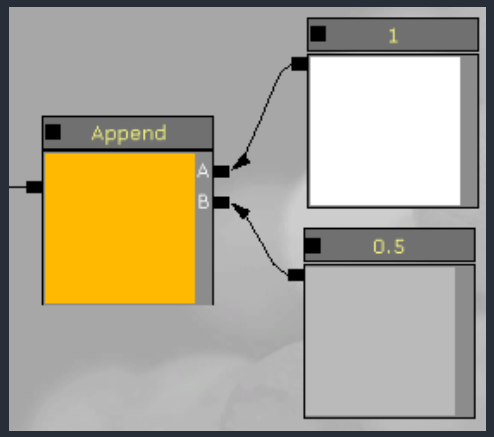


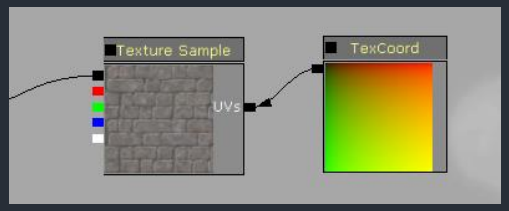


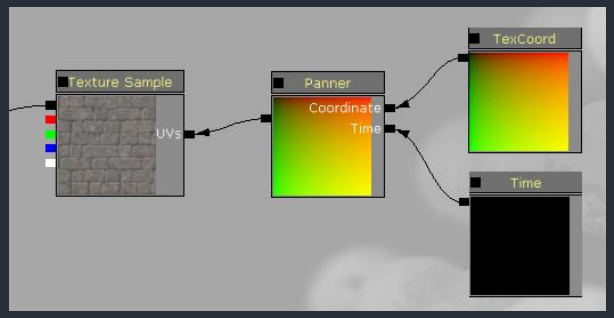


Wichtige Nodes

- Append
  - Fügt Werte zu Vektor zusammen
- TextureCoordinate
  - UVs vom (Static)Mesh
  - Layer kann ausgewählt werden, können skaliert werden
- Pan/Rotate/Scale
  - Modifiziert die Texture Coordinates
- Time
  - Gibt die absolute Zeit zurück

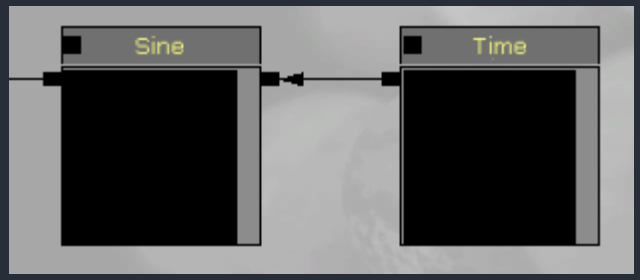


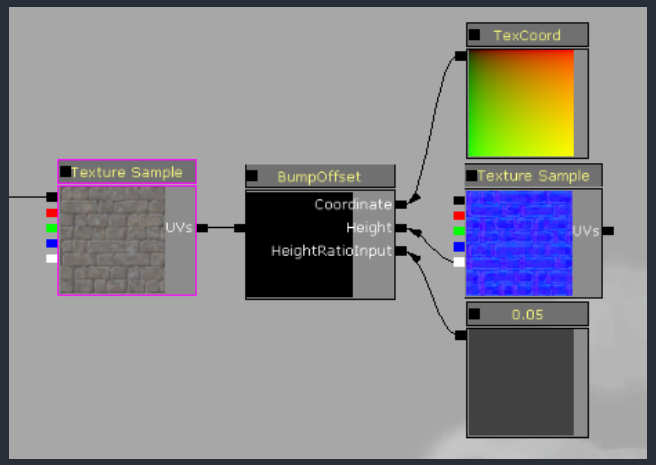




Wichtige Nodes

- Sine/Cos
  - Output = Sinus(Input) bzw. Cosinus(Input)
- BumpOffset
  - Sog. ParallaxMapping
  - Nimmt als Input eine Höheninformation und verzerrt die TextureCoordinaten passend zur Position der Kamera
- DepthBiasedAlpha
  - Pixel färben sich an Kollisionsstellen mit jeglichen Objekten dunkel
  - Wird oft bei Partikel verwendet (dazu mehr bei Partikeln)

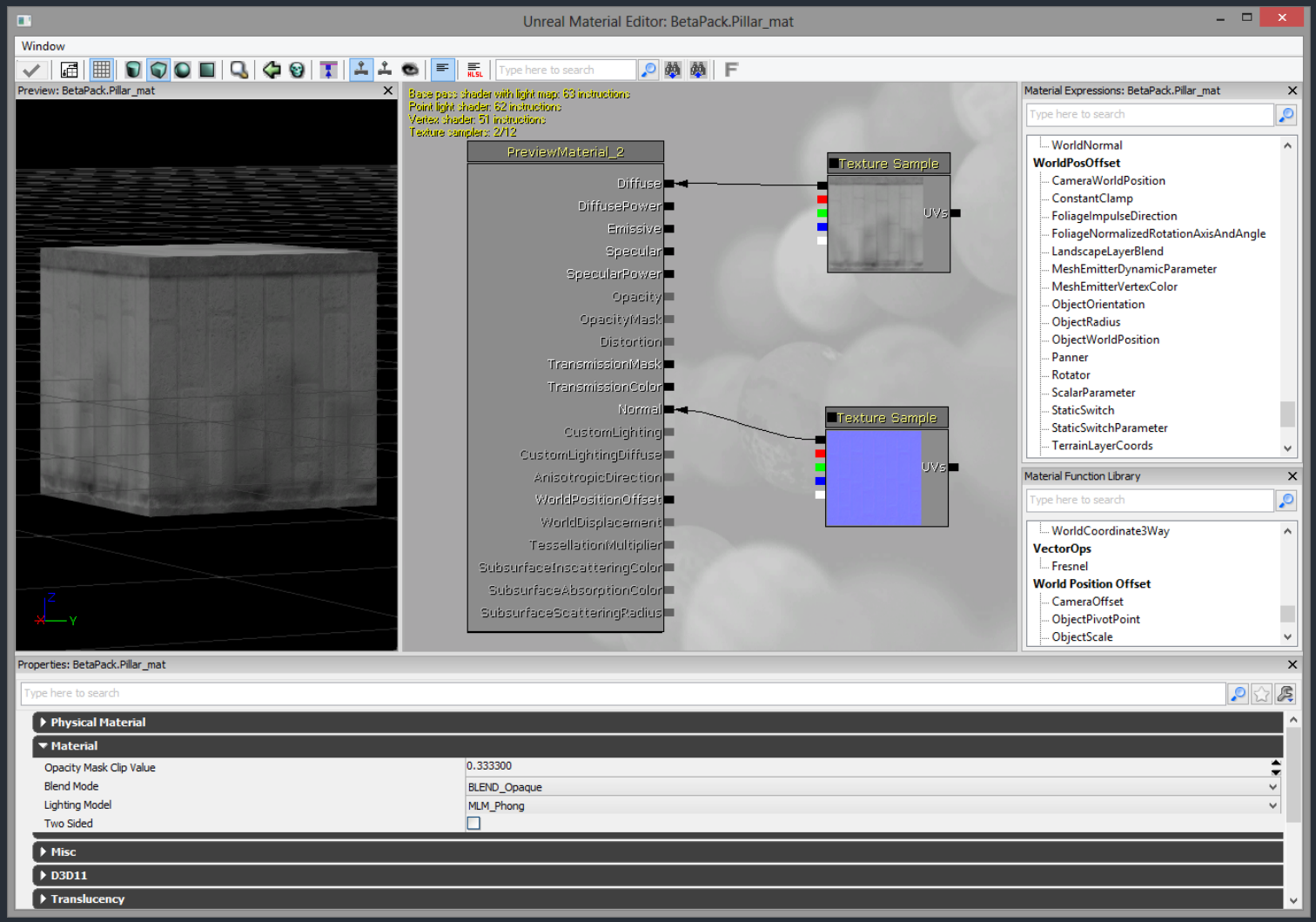




Für mehr Infos bitte UDN aufsuchen oder…

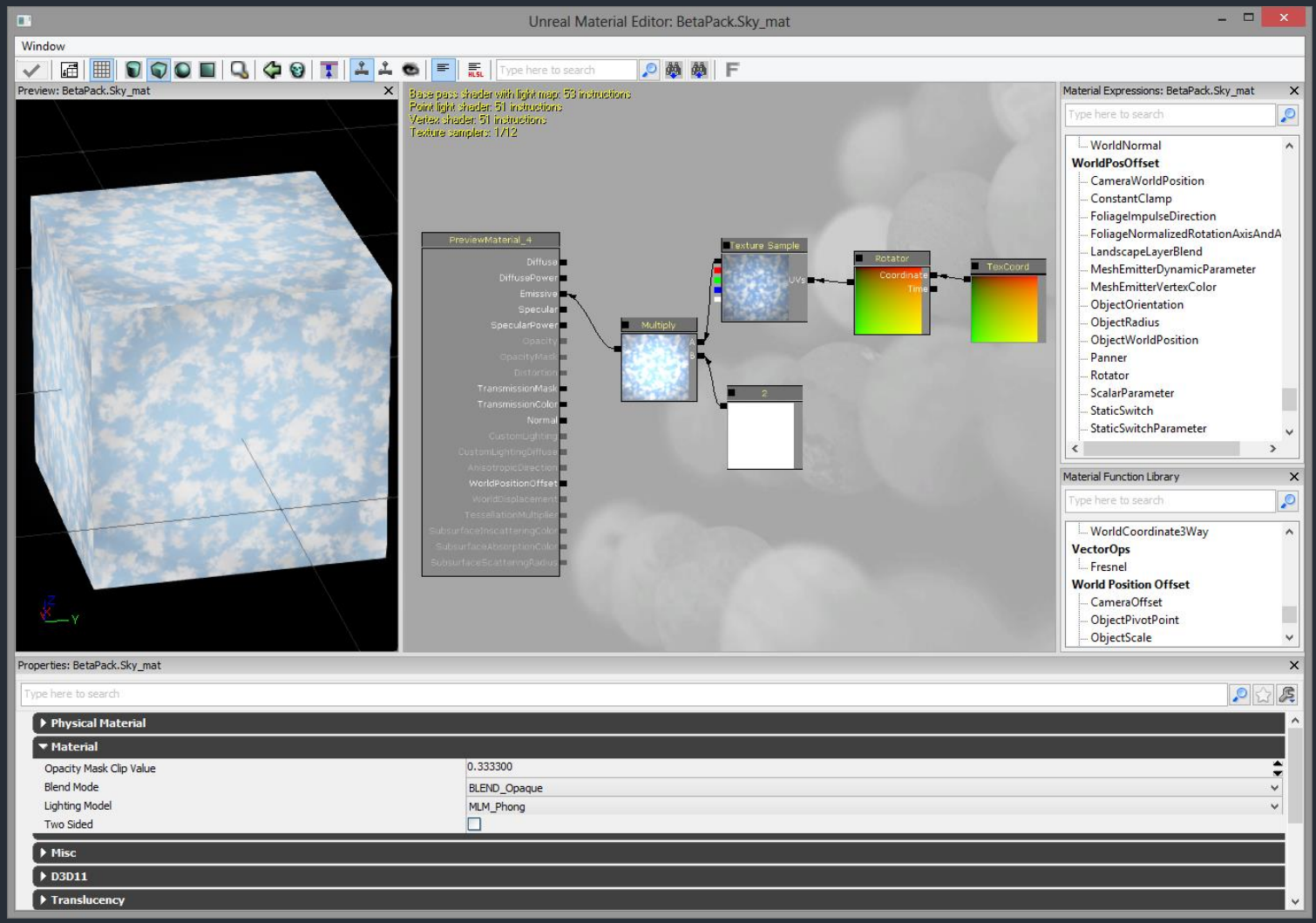
http://www.hourences.com/ue3-mat-expressions/

Beispiel: Pillar_mat



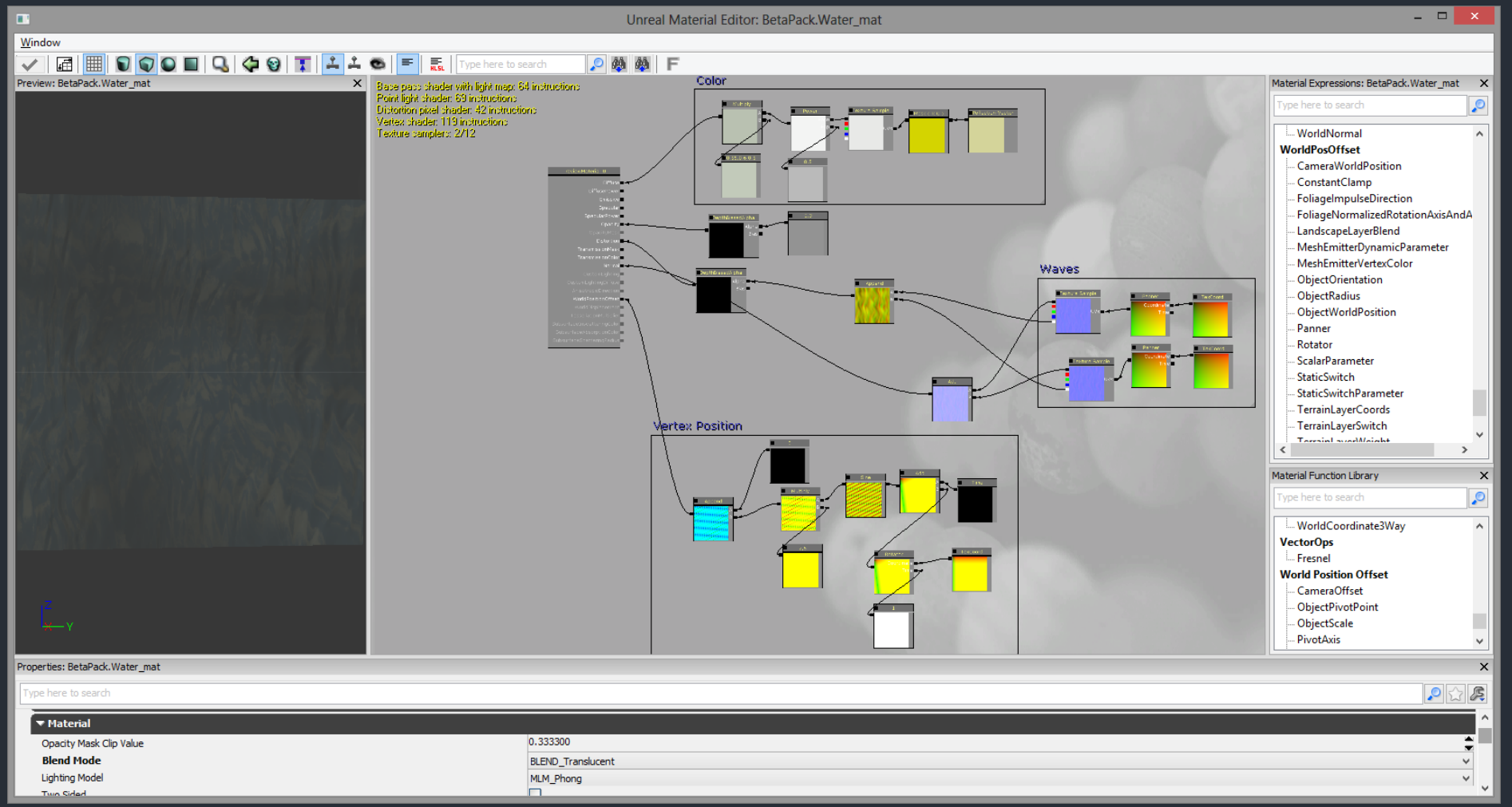
(Erklärung im Tutorium)

Beispiel: Sky_mat



(Erklärung im Tutorium, Tipp: „Cast Shadow“ im StaticMesh deaktivieren)

Beispiel: Water_mat



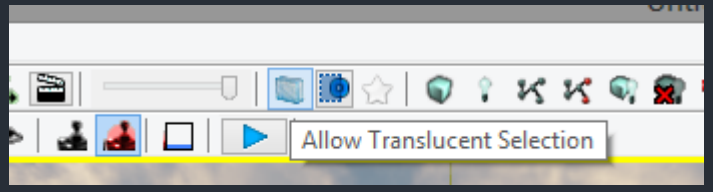
(Erklärung im Tutorium)

Reflections (grobe Übersicht)

- Um Reflektionen gibt es verschieden Möglichkeiten, dies ist nur eine kleine Auswahl
  - Single-Textur
    - Per ReflectionVektor oder per BumpOffset wird eine normale Textur verzerrt
  - (Real Time) Cubemap
    - Kombination aus 6 Texturen, welche eine Reflektion von genau einem Punkt im Raum aus darstellt
  - Scene Capture Reflect Actor
    - Ermöglicht realistische und performante Reflektion auf planaren Flächen

Bermerkungen

- Das UDK erkennt bei .PNG ob ein Alpha-Kanal verwendet wurde oder nicht. Falls nein wird DXT1 angewandt bei der Komprimierung. Gibt es einen Alpha-Kanal wird DXT5 verwendet
- Viele Input-Pins im Material Editor müssen nicht unbedingt belegt werden. Z.B. liegt an einem TextureSample immer ein nicht sichtbares TextureCoordinate an
- Um StaticMeshes mit translucenten Materialien selektieren zu können:



Hausaufgabe

Erstellt eine Wiese! In den Assets findet ihr Grass_diff-alpha und Grass_mesh. Erstellt ein passendes Material und füllt die Wiese entweder manuell mit StaticMeshes oder per FoliageTool

Inhalt

1. Einleitung und Editor
2. Geometry und Lighting
3. StaticMeshes und Materials
4. Kismet
5. Particles

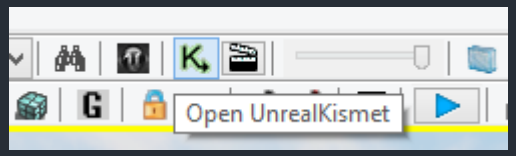
Bitte „myGammaMap.udk“ im UDK laden

Kismet

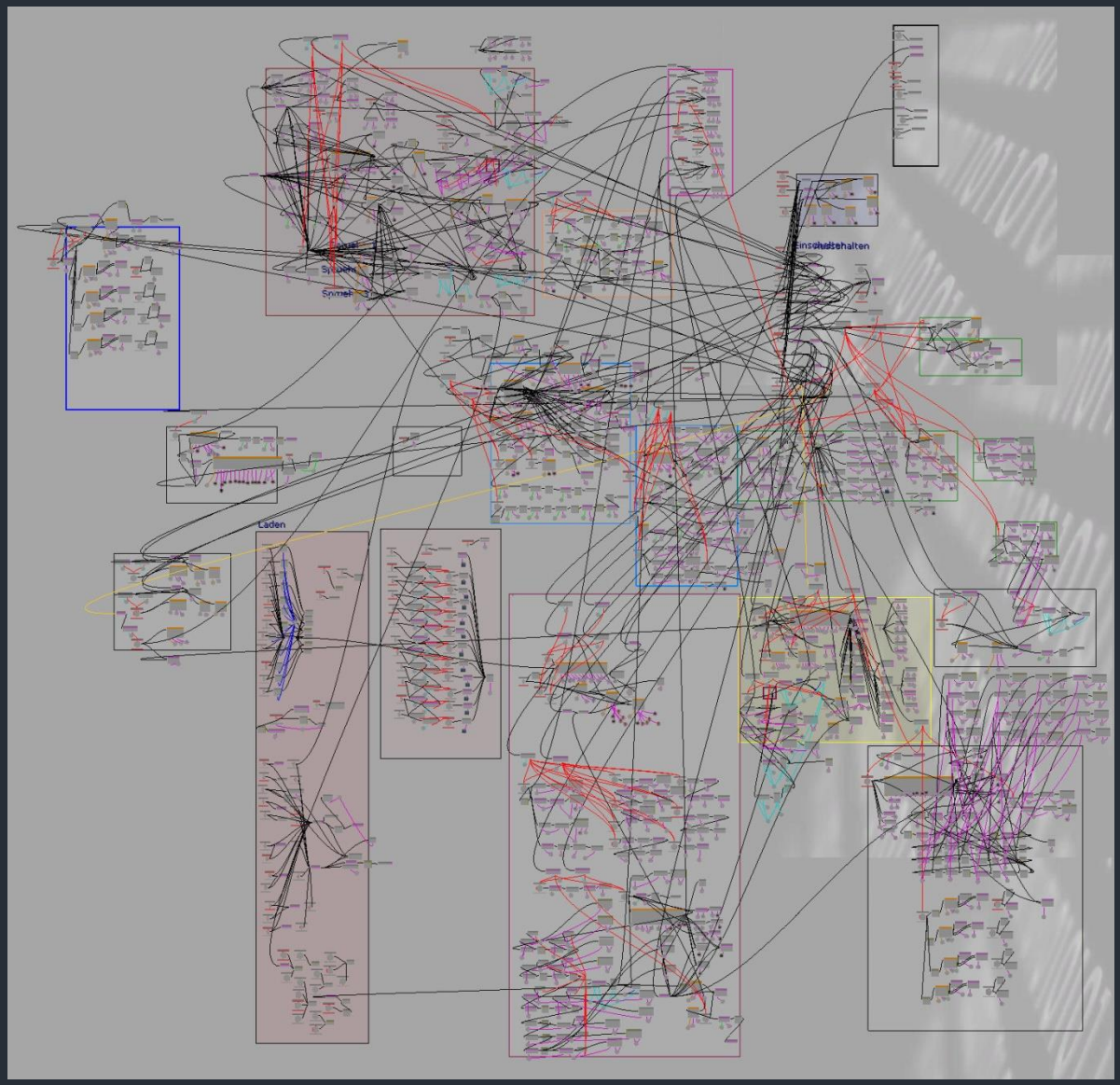
- Was ist Kismet?
- Events
- ActorClassBrowser
- AttachTo
- Matinee
- Licht per Matinee steuern
- Fahrstuhl per Matinee steuern
- Kamerafahrt mit Matinee

Was ist Kismet?

- Kismet wurde mit der Unreal Engine 3 eingeführt und soll das Scripting von Spielgeschehen selbsterklärend und übersichtlich ermöglichen
- Wie bei den Materialien werden Nodes benutzt, welche wiederum In- und Ouput-Pins haben
- Funktioniert ähnlich einem State-Chart
- Kismet-Graphen werden im Mapfile (.udk) gespeichert



Was ist Kismet?

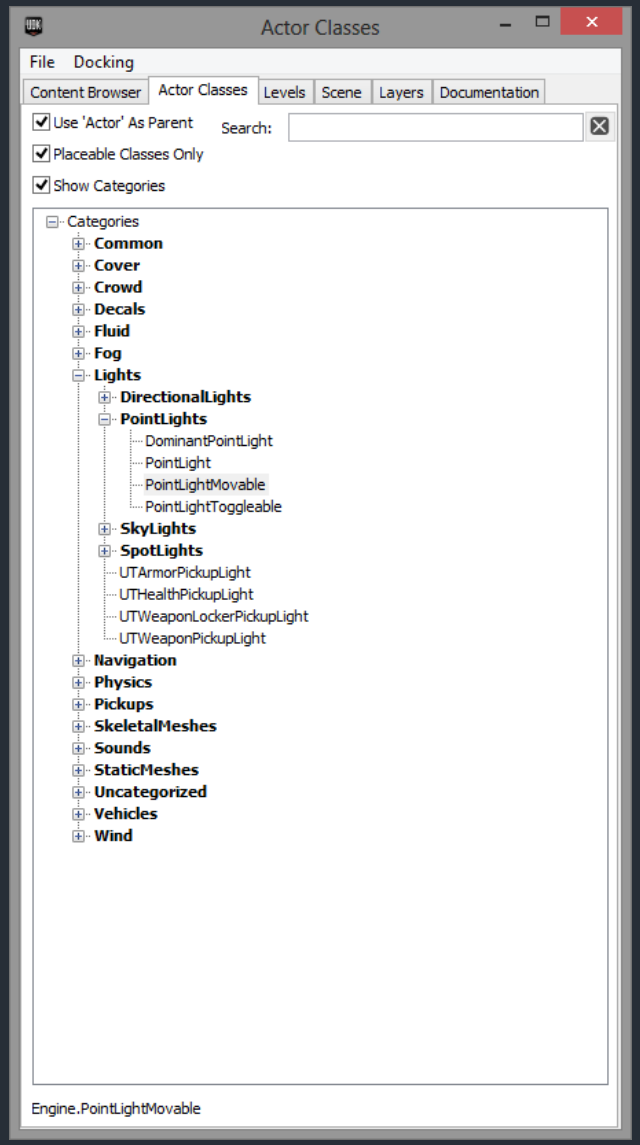


Events

- Die Nodes müssen durch ein Event ausgelöst werden
- Zum einen gibt es Events wie „Level Loaded“ o.ä., sodass dieses Event direkt bei Spielstart ausgelöst wird
- Eine andere Möglichkeit bieten „Tigger“ (als Actor oder Volume). Diese lösen z.B. bei Berührung durch den Spieler ein Event aus und können eine Nodekette in Kismet aktivieren

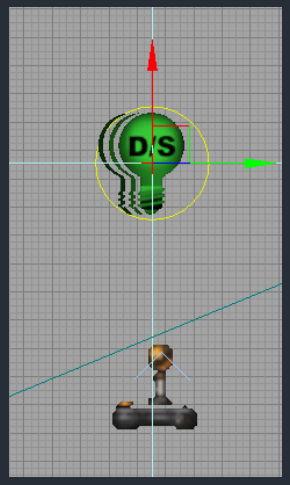
ActorClassBrowser

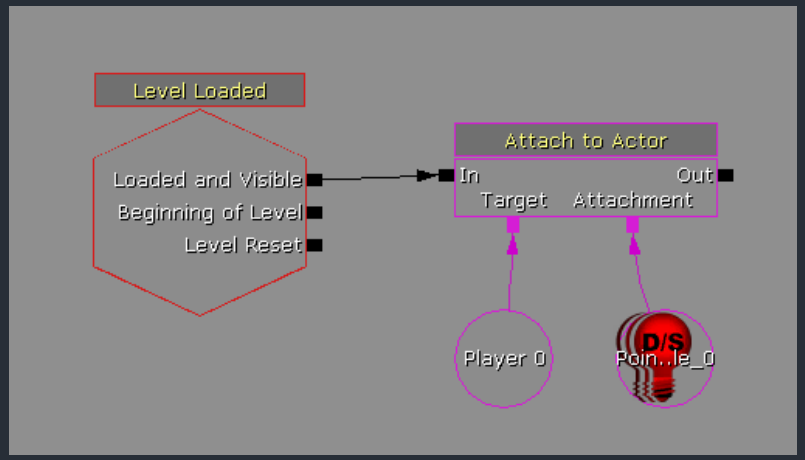
- Im Browser ist oben in den Tabs auch der ActorClassBrowser (ACB) zu finden, in dem alle verfügbaren Actorklassen gelistet sind
- Per Drag'n'Drop können sie in die Welt eingefügt werden



AttachTo

- „AttachTo“ ermöglicht es einen Actor an einen anderen dranzuhängen
- So lässt sich z.B. ein dynamisches Licht an den Spieler anhängen

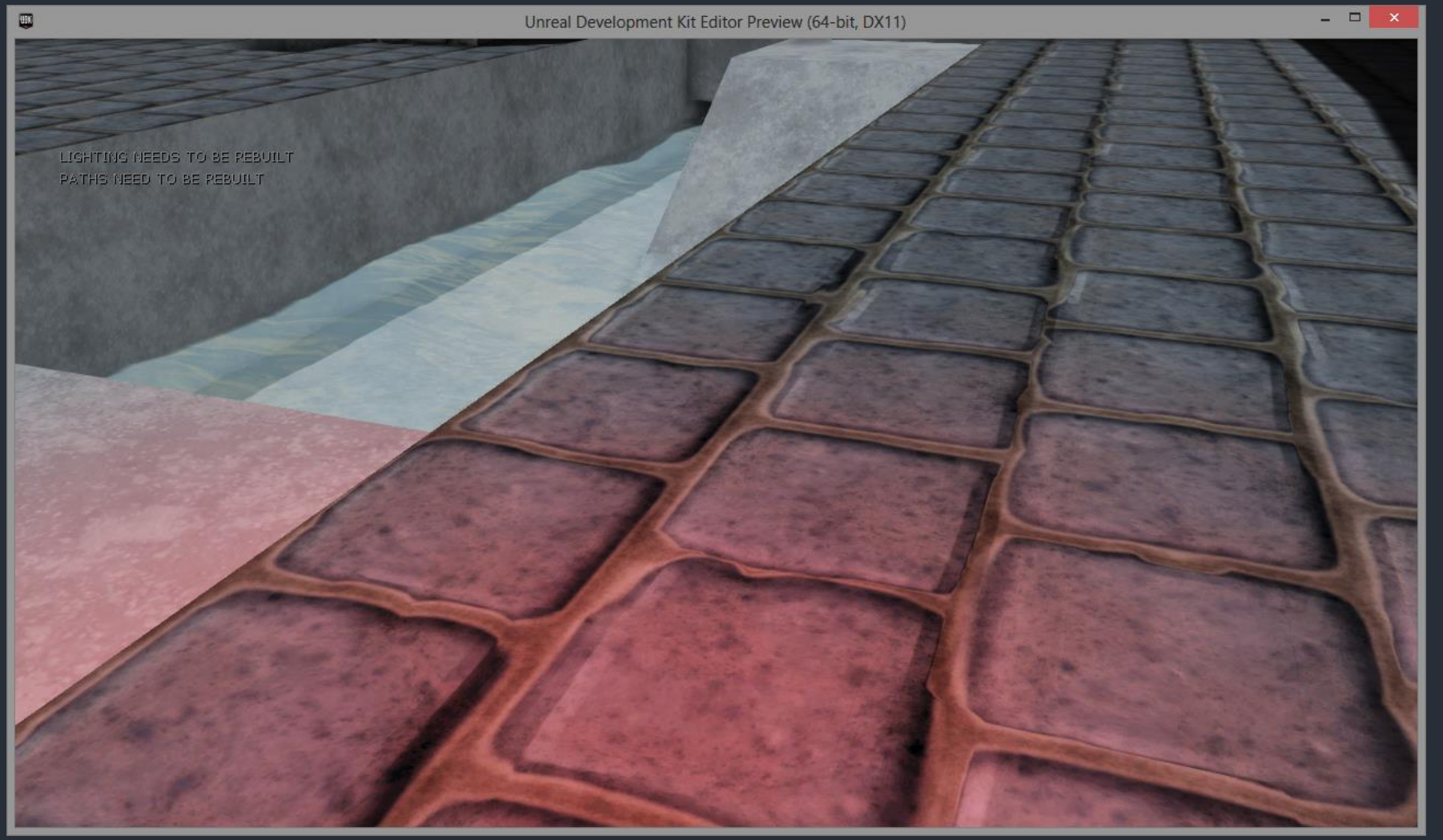




AttachTo

- Kurzanleitung
  - ACB → Lights → PointLights → PointLightMovable
  - Light Actor Properties → LightColor zu Rot und Radius anpassen
  - Kismet öffnen
  - Rechtsklick → New Event → Level Loaded
  - Rechtsklick → New Action → Actor →Attach To Actor
  - Rechtsklick → New Variable → Player → Player
  - Point Light in Welt selektieren → Rechtsklick in Kismet → New Object Var Using…
  - Alles verknüpfen wie auf letzter Folie gezeigt

AttachTo


Unreal Development Kit Editor Preview (64-bit, DX11)
LIGHTING NEEDS TO BE REBUILT
PATHS NEED TO BE REBUILT

Matinee

- Bei „Matinee“ handelt es sich um eine der wichtigsten Nodes in Kismet
- Bei Doppelklick auf die Node öffnet sich ein eigener Editor, mit Spuren wie bei einer Musik-/Videobearbeitungssoftware
- So gut wie alles kann mithilfe von Keyframes und diesen Spuren gesteuert werden

Matinee

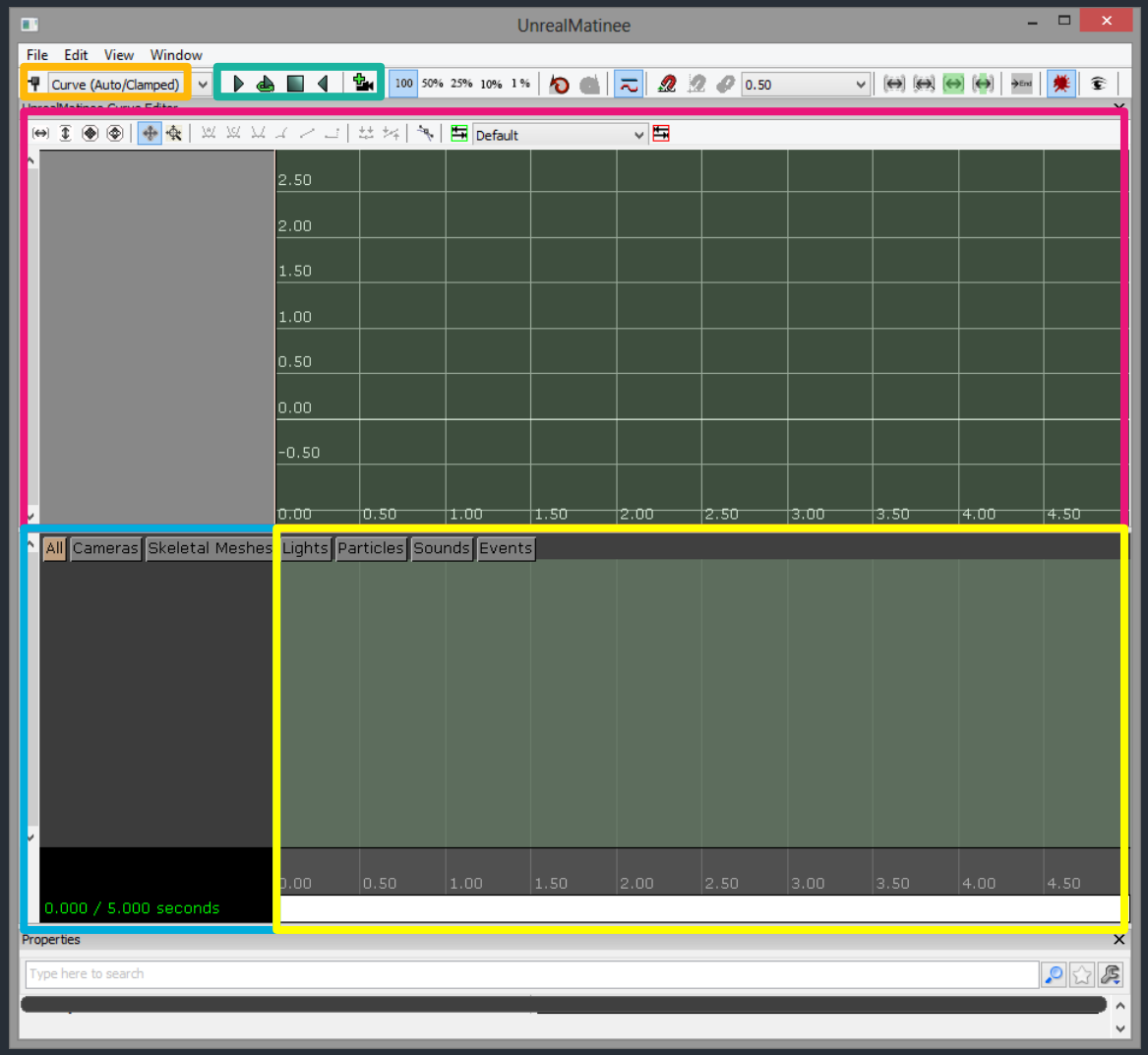
Keyframes setzen

Play/Pause/Stop /...

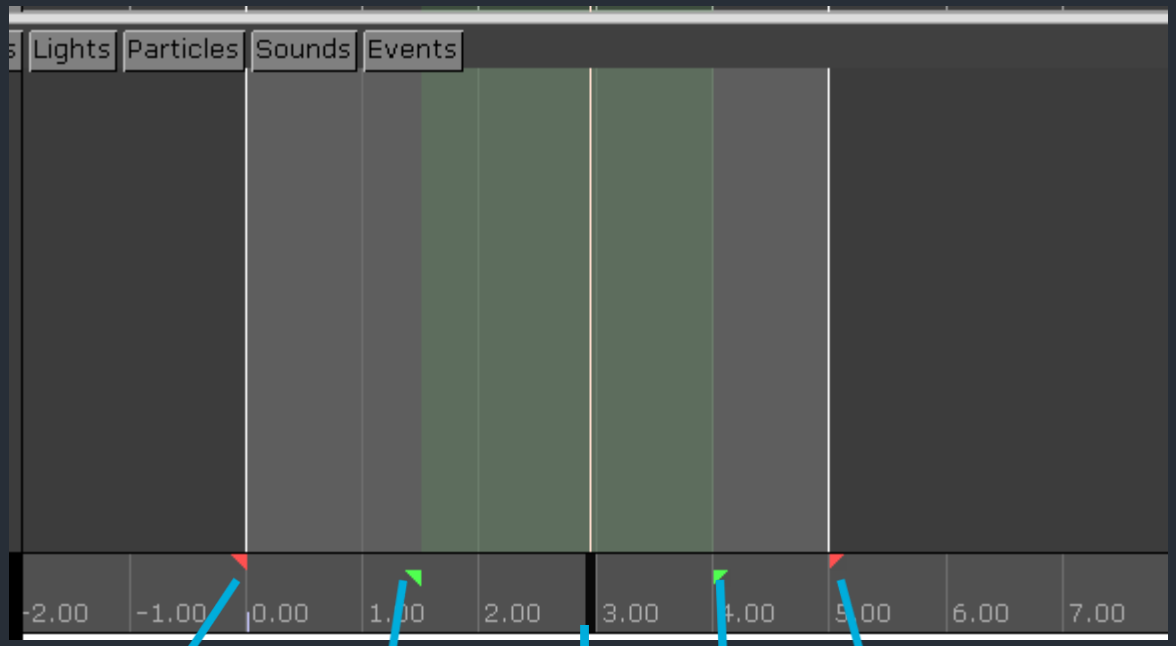
Curve Editor

Spurheader

Spuren bzw. Keyframes



Matinee



Start

Start der Vorschau

Zeiger auf aktuelle Position

Ende der Vorschau

Ende

Licht per Matinee steuern

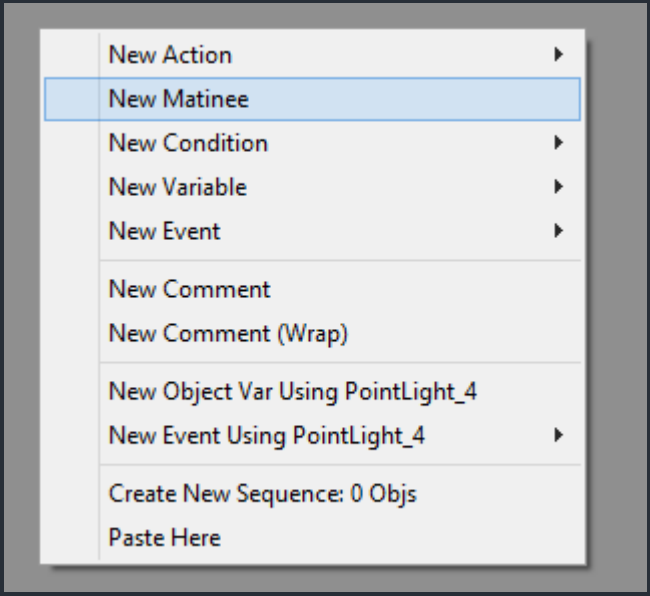
- Per Matinee kann man zum Beispiel die Lichtfarbe während der Zeit ändern
- Dazu wird ein „PointLightToggleable “ in die Welt gesetzt und mit einer Matinee-Node in Kismet gesteuert

Licht per Matinee steuern

- Kurzanleitung
  - ACB → Lights → PointLights → PointLightToggleable

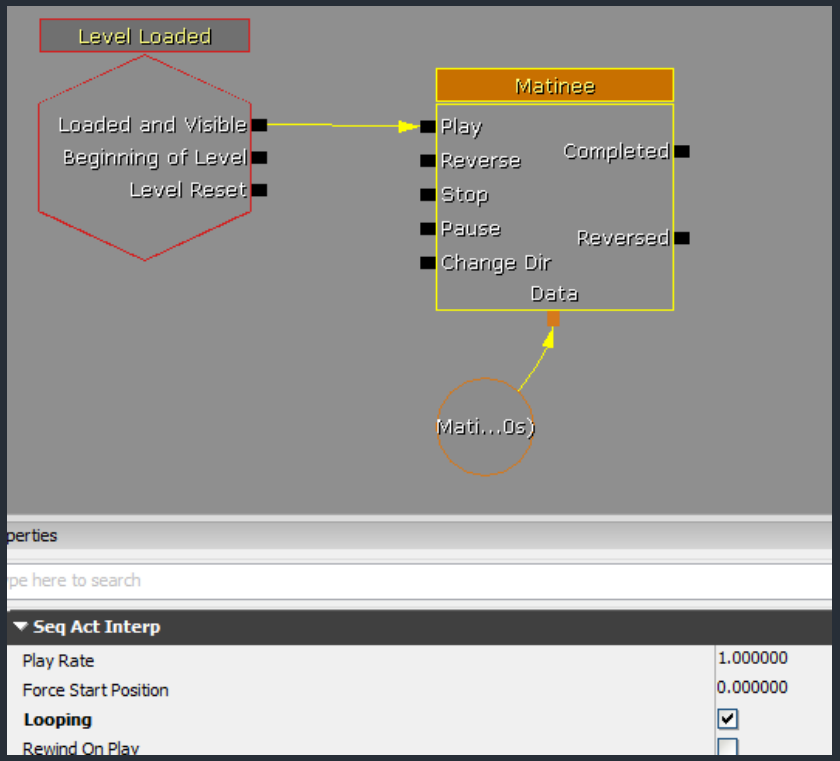
Licht per Matinee steuern

- Kismet öffnen, Rechtsklick und…



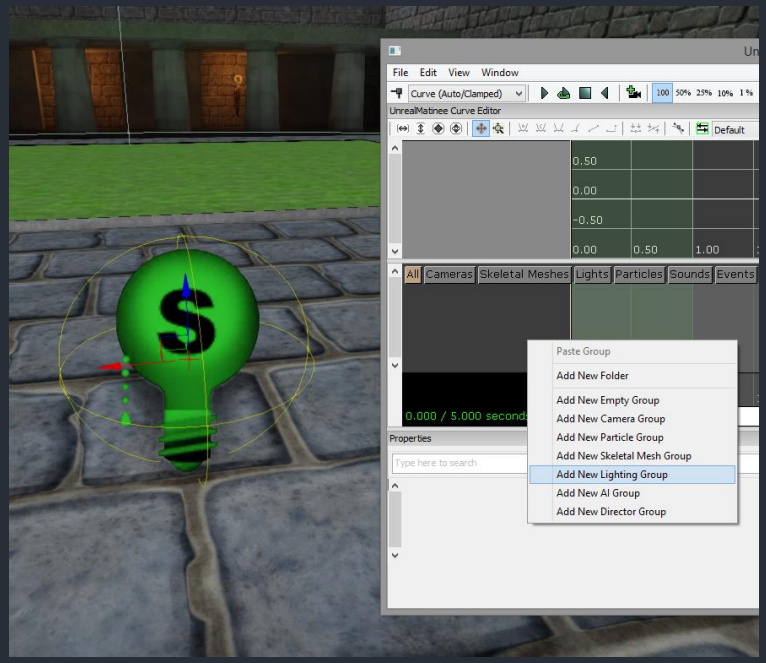
Licht per Matinee steuern

- Wie im Bild einrichten und Matinee in dessen Properties das Looping aktivieren



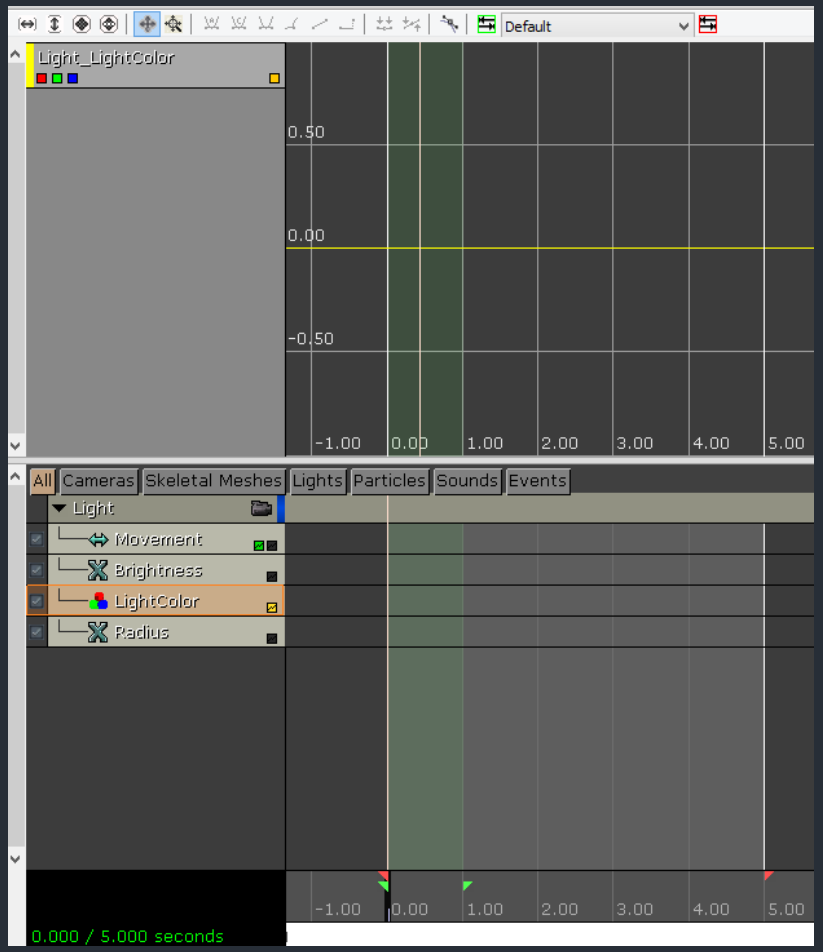
Licht per Matinee steuern

- Matinee-Editor per Doppelklick auf das Matinee-Node öffnen
- Licht in der Welt selektieren und Rechtsklick in das dunkelgrauen Bereich und „Add New Lighting Group“ wählen



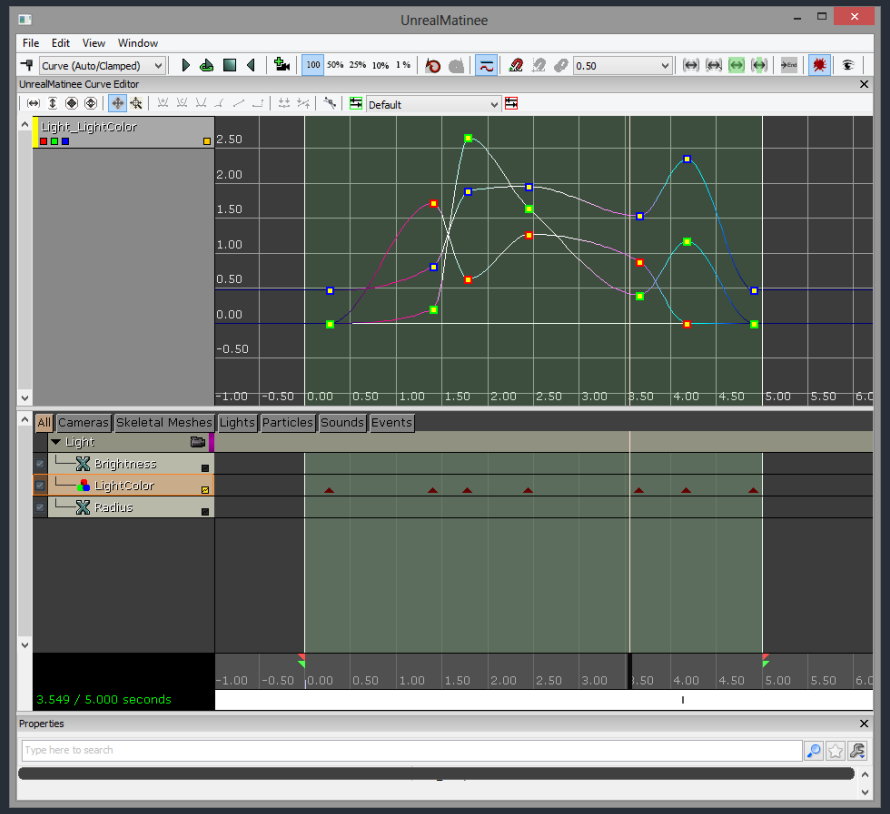
Licht per Matinee steuern

- Durch Markieren des kleinen Kästchens beim Header der LightColor werden die Werte im Curve Editor (pro Keyframe) steuerbar
- Die Movement-Spur per Rechtsklick und Pop-Up Menü löschen (brauchen wir nicht und macht nur Probleme)



Licht per Matinee steuern

- Zeiger verschieben und mit Knopf ganz oben links neue Keyframes setzen
- Keyframes (unten oder im Graphen) können per Linksklick selektiert und bei gedrücktem STRG + Linker Maustaste bewegt werden

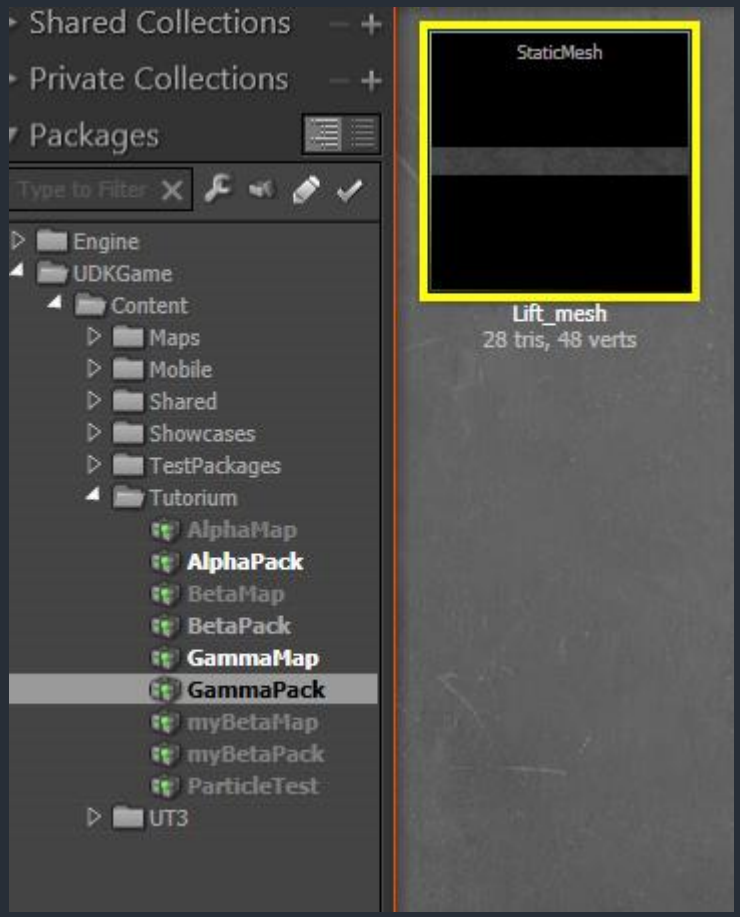


Fahrstuhl per Matinee steuern

- Fahrstühle lassen sich ebenfalls per Matinee verwirklichen
- Dafür wird ein „InterpActor“ als Fahrstuhl und einen Trigger für die Steuerung benötigt
- Der „InterpActor“ ist ähnlich einem StaticMesh (man wählt für einen InterpActor einen StaticMesh aus dem Browser, nur ist der dann in der Welt als Actor nicht statisch) und wird automatisch dynamisch beleuchtet

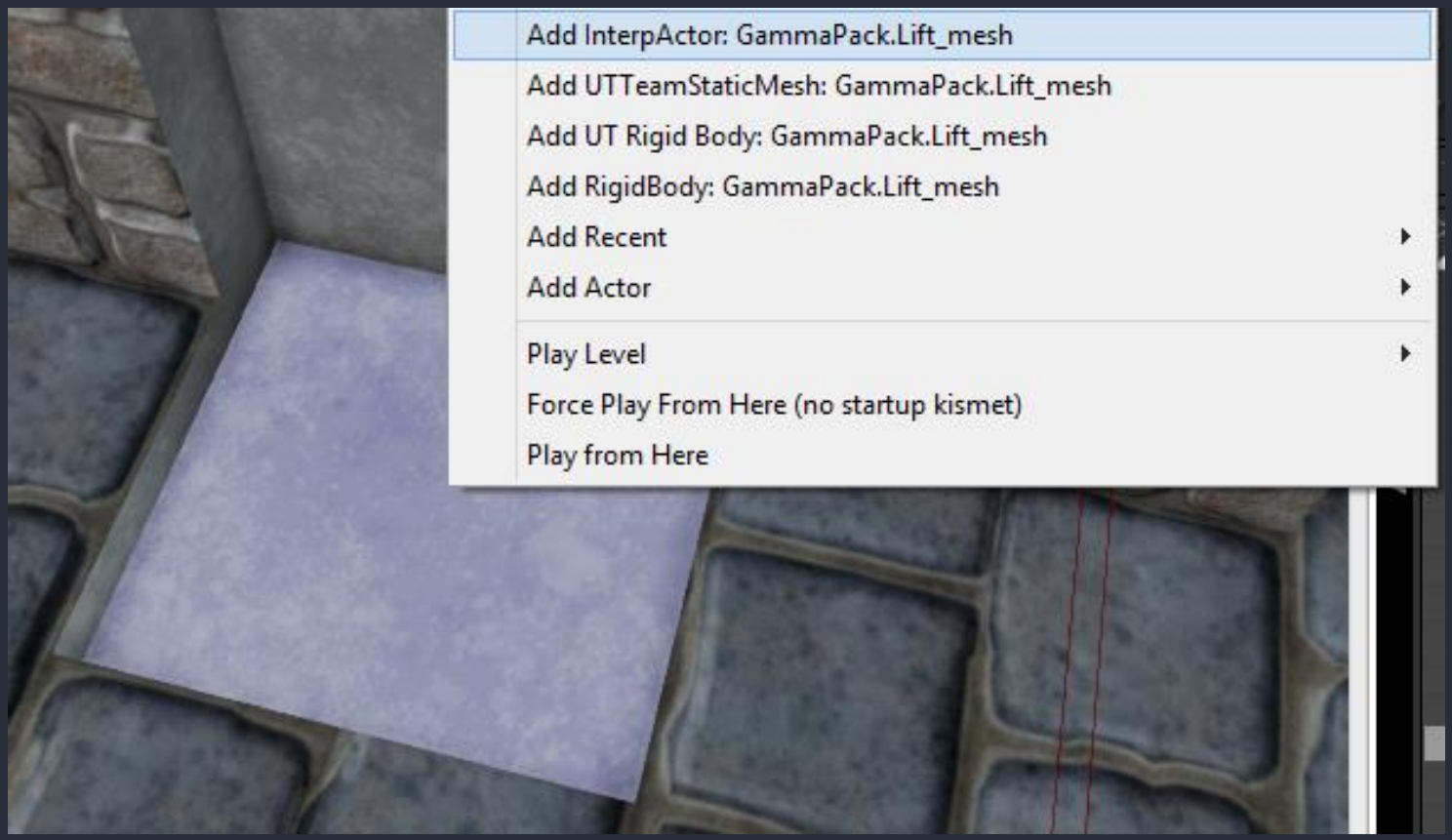
Fahrstuhl per Matinee steuern

- Kurzanleitung
  - ContentBrowser → „GammaPack.upk“ → Lift_mesh selektieren

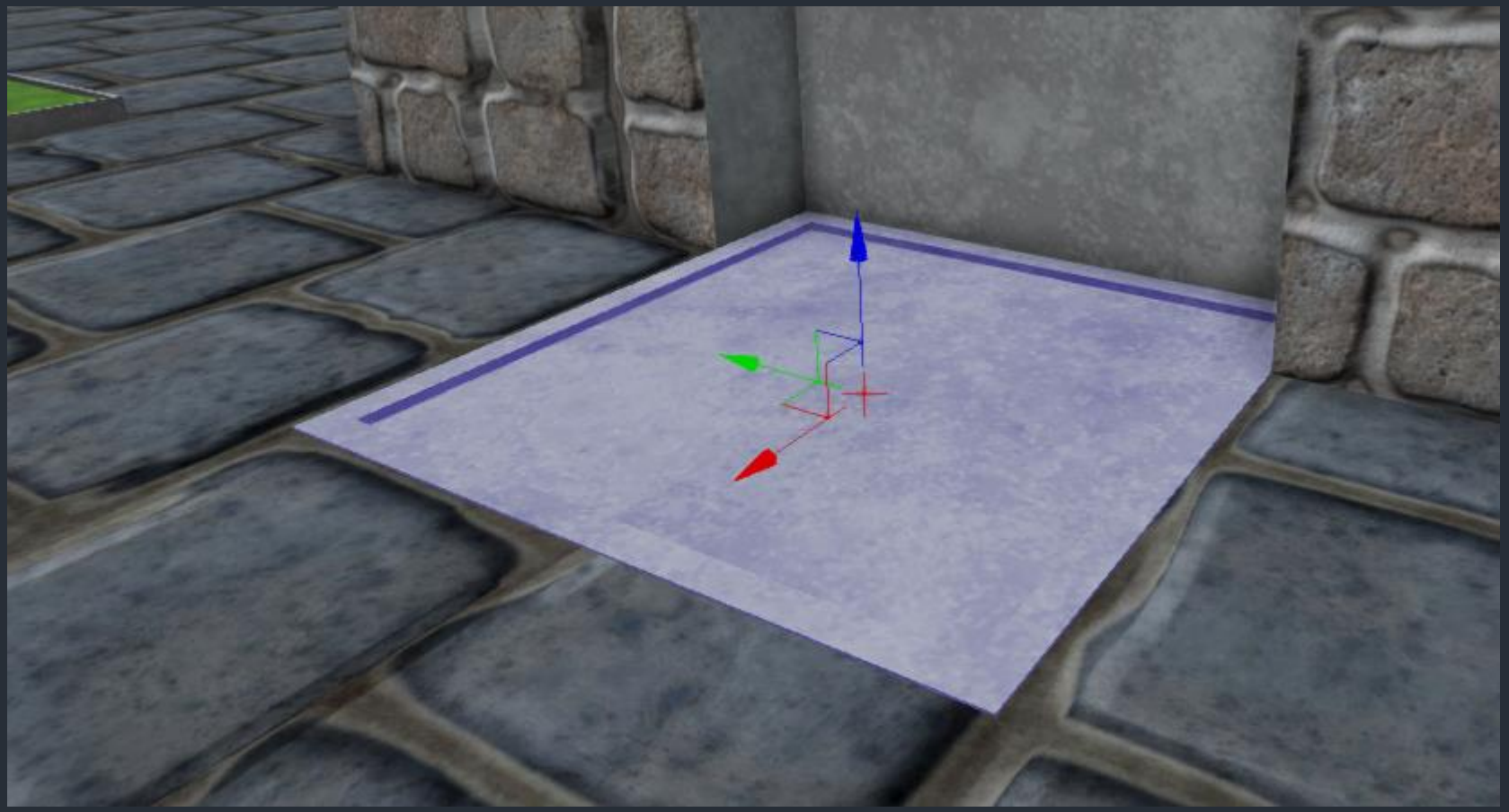


Fahrstuhl per Matinee steuern

- Rechtslick in die Welt und „Add InterpActor…“

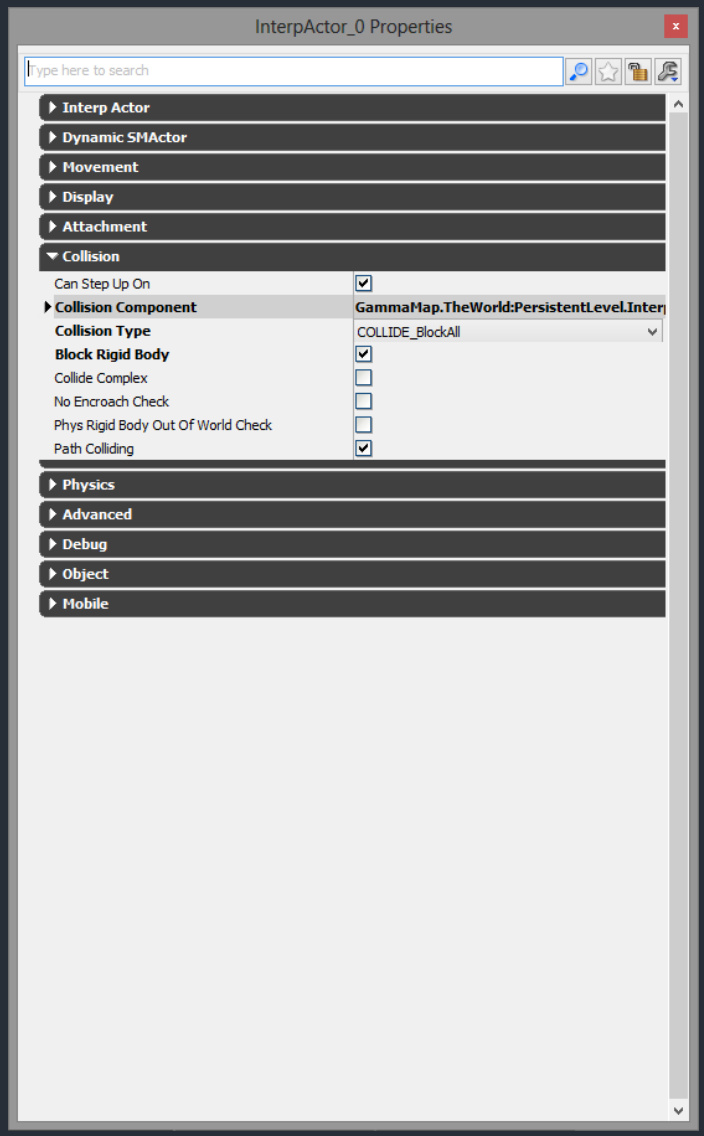


Fahrstuhl per Matinee steuern



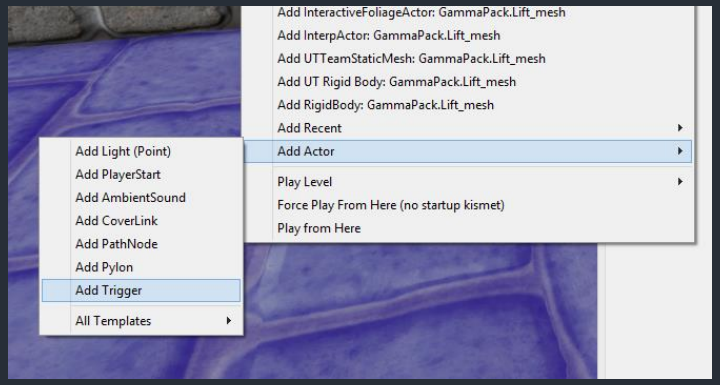
Fahrstuhl per Matinee steuern

- Collision Type des InterpActors auf „COLLIDE_BlockAll“ stellen

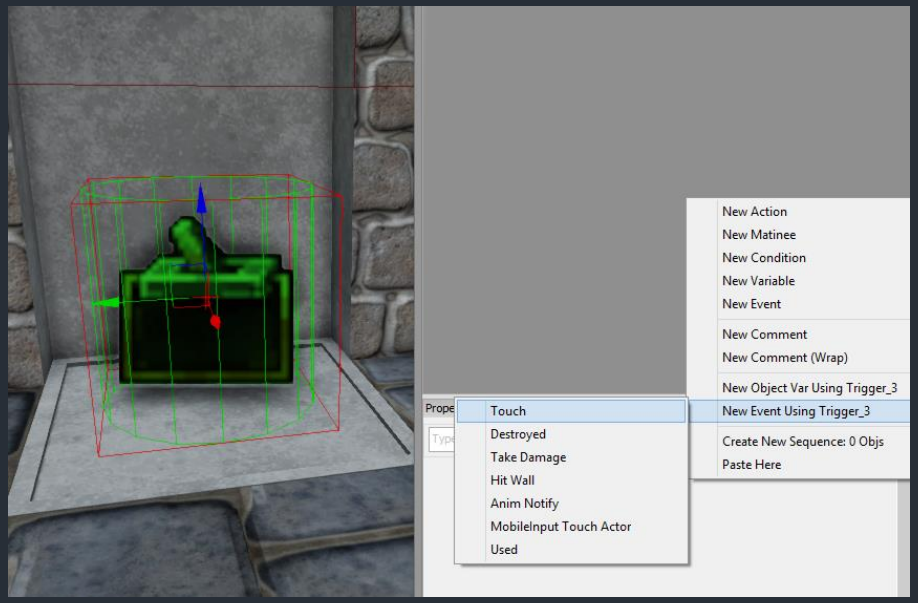


Fahrstuhl per Matinee steuern

- Trigger in die Welt einfügen

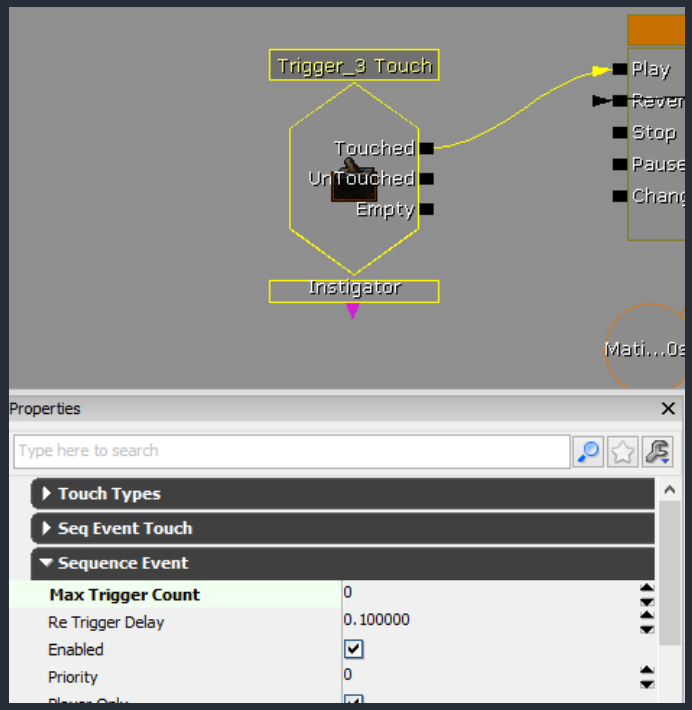


- Tigger in Kismet als Event einbringen



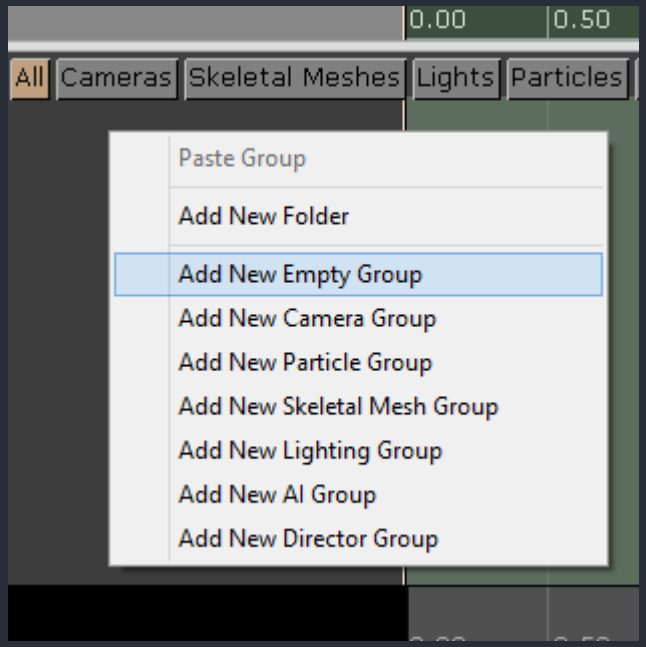
Fahrstuhl per Matinee steuern

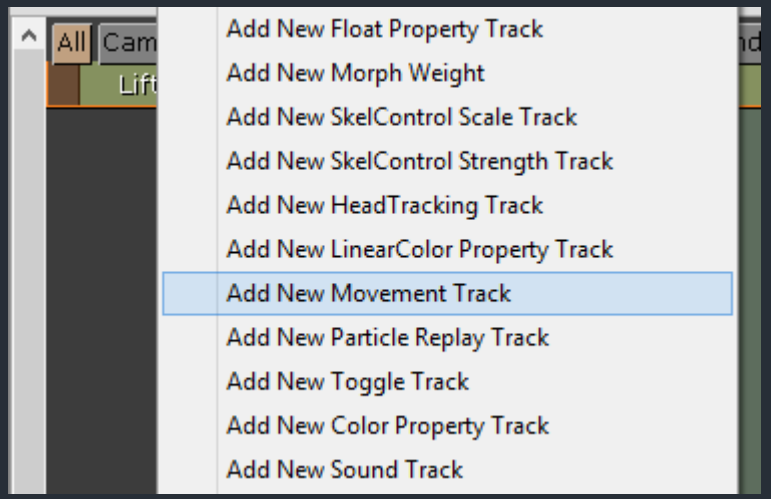
- In Properties des Nodes vom Trigger den „Max Trigger Count“ auf 0, was als unendlich interpretiert wird



Fahrstuhl per Matinee steuern

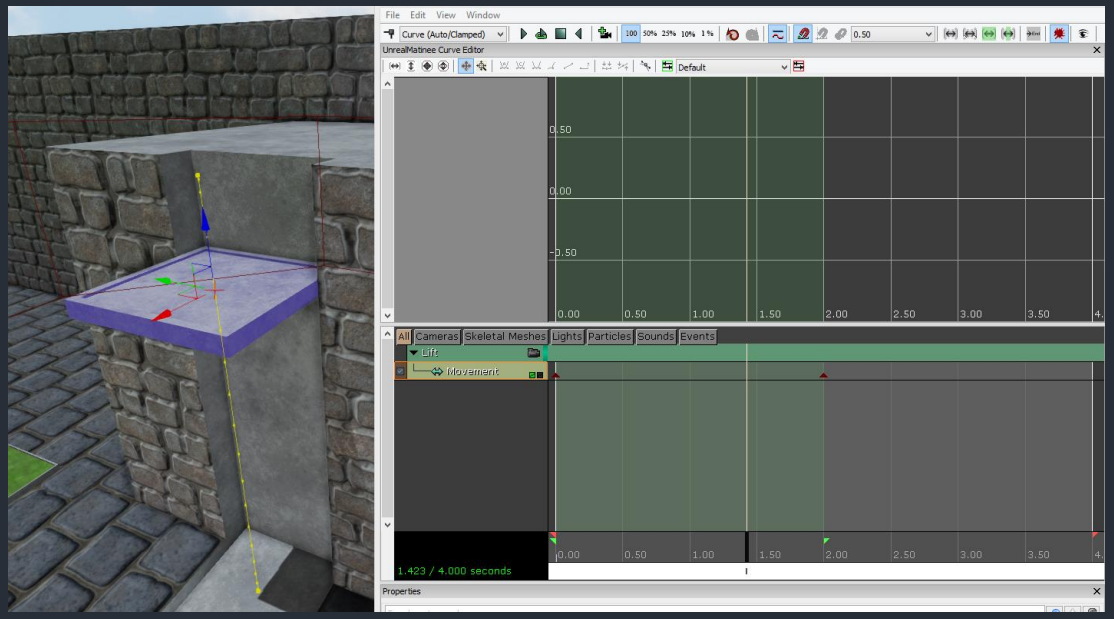
- Matinee-Editor öffnen und Empty Group mit Movement Track erstellen. Dabei den InterpActor in der Welt selektiert haben. Ansonsten nachträglich manuelles verlinken in Kismet…





Fahrstuhl per Matinee steuern

- Ersten Keyframe setzen (Position unten)
- Fahrstuhl in der Welt nachoben verschieben
- Zweite Keyframe einige Sekunden später setzen (Position oben)

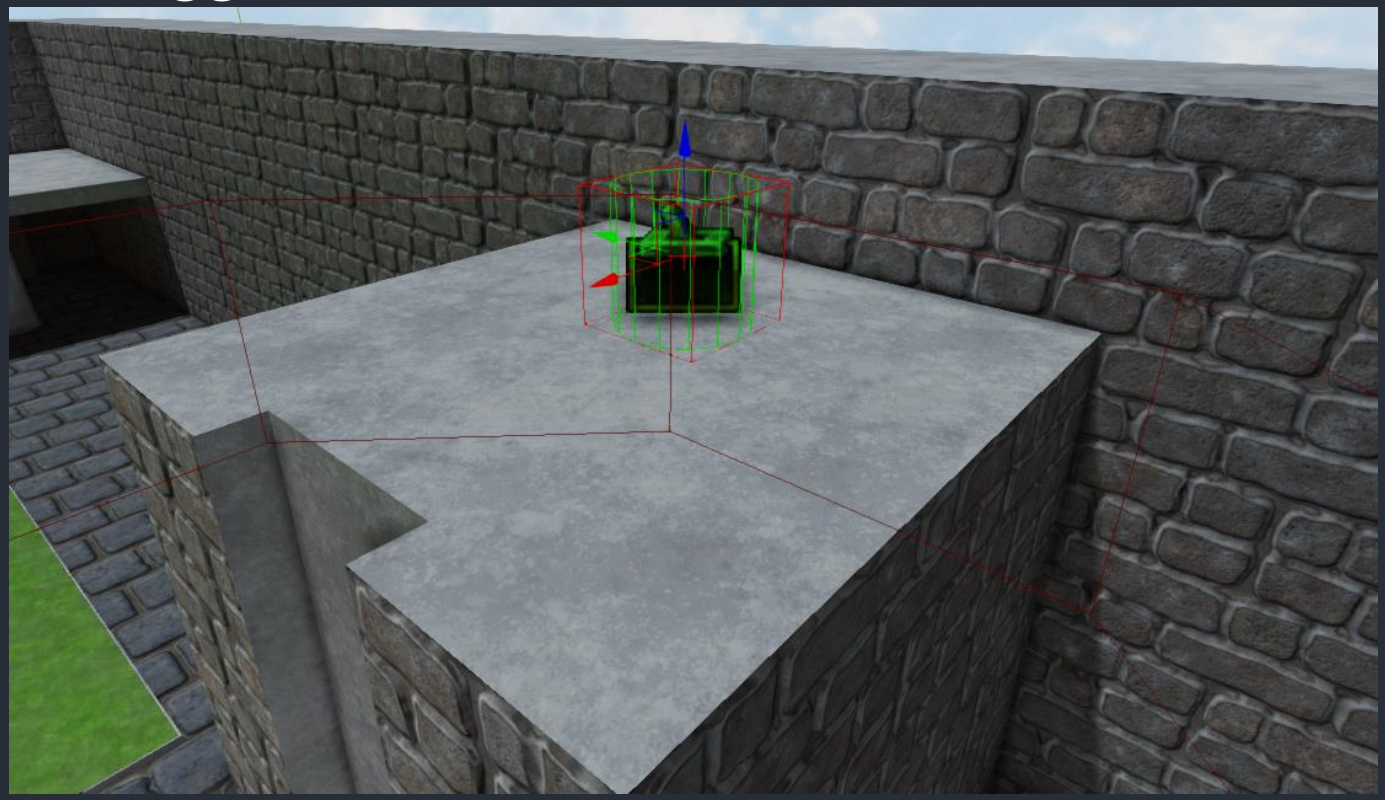


Kamerafahrt mit Matinee

- Kamerafahrt durch die Welt ist mit Matinee in Kombination mit speziellen Toggle Cinematic -Nodes möglich
- Ein Camera Actor wird wie schon der Fahrstuhl durch die Welt bewegt
- Im Matinee Node lässt sich die dann Camera, bzw. deren Spur, auswählen

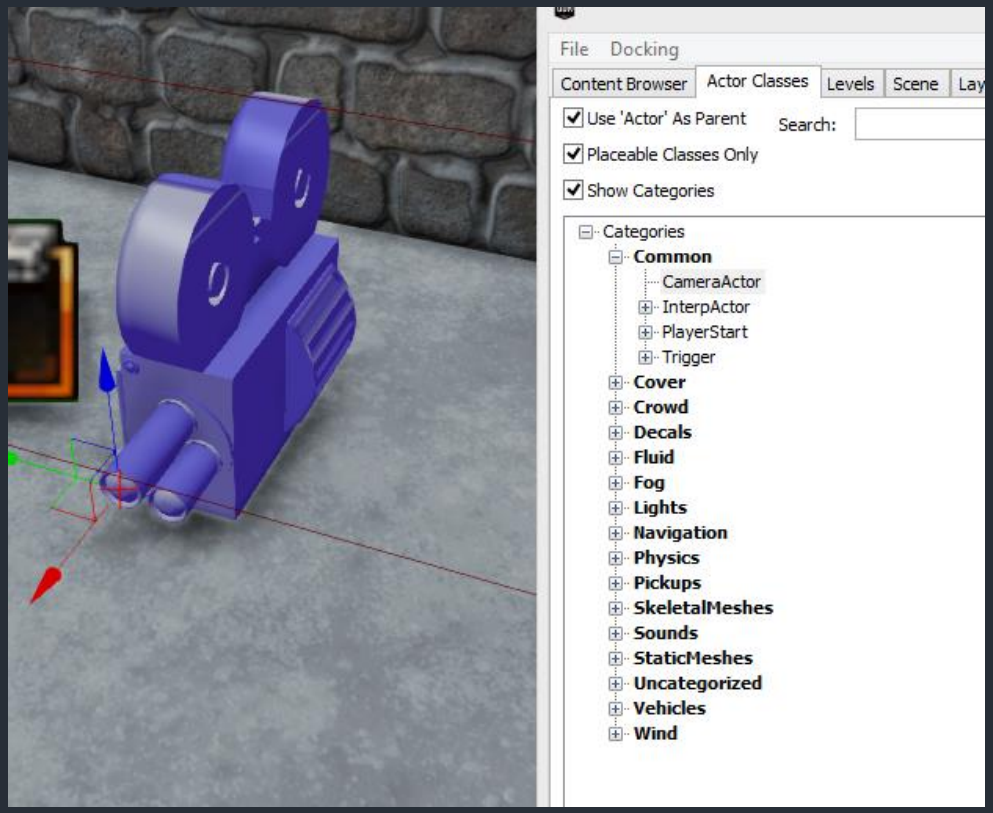
Kamerafahrt mit Matinee

- Kurzanleitung
  - Trigger setzen um Kamerafahrt auszulösen



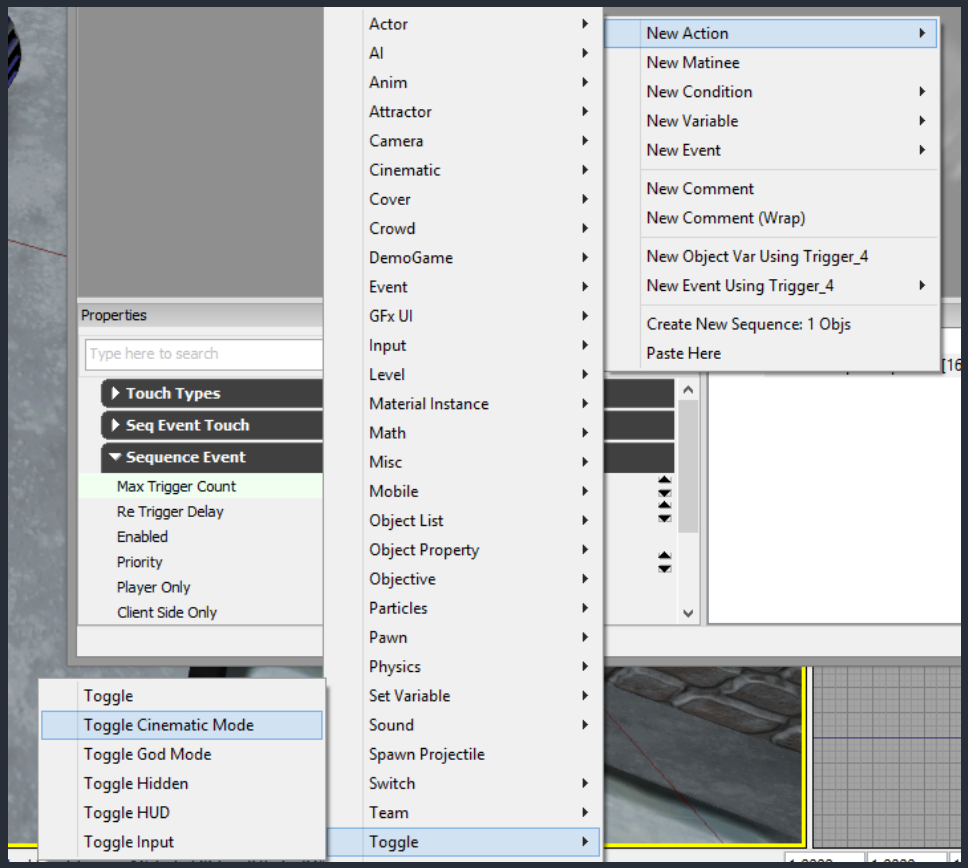
Kamerafahrt mit Matinee

- Camera Actor setzen



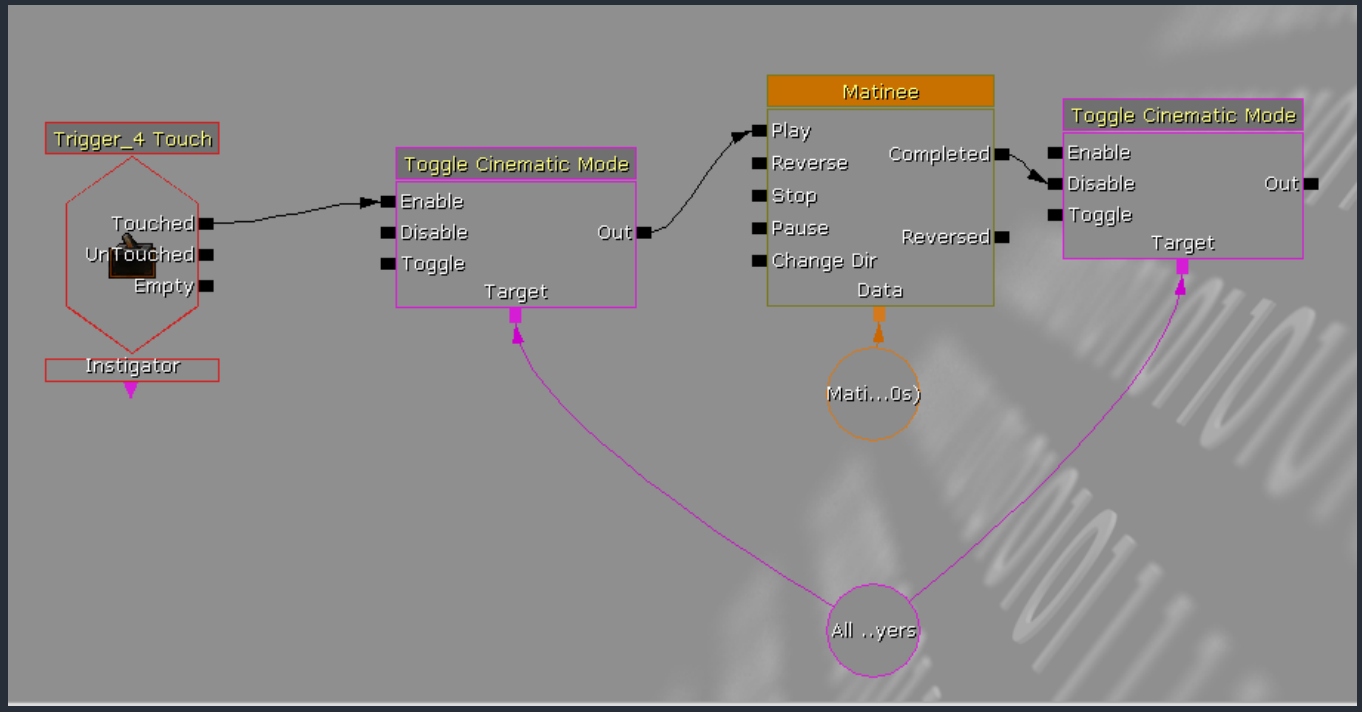
Kamerafahrt mit Matinee

- In Kismet werden 2 „Toggle Cinematic Mode“ benötigt



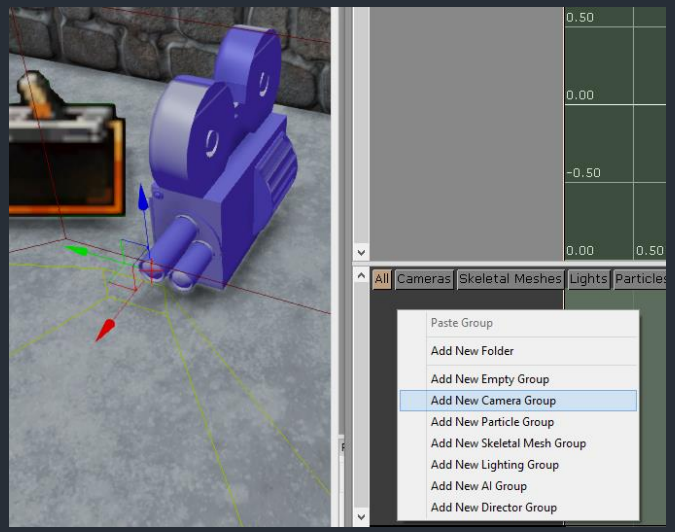
Kamerafahrt mit Matinee

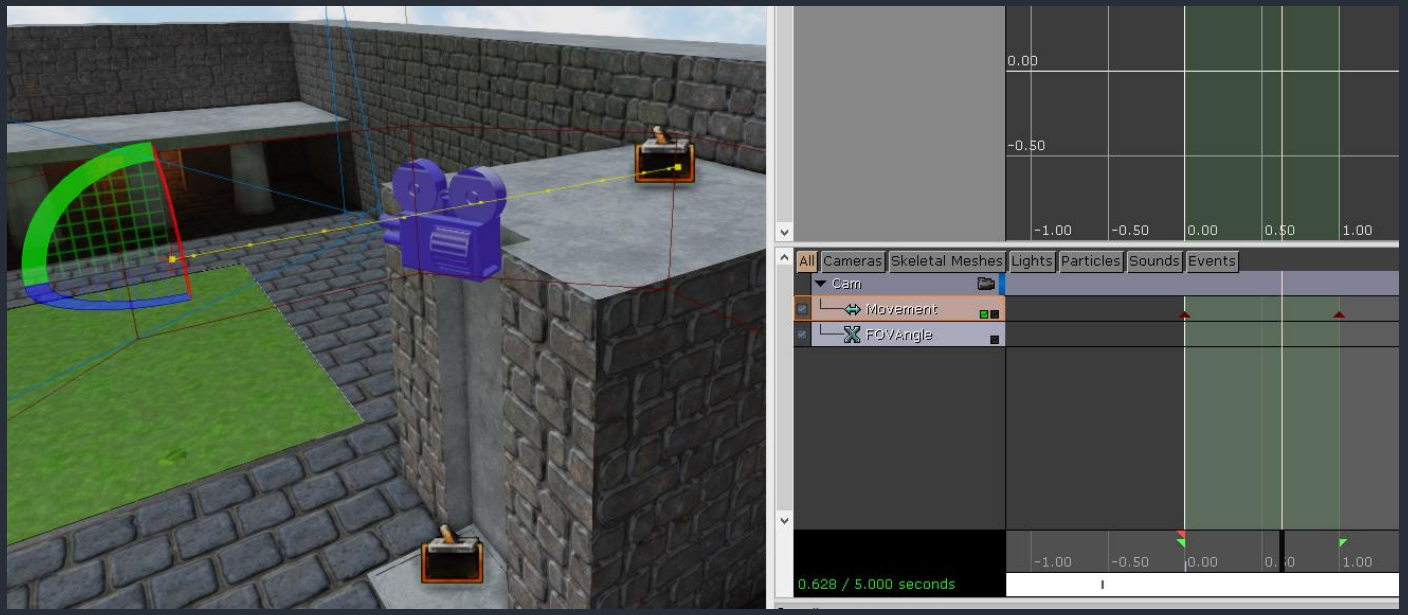
- Noch den Trigger als Event einfügen und ein neues Matinee erstellen



Kamerafahrt mit Matinee

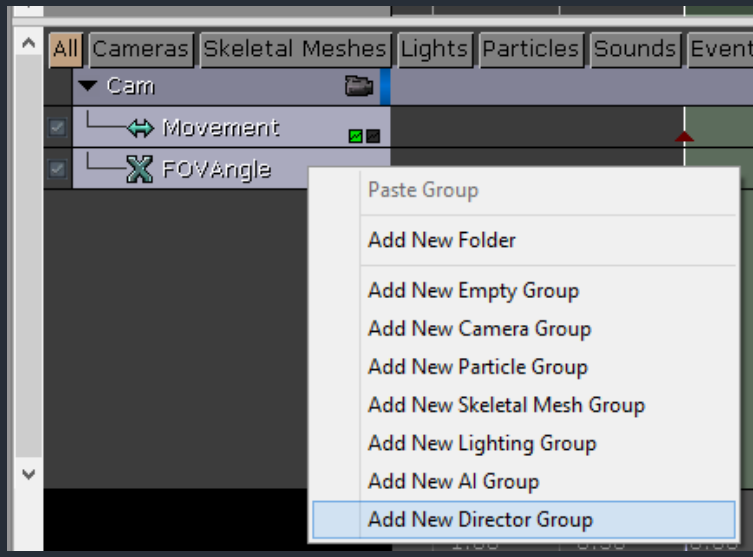
- Camera Actor in Welt selektieren und im Matinee-Editor eine Camera Group erstellen
- Kamera per Movement Track animieren

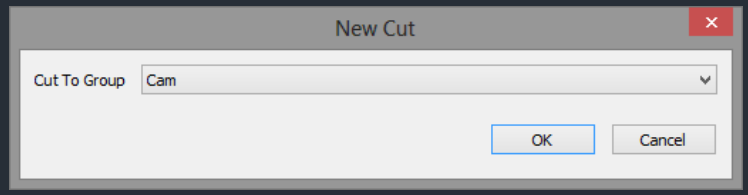




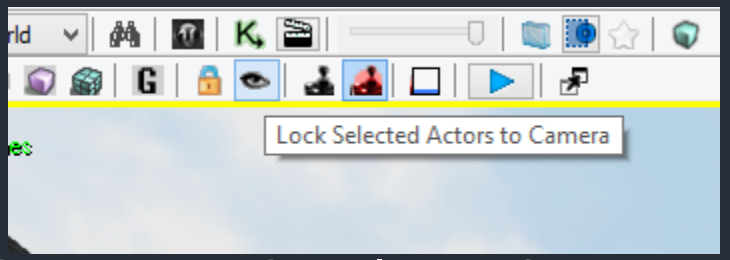
Kamerafahrt mit Matinee

- Eine Director Group erstellen, welche Kameras verwaltet und Effekte hinzufügen kann
- Die Director Group Spur wählen und einen Keyframes setzen. Daraufhin erscheint ein Fenster für die Auswahl, welche Camera Group benutzt werden soll





Kamerafahrt mit Matinee



- Tipps
  - Im Viewport gibt es die Funktion „Lock Selected Actors to Camera“
  - Man kann z.B. erst die Director Group erstellen, dann die Kamera hinzufügen, daraufhin diese Option wählen, sich zum gewünschten Standpunkt verschieben und einen Keyframe für die Movement Spur der Kamera setzen
  - Nachdem man fertig ist unbedingt die Option deaktvieren, sonst verschiebt man ausversehen Dinge…

Bemerkungen

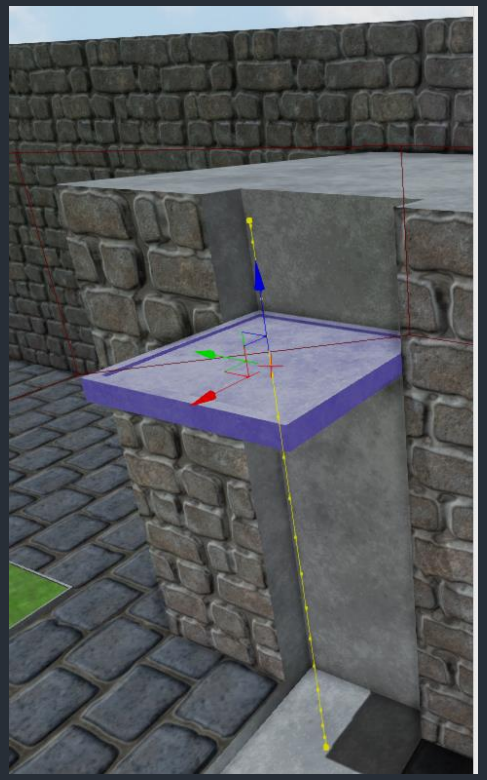
- Kismet wird meist für Interaktionen mit der Umgebung und dem Scripten von Handlung verwendet
- Einstellung zum Gameplay oder sehr komplexere Dinge wie neue Actor Klassen müssen weiterhin mit UScript verwirklicht werden

Resources

- Einiges zu Kismet:

http://www.avld.org/pages/tuts/tuts_Kismet.htm

Hausaufgabe



Baut mit einem extra Trigger und Kismet folgendes: Der Fahrstuhl soll automatisch hochfahren wenn man oben wartet und er aktuell unten ist.

Inhalt

1. Einleitung und Editor
2. Geometry und Lighting
3. StaticMeshes und Materials
4. Kismet
5. Particles

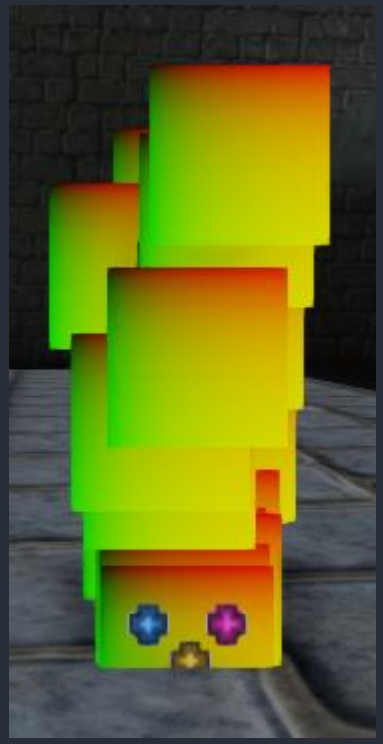
Bitte „myDeltaMap.udk“ im UDK laden
und Content in „myDeltaPack“ importieren

Particles

- Was sind Particles?
- Materialvorbereitung
- Erstellung eines ParticleSystems
- Cascade
- Curve Editor
- Wichtige Module
- Beispiel: Rauch
- Emitter in die Welt setzen
- SubUV
- Beispiel: Feuer
- Distortion
- Beispiel: Hitzeflimmern
- ParticleSystem um Emitter erweitern
- Particles und Kismet

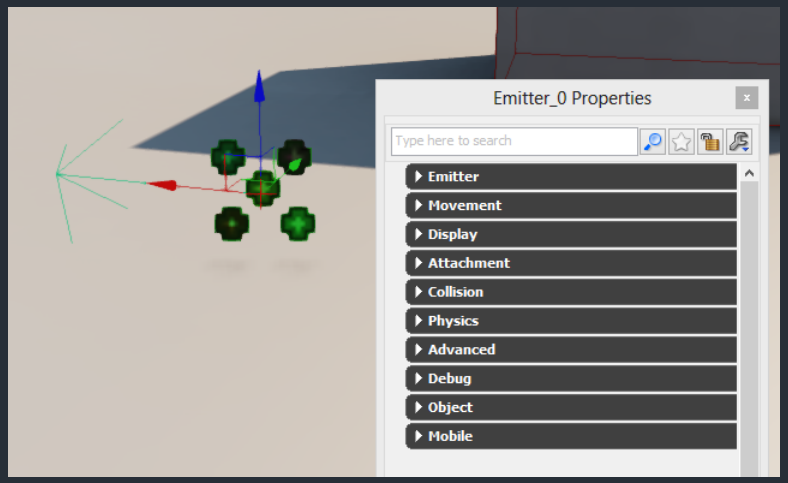
Was sind Particles?

- Es gibt verschiedene Partikelarten in der UE
- Wir behandeln nur die „Sprite"-Partikel
- Das sind 2 Tris (== Plane) welche sich immer Richtung Kamera ausrichten und die UV-Koordinaten von 0 bis 1 abdecken
- Emitter senden einzelne Partikel (bei uns Sprites) aus, deren Eigenschaften im ParticleSystem definiert wurden



Was sind Particles?

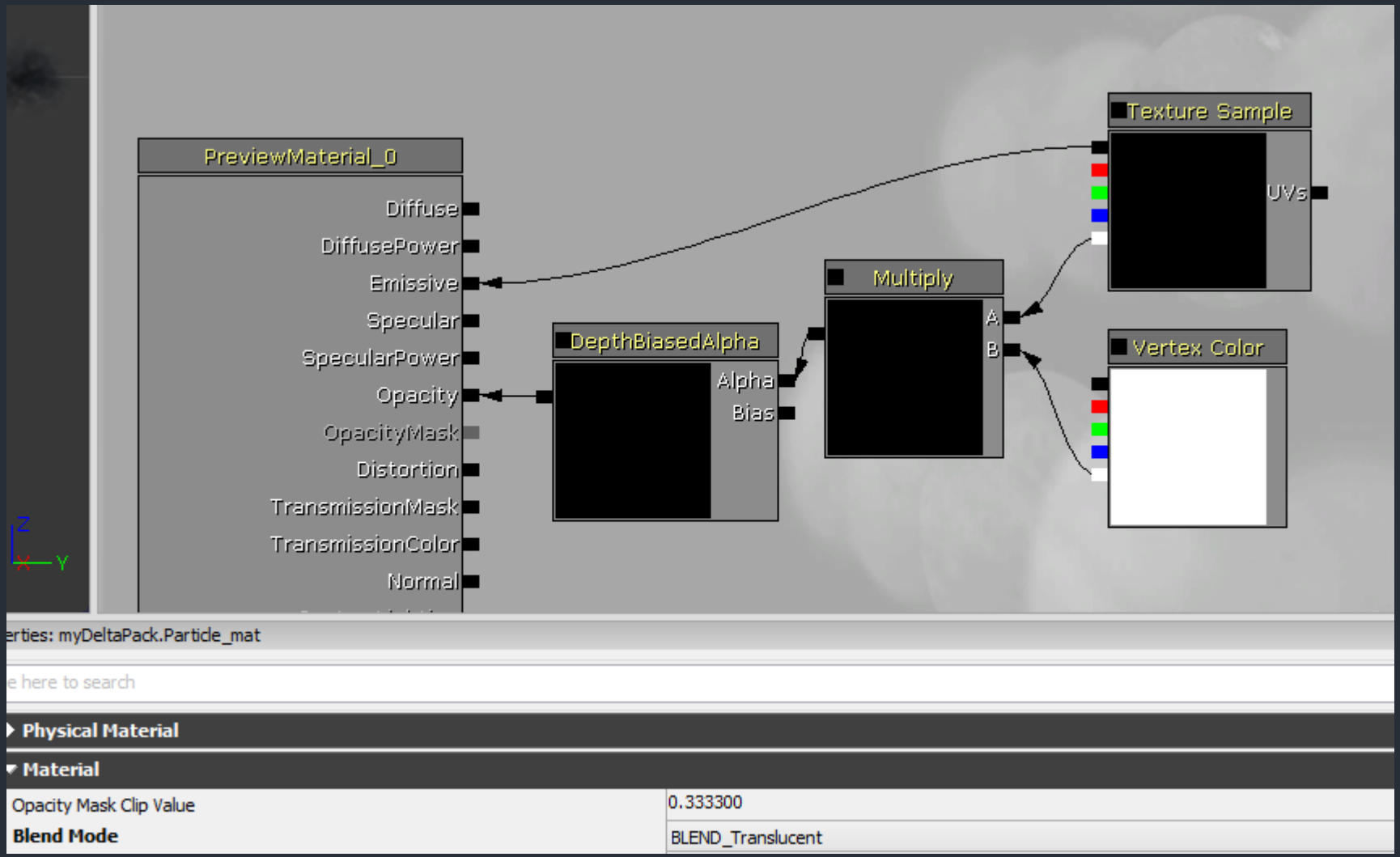
- Im Pack wird ein „ParticleSystem“ erstellt, welches wiederum einzelne „Emitter“ beinhalten kann (z.B. Flammen, Rauch, Hitzeflimmern..alles in einem System und später nur ein Actor)
- Im Level selbst wiederum werden ParticleSysteme von einem „Emitter“-Actor repräsentiert



Materialvorbereitung

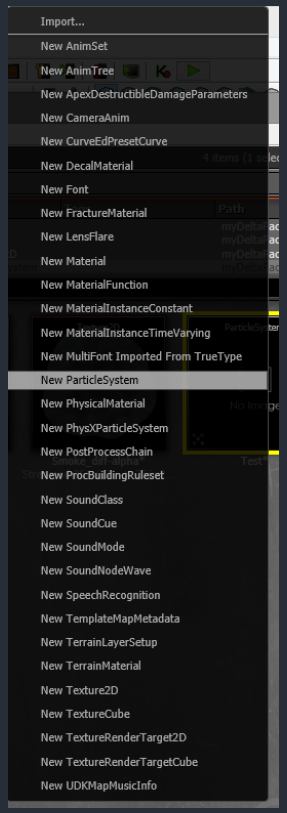
- VertexColor wird im ParticleSystem angesprochen und kann für die Transparenz verwendet werden (Ein- und Ausblenden)
- Vor allem bei Particeln wird oft „DepthBiasedAlpha“ benutzt um an Collisionsstellen Kantenbildung zu vermeiden (siehe Beispiel: Feuer)

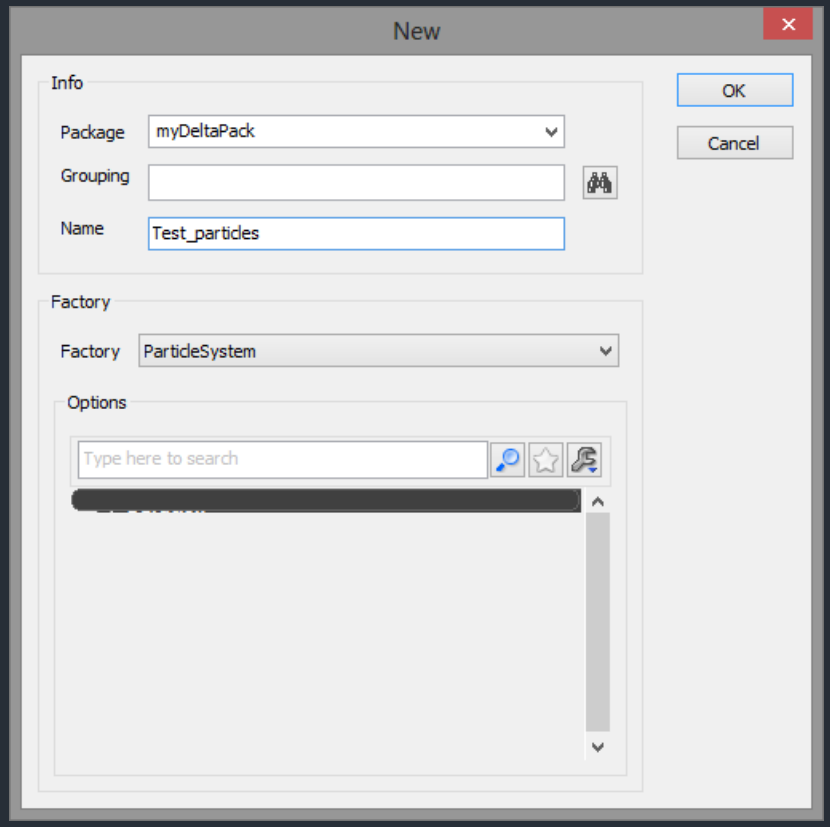
Materialvorbereitung



Erstellung eines ParticleSystems

- Pro ParticleSystem wird ein Objekt in einem Pack erstellt





Cascade

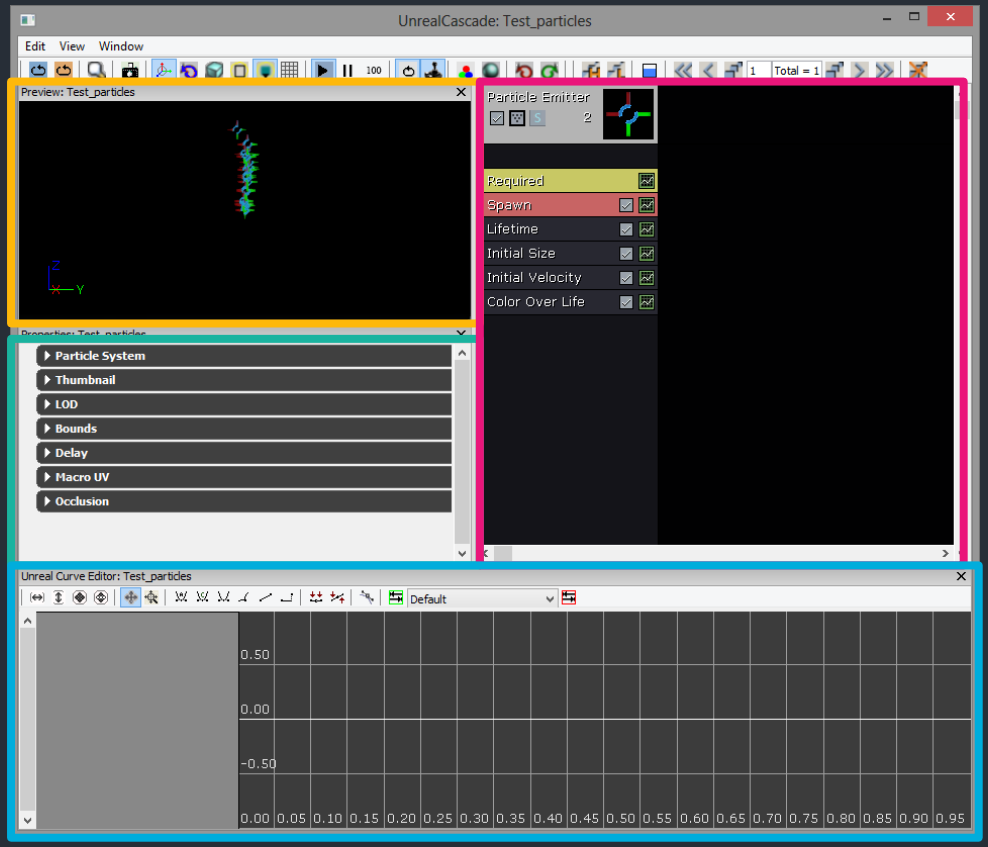
- Wie bei Materialien gibt es auch für ParticleSystems einen Editor: Cascade

3D-Vorschau

Properties des selektierten Blocks

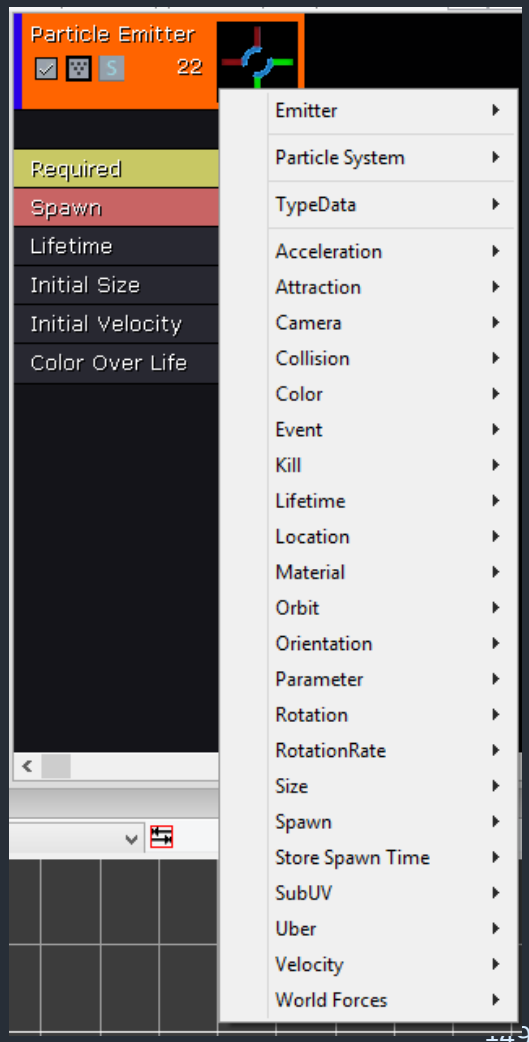
Emitter und deren Module

Curve Editor



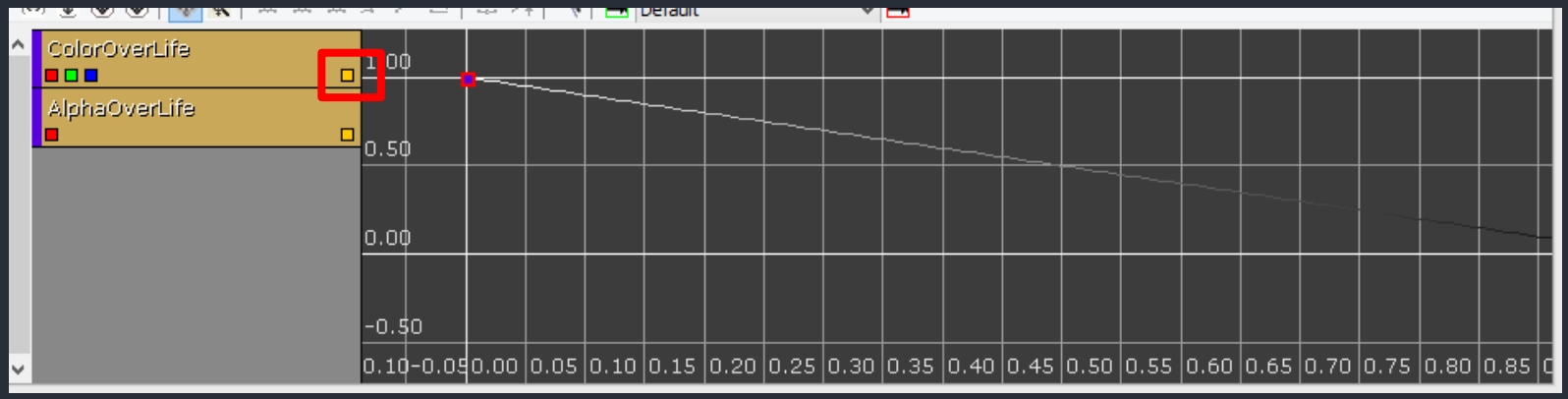
Cascade

- Rechtsklick auf den Emitter öffnet Auswahl an Modulen



Curve Editor

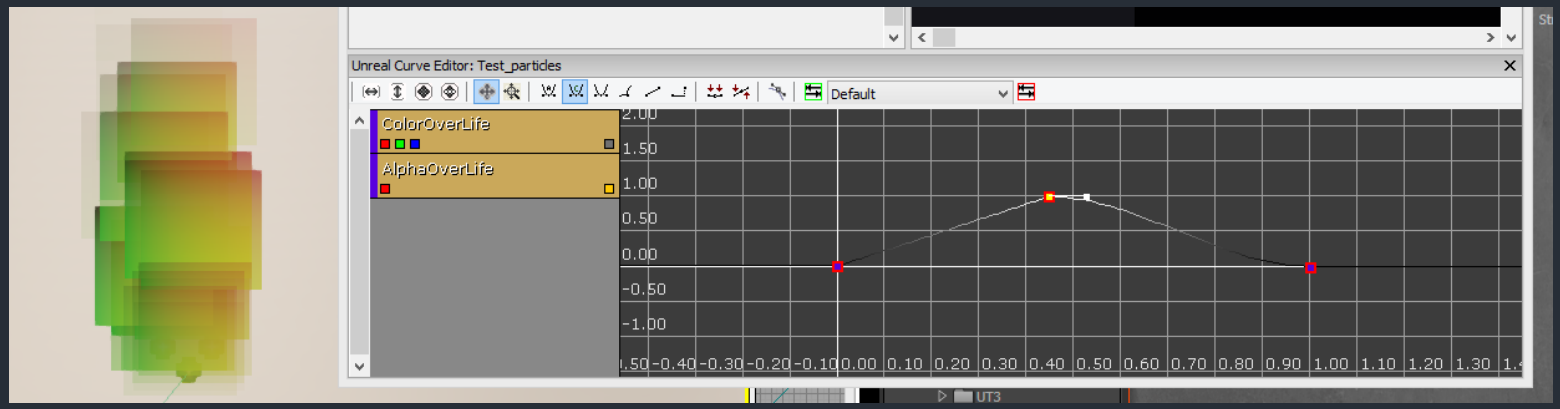
- Jedes Modul kann im Curve Editor angezeigt werden



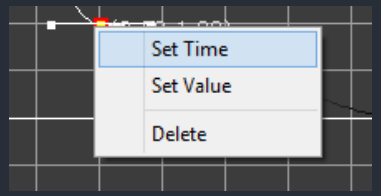
- Mit dem gelben Button kann man die Curve dieses Wertes ein-/ausblenden

Curve Editor

- Linke Maustaste + STRG bewegt vorhandene Punkte in der Curve oder erstellt neue Punkte in der Curve



- Rechtklick öffnet Pop-Up Menü
- X-Werte relativ, also X=1 immer am Lebensende des Particles



Wichtige Module

- Required
  - Material: Welches Material wird benutzt
- Initial Location
  - Start Location: Relative Position der Particle bei Aussendung
- Spawn
  - Rate: Wie viel Particle existieren maximal gleichzeitig
  - Rate Scale: Wie viele Particle werden pro Sekunde erzeugt
- Lifetime
  - Lifetime: Wie lange existiert ein Particle
- Initial Size
  - Start Size: Anfängliche Größe der Particles
- Initial Velocity
  - Start Velocity: Anfängliche Geschwindigkeit der Particles

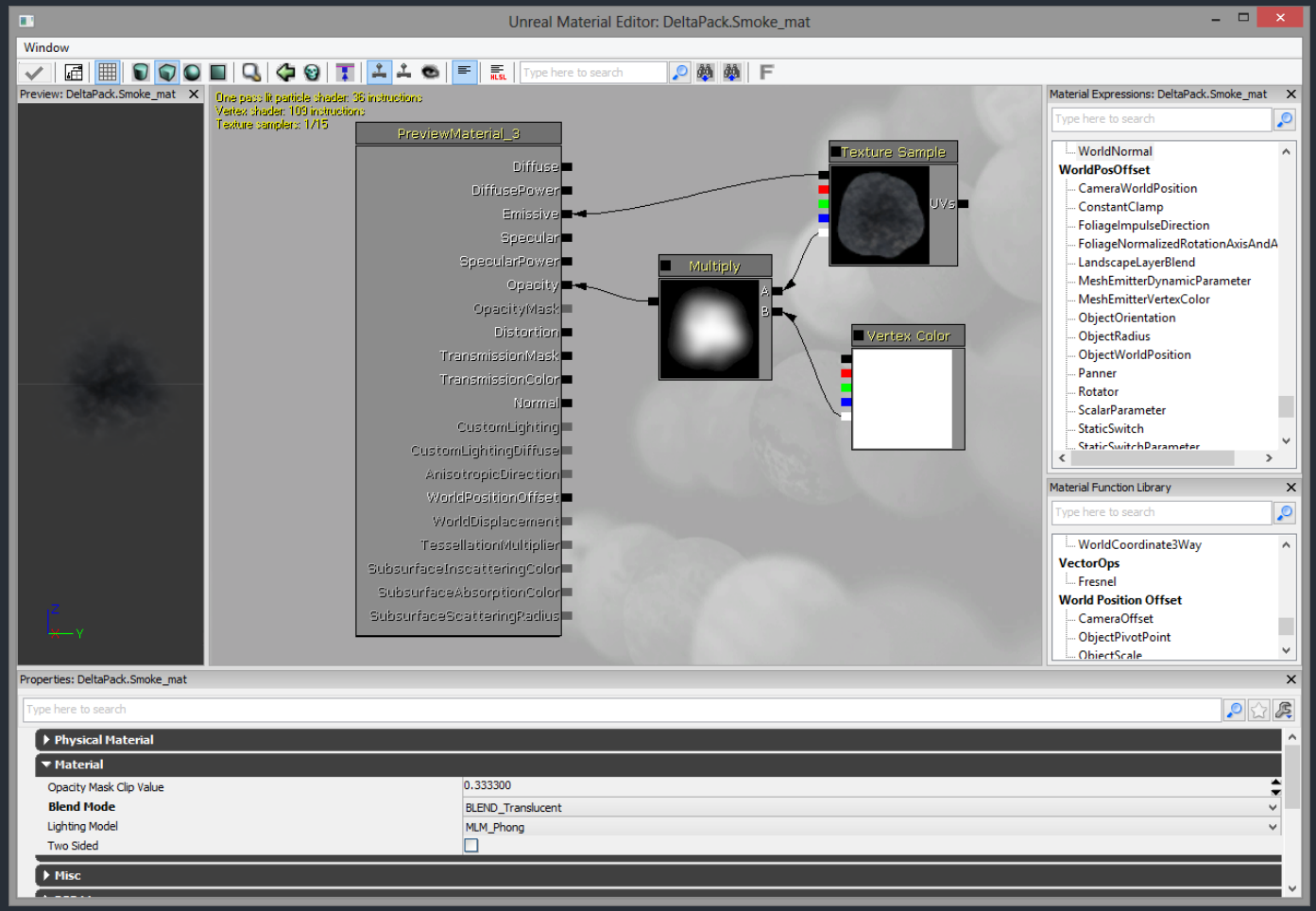
Wichtige Module

- Initial Velocity
  - Start Velocity: Anfängliche Geschwindigkeit der Particles
- Color Over Life
  - (Curve Editor): Steuerung der VertexColor im Material
- Size Over Life
  - (Curve Editor): Relative Größe der Particles in Abhängigkeit zur Zeit
- Intial Rotation
  - Start Rotation: Anfängliche Rotation der Particles
- Rotation Over Life
  - (Curve Editor): Relative Rotation der Particles in Abhängigkeit zur Zeit
- Acceleration
  - Acceleration: Anziehungkraft, welche auf Particles einwirkt

Beispiel: Rauch

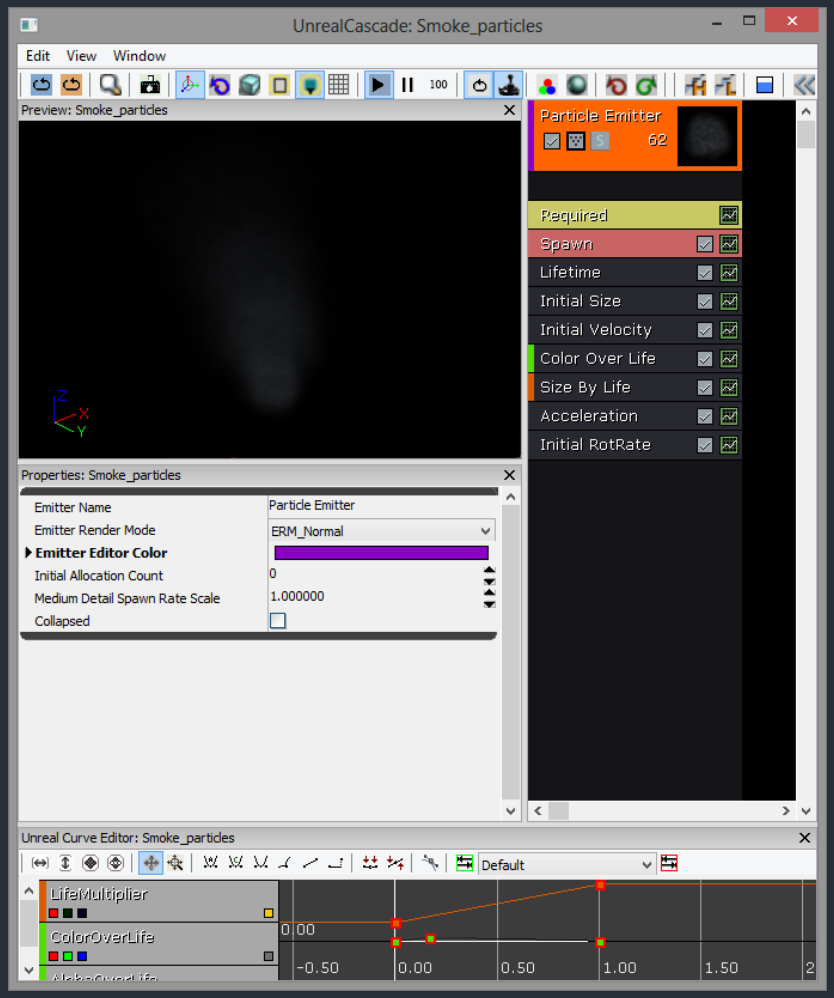
- Mit den eingeführten Elementen lässt sich schon Rauch erstellen
- Dazu „Smoke_diff-alpha“ in „myDeltaPack “ importieren
- Wir benötigen diese Textur, ein Material, ein ParticleSystem im Package und einen Emitter in der Welt (alles bitte in „myDeltaPack“ importieren/erstellen)

Beispiel: Rauch



Material (Erklärung im Tutorium)

Beispiel: Rauch



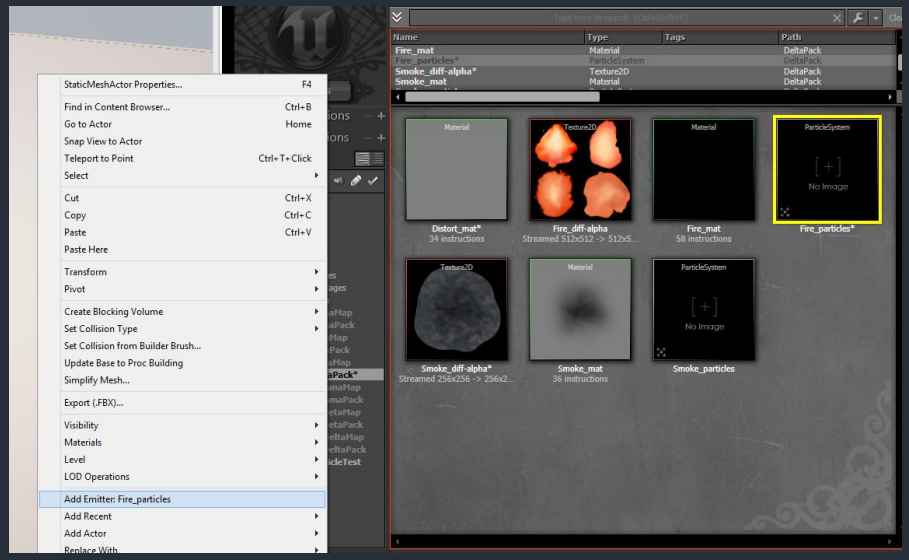
ParticleSystem (Erklärung im Tutorium)

Beispiel: Rauch

Emitter (Erklärung im Tutorium)

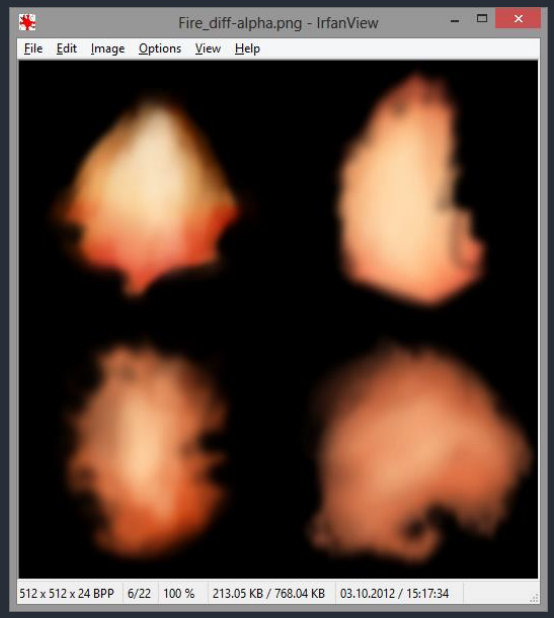
Emitter in die Welt setzen

- Funktioniert genau wie bei StaticMeshes
- ParticleSystem im Browser wählen und Rechtsklick in die Welt

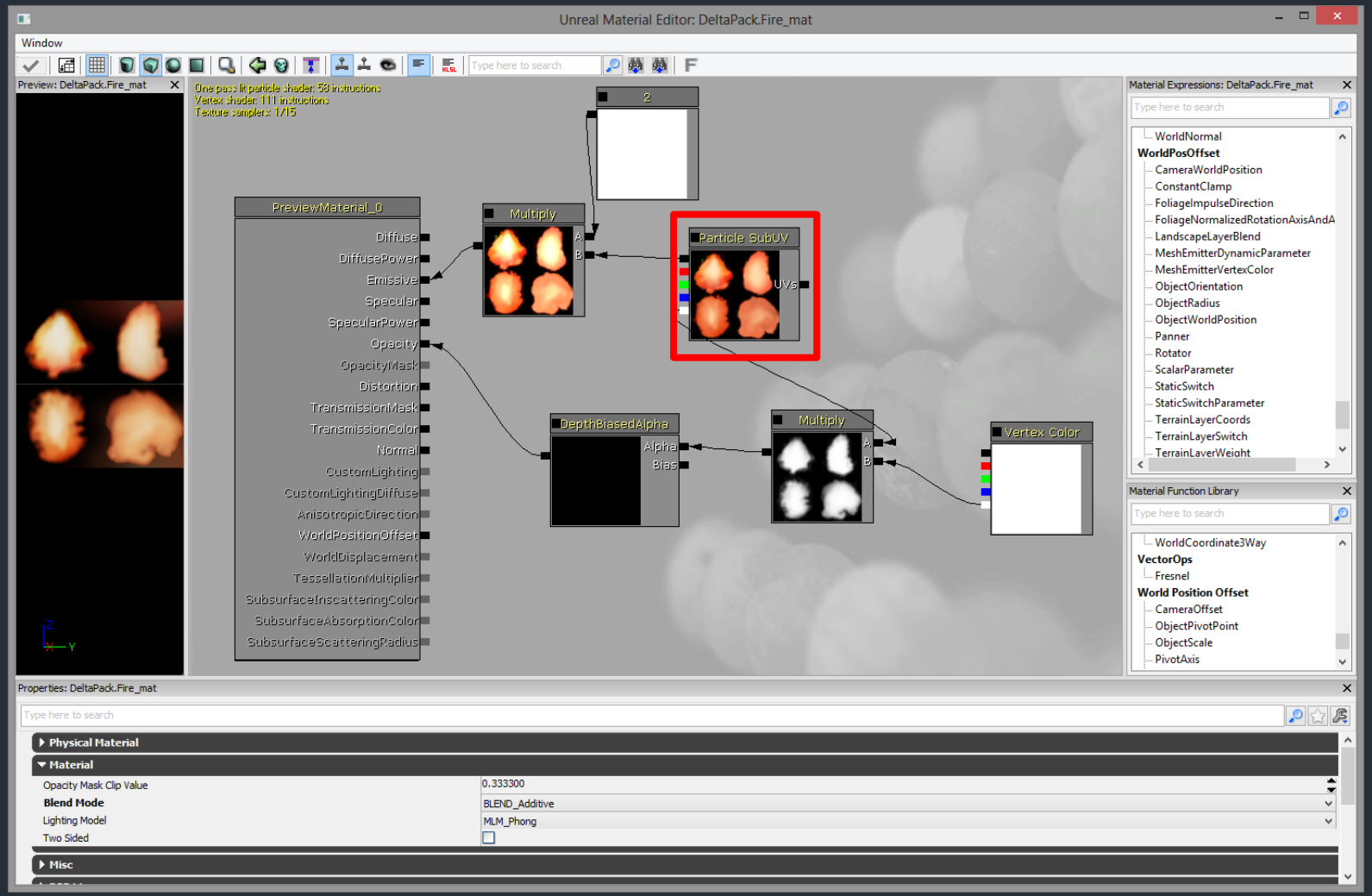


SubUV

- Es können auch mehrere Stadien eines Particles in einer Textur abgebildet werden
- Diese können z.B. zeitlich nacheinander auf dem Sprite angezeigt werden

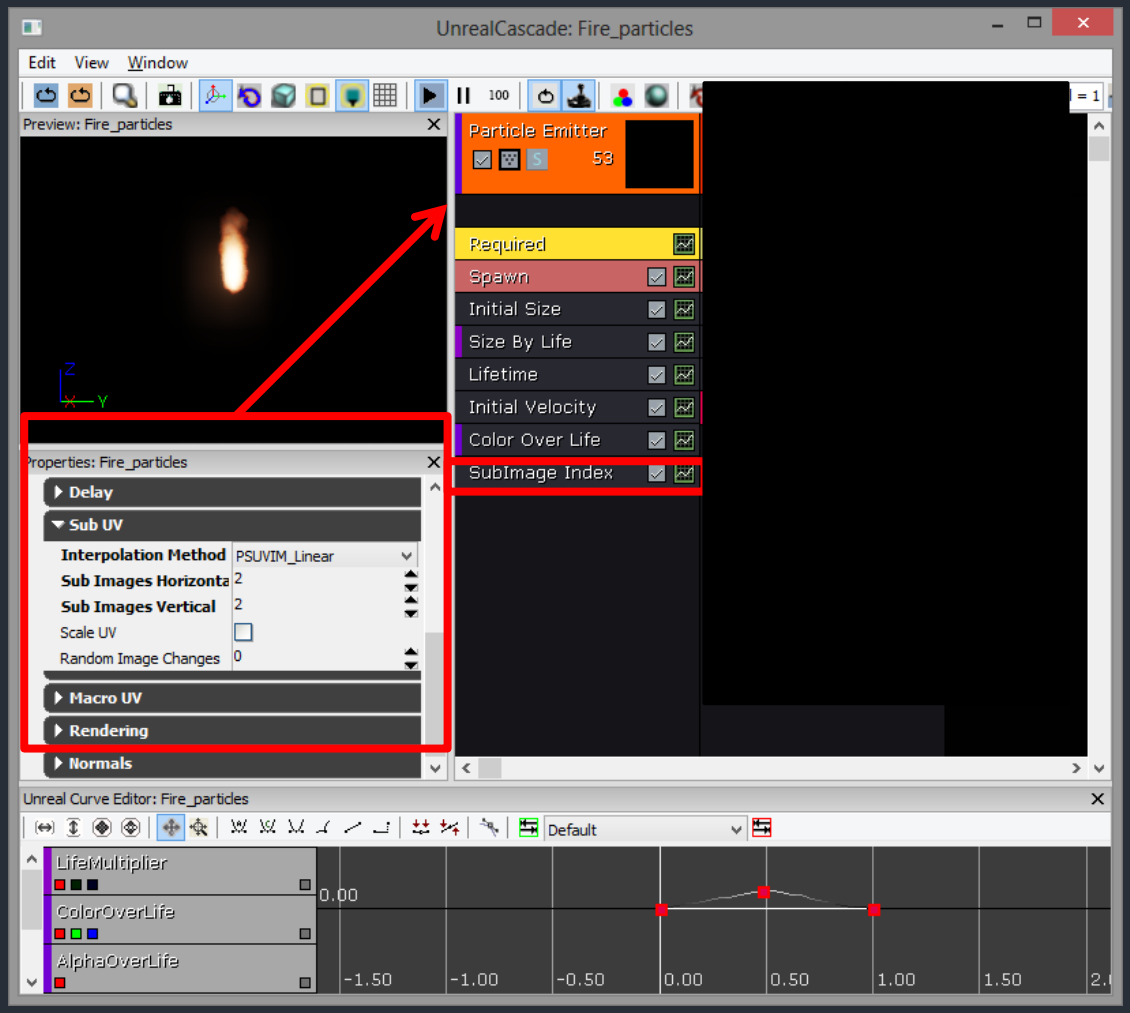


Beispiel: Feuer



Material mit Particle SubUV Node anstatt TextureSample (Erklärung im Tutorium)

Beispiel: Feuer

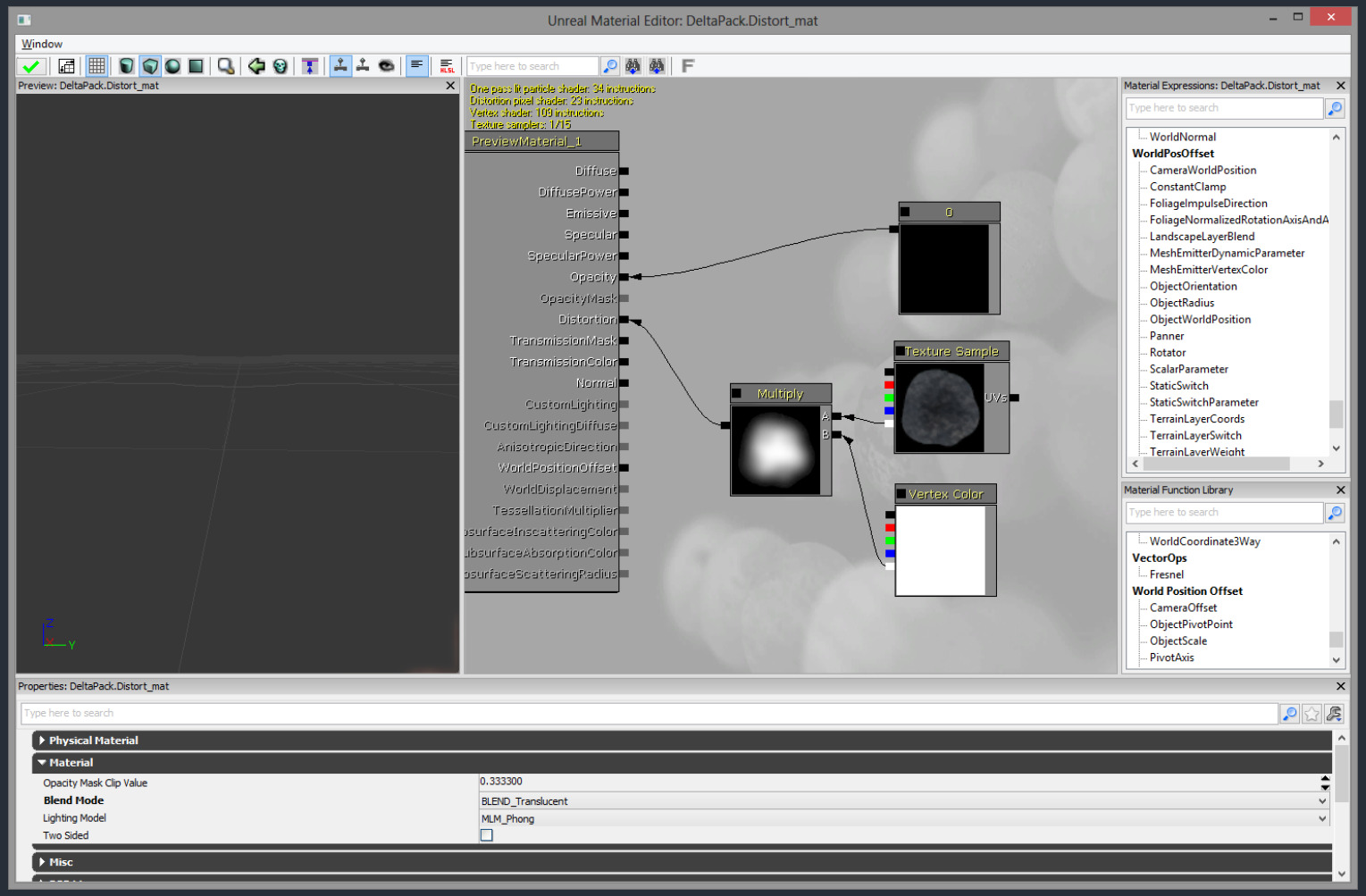


Feuer ParticleSystem mit SubUV (Erklärung im Tutorium)

Distortion

- Distortion ist an sich ein Materialeffekt und hat weniger direkt mit Particles zu tun, wird aber oft verwendet um z.B. Hitzeflimmern zu simulieren
- Distortion funktioniert z.B. nur bei additiven oder translucenten Materialien und verschiebt alles dahinter dargestellte um einen angegeben Wert (float2)

Beispiel: Hitzeflimmern



Distortion Material für Feuer (Erklärung im Tutorium)

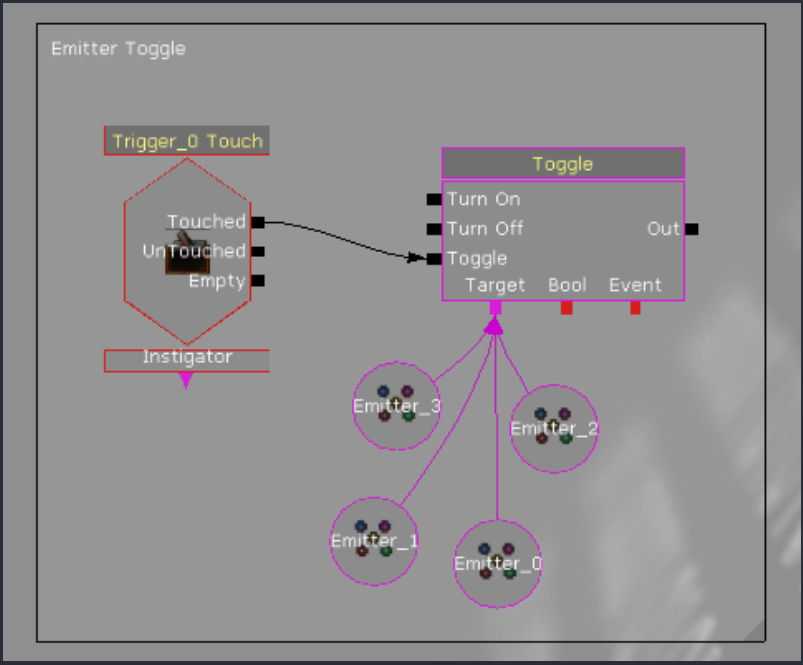
ParticleSystem um Emitter erweitern

- Rechtsklick in Cascade

- Alle Emitter des ParticleSystems werden in der Welt von einem Emitter Actor ausgesendet

Particles und Kismet

- Mit Toggle Node an und ausschalten

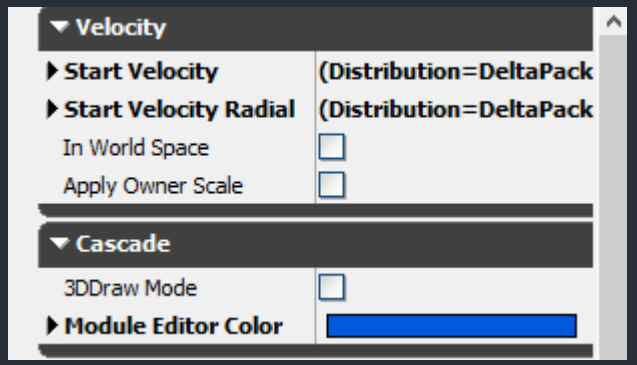


Particles und Kismet

- Per Matinee und einer Particle Spur lässt sich auch nicht mehr als togglen
- Für Veränderung der Farbe kann man Material Instances benutzen (nicht behandelt)
- Um das Scaling der Particles zu verändern, kann man vlt. was scripten? Keine Ahnung…

Bemerkungen

- Bei vielen Modulen in Cascade kann „In World Space“ ausgewählt werden. Dadurch sind die Werte wie z.B. die Geschwindigkeit nicht abhängig von der Rotation des aussendenden Emitter Actors



Resources

- UDN zu Partikeln:
  http://udn.epicgames.com/Three/CascadeUserGuide.html

Bonus (falls Zeit)

- Material Instances
- MeshPaint

Merkblatt

- Duplizieren (ALT+Widget oder STRG+W)
- Multiselection (STRG beim Selektieren gedrückt lassen)
- SPACE für Widget-Wechsel