



# Avid Maestro | Virtual Set

User Guide

Version 2018.6

## Legal Notices

Product specifications are subject to change without notice and do not represent a commitment on the part of Avid Technology, Inc.

This product is subject to the terms and conditions of a software license agreement provided with the software. The product may only be used in accordance with the license agreement.

This product may be protected by one or more U.S. and non-U.S. patents. Details are available at [www.avid.com/patents](http://www.avid.com/patents).

This document is protected under copyright law. An authorized licensee of may reproduce this publication for the licensee's own use in learning how to use the software. This document may not be reproduced or distributed, in whole or in part, for commercial purposes, such as selling copies of this document or providing support or educational services to others. This document is supplied as a guide for . Reasonable care has been taken in preparing the information it contains. However, this document may contain omissions, technical inaccuracies, or typographical errors. Avid Technology, Inc. does not accept responsibility of any kind for customers' losses due to the use of this document. Product specifications are subject to change without notice.

Copyright © 2018 Avid Technology, Inc. and its licensors. All rights reserved.

The following disclaimer is required by Epic Games, Inc.:

Unreal® is a trademark or registered trademark of Epic Games, Inc. in the United States of America and elsewhere.

Unreal® Engine, Copyright 1998 - 2018 Epic Games, Inc. All rights reserved."

The following disclaimer is required by Apple Computer, Inc.:

APPLE COMPUTER, INC. MAKES NO WARRANTIES WHATSOEVER, EITHER EXPRESS OR IMPLIED, REGARDING THIS PRODUCT, INCLUDING WARRANTIES WITH RESPECT TO ITS MERCHANTABILITY OR ITS FITNESS FOR ANY PARTICULAR PURPOSE. THE EXCLUSION OF IMPLIED WARRANTIES IS NOT PERMITTED BY SOME STATES. THE ABOVE EXCLUSION MAY NOT APPLY TO YOU. THIS WARRANTY PROVIDES YOU WITH SPECIFIC LEGAL RIGHTS. THERE MAY BE OTHER RIGHTS THAT YOU MAY HAVE WHICH VARY FROM STATE TO STATE.

The following disclaimer is required by Sam Leffler and Silicon Graphics, Inc. for the use of their TIFF library:

Copyright © 1988–1997 Sam Leffler

Copyright © 1991–1997 Silicon Graphics, Inc.

Permission to use, copy, modify, distribute, and sell this software [i.e., the TIFF library] and its documentation for any purpose is hereby granted without fee, provided that (i) the above copyright notices and this permission notice appear in all copies of the software and related documentation, and (ii) the names of Sam Leffler and Silicon Graphics may not be used in any advertising or publicity relating to the software without the specific, prior written permission of Sam Leffler and Silicon Graphics.

THE SOFTWARE IS PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

IN NO EVENT SHALL SAM LEFFLER OR SILICON GRAPHICS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER OR NOT ADVISED OF THE POSSIBILITY OF DAMAGE, AND ON ANY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

The following disclaimer is required by the Independent JPEG Group:

This software is based in part on the work of the Independent JPEG Group.

This Software may contain components licensed under the following conditions:

Copyright (c) 1989 The Regents of the University of California. All rights reserved.

Redistribution and use in source and binary forms are permitted provided that the above copyright notice and this paragraph are duplicated in all such forms and that any documentation, advertising materials, and other materials related to such distribution and use acknowledge that the software was developed by the University of California, Berkeley. The name of the University may not be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Copyright (C) 1989, 1991 by Jef Poskanzer.

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. This software is provided "as is" without express or implied warranty.

Copyright 1995, Trinity College Computing Center. Written by David Chappell.

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. This software is provided "as is" without express or implied warranty.

Copyright 1996 Daniel Dardailler.

Permission to use, copy, modify, distribute, and sell this software for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Daniel Dardailler not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. Daniel Dardailler makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

Modifications Copyright 1999 Matt Koss, under the same license as above.

Copyright (c) 1991 by AT&T.

Permission to use, copy, modify, and distribute this software for any purpose without fee is hereby granted, provided that this entire notice is included in all copies of any software which is or includes a copy or modification of this software and in all copies of the supporting documentation for such software.

THIS SOFTWARE IS BEING PROVIDED "AS IS", WITHOUT ANY EXPRESS OR IMPLIED WARRANTY. IN PARTICULAR, NEITHER THE AUTHOR NOR AT&T MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND CONCERNING THE MERCHANTABILITY OF THIS SOFTWARE OR ITS FITNESS FOR ANY PARTICULAR PURPOSE.

This product includes software developed by the University of California, Berkeley and its contributors.

The following disclaimer is required by Nexidia Inc.:

© 2010 Nexidia Inc. All rights reserved, worldwide. Nexidia and the Nexidia logo are trademarks of Nexidia Inc. All other trademarks are the property of their respective owners. All Nexidia materials regardless of form, including without limitation, software applications, documentation and any other information relating to Nexidia Inc., and its products and services are the exclusive property of Nexidia Inc. or its licensors. The Nexidia products and services described in these materials may be covered by Nexidia's United States patents: 7,231,351; 7,263,484; 7,313,521; 7,324,939; 7,406,415, 7,475,065; 7,487,086 and/or other patents pending and may be manufactured under license from the Georgia Tech Research Corporation USA.

The following disclaimer is required by Paradigm Matrix:

Portions of this software licensed from Paradigm Matrix.

The following disclaimer is required by Ray Sauers Associates, Inc.:

"Install-It" is licensed from Ray Sauers Associates, Inc. End-User is prohibited from taking any action to derive a source code equivalent of "Install-It," including by reverse assembly or reverse compilation, Ray Sauers Associates, Inc. shall in no event be liable for any damages resulting from reseller's failure to perform reseller's obligation; or any damages arising from use or operation of reseller's products or the software; or any other damages, including but not limited to, incidental, direct, indirect, special or consequential Damages including lost profits, or damages resulting from loss of use or inability to use reseller's products or the software for any reason including copyright or patent infringement, or lost data, even if Ray Sauers Associates has been advised, knew or should have known of the possibility of such damages.

The following disclaimer is required by Videomedia, Inc.:

"Videomedia, Inc. makes no warranties whatsoever, either express or implied, regarding this product, including warranties with respect to its merchantability or its fitness for any particular purpose."

"This software contains V-LAN ver. 3.0 Command Protocols which communicate with V-LAN ver. 3.0 products developed by Videomedia, Inc. and V-LAN ver. 3.0 compatible products developed by third parties under license from Videomedia, Inc. Use of this software will allow "frame accurate" editing control of applicable videotape recorder decks, videodisc recorders/players and the like."

The following disclaimer is required by Altura Software, Inc. for the use of its Mac2Win software and Sample Source Code:

©1993-1998 Altura Software, Inc.

The following disclaimer is required by Ultimatte Corporation:

Certain real-time compositing capabilities are provided under a license of such technology from Ultimatte Corporation and are subject to copyright protection.

The following disclaimer is required by 3Prong.com Inc.:

Certain waveform and vector monitoring capabilities are provided under a license from 3Prong.com Inc.

The following disclaimer is required by Interplay Entertainment Corp.:

The "Interplay" name is used with the permission of Interplay Entertainment Corp., which bears no responsibility for Avid products.

This product includes portions of the Alloy Look & Feel software from Incors GmbH.

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>).

© DevelopMentor

This product may include the JCifs library, for which the following notice applies:

JCifs © Copyright 2004, The JCIFS Project, is licensed under LGPL (<http://jcifs.samba.org/>). See the LGPL.txt file in the Third Party Software directory on the installation CD.

Avid Interplay contains components licensed from LavanTech. These components may only be used as part of and in connection with Avid Interplay.

This product includes the Warlib library, for which the following notice applies:

Copyright Jarle (jgaa) Aase 2000 - 2009

COPYRIGHT file which is included in the distribution:

warlib is copyright Jarle (jgaa) Aase 2000 - 2009

The warlib C++ Library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 3.0 of the License, or (at your option) any later version.

The warlib C++ Library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

Portions copyright © 2012 Avid Technology, Inc.

#### Attn. Government User(s). Restricted Rights Legend

U.S. GOVERNMENT RESTRICTED RIGHTS. This Software and its documentation are “commercial computer software” or “commercial computer software documentation.” In the event that such Software or documentation is acquired by or on behalf of a unit or agency of the U.S. Government, all rights with respect to this Software and documentation are subject to the terms of the License Agreement, pursuant to FAR §12.212(a) and/or DFARS §227.7202-1(a), as applicable.

#### Trademarks

003, 192 Digital I/O, 192 I/O, 96 I/O, 96i I/O, Adrenaline, AirSpeed, ALEX, Alienbrain, AME, AniMatte, Archive, Archive II, Assistant Station, AudioPages, AudioStation, AutoLoop, AutoSync, Avid, Avid Active, Avid Advanced Response, Avid DNA, Avid DNxcel, Avid DNxHD, Avid DS Assist Station, Avid Ignite, Avid Liquid, Avid Media Engine, Avid Media Processor, Avid MEDIArray, Avid Mojo, Avid Remote Response, Avid Unity, Avid Unity ISIS, Avid VideoRAID, AvidRAID, AvidShare, AVIDstripe, AVX, Beat Detective, Beauty Without The Bandwidth, Beyond Reality, BF Essentials, Bomb Factory, Bruno, C|24, CaptureManager, ChromaCurve, ChromaWheel, Cineractive Engine, Cineractive Player, Cineractive Viewer, Color Conductor, Command|24, Command|8, Control|24, Cosmonaut Voice, Countdown, d2, d3, DAE, D-Command, D-Control, Deko, DekoCast, D-Fi, D-fx, Digi 002, Digi 003, DigiBase, Digidesign, Digidesign Audio Engine, Digidesign Development Partners, Digidesign Intelligent Noise Reduction, Digidesign TDM Bus, DigiLink, DigiMeter, DigiPanner, DigiProNet, DigiRack, DigiSerial, DigiSnake, DigiSystem, Digital Choreography, Digital Nonlinear Accelerator, DigiTest, DigiTranslator, DigiWear, DINR, DNxchange, Do More, DPP-1, D-Show, DSP Manager, DS-StorageCalc, DV Toolkit, DVD Complete, D-Verb, Eleven, EM, Euphonix, EUCON, EveryPhase, Expander, ExpertRender, Fader Pack, Fairchild, FastBreak, Fast Track, Film Cutter, FilmScribe, Flexevent, FluidMotion, Frame Chase, FXDeko, HD Core, HD Process, HDpack, Home-to-Hollywood, HYBRID, HyperSPACE, HyperSPACE HDCAM, iKnowledge, Image Independence, Impact, Improv, iNEWS, iNEWS Assign, iNEWS ControlAir, InGame, Instantwrite, Instinct, Intelligent Content Management, Intelligent Digital Actor Technology, IntelliRender, Intelli-Sat, Intelli-sat Broadcasting Recording Manager, InterFX, Interplay, inTONE, Intraframe, iS Expander, iS9, iS18, iS23, iS36, ISIS, IsoSync, LaunchPad, LeaderPlus, LFX, Lightning, Link & Sync, ListSync, LKT-200, Lo-Fi, MachineControl, Magic Mask, Make Anything Hollywood, make manage move | media, Marquee, MassivePack, Massive Pack Pro, Maxim, Mbox, Media Composer, MediaFlow, MediaLog, MediaMix, Media Reader, Media Recorder, MEDIArray, MediaServer, MediaShare, MetaFuze, MetaSync, MIDI I/O, Mix Rack, Moviestar, MultiShell, NaturalMatch, NewsCutter, NewsView, NewsVision, Nitris, NL3D, NLP, NSDOS, NSWIN, OMF, OMF Interchange, OMM, OnDVD, Open Media Framework, Open Media Management, Painterly Effects, Palladium, Personal Q, PET, Podcast Factory, PowerSwap, PRE, ProControl, ProEncode, Profiler, Pro Tools, Pro Tools|HD, Pro Tools LE, Pro Tools M-Powered, Pro Transfer, QuickPunch, QuietDrive, Realtime Motion Synthesis, Recti-Fi, Reel Tape Delay, Reel Tape Flanger, Reel Tape Saturation, Reprise, Res Rocket Surfer, Reso, RetroLoop, Reverb One, ReVibe, Revolution, rS9, rS18, RTAS, Salesview, Sci-Fi, Scorch, ScriptSync, SecureProductionEnvironment, Serv|GT, Serv|LT, Shape-to-Shape, ShuttleCase, Sibelius, SimulPlay, SimulRecord, Slightly Rude Compressor, Smack!, Soft SampleCell, Soft-Clip Limiter, SoundReplacer, SPACE, SPACESHift, SpectraGraph, SpectraMatte, SteadyGlide, Streamfactory, Streamgenie, StreamRAID, SubCap, Sundance, Sundance Digital, SurroundScope, Symphony, SYNC HD, SYNC I/O, Synchronic, SynchroScope, Syntax, TDM FlexCable, TechFlix, Tel-Ray, Thunder, TimeLiner, Titansync, Titan, TL Aggro, TL AutoPan, TL Drum Rehab, TL Everyphase, TL Fauxlder, TL In Tune, TL MasterMeter, TL Metro, TL Space, TL Utilities, tools for storytellers, Transit, TransJammer, Trillium Lane Labs, TruTouch, UnityRAID, Vari-Fi, Video the Web Way, VideoRAID, VideoSPACE, VTEM, Work-N-Play, Xdeck, X-Form, Xmon and XPAND! are either registered trademarks or trademarks of Avid Technology, Inc. in the United States and/or other countries.

#### Footage

Arizona Images — KNTV Production — Courtesy of Granite Broadcasting, Inc.,  
Editor/Producer Bryan Foote.  
Canyonlands — Courtesy of the National Park Service/Department of the Interior.  
Ice Island — Courtesy of Kurtis Productions, Ltd.  
Tornados + Belle Isle footage — Courtesy of KWTW News 9.  
WCAU Fire Story — Courtesy of NBC-10, Philadelphia, PA.  
Women in Sports – Paragliding — Courtesy of Legendary Entertainment, Inc.

News material provided by WFTV Television Inc.

Avid Maestro | Virtual Set User Guide v2018.6 • Created 7/6/18 • This document is distributed by Avid in online (electronic) form only, and is not available for purchase in printed form.



# UNREAL ENGINE

# Contents




	Symbols and Conventions . . . . .	7
<b>Chapter 1</b>	<b>Introduction</b> . . . . .	8
	What is Maestro   Virtual Set? . . . . .	8
	Maestro   Virtual Set Workflow . . . . .	9
	System Requirements . . . . .	10
	User Tested Specifications . . . . .	10
	Licensing . . . . .	11
	Features . . . . .	11
<b>Chapter 2</b>	<b>Working with Unreal Engine 4</b> . . . . .	13
	Getting Started . . . . .	13
	Unreal Engine 4 Tutorials . . . . .	13
	Unreal Engine Game Framework . . . . .	13
	What is Blueprint and its Advantage over C++ . . . . .	14
	Learning Blueprint . . . . .	14
	Instructions . . . . .	14
	Blueprint Communication . . . . .	14
	Editor Tips and Tricks . . . . .	15
	Authoring Using Unreal Engine . . . . .	15
<b>Chapter 3</b>	<b>Authoring with Unreal Editor</b> . . . . .	16
	Maestro   Virtual Set Authoring Workflow . . . . .	16
	Avid Maestro   Virtual Set Solution for Unreal Editor . . . . .	17
	Supporting Blueprint Projects . . . . .	17
	Project resources hierarchy . . . . .	18
	Changing Scenes on Air . . . . .	18
	Preview for Scene Creator . . . . .	19
	Way of Handling Clips and Video Insertions . . . . .	19
	Plugins Support . . . . .	19
	Project Preparation . . . . .	19
	New Scene with Starter Content . . . . .	20
	Importing Content . . . . .	22
	Creating Map (level) . . . . .	22
	Preview . . . . .	23
	Final Cooking . . . . .	23
	Unreal Editor Extensions . . . . .	23
	Working with Exports . . . . .	24
	Example 1: Registering of text content for a TextRenderActor object . . . . .	25

	Example 2: Registering of an X axis position for a simple Cone object .....	25
	Helper Functions .....	26
	Working with Animations .....	27
	Skeleton Animations .....	31
	Working with Alpha Matte .....	33
	Augmented Reality Objects .....	33
	Garbage Objects .....	34
	Alpha Matte Composition (Helpers) .....	36
	AR Objects .....	36
	Garbage Objects .....	37
	Alpha Matte Composition (Manual) .....	38
	AR Objects .....	38
	Garbage Objects .....	39
	Troubleshooting .....	39
	Translucent Materials .....	39
	Generating Garbage Matte Objects .....	40
	Working with Depth of Field (DoF) .....	44
<b>Chapter 4</b>	<b>Building an Unreal Engine Scene</b> .....	49
	Building Scenes with the “Avid Base Template” .....	49
<b>Chapter 5</b>	<b>Troubleshooting</b> .....	57
	How to Improve the Performance of Maestro   Virtual Set .....	57
	Frequently Asked Questions .....	59
<b>Appendix A</b>	<b>Guidelines for exporting from 3Ds Max® to Unreal</b> .....	62
	Introduction .....	62
	Animations .....	62
	Materials .....	63
	Baked Texture Method .....	67

# Using This Guide

## Symbols and Conventions

Avid documentation uses the following symbols and conventions:

Symbol or Convention	Meaning or Action
	A note provides important related information, reminders, recommendations, and strong suggestions.
	A caution means that a specific action you take could cause harm to your computer or cause you to lose data.
	A warning describes an action that could cause you physical harm. Follow the guidelines in this document or on the unit itself when handling electrical equipment.
>	This symbol indicates menu commands (and subcommands) in the order you select them. For example, File > Import means to open the File menu and then select the Import command.
▶	This symbol indicates a single-step procedure. Multiple arrows in a list indicate that you perform one of the actions listed.
(Windows), (Windows only), (Macintosh), or (Macintosh only)	This text indicates that the information applies only to the specified operating system, either Windows or Macintosh OS X.
<b>Bold font</b>	Bold font is primarily used in task instructions to identify user interface items and keyboard sequences.
<i>Italic font</i>	Italic font is used to emphasize certain words and to indicate variables.
<b>Courier Bold font</b>	Courier Bold font identifies text that you type.
Ctrl+key or mouse action	Press and hold the first key while you press the last key or perform the mouse action. For example, Command+Option+C or Ctrl+drag.
(pipe character)	The pipe character is used in some Avid product names, such as Interplay   Production. In this document, the pipe is used in product names when they are in headings or at their first use in text.

# 1 Introduction

In this section:

- [Maestro | Virtual Set Workflow](#)
- [System Requirements](#)
- [User Tested Specifications](#)
- [Licensing](#)

## What is Maestro | Virtual Set?

A full Avid Maestro | Virtual Set system is made up of 3 software modules:

- **Maestro | Virtual Set Authoring** - set of plugins and templates for the Unreal® Editor tool.
- **Control Application** - control Maestro | Virtual Set Engine and broadcast the graphics created with Maestro | Virtual Set Authoring, with applications such as Maestro | News or Maestro | Live.
- **Maestro | Virtual Set Engine** - real-time application for rendering the graphics to a video output using powerful Unreal Engine on dedicated Render Device.

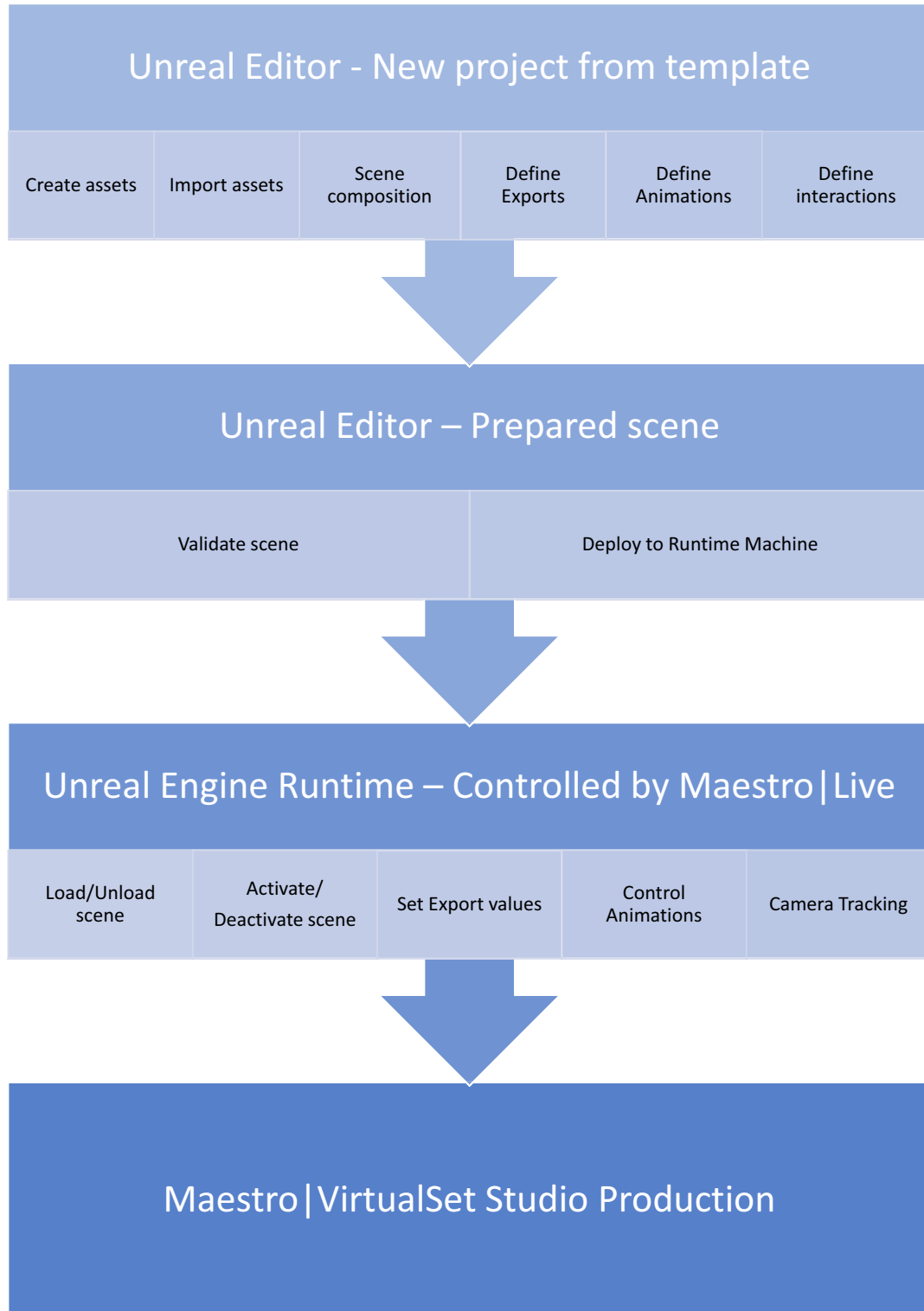
The Maestro | Virtual Set Authoring modules together with Unreal Editor address all requirements for integrating your OAG applications into an automated studio. First, 3D graphic scenes are created in the Unreal Editor, where you can define animated sequences and using additional plugins determine what can be changed in real time by “exporting” functions to the Control module. Reporters can then insert or import current data into the templates using a Controller, which can be run from a remote station.

Maestro | Virtual Set Authoring takes the advantage of Unreal Editor and addresses it's power into Virtual Studio creation tool.



# Maestro | Virtual Set Workflow

The following workflow contains the main tasks performed when creating a scene in Maestro | Virtual Set.



# System Requirements

The optimum platform for Maestro | Virtual Set uses an HDVG with authoring on Microsoft Windows® or Mac OS. To get started with creating or developing with Unreal Engine 4 right away, a considerably powerful setup is recommended.

Recommended hardware specifications for Maestro | Virtual Set:

- Windows 7/10 64-bit or Mac OS X 10.9.2 or later
- Intel® i5 or Intel® i7 Quad Core™ processor or similar (2.5 GHz or faster for high-performance graphics)
- NVIDIA® GeForce® GTX 1050Ti or AMD Radeon 6870 HD series card or higher
- HDD: 1TB (at least)
- RAM: minimum - 8 GB; recommended -16GB 3000MHz
- 24" 1920x1080 resolution capable monitor
- DVD-ROM/CD-Writer or DVD-Writer
- USB Keyboard and Mouse

## User Tested Specifications

This section lists PC specifications tested by Maestro | Virtual Set users. The user observations are not to be seen as official hardware recommendations from Avid.

### ASUS G75VW LAPTOP

Example configuration #1:

- OS: Windows 7 64bit
- Processor: Intel i7 2.3ghz
- System Memory: 12 GB
- Graphics: GTX670m
- Average fps: 20~30
- Settings: Default settings



*The ASUS G75VW ROG laptop is used for the engine. The fps is usually fine (unless there are particle-intense scenes).*

Example configuration #2:

- OS: Windows 8 (“8.0”, not 8.1)
- Processor: Intel i7 @2.5GHz
- System Memory: 16GB
- Graphics: 2x Nvidia GT755M
- Average fps: ~30fps with the scaling options all set to “Epic”
- Settings: Everything set to default

Example configuration #3:

- OS: Windows 7
- Processor: intel i7-3610QM
- System Memory: 4 GB
- Graphics: Nvidia Geforce GTX 660 M
- Average fps: around 30
- Settings: no special settings (high performance energy setting)

### **Building a Computer for Development**

If you are considering building a powerful machine for UE4 development for around \$2500, here is a list of recommended components:

- Intel Core i7 4930K processor
- Intel X79 chipset motherboard
- 32GB RAM
- 1TB SSD
- Nvidia GTX 770 video card

The 4930K is a great cost / CPU power trade-off for compiling the engine and running the editor.

SSDs were found to be the best user experience, and with the advent of the Samsung Evo series even large amounts of SSD storage have become attainable.

Copious amounts of RAM is the single biggest contributor to a smooth user experience. While the editor and engine don't use 32 GB, having the memory available for the disk cache is very beneficial, especially for a programmer compiling the full engine. Going above 32 GB can result in diminishing returns.

## **Licensing**

Install Maestro | Virtual Set Authoring using the Installation Wizard and the provided installation guide to help you through the various installation requirements.

After the installation, the HASP ID number or sysinfo of the Authoring Machine should be noted and sent to Avid support for licensing and registration.

## **Features**

The following features are available for the Maestro | Virtual Set rendering components:

<b>System Functionality</b>	<b>Unreal Engine</b>	<b>Render Engine</b>
Scene Editing	Unreal Editor	Maestro   Designer
Tracking Data	X	X
Animations	X	X
Exports	X	X

<b>System Functionality</b>	<b>Unreal Engine</b>	<b>Render Engine</b>
Imports from Designer tools	X	X
Garbage Matte	X	X
Alpha Matte for AR objects	i <sup>1</sup>	X
Newsroom System Integration	X	X
Scene Preview	X	X
Depth of Field	X	X
Clips		X
Video Insertions		X
Built-in Transitions		X
Built-in Charts		X
Audio		X
Ticker		X

1. Opacity elements are mixed with the Background and not with the Camera feed.

## 2 Working with Unreal Engine 4

In this section:

- [Getting Started](#)
- [Unreal Engine 4 Tutorials](#)
- [Unreal Engine Game Framework](#)
- [What is Blueprint and its Advantage over C++](#)
- [Learning Blueprint](#)
- [Authoring Using Unreal Engine](#)

### Getting Started

Unreal Engine has an extensive documentation, available online.

- To learn how to install the Engine, see [Installing Unreal Engine](#).
- To find out about the Terminology, see [Unreal Engine 4 Terminology](#).
- For a quick guide about the Editor, see [Tools and Editors](#).
- When you first run Unreal Editor, the Project Browser appears. To learn more about it, see [Project Browser](#).

### Unreal Engine 4 Tutorials

On the Unreal Engine website, you can also find comprehensive tutorials describing the authoring process in detail:

- Level Designer Quick Start: <https://docs.unrealengine.com/en-US/Engine/QuickStart>
- Realistic Rendering Example: <https://docs.unrealengine.com/en-US/Resources/Showcases/RealisticRendering>
- Lighting and Shadows Quick Start: <https://docs.unrealengine.com/en-US/Engine/Rendering/LightingAndShadows/QuickStart>
- Landscape Quick Start Guide: <https://docs.unrealengine.com/en-US/Engine/Landscape/QuickStart>
- Video Tutorials: [https://www.youtube.com/user/UnrealDevelopmentKit/playlists?sort=dd&view=50&shelf\\_id=17](https://www.youtube.com/user/UnrealDevelopmentKit/playlists?sort=dd&view=50&shelf_id=17)

### Unreal Engine Game Framework

If you are new to the editor, it will be useful to acquaint yourself with a few basic concepts regarding the user interface and general workflow. More information can be found in the Unreal Editor Interface guide (<https://docs.unrealengine.com/en-us/Engine/UI>).

# What is Blueprint and its Advantage over C++

The **Blueprints Visual Scripting** system in Unreal Engine is a complete gameplay scripting system based on the concept of using a node-based interface to create gameplay elements from within Unreal Editor. As with many common scripting languages, it is used to define object-oriented (OO) classes or objects in the engine. Objects defined using Blueprint are referred to as just “Blueprints.”

This system is extremely flexible and powerful as it provides the ability for designers to use virtually the full range of concepts and tools generally only available to programmers.

For more information on Blueprint, see **Blueprints Visual Scripting Documentation** (<https://docs.unrealengine.com/en-us/Engine/Blueprints>).

## Learning Blueprint

### Instructions

You can learn Blueprint from the following Unreal Engine tutorials:

<b>Tutorial</b>	<b>URL</b>
- Glossary	<a href="https://docs.unrealengine.com/en-us/Engine/Blueprints/Glossary">https://docs.unrealengine.com/en-us/Engine/Blueprints/Glossary</a>
- Getting started	<a href="https://docs.unrealengine.com/en-us/Engine/Blueprints/Scripting">https://docs.unrealengine.com/en-us/Engine/Blueprints/Scripting</a>
- Level Blueprint Editor	<a href="https://docs.unrealengine.com/en-US/engine/blueprints/editor/uibreakdowns/levelbpui">https://docs.unrealengine.com/en-US/engine/blueprints/editor/uibreakdowns/levelbpui</a>
- Blueprint Class UI	<a href="https://docs.unrealengine.com/en-us/Engine/Blueprints/Editor/UIBreakdowns/ClassBPUI">https://docs.unrealengine.com/en-us/Engine/Blueprints/Editor/UIBreakdowns/ClassBPUI</a>
- Anatomy of Blueprint	<a href="https://docs.unrealengine.com/en-us/Engine/Blueprints/Anatomy">https://docs.unrealengine.com/en-us/Engine/Blueprints/Anatomy</a>
- Workflow tools	<a href="https://docs.unrealengine.com/en-us/Engine/Blueprints/Workflow">https://docs.unrealengine.com/en-us/Engine/Blueprints/Workflow</a>
- Specialized Node Groups	<a href="https://docs.unrealengine.com/en-us/Engine/Blueprints/UserGuide">https://docs.unrealengine.com/en-us/Engine/Blueprints/UserGuide</a>

### Blueprint Communication

To learn how the Blueprints interact with one another, visit the following pages:

<b>Tutorial</b>	<b>URL</b>
- Blueprint Communication Usage	<a href="https://docs.unrealengine.com/en-us/Engine/Blueprints/UserGuide/BlueprintCommsUsage">https://docs.unrealengine.com/en-us/Engine/Blueprints/UserGuide/BlueprintCommsUsage</a>
- Direct Communication	<a href="https://docs.unrealengine.com/en-us/Engine/Blueprints/UserGuide/BlueprintComms">https://docs.unrealengine.com/en-us/Engine/Blueprints/UserGuide/BlueprintComms</a>
- Event Dispatchers	<a href="https://docs.unrealengine.com/en-US/engine/blueprints/editor/uibreakdowns/levelbpui">https://docs.unrealengine.com/en-US/engine/blueprints/editor/uibreakdowns/levelbpui</a>
- Blueprint Interfaces	<a href="https://docs.unrealengine.com/en-US/Engine/Blueprints/UserGuide/Types/Interface/UsingInterfaces">https://docs.unrealengine.com/en-US/Engine/Blueprints/UserGuide/Types/Interface/UsingInterfaces</a>

Tutorial	URL
- Casting	<a href="https://docs.unrealengine.com/en-US/Engine/Blueprints/UserGuide/CastNodes">https://docs.unrealengine.com/en-US/Engine/Blueprints/UserGuide/CastNodes</a>

## Editor Tips and Tricks

The Blueprint Editor has a lot of productivity boosting shortcuts built-in, and while many will come naturally as you use the editor, others are a little bit hidden. The table below lists some useful Unreal Engine tips and tricks:

Tutorial	URL
- Blueprint Editor Cheat Sheet	<a href="https://docs.unrealengine.com/en-US/Engine/Blueprints/UserGuide/CheatSheet">https://docs.unrealengine.com/en-US/Engine/Blueprints/UserGuide/CheatSheet</a>
- Best Practices	<a href="https://docs.unrealengine.com/en-us/Engine/Blueprints/BestPractices">https://docs.unrealengine.com/en-us/Engine/Blueprints/BestPractices</a>
- Performance and Profiling	<a href="https://docs.unrealengine.com/en-US/Engine/Performance">https://docs.unrealengine.com/en-US/Engine/Performance</a>
- Blueprint Interfaces	<a href="https://docs.unrealengine.com/en-US/Engine/Blueprints/UserGuide/Types/Interface/UsingInterfaces">https://docs.unrealengine.com/en-US/Engine/Blueprints/UserGuide/Types/Interface/UsingInterfaces</a>
- Casting	<a href="https://docs.unrealengine.com/en-US/Engine/Blueprints/UserGuide/CastNodes">https://docs.unrealengine.com/en-US/Engine/Blueprints/UserGuide/CastNodes</a>

## Authoring Using Unreal Engine

Artist Quick Guide (<https://docs.unrealengine.com/en-US/Engine/Content/QuickStart>) provides comprehensive information on how to create and modify export scenes to Unreal Engine.



*Make sure to select Blueprint instead of C++ when starting a project.*

To learn about Asset Creation, visit <https://docs.unrealengine.com/en-US/Engine/Content/AssetCreation>

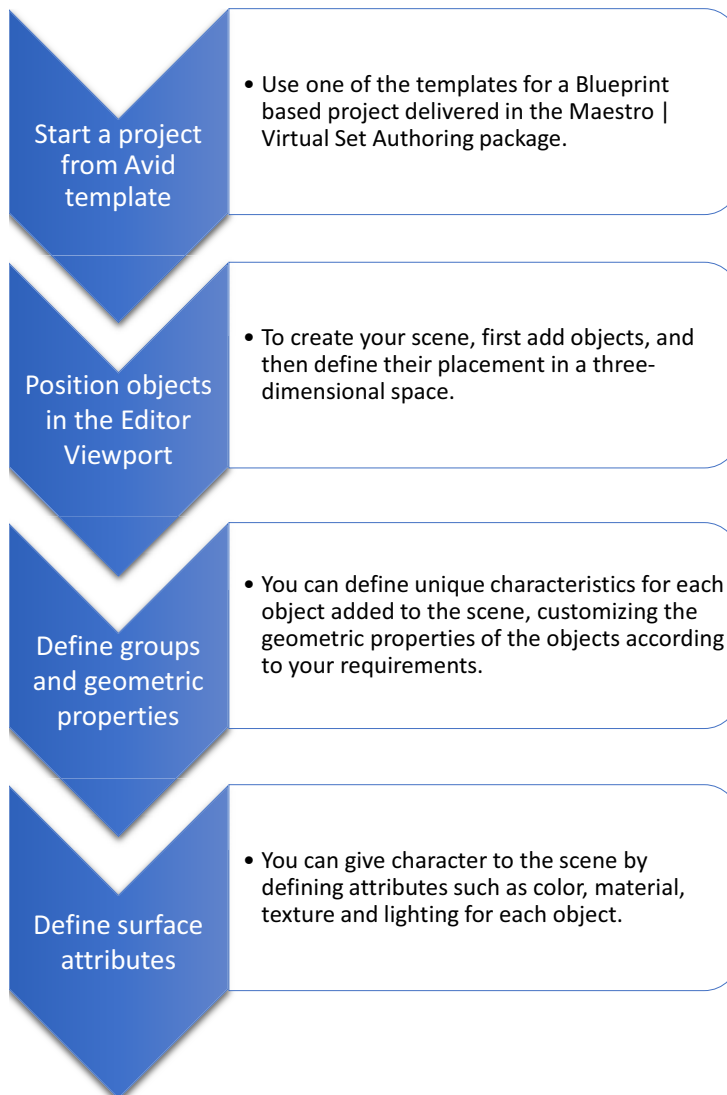
Unreal Engine features an FBX import pipeline which allows simple transfer of content from any number of digital content creation applications that support the format. To learn more about it, visit the FBX Content Pipeline (<https://docs.unrealengine.com/en-US/Engine/Content/FBX>) manual.

## 3 Authoring with Unreal Editor

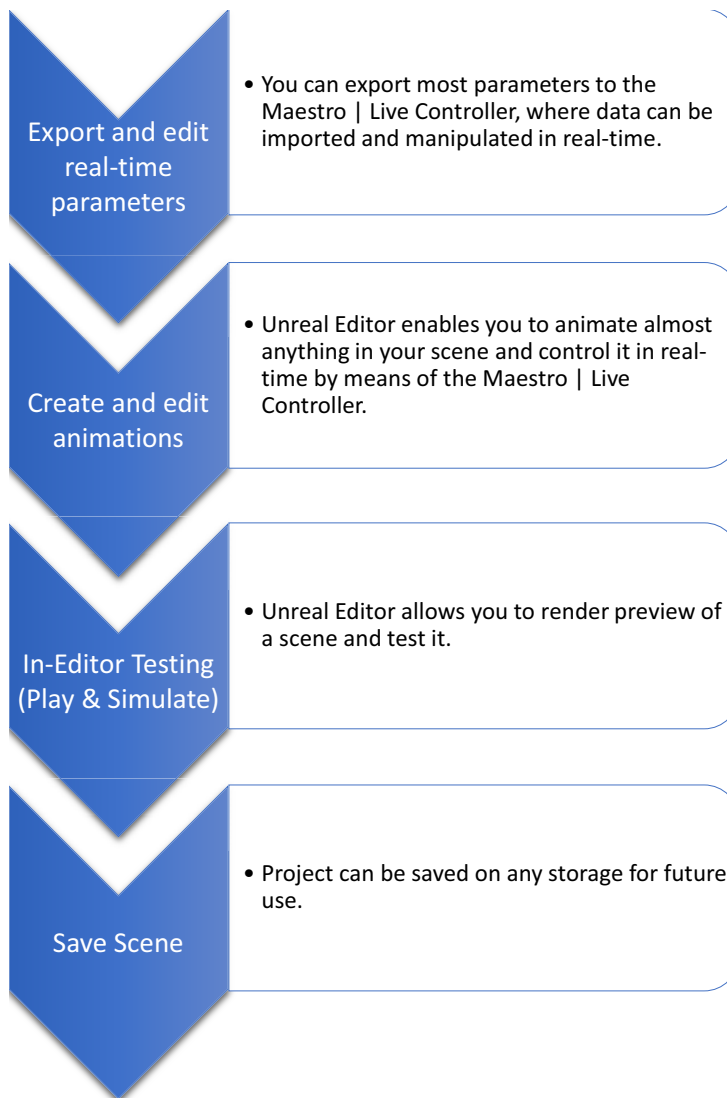
In this section:

- [Maestro | Virtual Set Authoring Workflow](#)
- [Avid Maestro | Virtual Set Solution for Unreal Editor](#)
- [Working with Exports](#)
- [Working with Animations](#)
- [Working with Alpha Matte](#)

### Maestro | Virtual Set Authoring Workflow





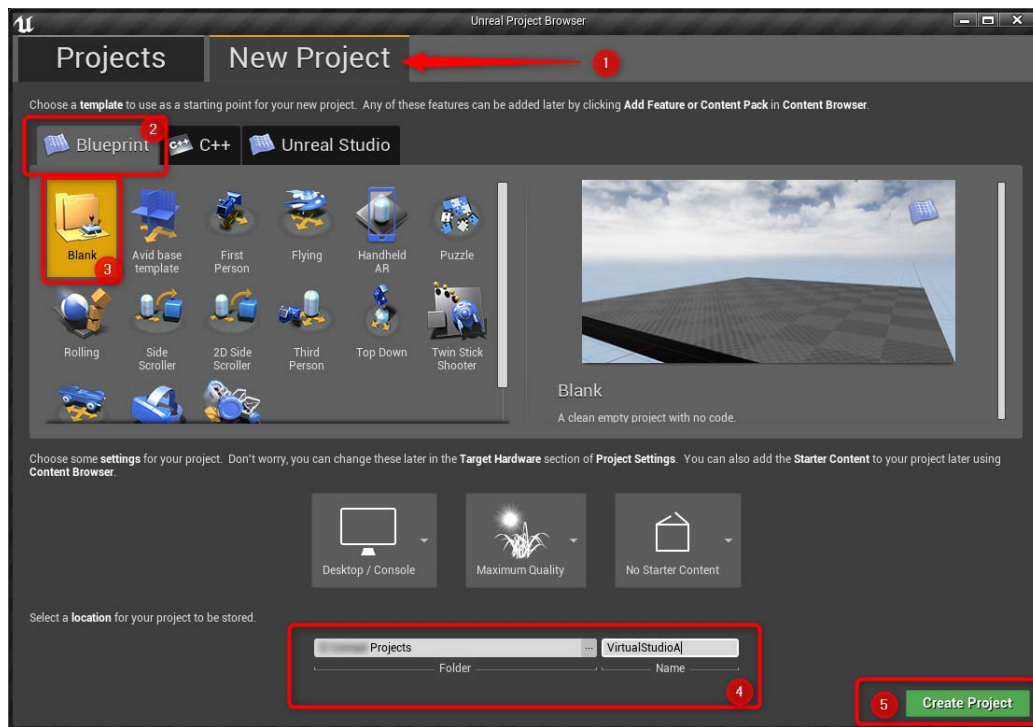


## Avid Maestro | Virtual Set Solution for Unreal Editor

There are a few basic concepts which distinguish the Maestro | Virtual Set workflow from a regular game development by means of Unreal Editor.

### Supporting Blueprint Projects

Our solution is only supporting Blueprint type projects. When starting a new project for the authoring purpose, you should follow the Blueprint approach and avoid C++.

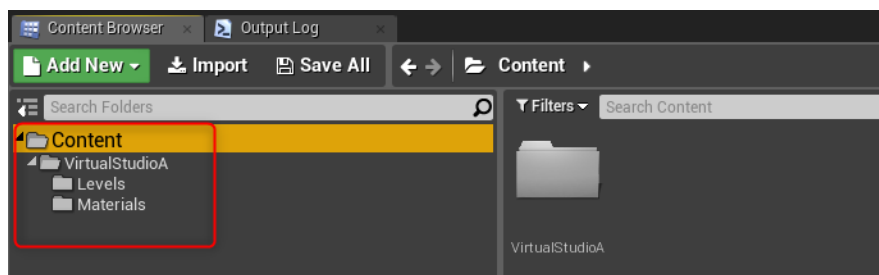


## Project resources hierarchy

All resources related to the project need to be stored in a single folder inside Content. The structure below that folder is not important and can be organized freely. It also applies to all asset resources binded to the project, e.g. Marketplace.

### Example 1:

Project **VirtualStudioA** was created in Unreal Editor. The whole tree contains resources (like levels, materials, textures, blueprints, etc.) which were stored under **Content/VirtualStudioA**.



## Changing Scenes on Air

Content prepared by the designer has to be cooked for the target platform and deployed to an available shared space on a HDVG machine (Control\_Data).

To enable this, the Unreal Editor contains an additional button which triggers an automatic process of cooking and packing the scene for a Linux based platform. It also deploys the scene for a future usage on a HDVG machine. The separation between the content and the launcher allows for a free scene manipulation in runtime. This in turn gives the controllers the possibility to load/unload content and show it without having to restart the rendering process.

## Preview for Scene Creator

Additionally, the Unreal Editor can still be employed as a real preview of the scene, using the tracked camera object available as an Add-on. This way, you could adjust final output parameters and settings in the Unreal Editor. Additional controls are built into the Unreal Editor to allow simulated control over the tracked camera for testing purposes.

## Way of Handling Clips and Video Insertions

Currently, MediaPlayer is not supported directly from an Unreal scene. Our solution takes advantage of the mixing with Render Engine functionality, which is handling all tasks related to Clips and Video Insertions.

## Plugins Support

Presently, our solution supports only content plugins from the Marketplace and external sources. There are no further restrictions, as long they follow the resource structure mentioned above.

For plugins containing any C++ code the support is limited as the cooked scene is migrating between platforms and it is controlled on-air.

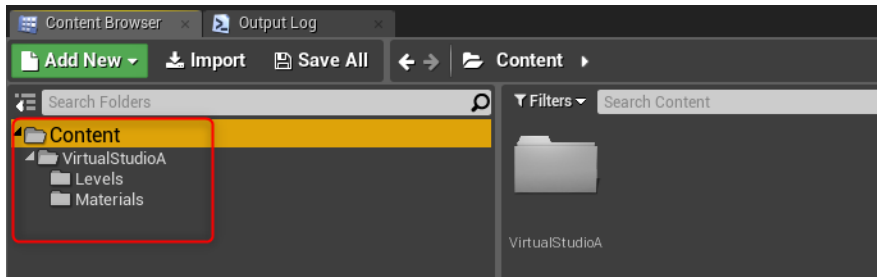
## Project Preparation

The Unreal Editor can be launched from the Desktop shortcut or from the Windows Start Menu.

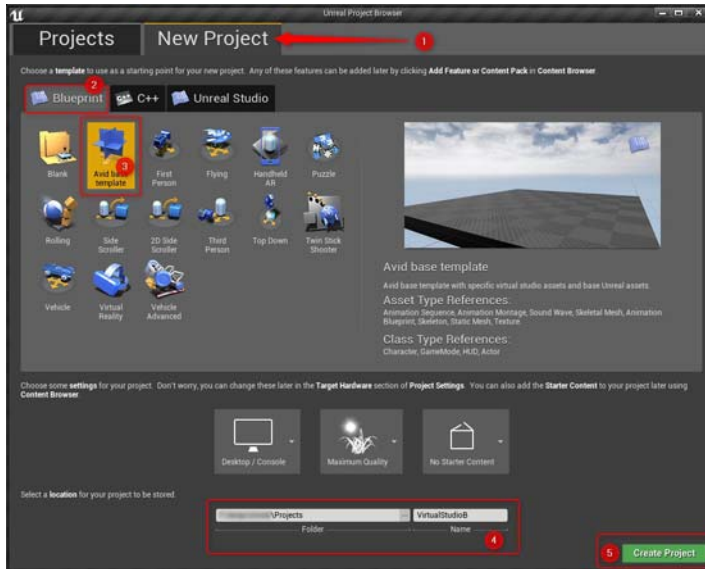


Both ways launch the Unreal Editor in the OpenGL4 mode.

Use the Blueprint project only and place all resources under an appropriate folder in **Content**.

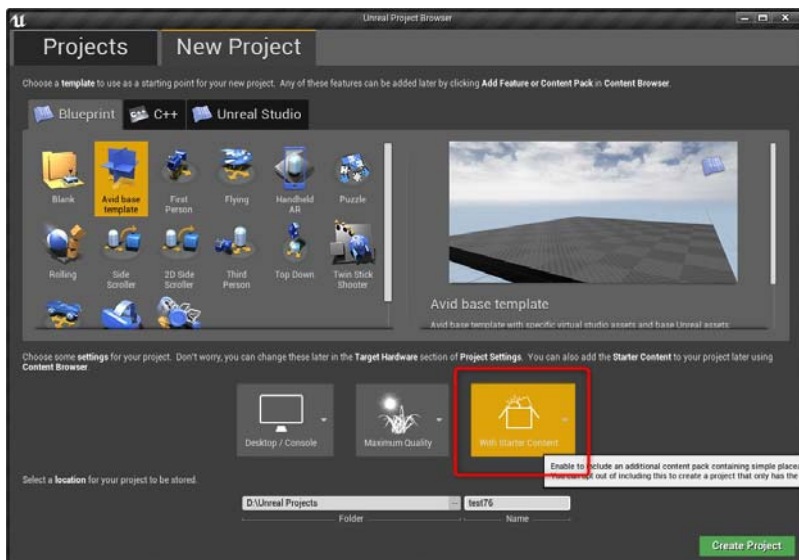


The Avid template projects should be used as a starting point.

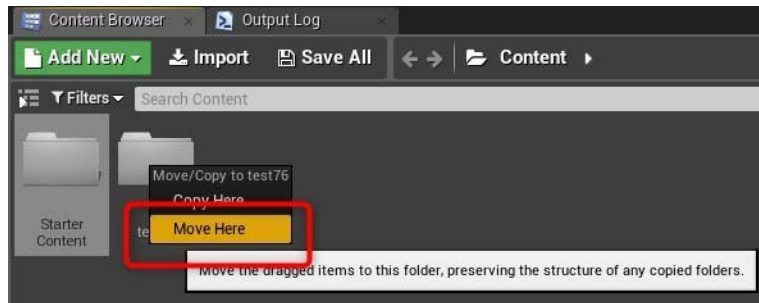


### New Scene with Starter Content

Creating a new scene with the starter content option requires additional steps.

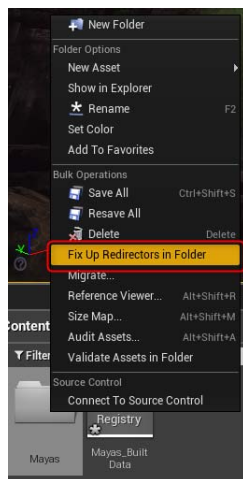


After creating a new scene using the Starter Content, you should move the content directory into the project directory:

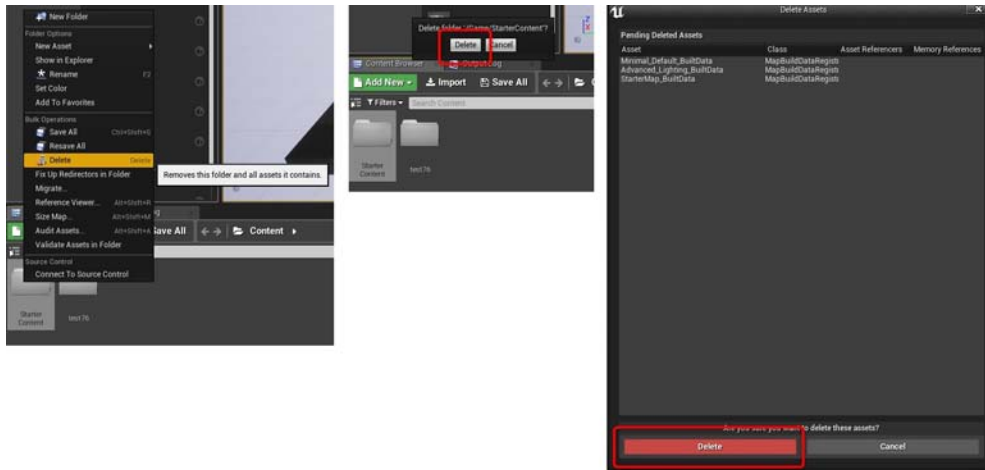


*Because of caching inside the Unreal Editor, it is sometimes required to fix the references.*

*To fix the references, right-click on the target folder and select the **Fix Up Redirections in Folder** option.*



After moving the content directory into the right place, you can delete the empty directory:



## Importing Content

Supplementary Unreal assets can be found on the Marketplace and added directly to the project using the **Epic Games Launcher** (as illustrated on the picture below) or migrated from a different project directly to the appropriate structure inside Content as described in the Unreal Engine Migrating Assets (<https://docs.unrealengine.com/en-us/Engine/Content/Browser/UserGuide/Migrate>) user guide.



External resources can be imported from FBX files which are created with applications such as Autodesk MotionBuilder, Autodesk Maya, and Autodesk 3ds Max. The approach is supporting particular **objects** (<https://docs.unrealengine.com/latest/INT/Engine/Content/FBX>) as well as **full scenes** (<https://docs.unrealengine.com/en-us/Engine/Content/FBX/FullScene>).

## Creating Map (level)

After the project preparation, you focus on the **Level** creation. Using Unreal Engine 4 terminology, a Level is made up of a collection of Static Meshes, Volumes, Lights, Blueprints and more. All working together to bring the desired virtual environment. All these elements are used afterwards to render a virtual studio on a dedicated rendering machine (HDVG). More information about Levels can be found in the **Unreal Engine Levels** (<https://docs.unrealengine.com/en-us/Engine/Levels>) manual. The actual Level name should be in line with the Project name to simplify controlling.

For preview purposes, a creator should add an **AvidCamera Pawn** instance to allow tracked camera preview.

## Preview

Any time during the Authoring process, an author is able to play the level to see how it will be rendered. When using an **AvidCamera Pawn** actor on a scene, an author can take control over a tracked camera and simulate its working directly from the editor. It can be also needed to set some additional settings and post-processing effects.

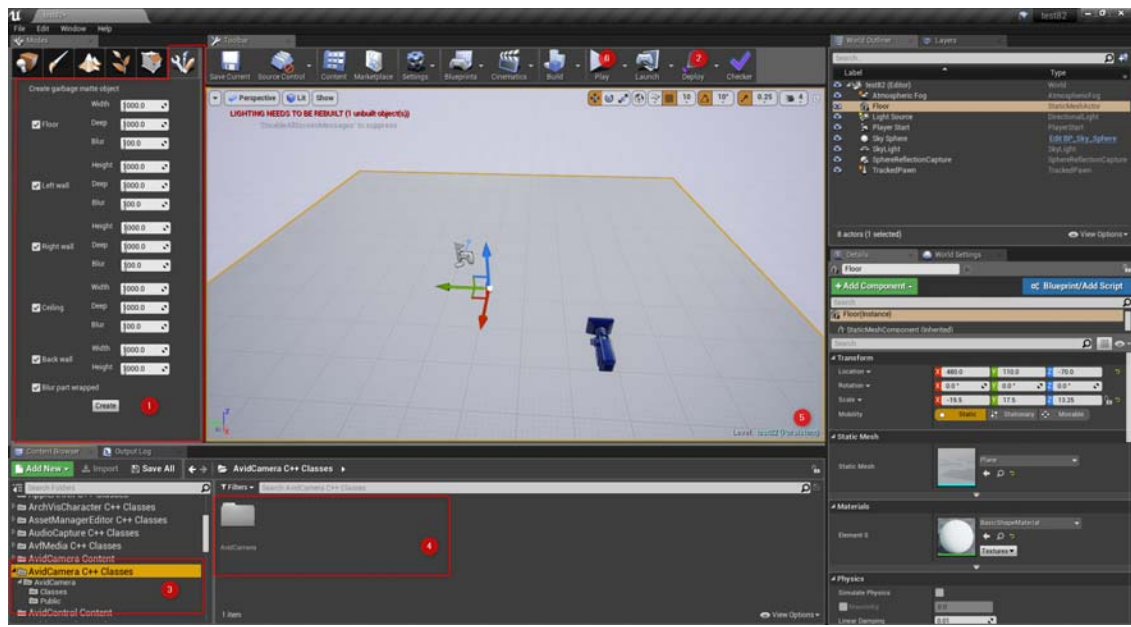
## Final Cooking

After finishing the map preparation, an author triggers deploying the project into **Control\_Data** and thus makes it available to the render machine and controller.

## Unreal Editor Extensions

The following Unreal Editor extensions are available:

- 1 click solution to cook the map (level) package and deploy it to HDVG.
  - cross-compilation for Linux platform
  - store the cooked map (level) in Control\_Data
  - mimic scene\_data.xml file to simplify controlling
- Tracked camera visualization
- Defining exports for level components and other parameters
- Additional content, e.g.:
  - Actors
  - Blueprints
  - Textures



Item	Description
1	Garbage matte object creation plugin
2	Map (level) cooking and deployment
3	Additional content inside the content tree
4	Additional content inside the content browser
5	Editor viewport
6	Play the level with fake tracking

## Working with Exports

Maestro | Virtual Set provides maximum flexibility for manipulation of on-air graphics in real time. By exporting parameters from Maestro | Virtual Set Authoring to the Control module, data can be sent to any actor on a scene (e.g. animated sequences, object properties, and text strings), allowing them to be updated in real time. For example, a bar may grow and change color in real time to represent the score and colors of the winning team.

Defining Exports is straightforward thanks to the built-in Blueprint graph editor in the Unreal Editor. Avid Maestro | Virtual Set Authoring plugins allow to connect any action and object property to the named Export, which is easy to control from an external controller.

An export is a predefined user function available in the Blueprint Editor. The name of the function is the actual name of the export and an external controller can use it to send data through a common communication protocol (ReTalk3). The solution combines simple use cases as well as complex functionalities involving multiple objects and math, which allows for animated actions based on real-time data.



## Example 1: Registering of text content for a TextRenderActor object

The below procedure describes how to register content for a particular TextRenderActor object.

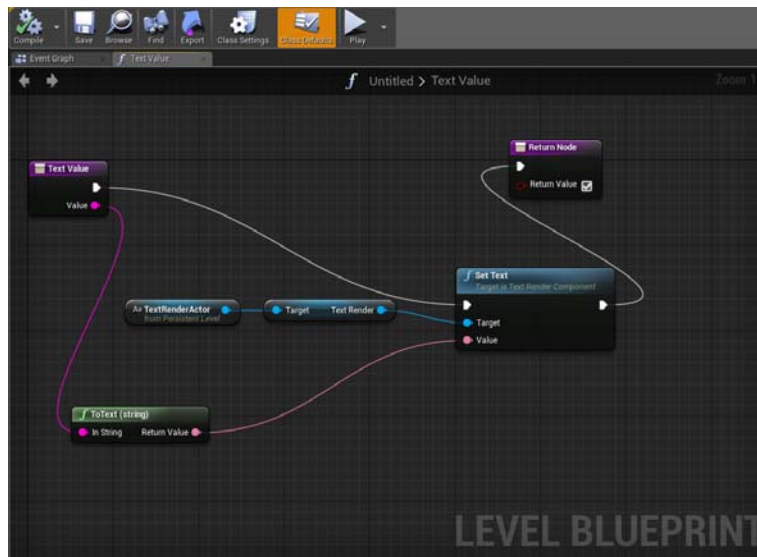
**To register the text content for a TextRenderActor object:**

1. Click the **Create Export** button in the Blueprint Editor.  
A new function with predefined signature is added to the list.
2. Change the function name to something meaningful. In this example, **Text\_Value** is used.



*The actual name of the export function can be changed anytime. It will be set in a scene during the Deploy process.*

3. Create logic inside the function. In our example, text for a particular TextRender is set.

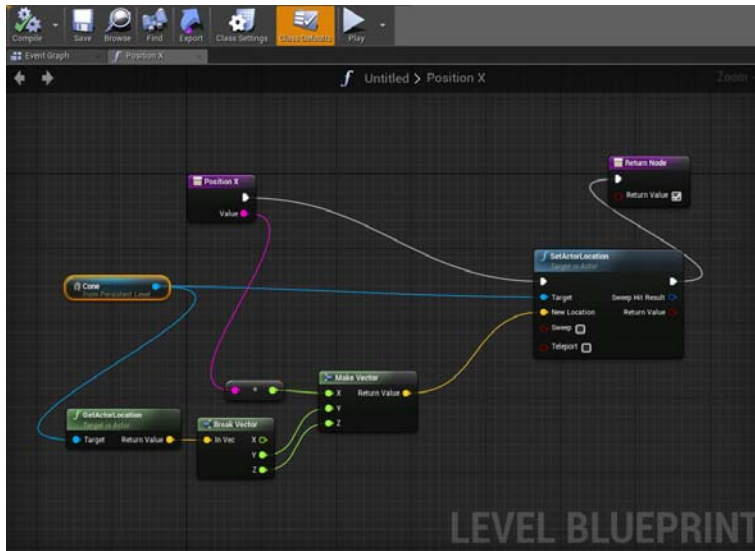


## Example 2: Registering of an X axis position for a simple Cone object

The below procedure describes how to register the X axis position of a Cone object.

**To register the X axis position of a Cone object:**

1. Click the **Create Export** button in the Blueprint Editor.  
A new function with predefined signature is added to the list.
2. Change the function name to something meaningful. In this example, **Position\_X** is used.
3. Create logic inside the function. In our example, text for a particular Cone object is set.



The list of available exports will be visible inside the Controller UI and the real value can be changed in runtime.

## Helper Functions

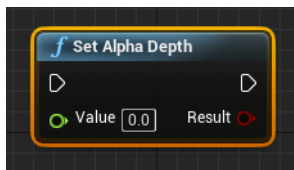
Additional helper functions can be used in the BlueprintEditor for various actions such as connecting to exports.

The AvidCamera plugin provides helper functions to manipulate some of the parameters:

### Set Alpha depth

Set the distance to which, the alpha will be generated.

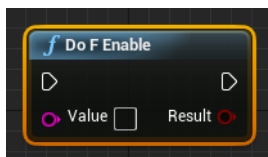
Value: distance [cm]



### DoF Enable

Enable/Disable Depth Of Focus (DoF).

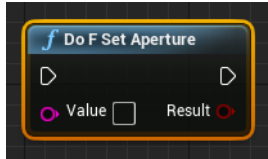
Value: 0/1



## DoF Set Aperture

Set Aperture if not provided by camera tracking.

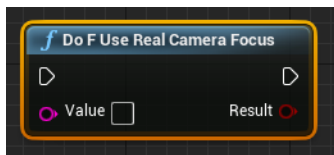
Value: Aperture as number (eg 5.6)



## DoF Use Real Camera Focus

Use focus distance from camera tracking.

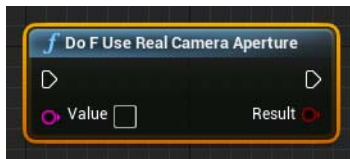
Value: 0/1



## DoF Use Real Camera Aperture

Use the Aperture value from camera tracking (if provided).

Value: 0/1

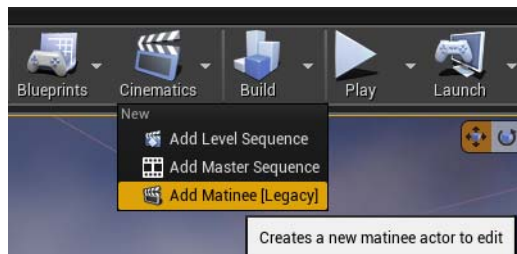


# Working with Animations

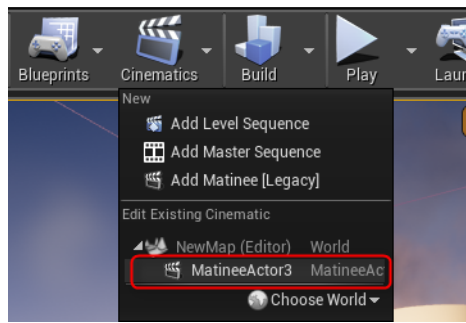
The SceneLoader supports animations done by a MatineeActor or LevelSequence.

**To create an animation with a Matinee Actor:**

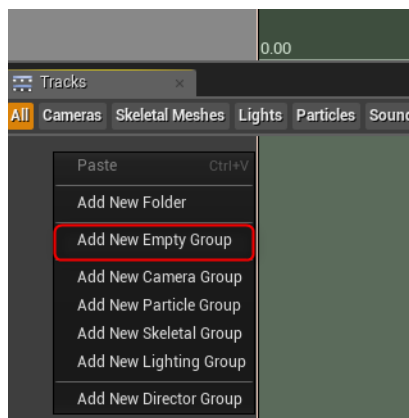
1. Add a Matinee Actor to the level,



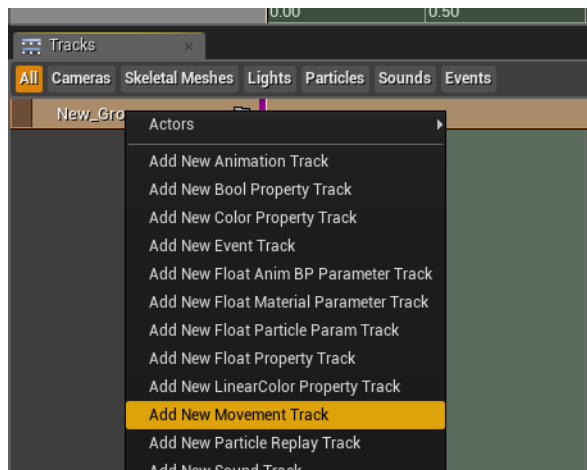
or edit a Matinee Actor.



2. Add an empty group.

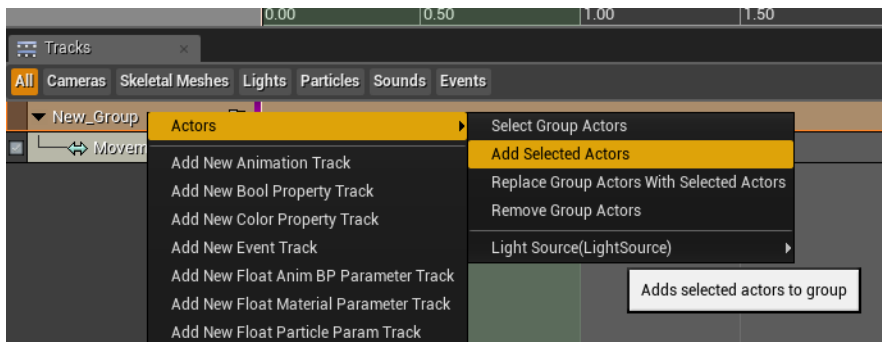


3. Add a track, for example a movement track.



4. Add actor(s) to the group.

- a. Select the actor(s).
- b. Add the selected actor(s) by right-clicking on the group.

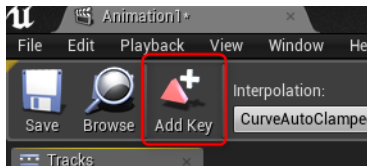


5. Add keyframes.

- a. Move the time slider to the desired time position.



- b. Move the actor (movement track).
- c. Add the keyframe.

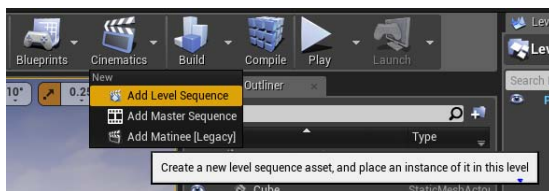


*Repeat these steps to add more keyframes.*

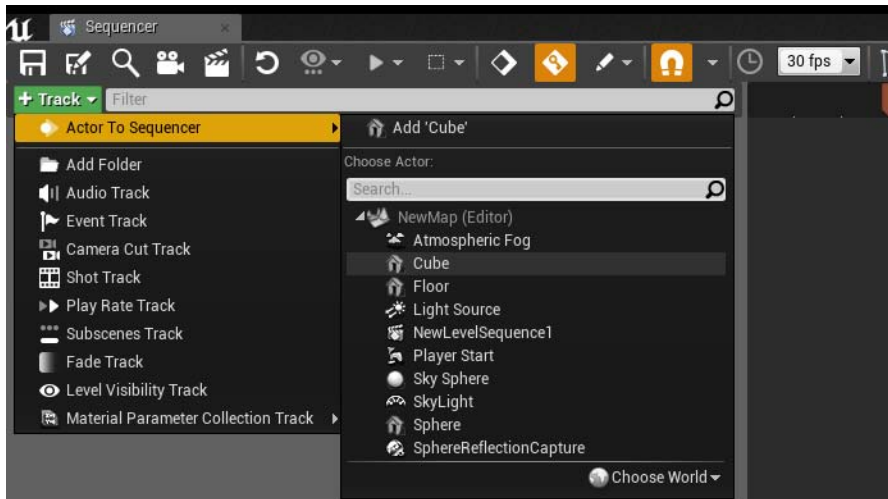
*To have the correct animation length in a Controller, keep the timeline length the same as the last keyframe of the animation.*

**To create a Level Sequence animation:**

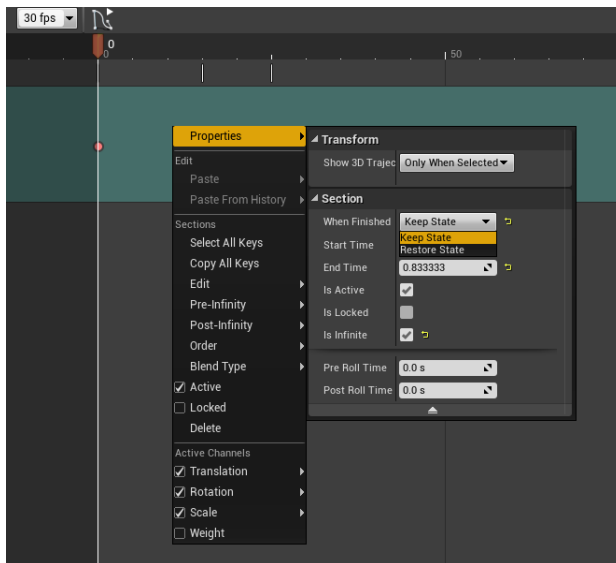
- 1. Add a Level Sequence to the level.



- 2. Add a track for a specific actor.

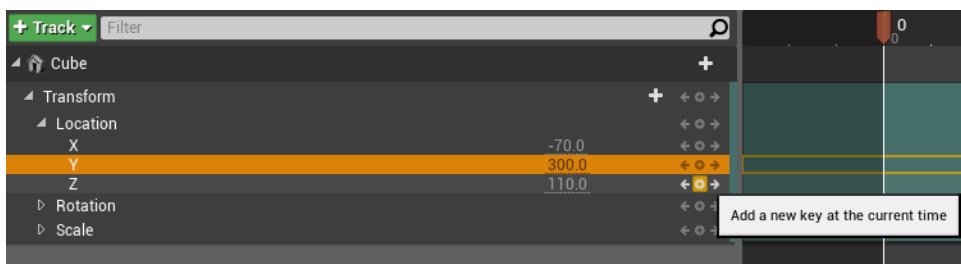


3. Right-click on the track and change **Properties > When finished** to **Keep State**.



4. Add keyframes.

- a. Move the time slider to the desired time position.
- b. Change the value of the property that you are animating.
- c. Add the keyframe.



 Repeat these steps to add more keyframes.



To have the correct animation length in a Controller, keep the timeline length the same as the last keyframe of the animation.

## Skeleton Animations

Skeleton animations can be controlled as simple exports.

In the example below, the idea is to have the same objects with the same animations sequences, but not running in a parallel motion. As presented on the image below:

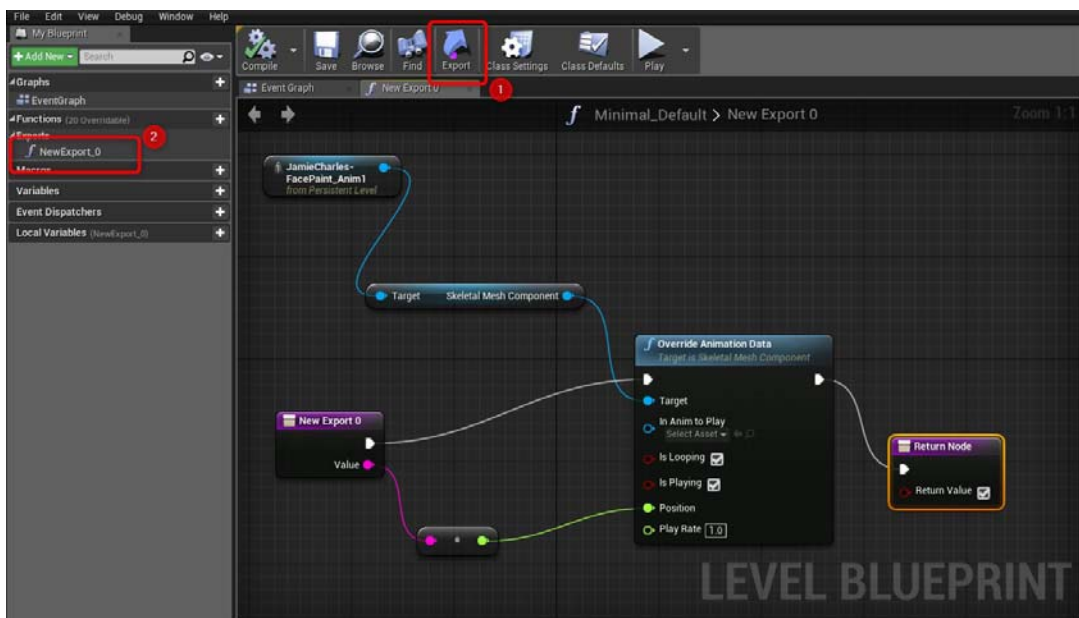


Offset for each of players are controlled by the following property **Animation > Initial Position**:





Using The Blueprint Editor, you can attach an Avid export to the selected property, as illustrated on the example below:





## Working with Alpha Matte

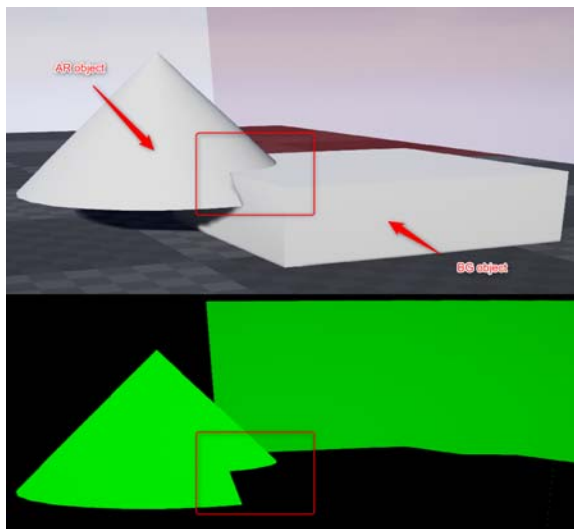
The Maestro | Virtual Set solution is able to produce a **Fill+Key** output. The Key consists of Matt for augmented reality (AR) objects and of garbage Matt. It can be used by ChromaKey to mix Virtual Studio background, camera feed foreground and AR elements on top of that.

### Augmented Reality Objects

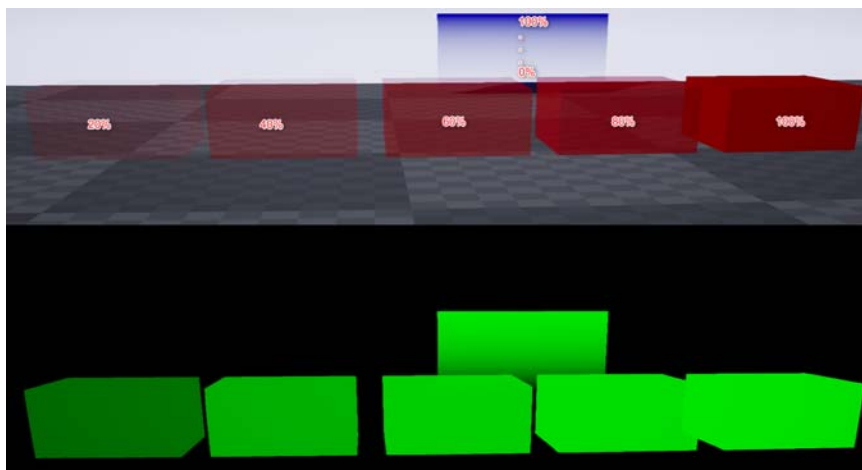
AR objects should be rendered on top of a talent in the Virtual Studio.

#### Properties

The alpha mask is computed taking depth test into account.



For translucent objects, Maestro | Virtual Set generates the Alpha Matte value according to their Opacity.



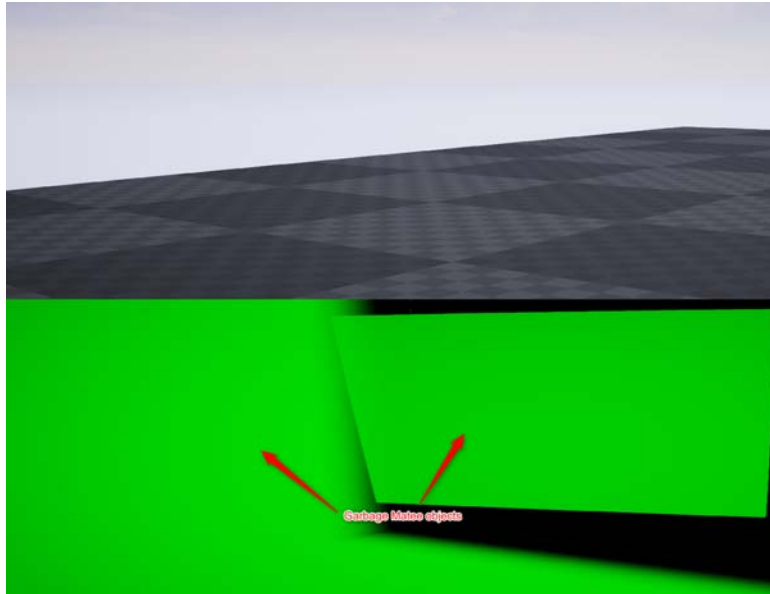
*Translucent objects will be mixed with the Background graphics and not with the talent.*

## Garbage Objects

Garbage objects are used when overshooting to composite a Virtual Studio into the shot that is larger than the green screen.

### Properties

These objects are not rendered on the Fill output - they exist only in the Key.

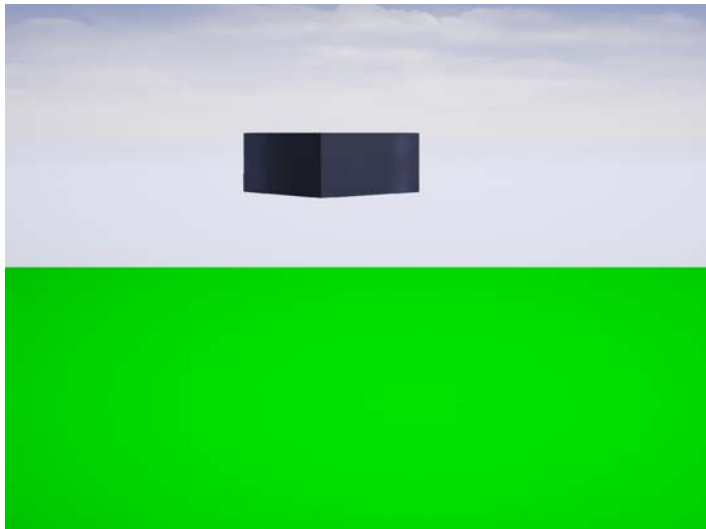


These objects do not take depth into account - the whole Garbage geometry will be visible on the Key output.

### Editor View



**Fill + Key**



The Alpha Matte value is according to Opacity of Garbage objects to allow for creation of smooth edges.

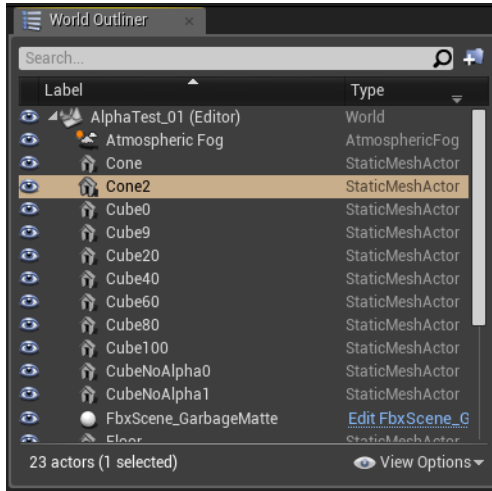


# Alpha Matte Composition (Helpers)

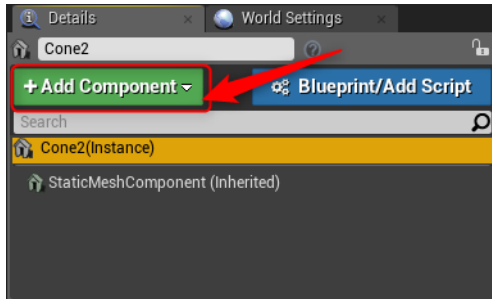
## AR Objects

To create Alpha Matte composition for AR objects:

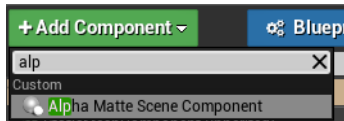
1. Select an object from World Outliner which should be visible on top of a talent.



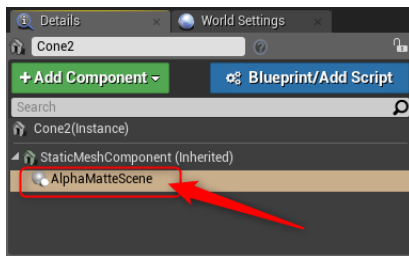
2. Click **Add Component** in the **Details** panel.



3. Start typing "alp" in the Search box to find the **AlphaMatteSceneComponent** and press Enter.



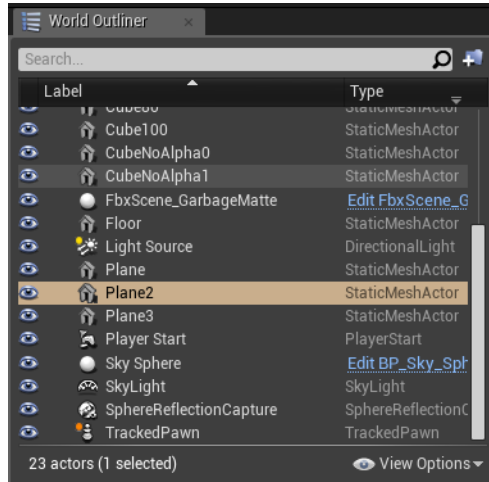
4. A component is added which will mark the object in the runtime for AR use case.



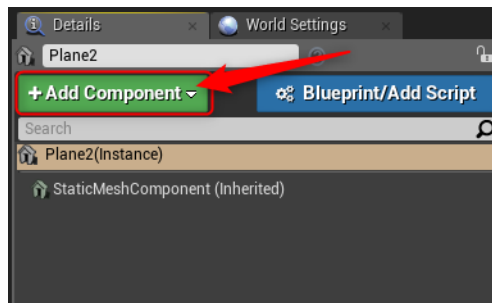
## Garbage Objects

To create Alpha Matte composition for Garbage objects:

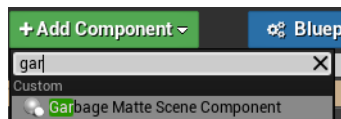
1. Select an object from World Outliner used when which overshooting to composite a virtual set into the shot that is larger than the green screen.



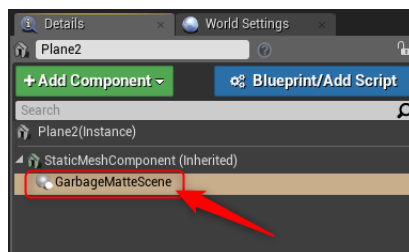
2. Click **Add Component** in the **Details** panel.



3. Start typing "gar" in the Search box to find the **GarbageMatteSceneComponent** and press Enter.



4. A component is added which will mark the object in runtime for Garbage use case.



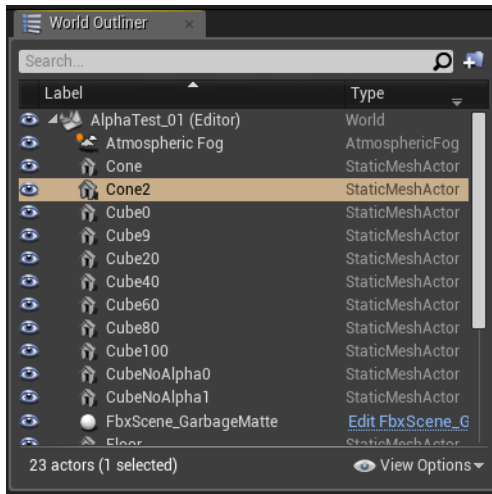
# Alpha Matte Composition (Manual)

## AR Objects

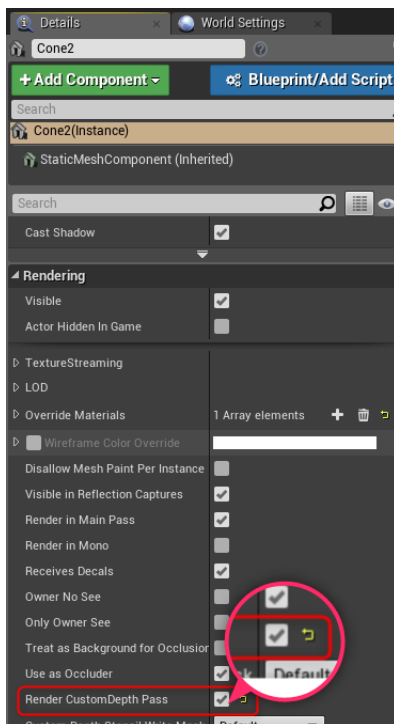
To mark an object to be an AR object and be rendered in Key, it has to be marked as rendered during CustomDepth Pass.

To manually create Alpha Matte composition for AR objects:

1. Select an object from World Outliner which should be visible on top of a talent.

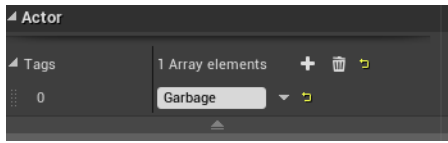


2. Enable **Render CustomDepth Pass** in the **Rendering** section of the **Details** panel.

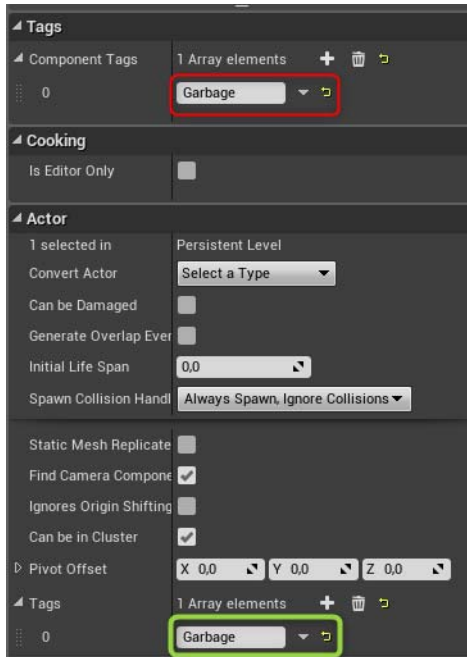


## Garbage Objects

All objects with tag **Garbage** will be treated as garbage objects which means that they will not be seen in the Fill viewport, but they will be visible in the Key viewport.



Use only Actor Tags (marked with green on the image below).



*Garbage objects should not be marked to be rendered in CustomDepth pass.*

## Troubleshooting

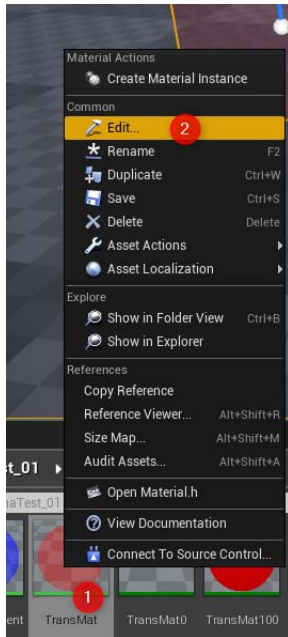
### Translucent Materials

Custom Depth does not work on translucent materials. For translucent materials, you need a second copy of the mesh using a simple opaque material with Custom Depth enabled and Render Main pass disabled. More information can be found on the [AnswerHub](#).

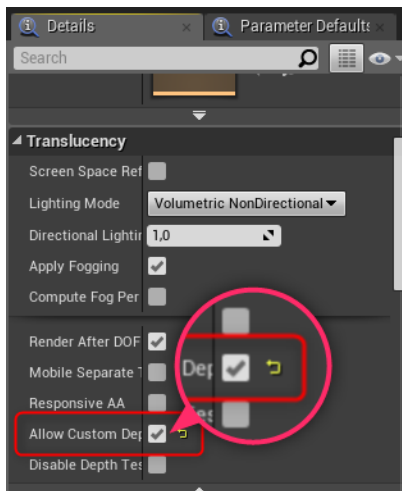
For simple object using **AlphaMatteSceneComponent** helper resolves this problem.

To force CustomDepth on Translucent materials used in an object:

1. Open the material editor.



2. Enable **Allow Custom Depth Writes** in the **Translucency** section of the material's **Details** panel.



*Because of Unreal optimizations, materials with opacity <30% are not rendered in the Custom Depth buffer at all. The option with additional mesh is the only solution then.*

## Generating Garbage Matte Objects

Garbage matte objects allow to create objects to mask part of a scene outside of the green or blue box, to generate chroma key outside of it.

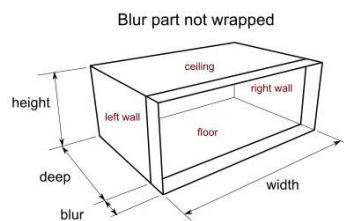
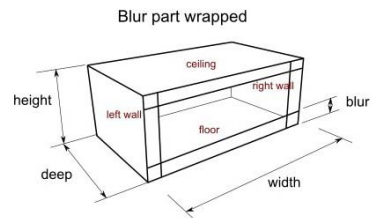
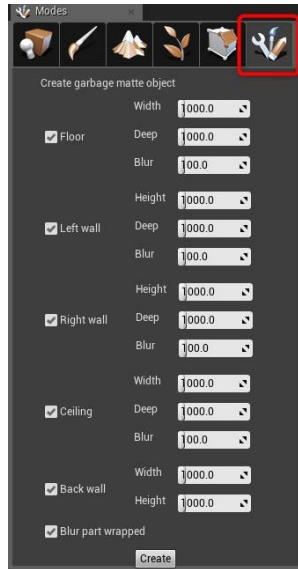
The blur part of garbage matte objects allows for a smooth transition between the keying areas.



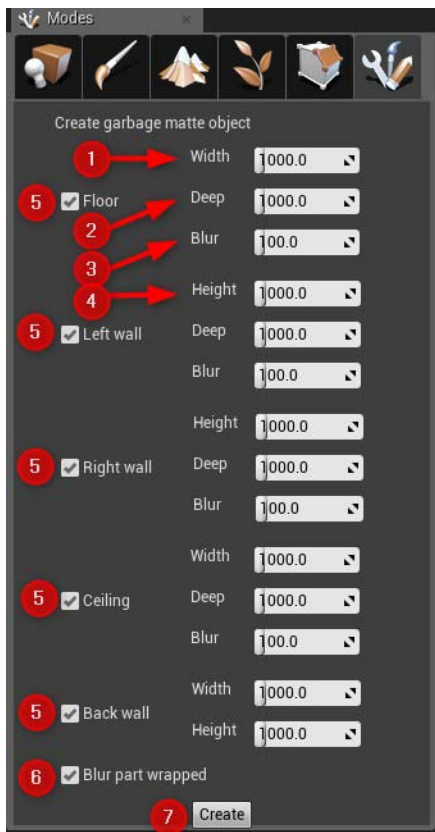
## To generate a garbage matte object:

1. Open the Avid Garbage Matte Editor.

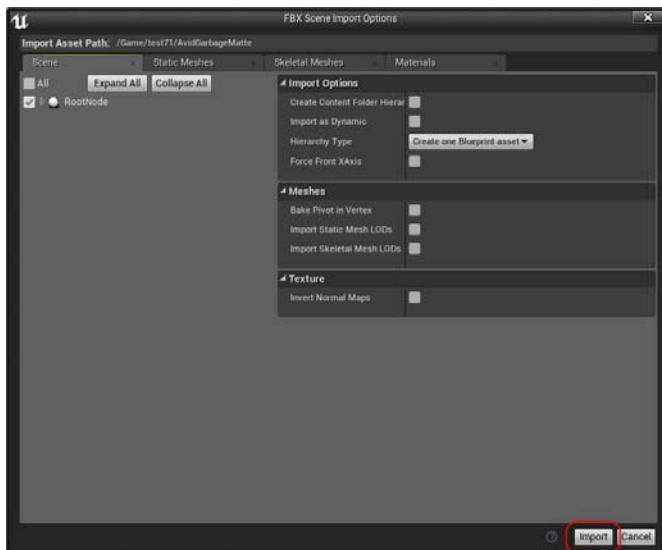
It allows to define a size for each of garbage matte object part. Additionally, it also gives the option to select one of two ways of garbage matte generation: wrapped blur part and not wrapped.



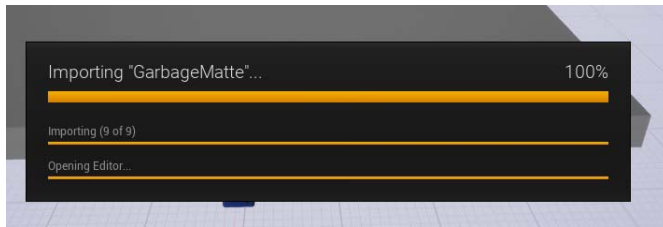
2. Enter the dimension values in cm, specific to your scene (1-4).  
Select the parts of the garbage matte object that you want to generate (5).  
Select the type of the object (Wrapped/ not wrapped) (6).  
Press the 'Create' button (7).



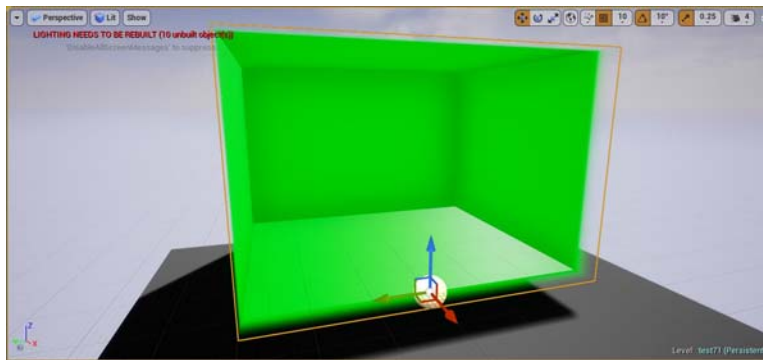
3. After pressing the 'Create' button, the FBX Scene Import Option window appears.  
Press the 'Import' button.



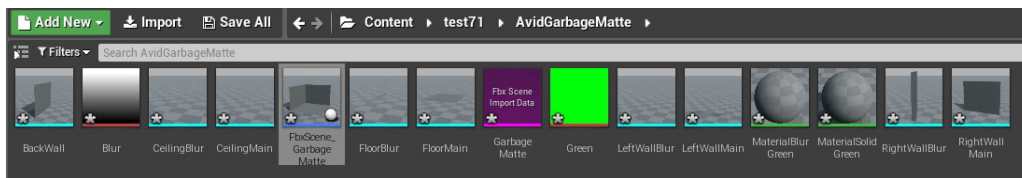
4. The window showing the progress of import appears for a few seconds.



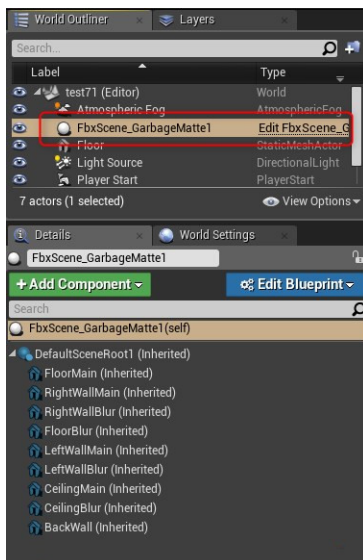
5. The geometry of the garbage matte object appears in your scene.



In the Content Browser window, you can see the components of the garbage matte object.



In the World Outliner you can see all components.



# Working with Depth of Field (DoF)



An in-depth description of the Depth of Field feature can be found in the Unreal Documentation at <https://docs.unrealengine.com/en-us/Engine/Rendering/PostProcessEffects/DepthOfField>.

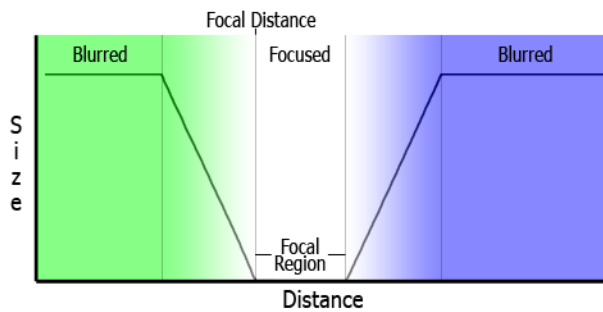
Depth of Field is broken up into three layers - Near, Far, and the Focal Region - each of which are processed separately and then later composited together to achieve the final effect. Objects in the near and far layers - objects not in the focal region - are always fully blurred. Then, these layers are blended with the non-blurred scene.

- Objects within the focal region use the non-blurred scene layer.
- Objects in the near or far layers, but outside any transition regions, are fully blended to the blurred layer.
- Objects within a transition region are blended linearly between the non-blurred scene layer and their blurred layer based on their position within the transition region.

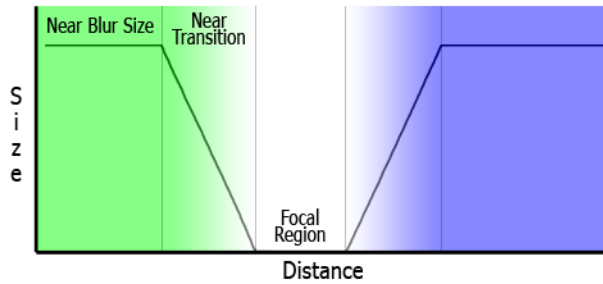
## Circle Depth of Field

Important parameters (which could be exposed as export parameters):

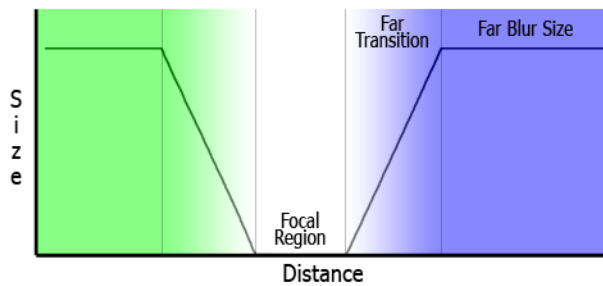
- Enable/Disable
- Aperture F-stop - it defines the Iris
- Focal distance in cm
- Depth Blur km for 50%
- Depth Blur Radius - Depth blur radius in pixels at 1920x
- Focal Region in cm



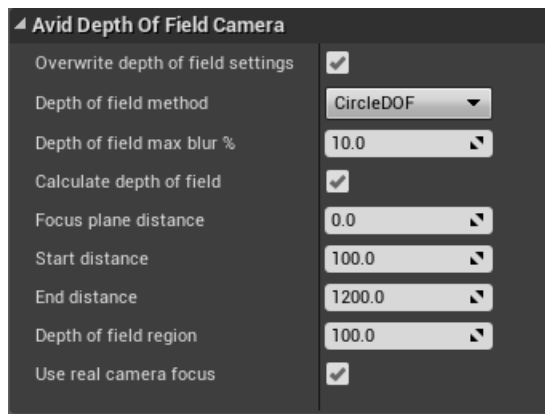
- Near Transition Range in cm



- Far Transition Range in cm

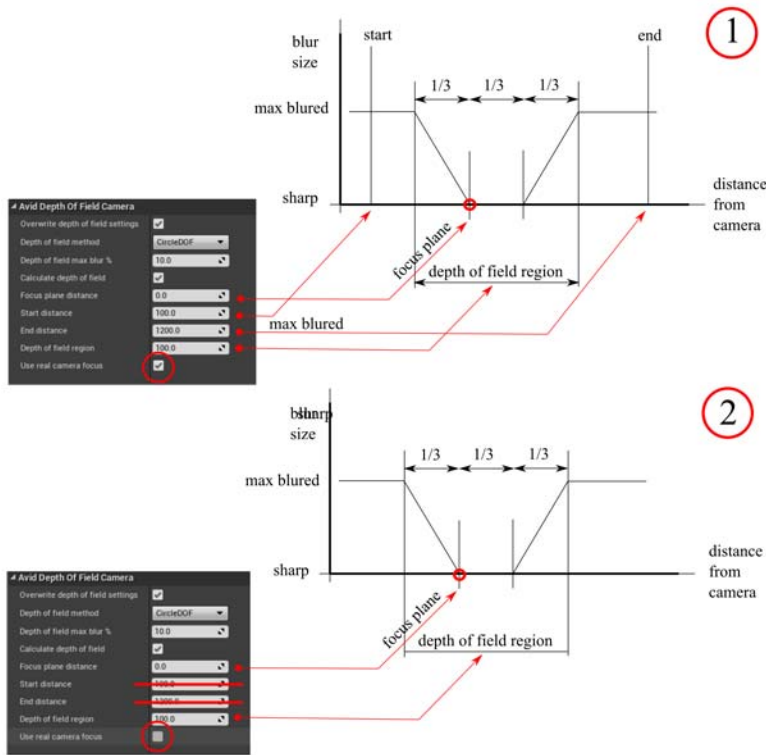


### Depth of Field Implementation Using Information from Tracking Camera



Parameter	Description
Overwrite depth of field settings	When enabled, it sets the current method of DoF depending on the next field - "Depth of field method" and changes the max blur size depending on the field "Depth of field max blur %".
Depth of field method	Select the DoF calculation method.
Depth of field max blur %	Change the max blur size.
Calculate depth of field	When enabled, it calculates the min and max focal region.

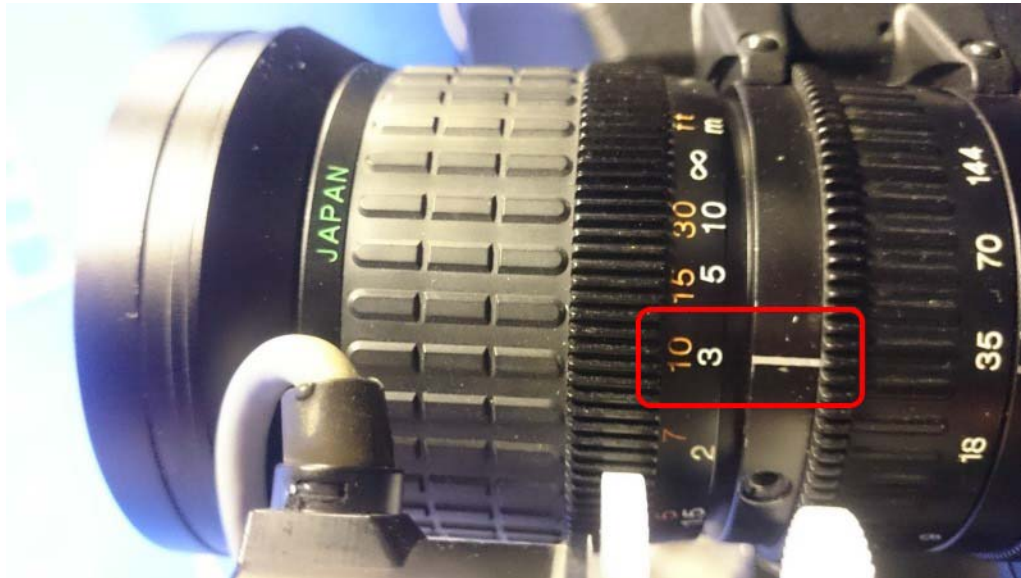
Parameter	Description
Focus plane distance	Value taken from Maestro   Tracker. It represents the real focal plane.
Start distance	Value used for calibration of focus in the scene, it represents the value of the minimum focal plane in the scene.
End distance	Value used for calibration of focus in the scene, it represents the value of the maximum focal plane in the scene.
Depth of field region	It represents the region which is a sum of "Near transition" + "Focal region" + "Far transition"
Use real camera focus	Switch between two methods 1: normalized focal plane value, 2: real focal plane value.



### Adding Focus Plane Values to Lens Configuration File

The focal plane values can be calibrated manually in relation to the focus ring position and added to the lens calibration file.

To do so, position the focus ring on the lens on each distance mark (eg. 3m distance mark set on the lens):



Next, read the focus sensor value on the ASB-9 (go to **STATUS > FOCUS**):



or in the TsDispCamera with the -r option:

```
Tracker 1 raw axes:
```

ID	NAME	VALUE	HEX
0	PAN	0	00000000
1	TILT	1	00000001
2	ZOOM	4138	0000102A
3	FOCUS	3777	00000EC1
4		0	00000000
5		0	00000000

Using these values, create a FOCUS PLANE section at the beginning of the lens.ecf file, just after the comments and the CCD SIZE field:

```
FOCUS PLANE nr_of_samples  
  
Focus_value_1 Distance_value_1  
  
Focus_value_2 Distance_value_2
```



*Distance inside the lens.cfg file is entered in cm.*



**The focus values order must be the same as in the FOCUS NODAL section.**

For the Near focus position, set the lens minimum focus distance or 0 if unknown.

For the Infinity focus position, set a distance at least twice as big as the one entered in Near focus.

Example of a FOCUS PLANE section with 9 rows:

```
CCD SIZE 2/3 INCH
```

```
FOCUS PLANE 9  
257 2000  
1574 1000  
2451 500  
3731 300  
5455 200  
7152 150  
8915 120  
11900 90  
12469 50  
FOCUS NODAL 20
```

To use the calibrated focus distances, the Unreal DoF configuration must be setup as follows:

- Depth of field method: CircleDoF
- Calculate depth of field: On
- Depth of field region: 1
- Use real camera focus: On



## 4 Building an Unreal Engine Scene

In this section:

- [Building Scenes with the “Avid Base Template”](#)

### Building Scenes with the “Avid Base Template”

This chapter includes a description of how to create a new Unreal Engine scene using the Avid Base Template.

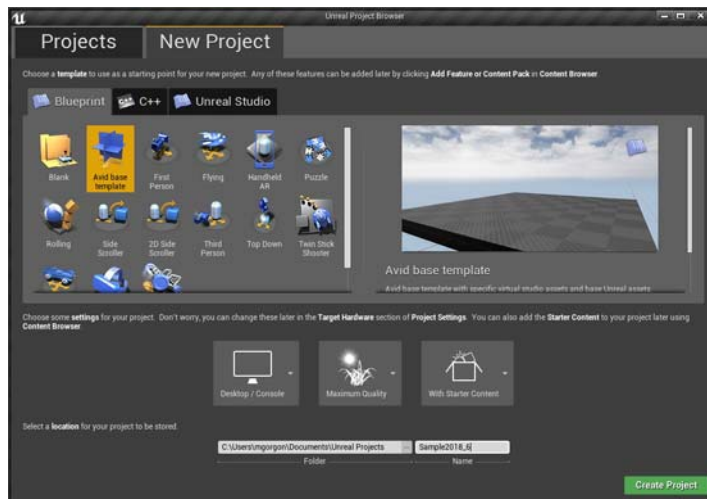
**To create an Unreal Engine scene with the Avid Base Template:**

1. Run the Unreal Editor from the desktop shortcut.

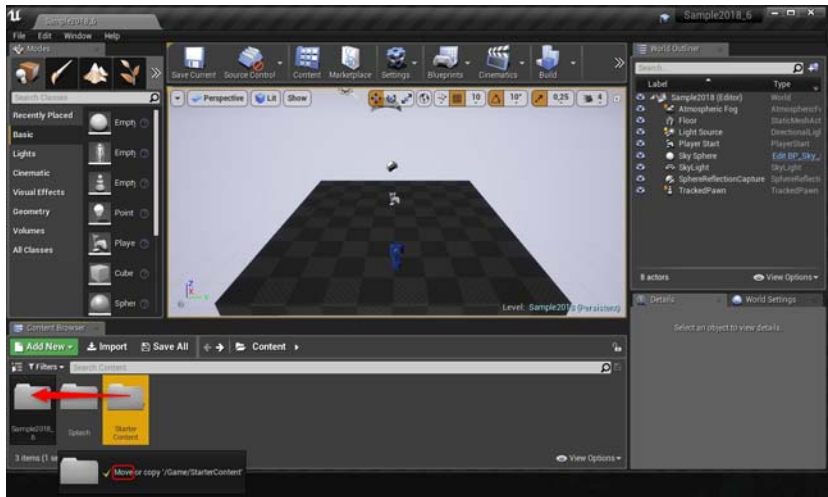


*This shortcut opens the Unreal Editor in the OpenGL mode. This mode is recommended to have the highest compatibility of scenes built on Windows, which are later used on Linux.*

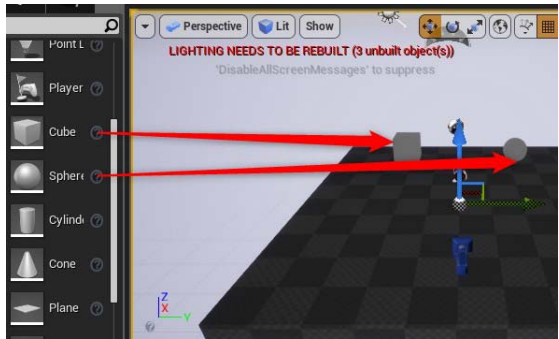
2. Select “New Project” and choose Avid base template. Add a name for your project (in our example we will use the name Sample 2018\_6).



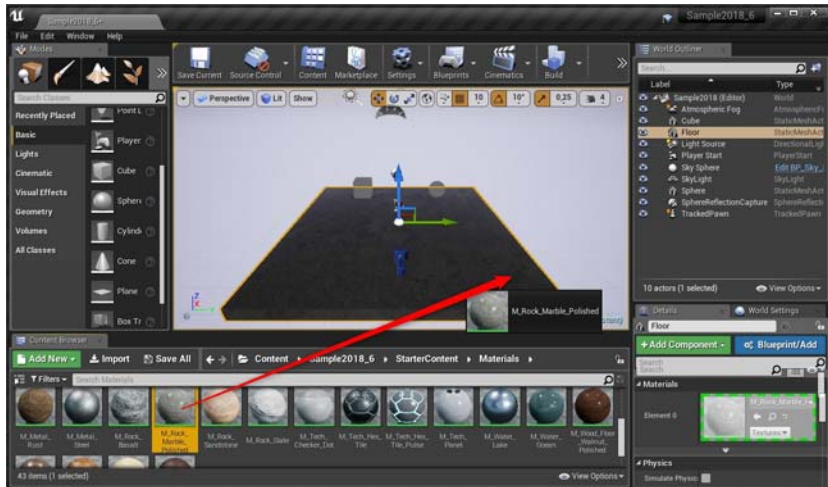
3. Move the StarterContent to your newly created (Sample 2018\_6 in our case) directory. Then, delete the StarterContent folder in the project root.



4. Add a Cube and a Sphere to the scene.



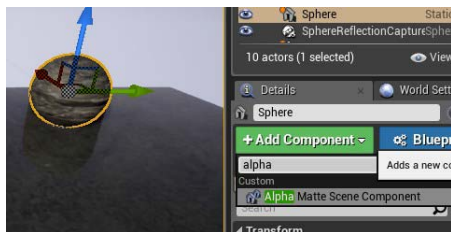
5. Go to the StarterContent and assign sample materials to the floor, the cube and the sphere.



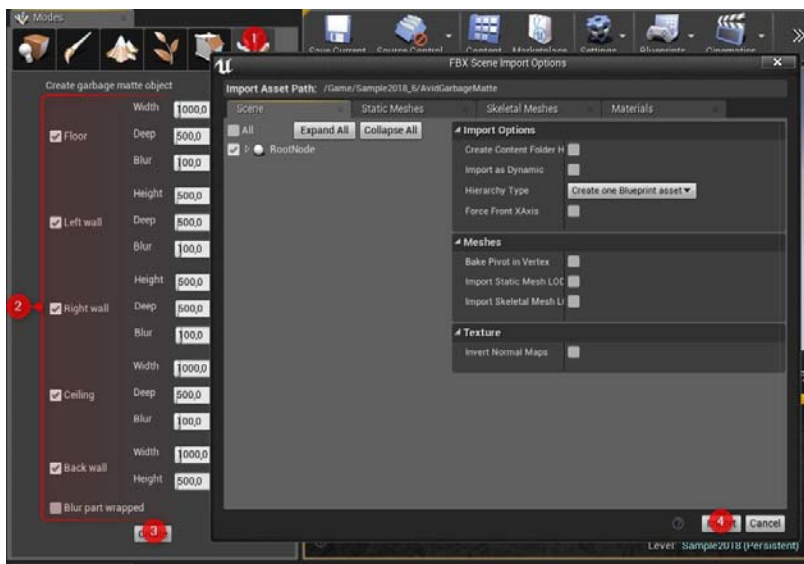
6. Change the sphere's and cube's mobility from static to movable. This is required to enable transformation changes for these objects from animations or exports.



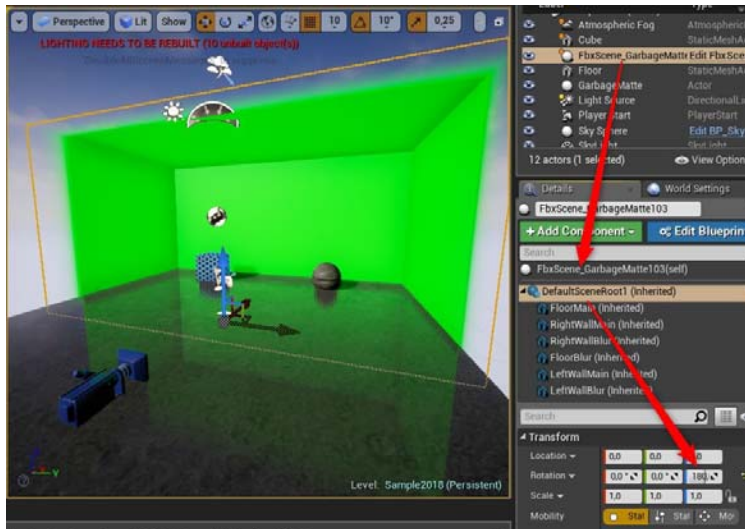
7. To enable alpha output for objects on the scene, you need to add an Alpha Matte Scene Component to the object instance.



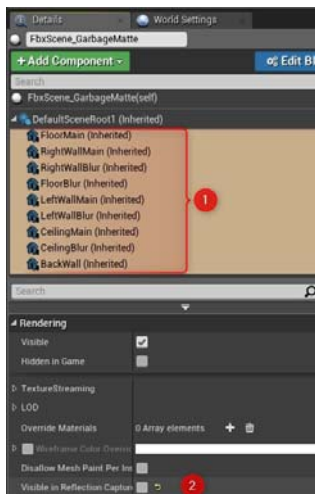
8. Now you can generate Garbage Matte.



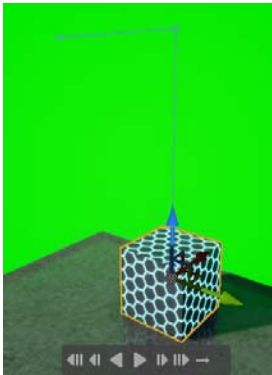
9. Rotate the generated garbage mask to the camera view.



Garbage matte casts shadows. This is a known issue. Garbage matte is only reflected on shiny surfaces. To eliminate this issue, you need to select all garbage objects and disable the “Visible in Reflection Capture” option. For more information, see “Frequently Asked Questions” on page 59.



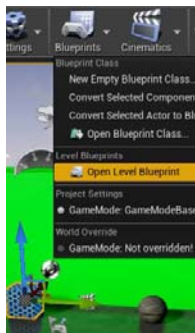
10. Now you can add two types of animations for your cube and sphere objects. For more information on animations, see [Working with Animations](#).
- Create an animation in “Level sequence” named animation\_seq for the Cube:



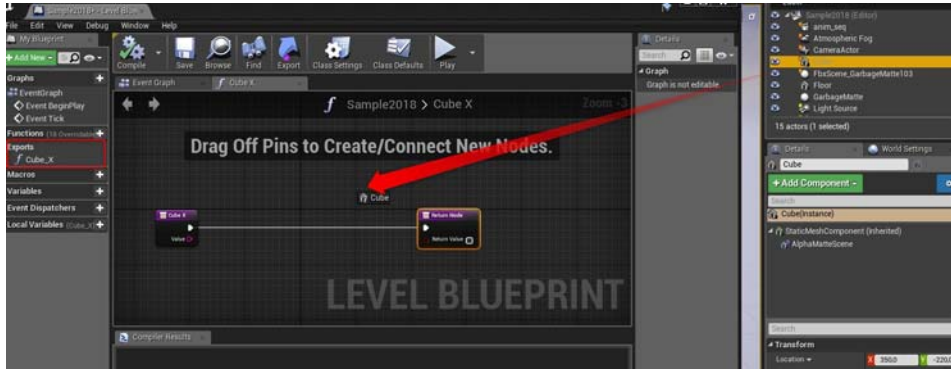
- Create a “Matinee [Legacy]” animation for the Sphere:



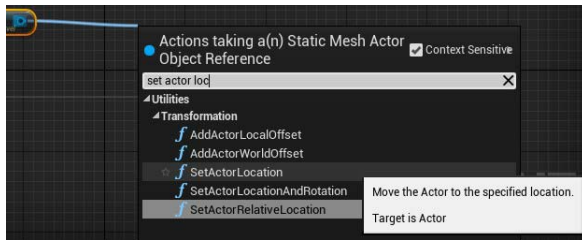
11. To create exports, you need open **Level Blueprint**.



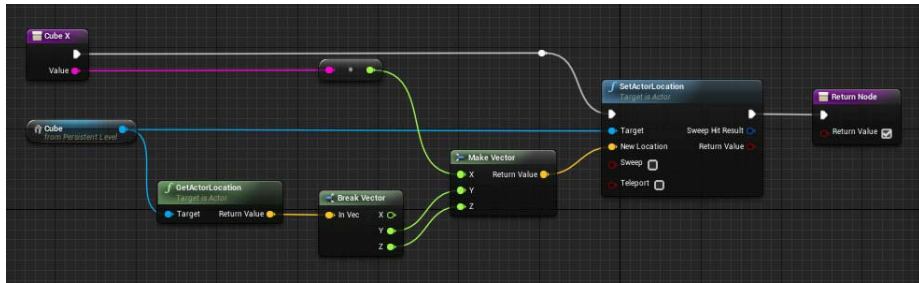
12. Click on **Export** to create a new export. Name it “Cube\_X”. Drag and drop the “cube” property to the Node Editor.



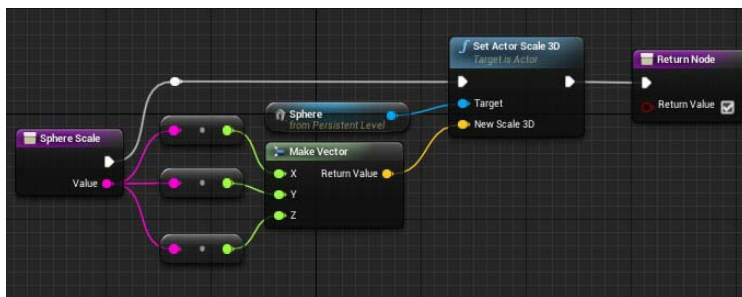
13. Drag and drop the target from the Cube Node and search for “set actor location”:



Export for changing the X position of the Cube should look like the image below:



Create a Sphere\_Scale export:

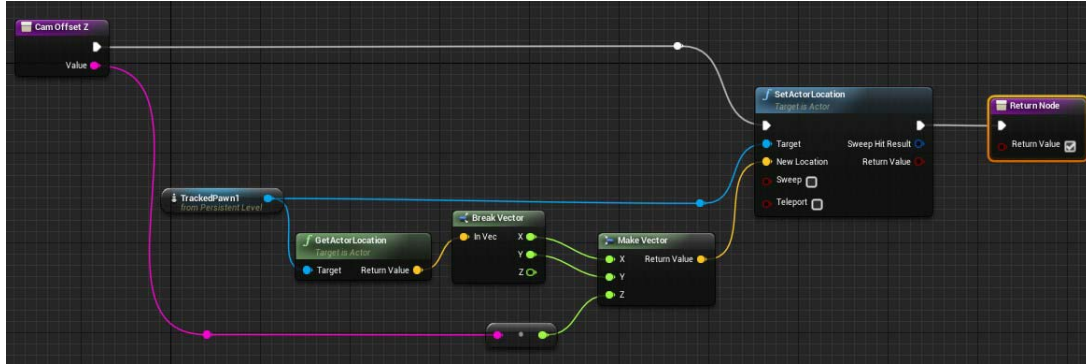


14. To adjust floor level:



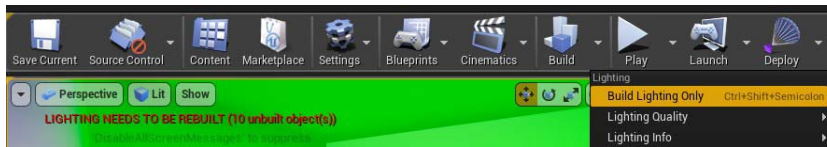
In case the floor level of the virtual scene is not in the zero height, the camera needs to be moved accordingly to the height of the floor (put exactly on the floor level).

- a. Create an export to adjust the Camera (TrackedPawn) Z position to match the actual floor level with the tracking.



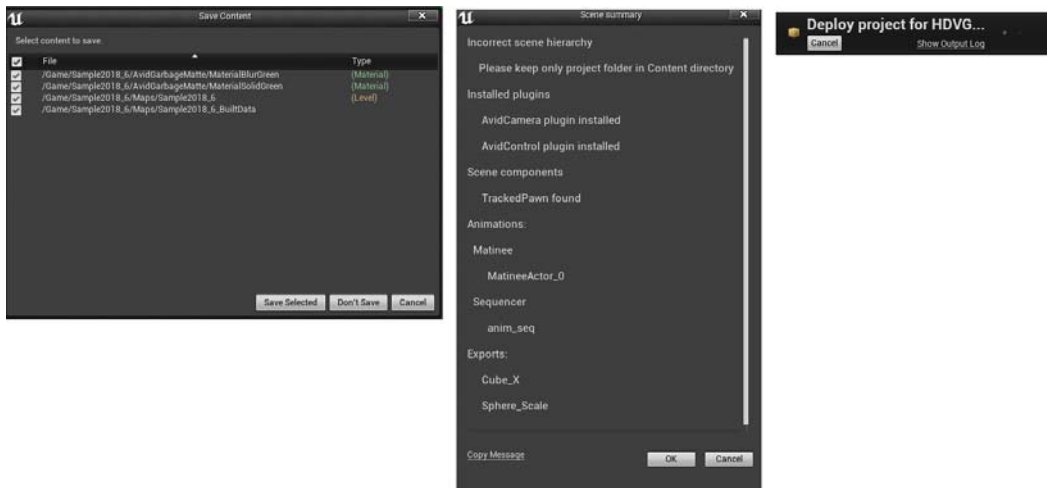
- b. Now, the export can be used in runtime by the controller.

15. Now you can rebuild the lighting in the map.



After rebuilding light you could see the following error(s): “Static mesh actor has NULL StaticMesh property. Ignore this message.”

16. After rebuilding the lighting, you are ready to deploy the scene. Click **Deploy**, save the entire content, confirm your selection in the summary window and wait till the scene is built.



17. If the deploy process is complete, then you should see the "/Sample2018\_6/Sample 2018\_6" directory in your Control Data.

 > This PC > Local Disk (G:) > Projects > Sample2018\_6 >

18. Now, the scene is prepared to be loaded from the controllers to the HDVG/Maestro | Engine 4K with the SceneLoader.



*First packaging of a new project for Linux takes approx. 1 hour on 4 cores i7. Consecutive packaging takes significantly less time.*



**If the catalog and scene with the same names exist, then the build process will overwrite them without prompting for a confirmation!**



## 5 Troubleshooting

In this section:

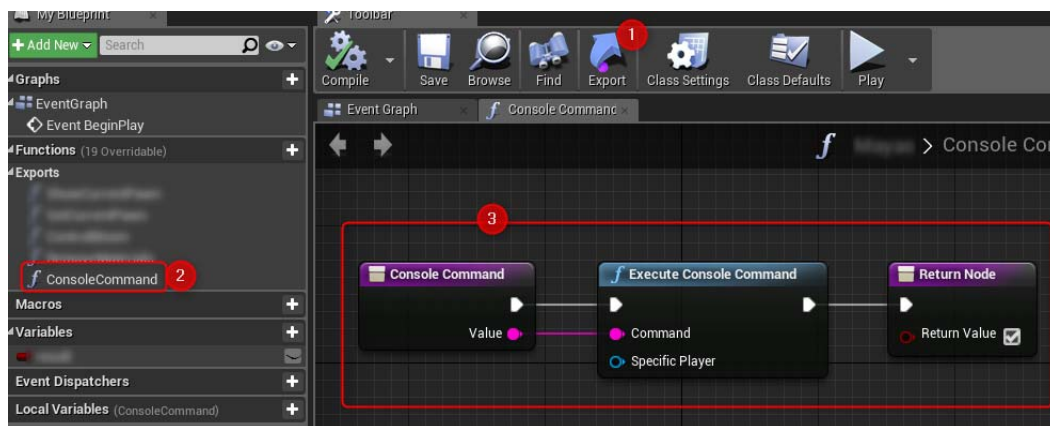
- [How to Improve the Performance of Maestro | Virtual Set](#)
- [Frequently Asked Questions](#)

### How to Improve the Performance of Maestro | Virtual Set

#### Built-in Diagnostic Tools

You can improve the performance of your Maestro | Virtual Set system by introducing a few changes in your working environment. Follow the steps below to increase the efficiency of your system.

1. Create a dedicated export on the scene which will give you access to the internal console of the Unreal Engine.



2. Deploy the scene to the HDVG4 machine.
3. Run the scene on the HDVG4 using a controller, such as Avid Maestro | Live.
4. Use the export manipulation inside the controller to execute the Console Command.

A full list of commands can be found in the Unreal Engine documentation (<https://docs.unrealengine.com/en-us/Engine/Performance/StatCommands>).

The most important commands are:

- stat FPS
- stat UnitGraph
- stat GPU
- stat Engine

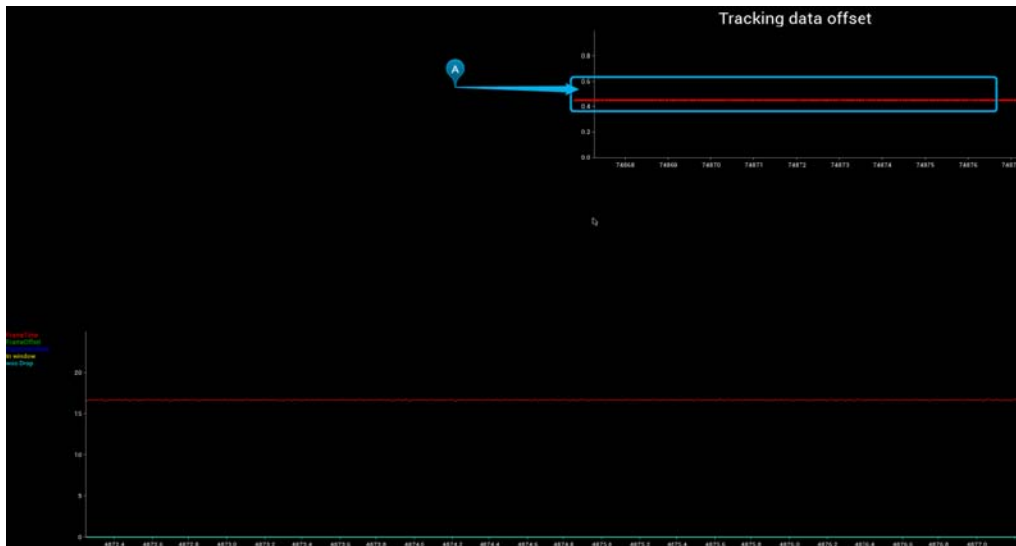
- stat Game

- Information on how to interpret this data and how to improve the overall performance of the Unreal Engine can be found in the Unreal Engine documentation (<https://docs.unrealengine.com/en-us/Engine/Performance>).

### Tracking Data Diagnostic Tool

- Using PuTTY, connect to the SceneLoader ReTalk3.
- Execute the following command:  

```
[1] Tracking.ShowOverlay(Value="On");
```
- It should activate the Tracking Data graph overlay.

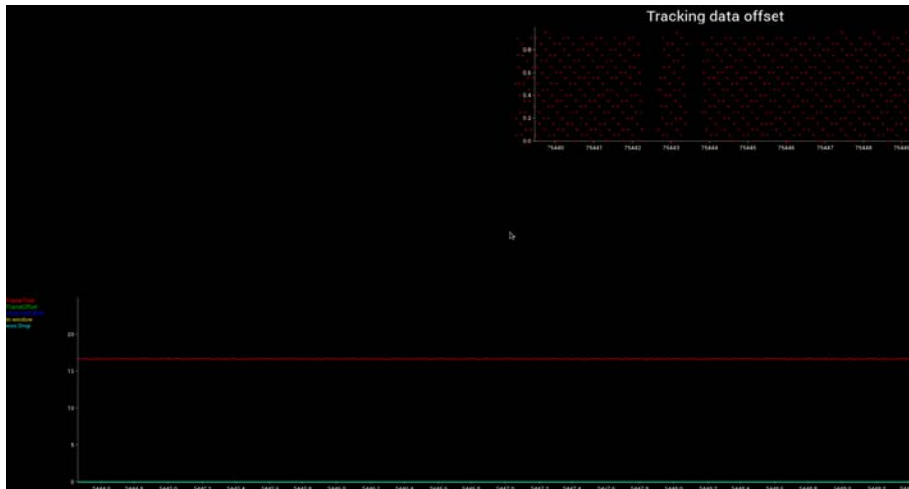


(A) Correct window frame from Tracking Data

- Adjust the SYNCHRO SOURCE DELAY parameter in the TrackingSet.cfg file to have data in the correct window frame (A).
- Disable overlay using a ReTalk3 command:  

```
[1] Tracking.ShowOverlay(Value="Off");
```

 *In case you encounter a problem with synchronization of TrackingSet where similar statistics are produced:*



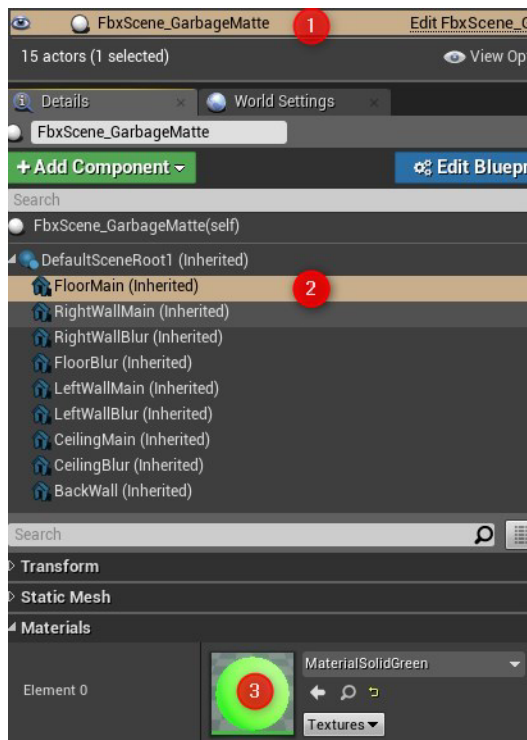
Adjust the *SYNCHRO SOURCE* parameter in the *TrackingSet.cfg* file to synchronize to the appropriate source.

## Frequently Asked Questions

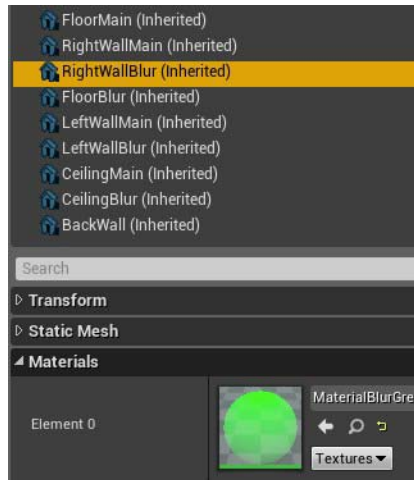
### Garbage Matte cast shadows. What can I do to disable them?

To disable casting shadows from garbage matte you need to edit two materials: *MaterialSolidGreen* and *MaterialBlurGreen*.

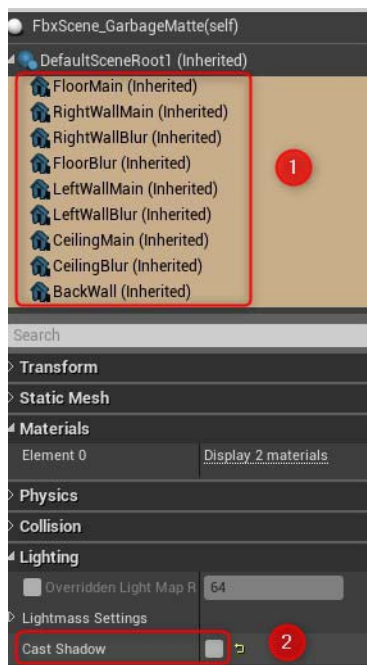
1. Select *FbxScene\_GarbageMatte* and double-click on Element 0 - *MaterialSolidGreen* to open the material editor.



2. Change Shading from Unlit to Default Lit.
3. Do the same for MaterialBlurGreen material (points 1 and 2).



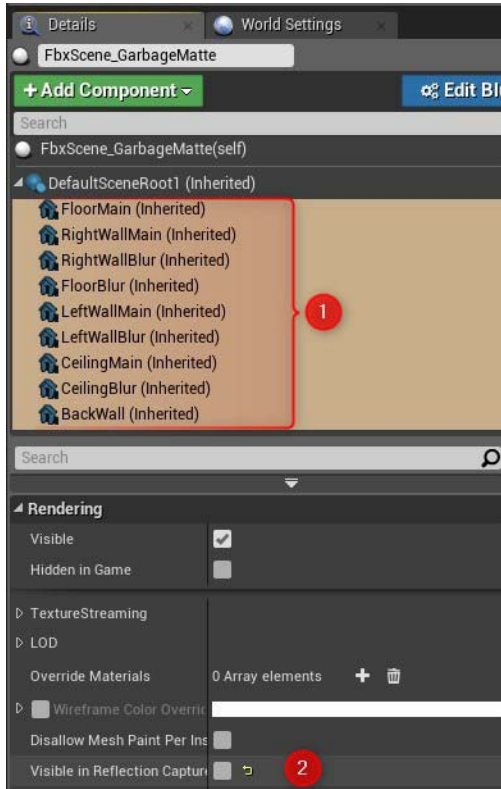
4. Select all elements from FbxScene\_GarbageMatte and disable the Cast Shadows option.



5. Rebuild the lights and deploy the scene.

**Garbage Matte is reflected on shiny surfaces. What can I do to eliminate this issue?**

You need to select all garbage objects and disable “Visible in Reflection Capture” option.



# A Guidelines for exporting from 3Ds Max® to Unreal

## Introduction

The scenes imported from 3Ds Max to the Unreal Editor are in the FBX format.

The Import Into Level command allows to import full FBX scenes into their levels instead of importing assets individually. Users are given full control of exactly which assets will be imported, as well as per-asset control over import settings. The workflow also allows for selective reimporting for individual assets that receive further edits outside of Unreal.

At this time, FBX full scene import supports the following asset types:

- Static Meshes
- Skeletal Meshes
- Animations
- Materials (Basic support; may not match original material in your content creation application)
- Textures
- Rigid Mesh
- Morph Targets
- Cameras (no animation)
- Lights

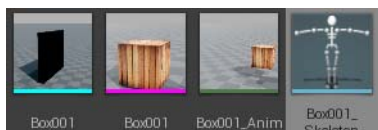
For more information, see “Import Into Level” (<https://docs.unrealengine.com/en-us/Engine/Content/FBX/FullScene>) in the **Unreal Editor Manual**.

For more information, see “FBX Import Options Reference” (<https://docs.unrealengine.com/en-US/Engine/Content/FBX/ImportOptions>) in the **Unreal Editor Manual**.

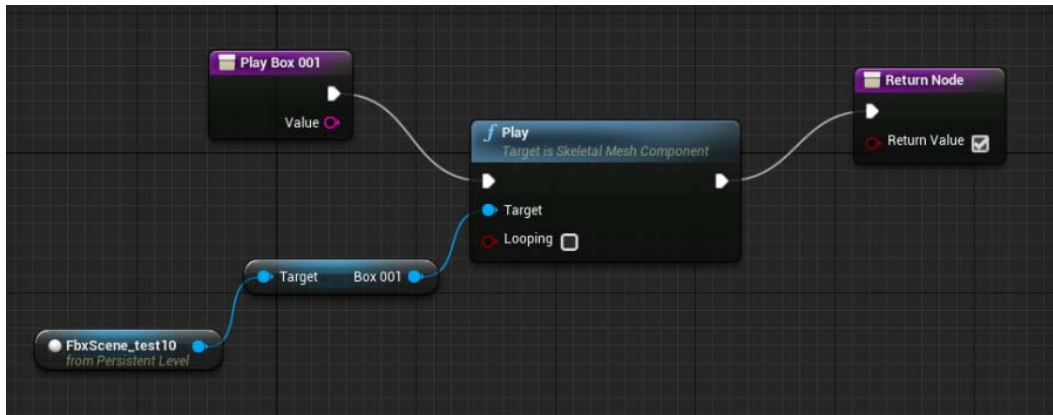
For more information, see “FBX Static Mesh Pipeline” (<https://docs.unrealengine.com/en-US/Engine/Content/FBX/StaticMeshes#ExportMesh>) in the **Unreal Editor Manual**.

## Animations

Keyframe animations created in 3Ds Max are converted into skeleton animations in the Unreal Editor



The Blueprint below shows how to prepare a simple trigger for an animation using export. In this simple case, sending any string using an export starts playing an animation.

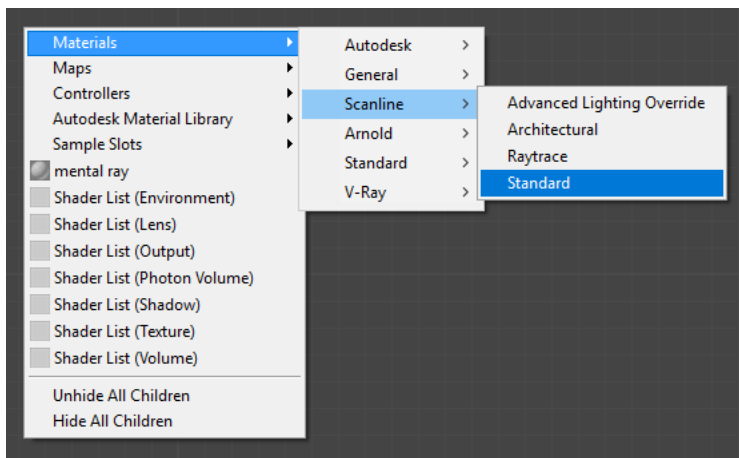


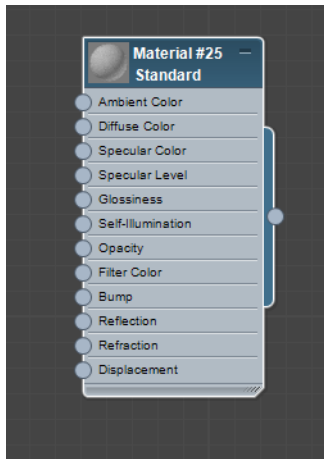
In a more complex case, depending on the incoming export, various actions may be taken, such as: stop, pause, play.

## Materials

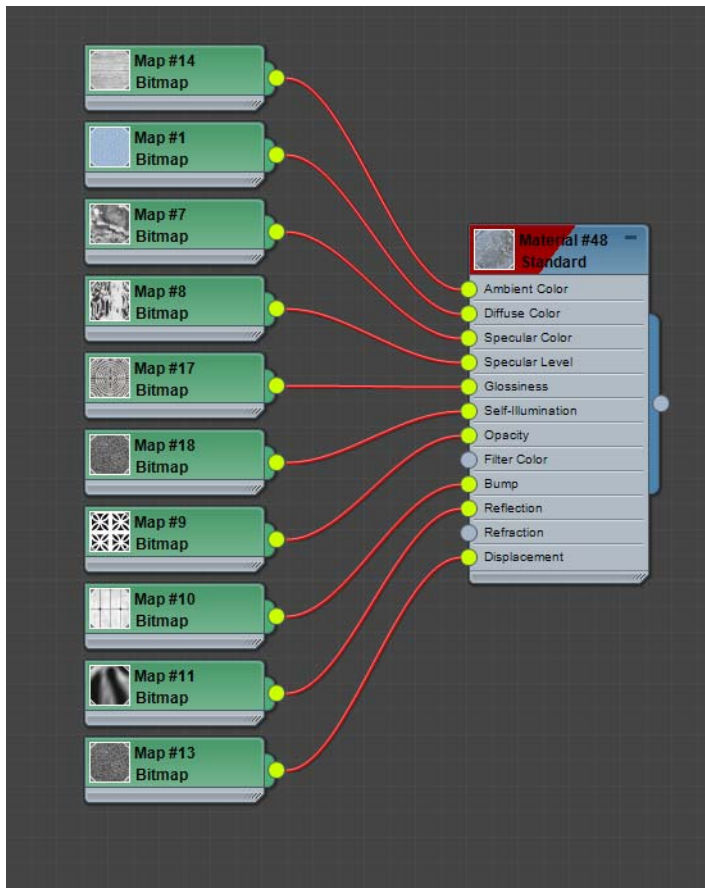
Preparing the stage requires knowledge of the conversion restrictions from the models used in 3Ds Max to the models used in Unreal.

Not all materials that can be used in 3DsMax can be converted to Unreal. The safest way to define material in 3Ds Max is to use a standard material.





In this case, the bitmap textures were pushed to most of the fields.

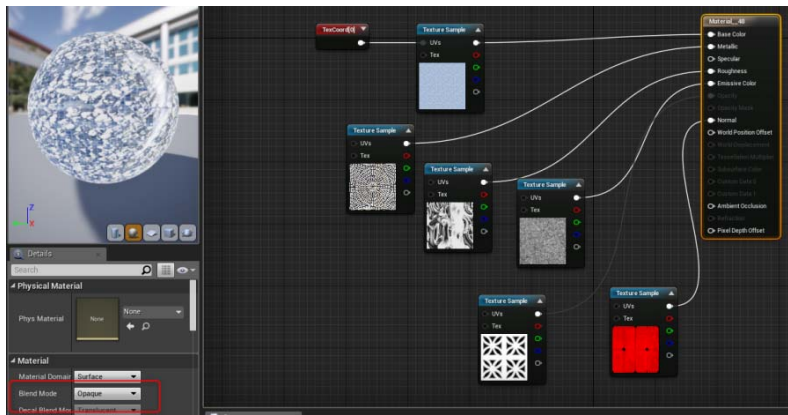


After conversion, the following set of connections is achieved.

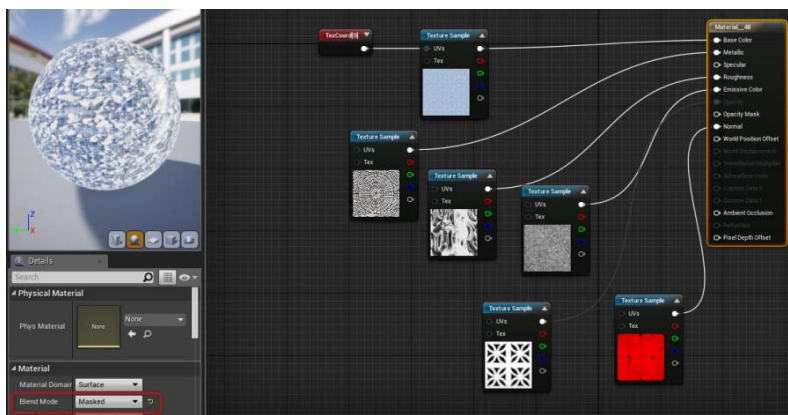
Depending on the selected “Blend Mode”, we will get various ways to represent the material after conversion.



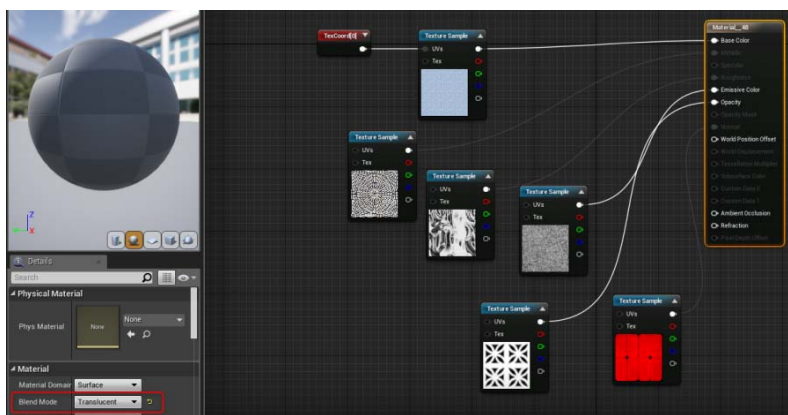
## Mode: Opaque



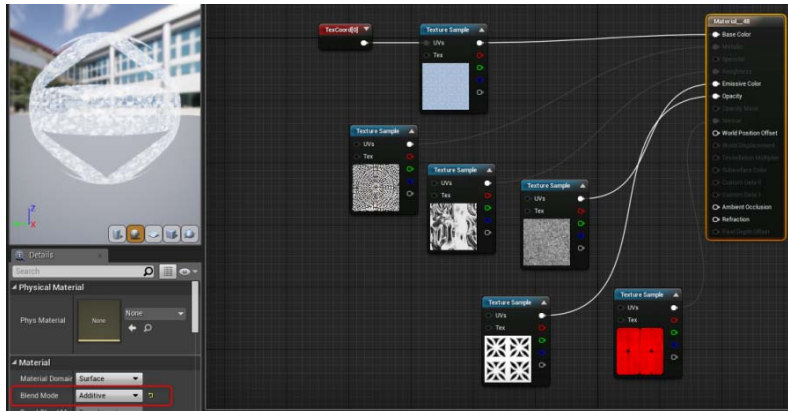
## Mode: Masked



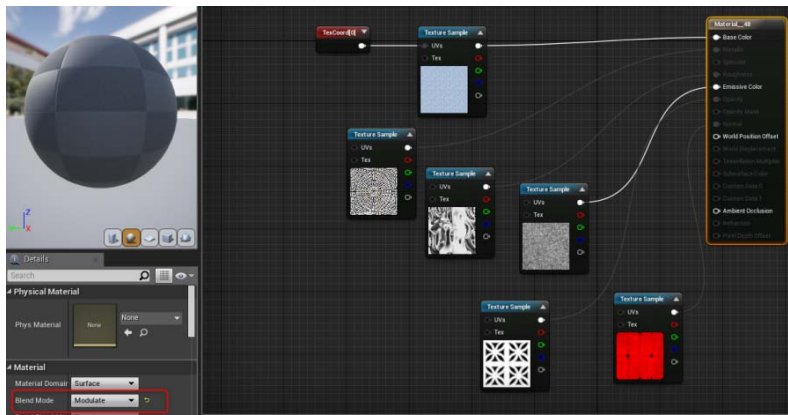
## Mode: Translucent



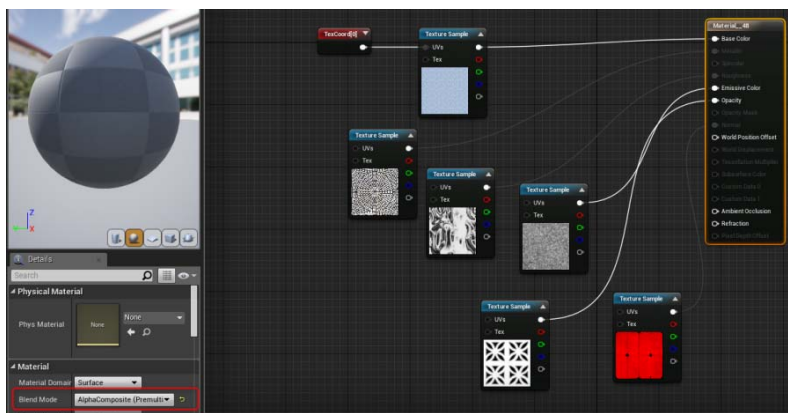
## Mode: Additive



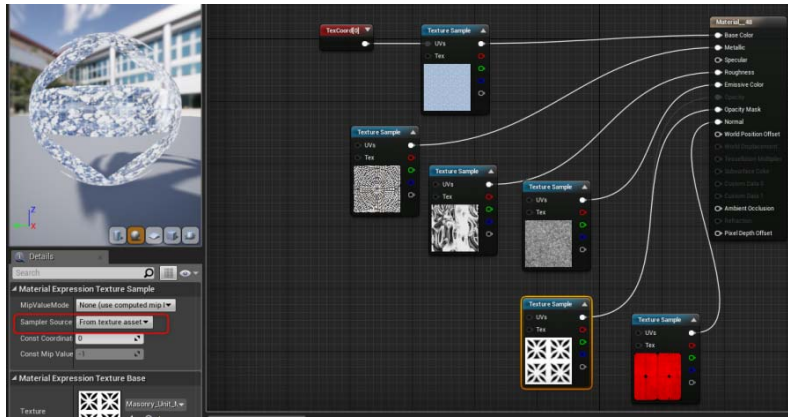
## Mode: Modulate



## Mode: Alpha Composite



## Mode: From Texture Asset

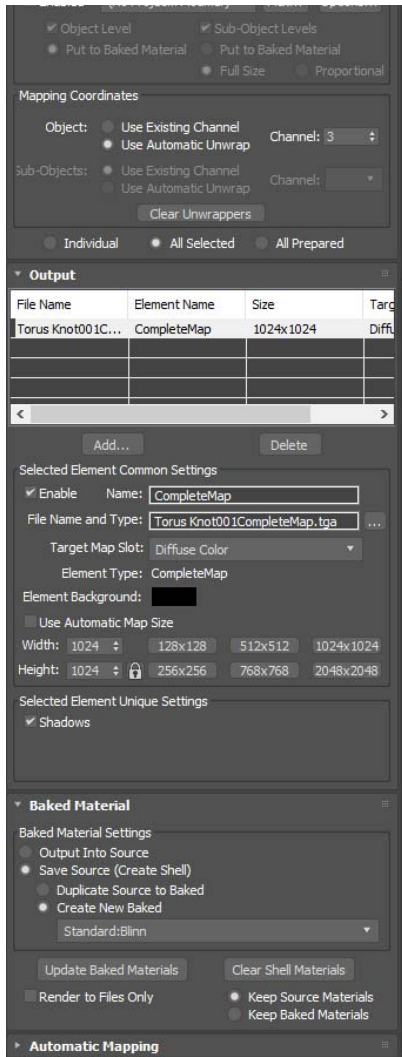


## Baked Texture Method

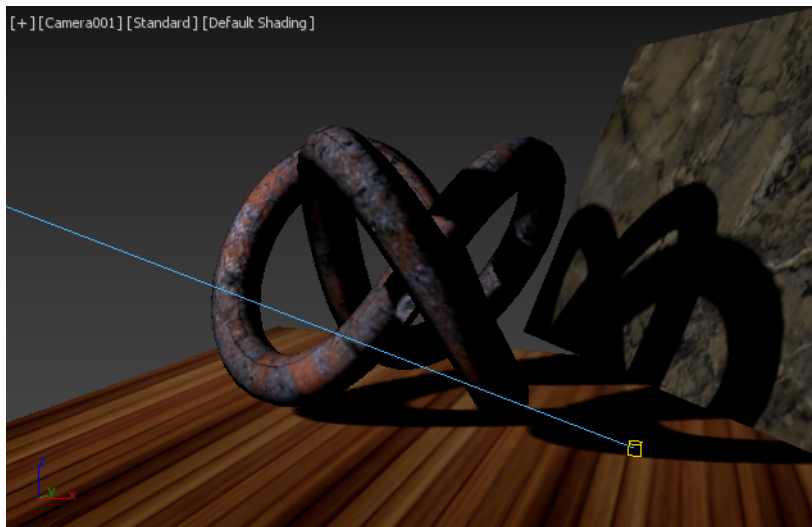
A common method is rendering to texture (known as “texture baking”). Its purpose is to make the created scene realistic by suturing the texture of the lighting existing in the scene. However, this method has some limitations. It can only be used if the objects do not move, and the lights do not change. It is also not applicable when objects are to be partially mirrored.

The process of baking textures from in 3Ds Max is well described in the 3ds Max Help. As a result, we receive the material as a standard model and, depending on the declared assignment method, appointment to the appropriate channel.

After being imported to the Unreal Editor, the result is an illuminated scene. Because the lighting is already sewn in the texture, you do not need to import the light separately.



Rendered Scene in 3Ds Max:



After importing to the Unreal Editor:

