

Арам Куксон
Райан Даулингсока
Клинтон Крамплер



Разработка игр на Unreal® Engine 4

за **24**
Часа

Обновлено до версии 4.21





Aram Cookson
Ryan DowlingSoka
Clinton Crumpler

Unreal® Engine 4 Game Development

in **24**
Hours

Арам Куксон
Райан Даулингсока
Клинтон Крамплер

Разработка игр на Unreal® Engine 4

за **24**
Часа

БОМБОРА™

Москва 2019

УДК 004.9
ББК 77.056с.я92
K89

Aram Cookson, Ryan Dowlingsoka, Clinton Crumpler
**UNREAL ENGINE 4 GAME DEVELOPMENT IN 24 HOURS,
SAMS TEACH YOURSELF**

Authorized translation from the English language edition, entitled UNREAL ENGINE 4 GAME DEVELOPMENT IN 24 HOURS, SAMS TEACH YOURSELF, 1st Edition by ARAM COOKSON; RYAN DOWLINGSOKA; CLINTON CRUMPLER, published by Pearson Education, Inc, publishing as Sams Publishing,
Copyright © 2016 by Pearson Education, Inc.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc.

RUSSIAN language edition published by Limited Company, Publishing House Eksmo,
Copyright © 2019.

Книга подготовлена в ходе проведения исследования в рамках Программы фундаментальных исследований Национального исследовательского университета «Высшая школа экономики» (НИУ ВШЭ) и с использованием средств субсидии в рамках государственной поддержки ведущих университетов Российской Федерации

Куксон, Арам.

K89 Разработка игр на Unreal Engine 4 за 24 часа / Арам Куксон, Райан Даулингсока, Клинтон Крамплер ; [перевод с английского М. А. Райтмана]. — Москва : Эксмо, 2019. — 528 с. : ил. — (Мировой компьютерный бестселлер. Геймдизайн).

ISBN 978-5-04-103162-6

Всего за 24 урока, каждый продолжительностью 1 час или меньше, вы узнаете, как начать проектировать великолепные игры с помощью движка Unreal Engine 4 под Windows, Mac, PS4, Xbox One, iOS, Android, Linux, Интернета или для всех сразу!

Пошаговый подход к обучению, представленный в книге, покажет, как работать с интерфейсом Unreal Engine 4, продемонстрирует рабочие процессы и самые мощные редакторы и инструменты движка. За считанные часы вы научитесь создавать эффекты, использовать приемы скриптинга, реализовывать физику и даже вести разработку для мобильных устройств и HUD-интерфейсов. Каждый урок дополняет знания, полученные вами в предыдущих, создавая крепкий фундамент для успешной работы с реальными задачами.

УДК 004.9
ББК 77.056с.я92

ISBN 978-5-04-103162-6

© Райтман М.А., перевод на русский язык, 2019
© Оформление. ООО «Издательство «Эксмо», 2019

Оглавление

Для кого эта книга	17
Структура книги	17
1-Й ЧАС. ВВЕДЕНИЕ В UNREAL ENGINE 4	23
Установка Unreal	24
Загрузка и установка лаунчера	24
Загрузка и установка Unreal Engine	25
Создание первого проекта	27
Панель Project Browser	27
Знакомство с интерфейсом	29
Строка меню	30
Панель Modes	31
Панель World Outliner	31
Панель Details	32
Панель Content Browser	33
Панель Viewport	34
Режимы просмотра и визуализаторы	37
Флажки Show	38
Навигация по сцене в перспективной проекции	38
Панель инструментов редактора уровней	39
Проигрывание уровня	40
Резюме	41
Вопросы и ответы	41
Семинар	41
Контрольные вопросы	41
Ответы	42
Упражнения	42
2-Й ЧАС. ИЗУЧЕНИЕ СИСТЕМЫ GAMEPLAY FRAMEWORK	43
Доступные ресурсы	43
Play in Editor (PIE)	45
Структура папок проекта	46
Связи ассетов и средство просмотра связей	52
Папка Saved	53
Система Gameplay Framework	53
Класс <i>GameMode</i>	54

Классы <i>Controller</i>	54
Классы <i>Pawn</i> и <i>Character</i>	55
Класс <i>HUD</i>	55
Резюме	58
Вопросы и ответы	58
Семинар	58
Контрольные вопросы	58
Ответы	59
Упражнения	59
3-Й ЧАС. КООРДИНАТЫ, ПРЕОБРАЗОВАНИЯ, ЕДИНИЦЫ ИЗМЕРЕНИЯ И ОРГАНИЗАЦИЯ	60
Понимание декартовой системы координат	60
Работа с трансформациями	62
Инструменты трансформации	62
Интерактивные и ручные трансформации	64
Мировые и локальные трансформации	65
Оценка единиц измерения и измерений	65
Единицы измерения сетки	66
Привязка сетки	67
Организация сцены	68
Панель World Outliner	68
Папки	69
Группировка	71
Слои	72
Связывание	73
Резюме	74
Вопросы и ответы	75
Семинар	75
Контрольные вопросы	75
Ответы	75
Упражнения	76
4-Й ЧАС. РАБОТА С АКТЕРАМИ СТАТИЧНЫХ МЕШЕЙ	77
Ассеты статичных мешей	77
Редактор статичных мешей	78
Как открыть окно редактора статичных мешей	79
Импорт статичных мешей	80
Просмотр UV-разверток	81
Назначение материала ассету статичного меша	83
Оболочки коллизий	83

Просмотр оболочек коллизий	83
Редактирование оболочек коллизий	84
Панель Convex Decomposition	87
Многополигональная коллизия	88
Актеры статичных мешей	89
Помещение актеров статичных мешей на уровень	90
Настройки мобильности	91
Изменение ссылки на меш для актера статичного меша	91
Смена материала актера статичного меша	92
Редактирование реакций на коллизии актера статичного меша	93
Резюме	96
Вопросы и ответы	96
Семинар	96
Контрольные вопросы	96
Ответы	97
Упражнения	97
5-Й ЧАС. ПРИМЕНЕНИЕ ОСВЕЩЕНИЯ И РЕНДЕРИНГА	99
Изучение терминологии освещения	99
Изучение типов источников света	100
Добавление точечных источников света	101
Добавление прожекторных источников света	102
Добавление небесных источников света	104
Добавление направленных источников света	105
Использование свойств освещения	107
Просчет освещения	108
Приложение Swarm Agent	108
Мобильность	110
Резюме	112
Вопросы и ответы	112
Семинар	113
Контрольные вопросы	113
Ответы	113
Упражнения	113
6-Й ЧАС. ИСПОЛЬЗОВАНИЕ МАТЕРИАЛОВ	115
Изучение материалов	115
Основанный на физике рендеринг (PBR, Physically Based Rendering)	116
Типы входных данных для материалов	117
Базовый цвет (альбедо)	117

Металлизированность	117
Шероховатость	118
Нормаль	118
Создание текстур	120
Размеры текстур	120
Степень двойки	120
Расширения файлов текстур	121
Импорт текстур	122
Создание материалов	123
Входные и выходные данные	125
Ноды значений	126
Экземпляры	128
Резюме	131
Вопросы и ответы	131
Семинар	133
Контрольные вопросы	133
Ответы	133
Упражнения	133
7-Й ЧАС. ИСПОЛЬЗОВАНИЕ ЭЛЕМЕНТОВ АУДИОСИСТЕМЫ	135
Введение в основы работы со звуком	135
Компоненты аудио	135
Импорт аудиофайлов	136
Использование звуковых актеров	138
Создание звуковых сигналов	141
Продвинутые звуковые сигналы	144
Настройка звука с использованием аудиопространств	145
Резюме	147
Вопросы и ответы	147
Семинар	147
Контрольные вопросы	147
Ответы	148
Упражнения	148
8-Й ЧАС. СОЗДАНИЕ ЛАНДШАФТОВ И РАСТИТЕЛЬНОСТИ	150
Работа с ландшафтами	150
Ландшафтные инструменты	151
Создание ландшафтов	152
Управление ландшафтами	154
Скульптурирование форм и объемов	155
Меню Tool	156

Меню Brush	156
Меню Falloff	157
Окрашивание	157
Материалы ландшафтов	157
Использование растительности	161
Размещение растительности	162
Резюме	164
Вопросы и ответы	164
Семинар	165
Контрольные вопросы	165
Ответы	165
Упражнение	165
9-Й ЧАС. СОЗДАНИЕ МИРА	167
Создание миров	168
Повествование через окружение	168
Анатомия уровня	169
Процесс создания мира	170
Создание масштаба	170
Создание пределов	171
Создание слоев и наброска	173
Помещение декораций и ассетов	176
Распространение света и звука	181
Игровое тестирование и отладка	183
Резюме	186
Вопросы и ответы	186
Семинар	187
Контрольные вопросы	187
Ответы	188
Упражнение	188
10-Й ЧАС. ЭФФЕКТЫ ВОСПРОИЗВОДСТВА В СИСТЕМАХ ЧАСТИЦ	190
Изучение частиц и типов данных	191
Работа с редактором Cascade	192
Использование эмиттеров и модулей	193
Использование редактора кривых	196
Использование основных модулей	198
Модуль Required	198
Модуль Spawn	199
Модуль Lifetime	200
Модули Initial Size и Size By Life	200

Модули Initial Color, Scale Color/Life и Color Over Life	201
Модули Initial Velocity, Inherit Parent Velocity и Const Acceleration	202
Модули Initial Location и Sphere	202
Модули Initial Rotation и Rotation Rate	203
Приисвоение материала частицам	203
Цвета частиц	204
SubUV-текстуры	204
Запуск систем частиц	207
Автоматическая активация	207
Активация систем частиц с помощью блупринтов уровней	207
Резюме	208
Вопросы и ответы	208
Семинар	209
Контрольные вопросы	209
Ответы	209
Упражнение	210
11-Й ЧАС. ИСПОЛЬЗОВАНИЕ АКТЕРОВ СКЕЛЕТНЫХ МЕШЕЙ	211
Определение скелетных мешей	211
Импорт скелетных мешей	216
Изучение редактора Persona	221
Режим Skeleton	222
Режим Mesh	223
Режим Animation	225
Режим Graph	228
Использование актеров скелетных мешей	230
Резюме	232
Вопросы и ответы	232
Семинар	233
Контрольные вопросы	233
Ответы	233
Упражнение	234
12-Й ЧАС. MATINEE И СИНЕМАТИКА	235
Актеры Matinee	235
Свойства актеров Matinee	236
Редактор Matinee	238
Панель Tracks	239
Установка продолжительности последовательности	239
Указатель воспроизведения	240

Группы	240
Треки	241
Редактор кривых	244
Режимы интерполяции	246
Работа с другими треками	248
Трек Sound	248
Работа с камерами Matinee	249
Группы и актеры камер	249
Группа Director	251
Работа с ассетами данных Matinee	253
Резюме	253
Вопросы и ответы	254
Семинар	255
Контрольные вопросы	255
Ответы	255
Упражнение	255
13-Й ЧАС. ИЗУЧЕНИЕ РАБОТЫ С ФИЗИКОЙ	257
Использование физики в UE4	257
Основные понятия физики	258
Назначение физического режима игры уровню	258
Настройки проекта и физики мира	259
Симуляция физики	261
Использование физических материалов	264
Создание ассета физического материала	265
Назначение физического материала актеру статичного меша	266
Назначение физического материала другому материау	267
Работа с ограничениями	269
Прикрепление актеров физики	269
Актеры ограничения физики	270
Использование актеров силы	274
Актеры физических движителей	274
Актеры радиальной силы	275
Резюме	276
Вопросы и ответы	276
Семинар	277
Контрольные вопросы	277
Ответы	278
Упражнение	278

14-Й ЧАС. ВВЕДЕНИЕ В СИСТЕМУ ВИЗУАЛЬНОГО ПРОГРАММИРОВАНИЯ БЛЮПРИНТОВ	280
Основы визуального программирования	281
Изучение редактора блюпринтов	282
Интерфейс редактора блюпринтов	283
Панель инструментов редактора блюпринтов	285
Панель My Blueprint	285
Панель Event Graph	285
Контекстное меню блюпринта	286
Ноды, провода и контакты	287
Фундаментальные понятия разработки скриптов	288
События	288
Функции	290
Переменные	293
Операторы и условия	297
Организация и комментирование скриптов	299
Резюме	301
Вопросы и ответы	301
Семинар	302
Контрольные вопросы	302
Ответы	303
Упражнение	303
15-Й ЧАС. РАБОТА С БЛЮПРИНТАМИ УРОВНЕЙ	306
Настройки коллизий актеров	308
Назначение актеров событиям	309
Назначение актеров ссылочным переменным	312
Компоненты актеров	312
Получение и присвоение значений свойств актеров	313
Цели функций	314
Свойство Activate	317
Функция Play Sound at Location	319
Использование актеров физики для активации событий	321
Резюме	322
Вопросы и ответы	323
Семинар	324
Контрольные вопросы	324
Ответы	324
Упражнение	324

16-Й ЧАС. РАБОТА С БЛЮПРИНТ-КЛАССАМИ	326
Использование блюпринт-классов	326
Добавление блюпринт-класса	327
Интерфейс редактора блюпринтов	328
Работа с компонентами	330
Добавление компонентов	330
Панель Viewport	331
Программирование блюпринтов с использованием компонентов	333
Работа с узлом Timeline	336
Треки и кривые Timeline	337
Скриптинг пульсирующего источника света	340
Наследование блюпринта из существующего класса	340
Резюме	347
Вопросы и ответы	347
Семинар	348
Контрольные вопросы	348
Ответы	348
Упражнение	349
17-Й ЧАС. ИСПОЛЬЗОВАНИЕ РЕДАКТИРУЕМЫХ ПЕРЕМЕННЫХ И СКРИПТА КОНСТРУИРОВАНИЯ	350
Подготовка	351
Создание редактируемых переменных	351
Использование скрипта конструирования	354
Добавление компонентов Arrow	355
Добавление компонентов Static Mesh	358
Ограничение редактируемых переменных	360
Свойство Show 3D Widget	361
Резюме	361
Вопросы и ответы	362
Семинар	362
Контрольные вопросы	362
Ответы	363
Упражнение	363
18-Й ЧАС. СОЗДАНИЕ АКТЕРОВ И ВВОДИМЫХ С КЛАВИАТУРЫ СОБЫТИЙ	364
Почему спаунинг важен	364
Создание блюпринт-класса для спауна	365
Использование скрипта конструирования	366

Использование свойства переменной Expose on Spawn	367
Создание запускающегося блупринта	369
Спаунинг актера из класса	371
Резюме	374
Вопросы и ответы	375
Семинар	375
Контрольные вопросы	375
Ответы	376
Упражнение	376
19-Й ЧАС. СОЗДАНИЕ ЭКШЕН-СТОЛКНОВЕНИЯ	379
Игровые режимы проекта	379
HUD-интерфейсы	380
Игровой таймер и система респауна	380
Изучение умений персонажей	380
Использование блупринт-классов	382
Папка BP_Common	383
Папка BP_Turrets	386
Папка BP_Respawn	387
Папка BP_Pickup	387
Папка BP_Levers	388
Теги актеров и компонентов	389
Резюме	390
Вопросы и ответы	390
Семинар	391
Контрольные вопросы	391
Ответы	391
Упражнение	391
20-Й ЧАС. СОЗДАНИЕ АРКАДНОГО ШУТЕРА: СИСТЕМЫ ВВОДА И АВАТАРЫ	393
Определение требований с помощью сводки проекта	394
Определение требований	394
Создание игрового проекта	394
Создание пользовательского режима игры	397
Создание пользовательского аватара и контроллера игрока	399
Наследование от класса DefaultPawn	400
Управление движением объекта Pawn	403
Отключение движения по умолчанию	405

Настройка входных воздействий и маппинг осей	406
Использование событий ввода для перемещения аватара	408
Настройка фиксированной камеры	410
Резюме	412
Вопросы и ответы	412
Семинар	413
Контрольные вопросы	413
Ответы	414
Упражнение	414
21-Й ЧАС. СОЗДАНИЕ АРКАДНОГО ШУТЕРА: ПРЕПЯТСТВИЯ И БОНУСЫ	415
Создание базового класса препятствия	416
Заставляем препятствие двигаться	419
Получение урона аватаром	423
Использование нода Cast To	425
Перезапуск игры в случае гибели персонажа	427
Создание бонуса здоровья	430
Спаун актеров	436
Удаление старых препятствий	441
Резюме	442
Вопросы и ответы	442
Семинар	443
Контрольные вопросы	443
Ответы	444
Упражнение	444
22-Й ЧАС. РАБОТА С UMG	446
Создание виджет-блюпринта	446
Навигация по интерфейсу UMG	447
Режим конструктора	447
Режим Graph	448
Настройка разрешения	449
Якорные точки и масштабирование DPI	452
Создание стартового меню	453
Импорт ассетов	453
Размещение виджетов на холсте	455
Создаем функциональность через скрипты	458
Пример меню	464

Резюме	465
Вопросы и ответы	465
Семинар	466
Контрольные вопросы	466
Ответы	466
Упражнение	467
23-Й ЧАС. СОЗДАНИЕ ИСПОЛНЯЕМОГО ФАЙЛА	468
«Приготовление» контента	469
Упаковка проекта для Windows	470
Ресурсы для упаковки под Android и iOS	475
Доступ к профессиональным настройкам упаковки	476
Резюме	477
Вопросы и ответы	477
Семинар	478
Контрольные вопросы	478
Ответы	478
Упражнение	479
24-Й ЧАС. РАБОТА С МОБИЛЬНЫМИ УСТРОЙСТВАМИ	480
Разработка приложений для мобильных устройств	481
Предварительный просмотр для мобильных устройств	482
Оптимизация для мобильных устройств	485
Установка платформы развертывания в редакторе	490
Сенсорные устройства	493
Виртуальные джойстики	493
События сенсора	495
Использование данных о движении устройства	498
Резюме	501
Вопросы и ответы	502
Семинар	502
Контрольные вопросы	502
Ответы	503
Упражнение	503

Предисловие

Unreal Engine 4 — мощный игровой движок, используемый многими профессиональными и инди-разработчиками игр. При первом использовании инструмента подобного Unreal Engine трудно разобраться с чего начать. Эта книга дает отправную точку, представляя интерфейс, рабочие процессы и множество редакторов и инструментов, которые предлагает Unreal Engine 4. Книга заложит прочный фундамент, на основе которого вы сможете удовлетворить возникший интерес к исследованию Unreal Engine и разработке игр. Каждая глава нацелена на быстрое погружение в ключевые области.

Для кого эта книга

Если вы хотите научиться создавать игры, приложения или интерактивные приключения, но не знаете, с чего начать, эта книга и движок Unreal Engine к вашим услугам. Эта книга для тех, кто заинтересован в понимании основ Unreal Engine. Неважно, являетесь ли вы новичком в разработке игр, или разрабатываете игры в качестве хобби, или получаете образование, чтобы стать профессиональным разработчиком, вы найдете что-то полезное на страницах этой книги.

Структура книги

В соответствии с подходом *Освой самостоятельно* эта книга состоит из 24 глав, работа с каждой занимает приблизительно один час.

- ▶ **1-й час «Введение в Unreal Engine 4».** Продемонстрировано, как загрузить и установить Unreal Engine 4, а также представлен интерфейс редактора.
- ▶ **2-й час «Изучение системы Gameplay Framework».** Этот час знакомит с Gameplay Framework, ключевым компонентом любого проекта, созданного в UE4.
- ▶ **3-й час «Координаты, преобразования, единицы измерения и организация».** Этот час поможет вам понять, как работают в UE4 измерение, контроль и организационные системы.
- ▶ **4-й час «Работа с актерами статичных мешей».** Вы узнаете, как импортировать 3D-модели и как использовать редактор статичных мешей.

- ▶ **5-й час «Применение освещения и отображения».** Вы узнаете, как разместить и настроить освещение на уровне.
- ▶ **6-й час «Использование материалов».** Этот час научит вас использовать текстуры и материалы в UE4.
- ▶ **7-й час «Использование элементов аудиосистемы».** Вы научитесь импортировать аудиофайлы, создавать ассеты звуковых сигналов и помещать на уровень актеры звуков окружения.
- ▶ **8-й час «Создание ландшафтов и растительности».** Вы научитесь работе с системой ландшафтов UE4 для создания ваших собственных ландшафтов, а также работе с системой растительности.
- ▶ **9-й час «Сборка мира».** Здесь вы примените полученные в предыдущих главах знания и создадите уровень.
- ▶ **10-й час «Эффекты воспроизведения в системах частиц».** Вы изучите базовые средства управления системы Cascade, которую можно использовать для воспроизведения эффектов динамических частиц.
- ▶ **11-й час «Использование актеров скелетных мешей».** Вы узнаете о редакторе Persona Editor и изучите различные типы ассетов, необходимые для оживления персонажей и существ.
- ▶ **12-й час «Matinee и синематика».** Вы научитесь пользоваться редактором Matinee Editor и анимировать камеры и меши.
- ▶ **13-й час «Обучение работе с физикой».** Вы научитесь заставлять актеров симулировать физику, чтобы реагировать на окружающий мир, а также ограничивать их.
- ▶ **14-й час «Введение в систему Blueprint Visual Scripting».** Вы ознакомитесь с основными понятиями создания скриптов и научитесь использовать Level Blueprint Editor.
- ▶ **15-й час «Работа с блупринтами уровней».** Вы узнаете о последовательностях событий блупринтов и создадите событие коллизии, реагирующее на действия игрока.
- ▶ **16-й час «Работа с блупринт-классами».** Вы узнаете, как создать блупринт-класс, использовать временную шкалу и создадите простой актер Pickup.
- ▶ **17-й час «Использование редактируемых переменных и Construction Script».** Вы научитесь использовать Construction Script и редактируемые переменные, чтобы создавать модифицируемых актеров.

- ▶ **18-й час «Создание вводимых клавиатурой событий и порождение актеров».** Вы научитесь создавать вводимые клавиатурой события, порождающие актеры в процессе геймплея*.
- ▶ **19-й час «Создание экшен-игры».** За этот урок вы используете существующий режим игры и блюпринт-классы, чтобы разработать и создать собственную экшен-игру от первого или третьего лица, базирующуюся на преодолении преград.
- ▶ **20-й час «Создание аркадного шутера: система ввода и аватар».** Вы начинаете работать над космическим шутером** в стиле игровых автоматов 1990-х. Вы узнаете о системе ввода и контролируемого пользователем актерах — аватарами.
- ▶ **21-й час «Создание аркадного шутера: преграды и захваты».** Вы продолжите работу над аркадным шутером, создавая преграды в виде астероидов и захваты здоровья, а также научитесь использовать наследование блюпринт-классов.
- ▶ **22-й час «Работа с UMG».** Вы научитесь использовать дизайнер пользователя интерфейса Unreal Motion Graphics и создадите стартовое меню.
- ▶ **23-й час «Создание исполняемого файла».** Вы узнаете короткий способ подготовки проекта к развертыванию на других устройствах.
- ▶ **24-й час «Работа с мобильными устройствами».** Вы изучите рекомендации по оптимизации и методы работы с мобильными устройствами, а также некоторые простые способы использования датчиков прикосновения и движения.

Мы надеемся, вы получите удовольствие от этой книги и извлечете для себя пользу. Удачи в путешествии по игровому движку UE4!

Сопутствующие файлы: Файлы проектов можно скачать по адресу:

http://addons.eksмо.ru/it/UE_24.zip

* Игровой процесс (gameplay) — интерактивное взаимодействие игрока с игрой. — *Прим. ред.*

** Шутер (shooter) — «стрелялка», разновидность компьютерных игр. — *Прим. ред.*

Об авторах

Эрэм Куксон — профессор кафедры интерактивного дизайна и разработки игр (ITGM, Interactive Design and Game Development) Колледжа искусств и дизайна Саванны (SCAD, Savannah College of Art and Design). Имеет степень бакалавра искусств скульптуры и магистра компьютерных искусств. После получения степени магистра искусств продолжал оказывать помощь в запуске программы ITGM и девять лет работал координатором выпускников. В течение последних пятнадцати лет Эрэм проектировал и вел очные и онлайн курсы по искусству и разработке игр с применением технологий Unreal Engine.

Райан Даулингсока — технический художник, работающий над франшизой Gears of War компании The Coalition, дочерней компании Microsoft Studio, в Ванкувере, Британская Колумбия. Он работает, прежде всего, над деталями содержимого для команды, системами воспроизведения разрушаемости, над растительностью, визуальными эффектами, постпроцессами и пользовательскими интерфейсами в Unreal Engine 4. Ранее работал в Microsoft, разрабатывая события для очков дополненной реальности Microsoft HoloLens на движке Unity5. Райан — эксперт во множестве пакетов создания развлекательного программного обеспечения, включая Maya, Houdini, Substance Designer, Photoshop, Nuke и After Effects. Он является бакалавром искусств визуальных эффектов Колледжа искусств и дизайна Саванны. Имея страсть к интерактивному строительству, корни которого уходят к консольным ролевым играм 1990-х (*Baldur's Gate II* и *Planescape: Torment*), Райан фокусируется на применении интерактивных технических решений для серьезных задач в современном гейминге. В свободное от работы над видеоиграми время Райана можно найти танцующим свинг вечера напролет со своей женой.

Клинтон Крамплер в настоящее время является главным художником окружающей среды в компании The Coalition (Microsoft Studio) в Ванкувере, Британская Колумбия. Ранее был художником в компании Battlecry Studios (Bethesda), KIXEYE, Army Game Studio, а также в других независимых студиях. Клинтон специализируется на создании окружающей среды, разработке шейдеров и художественных направлениях. Он выпустил множество обучающих видео в сотрудничестве с Digital Tutors, посвященных искусству разработки игр на движке Unreal Engine. Он получил степень магистра искусств интерактивного и игрового дизайна и бакалавра анимации в Колледже искусств и дизайна Саванны, Джорджия. До поступления в SCAD он получил степень бакалавра графического дизайна в Лонгвудском Университете в Фармвилле, Вирджиния. Найти больше информации о Клинтоне, а также взглянуть на его цифровые работы вы можете на сайте www.clintoncrumpler.com.

Триша, Найя и Элли, я люблю вас.

Эрэм

Дедушке Бобу: спасибо за постоянную поддержку в течение моего обучения и карьеры. Без твоего вклада в мое будущее я не добился бы того, что имею сейчас. Я всегда буду тебе благодарен.

Райан

Аманде: спасибо за то, что провела меня через пустыню, пока я писал эту книгу.

Клинтон

Благодарности

Моей семье: спасибо за понимание и терпение и за то, что дали мне время спрашаться.

Маме и папе: спасибо, что купили мне мой первый компьютер (TRS-80).

Льюису: спасибо, что думали обо мне. Вы были удивительным заведующим кафедрой.

Лоре, Шери, Оливии и всем рецензентам: спасибо за все ваши усилия.

Студии Epic Games: спасибо за то, что разработали и продолжаете разрабатывать столь поразительную технологию и игры.

Эрэм

Большое спасибо Саманте за терпимость и примирение с тем, что все мои выходные полностью проходили за клавиатурой. Твое терпение и поддержка в этом процессе были неоценимы.

Райан

Большое спасибо моему другу Брайану за то что, всегда помогал мне улучшить писательские навыки, редактировал мои работы и повышал мою уверенность братской поддержкой.

Спасибо Аманде и ее семье за то, что поддерживали меня, когда я писал эту книгу во время нашей поездки через всю страну. Я ценю ваши понимание и помощь.

Клинтон

Ждем ваших отзывов!

Как читатель этой книги, *вы* самый главный критик и комментатор. Мы ценим ваше мнение и хотим знать, что мы делаем правильно, что можем сделать лучше, какие области стоит осветить и любые другие мудрые слова, которые вы хотели бы нам передать.

Мы рады вашим комментариям. Вы можете отправить нам письмо на электронный адрес bombora@eksmo.ru, чтобы дать нам знать, что вам понравилось или не понравилось в этой книге, а также, что мы можем сделать, чтобы улучшить наши книги.

Пожалуйста, обратите внимание, что мы не можем помочь вам с техническими проблемами, связанными с темой данной книги.

1-Й ЧАС

Введение в Unreal Engine 4

Что вы узнаете в этом часе

- ▶ Установка Epic Games Launcher
- ▶ Установка Unreal Engine
- ▶ Создание нового проекта
- ▶ Использование интерфейса редактора Unreal Engine

Добро пожаловать в Unreal Engine! Unreal Engine 4 (UE4) — это игровой движок и редактор, разработанный компанией Epic Games для создания игр и приложений от мультиплатформенных игр с миллионными бюджетами до мобильных инди-разработок. Unreal Engine работает на операционных системах Windows и macOS и разработанные в нем проекты могут быть развернуты на платформах Windows, Mac, PlayStation 4, Xbox One, iOS, Android, HTML5 и Linux. В самой простой форме Unreal Engine 4 является коллекцией редакторов, используемых в различных стадиях производства игр или приложений.

За первый час вы научитесь загружать и устанавливать Unreal Engine, создадите свой первый проект и познакомитесь с интерфейсом редактора. Сначала вы создадите аккаунт пользователя и загрузите и установите Epic Games Launcher. После этого вы загрузите UE4. Затем вы создадите свой первый проект, изучите интерфейс редактора, научитесь передвигаться по уровню и запускать установочную карту без сборки игры.

ПРИМЕЧАНИЕ

Движок Unreal Engine бесплатен!

Именно так, движок UE4 является полностью бесплатным для использования! Вы имеете доступ ко всем возможностям — бесплатно! Почему компания Epic Games сделала его таким? Никогда не знаешь, кто создаст следующую отличную игру или приложение. Как они это сделали? Когда вы выпустите игру и начнете зарабатывать на ней деньги, то будете платить Epic Games 5% лицензионных отчислений. Подробности можно найти на сайте Epic Games. У Epic также есть магазин, в котором вы можете

приобрести и загрузить контент* для своих проектов. Вы не обязаны этого делать, поскольку, разумеется, можете сделать все что угодно с нуля, но отказ от повторного изобретения колеса может ускорить производство.

Установка Unreal

Установка Unreal Engine — это простой трехэтапный процесс.

1. Создайте новый аккаунт (учетную запись Epic).
2. Загрузите и установите Epic Games Launcher.
3. Загрузите Unreal Engine.

Загрузка и установка лаунчера

Лаунчер (Launcher) поможет отслеживать версии Unreal Engine, установленные на ваш компьютер. Он позволит управлять проектами, получать доступ к бесплатным тестовым проектам, а также к магазину, в котором вы можете покупать контент для использования в своих проектах. Лаунчер также содержит новости сообщества и ссылки на обучающие онлайн-ресурсы и документацию.

ПРИМЕЧАНИЕ

Поддерживаемые операционные системы и системные требования

Для эффективного использования Unreal Engine необходим персональный компьютер Windows или компьютер Macintosh, удовлетворяющий следующим критериям:

- операционная система: Windows 7 64-бит или macOS 10.9.2 или версии старше
 - процессор: Quad-core Intel или AMD, 2.5 GHz или быстрее
 - графическая карта: NVIDIA GeForce 470 GTX или AMD Radeon 6870 серии или выше с поддержкой DX11
 - ОЗУ: 8 ГБ
-

Выполните следующие шаги, чтобы загрузить и установить Epic Games Launcher.

1. Перейдите на сайт Unreal Engine (www.unrealengine.com), как показано на рис. 1.1.
2. Щелкните по кнопке **Get Started Now**.
3. Создайте новый аккаунт.

* Контент (content) — цифровое содержимое, продукт или компонент; набор информации. — Прим. ред.

4. Выберите установку для Windows или macOS в зависимости от вашей операционной системы и загрузите установочный файл в папку загрузок (.msi для Windows или .dmg для macOS).
5. Запустите инсталлятор, выберите место для установки и следуйте подсказкам на экране.

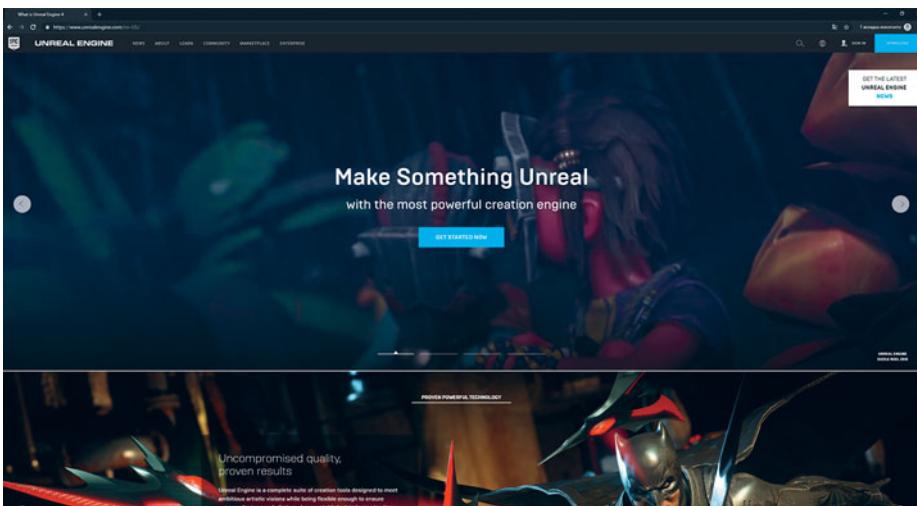


Рис. 1.1. Сайт Unreal Engine

СОВЕТ

Пространство на жестком диске

Игровые движки занимают большой объем на жестком диске. Когда вы установите новую версию движка UE4, он автоматически установится в ту же папку, что и лаунчер. Выберите место для установки лаунчера, объемом по меньшей мере 20 Гб свободного места. Изначально установка не требует такого количества свободного места, но по мере загрузки тестового контента или приобретения ассетов в магазине, необходимость в свободном пространстве увеличится. К счастью, когда вы создаете собственные проекты, то можете сохранять их в любой папке.

Загрузка и установка Unreal Engine

После установки лаунчера вы сможете загрузить и установить с его помощью Unreal Engine 4. UE4 гораздо больше, чем лаунчер, и его загрузка может занять некоторое время. Компания Epic постоянно совершенствует программное обеспечение, в результате чего существует множество версий UE4. Новичку лучше всего установить свежую официальную версию.

ПРЕДУПРЕЖДЕНИЕ

Версии для предпросмотра

Компания Epic, как правило, выпускает предварительный вариант следующей версии, которая тестируется на наличие сбоев. Когда вы только начинаете знакомиться с UE4, лучше загрузить самую последнюю версию, не являющуюся версией для предпросмотра. Вы всегда сможете перевести проекты на более новую версию, когда она станет доступна.

Выполните следующие шаги, чтобы загрузить и установить UE4 с помощью Epic Games Launcher.

1. Откройте лаунчер и щелкните по вкладке **Unreal Engine**.
2. Перейдите на горизонтальную вкладку **Library** (Библиотека).
3. В разделе **Unreal Engine version** (Версии Unreal Engine) щелкните мышью по кнопке **Add Version** (Установить версию), чтобы добавить новую версию в слот (см. рис. 1.2).
4. В только что созданном слоте щелкните по стрелке с выпадающим списком и выберите нужную версию.
5. Щелкните по кнопке **Install** (Установить), и лаунчер загрузит и установит Unreal Engine. Двигок UE4 имеет большой объем, поэтому загрузка отнимет некоторое время.

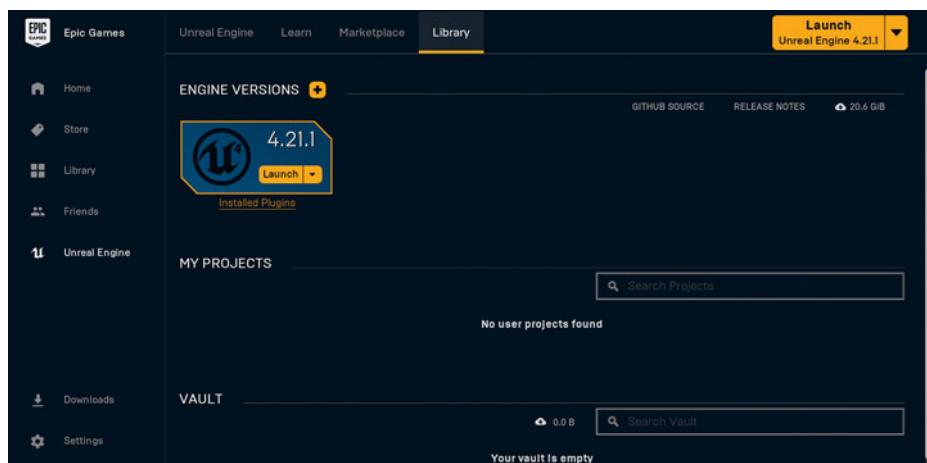


Рис. 1.2. Вкладка Library в лаунчере

Создание первого проекта

Как только движок UE4 будет установлен, настанет время создавать ваш первый проект. После первоначальной установки UE4, открывается Браузер проектов — **Project Browser**. На вкладке **Projects** в Браузере отображаются все проекты, над которыми вы в данное время работаете, а вкладка **New Project** позволяет создавать новые проекты на основе существующих основных шаблонов игр.

См. 2-й час, «Изучение системы Gameplay Framework», для получения более полной информации об анатомии проекта.

Панель Project Browser

Когда вы начнете новый проект в Браузере проектов (**Project Browser**), вам понадобится сделать выбор. Во-первых, вы можете создать проект на основе блюпринтов (blueprints) либо на основе языка C++. В этой книге раскрывается только работа с проектами на основе блюпринтов. Также вам понадобится выбрать целевые аппаратные средства: компьютер/консоль либо мобильный телефон / планшет. Кроме того, вы можете выбрать максимальное качество или масштабируемое разрешение трехмерной или двухмерной графики. Эти варианты меняют настройки проекта по умолчанию для разработки контента. Наконец, необходимо решить, хотите ли вы включить в ваш проект стартовый контент, который предлагает UE4. Если вы выберете вариант **With Starter Content**, у вас появятся ассеты (assets), на основе которых можно создать свой проект.

ПРИМЕЧАНИЕ

Проекты на основе блюпринтов или C++

Блюпринт (Blueprint) — это визуальная скриптовая среда, используемая для описания функционала игрового проекта. Проекты, основанные на C++, позволяют пользователям программировать функционал путем традиционного написания кода, не исключая при этом использования блюпринтов.

При работе с проектом, основанным на C++, потребуется установить компилятор, например, встроенный в IDE Visual Studio. Если вы никогда ранее не программировали или не использовали UE4, рекомендуем для начала ознакомиться с редактором и рабочими процессами, прежде чем приступать к работе с проектами, основанными на C++.

ПРЕДУПРЕЖДЕНИЕ

Размер проекта

Только созданный проект имеет небольшой размер, но размеры файлов проекта растут очень быстро, в зависимости от объемов контента и его качества, например

детализации моделей и размеров текстур. UE4 производит автосохранения и создает резервные копии файлов во время работы. Несмотря на то что эти файлы занимают пространство на диске, они очень полезны, когда что-то идет не так.

ПОПРОБУЙТЕ САМИ

Создайте проект

Вы можете создать проект в любом месте диска, где есть свободное пространство. Выберите место на диске объемом по меньшей мере 2 Гб, где проект будет доступен. На рис. 1.3 показаны базовые настройки вашего первого проекта. Выполните следующие шаги.

1. В лаунчере щелкните по кнопке **Launch** (Запустить).
2. Выберите вкладку **New Project**.
3. Выберите вкладку **Blueprint**.
4. Выберите шаблон **First Person**.
5. Выберите вариант **Desktop/Console** (целевые аппаратные средства).
6. Выберите вариант **Scalable 3D or 2D** (качество графики).
7. Выберите вариант **With Starter Content**.
8. Выберите раздел диска, на котором свободно не менее 2 Гб пространства.
9. Дайте проекту название.
10. Щелкните по кнопке **Create Project**.



Рис. 1.3. Настройки вашего первого проекта

ПРИМЕЧАНИЕ

Изменение настроек проекта

Вы можете изменить настройки проекта, такие как целевые аппаратные средства или качество графики, после создания проекта. Это можно сделать, выбрав пункт **Project Settings** в выпадающем списке вкладки **Edit** главного меню.

Знакомство с интерфейсом

Теперь, когда вы установили UE4 и создали свой первый проект, давайте приступим к практике. Как было сказано ранее, UE4 представляет собой коллекцию редакторов и инструментов для различных этапов производства игр. На данный момент вы должны сосредоточиться на изучении ключевых областей основного интерфейса и на способах перемещения по уровню. Главный интерфейс, который называется редактором уровней (**Level Editor**), как правило, используется для создания мира и уровней, а также для размещения ассетов.

Главный интерфейс редактора имеет семь ключевых панелей, которые необходимо знать: строка меню (menu bar), панель **Modes**, панель **World Outliner**, панель **Details**, панель **Content Browser**, панель инструментов редактора уровней и панель **Viewport** (см. рис. 1.4). Мы рассмотрим каждую из них в следующих разделах.

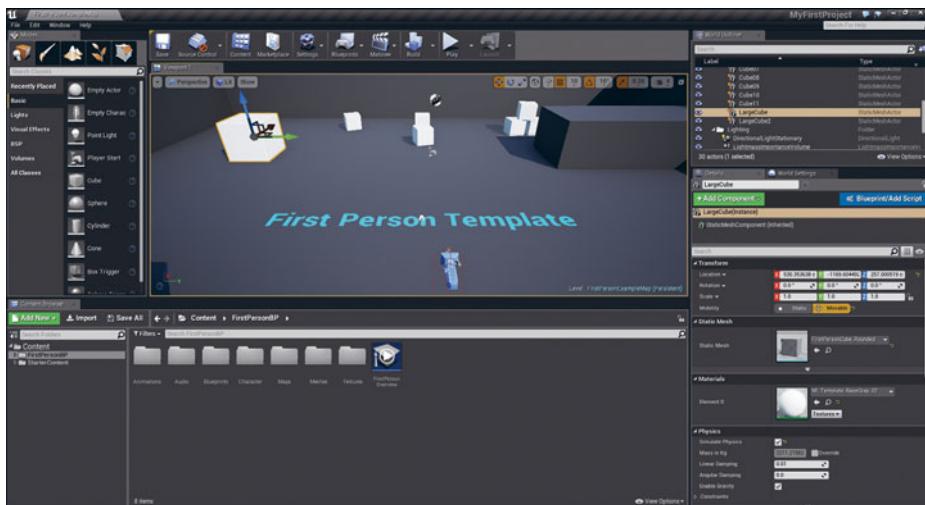


Рис. 1.4. Главный интерфейс по умолчанию редактора UE4

ПРИМЕЧАНИЕ

Разметка интерфейса

Вы можете настроить разметку интерфейса редактора, изменяя расположение панелей и окон, чтобы улучшить рабочий процесс. В этой книге все элементы интерфейса расположены по умолчанию, но, когда вы наберетесь опыта, возможно, захотите изменить расположение элементов, чтобы повысить продуктивность работы.

Строка меню

Строка меню, как и в большинстве современных приложений, состоит из пунктов **File**, **Edit**, **Window** и **Help**. **File** содержит операции загрузки и сохранения проектов и уровней. **Edit** включает стандартные операции копирования и вставки, а также предпочтения редактора и настройки проекта. **Window** открывает окно просмотра проекции (выюорт) и другие панели. Если вы закрыли окно или панель, то перейдите в меню **Window**, чтобы открыть его вновь. **Help** содержит ссылки на внешние ресурсы, такие как онлайн-документация и обучающие руководства.

Панель Modes

Панель Modes отображает различные режимы редактора уровней (см. рис. 1.5). Она позволяет специализированным интерфейсам редактирования работать с соответствующими типами актеров и геометрии.

СОВЕТ

Что такое актер?

Ключ к освоению любого программного продукта состоит в изучении его интерфейса, рабочего процесса и словаря. Unreal описывается обширной терминологией, которую вы изучите в этой книге. Термин **актер** (Actor) применяется к любым ассетам, помещенным на уровень. Например, 3D-модель на панели **Content Browser** упоминается как *Static Mesh asset* (ассет статичного меша). Но когда экземпляр ассета статичного меша помещен на уровень, он начинает упоминаться как *Static Mesh Actor* (актер статичного меша).

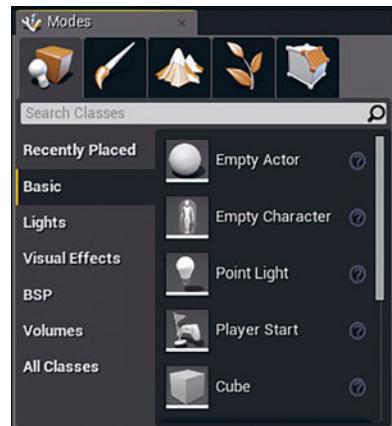


Рис. 1.5. Панель Modes

Панель Modes состоит из различных инструментальных режимов редактора уровней, которые позволяют изменять его работу. Здесь вы можете выбрать специализированную задачу, такую как помещение новых ассетов в мир, скульптурирование ландшафтов, создание геометрических кистей и объемов, генерирование растительности или окрашивание мешей. В табл. 1.1 перечислены режимы редактора и описаны их эффекты.

ТАБЛ. 1.1. Режимы редактора

Действие	Эффект
Режим Place	Помещение актеров на сцену
Режим Paint	Изменение цвета вершин в актерах статичных мешей
Режим Landscape	Редактирование ландшафтных актеров
Режим Foliage	Добавление актеров растительности на уровне
Режим Geometry Editing	Редактирование актеров кисти по вершинам на уровне

Панель World Outliner

Панель World Outliner отображает всех актеров на текущем уровне в виде иерархического дерева (см. рис. 1.6). Вы можете выбрать актера, щелкнув по его названию на панели World Outliner, его свойства отобразятся на панели Details. Если

дважды щелкнуть по названию, панель **Viewport** сфокусируется на выбранном вами ассете.

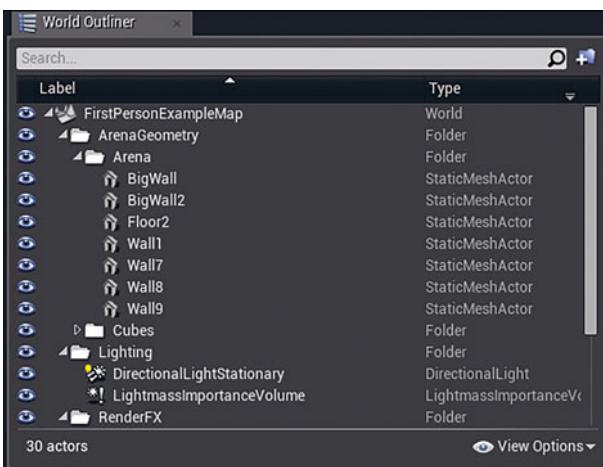


Рис. 1.6. Панель World Outliner

Панель Details

Панель **Details** является одной из наиболее часто используемых в UE4. Панель **Details** есть практически в каждом подредакторе. Она отображает все редактируемые свойства выбранных актеров на панели **Viewport**. Эти свойства зависят от типа выбранного актера, но существуют некоторые общие свойства, имеющиеся у большинства актеров (см. рис. 1.7). Типичные свойства включают название актера, блоки трансформации и редактирования для перемещения, поворота и масштабирования актеров, а также свойства отображения.

ПРИМЕЧАНИЕ

Выбор актеров

Чтобы выбрать актера, щелкните по нему в окне **Viewport** или на панели **World Outliner**. После этого актер будет подсвечен, а свойства появятся на панели **Details**. Можно выбрать несколько актеров одновременно.

- ▶ В окне **Viewport** или на панели **World Outliner** удерживайте клавишу **Ctrl** или **Shift**, когда добавляете или снимаете выделение с актеров.
- ▶ В окне **Viewport**, удерживая комбинацию клавиш **Ctrl+Alt**, щелкните левой кнопкой мыши и перетащите курсор, чтобы создать ограничивающую рамку выделения вокруг нескольких актеров. Перед этим окно **Viewport** должно быть активно.

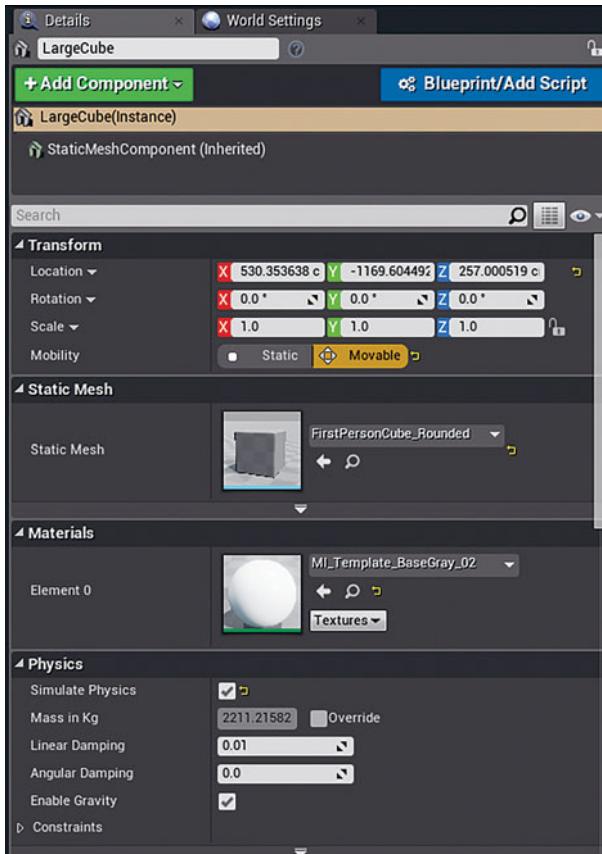


Рис. 1.7. Панель Details основного редактора

Панель Content Browser

Панель **Content Browser** — основная область управления ассетами в проекте (см. рис. 1.8). Этот браузер используется для задач, связанных с контентом, таких как создание, просмотр, изменение, импорт и организация. Он также позволяет управлять папками и выполнять базовые операции с ассетами, такие как просмотр ссылок, перемещение, копирование и переименование. В **Content Browser** есть строка поиска и флагки фильтра для быстрого поиска ассетов.

Можно представить панель **Content Browser** как игрушечную коробку с бесконечными ассетами. При необходимости вы можете достать из нее экземпляр (то есть копию) ассета и поместить его на уровень. Как только экземпляр будет помещен на уровень, он станет **актером**. Первоначальный экземпляр размещенного актера является точной копией оригинального ассета, содержащегося на панели **Content**.

Browser. В левой части панели **Content Browser** находится панель **Source**, которая отображает иерархию папок контента. Ее можно развернуть или свернуть, щелкнув по иконке в левом верхнем углу под зеленой кнопкой **Add New**. Правая сторона панели **Content Browser** называется **Asset Management** и показывает ассеты в выбранной папке на панели **Source**.

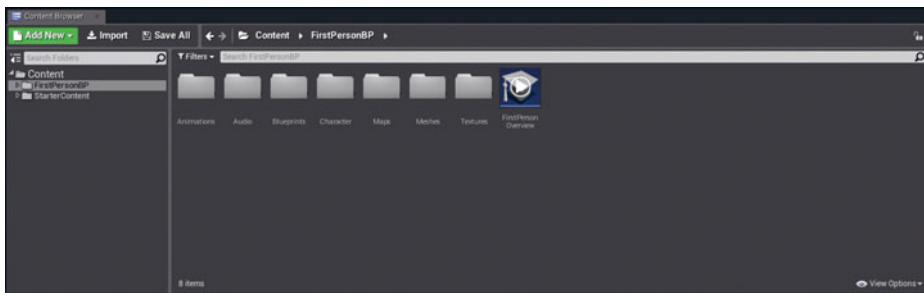


Рис. 1.8. Панель Content Browser с панелью Source слева и областью управления ассетами справа

СОВЕТ

Организация папок

Сложность проектов может расти очень быстро, поэтому то, как организованы файлы, чрезвычайно важно для поддержания среды эффективной работы. Хорошим правилом будет распределять ассеты по типам в отдельные папки. Папки можно вкладывать друг в друга, чтобы достичь максимальной организованности и гибкости.

Панель Viewport

Выопорты (Viewports) — это панели для конструирования и просмотра созданных вами миров. Панель **Viewport** используется для перемещения по текущему уровню. Эта панель имеет множество различных режимов, расположений и настроек, которые помогут вам создавать и редактировать уровни, а также управлять ими (см. рис. 1.9).



Рис. 1.9. Панель Viewport

Разметка панели Viewport

По умолчанию панель **Viewport** отображает представление **Perspective** на одной панели, но вы можете изменить вид на двух-, трех- или четырехпанельное отображение, щелкнув по выпадающему меню **Viewport**, выбрав пункт **Layouts**, где можно выбрать удобное для вас расположение панелей на экране (см. рис. 1.10). Вы можете настроить каждую панель во вьюпорте, изменив ее режим представления.

Типы окон предпросмотра

Существует два базовых типа предпросмотра: **Perspective** (перспективная проекция) и **Orthographic** (ортогональная проекция; см. рис. 1.11). Перспективная проекция отображает трехмерный мир с точками схода, в то время как ортогональная проекция показывает мир в двухмерном схематическом представлении. Перспективная проекция, скорее всего, будет вашей основной рабочей средой, а ортогональные проекции больше подойдут для точного размещения актеров на сцене.

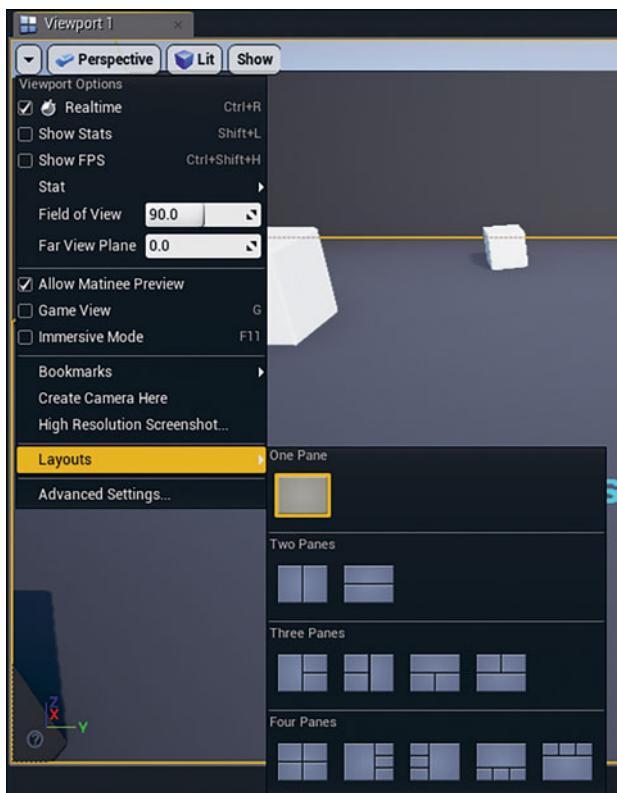


Рис. 1.10. Варианты разметки панели Viewport



Рис. 1.11. Настройки представлений выюпорта

Режимы просмотра и визуализаторы

Режимы просмотра (View modes; см. рис. 1.12) меняют отображение мира во вьюпорте, независимо от типа представления, и могут обеспечить важную обратную связь о состоянии уровня. В табл. 1.2 перечислены часто используемые режимы просмотра.



Рис. 1.12. Режимы просмотра

ТАБЛ. 1.2. Основные режимы просмотра окон предпросмотра

Режим	Эффект
Lit	Показывает окончательный вид сцены с материалами и примененным освещением
Unlit	Удаляет все освещение со сцены, показывая базовый цвет заданных материалов
Wireframe	Показывает все грани полигонов актеров на сцене
Detail Lighting	Отображает нейтральный материал на протяжении всей сцены, используя карты нормалей заданных материалов
Lighting Only	Отображает нейтральный материал, на который воздействует только освещение без данных у карт нормалей

ПРИМЕЧАНИЕ

Визуализация

Вам доступно более тринадцати различных режимов просмотра, а также другие инструменты визуализации. Вы можете использовать их, чтобы получать обратную связь на уровне, а также чтобы производить отладку и устранять неисправности.

Флажки Show

Как и режимы просмотра, флажки **Show** предоставляют возможность настраивать отображение важной информации прямо во виджете уровня, например выводить оболочки коллизий актеров или ограничительные рамки.

Навигация по сцене в перспективной проекции

Теперь, когда вы имеете базовое понимание каждой ключевой области основного интерфейса, необходимо узнать, как использовать виджет для перемещения по уровню. В табл. 1.3 и 1.4 приведены наиболее часто используемые команды для передвижения по уровню во виджете.

ТАБЛ. 1.3. Команды передвижения во виджете

Команда	Действие
Перспективная проекция	
ЛКМ + перетаскивание	Перемещает камеру виджета вперед и назад и поворачивает ее влево и вправо
ПКМ + перетаскивание	Поворачивает камеру виджета на месте без движения вперед и назад
ЛКМ + ПКМ + перетаскивание	Перемещает камеру виджета вверх, вниз
Ортогональная проекция (сверху, спереди, сбоку)	
ЛКМ + перетаскивание	Создает область выделения
ПКМ + перетаскивание	Передвигает Orthographic-представление влево и вправо
ЛКМ + ПКМ + перетаскивание	Приближает и отдаляет Orthographic-представление

ПРИМЕЧАНИЕ

Навигация по уровню

В отличие от приложений для 3D-моделирования, которые настроены на фокусирование на единственном создаваемом ассете и движении вокруг него, команды передвижения во виджете Unreal Engine разработаны для декорирования больших игровых уровней. Поэтому быстрое перемещение через большие области является ключевым.

ТАБЛ. 1.4. Команды вращения, приближения и передвижения виджета

Команда	Действие
F	При нажатии клавиши F камера виджета фокусируется на выбранном актере во виджете
Alt + ЛКМ + перетаскивание	Поворачивает виджет вокруг якорной точки или интересного места
Alt + ПКМ + перетаскивание	Приближает или отдаляет камеру от якорной точки или интересного места
Alt + СКМ + перетаскивание	Передвигает камеру влево, вправо, вверх и вниз по направлению движения мыши

СОВЕТ

Навигация в игровом стиле

Если выбрана перспективная проекция, при удерживании нажатой правой кнопки мыши вы можете использовать клавиши W, A, S и D для перемещения по уровню, как в шутере от первого лица.

Панель инструментов редактора уровней

Панель инструментов редактора уровней предоставляет быстрый доступ к часто используемым инструментам и операциям, таким как сохранение текущего уровня, просчет заранее рассчитанного освещения для статичных актеров, изменение свойств отображения редактора и игровое тестирование текущего уровня. На рис. 1.13 изображена панель инструментов редактора уровней.



Рис. 1.13. Панель инструментов редактора уровней

Проигрывание уровня

При создании нового проекта вам сразу доступен уровень по умолчанию. Он является первым, что вы видите в редакторе, когда открываете проект. Игровое тестирование уровня включает использование системы ввода, которую ваши игроки будут использовать для взаимодействия с игрой. Существует несколько различных режимов игрового тестирования уровня (см. рис. 1.14). Пока попробуйте основные режимы **Play in Editor** (PIE): **Selected Viewport** (выбранный виджет) и **New Editor Window** (новое окно редактора). Вы можете щелкнуть по иконке **Play**, чтобы проиграть уровень, или по стрелке выпадающего меню справа от иконки **Play** и выбрать один из режимов игры.

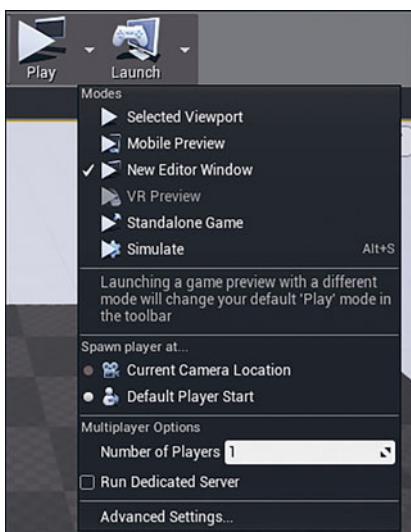


Рис. 1.14. Режимы игры

СОВЕТ

Игровое тестирование уровня

Последний режим игры, который вы использовали, автоматически становится режимом игры по умолчанию на панели инструментов редактора уровней. Если вы хотите использовать другой режим игры, щелкните по выпадающему меню **Play** и выберите нужный режим.

Резюме

Вы загрузили и установили Unreal Engine, познакомились с ключевыми областями основного интерфейса, создали свой первый проект, узнали, как передвигаться по уровню во вьюпорте и как производить игровое тестирование уровня. Чем ближе вы познакомитесь с этими задачами, тем больше навыков получите.

Вопросы и ответы

Вопрос: Зачем нужен Epic Games Launcher?

Ответ: Epic Games Launcher позволяет управлять проектами, обновлять UE4 и предоставляет доступ к магазину для приобретения контента.

Вопрос: Где установлен движок UE4?

Ответ: Движок UE4 установлен в том же месте, где и лаунчер.

Вопрос: Где следует сохранять проекты?

Ответ: Проекты необходимо сохранять на жестком диске, на котором имеется достаточно свободного места.

Семинар

Закончив этот час, попробуйте ответить на следующие вопросы.

Контрольные вопросы

1. Истинно или ложно высказывание: панель **Mode** позволяет переключаться между различными режимами редактирования.
2. Если вы хотите сфокусировать вьюпорт на выбранном актере, какую клавишу или комбинацию клавиш следует использовать?
3. Истинно или ложно высказывание: любой ассет, помещенный на уровень, называется актером.
4. Вы используете _____ для управления и создания новых проектов.
5. Истинно или ложно высказывание: разметка основного интерфейса полностью изменяется.
6. Истинно или ложно высказывание: аббревиатура PIE расшифровывается как Play in Editor.

Ответы

1. Истина. Панель **Mode** позволяет переключаться между расположением актеров, созданием ландшафта и созданием растительности.
2. Клавиша **F** сфокусирует выюорт на выбранном в данный момент актере.
3. Истина. Вне зависимости от типа, когда ассет помещен на уровень, он становится экземпляром исходного ассета и называется актером.
4. Браузер проектов (**Project Browser**). Вы можете работать над множеством проектов в одно и то же время. Браузер проектов позволяет переключаться между проектами.
5. Истина. Если вы щелкните по одной из вкладок панели и перетащите ее, то сможете перемещать ее по интерфейсу.
6. Истина. Аббревиатура PIE расшифровывается как Play in Editor. Вы можете просматривать уровень в отдельном окне или в выбранном выюорте.

Упражнения

Попрактикуйтесь самостоятельно с интерфейсом UE4. В этом упражнении потренируйтесь в создании нового уровня и помещении в него актеров, а затем сохраните уровень. Это простые, но базовые умения, и чем лучше вы их освоите, тем успешнее вы будете работать с Unreal Engine.

1. Создайте новый уровень по умолчанию, выбрав пункт **File ⇒ New Level ⇒ Default** или нажав комбинацию клавиш **Ctrl+N**.
2. Чтобы поместить актер точечного источника света на уровень, выберите его во вкладке **Place** панели **Modes**.
3. Поместите ассет статичного меша на уровень. Вы можете найти этот ассет в папке *StarterContent* на панели **Content Browser**.
4. Чтобы сохранить уровень, щелкните правой кнопкой мыши по папке *Content* на панели **Content Browser**, выберите пункт **Add New Folder** и назовите новую папку *Maps*.
5. Сохраните ваш уровень в только что созданной папке *Maps*, выбрав пункт меню **File ⇒ Save Current**.

2-Й ЧАС

Изучение системы Gameplay Framework

Что вы узнаете в этом часе

- ▶ Загрузка и установка примера проекта с контентом
- ▶ Импорт ассетов
- ▶ Перенос контента из одного проекта в другой
- ▶ Знакомство с системой Gameplay Framework

Unreal Engine 4 (UE4) — это проработанное, богатое приложение, которое может использоваться для создания чего угодно, начиная от двухмерных инди-игр и трехмерных многомилионных проектов и заканчивая интерактивными приложениями, архитектурной визуализацией и приложениями виртуальной реальности. UE4 способен создавать контент для различных платформ: от персональных компьютеров и консолей до мобильных устройств и веб-приложений с HTML-разметкой. Редактор UE4 интегрирует множество сложных процессов разработки в удобную для использования среду разработчика. Как и с другими сложными инструментами, потребуется время на изучение UE4. Этот час представляет терминологию, анатомию проекта и основы системы Gameplay Framework движка UE4.

Доступные ресурсы

Одно из важных нововведений, сделанных Epic с выходом версии 4 движка Unreal Engine и его редактора, заключается в предоставлении качественной онлайн-документации и примеров проектов. В лаунчере Epic Games есть раздел Unreal Engine, предлагающий новости, освещение проектов и ссылки на форумы, блоги, а также дорожная карта разработки движка. Раздел Learn (Обучиться) также предоставляет ссылки на онлайн-документацию, видеоИнструкции и пробные проекты, демонстрирующие различные темы. Существуют категории проектов, иллюстрирующие различные возможности движка UE4, представляющих примеры геймплея, примеры законченных игровых проектов и образцы проектов, являющиеся вкладом сообщества Unreal и партнеров Epic. Познакомившись

с интерфейсом и рабочими процессами UE4, лучший способ обучаться — это разбирать существующие проекты.

ПОПРОБУЙТЕ САМИ

Загрузите пример проекта с контентом

Чтобы увидеть, какие удивительные вещи можно сделать с помощью Unreal Engine 4, перейдите к разделу **Learn** (Обучиться) в Epic Games Launcher и исследуйте проект **Content Examples** в категории **Engine Feature Samples**. Загрузите и установите проект, использующий образцы контента.

1. В Epic Games Launcher откройте вкладку **Learn** (Обучиться).
2. В категории **Engine Feature Samples** найдите проект **Content Examples** и откройте, щелкнув по нему.
3. Убедитесь, что версия движка, указанная на открывшейся странице, соответствует установленной вами, и нажмите кнопку **Free**. Начнется процесс загрузки проекта объемом более двух гигабайт. Когда проект **Content Examples** будет установлен, то появится в разделе **Vault** (Хранилище) на вкладке **Library** (Библиотека) в лаунчере.
4. В лаунчере откройте вкладку **Library** (Библиотека) и найдите раздел **Vault** (Хранилище). Затем щелкните по кнопке **New Project** (Новый проект) под заголовком **Content Examples**.
5. Дайте проекту название или оставьте название по умолчанию.
6. Выберите нужное вам расположение на жестком диске или оставьте предложенный вариант.
7. Проверьте/выберите версию, которая будет соответствовать установленной вами версии движка.
8. Щелкните по кнопке **Create** (Создать). В течение нескольких секунд лаунчер создаст новый проект с предлагаемым контентом.
9. Откройте проект в редакторе. Дважды щелкните по только что созданному проекту в разделе **Samples**.
10. После того как проект будет загружен в редактор, выберите пункт меню **File** ⇒ **Open Level**.
11. Выберите любой уровень и дважды щелкните по нему или выделите его и щелкните по кнопке **Open**.
12. Когда уровень откроется, изучите его, щелкнув по кнопке **Play** на панели инструментов основного редактора.
13. Продолжайте открывать и изучать все интересное, что может продемонстрировать проект.

Play in Editor (PIE)

Play in Editor (PIE, воспроизвести в редакторе) предоставляет доступ к опциям, позволяющим проводить игровое тестирование уровня без предварительной компиляции или упаковки контента. Варианты предварительного просмотра PIE находятся в главной строке меню редактора уровней под кнопкой **Play** (см. рис. 2.1). Если щелкнуть по перевернутому треугольнику справа от кнопки **Play** на панели инструментов редактора уровней, можно увидеть множество вариантов предварительного просмотра уровня. По умолчанию UE4 использует вариант **Selected Viewport**, что хорошо, когда вы тестируете функционал. В то же время вам может понадобиться увидеть уровень в определенном разрешении или формате экрана целевой платформы.

Выбор варианта **New Editor Window** изменит иконку предварительного воспроизведения и запустит предварительный просмотр уровня в новом окне. Внизу выпадающего меню под вариантами предварительного просмотра есть вариант **Advanced Settings** (расширенная настройка), который можно выбрать, чтобы остановить автоматический выбор настроек редактора. Затем в правой части экрана под кнопкой **Play in New Window** вы можете установить разрешение окна, используя выпадающий список с основными настройками. Вы также можете установить расположение окна; по умолчанию это значение равно 0,0, то есть верхний левый угол вашего монитора. Вы можете установить флажок **Always Center Window to Screen**, если хотите, чтобы новый виджет редактора отображался в центре монитора.

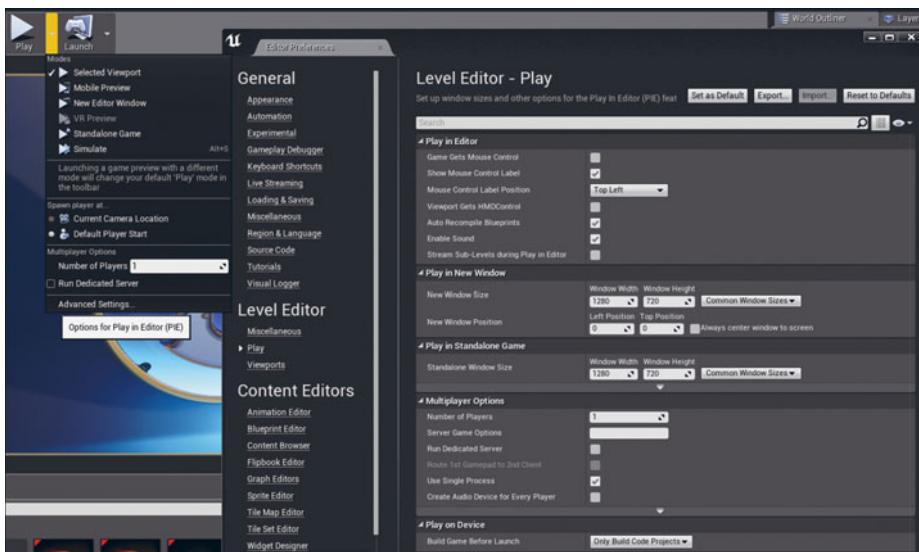


Рис. 2.1. Варианты Play in Editor (PIE) в настройках редактора

Структура папок проекта

Когда вы впервые создаете основанный на блюпринтах проект с нуля при помощи Epic Games Launcher, необходимо указать месторасположение и название проекта. После чего редактор копирует коллекцию установочных папок и файлов в папку проекта для вас. Если вы заглянете в папку проекта после его создания, обнаружите в ней папки *Config*, *Content*, *Intermediate* и *Saved*, а также файл *.uproject*:

- ▶ **Config.** Эта папка содержит установочные файлы *.ini*, в которых хранятся настройки редактора и проекта по умолчанию.
- ▶ **Content.** Эта папка хранит все ассеты проекта, которые являются важными или были созданы прямо в редакторе.
- ▶ **Intermediate.** В этой папке хранятся рабочие настройки проекта в файлах *.ini* и файлы настроек, наряду с файлом *CachedAssetRegistry.bin*.
- ▶ **Saved.** В этой папке содержатся файлы автосохранения редактора и резервные копии, папка *Collections* для ассетов, организованная в коллекции в Браузере проектов, настройки *Config* проекта для целевых платформ, файлы с журналами записей (логов) редактора, а также миниатюрное изображение проекта в формате *.png*, которое отображается в лаунчере.

По мере работы над проектом в него будет добавляться все больше файлов и папок.

ПОПРОБУЙТЕ САМИ

Создайте пустой проект без стартового контента

Чтобы ознакомиться с типичной структурой папок проекта, в этой рубрике «Попробуйте сами» вы создадите пустой проект без стартового контента.

1. В лаунчере на вкладке **Library** (Библиотека) откройте ту версию редактора, которую вы установили в 1-м часе, «Введение в Unreal Engine 4», щелкнув по кнопке **Launch** (Запустить) под номером версии.
2. В Браузере проектов Unreal выберите вкладку **New Project**.
3. На вкладке **Blueprint** выберите шаблон **Blank**.
4. Убедитесь, что правильно выбрана настройка **Desktop/Console**.
5. Убедитесь, что в настройке **Graphics Level** выбрано качество, поддерживаемое вашим оборудованием.
6. Убедитесь, что выбран вариант **No Starter Content**.
7. Выберите местоположение проекта.

8. Назовите проект **MyHour02**. Запомните, что название проекта не может содержать пробелы.
9. Щелкните по кнопке **Create Project**. Когда проект будет создан, он откроется автоматически.
10. Сохраните его на жестком диске и сверните проект.

Если вы изучите все папки, то обнаружите, что некоторые файлы уже созданы, в том числе файлы конфигурации. Все файлы с расширением *.ini* являются файлами конфигурации. Это текстовые файлы, которые хранят настройки редактора, движка и игры. Все ваши изменения в проекте или настройках отражаются в этих файлах.

ПРИМЕЧАНИЕ

Настройки редактора и проекта

Вы можете найти настройки редактора и проекта в пункте **Edit** в строке меню редактора уровней.

Папка Content

В папке *Content* редактор сохраняет весь импортированный и перенесенный контент проекта (см. рис. 2.2). В ней, как правило, находятся файлы двух типов: с расширением *.uasset* и *.umap*. Когда вы импортируете внешний ассет, он сохраняется как файл с расширением *.uasset* в папку *Content* проекта. Каждый раз, когда вы создаете новую карту и сохраняете ее, UE4 создает файл с расширением *.umap* и сохраняет его в эту же папку. Когда вы используете панель **Content Browser** в редакторе, вы видите структуру директорий в папке *Content*.

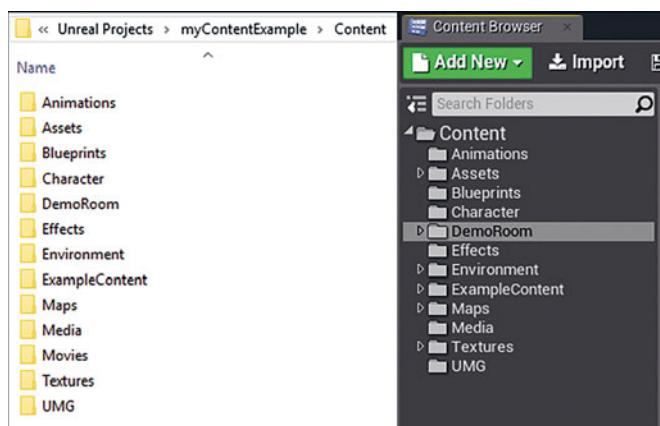


Рис. 2.2. Папка *Content* проекта слева и панель **Source** на панели **Content Browser** справа

Импорт контента

Unreal Engine 4 поддерживает множество типов файлов для импорта контента. В табл. 2.1 приведены некоторые из наиболее часто используемых типов файлов и ассетами, с которыми они связаны.

ТАБЛ. 2.1. Основные типы внешних файлов, которые могут быть импортированы в редактор

Тип ассета	Файловое расширение
Трехмерная модель, риги скелетных мешей, данные анимации	.fbx, .obj
Текстуры и изображения	.bmp, .jpeg, .pcx, .png, .psd, .tga, .hdr
Шрифты	.otf, .ttf
Аудио	.wav
Видео и мультимедиа	.wmv
PhysX	.apb, .aprx
Другие	.csv

СОВЕТ

Определение типов файлов

Операционные системы, как правило, скрывают файловые расширения по умолчанию. Когда вы начинаете работать над проектом, имеющим большой объем контента, может быть трудно отслеживать тип каждого файла. Вы можете включить отображение расширений файлов в вашей ОС, чтобы видеть тип искомого файла.

Существует несколько способов внести контент в проект. Например, вы можете импортировать контент, созданный во внешнем редакторе, таком как 3DS Max или Maya для создания моделей, Photoshop для создания текстур и Audacity для создания звуков.

Импортировать новый контент, созданный во внешнем приложении, можно двумя способами. Первый способ заключается в использовании панели **Content Browser**, как описано в следующей рубрике «Попробуйте сами». Второй — в том, чтобы открыть файловый менеджер вашей операционной системы, выбрать нужный файл и перетащить его в область управления ассетами на панели **Content Browser** (см. рис. 2.3).

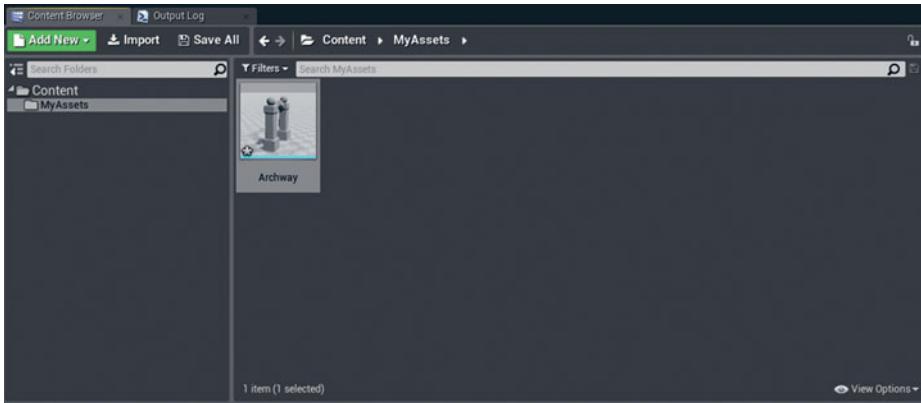


Рис. 2.3. Панель Content Browser с панелью Source в левой части окна и областью управления ассетами — в правой

ПОПРОБУЙТЕ САМИ

Создайте папку на панели Content Browser и импортируйте внешний ассет

Импорт ассетов — это одна из наиболее частых операций, которые вы будете выполнять. В этой рубрике «Попробуйте сами» вы научитесь простому способу импортирования ассета в проект.

1. Откройте проект *MyHour02*, который вы создали в предыдущей рубрике «Попробуйте сами».
2. Щелкните по иконке **Show or hide the sources panel** под зеленой кнопкой **Add New**, чтобы увидеть панель **Source** в Content Browser.
3. На панели **Source** в Контент браузере щелкните правой кнопкой мыши по папке **Content**, в выпадающем списке выберите пункт **New Folder** и назовите новую папку **MyAssets**.
4. Убедитесь, что папка **MyAssets** на панели **Source** выбрана, затем щелкните правой кнопкой мыши по области управления ассетами и выберите пункт **Import Asset** ⇒ **Import To** в выпадающем списке. Появится диалоговое окно **Import**.
5. В диалоговом окне **Import** найдите папку *Hour_02* в материалах проектов, выберите один из файлов в папке *RawAssets* и щелкните по кнопке **Open**. Этот файл будет добавлен в папку *Content*.
6. Щелкните правой кнопкой мыши по изображению ассета и выберите пункт **Save** в выпадающем списке, чтобы сохранить только что импортированный ассет.

СОВЕТ

Иконки ассетов

Иконки ассетов на панели **Content Browser** обеспечивают предварительный просмотр большинства ассетов, благодаря чему нет необходимости открывать их. Если вы наведете указатель мыши на иконку, вы увидите важную информацию об ассете. Например, звездочка в левом нижнем углу иконки ассета говорит о том, что ассет не был сохранен. Каждый раз, когда вы импортируете ассет или изменяете его, отображается звездочка, сигнализирующая о том, что изменения должны быть сохранены. Если вы закроете редактор без сохранения импортированных или измененных ассетов, то потеряете добавленные ассеты и все внесенные изменения. Нажмите комбинацию клавиш **Ctrl+S**, чтобы сохранить все ассеты, или щелкните правой кнопкой мыши по ассету и выберите пункт **Save**.

Перенос контента из существующего проекта

Другой способ добавления контента в проект заключается в его переносе из существующего проекта. Каждый проект имеет собственную папку *Content*, в которой хранятся все ассеты этого проекта. Перенос позволяет перемещать ассеты из одного проекта в другой. Когда вы переносите ассеты, вы также перемещаете их зависимости для поддержки структуры папок.

ПОПРОБУЙТЕ САМИ

Перенесите контент

Выполните приведенные ниже шаги, чтобы перенести контент из одного проекта в другой.

1. Откройте проект *Content Examples*, который вы использовали в начале этого часа.
2. На панели **Source** Браузера проектов, выберите пункт **Content** (наверху директории).
3. В области управления ассетами слева от поисковой строки щелкните по кнопке **Filters** и установите флажок **Particle System**. Теперь вы видите все системы частиц в этом проекте в области управления ассетами.
4. Зажмите клавишу **Ctrl** и щелкните левой кнопкой мыши, чтобы выбрать некоторые системы частиц, которые хотите перенести в свой проект.
5. После того как вы выбрали три-четыре системы частиц, щелкните правой кнопкой мыши по одной из подсвеченных систем и в выпадающем списке выберите пункт **Asset Actions ⇒ Migrate** (см. рис. 2.4).
6. В окне **Asset Report**, отображающем не только сами системы частиц, но и их связи, щелкните по кнопке **OK**.

7. Чтобы найти папку *Content* проекта, в который вы хотите скопировать эти ассе- ты, в окне **Choose a destination Content folder** найдите папку *Content* проекта *MyHour02* и, выбрав ее, щелкните по кнопке **OK**.

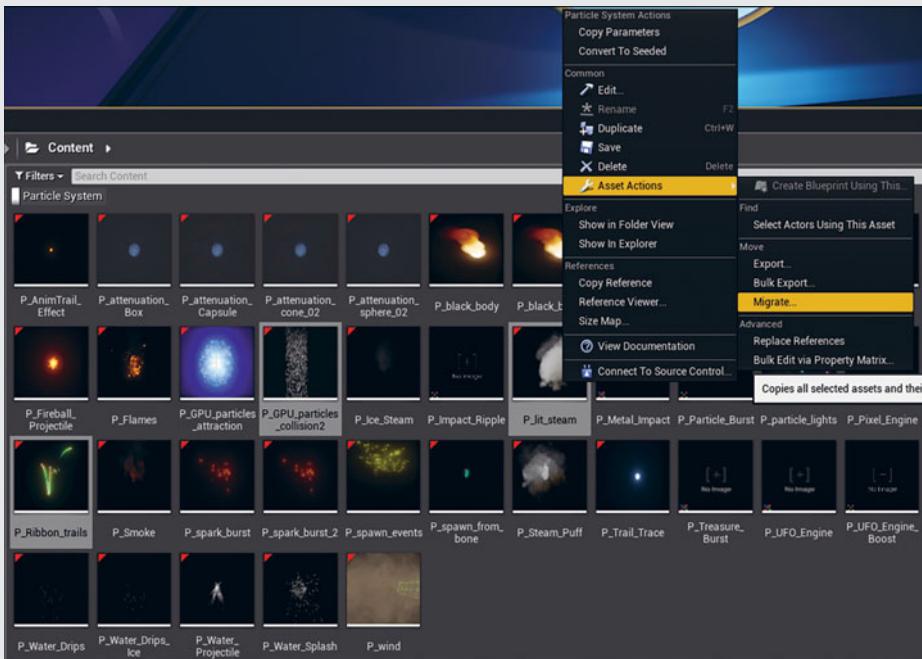


Рис. 2.4. Перенос контента на панели **Content Browser**

ПРИМЕЧАНИЕ

Фильтры панели **Content Browser**

Чем больше контента у вас в проекте, тем труднее становится поиск нужных файлов и папок. Вот почему организация папок и соглашение о названиях так важны. В верхней части обзора ассетов на панели **Content Browser** имеется поисковая строка и инструменты фильтрации, относящиеся к выбранной папке. Если вы, например, выбрали верхнюю папку *Content*, поисковая строка и инструменты поиска будут применяться к этой папке и всем вложенным в нее папкам.

Другие типы ассетов

Многие ассеты не импортируются, а создаются прямо в редакторе (например, классы блупринтов, системы частиц и данные анимации камеры). В течение следующих 22 часов вы научитесь добавлять и создавать некоторые из этих типов ассетов.

СОВЕТ**Создание папки для необработанных ассетов для ваших проектов**

При работе над крупными проектами в больших группах может потребоваться специальное ПО для управления файлами проекта. Но даже без специального ПО вы можете поддерживать порядок ваших файлов, храня их в различных папках.

Ниже приведен пример структуры директорий для хранения внешних ассетов организованно, прежде чем они будут импортированы в проект.

- ▶ Папка *Raw Assets*
 - › Вложенная папка *Models*
 - › Вложенная папка *Audio*
 - › Вложенная папка *Textures*

Связи ассетов и средство просмотра связей

Вам может показаться привлекательным перемещать файлы *.uasset* вручную или копировать их из одного проекта в другой. Несмотря на то что технически это будет работать, такая практика отрицательно сказывается на зависимостях. Вы должны вносить изменения в расположения и структуру папок только на панели **Content Browser**, чтобы редактор мог обновить зависимости.

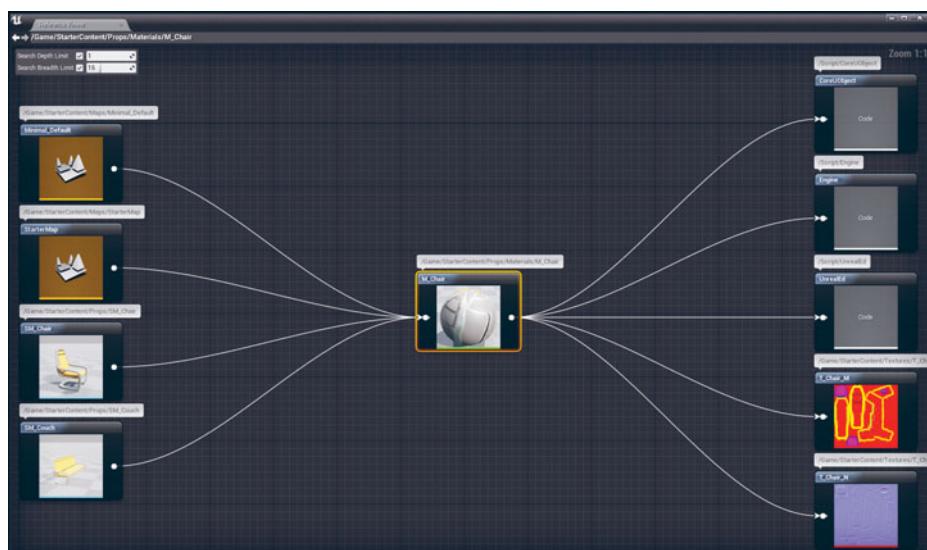


Рис. 2.5. Средство просмотра связей (Reference Viewer), отображающее зависимости ассета M_Chair Material

Средство просмотра связей (**Reference Viewer**) показывает зависимости ассета (см. рис. 2.5). Например, если вы присваиваете материал статичному мешу в редакторе статичных мешей, этот меш должен знать местоположение материала на панели **Content Browser**. С другой стороны, материал должен знать местоположение текстур, от которых он зависит. Если вы переместите материал в новую папку на панели **Content Browser**, редактор автоматически обновит связи ассета статичного меша. Чтобы увидеть связи ассета, щелкните правой кнопкой мыши по панели **Source** на панели **Content Browser** и выберите в выпадающем списке пункт **Reference Viewer**.

Папка **Saved**

Папка *Saved* содержит четыре папки: *AutoSaves*, *Backup*, *Config* и *Logs*. Редактор использует папки *AutoSaves* и *Backup* для создания файлов резервных копий и временных файлов для всего, что вы открываете или изменяете. Вы можете использовать эти файлы, чтобы сохранить день, если редактор когда-либо даст сбой. Однако эти файлы также значительно увеличивают размер проекта по мере работы, поэтому вы, возможно, захотите периодически чистить содержащие их папки, чтобы сохранять размер проекта небольшим. Папка *Config* содержит файлы *.ini*, которые используются для хранения настроек проекта.

Система **Gameplay Framework**

Gameplay Framework — это коллекция классов C++ или блупринтов, управляющих правилами игры, вводом игрока и аватарами, камерой, элементами HUD* в каждом проекте.

ПРИМЕЧАНИЕ

Больше чем игры

Не позволяйте терминологии вас одурачить. В то время как движок UE4 разрабатывался для создания игр и значительная часть терминологии связана с играми, этот движок может быть использован для создания множества различных приложений. Например, 3D-художник может создать средство просмотра моделей для демонстрации трехмерных моделей на сайте-портфолио.

* HUD (Heads-Up Display) — часть визуального интерфейса игры, отображающаяся поверх виртуального пространства игры и содержащая информацию, например, о здоровье и количестве амуниции персонажа, прицел или мини-карту. — Прим. ред.

Класс **GameMode**

Класс GameMode используется для задания правил игры и доступа к другим классам, необходимым для управления ключевой функциональностью игры. Например, класс GameMode — хорошее место для того, чтобы запрограммировать системы респаун* для шутера от первого лица или таймер для гоночной игры. Ниже представлен список классов Gameplay Framework, которые назначаются для класса GameMode:

- ▶ DefaultPawn
- ▶ HUD
- ▶ PlayerController
- ▶ Spectator
- ▶ ReplaySpectator
- ▶ PlayerState
- ▶ GameState

После того как были созданы режим игры и зависимости, вы можете назначить режим игры проекту или каждому отдельному уровню в проекте. Большинство проектов имеет два или три режима игры, но, разумеется, только один может быть назначен режимом по умолчанию. Это можно сделать в окне **Project Settings** в меню **Maps & Modes** ⇒ **Default Modes**. Когда режим игры установлен как режим по умолчанию, этот режим будет использоваться каждым уровнем игры, если для какого-либо уровня отдельно не был установлен режим игры по умолчанию в свойстве **GameMode Override** на вкладке **World Settings** редактора.

Классы **Controller**

Класс Controller управляет в игре аватаром (актером игрока). Класс PlayerController получает от игрока входные данные и использует их, чтобы направлять аватара. Существует два основных типа классов-контроллеров: PlayerController и AIController. Класс PlayerController управляет вводом от игрока и направляет аватар в игре, полностью владея ею. Ввод может осуществляться игроком с помощью мыши и клавиатуры, геймпадами и тачскринами, или контроллерами Xbox Kinect. При использовании класса PlayerController вы также можете включить отображение указателя мыши и проверить, как игра будет реагировать на события щелчков мыши. Каждый человек, играющий в игру, получает экземпляр класса PlayerController. Например, каждый раз, когда игрок

* Респаун (respawn) — возвращение персонажа в игру после его смерти. — Прим. ред.

присоединяется к мультиплеерной игре, экземпляр класса PlayerController создается внутри класса GameMode и присваивается этому игроку до конца игровой сессии. Классы PlayerController не имеют физического представления в мире игры.

Классы *Pawn* и *Character*

В Unreal Engine термин *Pawn* обозначает аватар игрока. Классы Pawn получают данные от класса PlayerController и используют их, чтобы направлять аватара в игре. Это может быть простым изображением местоположения игрока в уровне, или чем-то сложным, как анимированный скелетный меш с оболочкой коллизии, путешествующий по миру игры. Существует несколько классов, которые могут быть присвоены свойству класса DefaultPawn в режиме игры, такие как Pawn, Character и Vehicle. Класс Pawn — это обобщенный класс для создания различных типов аватаров, в то время как классы Character и Vehicle нацелены на работу со специфическими, но часто встречающимися во многих играх аватарами. Поскольку аватары получают указания от класса-контроллера, то могут управляться классами PlayerController и AIController.

Класс *HUD*

Класс HUD используется для окрашивания двухмерного интерфейса на экране игрока и создания внутриигрового визуального интерфейса игрока (HUD, Heads-Up Display). Вся HUD-система игры может быть заскриптована в классе HUD. Компания Epic также предоставляет редактор интерфейса, который называется Unreal Motion Graphics (UMG), представляющий собой коллекцию инструментов и классов для проектирования сложных интерфейсов и HUD-интерфейсов. (См. 22-й час, «Работа с UMG».)

В шаблонах проектов, предоставляемых компанией Epic, режимы игры для основных типов игр уже установлены. Как правило, шаблон режима игры выбирается каждый раз при создании нового проекта.

ПОПРОБУЙТЕ САМИ

Добавьте режим игры в проект

В начале этого часа вы создали пустой проект, который теперь можете использовать для того, чтобы научиться добавлять режимы игры, как пакеты функций. Выполните следующие шаги, чтобы добавить режим игры в пустой проект *MyHour02*.

1. Откройте проект *MyHour02*, который создали в предыдущей рубрике «Попробуйте сами».

2. На панели **Content Browser** щелкните по зеленой кнопке **Add New** над панелью **Source**.
3. Во всплывающем окне вверху выберите пункт **Add Feature or Content Pack**, после чего откроется окно **Add Content to the Project** (см. рис. 2.6).



Рис. 2.6. Окно Add Content to the Project

4. На вкладке **Blueprint Features** выберите вариант **Side Scroller** и щелкните по зеленой кнопке **+Add to Project**, чтобы добавить его в ваш проект.
5. На вкладке **Content Packs** выберите вариант **Starter Content** и щелкните по зеленой кнопке **+Add to Project**, чтобы добавить его в ваш проект.
6. Закройте окно **Add Content to the Project**.

Теперь на панели **Content Browser** вы видите несколько новых папок, которые были добавлены в ваш проект. Найдите папку *SideScrollerBP/Maps* и дважды щелкните по уровню **SideScrollerExampleMap**, чтобы открыть его. Просмотрите уровень, щелкнув по кнопке **Play** на панели инструментов редактора уровней. Как вы можете видеть, у вас есть основа сайд-скроллера*. Однако если вы откроете любой другой уровень, добавленный со стартовым контентом, в нем будет использоваться другой режим игры. Найдите папку *StarterContent/Maps*, откройте уровень

* Сайд-скроллер (side-scroller) — тип компьютерных игр, в которых виртуальная камера игрока смотрит на мир сбоку, а персонажи могут перемещаться в мире только по одной оси. — Прим. ред.

Minimal_Default и просмотрите его. Вы можете увидеть, что режим сайд-скроллерной игры не работает. Вам необходимо установить режим игры по умолчанию в проекте.

ПОПРОБУЙТЕ САМИ

Установите режим игры по умолчанию в проекте

Выполните следующие шаги, чтобы установить режим игры по умолчанию в ваш проект.

1. Откройте проект *MyHour02*, который вы создали в предыдущей рубрике «Попробуйте сами».
2. На панели **Content Browser** щелкните по зеленой кнопке **Add New** над панелью **Source**.
3. В строке меню редактора уровней выберите пункт **Edit ⇒ Project Settings**.
4. Под заголовком **Project** в левой части окна **Project Settings** выберите пункт **Maps & Modes** (см. рис. 2.7).

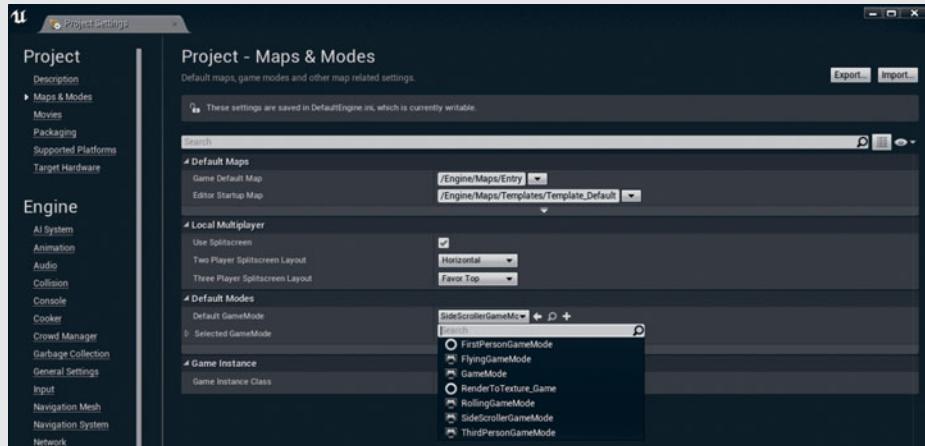


Рис. 2.7. Настройки проекта (Project Settings)

5. В правой части окна выберите пункт **SideScrollerGameMode** из выпадающего списка **Default GameMode**.
6. Закройте окно **Project Settings** и просмотрите уровень **Minimal_Default** еще раз.

Вы также можете присваивать режимы игры каждому отдельному уровню, воспользовавшись свойством **GameMode Override** на вкладке **World Settings** редактора и выбрав любой режим игры, добавленный в проект.

Резюме

В этом часе вы научились импортировать ассеты, переносить контент из одного проекта в другой и добавлять различные возможности в существующий проект. Вам также были представлены понятия, связанные с режимами игры, и вы узнали, как устанавливать режим игры по умолчанию для проекта.

Вопросы и ответы

Вопрос: Должен ли я присваивать режим игры каждому создаваемому мной уровню?

Ответ: Нет. Необходимо присваивать режим только тому уровню, режим игры которого будет отличаться от режима игры всего остального проекта. Когда режим игры установлен в настройках проекта, все уровни в игре используют этот режим.

Вопрос: Когда я переносил контент, редактор спросил меня, хочу ли я переписать существующий контент.

Ответ: Когда вы переносите контент, некоторые ассеты имеют одинаковые зависимости, и если они переносятся в разное время, редактор предупреждает, что некоторые исходные ассеты будут перезаписаны, что сотрет зависимость ассета, перенесенного первым. Вы должны восстановить зависимости вручную после переноса ассетов.

Вопрос: Что такое файл *.ini*?

Ответ: Файл *.ini* — это простой текстовый файл, хранящий настройки редактора и проекта. Несмотря на то, что файлы *.ini* можно открывать и редактировать в простом текстовом редакторе, в этом нет особой необходимости, поскольку ими управляет редактор UE4.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

- Истинно или ложно высказывание: лучший способ переместить контент из одного проекта в другой — скопировать файл *.uasset*.
- Истинно или ложно высказывание: не нужно сохранять ассет на панели **Content Browser** после того, как он был импортирован или изменен.
- Истинно или ложно высказывание: инструмент **Reference Viewer** позволяет увидеть зависимости ассета от других ассетов.

Ответы

1. Ложь. Правильным способом перемещения контента из одного проекта в другой является перенос.
2. Ложь. Когда вы впервые импортируете ассет или изменяете существующий, то должны сохранить его.
3. Истина. Инструмент **Reference Viewer** позволяет визуализировать зависимости ассета.

Упражнения

Создайте пустой проект и импортируйте внешние ассеты. Добавьте два шаблона режимов игры и стартовый контент для мобильных устройств. Назначьте один режим игры режимом по умолчанию, а второй присвойте уровню.

1. Создайте новый пустой проект (вариант **Blank**).
2. На панели **Content Browser** добавьте режим игры **Third Person**.
3. На панели **Content Browser** добавьте режим игры **Flying**.
4. На панели **Content Browser** добавьте **Mobile Starter Content**.
5. На панели **Content Browser** создайте новую папку **MyContent** и импортируйте внешний контент из папки *Hour_02*.
6. Назначьте режим **Third Person** режимом игры по умолчанию в этом проекте.
7. Создайте новый уровень и перезапишите режим игры, установив для свойства **GameMode Override** вариант **FlyingGameMode**

З-Й ЧАС

Координаты, преобразования, единицы измерения и организация

Что вы узнаете в этом часе

- ▶ Декартова система координат и как она связана с трансформациями в UE4
- ▶ Масштабирование, перемещение и поворот
- ▶ Система сеток и измерение актеров
- ▶ Организация сцены и ее структура
- ▶ Группировка актеров, слои и прикрепление

В этом часе вы узнаете об использовании координат и трансформаций. Вы научитесь использовать сетку для создания контента в трехмерном пространстве. Этот час исследует, какие типы трансформаций используются для контроля особых актеров в редакторе. Также вы узнаете, как управлять этими трансформациями, и какие инструменты помогут достичь наилучшего результата их применения. Затем вы познакомитесь с системой сеток и измерений, которая позволяет информации из пакетов многочисленных программных продуктов корректно транслироваться в UE4. Также мы рассмотрим некоторые организационные системы, используемые в UE4 для поддержания порядка хранения проектов.

Понимание декартовой системы координат

Умение создавать трехмерные объекты требует понимания прямоугольной системы координат. Декартова* система координат является прямоугольной системой координат с одинаковыми масштабами по всем трем осям. С ее помощью можно получить информацию или точки внутри заданного поля или области. Если в школе вы изучали геометрию, то имеете опыт использования декартовой системы

* Названа по имени Рене Декарта, французского математика XVII века. — Прим. ред.

координат. Когда точка помещается в двухмерное пространство, как на рис. 3.1, необходимо использовать два числа — одно для оси X, второе для оси Y. Место пересечения перпендикуляров, восстановленных из этих точек, описывает координаты точки или ее положение в заданном пространстве. Процесс аналогичен и в трехмерном пространстве, с той лишь разницей, что в нем существует три оси. Вместо использования двух чисел, трехмерное пространство использует координаты X, Y и Z. Каждая буква соответствует оси: Z — это ось, идущая сверху вниз, Y — слева направо, и X — спереди назад. Вся 3D-графика порождается на основе вычислений и отображения единственной точки, связанной с определенными значениями. Проводя прямые между точками, можно создать фигуру, в т. ч. объемную. Можно также использовать эти точки для манипуляции и перемещения, вычисляя положение и масштаб объекта в трехмерном пространстве.

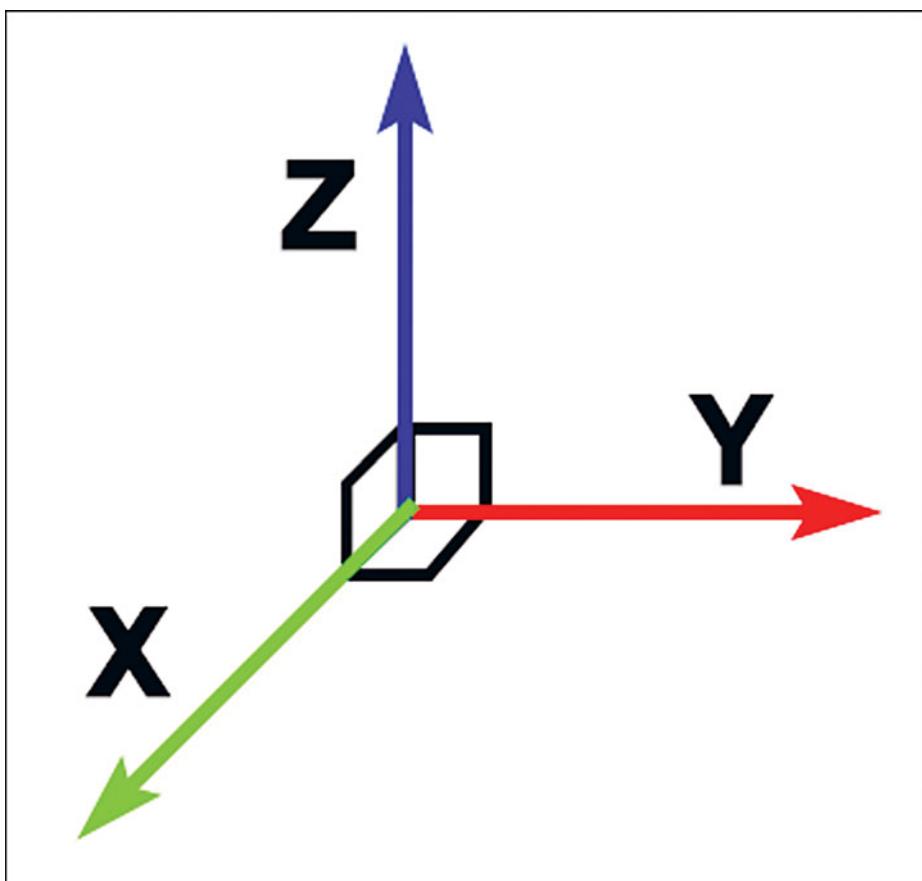


Рис. 3.1. Каждое направление в трехмерной системе координат привязано к осям X, Y и Z

Работа с трансформациями

Раздел ниже объясняет, как реализовать перемещение, масштабирование и поворот с помощью инструментов трансформации в редакторе UE4.

Инструменты трансформации

Инструмент трансформации используется для управления и манипуляций в трехмерном пространстве UE4. Существует три типа трансформации (см. рис. 3.2):

- ▶ переместить (Move);
- ▶ масштабировать (Scale);
- ▶ повернуть (Rotate).



Рис. 3.2. Виджет трансформации представляет каждый тип трансформации. Сверху вниз вы видите трансформации перемещения, масштабирования и поворота и соответствующее им отображение виджета

Трансформация перемещения

Трансформация перемещения (Move) — один из наиболее часто используемых инструментов трансформации в UE4. Этот инструмент позволяет перемещать актер в трехмерном пространстве из одного местоположения в другое.

Вы знаете, что актер имеет указанное в координатах X, Y и Z положение на сцене. Положение координат для каждого актера основано на якорной или исходной точке актера, в которую он был помещен в UE4 изначально. Чтобы переместить актер внутри сцены, необходимо сначала выбрать его, а затем использовать инструмент **Move** (переместить) трансформации или нажать клавишу **W**. Теперь вы можете перемещать актер в любом направлении, удерживая левую кнопку мыши и перетаскивая стрелку выбранного направления. Кроме того, удерживая левую кнопку мыши и перетаскивая цветной квадрат, расположенный в якорной точке, между двумя различными направлениями, вы можете двигать актер в двух направлениях одновременно. В UE4 каждая ось ассоциирована с определенным цветом, чтобы упростить использование инструментов:

- ▶ ось X — красный цвет;
- ▶ ось Y — зеленый цвет;
- ▶ ось Z — синий цвет.

ПРИМЕЧАНИЕ

Связи цветов

Обратите внимание, что цвета привязаны к осям координат не только в инструменте виджета трансформации, но и также во многих параметрах и контекстных меню. Например, опции ручной трансформации на панели **Details** имеют ту же цветовую схему.

Трансформация масштабирования

Масштабирование (Scale) позволяет увеличивать и уменьшать размеры актера по осям X, Y или Z. Вы можете совершать это с сохранением пропорций или с различными коэффициентами масштабирования по осям. Когда вы помещаете актер на сцену прямо из панели **Content Browser**, масштаб равен 1 по всем осям. Чтобы изменить масштаб, сначала выберите актер и затем выберите инструмент трансформации **Scale** (масштабировать) или нажмите клавишу **R**. Перемещая любой из ориентированных на направления манипуляторов, вы можете масштабировать актер в любом направлении. Выбрав белый куб в центре, вы можете масштабировать актер пропорционально по всем осям. Наконец, выбрав одну из панелей, соединенных с двумя кубами направлений, вы можете масштабировать актер пропорционально по двум осям.

Трансформация поворота

Поворот (Rotate) — это последний вид трансформации, который вы можете использовать для управления актерами внутри игрового мира. Поворот в UE4 производится так же, как в большинстве 3D-программ: установкой градусов поворота. Полный поворот составляет 360 градусов. Полный поворот может происходить по любой из трех осей: X, Y или Z. Каждая ось связана со специальным термином поворота:

- ▶ ось X: тангаж (pitch);
- ▶ ось Y: рыскание (yaw);
- ▶ ось Z: крен (roll).

Инструмент привязки градусов поворота находится рядом с другими инструментами привязки трансформации. Щелкнув по инструменту **Rotate** (повернуть), вы можете включить или выключить привязку и указать определенные градусы привязки. Например, вы можете повернуть актер и привязать его к повороту в 5 или 30 градусов. Это полезно, когда вы используете модульные наборы для контроля поворотов специфических измерений.

СОВЕТ

Использование интерактивного инструмента трансформации

Вы можете нажимать клавишу пробела, чтобы последовательно переключаться между всеми инструментами трансформации.

Интерактивные и ручные трансформации

Трансформации можно выполнять двумя способами: в интерактивном и ручном режиме.

Интерактивная трансформация — это процесс использования инструментов трансформации перемещения, поворота и масштабирования, чтобы выполнять неточные изменения в виджете пространства мира. Термин *виджет* (widget) означает инструмент, который используется для контроля действий в редакторе. Вы можете использовать эти инструменты для свободного управления актерами в пространстве мира, ориентируясь на визуальное представление без использования точных числовых значений.

Ручная трансформация выполняется с использованием специальных значений или числовых настроек актера на панели **Details**. Этот процесс точнее, и только он подходит для внесения точных изменений.

Мировые и локальные трансформации

Существует два типа систем трансформаций, которые можно использовать, чтобы производить дополнительные изменения актеров: мировой и локальный.

Система трансформации мира указывает всему миру, где находится верх, низ и т. д. Неважно, как актер будет деформирован, повернут или изменен, он будет подчиняться правилам системы мира.

Обратное верно для локальной системы, которая устанавливает правила специально для актеров. Когда актер впервые попадает на сцену, он точно такой же, как и в мировой системе координат. Но что, если вы повернете актер на 15 градусов вправо? В локальной системе координат актер подчиняется новым правилам, по которым все трансформации связаны с произведенным локальным изменением.

Оценка единиц измерения и измерений

Понимание масштабов и измерений пропорций способствует установлению стиля, контекста и непрерывности в мире игры. По умолчанию 1 единица Unreal (иц, Unreal unit) равна 1 сантиметру (см) реального мира. Это важно принять во внимание во всех аспектах дизайна и игры для надлежащего развития окружающей среды, персонажей, эффектов и т. д. В соответствии с настройками по умолчанию, игрок в мире игры, как правило, 6 футов в высоту, что равняется 180 см или 180 иц. Вы можете изменить эту настройку в соответствии с требованиями проекта, но вне зависимости от значения настройки по умолчанию вы можете использовать ее как базовое значение, чтобы установить контекст для высоты других актеров.

Если известно, что персонаж 180 иц в высоту, можно также предположить, что дверь, которой он будет пользоваться, должна быть в высоту в среднем 220 иц и 130 иц в ширину, чтобы персонаж мог пройти через дверной проем во время прохождения уровня. Также можно экстраполировать, что окно будет выглядеть правильно при высоте 180 иц и ширине 110 иц. Размеры других актеров можно рассчитать подобным образом. Такие связи между размерами помогают установить непрерывность всех актеров и гарантировать, что дизайн уровня будет функционировать соответственно геймплею. Некоторые актеры, размеры которых следует определить прежде, чем приступить к конструированию, включают лестницы, окна, двери, потолки, стены и уклоны. Знание основных размеров этих структур до начала конструирования актеров убережет вас от проблем на следующих этапах разработки и защитит от необходимости создавать новые актеры из-за ошибок измерений.

Наконец, хотя эти рекомендации, конечно, полезны при проектировании уровня, вы не должны бояться творческого подхода к проектированию. Понимание правил

масштабов и пропорций поможет вам лучше нарушать правила, чтобы усложнять формы и масштаб, создавая, таким образом, настроение или вовлекая в историю. Вы должны знать, как применяются эти рекомендации для поддержки непрерывности и играбельности, но вы свободны в своем творчестве для удовлетворения требований проекта.

Единицы измерения сетки

Для перемещения актеров в UE4 используется сетка (grid). Сетка в UE4 — это способ разделения на фрагменты фиксированных значений или сегментов карт. Применение сетки позволяет использовать числовые шкалы и устанавливать контекст реального мира в трехмерное пространство. Каждая ячейка сетки эквивалентна установленному числу или значению (см. рис. 3.3). Сетка всегда присутствует, хотя иногда является невидимой. В то время как любая из опций привязки включена, UE4 использует измерения сетки в каждый момент времени поворота или масштабирования как на визуальной сетке, так и вне ее.

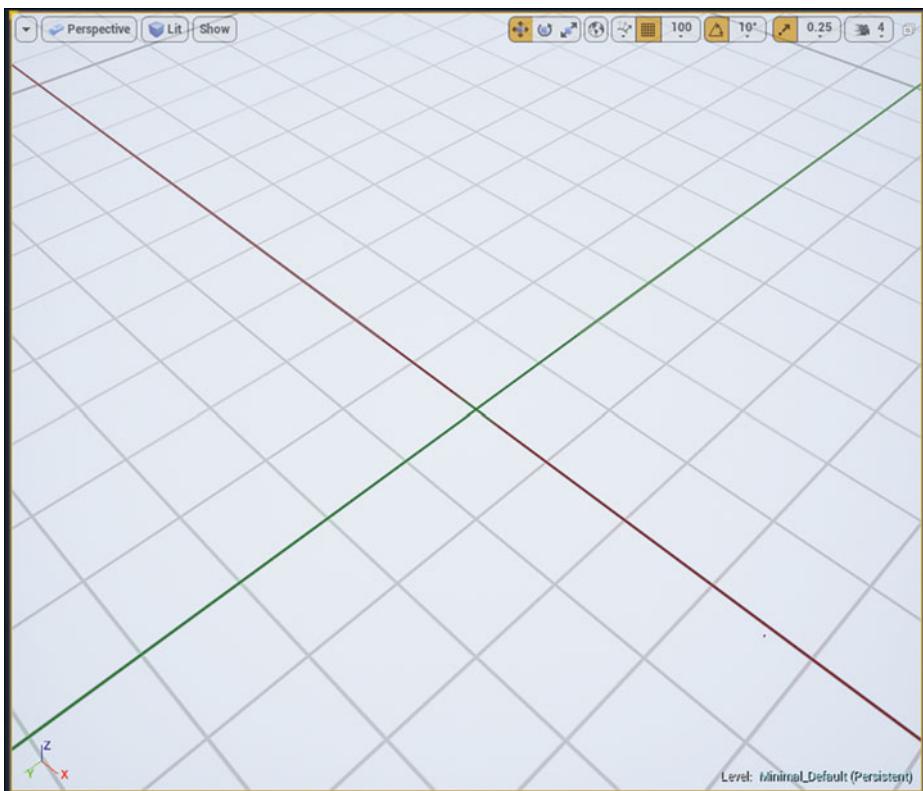


Рис. 3.3. Сетка всегда существует в UE4 и крайне важна для проектирования мира

В UE4 каждая ячейка сетки по умолчанию кратна пяти, чтобы легко было суммировать ячейки сетки для понимания масштабов актера. Если вы поместите актер на сетку, использующую 100 как размер сегмента сетки, и актер занимает 3 единицы сетки в длину, то длина актера равна 300. Инструмент для изменения размера сетки находится вверху панели **Viewport** справа от символа сетки (см. рис. 3.3). Когда вы щелкнете по значению текущей сетки, появится выпадающий список, содержащий доступные размеры единиц сетки.

Привязка сетки

Использование сетки позволяет легко привязывать актеров к заданным координатам или единицам измерения. Система сеток играет ключевую роль в создании актеров, пригодных для многократного использования, и модульных актеров. Существует три вида трансформационных привязок: перемещение, поворот и масштабирование (см. рис. 3.4). Каждая система сеток имеет собственные скалярные параметры, или измерения привязки, которые могут быть установлены вверху экрана предварительного просмотра рядом с соответствующим типом трансформации (см. рис. 3.4).

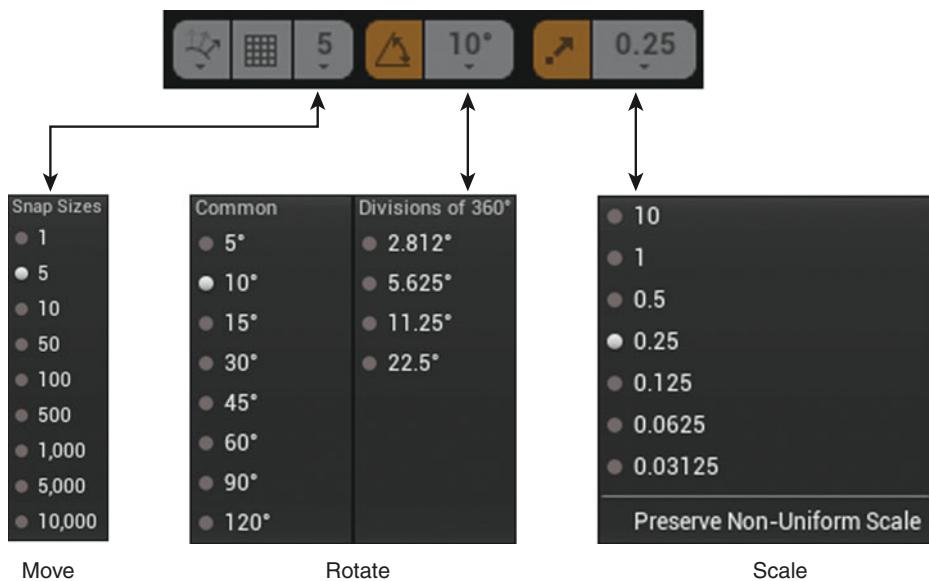


Рис. 3.4. Опции трансформации выюпорта и варианты привязки в выпадающем списке

- ▶ **Сетка перетаскивания** (Drag Grid). Сетка перемещения использует числа, кратные пяти, и связана напрямую со структурой ячеек сетки UE4, которая обсуждалась ранее.

- ▶ **Сетка поворота** (Rotation Grid). Движение поворота основано на скалярных градусах, кратных пяти, использующих основные углы поворота: 5 градусов, 10, 15, 30, 45, 60, 90 и 120. Есть также второстепенное меню, в котором 360 градусов разделены иначе, начиная с 2,812 и далее 5,625, 11,25 и 22,5.
- ▶ **Сетка масштабирования** (Scale Grid). Масштаб меняется, уменьшаясь вдвое от 1 до 0,5, 0,25, 0,125 и т. д.

Вы можете выравнивать значения масштаба для типов сетки, выбрав числовое значение рядом с соответствующим символом. При этом появится выпадающий список с числами, кратными базовому значению. Вы также можете включать и отключать каждую из этих привязок, просто щелкнув по символу трансформации. Символ подсвечивается оранжевым, когда функция включена, либо серым цветом (отключена). В то время как функция отключена, вы можете производить соответствующую трансформацию с актером без привязки к скалярным значениям.

ПРИМЕЧАНИЕ

Модификация настроек привязки

Вы можете включить множество дополнительных полезных инструментов привязки, выбрав пункт **Preferences ⇒ Level Editor** и затем выбрав соответствующие настройки в разделе **Snap**.

Организация сцены

При работе над проектом количество задействованных актеров, как правило, достаточно быстро увеличивается. Поэтому, работаете ли вы в одиночку или в команде, организация сцены крайне важна для обеспечения понимания каждым участником плана проекта и иерархии актеров. Также очень важно, чтобы вам было легко и удобно находить актеры на сцене или уровне. В следующих разделах мы рассмотрим возможности UE4, которые помогут вам сохранить высокую степень организованности при проектировании.

Панель **World Outliner**

Главный инструмент UE4 для эффективной организации актеров — это **World Outliner**. Панель **World Outliner** используется для того, чтобы организовать все аспекты сцены в единое удобное меню (см. рис. 3.5). По умолчанию она появляется в правом верхнем углу экрана, когда вы открываете проект, но вы можете также открыть **World Outliner**, выбрав команду меню **Window ⇒ World Outliner**. Каждый актер в **World Outliner** маркирован данным ему названием или меткой, например название, данное ему пользователем, или название по умолчанию,

которое присваивается актеру после помещения на сцену, а также тип актера, например статичный меш или источник света. Кроме того, рядом с каждым актером расположена иконка, описывающая его тип.



Рис. 3.5. Панель **World Outliner** по умолчанию находится в правом верхнем углу экрана

Одна из важнейших возможностей панели **World Outliner** заключается в том, что она позволяет искать актеры на сцене. Все актеры на сцене перечислены и доступны для поиска в поисковой строке вверху этой панели. Вы можете использовать строку поиска для поиска по всей сцене или искать отдельные типы актеров, находящихся на сцене. Вы также можете исключать слова из поиска, добавляя знак «-» перед поиском по ключевым словам. Например, вы создаете карту с двумя областями, на которой вы маркировали наземные актеры, как **area1_ground** или **area2_ground**. Если вы хотите найти актер, используя для поиска слово **ground**, но хотите найти все наземные актеры без слова **area1**, то можете просто напечатать **ground, -area1** в строке поиска. В результате отобразятся все актеры, содержащие слово **ground**, но без слова **area1**.

Папки

Панель **World Outliner** организована аналогично файловому менеджеру вашего компьютера. В папках хранятся отдельные файлы и группы файлов, и эти папки могут быть организованы и вложены одна в другую. Такая система используется во множестве цифровых организационных систем для поддержки

последовательности, простоты использования и эффективности. Вы легко можете создать и организовать папки в такой системе. В правом верхнем углу панели **World Outliner** находится небольшая иконка, на которой изображен символ плюс на фоне папки. Вы можете щелкнуть по этой иконке, чтобы добавить новую папку в **World Outliner** и дать ей название.

ПОПРОБУЙТЕ САМИ

Создайте новую папку и переместите актеры

Эта рубрика «Попробуйте сами» научит вас создавать папки и перемещать актеры внутри них. Откройте Unreal Editor и выполните следующие шаги

1. Найдите панель **World Outliner**.
2. Щелкните по кнопке **Create a new folder** в правом верхнем углу.
3. Дайте название новой папке.
4. Перейдите во вьюпорт и щелкните по актеру на уровне, чтобы выбрать его. Обратите внимание, что актер, который вы выбрали на сцене, теперь подсвечен в **World Outliner**.
5. Вернитесь в панель **World Outliner**, щелкните и перетащите название актера из **World Outliner** в новую папку.

ПРИМЕЧАНИЕ

Создание новых папок

Если уже был выбран актер на панели **World Outliner**, когда вы создавали новую папку, этот актер автоматически помещается в эту папку.

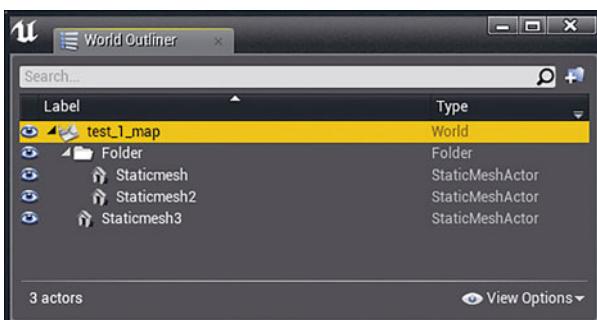


Рис. 3.6. Структура папок на панели **World Outliner**. Обратите внимание, что в папку *Folder* вложены папки *Staticmesh* и *Staticmesh2*, а папка *Staticmesh3* — нет

Как изображено на рис. 3.6, каждая папка на панели **World Outliner** имеет символ глаза рядом с названием. Щелкнув по этому символу, вы можете включить или отключить видимость актеров на сцене. Это полезно, когда во время работы со сценой вам нужно быстро скрыть некоторые актеры или группы актеров, расположенные в папках.

Группировка

Группировка — еще один простой способ быстрой организации аспектов проекта внутри сцены. Группировка похожа на использование папок, в которых выбор множества актеров производится путем выбора отдельно взятого размещенного актера на панели **World Outliner**. Вы можете группировать актеры на сцене, щелкая по одному актеру и удерживая клавишу **Ctrl**, пока щелкаете по другим актерам, которые хотите добавить в группировку. Затем вы можете щелкнуть правой кнопкой мыши по любому из выбранных актеров и выбрать вариант **Group** в контекстном меню. Вы также можете использовать комбинацию клавиш **Ctrl+G**, чтобы сгруппировать выбранные актеры.

Сгруппировав набор актеров, вы можете перемещать, масштабировать и поворачивать их одновременно. Когда вы будете применять перемещение, масштабирование и поворот к группе актеров, помните, что трансформации применяются относительно центра группы актеров. Это важно помнить, если вы решите сгруппировать актеры, находящиеся далеко друг от друга на сцене.

ПРИМЕЧАНИЕ

Группировка актеров

Актер может принадлежать только к одной группе единовременно. Это значит, что вы не можете поместить актер более чем в одну группу.

Некоторые настройки позволяют изменить установки группы. Каждая группа может быть заблокирована или разблокирована. По умолчанию группа создается заблокированной, отчего все ее элементы трансформируются, как одно целое. Чтобы управлять каждым элементом внутри группы, щелкните правой кнопкой мыши (откроется контекстное меню) и выберите в выпадающем списке вариант **Groups** ⇒ **Unlock**. Когда группа разблокирована, вы можете по отдельности управлять актерами внутри группы. Внеся все нужные изменения, вы можете заблокировать группу вновь, щелкнув правой кнопкой мыши по любому актеру, состоящему в группе, и выбрав пункт **Lock Group**. Это вернет ограничение, согласно которому актеры группы ведут себя как один актер.

Слои

Другой способ сохранить проект организованным заключается в системе слоев. В UE4 система слоев аналогична таким системам в 3D-редакторах, таких как Maya или Max. Чтобы получить доступ к панели **Layers**, в главном меню выберите пункт **Window ⇒ Layers**. В этой панели вы можете настроить, какие части сцены группируются в слои и могут включаться и отключаться. Если вы щелкнете правой кнопкой мыши по панели **Layers**, станут доступны все настройки, такие как **New Layer** для создания нового слоя (см. рис. 3.7).

Вы можете добавлять актеры на сцену в слои множеством способов. Один из них заключается в том, чтобы щелкнуть по имени актера на панели **World Outliner**, и затем щелкнуть левой кнопкой мыши и перетащить его в соответствующий слой на панели **Layers**. Другой метод состоит в том, чтобы выбрать все актеры, которые вы хотите добавить в слой, щелкнуть правой кнопкой мыши по этому слою на панели **Layers** и выбрать пункт **Add Selected Actors to Selected Layers** в появившемся контекстном меню. Вы также можете удалять добавленные ранее актеры из слоев: выделите актеры, которые хотите удалить, затем щелкните правой кнопкой мыши по слою на панели **Layers** и выберите пункт **Remove Selected Actors from Layers** в контекстном меню.

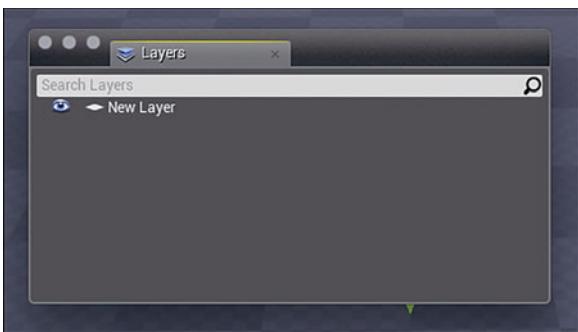


Рис. 3.7. Панель Layers

Чтобы выбрать все актеры в слое, щелкните по нему правой кнопкой мыши и выберите вариант **Select Actors** в контекстном меню. Это мощный способ хранить одинаковые актеры на сцене спаренными вместе и доступными для выбора все одновременно. Используя слои, вы можете помещать все освещение сцены в один слой, все статичные меши — в другой слой, и все эффекты постобработки и частицы — в третий. Вы можете контролировать слои на панели **World Outliner** так же, как контролируете актеров. Вы можете включать и отключать видимость слоев, щелкнув по символу глаза, расположенному рядом с их названиями.

ПРИМЕЧАНИЕ

Различия между группами и слоями

Между группами и слоями существует несколько отличий, но самое большое отличие заключается в возможности помещать один актер во множество слоев. В то время как актер может находиться только в одной группе, этот же актер может одновременно находиться в нескольких слоях. Это отличие усиливает гибкость соединения и выбора актеров.

Связывание

Связывание актеров между собой позволяет создавать иерархические связи между ними. При связывании двух актеров (см. рис. 3.8), один из них становится родительским, а второй — дочерним. Трансформации дочернего актера становятся связанными с родительскими. Это означает, что, когда вы перемещаете, масштабируете и поворачиваете родительский актер, дочерний последует за ним. В то же время изменение дочернего актера не повлечет за собой изменений родительского. Актер может иметь любое количество дочерних актеров, прикрепленных к нему, но при этом актер может иметь только одного родителя. Чтобы прикрепить один актер к другому, выберите актер, который должен стать дочерним, на панели **World Outliner**, щелкнув по его названию, и перетащите его в название актера, который должен стать родительским. Чтобы разорвать связь, в **World Outliner** щелкните и перетащите дочерний актер еще раз в название родительского актера.

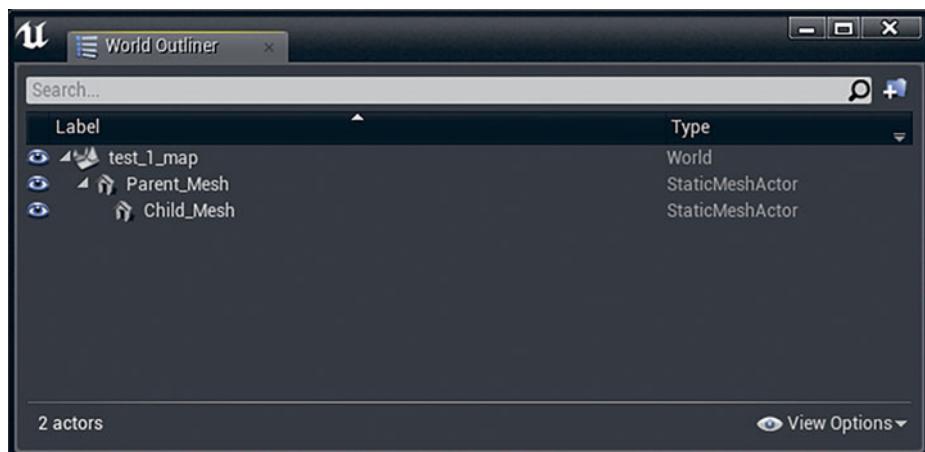


Рис. 3.8. Связанные актеры на панели **World Outliner**

Поскольку трансформации местоположения (Location), поворота (Rotation) и масштабирования (Scale) для актеров по умолчанию имеют настройку **Relative**,

дочерние актеры подражают изменениям, произведенным с родительским актером. Вы можете изменить тип трансформации для местоположения, поворота и масштабирования независимо друг от друга с **Relative** на **World**. Например, возможны ситуации, когда вы захотите, чтобы актер следовал за положением родительского элемента, но не повторял поворот и масштаб. Для этого тип местоположения должен быть **Relative**, а типы поворота и масштаба должны быть установлены в значении **World**. Чтобы изменить тип трансформации местоположения, поворота и масштабирования для дочерних актеров, выберите актер и на панели **Details** в разделе **Transform** щелкните по треугольнику рядом с названиями **Location**, **Rotation** или **Scale** и выберите тип трансформации (см. рис. 3.9).

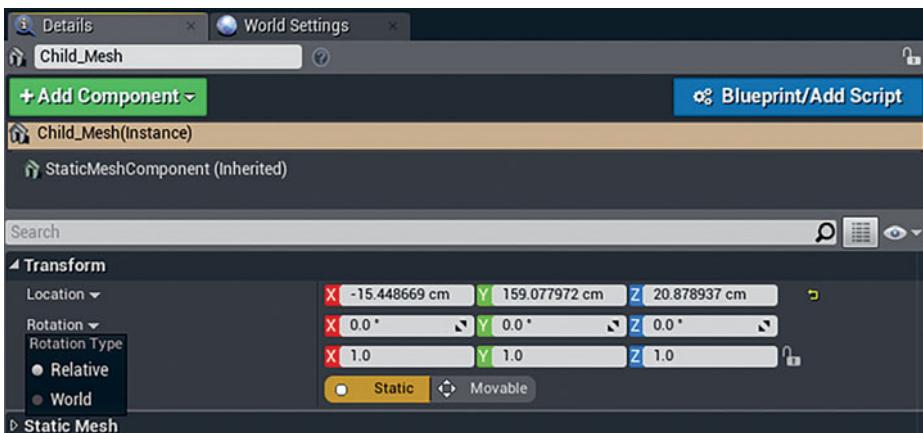


Рис. 3.9. Прикрепленные актеры на панели **World Outliner**

Резюме

В этом часе вы научились использовать инструменты трансформации, чтобы управлять состоянием актеров и видоизменять его внутри сцены. Вы также узнали, как максимально гибко управлять трансформациями. Теперь вы понимаете, как менять настройки по умолчанию, чтобы удовлетворить требованиям проекта. В этом часе вы познакомились с сеткой и использовали привязки UE4, чтобы повысить эффективность и поддерживать точное перемещение и измерения во время манипуляций с актерами на сцене. Наконец, вы узнали, как поддерживать актеры в порядке во время разработки, организуя их в группы и слои. Используя все эти инструменты, теперь вы можете чувствовать себя уверенно в использовании координат, трансформаций и организации в UE4.

Вопросы и ответы

Вопрос: Могу ли я использовать устаревшие измерения Unreal, которая использовала числа 2, 4, 8, 16, 32, 64 и т. д. в UDK?

Ответ: Да, вы можете использовать эту систему, выбрав в главном меню пункты **Edit** ⇒ **Editor Preferences** ⇒ **Level Editor** ⇒ **Viewports** ⇒ **Grid Snapping** и поставив флажок **Use Power of Two Snap Size**.

Вопрос: Как вернуть актер назад к использованию сетки после того, как он был убран с нее?

Ответ: Щелкнув правой кнопкой мыши по актеру, убранному с сетки, выберите пункты **Transform** ⇒ **Snap/Align** ⇒ **Snap Origin to Grid** в контекстном меню.

Вопрос: Как найти актер на сцене после того, как я нашел его на панели **World Outliner**?

Ответ: Выберите актер на панели **World Outliner** и просто нажмите клавишу **F** на панели главной сцены, чтобы сфокусироваться на актере.

Вопрос: Что произойдет, если в группе будет только один актер?

Ответ: Группа автоматически расформируется, и актер снова станет отдельным.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Назовите три типа инструментов трансформации.
2. Чему равняется 1 ии в эквиваленте реального мира?
3. Если вы отключаете отображение сетки, будет ли актер по-прежнему передвигаться по измерениям сетки?
4. Можно ли добавить новый свет в существовавший ранее слой?
5. Может ли актер, находящийся в группе, быть выбран и удален из этой группы?

Ответы

1. Перемещение (Move), поворот (Rotate) и масштабирование (Scale) — это три типа инструментов трансформации.

2. 1 ии равняется 1 см.
3. Да, даже если сетка отключена, актер по-прежнему будет двигаться, как если бы он был привязан к сетке.
4. Да, вы можете добавить новое освещение в слой, выбрав актер, щелкнув по нему правой кнопкой мыши и выбрав в выпадающем списке вариант **Add to Layer** (добавить в слой).
5. Да, щелкните правой кнопкой мыши по группе и выберите пункт **Unlock** (разблокировать), затем выберите актер, который хотите исключить из группы. Удалив актер, вновь заблокируйте группу.

Упражнения

В этом упражнении вы откроете редактор и сделаете несколько изменений в том, как вы управляете скалярными параметрами актера. Затем вы поместите этот актер в группу и в различные слои, чтобы управлять его видимостью. Знание элементов управления для выполнения всех этих задач позволяет гибко управлять всеми трансформациями вашего актера в редакторе. Также важно понимать, как организовывать актеры, чтобы получить максимальный контроль над организацией проекта и структурой групп и слоев.

1. Поместите любой актер на уровень.
2. Увеличьте объект по оси X на 15.
3. Поверните актер на 25 градусов вправо.
4. Переместите его на 12 ии по оси Y.
5. Переместите актер на 140 ии по оси Z.
6. Поместите второй актер на уровень.
7. Объедините два актера в группу.
8. Разблокируйте группу, выберите второй актер и исключите его из группы.
9. Переместите оставшийся актер в новый слой. Назовите слой **Newtestlayer**.

4-Й ЧАС

Работа с актерами статичных мешей

Что вы узнаете в этом часе.

- ▶ Знакомство с редактором статичных мешей
- ▶ Импорт файлов 3D-моделей
- ▶ Назначение материалов и оболочки коллизий ассетам статичных мешей
- ▶ Размещение актеров статичных мешей
- ▶ Изменение связей мешей и материалов в актерах статичных мешей
- ▶ Установка реакций на коллизии актеров статичных мешей

Статичные меши (Static Meshes) являются самыми распространенными художественными ассетами и типами актеров, с которыми вы будете работать в UE4.

Статичные меши — это 3D-модели, импортированные из таких приложений, как 3DS Max или Maya. Они обычно используются для создания внешнего вида и построения мира. Практически на каждом уровне, который вы будете создавать, вам потребуются статичные меши. В этом часе вы познакомитесь с импортом 3D-моделей, использованием редактора статичных мешей, редактированием оболочек коллизий, узнаете основные элементы работы с ассетами и актерами статичных мешей, а также способы назначения им материалов.

ПРИМЕЧАНИЕ

Подготовка к практике

Создайте новый проект с шаблоном **Third Person** и стартовым контентом.

Ассеты статичных мешей

Ассеты статичных мешей имеют следующие атрибуты: якорную точку (локальная ось), вершины, грани и полигоны, определяющие внешний вид модели, а также уровни детализации (LODs, Levels of detail). Ассеты статичных мешей также имеют

оболочки коллизий, сокеты и UV-развертки, используемые для материалов, текстур и карт освещения. Чем лучше вы будете понимать, как работают атрибуты ассетов и актеров статичных мешей, тем проще для вас будут понятия в последующих уроках.

ПРИМЕЧАНИЕ

Уровни детализации

Уровни детализации — это просто версии меша в различных полигональных разрешениях. Чем дальше меш от камеры, тем меньше полигонов требуется для отображения модели. Это эффективный способ обеспечения стабильно высокой частоты кадров во время игры.

Редактор статичных мешей

Редактор статичных мешей (**Static Mesh Editor**) позволяет редактировать, изменять и устанавливать основные свойства ассетов статичных мешей, хранящихся на панели **Content Browser**. Редактор статичных мешей состоит из панели меню, панели инструментов для включения и отключения отображения элементов, вьюпорта, позволяющего просматривать меш, панели **Details** для редактирования и изменения свойств меша, панели **Socket Manager** для добавления и редактирования сокетов, а также панели **Convex Decomposition Hull** для создания уникальных оболочек коллизий (см. рис. 4.1).

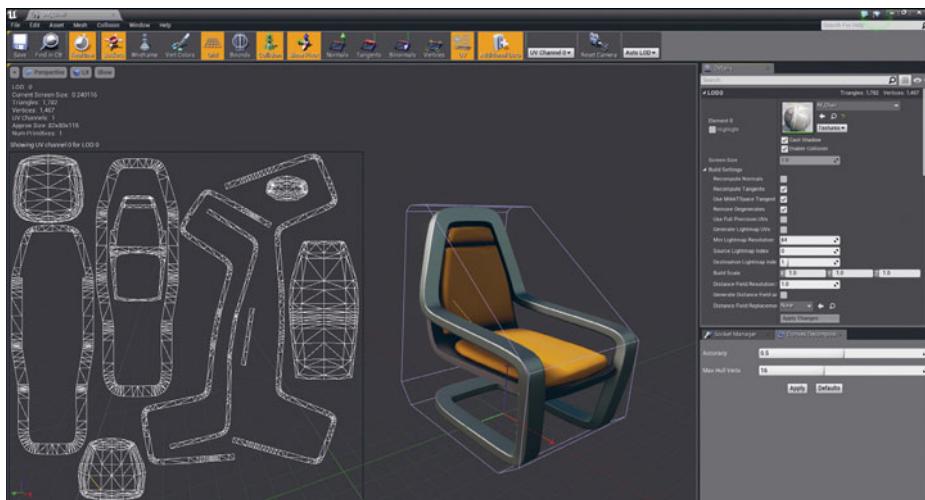


Рис. 4.1. Интерфейс редактора статичных мешей

Как открыть окно редактора статичных мешей

Чтобы просмотреть статичный меш в редакторе статичных мешей, дважды щелкните левой кнопкой мыши по его ассету на панели **Content Browser**, и редактор откроется в новом окне. Каждый ассет статичного меша, по которому вы дважды щелкнете, откроется в собственном окне редактора статичных мешей.

ПОПРОБУЙТЕ САМИ

Используйте Static Mesh Editor, чтобы просмотреть ассет статичного меша

Откройте редактор статичных мешей для существующего ассета статичного меша, следуя приведенным ниже инструкциям.

1. В лаунчере откройте проект, которые вы создали в 1-м часе «Введение в Unreal Engine 4».
2. На панели **Content Browser** найдите папку **StarterContent**. Если вы не добавляли стартовый контент в проект, щелкните по зеленой кнопке **Add New** и выберите пункт **Add Feature or Content Pack**. Во всплывающем окне выберите вкладку **Content Pack**, выделите вариант **Starter Content** и щелкните по кнопке **Add to Project**.
3. В папке **StarterContent** найдите папку **Props** (см. рис. 4.2).

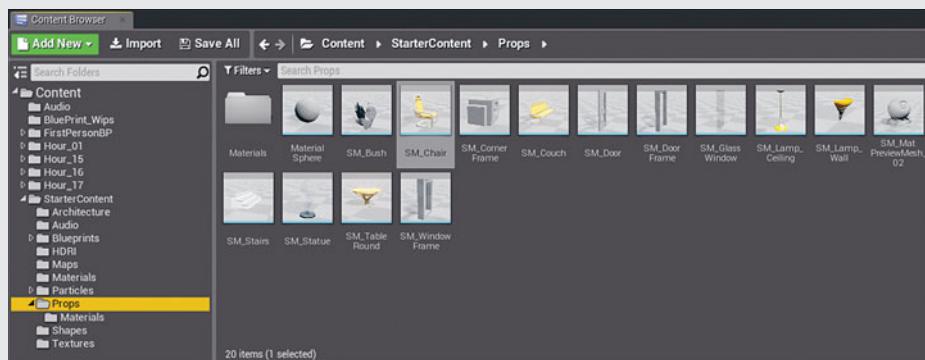


Рис. 4.2. Поиск ассета статичного меша на панели Content Browser. Левая панель: **Source**. Правая панель: область управления ассетами

4. В области управления ассетами (правая часть панели **Content Browser**) дважды щелкните по любому ассету статичного меша, чтобы открыть его в редакторе статичных мешей.
5. Попробуйте переместить вышупорт редактора статичных мешей, а также включать и выключать опции отображения на панели инструментов.

ПРИМЕЧАНИЕ

Использование вьюпорта редактора статичных мешей

Вьюпорт редактора статичных мешей работает по тому же принципу, что и вьюпорт основного редактора. Нажмите клавишу F, чтобы сфокусировать отображение на меше, и затем, зажав клавишу Alt и левую кнопку мыши, перетаскивайте меш, чтобы повернуть его.

Импорт статичных мешей

Для импорта 3D-моделей в редактор обычно используются файлы с расширениями *.obj* и *.fbx*. При импорте обоих типов файлов вы можете открыть окно FBX Import Options (Настройки импорта FBX-файлов) и настроить дополнительные параметры. Если у вас нет достаточного опыта работы с 3D-моделями, параметры, на которых следует сосредоточиться на данный момент, включают автоматическую генерацию коллизии (**Auto Generate Collision**), импорт материалов (**Import Materials**) и импорт текстур (**Import Textures**). Все они выбраны по умолчанию. Автоматически генерируемые оболочки коллизий и UV-развертки карт освещения, будучи импортированными, могут ускорить работу, и вы можете редактировать и изменять оболочки коллизий и UV-развертки карт освещения в редакторе статичных мешей.

ПОПРОБУЙТЕ САМИ

Импорт ассета статичного меша

Выше вы узнали, как открыть редактор статичных мешей для ассета. Следуйте инструкциям ниже, чтобы импортировать новый статичный меш в проект.

1. Откройте проект, который вы создали в 1-м часе.
2. Выберите на панели **Content Browser** папку или создайте новую для хранения ассета статичного меша.
3. Щелкните правой кнопкой мыши по папке **Content** и выберите пункт **New Asset** ⇒ **Import To** или щелкните по кнопке **Import** в навигационной панели вверху панели **Content Browser**.
4. В диалоговом окне **Import** найдите файлы ассета **Archway** с расширением *.obj* или *.fbx* в папке *Hour_04/Models* (которую можно скачать по адресу: http://addons.eksмо.ru/it/UE_24.zip) и дважды щелкните по одному из них, или выберите файл и щелкните по кнопке **Open**. (Вы можете пропустить этот шаг, просто перетащив файл в панель обзора ассетов панели **Content Browser** напрямую или из файлового менеджера вашей операционной системы). После того как файл был выбран и открыт, отобразится меню **FBX Import Options**.

- Убедитесь, что в появившемся окне **FBX Import Options** (см. рис. 4.3) установлен флашок **Auto Generate Collision**.

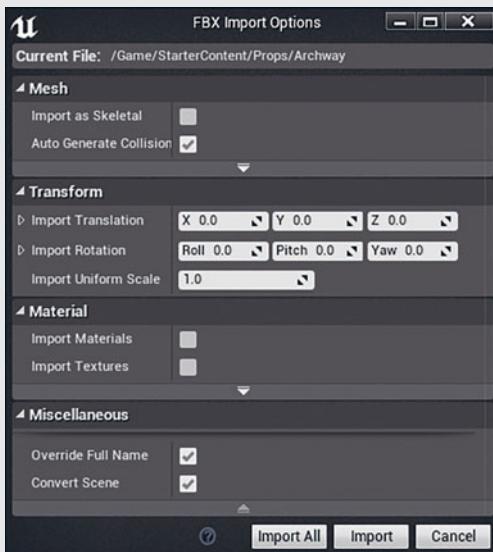


Рис. 4.3. Окно **FBX Import Options**

- Щелкните по кнопке **Import**.
- После того как меш был импортирован, его следует сохранить. На панели **Content Browser** щелкните правой кнопкой мыши по изображению ассета и щелкните по кнопке **Save** из выпадающего списка.

СОВЕТ

Якорные точки

Когда вы импортируете ассет меша, якорная точка статичного меша определяется позицией модели относительно координат мира в трехмерном пространстве, из которого он был импортирован, а не локальной осью модели. В окне **FBX Import Options** на вкладке **Transform** вы можете поменять позиционирование меша относительно якорной точки, изменения значения позиции, поворота и масштаба.

Просмотр UV-разверток

Прежде чем материал должным образом отобразится на поверхности модели, он нуждается в слое UV-карты, также известном как UV-канал. Если модель была корректно создана в приложении 3D-моделирования, она имеет по меньшей мере

один UV-канал. Статичные меши могут иметь несколько UV-каналов. Обычно у них как минимум один UV-канал для материала (**UV Channel 0**) и один — для данных карты освещения (**UV Channel 1**). Чтобы просмотреть UV-каналы статичного меша, щелкните по кнопке **UV** на панели инструментов и выберите в раскрывающемся списке пункт **UV Channel 0** или **UV Channel 1**, чтобы включить отображение соответствующего канала, или **None** — чтобы выключить (см. рис. 4.4).

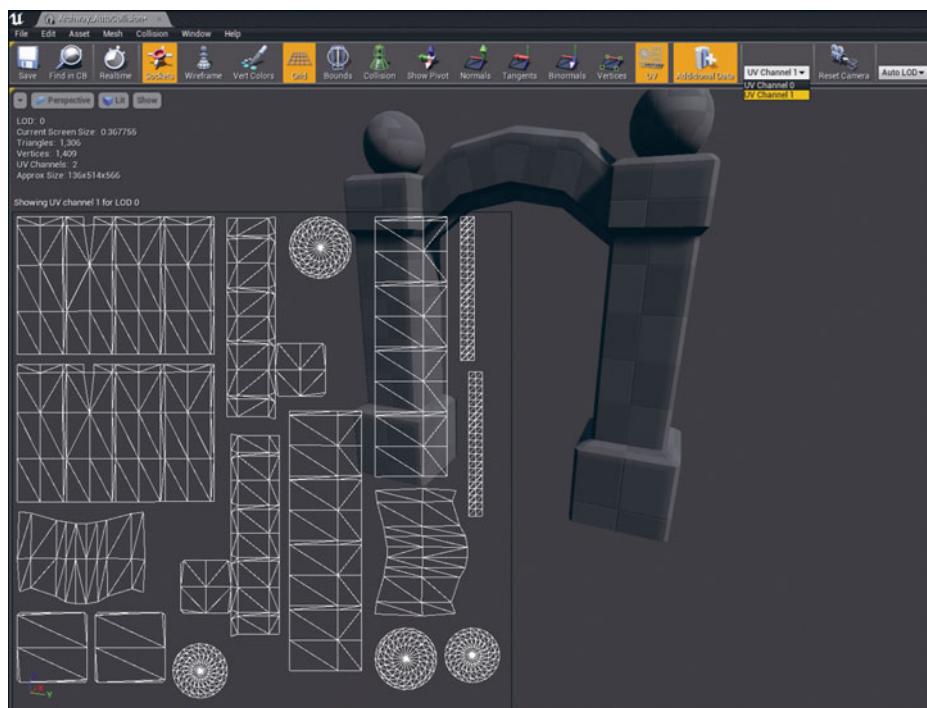


Рис. 4.4. UV-каналы меша отображаются во виджете редактора статичных мешей

ПРИМЕЧАНИЕ

UV-канал карты освещения

UV-канал карты освещения используется для хранения информации об освещении и тенях на поверхности меша. Редактор автоматически генерирует UV-канал карты освещения в процессе импорта, но вы также можете создать его после импорта, используя панель **Details** редактора статичных мешей. Кarta освещения UV-канала по умолчанию имеет номер 1, что означает вторую карту, поскольку индекс первой карты — 0. Кроме того, технически вы можете использовать любой UV-канал для карты освещения, однако новичку лучше использовать настройки по умолчанию.

Назначение материала ассету статичного меша

Поскольку ассет статичного меша — это базовый ассет, следует подготовить его для дальнейшего использования. Например, можно назначить ему материал по умолчанию, чтобы при каждом помещении ассета на уровень в качестве актера статичного меша он уже имел материал, который при желании можно заменить.

ПОПРОБУЙТЕ САМИ

Назначение материала статичному мешу

Чтобы назначить материал ассету статичного меша, выполните следующие шаги.

1. Откройте один из мешей в редакторе статичных мешей.
2. Выберите панель **Details** и найдите строку **Material Slots**.
3. Щелкните по названию выбранного материала справа от иконки материала и выберите материал из выпадающего списка. Вы также можете перетащить материал из панели **Content Browser** в иконку, чтобы изменить присвоение материала.
4. Щелкните по кнопке **Save** на панели инструментов редактора статичных мешей, чтобы сохранить изменения, произведенные с ассетом статичного меша.

Оболочки коллизий

Оболочка коллизии (collision hull) — простой примитивный контур, окружающий меш и используемый для определения событий коллизий. Событие коллизии (collision event) происходит, когда две оболочки коллизий актеров сталкиваются, соприкасаются или проникают друг в друга. При импорте 3D-модели редактор автоматически генерирует два варианта оболочек коллизий: простую и комплексную.

Просмотр оболочек коллизий

Вы можете просмотреть оболочку коллизии в редакторе статичных мешей, щелкнув по иконке **Collision** на панели инструментов редактора статичных мешей (см. рис. 4.5.).



Рис. 4.5. Панель инструментов редактора статичных мешей с кнопкой **Collision** и выпадающим списком

При этом вы сможете взаимодействовать с оболочкой коллизии, щелкнув по одной из граней каркаса. Можно перемещать, масштабировать и поворачивать оболочку, просто зажав клавишу пробела, чтобы переключаться между виджетами трансформации (см. рис. 4.6.). Вы можете удалить выбранную оболочку коллизии, нажав клавишу **Delete**.

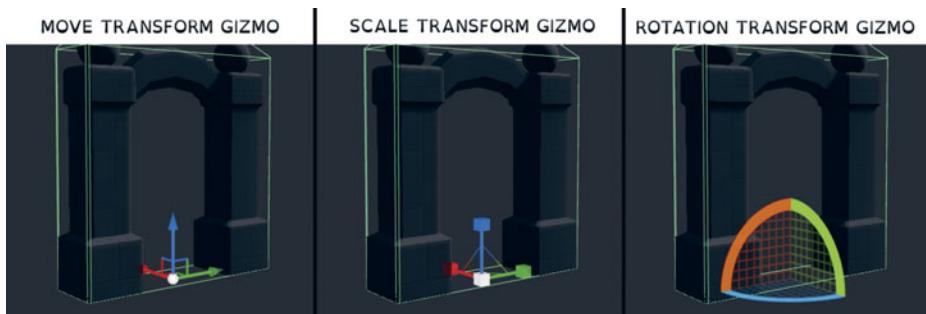


Рис. 4.6. Виджеты трансформации перемещения, масштабирования и поворота. Ось X — красная, Y — зеленая, Z — синяя

Редактирование оболочек коллизий

Автоматически генерируемые в процессе импорта оболочки коллизий могут сэкономить время, но они зависят от формы ассета меша, отчего автоматическая генерация не всегда уместна. На рис. 4.7 можно увидеть, что автоматически сгенерированная простая коллизия выходит за пределы всего меша **Archway**. Эта оболочка коллизии будет блокировать перемещение актеров через меш, однако она также не позволит актерам пройти через арку. В нашем случае оболочку коллизии необходимо изменить.

ПРИМЕЧАНИЕ

Упрощенная коллизия

Вы можете добавить несколько оболочек коллизий в один статичный меш. Чтобы добавить новую оболочку, щелкните по кнопке **Collision** в строке меню и выберите один из вариантов **Add нужная коллизия**. Можете добавить столько коллизий, сколько хотите. Если вы хотите удалить оболочку коллизии, выберите ее во вьюпорте и нажмите клавишу **Delete**. Если вы хотите удалить все оболочки коллизий из вьюпорта, выберите пункт **Collision ⇒ Remove Collision** в строке меню.

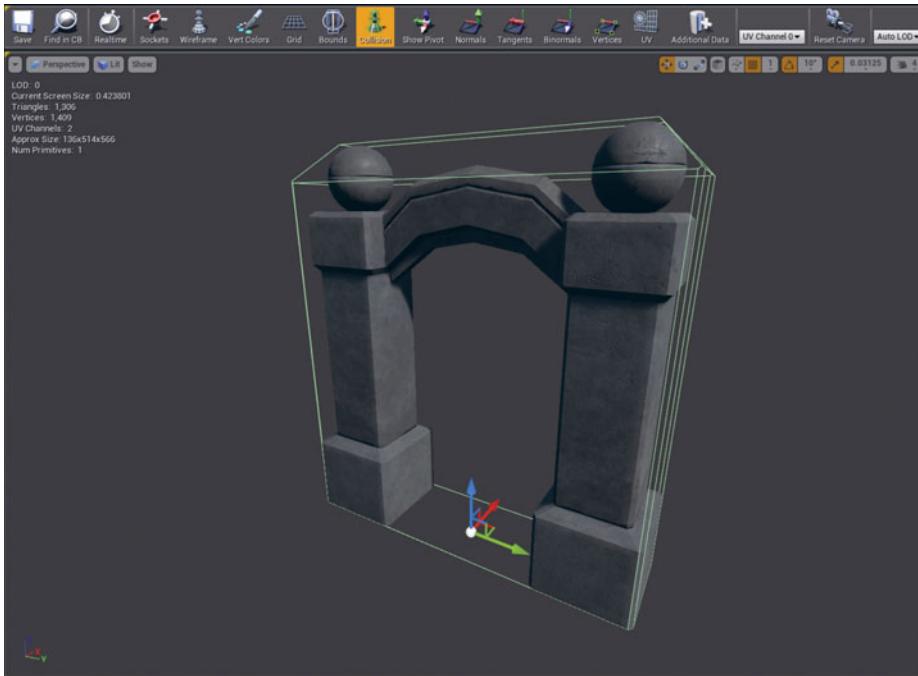


Рис. 4.7. Автоматически сгенерированная простая оболочка коллизии

ПОПРОБУЙТЕ САМИ

Работа с оболочками коллизий

Настало время попрактиковаться в редактировании оболочек коллизий. Попытайтесь отредактировать ассет **Archway**, который находится в папке *Hour_04* (доступной по адресу: http://addons.eksмо.ru/it/UE_24.zip).

1. Откройте меш **Archway** в редакторе статичных мешей. (Можно использовать ассет, импортированный ранее, либо скопировать ранее импортированный ассет на панели **Content Browser**, либо просто импортировать ассет еще раз.)
2. Выберите в строке меню пункт **Collision** ⇒ **Remove Collision**.
3. Выберите там же пункт **Collision** ⇒ **Add Box Simplified Collision** (см. рис. 4.8).
4. Выберите оболочку коллизии во вьюпорте и нажимайте клавишу пробел, чтобы переключаться между перемещением, масштабированием и поворотом, чтобы поместить упрощенную прямоугольную оболочку коллизии по вершине арки.
5. Повторите шаги 3 и 4 еще два раза для каждой из колонн арки.

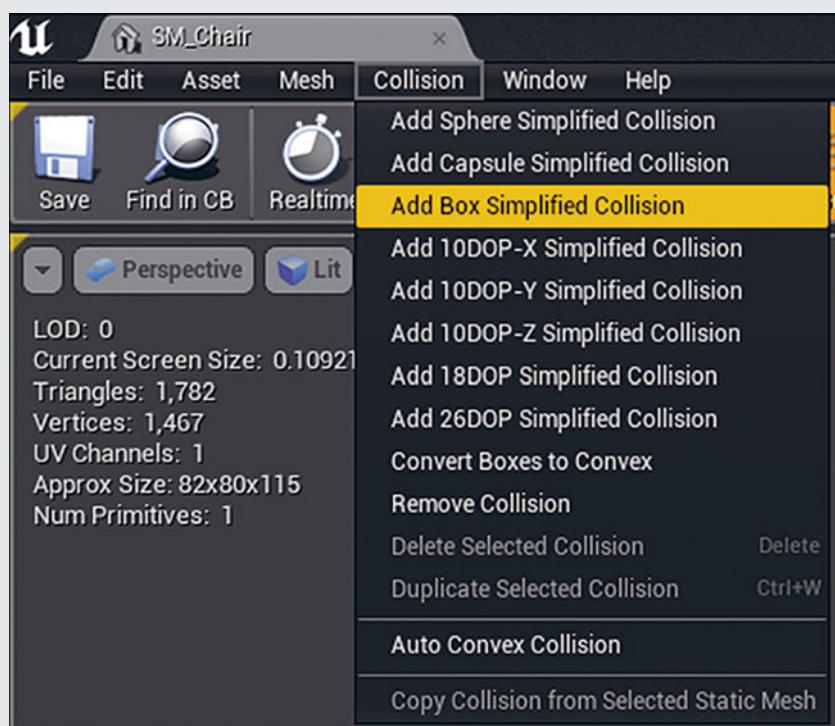


Рис. 4.8. Выбор пункта Collision ⇒ Add Box Simplified Collision в строке меню редактора статичных мешей

6. Щелкните по кнопке **Save** на панели инструментов, чтобы сохранить изменения, произведенные с ассетом статичного меша, на панели **Content Browser**. В итоге ваша коллизия должна выглядеть, как на рис. 4.9.

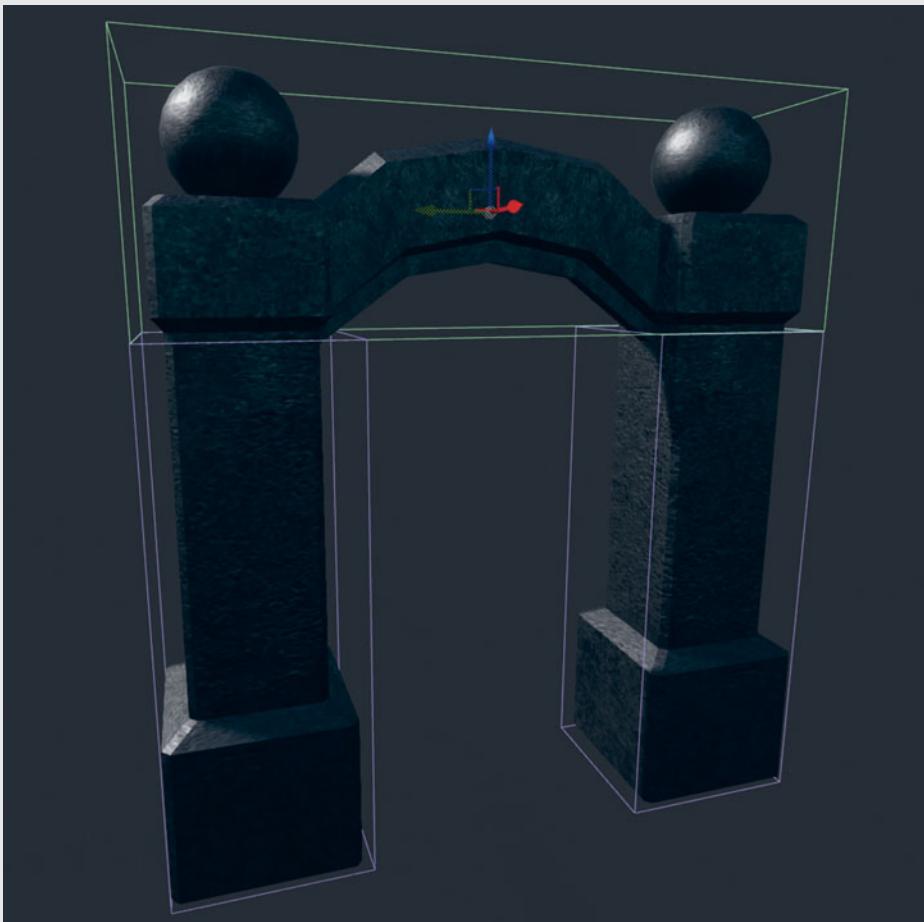


Рис. 4.9. Установленные вручную оболочки коллизий

Панель Convex Decomposition

Оболочки коллизий представляют собой упрощенные выпуклые примитивы. Редактор статичных мешей содержит панель **Convex Decomposition**, которая позволяет автоматически генерировать оболочки коллизий для более сложных моделей. Изменение сложности статичного меша и настроек декомпозиции приводит к различным результатам (см. рис. 4.10). Чтобы открыть панель **Convex Decomposition**, выберите пункт **Collision ⇒ Auto Convex Collision** в строке меню.

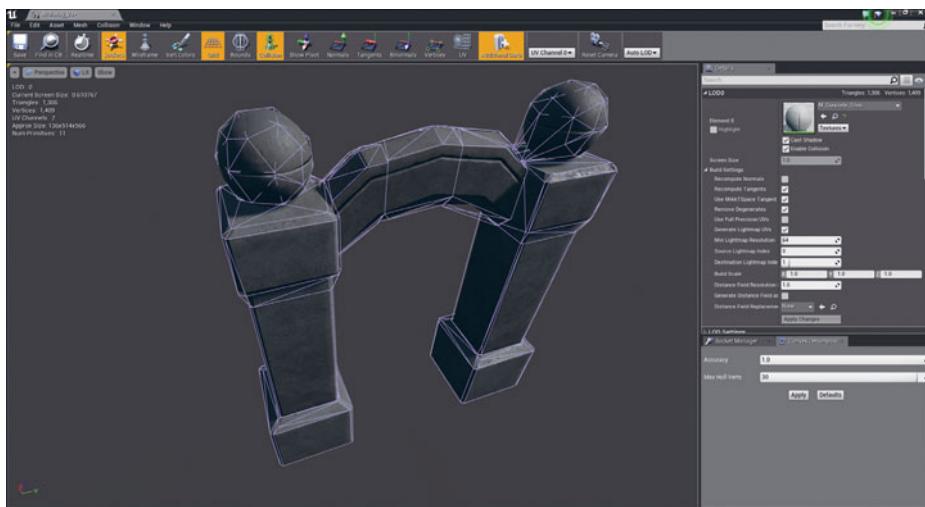


Рис. 4.10. Оболочки коллизий могут быть сгенерированы при помощи панели **Convex Decomposition**

Многополигональная коллизия

Вы также можете установить ассеты статичных мешей в многополигональную коллизию (per-poly collision), которая является наиболее точной из всех коллизий. Такая коллизия влечет наибольшую вычислительную нагрузку, поэтому вам стоит прибегать к ней только в особых случаях, когда требуется точность. На панели **Details** найдите раздел **General Settings** и установите параметр **Collision Complexity** в значении **Use Complex Collision as Simple** (см. рис. 4.11). В этом случае редактор будет использовать комплексную многополигональную коллизию вместо простого выпуклого примитива.

ПРЕДУПРЕЖДЕНИЕ

Обнаружение многополигональной коллизии

Если каждый актер статичного меша на экране станет обрабатывать обнаружение многополигональной коллизии во время игры, это потребует существенных вычислительных ресурсов и может привести к быстрому снижению частоты кадров.

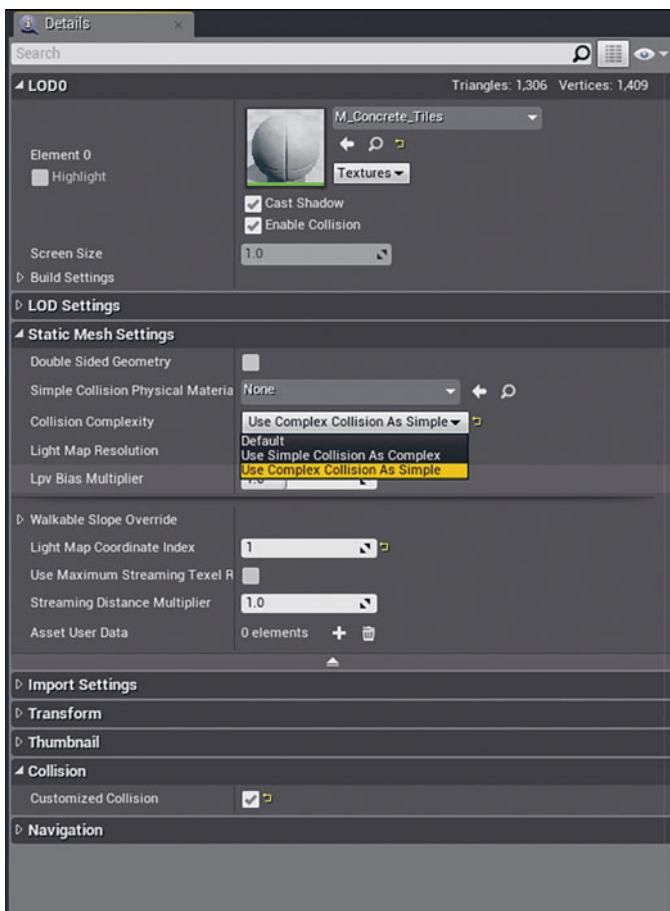


Рис. 4.11. Параметры настройки для многогранниковой коллизии

Актеры статичных мешей

Оставшаяся часть этого часа будет сфокусирована на работе с актерами статичных мешей. Актеры статичных мешей (Static Mesh Actors) — это помещенные на уровень экземпляра ассетов статичных мешей. Каждый размещенный актер статичного меша имеет собственные свойства, которые могут быть изменены независимо от связанного с ним ассета статичного меша. Изменение свойств ассета статичного меша влияет на все связанные с ним актеры статичных мешей. Например, если вы полностью удаляете оболочки коллизий из ассета меша, все актеры, использующие этот статичный меш, не смогут генерировать реакции на коллизии. При этом изменение свойств и настроек актера статичного меша не повлияет на исходный ассет.

Помещение актеров статичных мешей на уровень

Выше вы освоили процесс импорта и редактирования оболочек коллизий ассетов статичных мешей. Давайте поместим актеры статичных мешей на уровень. Чтобы поместить статичный меш на уровень, найдите его на панели **Content Browser**. (Можете использовать меш, который импортировали ранее, или импортировать новый меш и поместить его на уровень.) Чтобы поместить меш на текущий уровень, зажмите левую кнопку мыши и перетащите ассет из панели **Content Browser** на текущий уровень во вьюпорт.

Перетащив статичный меш на уровень, вы создали актер статичного меша, который ссылается на исходный ассет меша. Если выбрать помещенный на уровень актер статичного меша, его контур будет обведен и вы увидите, что в якорной точке появился виджет трансформации. На панели **Details** редактора уровней видно, что теперь актер имеет свойства трансформации, хранящие информацию о позиции, угле поворота и масштабе в мире (см. рис. 4.12).

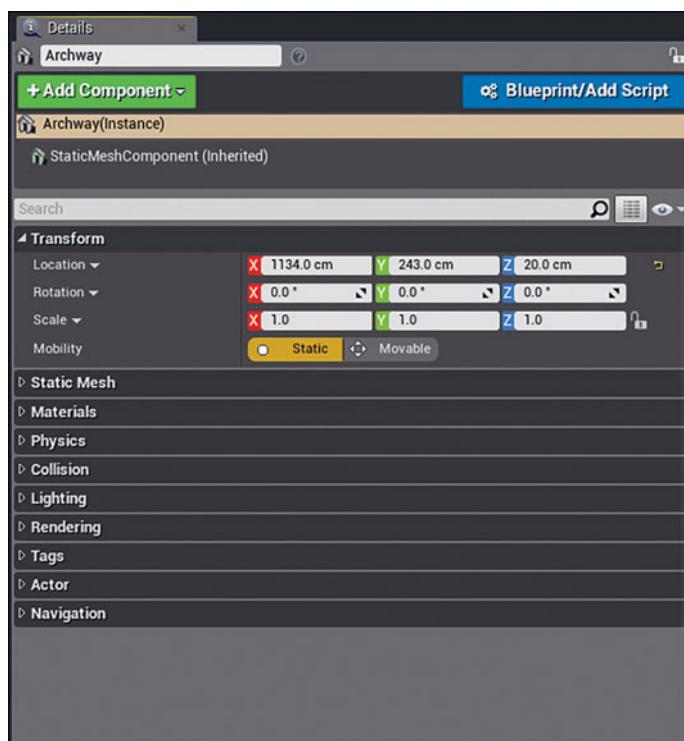


Рис. 4.12. Настройки трансформации актера статичного меша на панели **Details** основного редактора

СОВЕТ

Копирование актеров статичного меша

Если необходимо быстро скопировать размещенный актер статичного меша, удерживайте клавишу **Alt** и переместите или поверните актер с помощью виджета трансформации во вьюпорте уровня.

Настройки мобильности

В разделе **Transform** панели **Details** видно, что актер статичного меша имеет три возможных состояния мобильности: статичное (**Static**), стационарное (**Stationary**) и подвижное (**Movable**). Изменение состояния мобильности актера сильно повлияет на то, как UE4 будет обрабатывать информацию об освещении и тенях для актера. Статичное состояние говорит движку, что освещение должно быть рассчитано заранее. Подвижное же состояние указывает движку рассчитывать освещение в процессе игры. Если вы хотите анимировать актер статичного меша или симулировать физику, то должны установить мобильность в значение **Movable**. Чтобы поменять состояние мобильности актера статичного меша просто щелкните по одному из вариантов **Static**, **Stationary** или **Movable** под свойствами трансформации в разделе **Transform**.

ПРИМЕЧАНИЕ

Статичный меш: двигать или нет, вот в чем вопрос

Термин статичный меш может быть немного обманчивым. В данном случае статичный относится к базовому состоянию. По умолчанию статичный меш также называется статичным, потому что не будет двигаться в процессе геймплея. Если меш не будет двигаться, вы можете произвести просчет (заранее) данных об освещении, чтобы целевая платформа не беспокоилась о расчете освещения в каждом кадре, а могла просто загрузить и отобразить заранее рассчитанные данные об освещении на поверхности помещенного актера.

Изменение ссылки на меш для актера статичного меша

Как видно из рис. 4.13, актер статичного меша ссылается на исходный ассет статичного меша, который использовался при помещении актера на уровень. Можно легко поменять ссылку на меш, перетащив новый ассет меша из панели **Content Browser** в иконку ссылки на статичный меш на панели **Details** выделенного актера статичного меша или щелкнув по стрелке рядом с текущим присвоенным мешем и выбрав в выпадающем списке новый меш. При смене меша вы увидите обновление во вьюпорте уровня. Поскольку актер статичного меша хранит данные о трансформации в мире, новый присвоенный ему меш перенимает эти свойства.

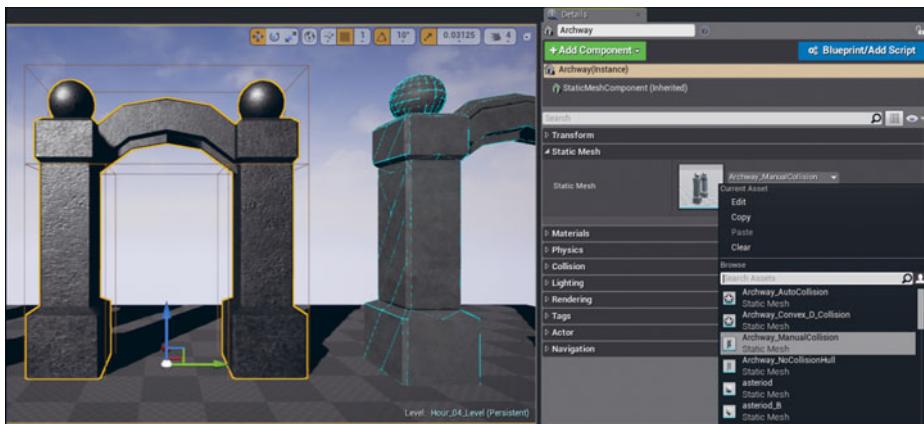


Рис. 4.13. Ссылка на меш актера статичного меша

Смена материала актера статичного меша

Актер статичного меша также имеет ссылку на материал, основанную на том, какой материал был присвоен ассете меша в редакторе статичных мешей. Вы можете изменять материал для каждого актера. На рис. 4.14 на панели **Details** основного редактора показан текущий материал выбранного актера статичного меша. Вы можете перетащить новый материал из панели **Content Browser** в изображение текущего материала на панели **Details**, либо щелкнуть по стрелке рядом с текущим материалом и выбрать новый из выпадающего списка, либо просто зажать левую кнопку мыши и перетащить новый материал из панели **Content Browser** в любой актер статичного меша прямо на уровне.

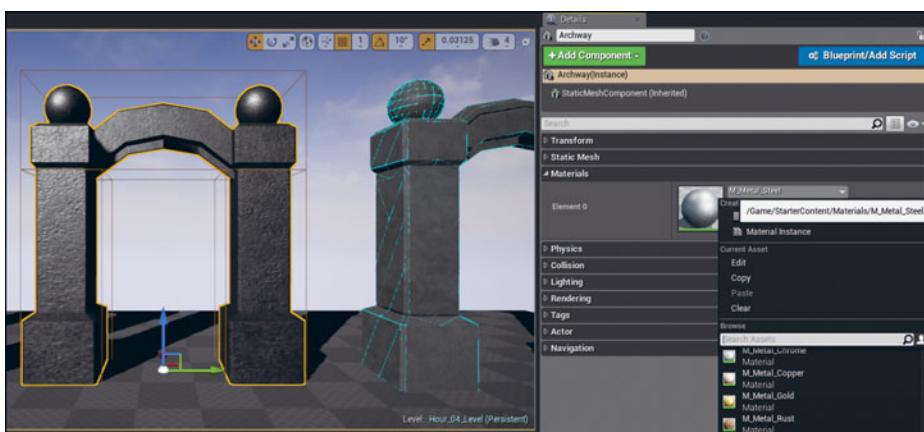


Рис. 4.14. Ссылка на материал актера статичного меша

Редактирование реакций на коллизии актера статичного меша

Вы уже знаете, как редактировать оболочки коллизий ассетов статичных мешей в редакторе статичных мешей. Теперь вы готовы к тому, чтобы изменять реакции на коллизии актеров статичных мешей, помещенных на уровень. Выбрав актер статичного меша на уровне, вы можете найти настройки коллизии актера на панели **Details** основного редактора. Пока можете сфокусироваться только на пресетах коллизий, реакциях на коллизии и типах объектов. Чтобы просмотреть оболочки коллизий актера во видпорте редактора уровней, нажмите комбинацию клавиш **Alt+C**, что включает и отключает отображение оболочек коллизий. На рис. 4.15 показан видпорт уровня с включенным отображением оболочек коллизий.

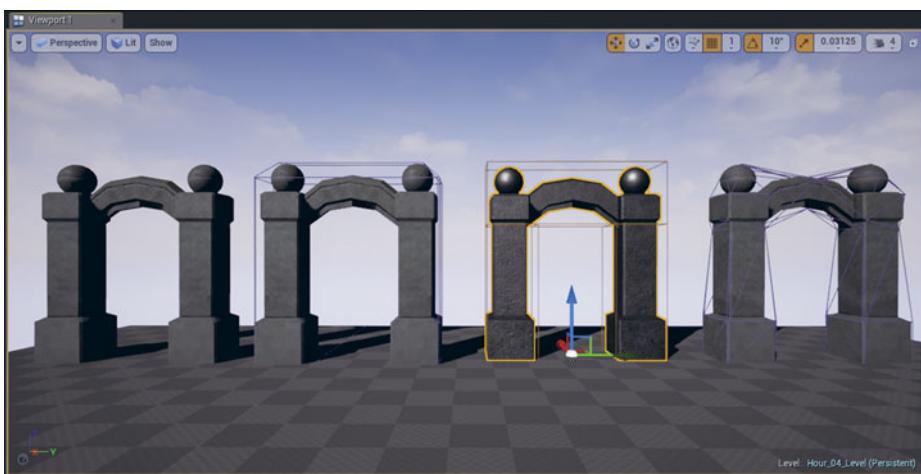


Рис. 4.15. Видпорт редактора уровней с четырьмя актерами статичных мешей, каждый из которых связан с различными ассетами статичных мешей с разными оболочками коллизий, и с включенным отображением коллизий

Пресеты коллизий

Щелкнув по треугольнику слева от настройки **Collision Presets** на панели **Details** основного редактора, разверните окно с дополнительными настройками. Пресеты* определяют основные настройки реакций на коллизии актеров с различными типами объектов. Чтобы разблокировать настройки, выберите вариант **Custom** (пользовательский) для настройки **Collision Presets**. Вы увидите, что появилось гораздо больше настроек, включая **Collision Responses** и **Object Type**. На рис. 4.16 показаны настройки коллизий.

* Пресет — набор предварительных настроек. — Прим. ред.

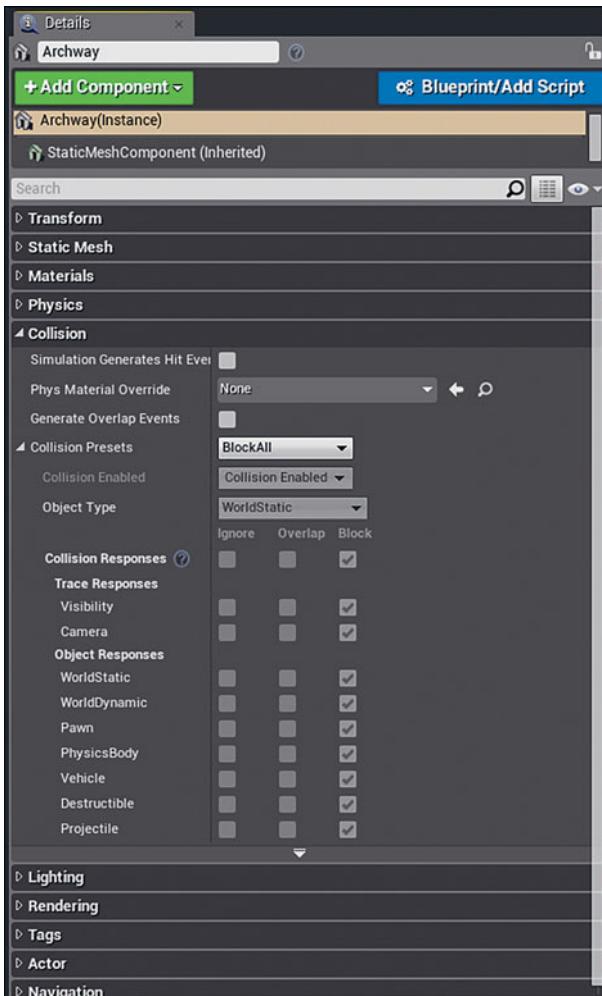


Рис. 4.16. Настройки коллизий актеров статичных мешей на панели Details основного редактора

СОВЕТ

Пользовательские пресеты коллизий

Вы можете создать собственные пресеты коллизий, выбрав пункт **Edit ⇒ Project Settings** в главном меню редактора уровней. Но на начальных этапах работы с UE4, проще выбрать пользовательский вариант и изменить реакцию на коллизии для каждого актера прямо в выбранном актере статичного меша.

Параметр Collision Enabled

Параметр **Collision Enabled** на панели **Details** позволяет включать и выключать коллизии для выбранного актера. Когда коллизии полностью отключены, даже если соответствующий ассет статичного меша для этого актера имеет оболочку коллизии, актер не будет обрабатывать взаимодействия и события коллизий.

Параметр Object Type

Параметр **Object Type** на панели **Details** позволяет устанавливать, к какому типу объектов принадлежит этот актер, чтобы при столкновении с другими актерами, они знали, как реагировать. Например, если актер статичного меша имеет настройку **Static** в разделе **Transform**, то его тип объекта должен быть **WorldStatic**, но если в разделе **Transform** был установлен вариант **Movable**, то тип объекта должен быть **WorldDynamic**.

ПРИМЕЧАНИЕ

Типы объектов

Любому актеру, способному к столкновению, присвоен тип объекта. Существует шесть типов объектов: **WorldStatic**, **WorldDynamic**, **Pawn**, **PhysicsBody**, **Vehicle** и **Destructible**.

Флажки Collision Responses

Раздел **Collision Responses** панели **Details** позволяет устанавливать, как выбранный актер статичного меша реагирует на другие актеры с определенными типами объектов.

Существует три состояния коллизионных взаимодействий:

- ▶ **Ignore**: игнорировать любые реакции на коллизии данных типов объектов;
- ▶ **Overlap**: проверить, пересекается ли оболочка коллизии меша с другим актером;
- ▶ **Block**: останавливать другой актер от проникновения в оболочку коллизии меша.

ПРИМЕЧАНИЕ

Нет оболочки коллизии в ассете, нет реакции на коллизию в актере

Если ассет статичного меша не имеет назначеннной в редакторе статичных мешей оболочки коллизии, то любой актер статичного меша, связанный с данным ассетом, не сможет обрабатывать ни одно событие коллизии, вне зависимости от того, какой тип реакции на коллизию назначен актеру.

Резюме

В этом часе вы изучили статичные меши и некоторые инструменты для работы с ними. Теперь вы умеете импортировать 3D-модели, работать в редакторе статичных мешей, создавать оболочки коллизий, помещать актеры статичных мешей на уровень и изменять базовые свойства коллизий. Конечно, вы узнали о статичных мешах далеко не все, но теперь имеете базовые знания, от которых будете отталкиваться.

Вопросы и ответы

Вопрос: Я знаю, что можно импортировать статичный меш в UE4, а можно ли экспортить статичный меш из UE4?

Ответ: Да, щелкните правой кнопкой мыши на панели **Content Browser** и выберите вариант **Asset Actions** ⇒ **Export** из выпадающего списка.

Вопрос: Могу ли я изменить якорную точку аксета статичного меша?

Ответ: Нет, по крайней мере, не в редакторе статичных мешей. И все же вы можете внести изменения в исходный аксет меша во внешнем приложении для 3D-моделирования и затем вновь импортировать меш. Если у вас нет исходного меша, вы можете экспортить меш в файл с расширением **.fbx**.

Вопрос: Что такое панель **Socket Manager** и для чего она используется?

Ответ: Панель **Socket Manager** позволяет создавать точки на меше, используемые для установления иерархических связей, то есть для прикрепления одного актера к другому. Эти связи полезны, когда вы анимируете или двигаете актер с помощью блюпринта.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

- Истинно или ложно высказывание: лучше всего всегда устанавливать для меша многополигональную коллизию.
- Какую клавишу нужно нажать вместе с клавишой **Alt**, чтобы отобразить оболочки коллизий во вьюпорте редактора уровней?
- Истинно или ложно высказывание: по умолчанию UV-слой карты освещения хранится в канале **UV Channel 1**.

4. Истинно или ложно высказывание: если ассету статичного меша не присвоена оболочка коллизии, то актер статичного меша, относящийся к данному мешу, все же будет иметь реакции на коллизии.
5. Истинно или ложно высказывание: если присвоить новый материал актеру статичного меша, он заменит материал, присвоенный ассету статического меша.

Ответы

1. Ложь. В целях эффективности лучше всегда использовать простые формы коллизий.
2. Комбинация клавиш **Alt+C** включит отображение оболочек коллизий во вьюпорте редактора уровней.
3. Истина. По умолчанию UE4 использует канал **UV Channel 1** для карт освещения.
4. Ложь. Если ассет статичного меша не имеет оболочки коллизии, то ни один актер статичного меша, связанный с этим ассетом статичного меша, не сможет обрабатывать никакие события коллизий.
5. Ложь. Присвоение нового материала актеру статичного меша влияет только на актер.

Упражнения

Найдите файлы моделей с расширением *.fbx* и *.obj* в сети Интернет или используйте файлы в папке *Hour_04* (доступной по адресу: http://addons.eksmo.ru/it/UE_24.zip) и импортируйте их. Отредактируйте их оболочки коллизий и поместите их на уровень.

1. Создайте папку *Maps* на панели **Content Browser**.
2. Создайте новую карту по умолчанию и сохраните ее в только что созданной папке *Maps*.
3. Создайте папку *Mesh* на панели **Content Browser**.
4. Импортируйте файл модели с расширением *.obj*.
5. Импортируйте файл модели с расширением *.fbx*.
6. Назначьте новый материал каждому ассету статичного меша.
7. Измените свойства оболочек коллизий, используя пресеты и автоматическую декомпозицию.

8. Поместите актеры статичных мешей на уровень несколько раз.
9. Для каждого меша произведите трансформацию перемещения, масштабирования и поворота.
10. Назначьте разные материалы каждому актеру статичного меша.
11. Сохраните уровень в папке *Maps*.

5-Й ЧАС

Применение освещения и рендеринга

Что вы узнаете в этом часе

- ▶ Изучение терминологии освещения
- ▶ Использование различных типов источников света
- ▶ Как применять свойства освещения
- ▶ Просчет освещения
- ▶ Использование настроек мобильности

В этом часе вы научитесь работать с актерами ИС (источников света) и изучите доступные типы ИС. Затем узнаете, как помещать актеры ИС на уровень, изменять их настройки и контролировать то, как они будут влиять на другие актеры в мире.

Источники света довольно просто поместить на уровень и редактировать. Но может быть непросто понять, как они работают и взаимодействуют с другими актерами и как применять настройки рендеринга.

ПРИМЕЧАНИЕ

Подготовка к практике

Создайте новый проект с шаблоном **Third Person** и стартовым контентом.

Изучение терминологии освещения

Некоторые базовые ключевые концепции помогут в понимании опций при работе со свойствами актеров ИС.

- ▶ **Прямое освещение** (direct lighting) — свет, падающий на поверхность актера, без каких-либо препятствий со стороны других актеров. Свет идет прямо из источника света к поверхности меша. Таким образом, актер статичного меша получает полный цветовой спектр от источника света.

- ▶ *Непрямое* (indirect) или *отраженное* освещение (bounced lighting) — свет, отраженный с поверхности другого актера на сцене. Поскольку свет поглощается или отражается в зависимости от свойств поверхности и цвета меша, отраженный свет передает некоторую информацию о цвете попадающимся на его пути поверхностям.
- ▶ *Статичное* освещение (static lighting) — освещение неподвижных объектов и ИС. Для неподвижных вещей освещение и тени вычисляются только однажды (во время просчета), что повышает и производительность, и качество «картинки».
- ▶ *Динамическое* освещение (dynamic lighting) относится к ИС и объектам, которые могут двигаться во время игры. Поскольку этот тип освещения вычисляется постоянно, он, как правило, медленнее и имеет более низкое качество, чем статичное освещение.
- ▶ *Тени* (shadows) создаются, когда движок сохраняет контур меша из точки ИС и проецирует его на поверхность актеров с противоположной стороны актера меша. Как актеры статичных мешей, так и актеры ИС имеют свойства теней, которые можно настраивать.

Изучение типов источников света

В Unreal Engine существует пять базовых актеров ИС: **Point Light**, **Spot Light**, **Directional Light**, **Rect Light** и **Sky Light** (см. рис. 5.1). Все они имеют некоторые общие настройки свойств. В то же время каждый тип актера ИС имеет и собственные уникальные настройки.

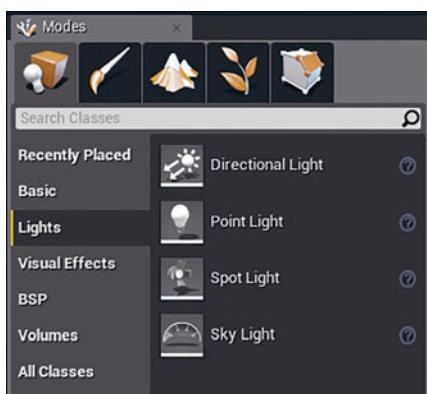


Рис. 5.1. Актеры ИС можно найти в разделе **Lights** на панели **Modes**

Добавление точечных источников света

Точечные источники света (**Point Lights**) похожи на лампочки в нашем реальном мире. Они излучают свет одинаково во всех направлениях из единственной точки в пространстве. Это наиболее распространенный тип освещения, особенно в сценах внутри помещений.

ПОПРОБУЙТЕ САМИ

Добавление точечного источника света на сцену

Выполните следующие шаги, чтобы создать пустой уровень без освещения и добавить различные типы освещения.

1. Создайте новый пустой уровень (**Empty Level**). Он должен быть черным.
2. На панели **Modes** выберите вкладку **Basic** и перетащите куб (**Cube**) на уровень.
3. На панели **Details** установите следующие координаты положения (**Location**) **0,0,0** и установите масштаб (**Scale**) **20,20,1**, чтобы создать поверхность пола.
4. На панели **Modes** выберите вкладку **Lights** и перетащите точечный ИС (**Point Light**) на уровень.
5. На панели **Details** установите координаты положения (**Location**) **400,0,200**.
6. На панели **Modes** выберите вкладку **Basic** и перетащите куб (**Cube**) на уровень. Обратите внимание на тень, падающую на пол от точечного ИС.
7. На панели **Details** установите координаты положения (**Location**) **500,100,90**.
8. Поиграйте с изменением различных параметров ИС и куба, чтобы увидеть, какие эффекты они имеют.
9. Сохраните уровень и назовите его **LightStudy**.

На рис. 5.2 изображен результат этой рубрики «Попробуйте сами».

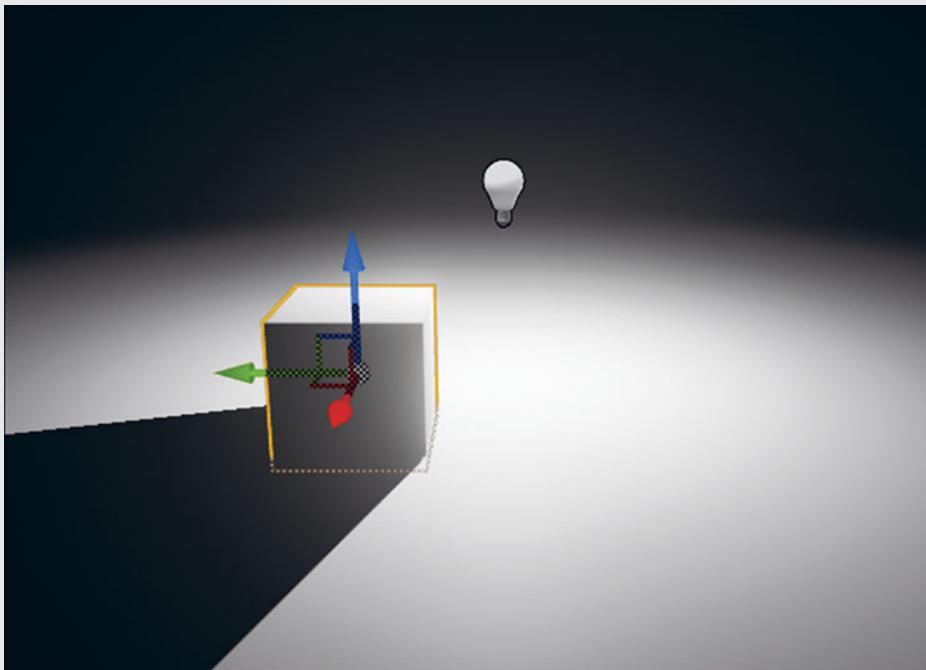


Рис. 5.2. Точечное освещение

Добавление прожекторных источников света

В UE4 прожекторные источники света (**Spot Lights**) излучают свет конической формы из одной точки в заданном направлении, как прожекторы в реальном мире. Направление прожекторного ИС устанавливается с помощью трансформации поворота актера прожекторного ИС. Вы можете менять затухание, чтобы устанавливать дистанцию, которую преодолевает свет от того места, где установлен прожекторный ИС. Свойства углов внутренней и внешней части конуса влияют на то, как быстро свет меняется от максимальной интенсивности в центре конуса до отсутствия света в его углах. Чем ближе эти значения окажутся друг к другу, тем резче будет свет в углах.

ПОПРОБУЙТЕ САМИ**Добавление прожекторного источника света на сцену**

Выполните следующие шаги, чтобы добавить несколько прожекторных ИС на уровень, который вы создали в предыдущей рубрике «Попробуйте сами».

1. Откройте уровень **LightStudy**, который вы создали в предыдущей рубрике.
2. Удалите точечный ИС со сцены.
3. На панели **Modes** выберите вкладку **Lights** и перетащите прожекторный ИС (**Spot Light**) на уровень.
4. На панели **Details** установите координаты положения (**Location**) **600,60,300**.
5. Поиграйте с изменением цвета и добавлением новых ИС.

На рис. 5.3 изображен результат этой рубрики «Попробуйте сами».

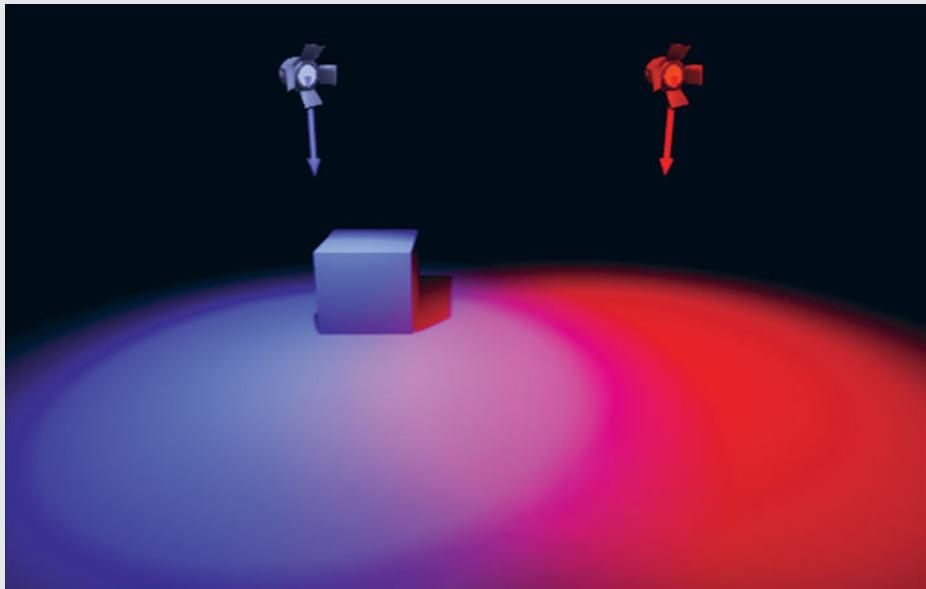


Рис. 5.3. Красный и синий прожекторные ИС, перекрывающие друг друга и создающие фиолетовое освещение

Добавление небесных источников света

Небесный источник света (**Sky Light**) охватывает самые отдаленные части уровня — все, что дальше, чем **SkyDistanceThreshold**, — и освещает их. Небесный ИС представляет собой отображение неба. Его освещение и отражения совпадут, даже если небо будет состоять из атмосферы или насыщаемых облаков вверху скайбокса* или отдаленных гор. Использование небесного ИС — хороший способ украсить весь уровень и повлиять на цвета теней.

ПРИМЕЧАНИЕ

Актеры тумана

В зависимости от того, как небесный ИС функционирует, может понадобиться добавление актера атмосферного тумана или растущего тумана, чтобы увидеть результаты небесного ИС на сцене.

ПОПРОБУЙТЕ САМИ

Добавление небесного источника света на сцену

Выполните следующие шаги, чтобы добавить небесный источник света на сцену, над которой вы работаете.

1. Откройте уровень, который вы создали в предыдущей рубрике «Попробуйте сами». При желании удалите другие ИС.
2. На панели **Modes** выберите вкладку **All Classes** и перетащите небесную сферу (**BP_Sky_Sphere**) на уровень. Это небо.
3. На панели **Modes** выберите вкладку **Lights** и перетащите небесный ИС (**Sky Light**) на уровень. Обратите внимание, что сцена приобрела цвет неба.
4. Поиграйте со свойствами небесной сферы и небесного ИС, чтобы увидеть различные эффекты.

На рис. 5.4 изображен результат этой рубрики «Попробуйте сами».

* Скайбокс (skybox) — объект в 3D-графике, играющий роль неба и горизонта. — *Прим. ред.*

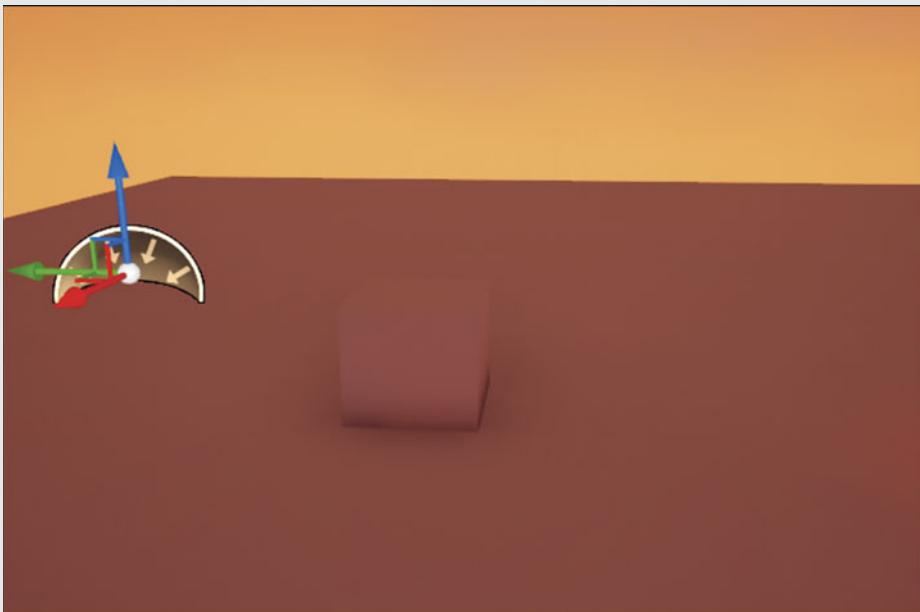


Рис. 5.4. Небесное освещение

Добавление направленных источников света

Направленный источник света (**Directional Light**) имитирует свет, излучаемый источником, который находится бесконечно далеко. Все тени, отбрасываемые им, будут параллельны, что делает направленный ИС идеальным для симуляции солнечного света. Когда вы используете направленный ИС на уровне, неважно, куда вы его поместите, только направление имеет значение.

ПОПРОБУЙТЕ САМИ

Добавление направленного источника света на сцену

Выполните следующие шаги, чтобы добавить направленный ИС на сцену.

1. Откройте уровень, который вы создали в предыдущей рубрике «Попробуйте сами».
2. На панели **Modes** выберите вкладку **Lights** и перетащите направленный ИС (**Directional Light**) на уровень.

3. Используйте инструмент поворота (**Rotate**), чтобы менять направление источника света, и посмотрите на результат.
4. Чтобы создать небо, выберите небесную сферу на панели **World Outliner**.
5. На панели **Details** установите **Directional Light Actor** в значение **DirectionalLight**. Теперь небо контролирует направленный ИС.
6. Поиграйте со свойствами небесной сферы и направленного ИС, чтобы увидеть различные эффекты.

На рис. 5.5 изображен результат этой рубрики «Попробуйте сами».

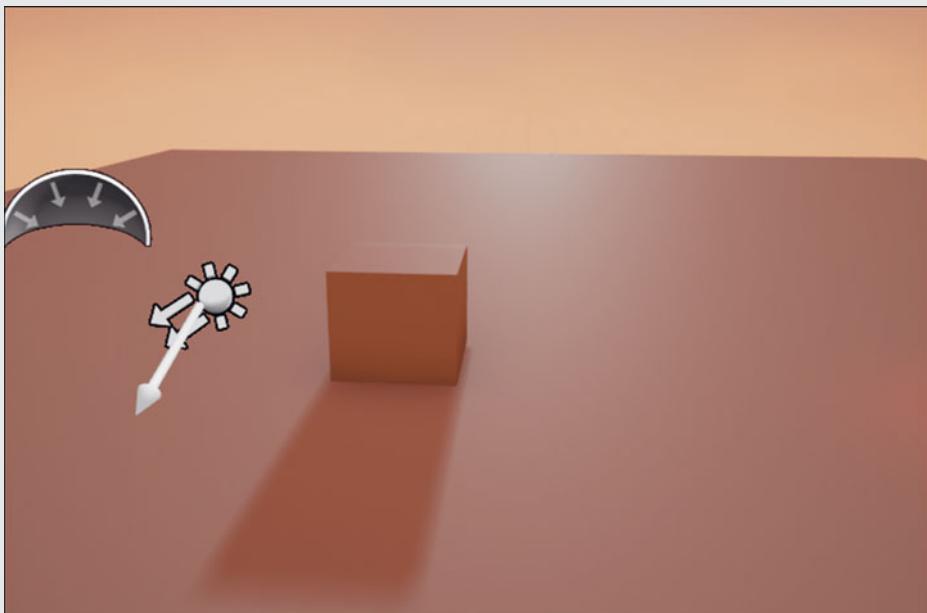


Рис. 5.5. Направленное и небесное освещение

Использование свойств освещения

На вкладке **Light** для каждого ИС на сцене содержатся свойства, включая перечисленные в табл. 5.1.

ТАБЛ. 5.1. Свойства источников света

Свойство	Описание
Intensity (Интенсивность)	Определяет яркость ИС в люменах для точечных и прожекторных ИС. 1700 люменов соответствует лампочке мощностью 100 Вт
Light Color (Цвет света)	Определяет цвет освещения. Цвета складываются, поэтому если красный свет упадет на синий объект, он приобретет фиолетовый цвет
Attenuation Radius (Радиус затухания)	Определяет максимальную дистанцию, которую достигает свет. Освещенность угасает от максимума в источнике света до нуля на грани радиуса
Cast Shadows (Наклонные тени)	Определяет, будут ли падать на объекты наклонные тени. Расчет динамических теней может повысить нагрузку на процессор
Inside Cone Angle (Внутренний угол конуса)	Устанавливает угол (в градусах) области освещения прожекторного ИС
Outside Cone Angle (Внешний угол конуса)	Устанавливает угол (в градусах) области ослабевания прожекторного ИС. Если он близок к внутреннему углу, края области прожекторного ИС будут острыми
Temperature (Температура)	Позволяет установить цвет света с помощью шкалы цветовой температуры Кельвина. Хорошо, если вы постараетесь соответствовать реальным цветам света. Для этого нужно установить флажок Use Temperature

Существуют и другие свойства, дающие полный контроль над освещением в игре, но для знакомства будет достаточно перечисленных выше.

ПРЕДУПРЕЖДЕНИЕ

Работа приложения

Неправильные выбранные параметры освещения могут значительно повлиять на работу приложения. Например, использование чрезмерного количества динамических ИС может вызвать серьезные проблемы в работе приложения. Кроме того, радиус затухания ИС может оказывать существенное влияние на работу приложения, поэтому старайтесь как можно реже использовать большие значения этого параметра.

Просчет освещения

Инструмент UE4 для просчета освещения называется *Lightmass*. Он имеет множество параметров, выходящих за пределы этой книги. Тем не менее вы можете управлять некоторыми из них, для этого выберите пункт **Window** ⇒ **World Settings** и в открывшейся панели найдете все настройки в разделе **Lightmass**.

Несмотря на то что UE4 может отображать все источники света и меши на уровне с динамическим освещением, это повлияет на производительность и качество. Если UE4 знает, что ИС не будет двигаться, он может заранее просчитать освещение и тени для этого ИС, а также все для всех актеров, которых они коснутся. Конечно, хранение заранее просчитанного освещения меньше нагружает процессор во время геймплея, но оно требует больше памяти.

Инструменты просчета освещения в UE4 используются, чтобы заранее вычислять информацию о тенях и освещении на уровне для статичных мешей, актеров ИС и BSP (см. главу 9, чтобы узнать больше о BSP). Эта информация хранится, как изображения, внедренные в уровень, которые можно найти, выбрав пункт **Window** ⇒ **World Settings** ⇒ **Lightmass** ⇒ **Lightmaps**.

После того как вы произвели просчет освещения, редактор будет отображать заранее просчитанное освещение для любых статичных ИС. Когда вы добавляете новый ИС на уровень или удаляете ИС и меши, источники света отображаются, как динамические и обновляемые в режиме реального времени до тех пор, пока освещение не будет просчитано заново.

Чтобы просчитать освещение, щелкните по стрелке на кнопке **Build** (просчитать) на панели инструментов (см. рис. 5.6). В появившемся подменю выберите пункт **Lighting Quality** ⇒ **Preview**, чтобы получить быстрый результат. Когда конечный уровень готов, выберите пункт **Lighting Quality** ⇒ **High**. Высокое качество требует больше времени на генерацию, но дает результаты точнее.



Рис. 5.6. Кнопка Build

Приложение Swarm Agent

Обратите внимание, когда производится просчет освещения, автоматически в фоновом режиме запускается приложение **Swarm Agent**. Оно управляет коммуникацией между редактором и инструментом **Lightmass**. При просчете освещения **Swarm Agent** отслеживает и отображает прогресс просчета. По мере роста сложности уровня увеличивается время на просчет освещения.

Приложение **Swarm Agent** может быть настроено на коммуникацию с удаленными компьютерами в сети и использовать их вычислительную мощность, чтобы уменьшить время вычислений. Это не требуется для маленьких проектов и уровней, но полезно знать, что приложение поддерживает распределенный сетевой рендеринг.

ПРИЛОЖЕНИЕ

Повторный просчет освещения

Каждый раз, когда вы помещаете ИС, настроенный на отбрасывание статичных наклонных теней, или актер статичного меша, который установлен как статичный, редактор напомнит о необходимости просчета освещения. Чем больше источников света и объектов, тем дольше будет производиться повторный просчет освещения. При работе со светом лучше всего просчитывать освещение после внесения всех изменений. Вы можете просматривать и производить игровое тестирование уровня без повторного просчета освещения, но освещение не будет правильным без него.

ПОПРОБУЙТЕ САМИ

Произведите просчет статичного освещения сцены

Выполните следующие шаги, чтобы добавить статичное освещение сцены, над которой вы работаете.

1. Создайте новый уровень по умолчанию.
2. Найдите ассет статичного меша **Shape_Cube** в папке *StarterContent/Shapes* на панели **Content Browser** и поместите его на уровень таким образом, чтобы он стоял на полу.
3. В главной панели инструментов щелкните по стрелке рядом с кнопкой **Build**, чтобы раскрыть выпадающее меню.
4. Выберите пункт **Lighting Quality ⇒ Preview**.
5. Щелкните по иконке **Build**, чтобы начать просчет освещения. После того как просчет будет закончен, тень актера статичного меша обновится и отобразятся по-новому просчитанное освещение и тени.
6. Теперь поменяйте качество заранее просчитанной тени. Выберите актер статичного меша, представляющий собой пол, который был добавлен при создании уровня по умолчанию. На панели **Details** в разделе **Lighting** установите флагок **Overridden Light Map Res** и установите значение **1024**.
7. Щелкните по иконке **Build**, чтобы вновь начать просчет освещения. Вы увидите, как изменилось качество тени. На рис. 5.7 изображен результат изменения разрешения карты освещения.

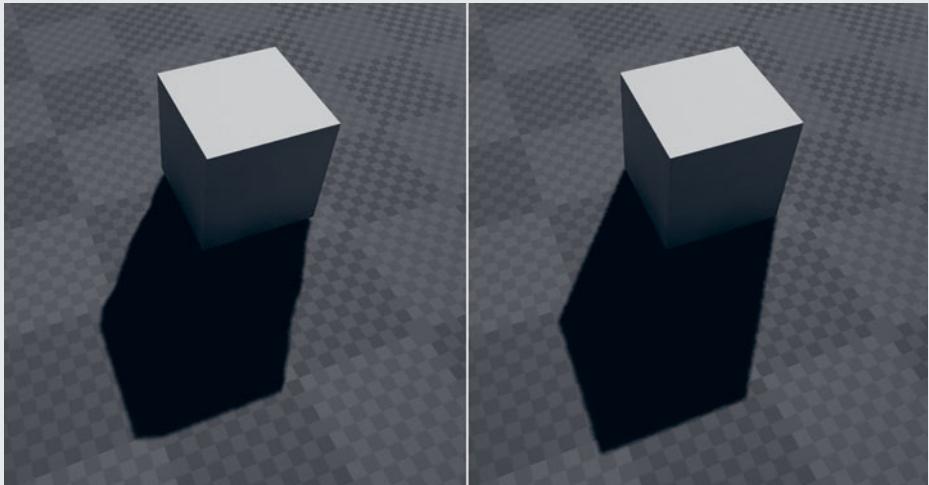


Рис. 5.7. Статичное освещение с разрешением карты освещения по умолчанию (слева) и с разрешением 1024 (справа)

ПРИМЕЧАНИЕ

Настройка разрешения карты освещения

При необходимости вы можете изменять разрешение карты освещения для каждого актера на уровне или изменить разрешение карты освещения по умолчанию для ассета статичного меша в редакторе статичных мешей. Повышая разрешение, вы увеличиваете время просчета освещения.

Мобильность

Каждый ИС имеет настройку мобильности, значениями которой являются **Static** (статичный), **Movable** (подвижный) и **Stationary** (стационарный). Эти параметры помогут UE4 определить, какие ИС являются динамичными, чтобы заранее пропросчитать и сохранить («запечь») их в карте освещения.

- ▶ *Статичными* ИС являются источники света, которые не могут быть изменены или перемещены никоим образом в процессе игры. Информация об освещении просчитывается перед геймплеем и хранится в специальной текстуре, которая называется *карта освещения* (light map). Статичный ИС не требует много ресурсов, но не работает с подвижными объектами в радиусе освещения. Главной причиной использования параметра **Static** является производительность, например мобильных устройств.

- ▶ *Подвижные* ИС производят полностью динамические свет и тени, они могут менять местоположение, угол наклона, цвет, яркость, затухание, радиус и практически все имеющиеся у них свойства. Производимый ими свет не «запекается» в картах освещения, и они не могут иметь непрямое освещение. Такие источники света, как правило, затрачивают много ресурсов для отображения и не имеют столь высокого качества, как статичные и стационарные источники света. Можно использовать подвижные ИС для персонажей, которые движутся, например, для игрока, который держит в руке фонарь.
- ▶ *Стационарные* ИС похожи на статичные в том, что не способны двигаться, но их яркость и цвет могут меняться в процессе игры. Это полезно, например, для лампочек, которые можно включить и выключить, но нельзя передвинуть. Параметр **Stationary** дает среднюю производительность и высокое качество.

Табл. 5.2 поможет определить, какой из параметров настройки мобильности следует использовать для актеров статичных мешей.

ТАБЛ. 5.2. Параметры мобильности источников света и мешей

Параметры настройки ИС			
Параметры настройки статичных мешей	Статичные	Стационарные	Подвижные
Статичные	запекаемое освещение	запекаемое освещение	запекаемое освещение
Подвижные	динамичные тени	динамичные тени	динамичные тени

Чтобы изменить параметры настройки мобильности любого актера ИС на уровне, выберите ИС и на панели **Details** уровня в разделе **Transform** задайте нужное значение параметра **Mobility**. Направленный ИС, который находится на сцене по умолчанию, уже имеет стационарный параметр настройки мобильности. Поэтому, если вы измените мобильность актера статичного меша **Shape_Cube**, который вы поместили в предыдущей рубрике «Попробуйте сами», на подвижный, вы измените то, как редактор отбрасывает тени для актера.

ПОПРОБУЙТЕ САМИ

Отбрасывание динамичных теней

Выполните следующие шаги, чтобы изменить параметры настройки мобильности актера статичного меша и заставить его отбрасывать динамичные тени.

1. Продолжите работу над уровнем, который вы создали в предыдущей рубрике «Попробуйте сами».
2. На уровне выберите актер статичного меша **Shape_Cube** и на панели **Details** в разделе **Transform** установите параметр **Mobility** в значение **Movable**, щелкнув по названию этого значения.

Резюме

Этот час начался с освещения некоторой базовой терминологии, связанной с освещением. Вы узнали о различных типах ИС в UE4 и их предназначении, как помещать источники света и настраивать их. Вы также узнали о построении освещения и как параметры настройки мобильности влияют на статичное и динамичное освещение. Освещение — один из сложнейших и мощных аспектов UE4, который может быть непреодолимым. Потребуется много времени для того, чтобы изучить подробности. Знаний из этого часа достаточно, чтобы начать работать над освещенными сценами.

Вопросы и ответы

Вопрос: Как много ИС я могу добавить на сцену?

Ответ: На этот вопрос непросто ответить. Если речь идет о маленьких статичных ИС, которые не касаются большого количества объектов, вы можете добавить сотни, а то и тысячи. С другой стороны, использование всего нескольких динамичных ИС с огромным радиусом, покрывающих всю сцену, может быть чрезмерным. Лучше всего экспериментировать и смотреть, что работает.

Вопрос: Почему тени или освещение на моей сцене не выглядят правильно?

Ответ: Если ваши ИС статичные, вам, возможно, необходимо просчитать освещение еще раз, щелкнув по кнопке **Build** на панели инструментов.

Вопрос: Как разработчики игр делают некоторые сцены выглядящими столь реалистично?

Ответ: Взгляните на все примеры сцен, входящие в UE4, и проверьте, как установлены источники света. Возможно, вы определите для себя некоторые

приемы, используемые для создания различных эффектов. Как и в магии, дым и отражения всегда используются в разработке игр.

Вопрос: Зачем карты освещения генерируются в процессе просчета освещения и вставляются в уровень?

Ответ: Размещение источников света и актеров на уровне уникально для каждого уровня, поэтому генерируемая информация об освещении важна только для этого уровня.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Какой тип источника света выбрать, чтобы осветить всю сцену одним ИС?
2. Когда использовать статичный, а когда стационарный ИС?
3. Когда использовать статичный, а когда подвижный ИС?
4. Что такое **Lightmass**?

Ответы

1. Чтобы осветить всю сцену одним ИС, используйте небесный или направленный ИС.
2. Используйте статичный или стационарный ИС, когда неподвижны и ваш ИС, и все, на что упадет его свет.
3. Используйте стационарный или подвижный ИС, когда ваш ИС или актер, на который падает его свет, будут двигаться.
4. **Lightmass** — это движок статичного освещения UE4, который используется при просчете освещения.

Упражнения

В этом упражнении создайте простую сцену, поместив в нее статичные меши и все типы ИС.

1. Создайте новый пустой уровень (**Empty Level**).
2. На панели **Modes** выберите вкладку **Geometry** и переместите куб (**Box**) во виджит.

3. Убедитесь, что куб выбран, и на панели **Details** в разделе **Brush Settings** установите значение **1000** для осей X и Y, а для оси Z — **20**. UE4 создаст большую платформу.
4. На панели **Modes** выберите вкладку **Basic** и перетащите стартовую точку игрока (**Player Start**) в центр платформы прямо над поверхностью.
5. На панели **Modes** выберите вкладку **Basic** и перетащите два куба (**Cube**). Поместите их на платформу прямо над поверхностью.
6. Выберите один из актеров-кубов и на панели **Details** в разделе **Physics** установите флажок **Simulate Physics**.
7. Добавьте направленный ИС (**Directional Light**), установив для него угол поворота (**Rotation**) **0,200,45** и цвет (**Light Color**) **255,205,105**.
8. Добавьте точечный ИС (**Point Light**) на сцену так, чтобы он находился непосредственно над платформой. Установите его интенсивность (**Intensity**) в значение **15 000**, цвет (**Light Color**) в значение **255,0,255** и радиус затухания (**Attenuation Radius**) в значение **250**.
9. Добавьте прожекторный ИС (**Spot Light**) на сцену и поместите его на пустой участок на **300** единиц над платформой. Установите его интенсивность (**Intensity**) в значение **30000**, цвет (**Light Color**) в значение **210,255,15**, внутренний угол конуса (**Inner Cone Angle**) в значение **22** и внешний угол конуса (**Outer Cone Angle**) в значение **24**.
10. На панели **Modes** выберите вкладку **Visual Effects** и перетащите на сцену атмосферный туман (**Atmospheric Fog**).
11. Добавьте небесный ИС (**Sky Light**). Установите его интенсивность (**Intensity**) в значение **10** и цвет (**Light Color**) в значение **215,60,15**.
12. Щелкните по кнопке **Build** на панели инструментов, чтобы произвести просчет освещения.
13. Просмотрите уровень, перемещаясь по нему и передвигая куб с флагком симуляции физики, чтобы увидеть, как он взаимодействует с источниками света.
14. Произведите настройки ИС по своему усмотрению. Перед просмотром уровня не забудьте произвести повторный просчет освещения.

6-Й ЧАС

Использование материалов

Что вы узнаете в этом часе

- ▶ Изучение материалов и их использования
- ▶ Использование основанного на физике рендеринга
- ▶ Использование редактора материала
- ▶ Использование типов текстур, размеров и импорт текстур
- ▶ Изучение нод материалов и констант
- ▶ Использование экземпляров и параметров

В этом часе вы узнаете, что представляют собой материалы и как использовать их в UE4. Вначале вы получите базовые знания, как использовать основанный на физике рендеринг. Затем вы узнаете о каждом типе входных данных для материалов, и как они отображаются в реальном времени. Затем познакомитесь с идеальными размерами текстур, разрешением и параметрами настройки, а также как использовать их при установке материала. Вы узнаете, как создавать новый материал с помощью редактора материала, и об экземплярах и параметрах нодов. Наконец, вы создадите ваш собственный материал.

ПРИМЕЧАНИЕ

Подготовка к практике

Создайте новый пустой проект со стартовым контентом.

Изучение материалов

Материал (material) или шейдер (shader) — это совокупность текстур, векторов и других математических вычислений для описания поверхности и свойств ассетов в UE4 (см. рис. 6.1). Сначала материалы могут показаться сложными, но они являются одной из наиболее визуально простых частей UE4. Вы можете использовать материалы, чтобы описывать свойства поверхности ассета для игрока,

изменяя визуальный контекст и стиль. Главным образом материалы информируют UE4, как свет реагирует на их поверхность. Ассеты в UE4 имеют специфичные материалы, примененные к ним. По умолчанию к различным объектам применяны стандартные материалы UE4. Когда вы смотрите на скалу, дерево или бетонную стену в игре, каждый из этих ассетов имеет определенный материал, дающий им уникальное отображение.

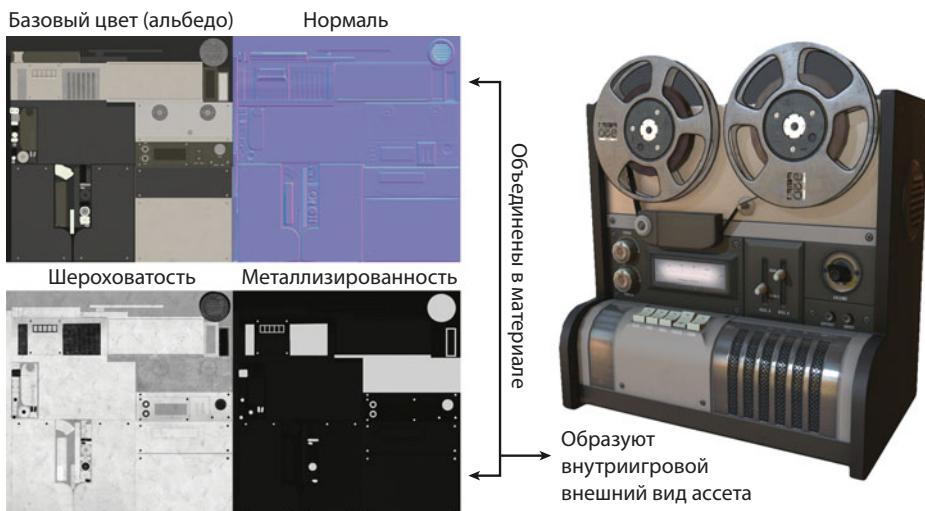


Рис. 6.1. Аспекты и текстуры, объединенные в материале, создают финальный результат, отображаемый в игре

Основанный на физике рендеринг (PBR, Physically Based Rendering)

Система отображения материала в UE4 использует основанный на физике рендеринг (PBR, Physically Based Rendering) для генерации в режиме реального времени. PBR — относительно новая концепция в мире создания текстур и материалов для игр. Изначально основанный на физике рендеринг был частью процесса помещения деталей освещения прямо в текстуры, чтобы придать форму и объем поверхности ассета. Проблема этого метода стала очевидной, когда ассеты перемещались в другие типы сценариев освещения и появлялись видимые расхождения между тенями и ИС, а также направлением света и теней внутри сцены. Эти расхождения приводили к визуальным разрывам в целостности и правдоподобности игрового мира.

Теперь, благодаря росту вычислительной мощности и технологии UE4, параметры материалов используются, чтобы позволять ассетам формировать собственную информацию об освещении и тенях. Эта техника защищает ассеты от «запекания»

или перманентного помещения в текстуры. Кроме того, ассеты могут быть освещенными как любое время суток, по любому сценарию освещения без необходимости повторной разработки текстур. Это позволяет стабильнее и проще создавать текстуры для игровых ассетов.

ПРИМЕЧАНИЕ

Система материалов PBR

Основанный на физике рендеринг адаптирован под основное использование для разработки игр в течение прошедших нескольких лет, но индустрии кино и телевидения использовали его в течение некоторого времени для 3D-анимации и рендеринга. PBR помогает материалам реагировать на различные сценарии освещения более реалистично. Эта технология также позволяет повторно использовать текстуры и ассеты в процессе разработки и создавать более реалистичное освещение в каждой сцене.

Типы входных данных для материалов

Для создания отличных материалов и текстур используется редактор материалов (*Material Editor*). В следующих разделах рассмотрены наиболее часто используемые входные данные редактора материалов.

Базовый цвет (альбедо)

Базовый цвет (Base color), который также иногда называют *альбедо* (albedo) или *рассеивание* (diffuse) — ключевое описание цвета поверхности материала без учета всех деталей теней и освещения. По существу, входные данные базового цвета используют текстуру альбедо, которая представляет собой чистое значение цвета создаваемого вами материала. Они не должны учитывать информацию о тени и освещении, а показывать только цвет, который вы хотите представить в материале. Это свойство может использовать входные данные текстур или даже простое векторное значение, которое является числом, представляющим чистый цвет.

Металлизированность

Входные данные *металлизированности* (Metalness) материала используются, чтобы описать, является ли материал металлическим. Эти входные данные являются одним из наиболее простых для понимания элементов редактора материалов, а также одним из важнейших параметров для получения правильного отображения. Способ использования металлизированности в UE4 весьма удобен и позволяет быстро контролировать и понимать, какой из материалов является металлическим. Каждый пиксель в такой карте текстур, как правило, черный или белый, с небольшим количеством серого цвета. Черный представляет собой материалы,

которые не являются металлическими, например камень, кирпич или дерево, а белые представляют аспекты металлических материалов, таких как железо, серебро или медь. Часто, если материал не является металлическим, простой вектор 0 можно использовать для упрощения ввода. Вы также можете сократить ввод до текстуры в оттенках серого или до всего одного канала текстуры благодаря ее простоте.

ПРИМЕЧАНИЕ

Металлизированность и PBR-система, основанная на свойстве Блеск (Specular)

Имеются два существенных типа основанных на PBR систем для создания текстур в играх. В то время как UE4 использует системы, основанные на металлизированности, некоторые движки используют системы, основанные на блеске. Обе системы производят практически идентичные результаты. Главное отличие состоит в том, как разработчик создает текстуры, чтобы движок мог лучше понять их.

Шероховатость

Шероховатость (Roughness), глянец (gloss) или микроповерхностная детализация (micro surface detail), является одним из наиболее гибких аспектов PBR-системы. Такая текстура используется для представления шероховатости и изношенности поверхности создаваемого материала. Шероховатость изображает миниатюрные детали и описывает глянец или количество света, отражаемого с поверхности. К примеру, если вы создаете новый стальной металлический материал, металличность и альбедо материала будут довольно простыми, вам понадобится сделать небольшие изменения в цвете и колебании шума, но вы будете использовать шероховатость, чтобы описать все мелкие детали поверхности, такие как мелкие царапины, загрязнения или пыль. Очень редко бывает одноцветная шероховатость, поскольку в реальной жизни поверхность часто имеет износ и повреждения. Вы можете упростить шероховатость до текстур в тонах серого, поскольку информация о цвете не используется для описания шероховатости.

Нормаль

Для задания нормалей используются карты нормалей текстур (normal map) или тройки векторов (координаты X, Y и Z, как обсуждалось в 3-м часе, «Координаты, преобразования, единицы измерения и организация»). Входные данные о нормали описывают направление, в котором каждый ИС должен взаимодействовать с поверхностью. Карты нормалей имитируют детали поверхности в высоком разрешении и форму в ассете, забирая свет в отображение этих деталей, основываясь

на карте текстур, освещенной попиксельно. Входные данные о нормали аналогичны карте высот в других 3D-пакетах и рендеринге, но с некоторыми незначительными отличиями в создании.

Понимание входных данных о нормали может сначала показаться трудным, поэтому удобнее для восприятия разбить ввод нормали на цветовые каналы, включенные в процесс. Каждый канал карты текстур (красный, зеленый и синий) составлен для представления различных углов направлений поверхности по цвету.

Красный представляет ось X или направление света слева направо, сталкивающееся с поверхностью. Зеленый представляет ось Y или направление сверху вниз. Синий представляет ось Z или направление спереди назад (см. рис. 6.2).

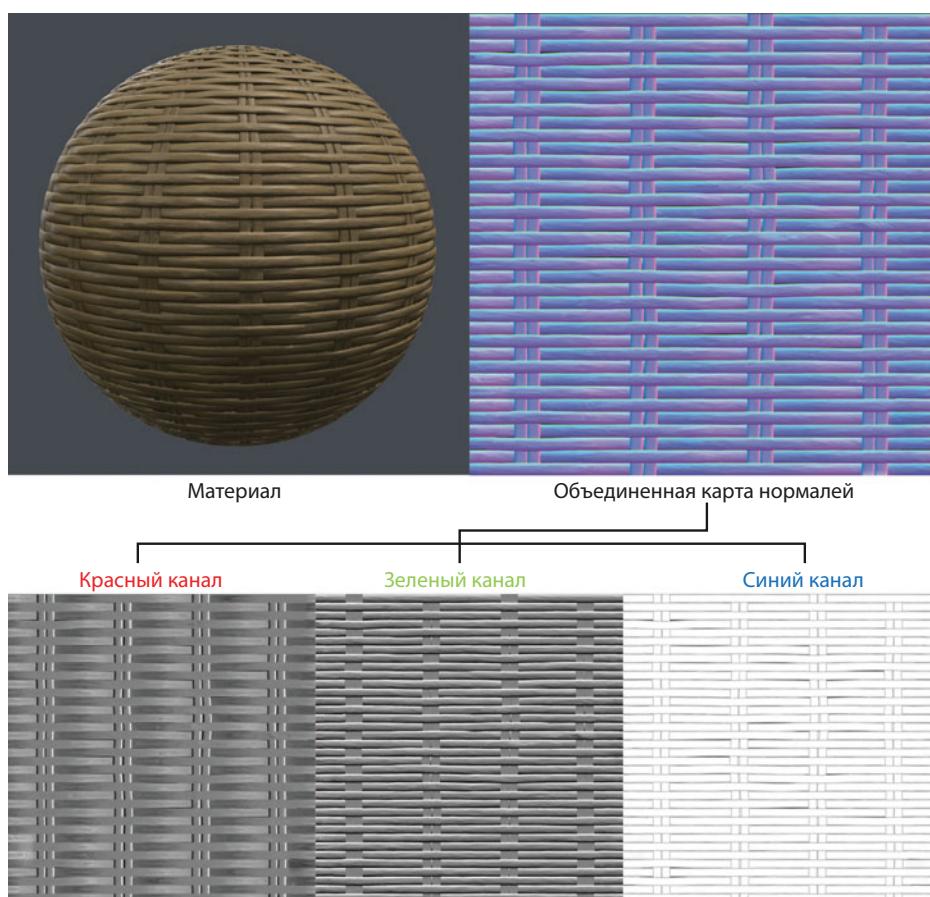


Рис. 6.2. Каждый из каналов карты нормалей имеет информацию о направленном освещении, которую UE4 использует, чтобы определить объем и форму поверхности

ПРИМЕЧАНИЕ

Зеленый канал

Зеленый канал иногда меняется или отражается, в зависимости от того, какое программное обеспечение трехмерной графики вы используете. Maya использует нормальный формат, в то время как Max и UE4 отражают зеленый канал для отображения карт нормалей. Чтобы отразить зеленый канал в UE4, выберите и откройте текстуру на панели **Content Browser** и включите настройку **Flip Green Channel**.

Представьте плоский полигон, освещенный источником света. Когда свет сталкивается с полигоном, он реагирует просто, показывая, что он освещен, как плоская поверхность, потому что излучение света сталкивается с полигоном и реагирует в одном направлении. Теперь подумайте о создании поверхности кирпича. Чтобы сделать это, если у вас нет модели, имеющей тысячи полигонов, вы можете создать карту нормалей для имитации мелких деталей поверхности, которые будут отображать кирпичи. Такая карта нормалей при подключении к редактору материалов дает знать UE4, что, когда свет сталкивается с поверхностью этого материала, он должен реагировать, как указано, попиксельно, что отражено в карте. Когда затем тот же ИС освещает поверхность, она реагирует соответственно и дает иллюзию формы и деталей, которых на самом деле нет.

Создание текстур

Текстуры являются основой колоризации и придают визуальный язык материалам и ассетам в UE4. В следующих разделах раскрывается, как создавать и использовать их.

Размеры текстур

Размеры текстур — это важный аспект при разработке. Создание текстуры, которая не использует правильные пропорции, может привести к тому, что она не будет отображаться правильно — станет скошенной или искаженной, или не будет импортироваться. Современные игры используют определенные стандарты размеров текстур.

Степень двойки

Понимание того, как текстуры отображаются (рендерятся) в UE4, является ключевым для того, чтобы знать, какие необходимо указать размеры текстур. Чтобы движок UE4 мог обрабатывать размеры текстур в реальном времени, все текстуры отображаются в выражении дистанции от обзора камеры игрока. Поэтому если вы видите ассет близко в мире игры, разрешение текстуры объекта близко к точному

размеру, с которым текстура была создана и импортирована. Если камера игрока отдаляется, мелкие детали текстур становятся ненужными, чтобы различать представленные цвета и формы, поэтому UE4 начинает сокращать размеры текстур, непрерывно уменьшая их в два раза по мере того, как игрок отдаляется от ассета. Этот процесс называется миппингом (mipping) или множественным отображением (mip mapping; см. рис. 6.3).

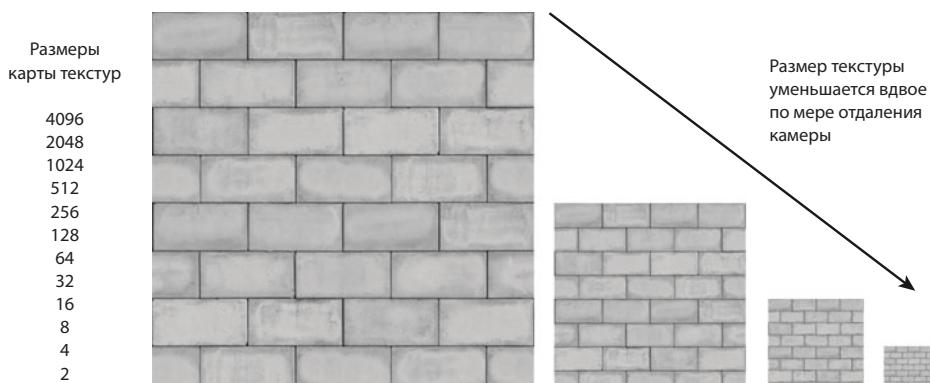


Рис. 6.3. Размер текстуры напрямую зависит от того, как динамично меняется размер текстуры в UE4 для экономии памяти

Теперь, когда вы знаете, как UE4 использует и отображает текстуры, вы готовы к дискуссии об аспектах отношений или пропорций. Размеры текстур начинаются примерно с 256 пикселей (px) и увеличиваются, каждый раз умножаясь на 2 (512, 1024, 2048 и реже 4096 пикселей). Это умножение важно, поскольку оно позволяет текстуре быть уменьшенной или увеличенной в два раза для простоты использования в UE4. Размеры текстур при необходимости могут быть и меньше — 128, 64, 32, 16, 8, 4 и даже 2 пикселя. Текстура может иметь разные размеры в высоту или ширину, но и высота, и ширина должны быть кратны степени двойки. Так, большинство текстур создано прямоугольной или квадратной формы (например, 512×512 пикселей или 1024×1024 пикселей), но это не значит, что тот же размер должен быть использован в каждом измерении. Текстура с высотой 512 пикселей и 1024 пикселя в ширину вполне уместна, поскольку UE4 делит на два каждую сторону по отдельности при отображении, UE4 уменьшает размер до 256×512 пикселей и снова до 128×256 пикселей.

Расширения файлов текстур

Чтобы создать материалы из созданных вами текстур, необходимо импортировать текстуры на панель **Content Browser**. Чтобы корректно импортировать и интегрировать эти текстуры в редактор материалов, требуется файлы определенных

типов и правильные параметры настройки. В настоящее время UE4 поддерживает следующие расширения файлов:

- ▶ .tga
- ▶ .psd
- ▶ .tiff
- ▶ .bmp
- ▶ .float
- ▶ .pcx
- ▶ .png
- ▶ .jpg
- ▶ .dds
- ▶ .hdr

Импорт текстур

Теперь, когда вы научились создавать текстуры и узнали, какие форматы являются приемлемыми, вы можете импортировать некоторые текстуры на панель **Content Browser**.

ПОПРОБУЙТЕ САМИ

Импортируйте текстуры на панель Content Browser

Чтобы импортировать текстуру в редактор, откройте UE4 и выполните следующие шаги.

1. Откройте панель **Content Browser**, щелкнув по кнопке **Content Browser** на панели инструментов или нажав комбинацию клавиш **Ctrl+Shift+F**.
2. Выберите папку на вашем компьютере для импорта текстуры.
3. Щелкните правой кнопкой мыши по пустой области в правой части панели **Content Browser** и выберите вариант **Import to /Game/StarterContent/Materials**.
4. Выберите папку **Windows** и найдите текстуру, которую хотите импортировать.
5. Щелкните по кнопке **Open**.

ПРИМЕЧАНИЕ

Перетащите и отпустите

Чтобы импортировать любой контент, включая текстуры, модели, видеофайлы и другие ассеты, в UE4, можно просто щелкнуть по локальному файлу на вашем компьютере, перетащить его на панель **Content Browser** и отпустить. Например, если файл находится на рабочем столе, вы можете просто щелкнуть по нему и перетащить его на панель **Content Browser**.

Создание материалов

Следующая рубрика «Попробуйте сами» поможет вам самостоятельно создать новый материал на панели **Content Browser**.

ПОПРОБУЙТЕ САМИ

Создайте материал на панели Content Browser

Чтобы создать материал на панели **Content Browser**, откройте UE4 и следуйте указаниям.

1. Откройте панель **Content Browser**.
2. Выберите одну из папок в каталоге *Content* для создания материала.
3. Щелкните правой кнопкой мыши по пустой области в правой части панели **Content Browser** и в разделе **Create Basic Asset** выберите вариант **Material**.
4. Измените название по умолчанию на новое.

Создав материал, вы можете свободно управлять всеми аспектами его входных данных. Вы можете дважды щелкнуть по любому материалу в редакторе контента, чтобы открыть его в редакторе материала (см. рис. 6.4). В редакторе материала вы можете вносить изменения в любые входные данные. Обратите внимание, что редактор материала имеет множество доступных опций, таких как просмотр материала в реальном времени.

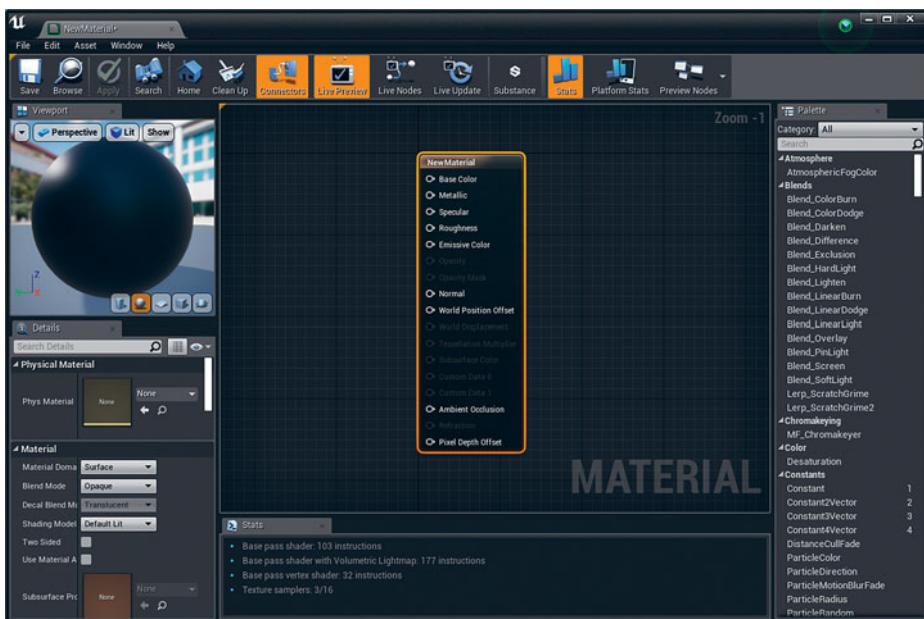


Рис. 6.4. Редактор материала

Помимо главной панели инструментов в редакторе материалов обычно используется четыре панели.

- ▶ **Выопорт (панель Viewport).** В левом верхнем углу редактора материалов находится выопорт, позволяющий просматривать материал в реальном времени. Выопорт показывает финальный результат сборки материала. Можно менять объект, на котором отображается материал, с помощью параметров формы в нижнем углу окна, а также визуальные атрибуты или атрибуты перспективы с помощью настроек в верхней части окна.
- ▶ **Панель Details.** Прямо под выопортом находится панель Details. С ее помощью можно менять все свойства материала и настройки рендеринга, которые использует материал в пространстве игры, такие как настройки прозрачности (opacity), подповерхности (subsurface) и затенения (shading) моделей. Эта панель полезна для углубленного редактирования материала.
- ▶ **Панель Graph.** Центральная панель — панель Graph, в которой производится все визуальное редактирование материала. Она позволяет перетаскивать внутрь или размещать текстуры, чтобы направлять входные данные или использовать специальные ноды для создания различных эффектов

на материале. Здесь вы соединяете текстуры, которые импортируете в UE4, с материалом, чтобы достичь окончательного результата.

- ▶ **Панель Palette.** Это самая правая панель. В ней хранятся все специальные ноды и математические функции для создания особых эффектов в материале.

Каждая панель в редакторе материалов играет свою роль в конструировании конечного материала и оптимизации результата.

Входные и выходные данные

Можно представить панель **Graph**, как панель, в которой слева направо проходит электрический ток. Когда ток достигает последнего нода, нода материала, панель **Graph** создает комбинацию всех эффектов материала, демонстрируемую в игре. Нод материала является установочным для любого нового материала и содержит все атрибуты конечного материала (см. рис. 6.5).

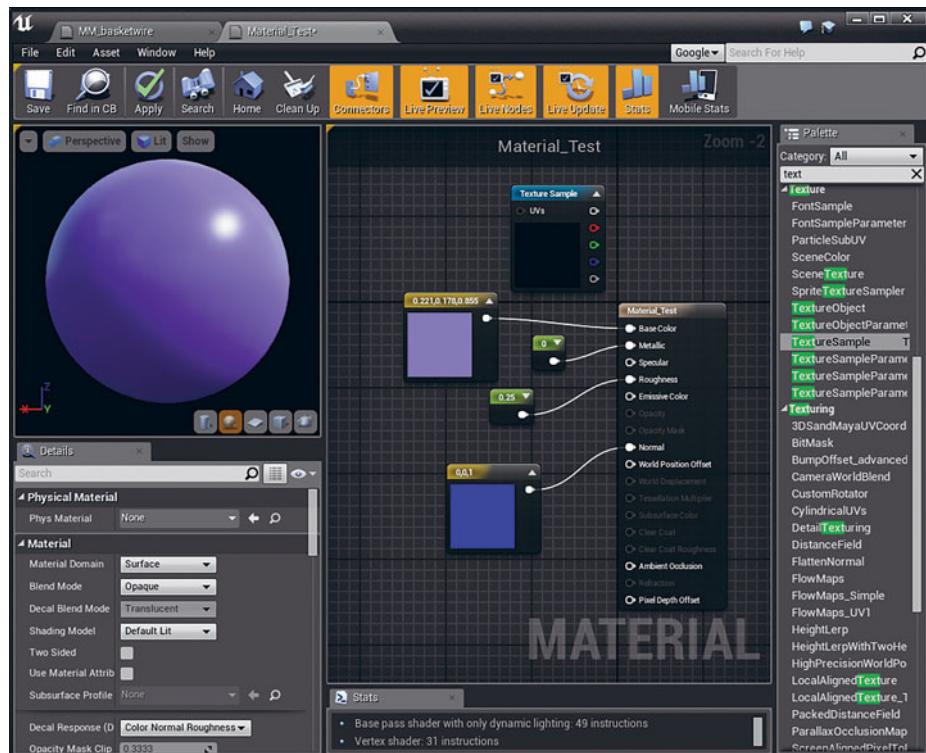


Рис. 6.5. Ноды, соединенные через ввод и вывод с нодом конечного материала

Каждый нод на панели **Graph**, для текстуры или нода, имеет выходные данные (вывод) и иногда входные (ввод) и соединяется с нодом конечного материала. Вывод нода находится с правой его стороны. Если нод имеет входные данные, то ввод находится слева. Вы можете соединять ноды друг с другом для создания различных эффектов, влияющих на итоговое представление материала. Чтобы соединить один нод с другим, просто щелкните по соединению вывода одного нода и перетащите видимое соединение к вводу другого нода.

ПРИМЕЧАНИЕ

Дополнительные ноды

При использовании редактора материалов могут отображаться плавающие ноды, которые используются для экспериментов, но они не должны появиться в итоговом варианте. Материал принимает в учет только ноды, соединенные с нодом материала. Все остальные ноды будут проигнорированы.

Ноды значений

Теперь, когда вы знакомы с элементами редактора материала, то можете создать простой материал, используя постоянные значения. Константы (constant values) — числа, которые могут задавать значения или цвета, в зависимости от количества используемых значений. Вы можете брать ноды значений из панели **Palette** и использовать их в материале в качестве входов.

Два наиболее часто используемых нода значений — это нод **Constant** и **Constant3Vector**. Нод **Constant** представляет единственное число или значение. Нод **Constant3Vector** представляет вектор или набор из трех чисел, каждое из которых представляет соответствующее значение RGB. Например, если нод **Constant3Vector** равен 1,4,6, это значит, что значение 1 для красного цвета, 4 — для зеленого и 6 — для синего.

Большинство часто используемых нодов в редакторе материала имеет соответствующие комбинации клавиш для быстрого размещения. Например, чтобы разместить простой нод **Constant**, зажмите клавишу **1** и щелкните правой кнопкой мыши по области панели **Graph**. Или щелкните по этим нодам на панели **Palette** и перетащите их в панель **Graph** для использования. После того как вы поместили один из нодов значений в панель **Graph**, вы можете щелкнуть по ноду и использовать панель **Details** слева, чтобы изменять значения, название и другие аспекты нода. Далее вы попрактикуетесь создавать материал с векторными значениями.

ПОПРОБУЙТЕ САМИ

Создайте материал с векторными значениями

Чтобы создать материал с векторными значениями, откройте созданный вами новый материал в редакторе материалов и выполните следующие шаги.

1. Создайте нод постоянного значения (**Constant**), нажав клавишу **1** и щелкнув правой кнопкой мыши на панели **Graph**.
2. Щелкните по выводу нового постоянного нода (**Constant**) и перетащите его, соединив с вводом шероховатости материала (**Roughness**).
3. Щелкните по ноду постоянного значения (**Constant**), чтобы увидеть информацию о ноде на панели **Details**.
4. На панели **Details** измените значение константы на **1** вместо значения по умолчанию **0** (обратите внимание на изменение шероховатости во вьюпорте).
5. Создайте нод **Constant3Vector**, удерживая клавишу **3** и щелкнув правой кнопкой мыши по пустому пространству на панели **Graph**.
6. Щелкните по ноду значения **Constant3Vector**, чтобы увидеть информацию о ноде на панели **Details**.
7. На панели **Details** измените значение **R** на **1**. (Если значения цветов не отображаются на панели **Details**, щелкните по стрелке слева от опции **Constant** на панели **Details**. Появятся варианты **R**, **G** и **B**, и вы можете поменять каждый из них, управляя значениями красного, зеленого и синего.)
8. Щелкните и перетащите вывод **Constant3Vector** в параметр базового цвета (**Base Color**) нода материала. (Обратите внимание, что цвет поменяется на чистый красный, как показано на рис. 6.6.)

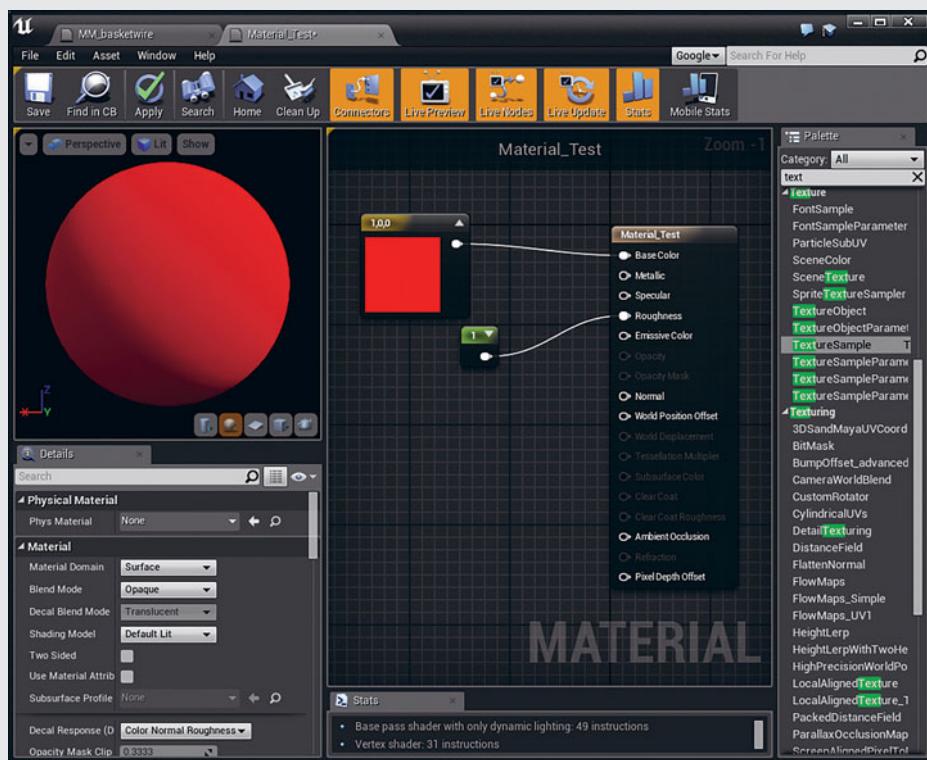


Рис. 6.6. Обратите внимание на изменения, которые происходят, когда нод материала получает новые входные данные

Экземпляры

Экземпляры имеют ключевое значение для многократного использования материала без необходимости каждый раз создавать новый материал. Изменив некоторые ноды в материале, вы можете создать в нем особые ноды, способные динамично меняться. Например, если основной материал выкрашен нодом **Constant3Vector** в зеленый, превратив нод в параметр, вы сможете легко менять цвет другого экземпляра этого материала, вместо того чтобы создавать новый с нуля.

Чтобы работать с экземплярами, главный материал должен приспособиться к динамичной трансформации определенных нодов, превратив их в параметризованные ноды. В редакторе материалов основного материала щелкните правой кнопкой мыши по любому ноду констант или векторов и выберите вариант

Convert to Parameter (конвертировать в параметр) в контекстном меню (см. рис. 6.7). Преобразовав материал в параметр, вы можете переименовать его на панели **Details**. Теперь любое значение или информация, которую вы внесете в основной материал на панели **Details** станут параметрами настройки по умолчанию для данного нода, пока вы не поменяете любой из параметров настройки в экземпляре материала. Чтобы создать экземпляр материала, щелкните правой кнопкой мыши по материалу в редакторе контента и выберите вариант **Create Material Instance** в выпадающем списке. Этот экземпляр материала использует параметры родительского материала, который мы создали.

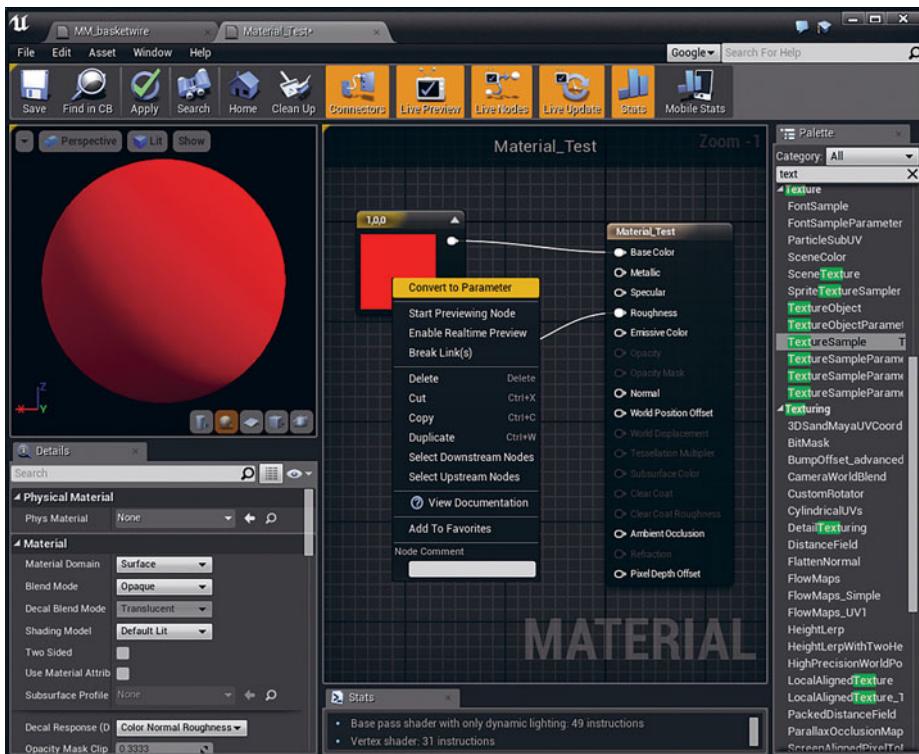


Рис. 6.7. Мы превратили константу в параметр. Теперь он может быть переименован или помечен. Тогда он будет доступен для динамичного изменения в экземпляре материала

Дважды щелкнув по экземпляру материала, который вы только что создали из основного, вы можете увидеть варианты настройки параметра в разделе **Parameter Groups** на панели **Details**. При установке флажка рядом с параметром, который вы хотите изменить, активируется параметр внутри экземпляра, и вы сможете менять любые детали этого параметра.

ПОПРОБУЙТЕ САМИ

Создайте экземпляр материала

Открыв редактор контента, выполните следующие шаги, чтобы создать экземпляр материала.

1. Откройте основной материал, который вы создали ранее в редакторе материала.
2. Щелкните правой кнопкой мыши по узлу **Constant3Vector**, который вы создали ранее, и выберите вариант **Convert to Parameter** в выпадающем списке.
3. На панели **Details** переименуйте узел **Constant3Vector** в **Color Param**.
4. Сохраните изменения и закройте редактор материала.
5. В редакторе контента щелкните правой кнопкой мыши по основному материалу и выберите вариант **Create Material Instance** в выпадающем списке.
6. Переименуйте новый экземпляр материала, который вы только что создали, на **Mat_inst** и затем дважды щелкните по нему в браузере контента, чтобы открыть его.
7. Найдите раздел **Parameter Groups** на панели **Details** и установите флажок **Color Param**. Параметр станет активным.
8. Поменяйте значения цвета следующим образом: для свойства **R** установите значение **0**, **G — 1** и **B — 0**. (Обратите внимание, что цвет изменился на зеленый, как показано на рис. 6.8.)

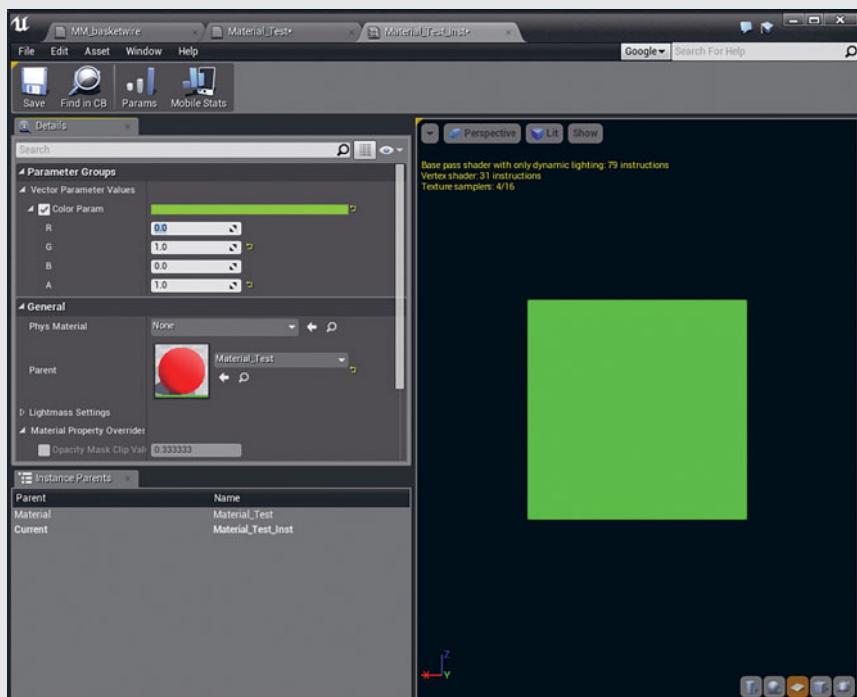


Рис. 6.8. В параметре материала вы можете выбрать цвет и изменить его на любой другой без изменения исходного материала

Резюме

В этом часе вы узнали, почему материалы являются важным элементом производственного конвейера, а также как они создаются и используются в UE4. Понимание PBR-системы может потребовать некоторой практики, но ее использование позволяет создавать более реалистичные и правдоподобные игровые миры, ассеты и персонажи. Использование экземпляров материалов и параметров помогает повысить скорость производства и ограничить расход памяти. Сокращая количество материалов в проекте с помощью продуманного создания экземпляров и параметров, вы можете сохранить время и энергию, создавать большие игровые пространства и добиваться более простого использования ассетов, имеющих схожую структуру материала. Хороший художник материалов и шейдеров находит пути для создания инновационных и повторно используемых материалов, которые будут легкоузнаваемыми и гибкими для множества экземпляров.

Вопросы и ответы

Вопрос: Необходимо ли использовать PBR-систему в UE4? Я привык к старой системе, и в редакторе материалов по-прежнему есть нод ввода свойств блеска.

Ответ: Да, использовать PBR-систему действительно необходимо. Большинство функций и структур приспособлены для PBR. Нод ввода свойств блеска имеет некоторое влияние на конечный результат, но не является таким же мощным, каким был в предыдущих версиях Unreal Engine.

Вопрос: Могу ли я создать и импортировать текстуру размером больше 4096 пикселей?

Ответ: Да, это возможно, но это весьма обескураживающе. Отображение текстуры больше 4096 пикселей в реальном времени — тяжелая задача не только для UE4, но и для большинства компьютеров, и вы можете обнаружить, что попытка отобразить столь большую текстуру приводит к проблемам с частотой кадров.

Вопрос: Почему некоторые вводы материалов затемнены и не могут использоваться?

Ответ: Структуры материалов используют особые вводы для генерирования конечного результата. Панель **Details** нода материала управляет тем, какие вводы видимы для соединения с выбранным типом материала. Например, если вы создаете стеклянный материал, вам могут понадобиться входные данные прозрачности, но если вы создаете кирпичную стену, входные данные прозрачности не нужны.

Вопрос: Могу ли я провести более одного вывода в один ввод нода?

Ответ: Нет, в один ввод каждого нода принимается только один вывод. Иногда ноды имеют несколько входящих нодов (математические функции и другие специальные ноды), но в таких случаях присутствует несколько слотов для ввода.

Вопрос: Могу ли я поменять материал, с которым связан экземпляр материала?

Ответ: Да, это возможно. На панели **Details** экземпляра материала можно изменить ссылку на другой материал. Помните, что в результате этого экземпляр материала обновится и изменится в зависимости от параметров внутри нового материала и может потерять часть информации, которая была установлена в его параметрах.

Вопрос: Могу ли я удалить основной материал и сохранить его экземпляр?

Ответ: Нет, экземпляр материала использует информацию из базового материала. Однако вы можете изменить ссылку на материал. Вам только понадобится найти материал, из которого экземпляр сможет получать информацию.

Вопрос: Если я изменю основной материал, изменится ли экземпляр?

Ответ: В этом и заключается все великолепие экземпляров материалов! При изменении одного материала, все его экземпляры обновляются. Если вы активируете параметр внутри экземпляра, он попытается сохранить значение, но обновит другие неактивированные параметры или добавит новые параметры, которые были добавлены позднее в процессе разработки.

Вопрос: Что представляют собой все остальные ноды на панели Palette?

Могу ли я использовать их для создания материалов?

Ответ: Эти ноды являются коллекцией математических уравнений, фигур и других специальных нодов, предназначенных помочь вам достичь особых результатов в материале помимо базовых постоянных значений и входных данных текстур, и вы можете свободно использовать их. Поэкспериментируйте, чтобы увидеть, какие ноды дадут вам желаемый эффект.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Как расшифровывается аббревиатура PBR?
2. Является ли размер текстуры 512×256 подходящим для использования в UE4?
3. Какие входные данные материала отвечают за его цвет?
4. Какая панель показывает изображение материала в редакторе материалов?
5. Почему создание экземпляров материалов так важно?

Ответы

1. PBR расшифровывается как physically based rendering — основанный на физике рендеринг.
2. Да, это подходящий для использования в UE4 размер. Несмотря на отличающиеся по размеру вертикаль и горизонталь, размер текстуры 512×256 по-прежнему соответствует правилам и может быть надлежащим образом уменьшен в UE4.
3. Входные данные базового цвета (**Base Color**) описывают цвет материала.
4. Выюорт (панель **Viewport**) показывает изображение материала в редакторе материалов.
5. Создание экземпляров материалов позволяет быстро множить их и вносить изменения в значения параметров без необходимости создавать полностью новый материал.

Упражнения

В этом упражнении вы создадите базовый материал, значения его шероховатости, базовый цвет и металлизированность, затем создадите экземпляр материала, чтобы управлять его вариациями. Создание материалов и экземпляров — важная часть управления тем, как выглядит и ощущается сцена и как свет взаимодействует с каждым аспектом ассета. То, как каждый ввод непосредственно влияет на визуальную составляющую игры, — важный аспект для понимания каждым разработчиком. Наконец, умение создавать экземпляры и параметры даст

максимальную гибкость и возможность многократного использования материалов в проекте.

1. Создайте новый материал в редакторе контента.
2. Дайте материалу название.
3. Откройте материал и создайте постоянные значения для шероховатости и металлизированности.
4. Создайте нод **Constant3Vector** для базового цвета.
5. Преобразуйте все постоянные ноды и нод **Constant3Vector** в параметры.
6. Переименуйте новые параметры и дайте им значения по умолчанию.
7. Создайте экземпляр из основного материала.
8. Активируйте и измените настройки параметров внутри экземпляра.

7-Й ЧАС

Использование элементов аудиосистемы

Что вы узнаете в этом часе

- ▶ Изучение основ аудио
- ▶ Использование звуковых актеров
- ▶ Создание звуковых сигналов
- ▶ Настройка звука с использованием аудиопространств

В этом часе вы узнаете о работе со звуком в Unreal Engine. Вы начнете с изучения базовых компонентов аудио в UE4 и научитесь помещать звуки на сцену с помощью звуковых актеров. Также вы узнаете о широких возможностях звуковых сигналов и о работе с редактором звуковых сигналов.

ПРИМЕЧАНИЕ

Подготовка к практике

Создайте новый проект с шаблоном **Third Person** и стартовым контентом.

Введение в основы работы со звуком

Вне зависимости от того, какую игру вы создаете, скорее всего, звук будет играть в ней важную роль. От звуков окружения на сцене до разговорных диалогов между персонажами и даже фоновая музыка — аудио в игре может значительно изменить ее восприятие. Большую часть времени игроки не замечают его, но звук играет большую роль во всем геймплее.

Компоненты аудио

UE4 содержит мощную систему работы со звуком, содержащую много компонентов и связанных с ней терминов. Новичкам она может показаться непреодолимой, но со временем вы освоитесь. Если на каком-то этапе вы почувствуете,

что слишком углубились в изучение аудио, переходите к более поздним урокам и затем возвращайтесь к этому, когда будете готовы узнать новые подробности.

Этот урок раскрывает несколько основных компонентов.

- ▶ *Ассет звуковой волны* (Sound Wave asset) представляет импортированный аудиофайл и параметры настройки, связанные с его воспроизведением и хранением.
- ▶ *Актер звуков окружения* (Ambient Sound Actor) используется для представления источника звуков на сцене.
- ▶ *Ассеты звуковых сигналов* (Sound Cue assets) и *редактор звуковых сигналов* (Sound Cue Editor) позволяют объединять звуки и модифицировать их, чтобы изменить конечный вывод.
- ▶ *Ассет затухания звука* (Sound Attenuation asset) определяет, как будет слышен звук, в зависимости от дистанции игрока по отношению к источнику звука.

Импорт аудиофайлов

UE4 поддерживает импорт несжатых файлов с расширением *.wav*. Если ваши исходные аудиофайлы в каком-либо другом формате, вы можете просто преобразовать их, используя имеющийся в свободном доступе звуковой редактор Audacity или любой другой. Компания Epic также предоставляет большое количество аудио-контента в примерах проектов, и магазин может стать для вас отправной точкой.

ПРИМЕЧАНИЕ

Audacity

Редактор Audacity можно загрузить по адресу: www.audacityteam.org.

Самый простой способ импортировать аудио — это перетащить файл с расширением *.wav* из файлового менеджера вашей ОС на панель **Content Browser** или щелкнуть по кнопке **Import** на панели **Content Browser**, найти и выбрать файл для импорта. После того как файл импортирован, вы можете дважды щелкнуть по аудиоассету, чтобы увидеть его свойства на панели **Details** в общем редакторе ассетов. Следующая рубрика «Попробуйте сами» проведет вас через процесс импорта файла с расширением *.wav*.

ПОПРОБУЙТЕ САМИ

Импорт аудиофайлов

На панели **Content Browser** создайте новую папку *MyAudio*. Затем выполните следующие шаги, чтобы импортировать файлы с расширением .wav.

1. Найдите в папке *Hour_07* (доступной по адресу: http://addons.eksмо.ru/it/UE_24.zip) файл *storm.wav*.
2. Щелкните по файлу *storm.wav* и перетащите его в папку *MyAudio* на панели **Content Browser**. Таким образом вы создадите новый ассет.
3. Повторите шаги 1 и 2 для файла *thunder.wav*, также находящегося в папке *Hour_07*. На рис. 7.1 показаны ассеты, которые вы только что импортировали — гроза и гром — наряду с ассетом **Steam01**.



Рис. 7.1. Ассеты звуковых волн на панели **Content Browser**

После того, как вы импортировали звуковой файл на панель **Content Browser**, вы можете редактировать свойства ассета звуковой волны в общем редакторе ассетов, дважды щелкнув по нему. Вы можете настраивать такие свойства, как сжатие, циклы звуковых ассетов по умолчанию и тон, и добавлять подзаголовок. И все же прямо сейчас менять ничего не нужно. На рис. 7.2 показана панель **Details** общего редактора ассетов.

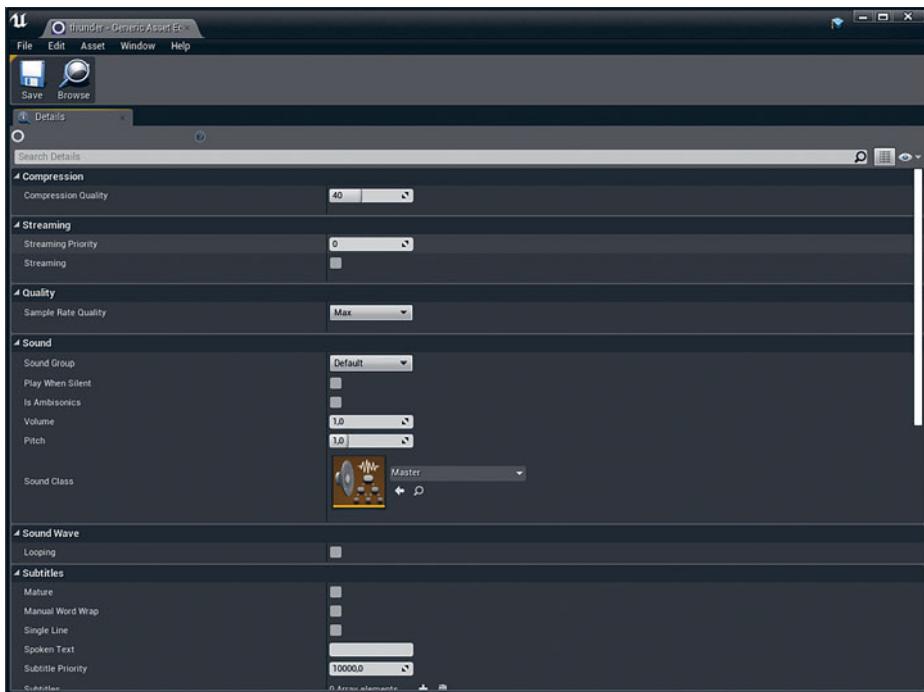


Рис. 7.2. Свойства звуковой волны

Использование звуковых актеров

Ассеты звуковых волн ни к чему без проигрывателя! Актеры звуков окружения — это компоненты, позволяющие воспроизводить звуки на уровне. Самый простой способ создать их — перетащить ассет звуковой волны на сцену. Вы можете создать множество актеров звуков окружения и дать им различные свойства. В табл. 7.1 перечислены свойства, доступные на панели **Details**, когда выбран актер звуков окружения.

ТАБЛ. 7.1. Свойства актера звуков окружения

Свойство	Описание
Sound	Указывает на ассет звуковой волны или ассет звукового сигнала
Is UI Sound	Определяет, продолжится ли воспроизведение звукового ассета, когда игра на паузе
Volume Multiplier	Устанавливает громкость всего звука

Свойство	Описание
Pitch Multiplier	Устанавливает тон всего звука
Instance Parameters	Позволяет добавлять параметры экземпляров звука
Sound Class Override	Выборочно добавляет звуковые ассеты в группы

ПОПРОБУЙТЕ САМИ

Поместите актер звуков окружения

Продолжая рубрику «Попробуйте сами», добавьте источник аудио.

1. Откройте панель **Content Browser** и найдите один из ассетов звуковых волн, которые вы импортировали в предыдущей рубрике «Попробуйте сами».
2. Щелкните по ассету и перетащите его на сцену.
3. Вы увидите ваш новый актер на панели **World Outliner** и его свойства на панели **Details**.
4. Щелкните по кнопке **Play**. Вы должны услышать звук, но вы не сможете определить его источник.

Установка затухания

Чтобы можно было определить положение звука в трехмерном пространстве, необходимо указать его затухание. **Затухание (Attenuation)** — это приглушение звука по мере удаления от него в трехмерном пространстве. В табл. 7.2 перечислены свойства затухания.

В следующей рубрике «Попробуйте сами» вы будете использовать параметр **Override Attenuation**, чтобы контролировать дистанцию, которую звук может преодолеть от актера.

ТАБЛ. 7.2. Свойства затухания

Свойство	Описание
Attenuation (группа свойств)	Включает возможность использования затухания посредством громкости
Attenuation Spatialization (группа свойств)	Включает позиционирование источника в трехмерном пространстве

Свойство	Описание
Attenuation Distance (группа свойств)	Указывает тип громкости по отношению к алгоритму дистанции для использования модели затухания
Attenuation Shape	Указывает форму громкости затухания (как правило, сферическую)
Inner Radius	Указывает полный размер громкости. За пределами этого радиуса звук не будет слышен
Falloff Distance	Указывает дистанцию, на которой происходит затухание
Non-Spatialized Radius	Указывает дистанцию, на которой начинается распространение звука

ПОПРОБУЙТЕ САМИ

Override Attenuation

Продолжая рубрику «Попробуйте сами», установите затухание.

- На панели **World Outliner** выберите из списка ваш актер звуков окружения.
- На панели **Details** в категории **Attenuation** установите флагок **Override Attenuation**. Вокруг актера на уровне появится желтый сферический каркас. Он представляет дистанцию, на которой будет распространяться звук.
- В разделе **Attenuation Distance** установите значение **200** для внутреннего радиуса (**Inner Radius**) и значение **50** для дистанции приглушения (**Falloff Distance**).
- Протестируйте уровень. Если вы стоите за пределами сферы затухания, вы не можете услышать звук.

Вы можете настраивать затухание для каждого актера звуков окружения. Вы также можете создать ассеты затухания звука, которые можно многократно использовать, и применить их к ассетам звуковых волн и актерам звуков окружения.

СОВЕТ

Распределение параметров настройки затухания

По мере роста проекта хорошим приемом является создать ассет затухания звука и распределить его между звуковыми актерами. Это упростит настройку большого количества аудиоисточников.

Использование свойств модуляции

Эффекты модуляции придают звукам движение и глубину. Параметры настройки модуляции позволяют контролировать минимальную и максимальную модуляцию тона и громкости, а также устанавливать мультиплликатор высокочастотного усиления. В табл. 7.3 перечислены и описаны свойства модуляции.

ТАБЛ. 7.3. Свойства модуляции

Свойство	Описание
Pitch Modulation Min	Указывает нижнюю границу использования при случайно определенном мультипликаторе тона
Pitch Modulation Max	Указывает верхнюю границу использования при случайно определенном мультипликаторе тона
Volume Modulation Min	Указывает нижнюю границу использования при случайно определенном мультипликаторе громкости
Volume Modulation Max	Указывает верхнюю границу использования при случайно определенном мультипликаторе громкости
High Frequency Gain Multiplier	Указывает мультипликатор высокочастотного усиления для звуков, генерируемых компонентом

Создание звуковых сигналов

До настоящего времени вы изучали, как применять ассет звуковой волны к актеру звуков окружения, но в каждой локации, где используется волна, может понадобиться и использование сигнала. Сигналы дают обширный контроль над аудио. Что если вы захотите случайно изменить звуки, например звуки шагов или шелест листвьев на ветру? Или если вы захотите применить модуляцию и другие эффекты? В таком случае в дело вступают звуковые сигналы. Редактор звуковых сигналов имеет следующие панели и кнопки (см. рис. 7.3).

- ▶ **Панель Graph.** Эта панель отображает поток аудио слева направо. Нод вывода, на котором изображен динамик, представляет конечный вывод.
- ▶ **Панель Palette.** Эта панель перечисляет различные звуковые ноды, которые можно перетащить в панель **Graph** и соединить вместе, чтобы создавать сложные звуки.
- ▶ **Кнопка Play Cue.** Эта кнопка панели инструментов воспроизводит весь звуковой сигнал, что эквивалентно воспроизведению нода вывода.

- ▶ Кнопка **Play Node**. Эта кнопка панели инструментов воспроизводит только аудио из выбранного нода (то есть часть сигнала, воспроизводимого кнопкой, описанной выше).



Рис. 7.3. Редактор звуковых сигналов

Чтобы открыть редактор звуковых сигналов, необходимо сначала создать ассет звукового сигнала. В следующей рубрике «Попробуйте сами» вы создадите новый объект звукового сигнала и затем добавите под **Wave Player**.

ПОПРОБУЙТЕ САМИ

Создайте звуковой сигнал

Продолжая рубрику «Попробуйте сами», вы создадите под **Wave Player** звукового сигнала.

1. На панели **Content Browser** щелкните по кнопке **Add New** или правой кнопкой мыши по пустому пространству области управления ассетами на панели **Content Browser**, чтобы открыть контекстное меню. В разделе **Create Advanced Asset** выберите пункт **Sound Cue** в списке **Sounds**.
2. Назовите новый звуковой сигнал **thunder** и дважды щелкните по нему, чтобы открыть редактор звуковых сигналов.
3. Чтобы добавить новый ассет грома, перетащите под **Wave Player** из панели **Palette** во виджет **Graph**.

4. На панели **Details** нода **Wave Player** выберите ваш звуковой ассет. Перетащите его контакт вывода в контакт ввода динамика. Результат должен выглядеть, как на рис. 7.4.
5. Воспроизведите звуковой сигнал с помощью кнопки **Play Cue** на панели инструментов редактора звуковых сигналов.
6. Перетащите ваш звуковой сигнал из панели **Content Browser** на сцену и просмотрите уровень.

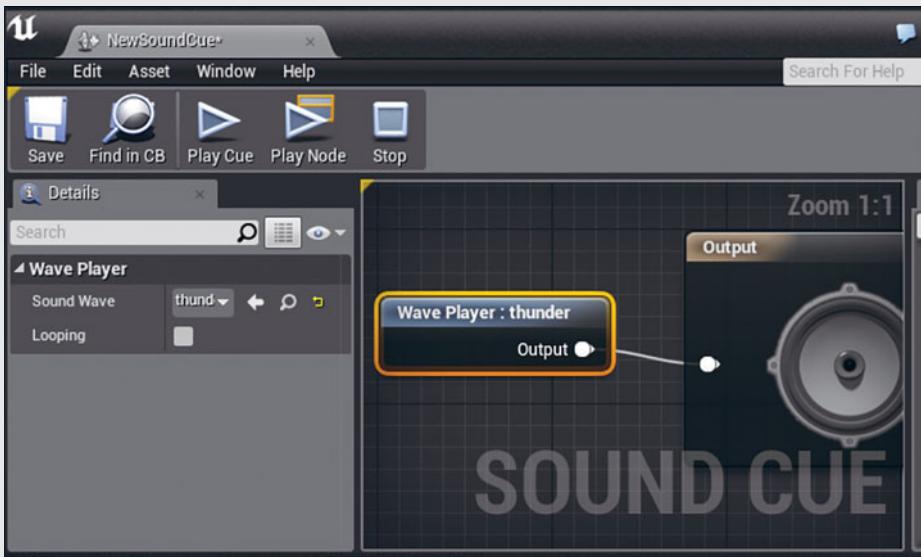


Рис. 7.4. Редактор звуковых сигналов, воспроизводящий звук

Во время воспроизведения звукового сигнала в целях упрощения отладки провода задействованных в данный момент нодов становятся красными. Это упрощает отслеживание конструкции звуковых сигналов в реальном времени.

ПОПРОБУЙТЕ САМИ

Смешайте звуки в звуковом сигнале

Продолжая рубрику «Попробуйте сами», постройте звуковой сигнал с нодом-микшером, чтобы создать атмосферный эффект грозы.

1. Откройте звуковой сигнал, который вы создали в предыдущей рубрике «Попробуйте сами», добавьте второй нод **Wave Player** и присвойте файл *Storm.wav*, импортированный ранее в этом часе.
2. Убедитесь, что для каждого нода **Wave Player** установлен флајок параметра **зацикливания Looping**.
3. Перетащите нод-микшер (**Mixer**) из панели **Palette** в редактор звуковых сигналов.
4. Перетащите контакты вывода из каждого нода **Wave Player** в контакты ввода микшера.
5. Перетащите контакт вывода из микшера в контакт выходного нода (динамика).
6. Протестируйте и сохраните ваш звуковой сигнал. По окончании звуковой сигнал должен выглядеть, как на рис. 7.5.
7. Просмотрите уровень.

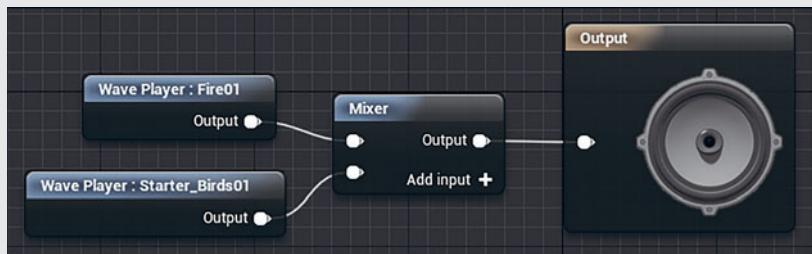


Рис. 7.5. Звуковой сигнал, в котором смешаны два звука

Продвинутые звуковые сигналы

Со звуковыми сигналами вы можете достигать невероятно сложного поведения, уходящего далеко за пределы этого урока. Следующим шагом правильно заняться чтением документации и примеров от Epic, включающих детальную информацию о каждом доступном типе нодов. На рис. 7.6 показан пример одного из продвинутых звуковых сигналов.

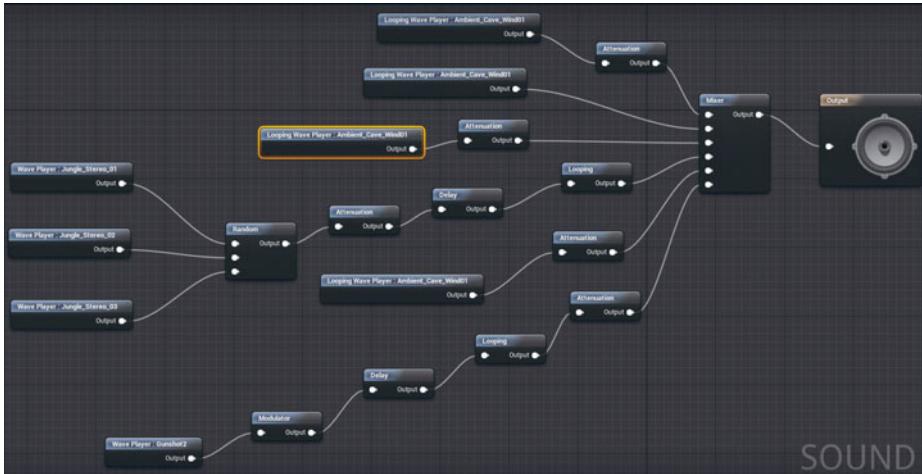


Рис. 7.6. Звуковой сигнал, в котором смешаны звуковые волны с различными свойствами, включая затухание, рандомизацию, зацикливание и задержку

Настройка звука с использованием аудиопространств

Аудиопространства (Audio volumes) не являются звуковыми ассетами, но они могут использоваться для управления различными звуками на сцене. Вы также можете использовать их определения зон, где звуки могут быть слышны. Например, аудиопространство в небольшом туннеле может иметь эффект реверберации, имитирующий настоящее эхо, которое вы ожидаете услышать в подобном туннеле.

Эффекты реверберации позволяют контролировать эффекты реверберации, эха, абсорбции воздуха и другие. Можно легко настраивать и применять их к любому аудиопространству на уровне.

ПОПРОБУЙТЕ САМИ

Работа с аудиообъемами

В этой рубрике «Попробуйте сами» вы создадите замкнутое аудиопространство с эффектами реверберации. Начните с одной из предыдущих сцен этого урока, не имеющих каких-либо закрытых пространств, и выполните следующие шаги.

1. В редакторе уровней добавьте аудиопространство на сцену, выберите пункт **Modes** ⇒ **Volumes** ⇒ **Audio Volume** и перетащите аудиопространство на уровень. Желтый каркас обозначает пределы распространения звука.
2. Выделите актер аудиопространства на уровне и на панели **Details** щелкните по параметру **Reverb Effect**, чтобы добавить новый эффект реверберации.
3. В появившемся меню выберите вариант **Reverb Effect** в разделе **Create New Asset** и назовите новый эффект **MyEffect**. На панели **Content Browser** появится новый эффект реверберации.
4. На панели **Content Browser** дважды щелкните по созданному ассету **MyEffect**, чтобы открыть общий редактор ассетов.
5. На панели **Details** общего редактора ассетов наведите указатель мыши на каждый параметр в списке параметров реверберации и прочитайте их краткое описание. Чтобы создать простой эффект эха/деформации, установите очень низкое значение параметра плотности звука (**Density**) и очень высокое значение параметра усиления отражения (**Reflection Gain**). Поэкспериментируйте с другими значениями, как показано на рис. 7.7. Когда вы закончите, щелкните по кнопке **Save** на панели инструментов.
6. Во выпорте редактора уровней щелкните по кнопке **Play** и обратите внимание на воспроизводимые звуки, в том числе звуки грозы. Теперь войдите в пространство и послушайте еще. Теперь у вас есть представление о том, что можно делать с аудиопространствами.



Рис. 7.7. Параметры реверберации

Резюме

В этом часе вы узнали, как работать со звуком в UE4. Вы начали с изучения основ аудио и компонентов, необходимых для работы. Затем рассмотрели актер звуков окружения. Вы узнали, как тестировать аудио и использовать затухание. Вы узнали о редакторе звуковых сигналов и создали свой первый звуковой сигнал. В конце часа вы изучили аудиопространства.

Вопросы и ответы

Вопрос: Поддерживает ли UE4 двухмерное аудио?

Ответ: Несомненно! Двухмерное аудио — это, как правило, звук без затухания, которое было раскрыто в этом уроке. Для блюпринтов существует специальный нод **PlaySound2D**, который наилучшим образом подходит для двухмерного аудио и аудио пользовательского интерфейса.

Вопрос: Могу ли я использовать другую форму, помимо сферы, при использовании затухания?

Ответ: Да. В параметрах настройки **Attenuation Distance** для актера звуков окружения найдите свойство **Attenuation Shape** и выберите один из вариантов.

Вопрос: Меня не устраивает, как затухает звук в моей игре. Что я могу сделать, чтобы изменить способ затухания?

Ответ: В параметрах настройки **Override Attenuation** для актера звуков окружения есть свойство **Distance Algorithm**, который можно установить, чтобы изменить способ звучания при затухании.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Истинно или ложно высказывание: зациклить звук можно только в ассете звукового сигнала.
2. Истинно или ложно высказывание: чтобы импортировать звук, он должен быть несжатым файлом с расширением **.wav**.
3. Истинно или ложно высказывание: если вы хотите добавить эффект реверберации, вы можете использовать аудиопространство

4. Истинно или ложно высказывание: вы можете смешивать между собой звуковые сигналы.
5. Истинно или ложно высказывание: если вы используете актер звуков окружения для воспроизведения фоновой музыки на уровне, вы можете захотеть добавить эффект затухания.

Ответы

1. Ложь. Вы можете зациклить воспроизведение любого ассета звуковых волн в общем редакторе ассетов. На панели **Content Browser** дважды щелкните по ассету звуковой волны, чтобы открыть общий редактор ассетов.
2. Истина. Файлы с расширением .wav являются общепринятыми аудиофайлами.
3. Истина. Вы можете поместить на уровень столько актеров аудиопространств, сколько потребуется, и применить различные эффекты реверберации к каждой.
4. Истина. Ассеты звуковых сигналов и редактор звуковых сигналов дают возможность смешивать и изменять ассеты звуковых волн.
5. Ложь. Как правило, фоновая музыка должна звучать для игрока одинаково на любом отрезке уровня. Поэтому вы не захотите применять затухание.

Упражнения

В одном из упражнений «Попробуйте сами» в этом часе вы создали ассет звукового сигнала, в котором были смешаны две звуковые волны. Однако зацикленный звук грома, который вы в него поместили, был нереалистичен. Как можно улучшить его? В этом упражнении вы улучшите качество эффекта, используя ноды задержки и зацикливания.

1. Откройте звуковой сигнал, который вы создали в упражнении «Смешайте звуки в звуковом сигнале».
2. Перетащите ноду задержки (**Delay**) из панели **Palette** во вьюпорт панели **Graph**. Соедините его с **Wave Player**: контакт вывода грома соедините с нодом **Delay**. На панели **Details** редактора звуковых сигналов установите значение 1 для параметра **Delay Min** и значение 5 для параметра **Delay Max**.

3. Перетащите нод зацикливания (**Looping**) из панели **Palette** во вьюпорт панели **Graph**. Соедините контакт вывода задержки с нодом зацикливания.
4. Соедините контакт вывода нода зацикливания с существующим нодом микшером. По окончании ваш звуковой сигнал должен выглядеть, как на рис. 7.8.

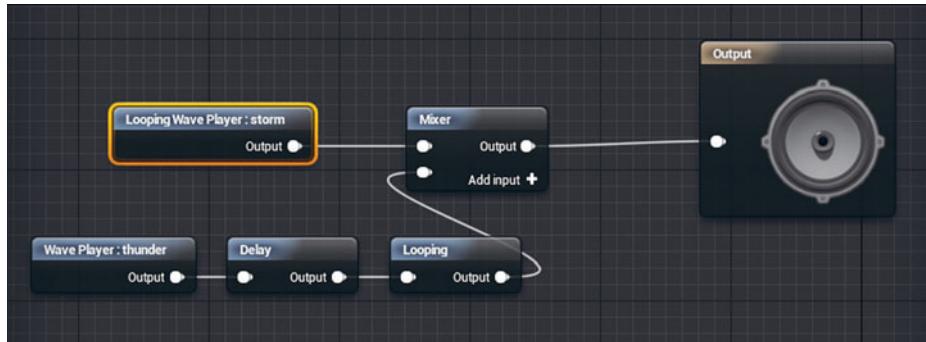


Рис. 7.8. Такой звуковой сигнал имеет случайную задержку между циклами звука грозы

5. Протестируйте уровень.

8-Й ЧАС

Создание ландшафтов и растительности

Что вы узнаете в этом часе

- ▶ Работа с инструментами и параметрами настройки ландшафтов
- ▶ Использование карт высот
- ▶ Как использовать ландшафтные материалы
- ▶ Как пользоваться инструментами и параметрами настройки растительности

В этом часе вы узнаете, как создавать и использовать ландшафты. После создания собственного материала ландшафта вы используете инструменты для окрашивания и наложите результат на новый ландшафт. Затем вы сфокусируетесь на разработке ассетов растительности и поместите их в игровое пространство с помощью инструментов создания растительности UE4.

ПРИМЕЧАНИЕ

Подготовка к практике

Создайте новый проект с шаблоном **Third Person** и стартовым контентом.

Работа с ландшафтами

При работе с новым проектом вы можете столкнуться с тем, что статичные меши не покрывают пространство, которые вы хотели бы сделать доступным для исследования игроком, особенно если вы создаете свободные для путешествий внешние пространства. В таких ситуациях нужно использовать инструменты создания ландшафтов. Они довольно мощны и допускают создание мира, который может быть отредактирован посегментно, то есть вы можете производить быстрое редактирование для расширения пространства игры и сделать игру с эффективным рендерингом.

Ландшафтные инструменты

Для создания и редактирования ландшафтов и их параметров в UE4 доступно множество инструментов. Доступ к элементам управления по умолчанию находится в левой части экрана на панели **Modes**. По щелчку по кнопке **Landscape** — центральной кнопке с иконкой горы (см. рис. 8.1) — откроется панель **Landscape**. Вы также можете нажать комбинацию клавиш **Shift+3** для быстрого доступа к панели **Landscape**.



Рис. 8.1. Панель **Landscape** можно открыть, щелкнув по кнопке **Landscape** на панели **Modes** или нажав комбинацию клавиш **Shift+3**

На панели **Landscape** доступны три главные вкладки:

- ▶ **Manage.** Эта вкладка контролирует аспекты конструирования и управления ландшафтами;
- ▶ **Sculpt.** Эта вкладка меняет пространство и форму ландшафтной геометрии;
- ▶ **Paint.** Эта вкладка управляет типами материалов, примененными к поверхности ландшафтов.

Вкладки **Sculpt** и **Paint** затемнены и не могут быть использованы, пока вы не создадите свой первый базовый ландшафт, где станут доступны эти опции.

Вкладка **Manage**

Первый раздел панели **Landscape** — это раздел управления (**Manage**). С его помощью вы можете создавать новые ландшафты и управлять существующими. Вы можете создать ландшафт двумя способами: новый ландшафт, основанный на наборе параметров, или ландшафт, основанный на импортированной карте высот. В этом уроке раскрывается создание ландшафта с нуля.

Карты высот

Карта высот — это текстура, предоставляющая информацию о разности высот, выполненная в оттенках серого. Белые поверхности на текстуре несут информацию ландшафту, что высота возрастает, в то время как черные пиксели сигнализируют о снижении высоты. Карта высот (*height map*) аналогична картам высот (*bump map*) в других программах трехмерной графики. Эти текстуры могут быть созданы во многих программах для моделирования и обработки изображений

и затем импортированы и использованы в UE4, или могут создаваться напрямую в UE4.

Карта высот полезна, когда вы воссоздаете определенную локацию реального мира и полностью имитируете реальный ландшафт внутри игрового пространства. Также вы можете использовать карту высот как маску, чтобы уведомить UE4, какие области имеют указанные типы растительности или материала ландшафта. Чтобы использовать карту высот в качестве маски, необходимо экспортировать ее из редактора и повторно сохранить в файле с подходящим расширением во внешнем приложении, таком как Photoshop. После этого вы сможете импортировать карту как текстуру и использовать в качестве маски.

Поскольку в этом уроке не раскрывается, как использовать внешние карты высот, чтобы изучить создание ландшафтов, поскольку вы создаете ландшафт в UE4, вы создаете внутреннюю карту высот в UE4, используя инструменты из вкладки **Sculpt**. Вы можете даже экспортить карту высот уже созданного в UE4 ландшафта, чтобы редактировать ее в других программах, использующих формат текстур. Форматы текстур — это часто используемые типы текстур, которые сохраняют перепады и подробности карты высот с наименьшими потерями данных.

Создание ландшафтов

При создании нового ландшафта у вас есть несколько начальных вариантов. Во-первых, необходимо определить, какой материал будет использоваться для ландшафта. Выше мы рассмотрели материалы в разделе «Материалы ландшафтов», но обратите внимание, что впервые материал и слои материалов могут быть прикреплены к новому ландшафту.

После материалов и слоев материалов следует настройка трансформации нового ландшафта (см. рис. 8.2). Здесь вы можете указать, куда поместить новый ландшафт в пространстве мира, а также его размер и угол поворота.

Следующий раздел элементов управления ландшафтами посвящен техникам уровней детализации (LOD, level of detailing) ландшафтов, которые вы можете изменять, чтобы ландшафт отображался быстро и эффективно. Этими элементами управления являются размер секции (Section Size) и количество секций в компоненте (Sections per Component), они связаны с тем, сколько информации отображается игроку одновременно. Чем больше размер секции, тем меньше отображается внутри компонента или секции, что снижает нагрузку на процессор. Чем больше секций в компоненте, тем лучше UE4 распределяет и решает, какое качество отображать в каждой секции. В этом заключается тонкий баланс между обеспечением высокой частоты кадров и желаемым разрешением ландшафта.

Экспериментирование в данном случае является ключевым, поскольку не существует простого ответа на вопрос об оптимальных числах для каждого конкретного проекта.

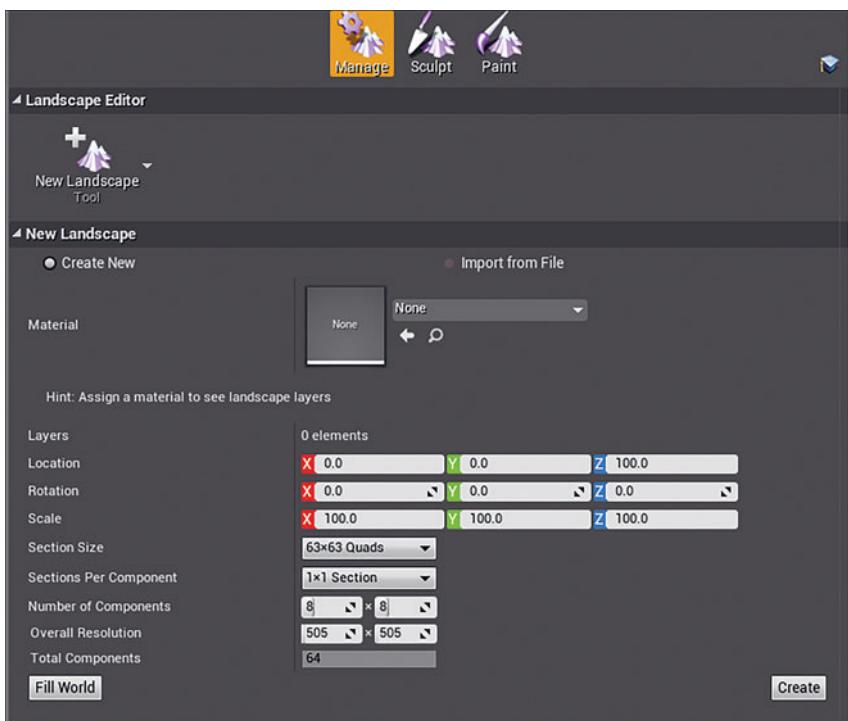


Рис. 8.2. Меню Manage на панели Landscape для создания нового ландшафта

Затем вы можете поменять параметры настройки плотности и размера каждого тайла ландшафта, которые используются для составления цельного ландшафта. В UE4 по умолчанию каждая подразделенная секция ландшафта или плана имеет плотность в 1 вершину на метр. По вертикали или оси Z, масштаб, по умолчанию имеющий значение 100, диапазон высоты составляет 256 метров вверх или вниз. Эти измерения важно знать при управлении и контроле плотности и размеров этих панелей для нового ландшафта. Если вам известны эти измерения, вы можете управлять количеством вершин всего ландшафта, используя настройки разрешения и количество всех сегментов ландшафта, включая настройки компонентов. Объединяя эти параметры настройки, вы можете находить баланс между разрешением и плотностью вершин на каждую секцию ландшафта.

ПРИМЕЧАНИЕ

Размер секций

Чтобы обеспечить приемлемую частоту кадров, размеру секций необходимо уделять больше всего внимания на панели **Landscape**. Чрезмерное увеличение размера секций приведет к снижению как частоты кадров, так и удобства использования редактором, поскольку потребуется больше вычислительных мощностей.

Две последние настройки для создания нового ландшафта — это кнопки **Fill World** и **Create** внизу панели **Landscape**. Кнопка **Create** подтверждает настройки и использует их для создания нового ландшафта. Кнопка **Fill World** создает ландшафт по всему доступному в текущий момент игровому пространству.

ПОПРОБУЙТЕ САМИ

Создайте новый ландшафт

Чтобы создать новый ландшафт, откройте панель **Landscape** и выполните следующие шаги.

1. Установите значение **31x31 Quads** параметра **Section Size**.
2. Установите значение **2x2 sections** параметра **Sections per Component**.
3. Установите значение **10x10** параметра **Number of Components**.
4. Щелкните по кнопке **Create**, чтобы создать новый ландшафт.

Управление ландшафтами

После создания нового ландшафта вам будут доступны новые опции для управления им. На вкладке **Manage** вы можете увидеть, что инструментом по умолчанию стала кнопка **Selection**, и выпадающее меню показывает новые варианты аспектов контроля только что созданного ландшафта. Некоторые из этих вариантов предназначены для удаления и добавления компонентов, эти варианты используются для изъятия секций из существующего ландшафта или добавления к нему новых. Другой вариант — **Change Component Size** — позволяет редактировать значения компонентов оригинального ландшафта, если вам нужно более высокое разрешение или если ландшафт отличается от того, каким вы его создавали изначально. Кроме того, вы можете использовать вариант **Move to Level** для перемещения секций на определенные уровни. Это более продвинутый вариант разработки для особых стриминговых областей или уровней, которые используются для снижения

мощности рендеринга, чтобы не отображать области, которые не нужны в данный момент. Наконец, у вас есть варианты для создания дополнительных ландшафтов, а также варианты контроля сплайнов, связанных с выбранным в данный момент ландшафтом.

ПРИМЕЧАНИЕ

Сплайны

Сплайны (splines) — это серии связанных точек, которые стремятся вверх ландшафта. Сплайны полезны для создания пластичных дорог, тротуаров и других структур, более естественно проходящих по поверхности ландшафта.

Скульптурирование форм и объемов

После создания нового ландшафта вы можете начать создавать формы и объемы на ландшафте. Вкладка **Sculpt** имеет три главных выпадающих меню: **Tool**, **Brush** и **Falloff** (см. рис. 8.3).

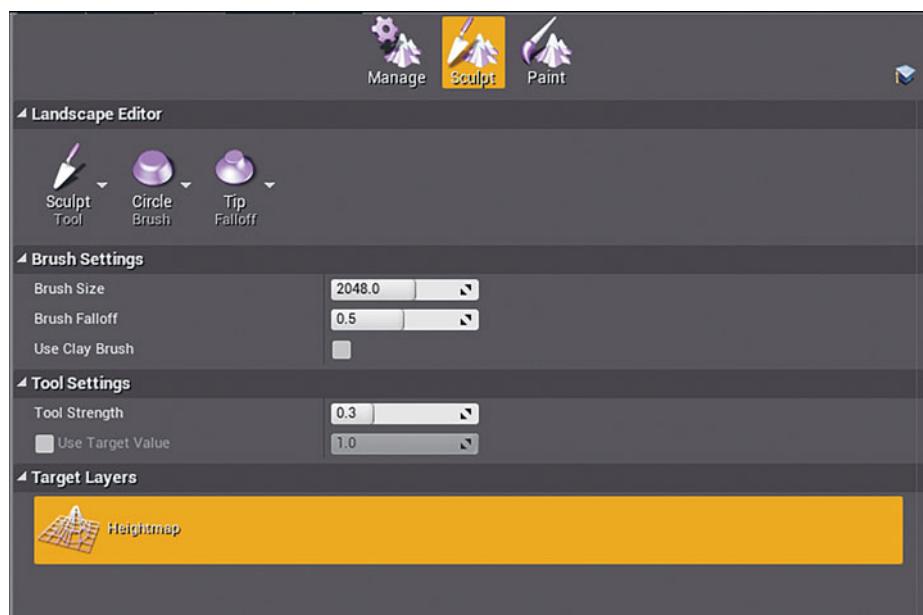


Рис. 8.3. Вкладка **Sculpt** на панели **Landscape**

Меню Tool

Меню **Tool** содержит инструменты для управления поверхностью и определением объемов ландшафта. Каждый инструмент имеет особые параметры настройки, связанные с типом инструмента.

- ▶ **Sculpt.** Этот инструмент скульптурирует выпуклые или вогнутые поверхности в ландшафтном меше.
- ▶ **Smooth.** Этот инструмент смягчает или сглаживает созданные инструментом **Sculpt** колебания высот между областями.
- ▶ **Flatten.** Этот инструмент выравнивает ландшафт до высоты, указываемой при первом щелчке по ландшафту с активированным инструментом **Flatten**. Он перемещает местность вверх или вниз, в зависимости от выбранного значения высоты.
- ▶ **Ramp.** Этот инструмент соединяет две области, наклоняя ландшафт между двумя областями с постоянным изменением наклона между точками.
- ▶ **Hydro Erosion** и **Erosion**. Эти инструменты моделируют общее изнашивание земли, которое происходит в мире, чтобы симулировать этот эффект на ландшафте игрового пространства.
- ▶ **Noise.** Этот инструмент применяет общий шум к ландшафту и использует параметры настройки для определения объема и интенсивности.
- ▶ **Retopologize.** Этот инструмент сокращает расстояние и различия между компонентами поверхности, чтобы уменьшить растяжение.
- ▶ **Visibility.** Этот инструмент скрывает или делает видимыми выделенные поверхности в ландшафтном меше.
- ▶ **Selection.** Этот инструмент маскирует выделенные области ландшафтного меша.
- ▶ **Copy/Paste.** Этот инструмент позволяет выделить секцию ландшафта, чтобы скопировать и перенести аналогичные параметры настройки высоты в другую секцию.

Меню Brush

Меню **Brush** (Кисть) позволяет выбирать форму используемого инструмента. Существует четыре типа кистей.

- ▶ **Circle.** Эта кисть является базовой. Она имеет форму окружности.
- ▶ **Alpha.** Эта кисть использует указанную текстуру, как маску, использующую тона серого, как карта высот.

- ▶ **Pattern.** Эта кисть использует повторяющийся шаблон на протяжении всего ландшафта и ведет себя как маска для скульптурирования.
- ▶ **Component.** Эта кисть влияет на все элементы компонента скульптурируемой области.

Меню Falloff

Меню Falloff (Затухание) позволяет контролировать силу воздействия кистей на скульптурируемый ландшафт. Существует четыре типа затухания.

- ▶ **Smooth.** Это один из наиболее часто используемых типов затуханий. Он представляет собой легкую смесь сильной и слабой частей кисти.
- ▶ **Linear.** Это прямое постоянное затухание.
- ▶ **Spherical.** Затухание наиболее слабое в центре и усиливается до наиболее сильного по краям кисти.
- ▶ **Tip.** Затухание наиболее сильное в центре и ослабляется до наиболее рассеянного по краям.

Окрашивание

Вкладка Paint на панели Landscape используется для окрашивания слоев материалов в ландшафтном меше. Эта вкладка содержит множество настроек и инструментов, схожих с теми, что находятся на вкладке Sculpt. Как и на вкладке Sculpt, три главных инструмента вкладки Paint — это Tool, Brush и Falloff. Каждый из этих разделов использует те же правила и приложения, как на вкладке Sculpt, но применяет эти правила к окрашиванию материалов.

Материалы ландшафтов

Настройки ландшафтных материалов незначительно отличаются от настроек обычных материалов. Процесс создания материала для ландшафта тот же, что и для создания нового материала, за исключением, что смешивание слоев определяется с помощью специального нода в редакторе материалов, который называется **LandscapeLayerBlend** (см. рис. 8.4). С этим нодом различные текстуры могут использоваться и распределяться в специальные слои, вызываемые в меню настроек ландшафта. Чтобы использовать этот нод, добавьте его из панели **Palette** в правой части редактора материалов. Затем вы можете щелкнуть по ноду и щелкнуть по симолу + на ноде, чтобы добавить слои. Вы можете добавлять и использовать множество слоев.

Однажды помещенные в материал, текстуры, которые обычно объединены или смешаны в обычном материале, в данном случае направляются в слои нода

LandscapeLayerBlend и затем помещаются в соответствующий вывод конечного материала, такой как базовый цвет или нормаль. Вы должны внимательно вводить названия для этих слоев, тогда UE4 корректно определит слои в ландшафте.

ПОПРОБУЙТЕ САМИ

Создайте новый ландшафтный материал

Чтобы создать новый ландшафтный материал и установить собственные текстуры, выполните следующие шаги.

1. Создайте новый материал на панели **Content Browser** и назовите его **Landscape_Material_Test01**.
2. Включите возможность использования этого материала для ландшафтов: для этого на панели **Details** для этого материала установите флагок **Used with Landscape**.
3. Создайте нод **LandscapeLayerBlend** из панели **Palette** для нового материала.
4. Импортируйте желаемые текстуры для материала ландшафта. (В этом упражнении вы будете использовать примеры названий текстур **Dirt_01** и **Grass_01** для ясности.) Затем перетащите в новый материал желаемые текстуры из панели **Content Browser** или создайте образец текстуры. Текстуры базового цвета **Grass_01** и **Dirt_01**, наряду с другими текстурами (нормаль, шероховатость и т. д.), должны теперь быть внутри материала, как образцы текстур.
5. В ноде **LandscapeLayerBlend** установите количество слоев для использования в материале ландшафта. Чтобы сделать это, выберите нод **LandscapeLayerBlend** и обратитесь к панели **Details**. Щелкните по кнопке **+** рядом с вариантом **Layers**. Создайте два дополнительных слоя, поскольку вам нужно смешивать материалы в ландшафте.
6. Для только что созданных слоев на панели **Details** нода **LandscapeLayerBlend** используйте вариант **Layer Name**, чтобы назвать слои **Dirt** и **Grass**. Соедините текстуру базового цвета **Grass_01** со слоем **Grass** и базовый цвет **Dirt_01** со слоем **Dirt**.
7. Соедините нод **LandscapeLayerBlend** с вводом базового цвета материала. Для каждого типа текстур (нормаль, базовый цвет, шероховатость и т. д.) продублируйте нод **LandscapeLayerBlend**, который только что создали, и соедините его с материалом.
8. По умолчанию текстуры не отображаются во вьюпорте материала, поэтому, чтобы протестировать слои, обратитесь к ноду **LandscapeLayerBlend** и измените значение **Preview Weight** на любое значение больше 0 и меньше 1.

Ландшафтные материалы накладываются в слои иначе, чем обычные материалы, и они используют другую форму смешивания. Типы смешивания доступны в деталях нода **LandscapeLayerBlend** для каждого слоя. Существует три типа смешивания ландшафтных слоев.

- ▶ **LB_WeightBlend.** Это тип смешивания по умолчанию для любого типа ландшафтных слоев. Он смешивает аддитивное значение 0 или 1. Чем больше этого слоя окрашено на ландшафте, тем более видимым он становится.
- ▶ **LB_HeightBlend.** Этот тип смешивает слои на основе соответствующей карты высот, назначенной с вводом нода **LandscapeLayerBlend** слоя высот.
- ▶ **LB_AlphaBlend.** Этот тип подобен смешиванию обычных материалов в вершинах, в котором используется маска для разделения переходов текстур между слоями. При использовании определенной карты слои разделены и переход основывается на оттенках серого альфа-карты.

Соединяя текстуры с их слоями и затем с соответствующим конечным материалом, вы можете использовать материал в качестве материала ландшафта. В таком случае материал может быть применен к ландшафту с помощью панели **Details** нужного ландшафта. Когда текстура присоединена к слою, панель **Palette** отображает соответствующие слои, которые теперь можно выбрать и накладывать на ландшафт.

ПРИМЕЧАНИЕ

Ноды ландшафтного материала

В редакторе материалов существуют и другие ландшафтные ноды для того, чтобы помочь контролировать аспекты использования слоев и текстур. Экспериментируйте с этими нодами, чтобы увидеть, как они могут быть применены к ландшафту проекта.

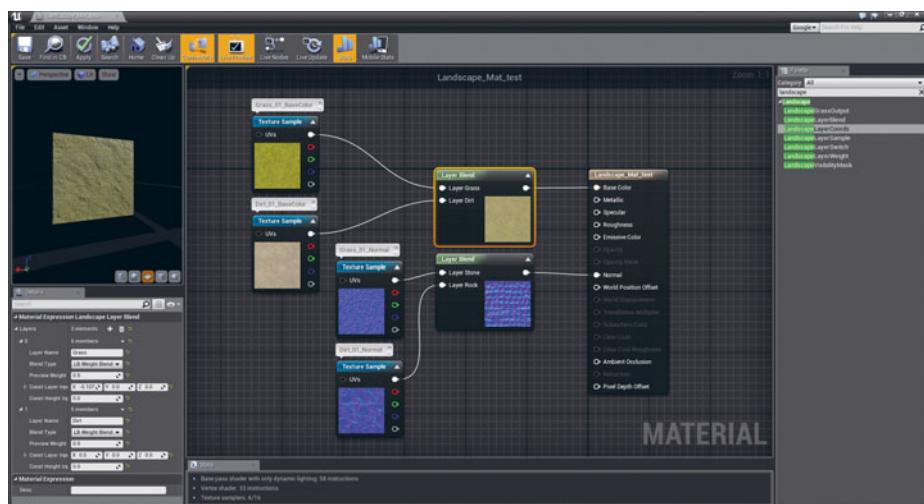


Рис. 8.4. Нод **LandscapeLayerBlend** в редакторе материалов позволяет контролировать множество ландшафтных слоев. Здесь представлено смешивание слоев двух типов материалов для окрашивания ландшафта

ПОПРОБУЙТЕ САМИ

Примените новый материал к ландшафту

После того как вы создали ландшафт и материал для него, вы можете применить материал к ландшафту, выполнив следующие шаги.

1. Выберите ландшафт и откройте панель **Details**.
2. Добавьте материал в секцию **Landscape** в области ландшафтного материала. Созданный вами ранее материал будет применен к ландшафту.
3. Чтобы окрасить созданные в материале слои, необходимо создать ландшафтные слои, для этого щелкните по кнопке **Landscape** на панели **Modes** (см. рис. 8.5).
4. Выберите вкладку **Paint** на панели **Landscape**.
5. Найдите среди вариантов **Target Layers** слои, созданные в вашем материале, который теперь связан с ландшафтом. Щелкните по символу + слева от каждого слоя в окне деталей, чтобы создать новый ландшафтный слой для каждого слоя в материале. Теперь каждый из этих слоев должен быть доступен для выбора и готов к использованию для окрашивания ландшафта.
6. Выберите целевой слой для окрашивания, щелкните по нему и перетащите в ландшафт, чтобы начать окрашивание. Выберите другой целевой слой, который хотите изменить, окрасив его.
7. Чтобы изменить размер кисти, ее затухание или другие параметры окрашивания, установите параметры в разделе **Target Layers** в настройках **Brush** или **Tool**.



Рис. 8.5. Панель **Landscape** и находящиеся на ней вкладки **Manage**, **Sculpt** и **Paint**

Использование растительности

Растительность (Foliage) — коллекция ассетов, помещенных на ландшафтный меш или другие ассеты на сцене и напрямую с ними связанных. Как правило, ассеты растительности представляют собой деревья, камни, траву, кустарники и другие ассеты, соединенные с ассетами под ними. Вкладка **Foliage** чрезвычайно полезна для быстрого помещения большого числа ассетов на сцену, вместо того чтобы использовать определенные параметры и ограничения в том месте, куда они могут быть помещены. Помещение полей и деревьев в большие открытые области вручную займет долгое время, но используя вкладку **Foliage**, вы сможете быстро поместить эти ассеты с помощью инструментов кистей. Актеры растительности могут представлять собой любой тип статичных мешей, доступных на панели **Content Browser**. Просто перетащив меш на вкладку **Foliage** (четвертая слева с иконкой листьев), вы сможете использовать его в качестве ассета растительной кисти (см. рис. 8.6).



Рис. 8.6. Вкладка **Foliage**

В левой части вкладки **Foliage** доступно пять следующих вкладок.

- ▶ **Paint.** Это наиболее часто используемая вкладка инструмента растительности, которая контролирует параметры окрашивания статичных мешей и то, на какие поверхности оно будет распространяться.
- ▶ **Reapply.** Эта вкладка применяет текущие настройки ко всем статичным мешам растительности, помещенным в данный момент на уровень. Она полезна, если изменения настроек статичных мешей растительности были сделаны после помещения их на сцену.

- ▶ **Select.** Эта вкладка позволяет выбирать определенные группы выборок статичных мешей растительности во всем пространстве мира.
- ▶ **Lasso Select.** Эта вкладка позволяет выбирать определенные группы выборок статичных мешей растительности.
- ▶ **Fill.** Эта вкладка позволяет заполнять выборки на сцене желаемыми статичными мешами растительности.

Параметры настройки вкладки **Paint** контролируют размер кисти или ее область влияния. Также на этой вкладке есть варианты плотности, контролирующие, насколько плотно ассеты будут размещаться при использовании инструмента. Наконец, флагки на вкладке **Paint** позволяют управлять тем, на какие элементы сцены будут влиять кисти. Например, если вы уберете флагок **Landscape**, кисти будут влиять только на BSP и статичные меши применяться только к статичным мешам на сцене.

Размещение растительности

Чтобы поместить растительность на сцену, необходимо добавить статичные меши на вкладку **Foliage** (см. рис. 8.7). Затем нужно выбрать и окрасить все статичные меши в группах или индивидуально, включая или отключая их. Чтобы включить статичный меш, щелкните по иконке ассета и установите флагок в левом верхнем углу; чтобы выключить статичный меш, уберите флагок. Все включенные ассеты участвуют в окрашивании, когда вы используете кисть растительности на сцене. Вы можете сохранить каждый статичный меш в этом списке для дальнейшего использования, щелкнув по символу сохранения в правом верхнем углу каждого статичного меша в списке. Теперь каждый ассет в списке имеет собственные параметры того, как они рассеиваются и используются в кисти. Каждый параметр настройки имеет значение по умолчанию, но если вы щелкните по статичному мешу из списка, вы можете включать опции, показанные ниже, расставляя флагки рядом с ними. Эти опции влияют на плотность, угол наклона, направление, масштаб и на многие другие факторы, которые воздействуют на то, как статичные меши будут размещаться на сцене.

Чтобы производить окрашивание с помощью выбранной растительности, щелкните в любом месте на сцене левой кнопкой мыши. Тем самым вы начнете помещать ассеты, основанные на настройках кисти для каждого статичного меша растительности. Чтобы удалить окрашивание, зажмите клавишу **Shift** и левую кнопку мыши и перетаскивайте курсор по тем окрашенным областям, в которых хотите отменить окрашивание. Чтобы полностью удалить статичный меш из инструмента **Foliage**, просто щелкните правой кнопкой мыши по желаемому ассету в типах вкладки **Foliage** и выберите вариант **Delete**. В этом случае все экземпляры меша будут удалены со сцены.

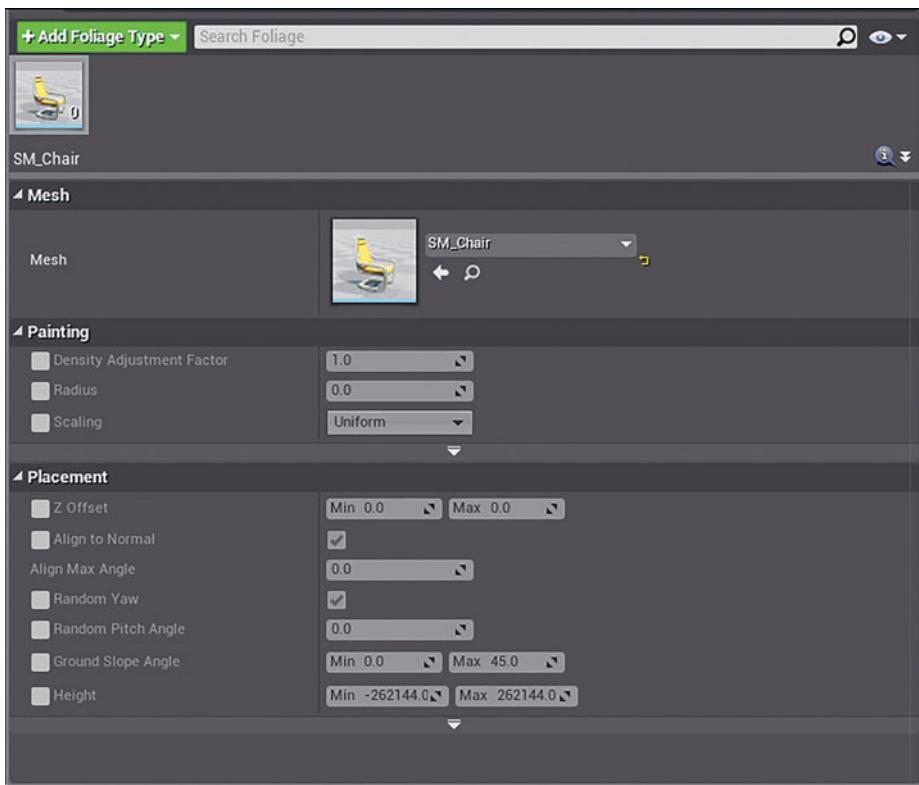


Рис. 8.7. Параметры настройки статичных мешей растительности

ПОПРОБУЙТЕ САМИ

Поместите растительность на сцену

Чтобы поместить растительность, откройте вкладку **Foliage** и выполните следующие шаги.

1. Перетащите актер статичного меша из панели **Content Browser** и поместите его в область **Drop Foliage Here** панели **Foliage**.
2. Измените значение плотности статичного меша растительности на **50**.
3. Измените значение радиуса на **2**.
4. Щелкните и окрасьте созданный ранее ландшафт.
5. Зажмите клавишу **Shift** и левую кнопку мыши и перетаскивайте курсор, чтобы удалить статичные меши растительности.

Резюме

В этом часе вы испытали инструменты создания ландшафтов, доступные в UE4. Вы узнали, как менять форму и поверхность ландшафтов. Вы также сформировали представление о том, как создавать слои ландшафтных материалов и наносить их на ландшафт. Затем вы изучили инструменты создания растительности и как использовать статичные меши, чтобы размещать растительность на сцене. Понимание этих инструментов увеличит ваши возможности создания широких просторов доступного игрового пространства и поможет вам быстро наполнить эти ландшафты деревьями, кустарниками и другими видами растительности.

Вопросы и ответы

Вопрос: Почему необходимо использовать ландшафт вместо серии статичных мешей?

Ответ: Для больших игровых пространств ландшафты отображаются более эффективно и дают разработчику максимальный контроль над всеми визуальными аспектами ландшафта одновременно. Статичные меши контролируются индивидуально и не предоставляют гибких средств настройки, как у ландшафтов.

Вопрос: Могу ли я выбрать статичные меши растительности после того, как поместил их на сцену?

Ответ: Да. При открытой вкладке **Foliage** вы можете выбирать размещенные статичные меши и трансформировать их независимо друг от друга.

Вопрос: Могу ли я использовать параметры материалов, такие как базовый цвет, нормаль, шероховатость и металлизированность для ландшафтов, так же, как и для стандартных материалов?

Ответ: Да. Когда вы создаете нод слоя для каждого ввода, убедитесь, что назвали каждый слой так же, как соответствующий нод. Например, название слоя грязи для параметров нормали, шероховатости и базового цвета должно быть таким же, как ноды ландшафтных слоев по порядку для их корректного функционирования вместе.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Можете ли вы использовать анимированный меш на вкладке **Foliage**?
2. После того как вы создали ландшафт, можете ли вы менять какие-либо настройки?
3. Как называется нод, который используется в материале для смешивания слоев ландшафтных материалов?

Ответы

1. Нет, можно использовать только статичные меши. Статичные меши могут иметь деформацию вершин их материалов, чтобы симулировать движение, но вы не можете напрямую использовать скелетные меши со своейственной им анимацией.
2. Да, вы можете менять настройки ландшафта в разделе **Change Component Size** панели **Landscape**.
3. Нод **LandscapeLayerBlend** используется для смешивания слоев ландшафтных материалов.

Упражнение

В этом упражнении вы создадите собственный ландшафт с новыми настройками и добавите вариации поверхности. Затем вы создадите ландшафтный материал с несколькими слоями и свяжете его с ландшафтом для использования с вкладкой **Paint**. Наконец, вы добавите статичный меш на вкладку **Foliage** и используете его для того, чтобы окрасить поверхность ландшафта. Это упражнение поможет вам понять, как с нуля создавать новый ландшафт и помещать материалы со слоями и мешами на его поверхность.

1. Создайте новый ландшафт.
2. Используйте вкладку **Sculpt**, чтобы скульптурировать его поверхность.
3. Используйте инструмент **Ramp**, чтобы смешать две области скульптурированной поверхности.

4. Используйте инструмент **Smooth**, чтобы смягчать области скульптурированной поверхности ландшафта.
5. Создайте новый слоистый ландшафтный материал.
6. Добавьте слоистый ландшафтный материал на ландшафт.
7. Окрасьте различные слои ландшафта.
8. Добавьте несколько статичных мешей в инструмент **Foliage**.
9. Произведите окрашивание каждым статичным мешем с помощью вкладки **Foliage**.
10. Произведите окрашивание всеми статичными мешами с помощью вкладки **Foliage** одновременно.

9-Й ЧАС

Создание мира

Что вы узнаете в этом часе

- ▶ Добавление уровня в проект
- ▶ Как украсить уровень
- ▶ Объединение размещенных актеров в блюпринт-класс

Цель этого часа заключается в том, чтобы упрочить ваше знание интерфейса основного редактора и изучить анатомию типичного уровня. Вы попрактикуетесь в создании мира и его украшении и примените знания, полученные в предыдущих уроках. Создание мира — это процесс помещения актеров и художественных ассетов на уровень. При создании мира границы между дизайнерами уровней и художниками среды размываются. (В некоторых производственных средах создание мира может быть обязанностью дизайнера уровней.) В этом часе вы создадите новый проект и поработаете с существующими ассетами, создадите новый уровень, поместите на него ассеты различных типов и познакомитесь с хорошими приемами создания мира.

ПРИМЕЧАНИЕ

Подготовка к практике

Создайте новый проект с шаблоном **Third Person** и стартовым контентом.

ПОПРОБУЙТЕ САМИ

Установите проект

В этом часе вам понадобится новый проект на основе шаблона от третьего лица. Выполните следующие шаги, чтобы создать проект.

1. Откройте лаунчер и загрузите главный редактор.
2. Выберите вариант **New Project**, чтобы создать новый проект.

3. На вкладке **Blueprint** выберите шаблон игры **Third Person**.
4. Выберите вариант **Desktop/Console**.
5. Выберите вариант **Maximum**.
6. Выберите вариант **With Starter Content**.
7. Щелкните по кнопке **Create Project**.

ПРИМЕЧАНИЕ

Пакеты контента

В этом часе вы можете использовать стартовый контент. Для этого найдите бесплатный контент в лаунчере в разделе **Learn** (Обучиться) или загрузите бесплатные пакеты **Infinity Blade** из магазина. Чтобы получить эти пакеты контента в лаунчере на вкладке **Store** (Магазин) напечатайте в строке поиска **Infinity** и среди результатов поиска найдите вариант **Infinity Blade**. Если вы хотите использовать контент из существующего проекта, необходимо перенести его в ваш проект. Если вы загрузите контент **Infinity Blade**, он будет добавлен в **Vault** (Хранилище) на вкладке **Library** (Библиотека), откуда вы сможете добавить его в свой проект.

Создание миров

Создание мира происходит по-разному, в зависимости от типа разрабатываемой вами игры, но основной процесс одинаков для любого производства. На данном этапе вы можете работать с существующими ассетами, не беспокоясь о геймплее или моделировании и текстурировании нового контента. Таким образом вы укрепите свое знание интерфейса редактора и сфокусируетесь на навыках создания мира.

Хорошо проработанный мир, при создании которого использовалось масштабирование, цвет, освещение, звук и размещение ассетов, создает настроение и вызывает желаемый эмоциональный отклик у игроков. Декорации должны не только хорошо выглядеть, но и помогать игрокам ориентироваться на уровне и создавать эффект погружения.

Повествование через окружение

Если вы работаете над игрой, у вас есть отличный шанс создать повествование, раскрывающееся местом действия, которое изображает уровень. Декорирование поможет сформировать ваше собственное визуальное повествование каждого создаваемого пространства. Прежде чем вы начнете создавать уровень, взгляните на все визуальные ассеты, которые есть для работы, и создайте простой рассказ, объясняющий, что произошло с создаваемым пространством до того, как игрок

попал в него. Позже это поможет вам принимать решения о размещении ассетов и освещения.

ПОПРОБУЙТЕ САМИ

Создайте уровень по умолчанию

Каждый проект начинается с карты по умолчанию, которая позволяет быстро тестировать игровой режим и элементы управления игрока. Вы уже создали проект и теперь должны создать в нем уровень.

1. На панели **Content Browser** создайте новую папку и назовите ее *Maps*.
2. В главном меню выберите вариант **File ⇒ New Level** (или нажмите комбинацию клавиш **Ctrl+N**).
3. В появившемся диалоговом окне выберите вариант **Default Level**.
4. Сохраните ваш новый уровень в папке *Maps*.

Анатомия уровня

На только что созданном уровне взгляните на панель **World Outliner**, и вы увидите, что карта по умолчанию уже имеет несколько размещенных актеров. Уделите некоторое время тому, чтобы выбрать каждый из них и посмотреть на их названия, типы и свойства на панели **Details**. Среди них есть актер статичного меша **Floor**, представляющий плоскость пола. Актер **Player Start** определяет, где появится аватар во время игры на этом уровне. Также на уровне есть два источника света: небесный ИС, охватывающий кубическую карту и дополняющий все освещение сцены, и направленный ИС, устанавливающий направление солнечных лучей. На уровне присутствует актер **Atmospheric Fog**, который имитирует рассеивание света через атмосферу планеты. Наконец, на нем есть блюпринт-класс **Sky Sphere**, который использует блюпринт для управления динамичным материалом на статичном меше; небесная сфера, задающая внешний вид неба на уровне. Если вы переключите выпорт на вид сверху и максимально уменьшите масштаб изображения, то увидите, что небесный купол масштабирован так, чтобы быть больше размеров сетки.

ПРИМЕЧАНИЕ

Направленные источники света

Местоположение направленного ИС не имеет значения, важна только его ориентация. Свет направленного ИС начинает свой путь из-за пределов мира и движется в сторону направленного ИС. Поскольку источник света бесконечно далек, он используется для того, чтобы имитировать солнце, и не имеет направленного затухания.

Процесс создания мира

Теперь, когда у вас есть установленный проект, давайте поговорим о процессе. Хорошой практикой при декорировании является поэтапная работа с использованием следующего повторяющегося процесса.

1. **Масштаб и пределы.** Вы должны установить масштаб и рабочее пространство уровня, как вы обычно делаете с простыми примитивными объектами.
2. **Создание слоев и наброска.** Это процесс обработки архитектурных и структурных потребностей уровня при помощи помещения на него архитектурных и структурных форм, таких как здания и стены или другие структурные элементы. Вы можете делать это с примитивными формами, прежде чем будут созданы конечные ассеты.
3. **Помещение декораций и ассетов.** Эта стадия включает помещение декораций и декоративных ассетов проекта, таких как скамьи, кусты или урны.
4. **Распространение света и звука.** На этой стадии вы помещаете источники света и актеры звуков окружения.
5. **Игровое тестирование и отладка.** Этот аспект связан с перемещением по уровню с точки зрения игрока и поиском, корректировкой и устранением любых проблем.

Повторяйте шаги 3–5 до тех пор, пока не достигнете нужного результата. В следующих разделах подробно описаны все части этого процесса.

Создание масштаба

При разработке уровня, задание соответствующего и эффективного масштаба среды — это один из наиболее важных аспектов создания настроения на уровне. Чтобы сделать это, необходимо знать размеры персонажа игрока. Помните, что в UE4 1 единица Unreal (ui) по умолчанию равна 1 сантиметру (см) в реальном мире, поэтому, если средний персонаж в вашей игре 6 футов в высоту, это значит, что его высота составляет 182,88 см (1,82 м) или 182,88 ui. Даже если конечный персонаж еще не создан, помещение временного визуального представления персонажа имеет большое значение для создания масштаба.

ПОПРОБУЙТЕ САМИ

Поместите ссылку на масштаб

Использование ассета персонажа или даже простой примитивной формы, представляющей размеры среднего персонажа, поможет установить масштаб уровня. Выполните следующие шаги, чтобы поместить скелетный меш, предоставленный игровым шаблоном **Third Person**.

1. Откройте уровень по умолчанию, который вы создали в предыдущей рубрике «Попробуйте сами».
2. На панели **Content Browser** откройте папку **Mannequin** ⇒ **Character** ⇒ **Mesh** и найдите ассет **SK_Mannequin**.
3. Перетащите ассет скелетного меша **SK_Mannequin** на уровень и поместите его на актер **Floor**.
4. Сохраните и просмотрите уровень. На рис. 9.1. показана ссылка на масштаб.

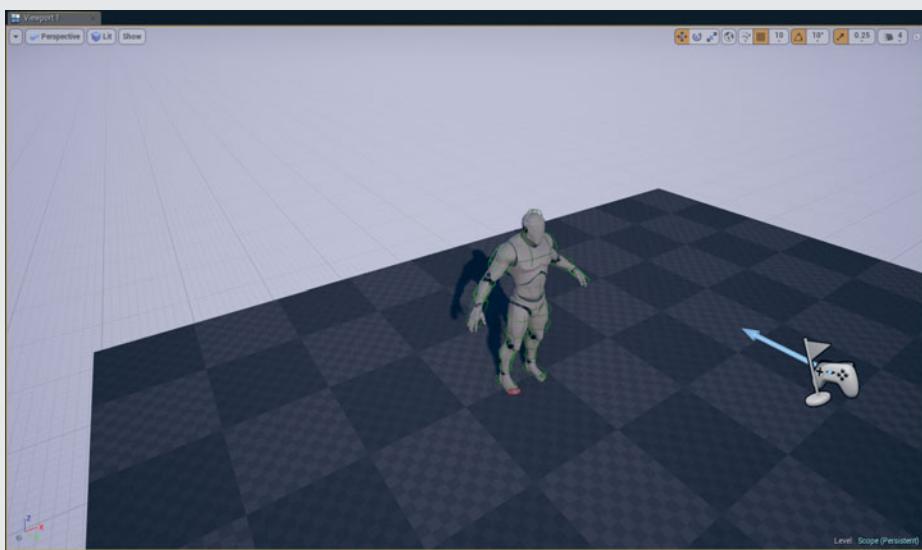


Рис. 9.1. Ссылка на масштаб персонажа

Создание пределов

После создания карты по умолчанию следует установить пределы уровня. Больше не значит лучше, но многие начинающие художники и дизайнеры делают распространенную ошибку, увеличивая свои проекты и непременно стараясь

создать огромные уровни, размерами как в ММО*-играх. Это часто приводит к разочарованиям и увеличению времени на решение любых неожиданно возникающих проблем. Лучше всего в этом случае следовать эмпирическому правилу, что в самом начале нужно предпочтительнее качество количеству. Когда вы будете лучше знать UE4 и редактор, вы сможете увеличивать сложность своих проектов.

В этом часе вы узнаете, как начать работу с установочным статичным мешем **Floor** и добавить некоторые незначительные дополнения. На панели **Modes** выберите вкладку **Place**. Выберите вариант **Geometry** из списка в левой части панели. Вы увидите список геометрических актеров, которые можете поместить на уровень. Щелкните и перетащите куб (Box) во виджерт. Выделите его и установите в разделе **Brush Settings** форму кисти по оси Y **440** единиц и по оси Z **50** единиц. Теперь поместите куб в центр одного конца статичного меша **Floor** так, чтобы нижний край куба был вровень с верхним краем пола. Эта конструкция будет рабочим объемом уровня (см. рис. 9.2). Это не очень грандиозно, но это хорошая отправная точка.

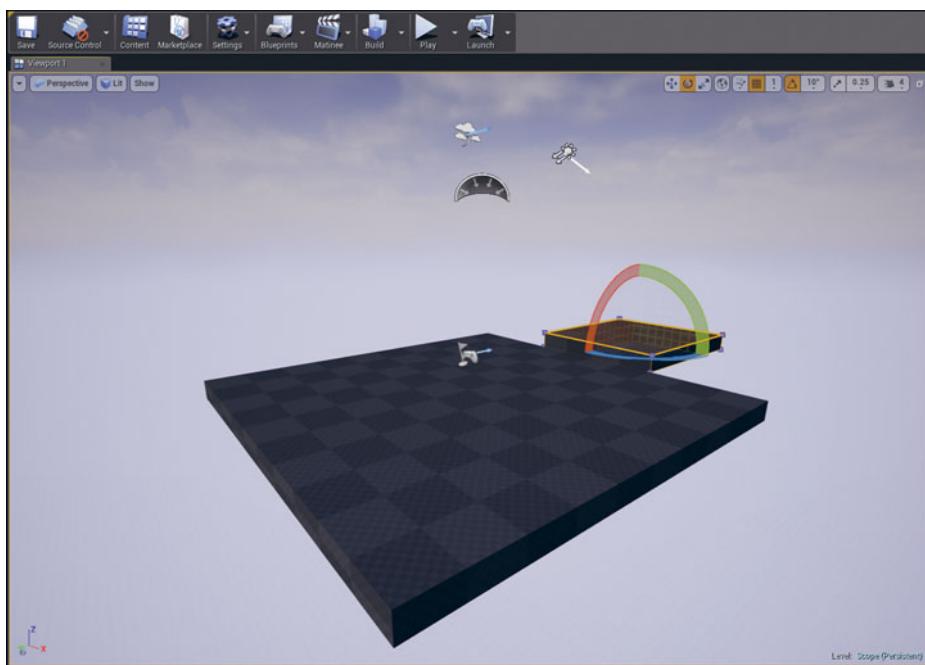


Рис. 9.2. Куб, помещенный на уровень для создания его рабочего объема

* Массовая многопользовательская онлайн-игра (Massively Multiplayer Online game) — вид сетевых компьютерных игр. — Прим. ред.

СОВЕТ

Работа с BSP-актерами

BSP-актеры — это процедурно генерируемые движком UE4 геометрические примитивы. Аббревиатура BSP расшифровывается как binary space partitioning — «двоичное разбиение пространства». Такие актеры представляют собой трехмерные геометрические формы, но обрабатываются иначе, нежели меши, импортированные из 3D-приложений. BSP-актеры хорошо подходят для быстрого чернового наброска уровня. Еще один термин, который можно услышать при разговоре о BSP, — конструктивная стереометрия (CSG, constructive solid geometry), который означает, что актеры представляют собой соприкасающиеся формы. Вы можете выполнять простые булевые операции по моделированию с множеством BSP-актеров, добавляя (**Add**) или вычитая (**Subtract**) их. Вы можете накладывать на них текстуры в редакторе с помощью простых плоских проекций, а также можно накладывать текстуры на каждый полигон BSP-примитива отдельно, перетаскивая материал из панели **Content Browser** на желаемую поверхность.

Вы можете выполнять другие простые задачи по моделированию на BSP-актерах, переключившись на вкладку **Geometry Editing** (редактирование геометрических форм) на панели **Modes**. Однако для более сложных задач следует воспользоваться приложением для моделирования и импортировать из него статичный меш. При работе с BSP-актерами лучше всего устанавливать их размеры прямо в разделе **Brush Settings** на панели **Details** и избегать их масштабирования.

Создание слоев и наброска

Настроив рабочее пространство, вы можете приступить к черновому наброску структурных ассетов уровня. На панели **Content Browser** в папке **Starter Content** ⇒ **Architecture** вы можете найти статичные меши, пригодные для работы. Выберите асset статичного меша **wall_door_400×300**, перетащите его на уровень и поместите на BSP-куб лицевой стороной к актеру **Floor**. Повторите действие с другими сторонами и крышкой, чтобы создать простую коробку, как показано на рис. 9.3.

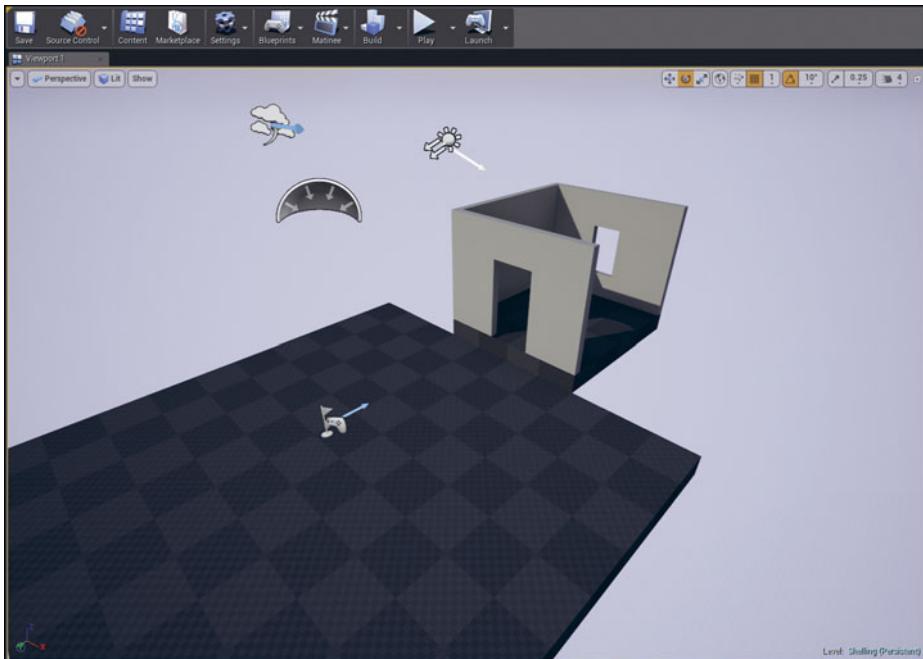


Рис. 9.3. Создание слоев и наброска уровня с помощью BSP-актеров

Когда небольшая комната готова, перетащите еще один кубический BSP-актер и начните создавать стены вокруг меша пола. Когда вы закончите, конечный набросок будет выглядеть, как на рис. 9.4.

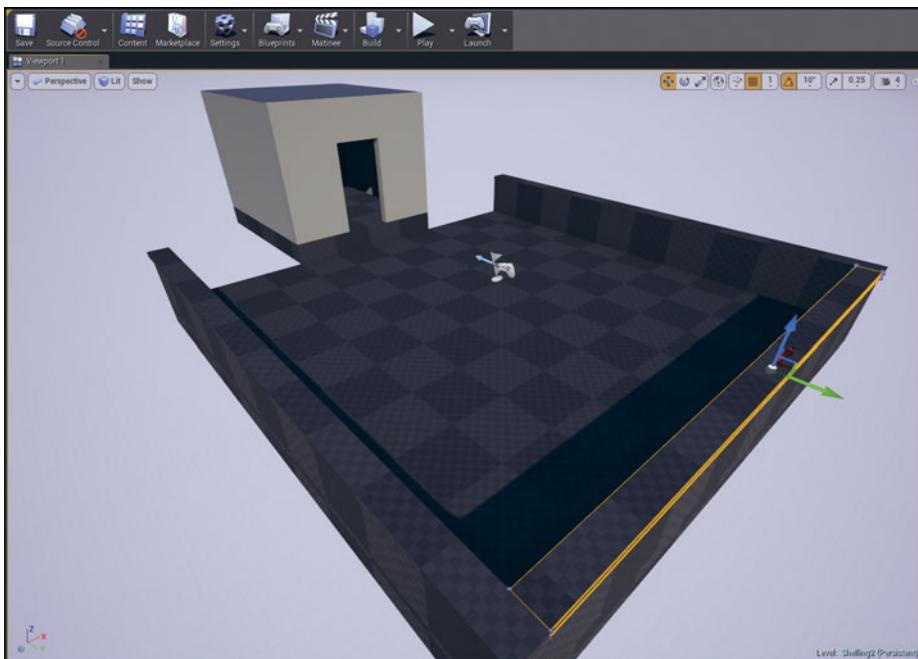


Рис. 9.4. Продолжение создания слоев уровня

СОВЕТ

Привязки и рабочее четырехное представление

Правильное размещение и выстраивание ассетов может быть утомительным. Вам может прийти на помощь включение функции привязки трансформации для выполнения перемещения, масштабирования или поворота. (Вы изучали этот вопрос в 3-м часе «Координаты, преобразования, единицы измерения и организация».) Также вам может помочь смена разметки окна представления на четырехпанельное, чтобы вы могли работать с видами сверху, сбоку и спереди при выстраивании ассетов. Чтобы изменить разметку вьюпорта, щелкните по небольшой иконке **Maximize/Restore** в крайнем правом углу настроек привязки (см. рис. 9.5).



Рис. 9.5. Привязка функций трансформации и сетки и включатель вьюпорта

Ниже перечислены четыре полезные панели настроек привязки, отмеченных на рис. 9.5.

1. Привязка к сетке при перетаскивании.
 2. Привязка объектов к сетке поворота.
 3. Привязка объектов к сетке масштабирования.
 4. Максимизация/восстановление выюпорта.
-

Помещение декораций и ассетов

После создания наброска и слоев уровня вы можете поместить на него декорации и ассеты. На панели **Content Browser** в папке **Starter Content ⇒ Props** вы можете найти ассеты, которые можно использовать для украшения уровня. Поскольку вы работаете в виртуальном пространстве, не стоит беспокоиться о структурной точности, но при этом вы не хотели бы, чтобы игроки могли заглянуть за декорации игры, для чего необходимо соблюдать баланс. Перетащите несколько экземпляров статичного меша **SM_Rock** из папки *Props*, разместите их вокруг комнаты за ее пределами и под полом, чтобы казалось, что комната встроена в гору. После этого присвойте материалы некоторым поверхностям. Добавьте любые меши для украшения пространства (см. рис. 9.5). Периодически производите просчет освещения по мере продвижения. Вы можете заметить, что чем больше вы добавляете контента, тем длительнее становится процесс просчета освещения.

Визуальная сложность и фрейминг

Визуальная сложность — это концепция нахождения баланса между визуальными деталями и использованием ассетов. Беспорядочное размещение большого количества ассетов на уровне может добавить деталей, но, скорее всего, создаст впечатление непроработанности и хаотичности. При размещении ассетов продумайте функцию пространства, основываясь на повествовании, и помещайте ассеты так, чтобы они определяли ключевые локации как покадровые моменты (см. рис. 9.6). *Фрейминг* (framing) — это идея создания мини-композиций в игровом мире. Эти области деталей могут привлечь внимание игрока и помочь направить их в пространстве. Попытайтесь создать интересные пространственные отношения, где игрок должен двигаться между открытыми и ограниченными пространствами так, чтобы привлекать его интерес и помочь определить области деталей.

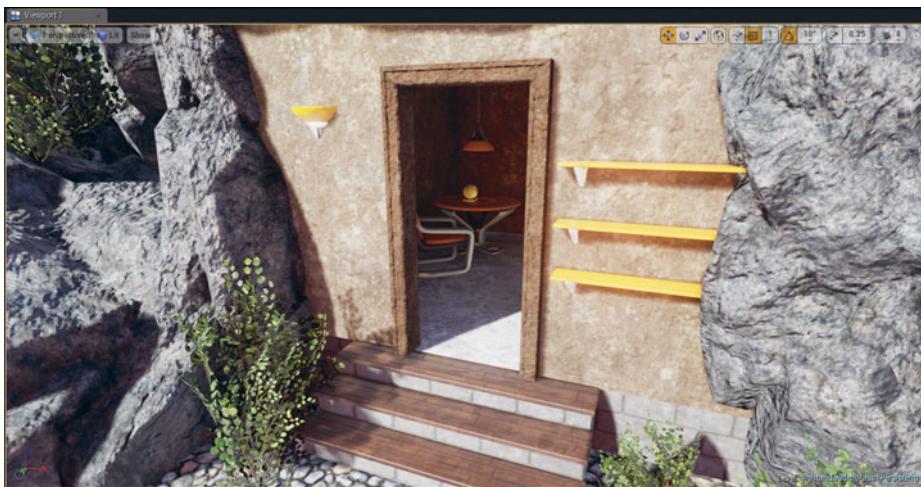


Рис. 9.6. Размещение декораций и ассетов с присвоенными материалами

Работа с модульными ассетами

Вы заметите, что многие ассеты статичных мешей в папке *Starter Content* предназначены для модульного моделирования. Хорошо смоделированные модульные ассеты опираются на последовательные единицы измерения с соответствующим образом расположеными якорными точками для простой привязки их друг к другу. Это помогает сократить время на декорирование, но это также повышает шанс чрезмерного повторения и однородности. Хорошее декорирование происходит быстро благодаря модульности, но оно требует маскировки повторного использования подобных ассетов для создания реалистичного игрового пространства.

СОВЕТ

Сетки и привязка

Оборотная сторона работы с модульными ассетами и привязкой сетки заключается в том, что вы рискуете сделать свой уровень слишком однородным или чрезмерно геометрически правильным. Чтобы избежать этого, следует сделать специальный проход на уровне, в котором будут произведены небольшие трансформации, где это применимо, чтобы обеспечить максимальную натуральность.

Объединение актеров в блюпринт-классы

При декорировании будет уместным объединение ассетов, которое можно выполнить несколькими способами. Как говорилось в 3-м часе, отличные способы сохранения организованности — это группировка и прикрепление актеров,

перемещение актеров в папку на панели **World Outliner** и объединение актеров в слои. Хотя эти методы быстрые и имеют преимущества, они уникальны для текущего уровня. Объединение актеров в блюпринт-классы дает преимущества как с точки зрения группировки, так и с точки зрения прикрепления. Объединяя таким образом актеры, вы получаете преимущество многократного их использования на любых уровнях, а также в итоге сможете автоматизировать и скриптовать ваши задачи.

СОВЕТ

Объединение ассетов

При декорировании вы можете не найти нужный ассет или не иметь свободного разработчика моделей для создания нового контента. Если проявить креативность, зачастую можно создать новый ассет из уже имеющегося контента, просто объединив ассеты в блюпринт-класс. После того как вы создадите такие ассеты и поместите их на уровень, вы можете перемещать, масштабировать и поворачивать их, как любые другие актеры.

ПОПРОБУЙТЕ САМИ

Создайте простой блюпринт-класс

Подробнее блюпринт-классы раскрываются в следующих уроках, а сейчас вы научитесь простому способу объединения нескольких актеров в один пригодный для многократного использования блюпринт — факел.

1. Создайте новую папку на панели **Content Browser** и назовите ее **MyBlueprints**.
2. Найдите ассет статичного меша **Shape_Cylinder** в папке **Starter Content** на панели **Content Browser**.
3. Поместите цилиндрический меш на уровень и отмасштабируйте его до желаемой величины.
4. Присвойте цилинду материал, перетащив его из панели **Content Browser** на актер статичного меша.
5. Найдите ассет звукового сигнала **Fire01_Cue** в папке **Starter Content**, перетащите его на уровень и поместите наверх добавленного цилиндра.
6. Добавьте актер точечного источника света из вкладки **Place** панели **Modes**. Поместите ИС так, чтобы он находился прямо над цилиндром. Установите желаемый цвет, интенсивность и радиус затухания ИС.
7. Найдите систему частиц **P_Fire** в папке **Starter Content**, добавьте ее на уровень и поместите прямо над цилиндром.
8. Выберите все актеры, помещенные на уровень в шагах 2–7, щелкните по иконке **Blueprint** в главной панели инструментов и выберите вариант **Convert**

Selected Components to Blueprint Class (преобразовать выбранные компоненты в блюпринт-класс).

- В появившемся диалоговом окне (см. рис. 9.7) выберите путь, дайте блюпринт-классу название **P_Fire_Blueprint** и щелкните по кнопке **Create Blueprint**.



Рис. 9.7. Объединение актеров в единый блюпринт-ассет, пригодный для многоократного использования

- В главном интерфейсе редактора уровней выберите вариант **File ⇒ Save All**.
- На панели **Content Browser** найдите только что созданный блюпринт-ассет и перетащите его на уровень столько раз, сколько вам нужно.

Создание мира в отдалении

Миром в отдалении (world beyond) можно назвать области, находящиеся за пределами досягаемости игрока. Мир в отдалении создает иллюзию того, что уровень находится в гораздо большем мире, чем есть на самом деле (см. рис. 9.8). В качестве примеров мира в отдалении можно назвать горные цепи и городские пейзажи на горизонте. Концепция мира в отдалении также применяется для более близких пространств, таких как комнаты, в которые игрок может заглянуть, но не войти, или огражденные внутренние дворы зданий.

В процессе создания уровня подумайте, какими способами вы могли бы реализовать эту концепцию. Следующие разделы покажут вам несколько решений.

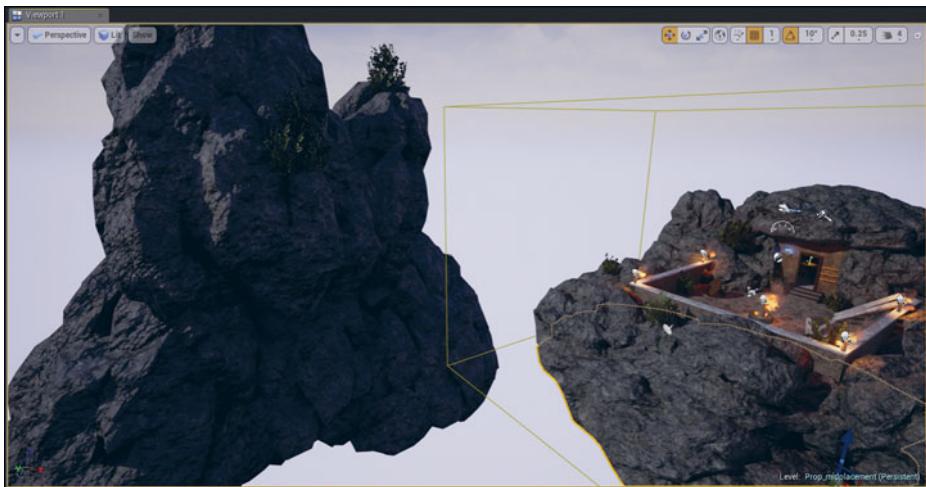


Рис. 9.8. Простая реализация мира в отдалении

Как видно на рис. 9.8, большая скала, висящая в воздухе, помещена для того, чтобы представить мир в отдалении. Было бы лучше, если бы игроки не могли видеть, что она висит в воздухе. Добавление актера **Exponential Height Fog** (туман, распространяющийся в высоту) на уровень поможет сократить обзор игрока, как показано на рис. 9.9.



Рис. 9.9. Изображение декорированного уровня с распространяющимся в высоту туманом

Распространение света и звука

Перейдя к четвертой стадии декорирования нашего уровня, мы можем начать помещать источники света и актеры звуков окружения.

Освещение

Освещение — один из наиболее важных аспектов декорирования. Оно связывает пространство и все размещенные ассеты в последовательное визуальное впечатление, создает настроение и эмоциональную реакцию игрока на среду. Новички часто сосредоточиваются на регулировке интенсивности источников света и забывают настроить цвета ИС на уровне. Цвет освещения является важнейшим элементом создания настроения на уровне, кроме того, выбор цвета ИС также помогает направлять игрока из одного пространства в другое.

При работе с источниками света на помощь приходит переключение между режимами **Lit**, **Detailed Lighting** и **Lighting Only**. Так вы сможете увидеть каждый помещенный ассет на уровне с серой поверхностью, что позволит вам оценить цвет и интенсивность всех источников света, помещенных в мир, и их совместную работу (см. рис. 9.10). Добавьте актеры точечных и прожекторных ИС при необходимости. Затем настройте интенсивность, цвет ИС и параметры затухания. Не забудьте об уже помещенных актерах направленного и небесного ИС.

См. 5-й час, «Применение освещения и рендеринга», чтобы изучить работу с источниками света.

СОВЕТ

Режимы просмотра

Вы можете легко поменять режим просмотра вьюпорта. Выберите режим при помощи следующих комбинаций клавиш.

- ▶ **Wireframe** (каркасный): **Alt+2**
- ▶ **Unlit** (неосвещенный): **Alt+3**
- ▶ **Lit** (освещенный): **Alt+4**
- ▶ **Detailed Lighting** (детализированное освещение): **Alt+4**
- ▶ **Lighting Only** (только освещение): **Alt+5**
- ▶ **Lighting Complexity** (сложность освещения): **Alt+6**



Рис. 9.10. Установив режим просмотра выюпорта **Lighting Only**, вы визуализируете освещение на уровне без материалов

СОВЕТ

Добавление ИС без представления

Поскольку актеры ИС легко поместить на уровень, новички часто освещают области на уровне, но забывают поместить визуальный источник. Будь это фонарь, лампа или факел, убедитесь, что установили актер, объясняющий, откуда берется свет, чтобы придать освещению логику.

Цвета теней

Изменение цвета и интенсивности ИС на сцене может помочь создать настроение, но помните, что тени всегда черные. Хотя нельзя прямо поменять цвет тени на сцене, можно повлиять на их цвет, поместив небесный ИС. Выберите актер небесного ИС на уровне и на панели **Details** в категории **Light** настройте свойства цвета и интенсивности ИС по своему усмотрению.

Параметр *Lightmass Importance Volume*

Если вы выполняли просчет освещения по мере описываемого процесса, то могли заметить, что длительность просчета освещения возросла с увеличением количества контента на уровне. При просчете освещения движок **Lightmass** вычисляет количество отражений света на уровне (три отражения по умолчанию). Если свет сталкивается с поверхностью, отражается от нее и продолжает путь, не встретившись с новой поверхностью, UE4 продолжает обрабатывать свет до тех пор, пока

он не покинет пределы уровня. Чтобы минимизировать эту обработку, вы можете настроить параметр **Lightmass Importance Volume**, чтобы установить область, за пределами которой свет больше не будет обрабатываться. Сделав это, вы значительно сократите продолжительность просчета освещения.

Параметр **Lightmass Importance Volume** находится в категории **Volume** панели **Modes**. Когда вы добавили этот объем на уровень, вы можете настроить его размер и форму на панели **Details** в разделе **Brush Settings**. Объемы, помещенные на уровень, должны охватывать важные области уровня.

Аудио

Простые звуки окружения могут вдохнуть жизнь в статичный уровень. Аудио играет огромную роль в восприятии игрока. Порывы ветра, щебетание птиц, отдаленные раскаты грома и гул генератора — это звуки среды, помогающие оживить статичный мир. Звуки окружения, как правило, зациклены.

Чтобы добавить актер звуков окружения на уровень, найдите ассет звуковой волны на панели **Content Browser**, перетащите его на уровень и настройте его свойства. В разделе **Attenuation** снимите флажок **Override Attenuation** и настройте радиус и дистанцию затухания.

См. 7-й час, «Использование элементов аудиосистемы», чтобы изучить работу со звуковыми актерами.

Игровое тестирование и отладка

После того как вы закончили все этапы создания мира, такие как создание слоев уровня, помещение на него актеров статичных мешей и установка освещения и аудио, необходимо проверить всю свою работу игровым тестированием и пройти отладку. Надеемся, периодически вы уже проводили игровое тестирование своего уровня в процессе работы над ним. Если нет, пора щелкнуть по кнопке **Play** на панели инструментов редактора уровней и прогуляться по уровню.

Попытайтесь взглянуть на свой уровень с точки зрения игрока и определить проблемы. Поиските места, которые недостаточно детализированы. Это может проявляться по-разному, например поверхности не присвоен материал или объекты висят над землей или утопают в полу. Взгляните на размещение ассетов: всегда ли декорации помещаются по периметру комнаты, не поглощая другие объекты? Также обеспечьте, чтобы архитектурное пространство не было идеально симметричным и не имело слишком много различных материалов на уровне, утомляя глаза игрока.

После того как вы произведете необходимую отладку уровня, вы можете добавить несколько других актеров, которые будут иметь совершенно иной окончательный

внешний вид. Вы можете найти эти актеры визуальных эффектов на панели **Modes** на вкладке **Visual Effects**: актеры **Sphere Reflection Capture**, **Fog** и **Post Processing Volume**. Если поместить эти актеры на уровень и настроить их, ваша игра будет выглядеть более профессионально.

Актеры захвата отражений

Актеры сферического или кубического захвата отражений захватывают изображение уровня со своей позиции и проецируют отражения на другие актеры по соседству, если ближайшие актеры имеют материалы со свойствами отражения. Несмотря на то, что эти актеры не создают точные отражения, их использование весьма эффективно и целесообразно, поскольку захват сцены вычисляется до начала игры. Вы можете помещать их при необходимости на вашем уровне.

Актеры тумана

Разработчики игр раньше использовали туман, чтобы скрыть тот факт, что уровни были небольшими и имели мало ассетов, или для прикрытия удаленных ассетов, чтобы повысить эффективность отображения. Сегодня туман — по большей части, выбор эстетики. Будьте осторожны: туман может легко разгладить сцену, поскольку смывает контрастность освещения.

Существует два вида актеров тумана, которые вы можете поместить на сцену.

- ▶ **Atmospheric Fog.** Атмосферный туман, который обычно используется с внешними уровнями и имитирует атмосферное рассеивание света. Он работает непосредственно с направленным ИС, помещенным на уровень.
- ▶ **Exponential Height Fog.** Актер распространяющегося в высоту тумана, который контролирует плотность тумана на уровне, базируясь на высоте: меньшие по высоте области на уровне имеют большую плотность тумана, чем более высокие области.

Актеры объемов постобработки

Постобработка позволяет применять к сцене эффекты камеры. Вы можете настраивать такие свойства, как **Depth of Field**, **Motion Blur** и **Scene Color** в отображаемых кадрах. Вы можете использовать актеры объемов постобработки, чтобы применять эти эффекты к областям на сцене, которые определены примитивными формами. После того как актер объема постобработки был помещен на уровень и выбран, вы можете настроить размер и форму актера в разделе **Brush Settings** панели **Details**. Когда камера попадет в актер объема постобработки, эффекты будут применены. На рис. 9.11 показан уровень до и после помещения на него актера объема постобработки с настроенными свойствами **Color Grading**, **Scene Color** и **Depth Of Field**.

Вы можете помещать на сцену как множество актеров объемов постобработки, так и всего один, влияющий на весь уровень, включив параметр **Unbound** в категории **Post Process Volume** в разделе **Brush Settings** панели **Details**. Здесь же вы можете найти все остальные свойства, которыми можете управлять.

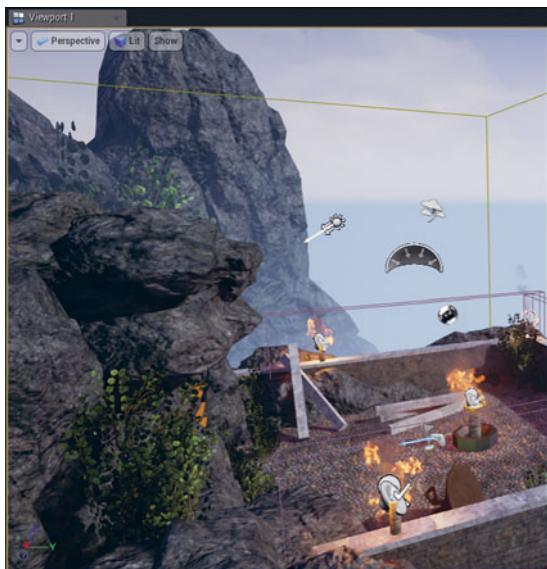


Рис. 9.11. Изображение украшенного уровня до (сверху) и после добавления постобработки (снизу)

Резюме

В этом часе вы научились добавлять новый уровень к проекту и ознакомились с новыми возможностями, связанными с украшением уровня. Вы также узнали некоторые базовые понятия, связанные с созданием виртуальных игровых миров, и применили на практике полученные в предыдущих уроках знания. В этом часе вы увидели, что хороший художник уровней должен думать как дизайнер интерьера и архитектор ландшафтов при создании областей интереса и интриги, основанных на конечном повествовании игры.

Вопросы и ответы

Вопрос: Когда я помещаю актеры статичных мешей на уровень и они перекрывают друг друга, почему некоторые из перекрытых полигонов начинают мерцать?

Ответ: Нет ничего страшного в том, что актеры статичных мешей перекрывают друг друга, но если какие-либо полигоны копланарны (то есть занимают одно и то же пространство), движок рендеринга не знает, какой полигон отобразить. В результате появляется эффект мерцания, когда UE4 пытается определить порядок сортировки. Чтобы исправить эту проблему, просто сместите один из перекрывающихся ассетов.

Вопрос: Когда я просматриваю уровень, почему я могу перемещаться сквозь некоторые актеры статичных мешей?

Ответ: Скорее всего, ассеты статичных мешей не имеют оболочки коллизий. Чтобы исправить это, найдите статичный меш на панели Content Browser, откройте его в редакторе статичных мешей и сгенерируйте новую оболочку коллизии. Обратитесь к 4-му часу, «Работа с актерами статичных мешей».

Вопрос: Пытаюсь присвоить цвет неба актеру Sky_Sphere, но мой выбор не дает никакого эффекта. Почему?

Ответ: По умолчанию цвета небесной сферы определяются направленным ИС. Вы также можете изменить угол поворота направленного ИС или с выбранным актером Sky_Sphere снять флажок Colors Determined By Sun Position (цвета, определяемые положением солнца) на панели Details. Тогда вы сможете изменять параметры настройки актера Sky_Sphere.

Вопрос: Почему на некоторых помещенных статичных мешах я вижу слово preview (предпросмотр)? И почему редактор предупреждает меня, что освещение требует повторного просчета?

Ответ: Если у вас есть актеры статичных мешей и ИС с параметром мобильности в значении Static, необходимо произвести просчет освещения, чтобы

создать освещение и тени для этих актеров. Если ваш уровень не слишком большой и просчет освещения не занимает слишком много времени, то нет ничего плохого в том, чтобы производить просчет освещения регулярно.

Вопрос: Почему я продолжаю видеть сообщение `there is no Lightmass importance volume?`

Ответ: Необходимо добавить объем обработки Lightmass на сцену и установить ее размер таким образом, чтобы в нее были включены все ассеты.

Вопрос: Должен ли я использовать BSP-актеры для создания наброска моего уровня?

Ответ: Это не обязательно. Можно использовать только ассеты статичных мешей.

Вопрос: Мой персонаж может перепрыгнуть стену и попасть за пределы уровня. Как это предотвратить?

Ответ: В зависимости от масштабов ассета игрок может иметь возможность прыгать выше высоты ассета. Это дело предварительного плана, неплохо было бы знать возможности аватара, прежде чем начинать украшать пространство. Тем не менее вы можете поместить актеры объемов блокирования (Blocking Volume), чтобы определить невидимые области, в которые игрок не может попасть. Объемы блокирования можно найти на панели Modes на вкладке Volumes.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

- Если просчет освещения уровня занимает длительное время, какую опцию вы можете установить, чтобы сократить время просчета, убрав лучи света, выходящие за пределы определенной области?
- Если вы используете материалы со связанными свойствами, но отражения не появляются, какой вид актеров вы можете добавить?
- Если аватар может перепрыгивать стены и выпадать за пределы уровня, какой актер вы можете поместить для создания невидимых оболочек коллизий?
- Истинно или ложно высказывание: группировка актеров и создание ссылок на панели **World Outliner** — это единственный способ объединения актеров.

5. Истинно или ложно высказывание: концепция мира в отдалении относится только к далеким областям, в которые игрок не может попасть.

Ответы

1. Если просчет освещения уровня занимает много времени, вы можете добавить параметр **Lightmass Importance Volume**, чтобы сократить время просчета.
2. Если вы используете материалы со свойствами отражения, но отражения не отображаются, вы можете поместить актеры сферического и/или кубического захвата отражений на ваш уровень.
3. Если аватар может перепрыгнуть стену и упасть за пределы уровня, вы можете поместить актеры объемов блокирования на ваш уровень.
4. Ложь. Помещенные на уровень актеры могут быть объединены в блю-принт-класс, который пригоден для многократного использования на всех уровнях вашего проекта.
5. Ложь. Несмотря на то что концепция мира в отдалении по большей части применяется к далеким областям, она также может применяться к любым близким областям, которые игрок может видеть, но в которые не может попасть.

Упражнение

В этом упражнении создайте и украсьте еще один уровень, но в этот раз начните с шаблона пустого уровня. Шаблон пустого уровня не имеет заранее помещенных на него актеров, как в шаблоне по умолчанию. Это хорошая практика для начинающих знакомиться со всеми актерами, которые необходимо установить в базовый уровень. Украсьте небольшую среду, используя существующие ассеты и материалы. Следуйте последовательному визуальному подходу к пространству и центральному объекту, соединяющему все элементы вместе. Вы должны продемонстрировать свое понимание и способность применять простые концепции украшения, такие как визуальная сложность и связь с миром в отдалении.

1. В проекте, который вы создали в этом часе, в главном меню выберите пункт **File ⇒ New Level** (или нажмите комбинацию клавиш **Ctrl+N**).
2. В появившемся диалоговом окне выберите вариант **Blank Level**.
3. Сохраните только что созданный уровень в папку **Maps**.

4. Добавьте BSP-куб и в разделе **Brush Settings** на панели **Details** установите значение формы кисти **2000** для оси X, **2000** для оси Y и **50** для оси Z.
5. Добавьте актер стартовой точки игрока.
6. Добавьте направленный ИС.
7. Добавьте небесный ИС.
8. Добавьте актер атмосферного тумана.
9. Добавьте блюпринт-актер **Sky_Sphere**.
10. Перенесите статичные меши, материалы, системы частиц и аудио-ассеты из других проектов или добавьте пакеты контента из магазина.
11. Украсьте уровень.
12. Добавьте актер **Exponential Height Fog**.
13. Добавьте актер объема обработки **Lightmass**.
14. Добавьте актеры сферического захвата отражений.
15. Добавьте актер объема постобработки.

10-Й ЧАС

Эффекты воспроизведения в системах частиц

Что вы узнаете в этом часе

- ▶ Изучение частиц и типов данных
- ▶ Работа с редактором **Cascade**
- ▶ Использование эмиттеров и модулей
- ▶ Использование редактора кривых
- ▶ Настройка материалов для частиц
- ▶ Запуск систем частиц

Системы частиц (Particle Systems) — это строительные блоки визуальных эффектов игр. Вы можете использовать частицы во взрывах, вспышках выстрелов оружия, в падении листьев на ветру, водопадах, магической энергии, освещении, дожде, пыли, огне и многом другом. В UE4 частицы имеют большое разнообразие, которым можно манипулировать, чтобы создавать множество эффектов с помощью редактора частиц **Cascade**. Этот редактор частиц в режиме реального времени и модульный подход UE4 к управлению поведением частиц позволяет проектировать очень сложные эффекты быстро и просто. В этом часе вы узнаете о различных типах частиц в UE4, как использовать **Cascade** для создания частиц и управления их поведением, как использовать SubUV-текстуры и эффекты частиц с блюпринтами уровней.

ПРИМЕЧАНИЕ

Проект 10-го часа

Для этого урока вам потребуется открыть папку *Hour_10*, которая доступна по адресу: http://addons.eksмо.ru/it/UE_24.zip. Эта папка содержит готовые к использованию полезные наборы текстур и готовые системы частиц, которые вы можете изучить для лучшего понимания концепций этого урока.

Изучение частиц и типов данных

В компьютерных играх *частица* (particle) — это точечный объект в пространстве с набором атрибутов, определяющих его визуализацию. Как правило, существуют некоторые формы мешей, прикрепленных к этим точкам, которые вызывают появление частиц перед игроком. Существуют разные типы частиц с различными видами мешей или конструкций, прикрепленных к точкам.

UE4 включает следующие типы эмиттеров частиц (Particle Emitters), каждый из которых имеет свои преимущества для различных ситуаций.

- ▶ **Спрайты (Sprites).** Безусловно, наиболее распространенный тип эмиттеров — спрайт — это единственный направленный в камеру меш с четырехугольными полигонами с текстурой для определения внешнего вида. Спрайты чаще всего используются для создания дыма и огня, но они могут быть использованы и для множества других эффектов. Чаще всего эффекты создаются с установочными направленными в камеру эмиттерами спрайтов.
- ▶ **Данные мешей (Mesh data).** В этом типе эмиттеров частицы прикрепляются к единственному полигональному мешу. Это позволяет создавать такие впечатляющие эффекты, как оползни и летящие от взрывов обломки.
- ▶ **Данные анимированных следов (Anim-trail data).** Использующиеся исключительно со скелетными мешами и анимацией, эмиттеры этих анимированных следов создают следы с помощью сокетов скелетных мешей. Они хорошо подходят для создания следа от меча или другого холодного оружия.
- ▶ **Данные лучей (Beam data).** Этот тип эмиттера рисует направленный в камеру набор четырехугольников, тянувшихся между только что созданными частицами. Он часто используется для лазеров, молний и подобных эффектов.
- ▶ **GPU-спрайты (GPU Sprites).** Внешне похожие на обычные спрайты, GPU-спрайты отличаются от них тем, что полностью моделируются графическим процессором. Это позволяет моделировать и отображать на порядок больше частиц, чем это может делать центральный процессор. Некоторые возможности, доступные обычным спрайтам, не доступны GPU-спрайтам, однако GPU-спрайты полезны, когда необходимо смоделировать большое количество отдельных, зернистых эффектов, таких как искры, фейерверк, снег или дождь.
- ▶ **Данные осколков (Ribbon data).** Этот тип эмиттеров создает четырехугольники между каждой парой только что помещенных частиц, интерполируя изгибы с плавными кривыми. Этот тип часто используется с движущимися эмиттерами для создания двигателя или следов пуль.

Этот час сфокусирован на трех наиболее часто используемых эмиттерах частиц: спрайтах, GPU-спрайтах и данных мешей.

Работа с редактором Cascade

В Unreal Engine 4 есть мощный редактор частиц, который называется **Cascade**. Его интерфейс и количество настроек поначалу могут обескуражить, но универсальность и модульный подход к поведению частиц делает его невероятно полезным инструментом.

Система частиц — это коллекция из одного или нескольких эмиттеров частиц (возможно, разных типов), создающих нужный эффект. Каждый эмиттер частиц порождает произвольное количество частиц и управляет их поведением и отображением. Он контролирует их поведение с помощью модулей, которые вы можете добавить или убрать из эмиттера частиц.

Модули могут управлять эффектами, такими как размер частицы, цвет, скорость и поворот, и они также могут обрабатывать коллизии.

Редактор **Cascade** можно открыть, дважды щелкнув по любому шаблону частиц на панели **Content Browser**. Cascade состоит из шести основных частей, как показано на рис. 10.1 и описано в следующем ниже.

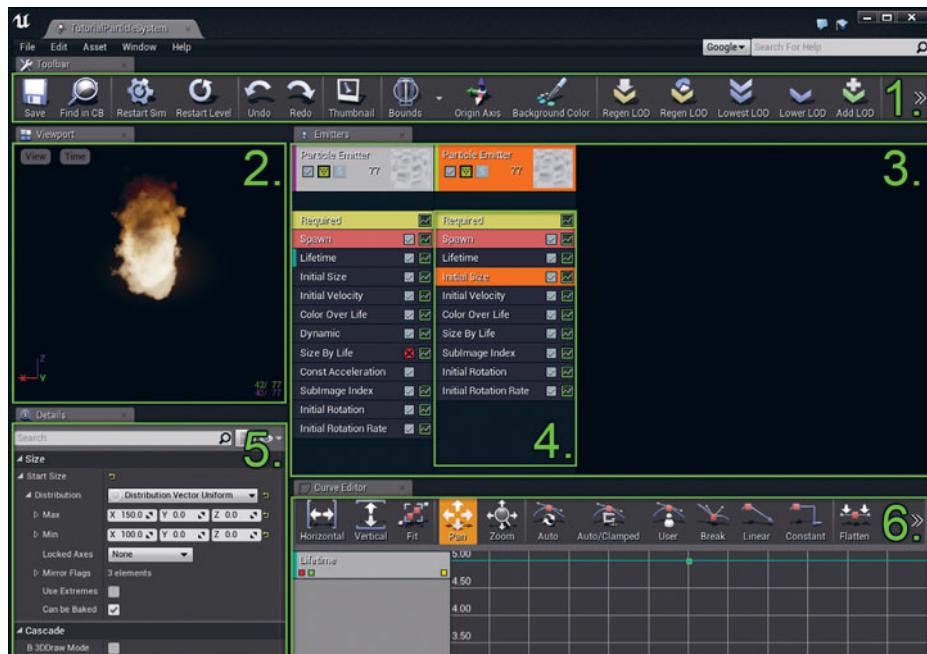


Рис. 10.1. Шесть важных разделов редактора **Cascade**

Эти шесть частей включают: 1) панель инструментов; 2) выюпорт; 3) панель **Emitters**; 4) панель **Modules**; 5) панель **Details**; 6) редактор кривых. Данные разделы описаны в следующем списке.

- ▶ **Панель инструментов.** Как и в большинстве других редакторов в UE4, с помощью панели инструментов можно сохранять изменения и обрабатывать действия с ассетами. Важные часто используемые кнопки — **Restart Sim** и **Restart Level**, которые заставляют UE4 обновить систему частиц и запустить ее заново.
- ▶ **Выюпорт.** Эта панель представляет полноценный предварительный просмотр системы частиц и позволяет использовать стандартные средства управления движением.
- ▶ **Панель Emitters.** Эта панель содержит все эмиттеры, содержащиеся в выбранном шаблоне. Каждый эмиттер — это часть эффекта, контролирующая различные частицы и представленная в столбце. Можно добавить новый эмиттер, щелкнув правой кнопкой мыши в пустом месте.
- ▶ **Панель Modules.** В этой панели содержится поведение, которое контролирует эмиттер, выбранный на панели **Emitters**. Каждый модуль отображается в виде строки с названием типа и флажком для включения и выключения модуля. Можно добавить новые модули, щелкнув правой кнопкой мыши по этому столбцу и используя контекстное меню для выбора определенного модуля.
- ▶ **Панель Details.** Эта панель отображает доступные свойства выбранного модуля. Когда не выбран ни один модуль или эмиттер, она показывает глобальные свойства шаблона частиц.
- ▶ **Редактор кривых.** Эта панель визуализирует значения свойств в виде кривых, допуская сложные эффекты, которые обычно происходят в течение всего жизненного цикла частицы. Этот редактор, как правило, используется для создания эффекта исчезновения частиц или изменения их размеров с течением времени и управления скоростью этих эффектов. Можно визуализировать кривые модулей в редакторе кривых, щелкнув по иконке диаграммы в нужном вам модуле.

Использование эмиттеров и модулей

Чтобы понять, как использовать редактор **Cascade** для создания эффектов частиц, необходимо сначала понять, как модули могут управлять частицами внутри эмиттера. Диапазон модификаций и разнообразие вариантов поведения, которые может описать модуль, огромны.

Эмиттер описывает коллекцию частиц, и это описание полностью определяется модулями, которые составляют эмиттер. Модули могут влиять на движение эмиттера, его поведение, цвет и отображение, какой тип данных он отрисовывает, сложные события в течение жизненного цикла частиц и многое другое.

Модули отображаются в виде отдельных строк в столбце эмиттера на панели **Emitters**. Вы можете добавить модули в эмиттер, щелкнув правой кнопкой мыши по эмиттеру на панели **Emitters**. Вы можете изменять различные свойства и параметры модуля с помощью панели **Details**.

Необходимые модули

В каждом эмиттере содержится три обязательных модуля. Первый не выглядит как модуль, но содержит специфическую для эмиттера информацию, такую как название эмиттера и информация о качестве. Этот модуль содержится вверху столбца эмиттера и отображает название эмиттера и уменьшенное изображение эффекта эмиттера. Здесь можно отключить весь эмиттер с помощью флагка под названием эмиттера.

Под верхним модулем находится черная полоса — слот (ячейка), в который могут быть помещены данные о типе эмиттера. Этот модуль всегда существует, но может быть пустым. Решение не заполнять слот данными об определенном типе приведет к тому, что эмиттер будет создавать частицы CPU-спрайтов.

Следующий модуль называется **Required**, в нем содержатся параметры, которые требуются эмиттеру частиц. Информация о примененном материале, продолжительности жизненного цикла эмиттера, зациклен эмиттер или нет, может быть найдена в этом модуле. Будет уместным взглянуть на все параметры, содержащиеся в модуле **Required** и получить представление обо всем, что в нем находится. Вы будете часто возвращаться к этому модулю при работе с UE4.

Последний модуль называется **Spawn**, и он отвечает за количество новых частиц и их частоты или их объем выпуска. Без этого модуля не может быть создана ни одна частица.

Свойства модулей

Каждый модуль имеет набор свойств, управляющих тем, как он влияет на содержащиеся в нем частицы. Поле **Distribution** контролирует, как каждая частица определяет значение для этого свойства. Доступные виды распределения сгруппированы в несколько типов.

ПРИМЕЧАНИЕ

Векторные и скалярные распределения

Каждое распределение задано для скалярных вещественных значений и вещественного вектора с тремя координатами. Тип распределения (скаляр или вектор) определяется свойством и не может быть изменен пользователем.

Чтобы узнать больше о распределениях и о том, как они работают в *Cascade*, изучите подробную документацию по адресу: docs.unrealengine.com/latest/INT/Engine/Basics/Distributions.

- ▶ **Distribution Float/Vector Constant.** Некоторые свойства, такие как **Duration**, могут иметь только одно значение. Независимо от точки симуляции постоянное распределение всегда возвращает одно и то же значение. Другие свойства (например, свойство **Lifetime** одноименного модуля) не обязательно должны быть постоянными, но могут интерпретироваться как таковые.
- ▶ **Distribution Float/Vector Constant Curve.** Когда свойства должны изменяться в процессе существования системы частиц или эмиттера, они могут быть интерпретированы в виде кривых. На протяжении существования каждой частицы в любой точке свойство рассчитает кривую одинаково. Это зачастую используется для предсказуемого изменения цвета или прозрачности частиц в течение их жизни. Кривые лучше всего редактировать в редакторе кривых, также они настраиваются и вручную на панели **Details**.
- ▶ **Distribution Float/Vector Uniform.** Свойства, которые могут не быть константами, задаются разнообразными распределениями. Самый простой — равномерное распределение, заданное минимальным и максимальным значениями, и возвращающее равномерно распределенные случайные величины между ними (см. рис. 10.2). Этот тип распределения чаще всего используется с модулями, которые влияют на исходное состояние частицы. Например, модуль **Initial Size**, который применяет равномерные распределения, чтобы задать каждой частице произвольный масштаб.
- ▶ **Distribution Float/Vector Uniform Curve.** Самый сложный вариант свойства — случайное распределение скалярного или векторного значения в диапазоне, заданном парой кривых. Он обладает всеми преимуществами константных кривых, но позволяет контролировать значения случайных величин при помощи диапазонов, заданных кривыми. Этот тип распределения лучше использовать, когда эффект должен быть одновременно случайным и модулируемым со временем. Как и с константными кривыми, лучше использовать редактор кривых для внесения изменений.

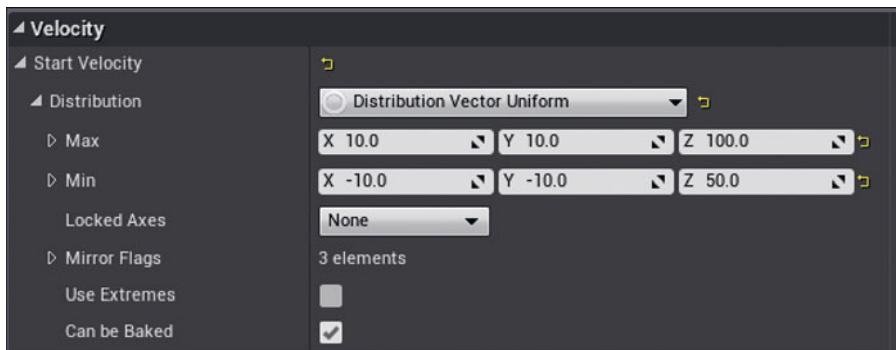


Рис. 10.2. Пример использования параметра настройки кривыми, который описывает рандомизированную (случайную) исходную скорость каждой частицы. В этом случае скорости X и Y разняются от -10,0 до 10,0, в то время как скорость Z варьируется от 50,0 до 100,0

ПРИМЕЧАНИЕ

Initial или Over Life

Названия некоторых модулей включают слова *Initial* или *Over Life* (например, **Initial Color**, **Color over Life**). Эти названия довольно точно описывают поведение модулей и имеют незначительные отличия с позиции их оценки.

Если название модуля содержит слово *Initial* (исходный), распределения кривых и равномерные распределения кривых оцениваются в течение всей продолжительности эмиттера, не только продолжительности жизни отдельных частиц. Например, если свойство **Initial Color** определено в виде кривой от красного к зеленому, в то время как весь эмиттер остается активным, новые частицы начинают становиться зелеными по мере их появления.

С другой стороны, когда в названии модуля содержатся слова *Over Life* (в течение цикла), распределения кривых и равномерные распределения кривых оцениваются с точки зрения продолжительности жизни каждой частицы. Например, свойство **Color over Life** определяется в виде кривой от красного к зеленому, и каждая новая частица имеет переход цвета с красного в зеленый отдельно.

Использование редактора кривых

Многие модули уместно использовать с кривыми распределений. Редактирование кривых вручную с помощью панели **Details** возможно, но не интуитивно. К счастью, **Cascade** предоставляет полноценный редактор кривых, упрощающий их создание и управление.

Просмотреть кривые в редакторе кривых (**Curve Editor**) можно, щелкнув по иконке диаграммы в любом модуле на панели **Emitters**. На рис. 10.3 показаны наиболее важные возможности редактора кривых: 1) панель инструментов; 2) визуализаторы каналов; 3) визуализатор свойств; 4) ключ. Ниже описаны эти возможности.



Рис. 10.3. Редактор кривых, показывающий несколько кривых с подсвеченными точками интереса

- ▶ **Панель инструментов.** Панель инструментов содержит кнопки для обработки и управления кривыми. Первые три, подсвеченные на рис. 10.3, — инструменты фрейминга, которые используются для быстрой настройки выюпорта для отображения минимальных и максимальных значений или всех видимых кривых.
- ▶ **Визуализаторы каналов.** Каждая кривая распределения — это либо конкретная числовая кривая, либо набор кривых, задающих распределение векторов. В случае с распределением векторов три блока (красный, зеленый и синий) могут быть использованы для отображения отдельных каналов вектора. Если щелкнуть по любому из этих блоков, включится или отключится соответствующая кривая в редакторе.
- ▶ **Визуализатор свойств.** Этот блок включает или отключает визуализацию всех каналов соответствующего свойства. Вы можете использовать его, чтобы одним щелчком отключить отображение всех кривых в свойстве.
- ▶ **Ключ.** Как и многие другие редакторы кривых или анимации, этот редактор кривых работает сложенными в него ключами. Вы можете управлять ключами непосредственно с помощью комбинаций клавиш или щелкнув правой кнопкой мыши и выбрав значения вручную.

Добавление ключей и поиск во выпорте требуют практики. Команды управления, перечисленные в табл. 10.1, незаменимы для эффективного использования редактора кривых при определении эффектов.

ТАБЛ. 10.1. Команды управления редактора кривых

Команда	Описание
ЛКМ + перетаскивание на фон	Панорамный обзор
Колесо мыши	Приближение и отдаление
Щелчок по ключу	Выбор ключа
Ctrl + щелчок по ключу	Включение/выключение выбора ключа
Ctrl + щелчок по кривой	Добавление нового ключа в выбранную локацию
Ctrl + ЛКМ + перетаскивание	Перемещение текущего выбора
Ctrl + Alt + ЛКМ+перетаскивание	Выбор блока
Ctrl + Alt + Shift + ЛКМ+перетаскивание	Выбор блока и добавление текущего выбора

Использование основных модулей

Вам доступно множество модулей. Однако существуют модули, которые используются почти всегда, и их универсальность и различные свойства стоит упомянуть. Они описаны в следующих разделах.

Модуль Required

Модуль **Required**, как упоминалось выше, обрабатывает большую часть информации и необходим эмиттеру. Обратите внимание на некоторые из важнейших свойств этого модуля, описанных в этом разделе.

Категория **Emitter** включает следующие важные свойства.

- ▶ **Material.** Это свойство описывает, какой материал используется каждой частицей в эмиттере. (Вы узнаете о дружественных к частицам материалах далее в этом часе.)
- ▶ **Use Local Space.** Это булево свойство определяет, должна ли система частиц полностью наследовать позицию ее актера, поворот и масштаб или эти свойства должны симулироваться в пространстве мира. По умолчанию флагок для этого свойства не установлен, отчего эмиттер перемещается по миру, а частицы остаются. Это также означает, что частицы игнорируют поворот содержащего их актера. Будет уместно включить это свойство, когда

необходимо, чтобы эффект следовал за актером, например при вспышках выстрелов оружия или вспышках двигателя ракеты.

- ▶ **Kill on Deactivated** и **Kill on Completed**. Эти свойства направлены на эффективность, вы можете использовать их для того, чтобы заставить UE4 автоматически обновлять системы частиц. Свойство **Kill on Deactivated** уничтожает эмиттер каждый раз, когда он деактивирован, а свойство **Kill on Completed** уничтожает эмиттер, как только истекает время продолжительности эмиттера.

Категория **Duration** включает следующие свойства, связанные с продолжительностью и зацикливанием эмиттера.

- ▶ **Emitter Duration**. Это свойство представляет собой плавающее значение, определяющее длительность отдельного цикла системы частиц. Это значение изменяется в секундах, поэтому при значении **Emitter Duration** равном 5,0 эмиттер завершит свой цикл через 5 секунд. Важно отметить, что частицы в принципе могут иметь более долгую жизнь, чем это установлено для эмиттера.
- ▶ **Emitter Loops**. Это свойство представляет собой целочисленное значение, определяющее количество циклов. Когда его значение равно 0, циклы повторяются бесконечно.

Наконец, категория **SubUV** контролирует размеры и элементы управления различными свойствами, использующими SubUV-текстуры. Серии свойств в этом модуле определяют, как производится управление SubUV-текстурами. SubUV-текстуры чаще всего используются для отображения простой анимации на частицах для создания сложных многослойных эффектов. Вы узнаете о том, как использовать SubUV-текстуры далее в этом часе.

Модуль **Spawn**

Модуль **Spawn** всегда используется для определения того, как много новых частиц должно создаваться и как часто это должно происходить. Его можно грубо разделить на две категории: ежесекундные спауны (**spawn**) и взрывные спауны. Первая категория определяет, как много частиц создается в секунду, вторая категория указывает эмиттеру создать заданное количество частиц в определенное время.

Категория **Spawn** включает следующие свойства.

- ▶ **Rate**. Это свойство является скалярным распределением, определяющим количество частиц, выпускаемых в секунду.
- ▶ **Rate Scale**. Это свойство — вспомогательная скалярная величина, применяемая к свойству **Rate** для модуляции количества частиц. Значение **Rate**, умноженное на **Rate Scale**, определяет количество частиц к выпуску в данном кадре.

Категория **Burst** включена в несколько большем объеме, чем категория **Spawn**, поскольку она позволяет определять отрезок времени, в который необходимо усилить выпуск определенного количества частиц. Данная категория включает следующие свойства.

- ▶ **Burst List.** Это свойство содержит список количества и времени спауна частиц. Вы можете добавить новые элементы взрыва, щелкнув по иконке + в **Burst List**. Элемент взрыва имеет три свойства: **Count**, **Count Low** и **Time**. **Count** и **Count Low** определяют минимальное и максимальное количество частиц, которое необходимо выпустить в данном кадре, определенном свойством **Time**. Если значение **Count Low** отрицательное, эмиттер выпускает количество частиц, указанное в свойстве **Count**, в противном случае эмиттер выпускает случайное количество частиц, между значениями этих двух свойств. Важно отметить, что свойство **Time** имеет значение между 0,0 и 1,0, где 1,0 означает максимальную продолжительность эмиттера.
- ▶ **Burst Scale** (масштаб взрывов). Это свойство представляет собой распределение, масштабирующее значения, определенные свойством **Burst List**.

Модуль **Lifetime**

В большинстве систем частиц модуль **Lifetime** нужно считать необходимым. Этот модуль описывает, как долго существуют частицы. Он имеет только одно свойство, которое может быть любого типа распределения. Можно использовать этот модуль, чтобы указать частицам случайную продолжительность жизни.

Важно помнить, что многие модули не работают с продолжительностью жизни отдельных частиц, поэтому изменение модуля **Lifetime** может значительно изменить скорость, с которой другие модули меняют поведение частиц.

Модули **Initial Size** и **Size By Life**

Один из аспектов, который часто необходимо контролировать в системах частиц, — это размер и масштаб отображаемых частиц. Модули **Initial Size** и **Size By Life** обычно используются совместно во многих эффектах только с этой целью. Чтобы найти эти модули, щелкните правой кнопкой мыши по эмиттеру и выберите вариант **Add Module** ⇒ **Size**. Ниже приведены детали об этих двух модулях.

- ▶ **Initial Size.** Модуль **Initial Size** устанавливает размер частиц во время их появления. Он чаще всего используется с параметром **Distribution Uniform Vector** для создания некоторого разнообразия случайности между различными частицами.

- ▶ **Size By Life.** Этот модуль обрабатывает важную задачу модуляции размеров отдельных частиц в течении их срока жизни. Вы можете использовать этот параметр настройки с модулем **Initial Size**, чтобы увеличить или уменьшить частицы с течением времени. Частый пример использования — распределение кривых, в котором размер быстро увеличивается с 0,0 на старте до значения близкого к 1,0. Это создает визуальный эффект, похожий на быстрое распространение, который часто используется при создании эффекта взрыва.

Модули Initial Color, Scale Color/Life и Color Over Life

Модули, предназначенные для обработки цвета частиц, включают **Initial Color**, **Scale Color/Life** и **Color Over Life**. Они очень похожи на модули, связанные с масштабированием. Помимо свойств RGB-цвета каждой частицы, эти модули могут контролировать альфа-канал (прозрачность) каждой частицы. Изменяя альфа-канал, вы можете легко скрывать создание и удаление каждой частицы, что особенно полезно для легких эффектов, таких как дым или огонь.

Установка значений, близких к 1,0, создает эффект цветения в виде постпроцесса, который может использоваться для создания таких ярких, сверкающих эффектов, как вспышки или пламя.

Эти модули требуют, чтобы материал, примененный к эмиттеру частиц, имел ввод нода цвета частиц. Ниже описаны некоторые подробности об этих трех модулях.

- ▶ **Initial Color.** Модуль **Initial Color**, как и модуль **Initial Size**, устанавливает цвет каждой частицы при ее порождении. Этот модуль часто используется для рандомизации стартового цвета каждой частицы.
- ▶ **Scale Color/Life.** Этот модуль принимает существующий цвет частицы и модулирует результат в течение продолжительности жизни частицы. Распределения кривых могут использоваться для достижения лучшего эффекта с этим модулем. Вы можете использовать модуль **Scale Color/Life** с модулем **Initial Color** для создания незначительно отличающихся случайных частиц, которые меняют цвет или альфа-канал с течением времени, если значения итогового цвета являются комбинацией обоих модулей.
- ▶ **Color Over Life.** Этот модуль отличается от модулей **By Life**, которые мы изучали ранее, тем, что он устанавливает значение цвета частиц напрямую. Это означает, что этот модуль не учитывает значения, установленные в модулях **Initial Color** или **Scale Color/Life**. Модуль **Color Over Life** обычно используется отдельно, заменяя собой роли двух других модулей.

Модули **Initial Velocity**, **Inherit Parent Velocity** и **Const Acceleration**

Редактор **Cascade** поддерживает некоторые модули, предназначенные для обработки движения частиц: модули **Initial Velocity**, **Inherit Parent Velocity** и **Const Acceleration**. Многие эффекты частиц требуют только самые простые из них — модули, предназначенные для применения скорости или ускорения в постоянном направлении. Ниже описан функционал этих трех модулей.

- ▶ **Initial Velocity.** Этот модуль определяет стартовую скорость любой частицы. Он хорошо работает с универсальными распределениями для создания небольших объемов случайности любой амплитуды и направления.
- ▶ **Inherit Parent Velocity.** Этот модуль определяет стартовую скорость частицы исходя из скорости ее родителя. Если эмиттер движется с какой-либо скоростью, когда появляется частица, модуль применяет к ней скорость и направление родительского эмиттера (или актера). Это работает лучше всего для эффектов частиц, которые должны отображаться на основе физики, особенно когда они прикреплены к актерам, перемещающимся по миру.
- ▶ **Const Acceleration.** Этот модуль применяет последовательную величину ускорения ко всем частицам эмиттера. Его очень уместно использовать для имитации гравитации. Многие эффекты частиц, которые имеют различные физические элементы (например, вспышки, грязь, камни или вода) должны использовать **Const Acceleration** с отрицательным значением компонента Z, чтобы симулировать эффект гравитации.

Модули **Initial Location** и **Sphere**

Категория **Location** для большинства модулей связана с местоположением появления каждой частицы. Когда ни один из этих модулей не используется, движок **Cascade** просто создает частицы в исходном местоположении эмиттера. Эти модули местоположения часто могут складываться, что создает интересные и разнообразные эффекты, но чаще всего этих двух простых модулей вполне достаточно.

- ▶ **Initial Location.** Наиболее часто используемым модулем местоположения, безусловно, является модуль **Initial Location**. Он использует векторное распределение для выбора начального местоположения каждой частицы. При использовании с параметром **Distribution Uniform Vector** этот модуль может заполнить прямоугольный объем новыми создаваемыми частицами. Этот модуль подходит для создания атмосферных эффектов, заполняющих область.
- ▶ **Sphere.** Так же, как и модуль **Initial Location**, модуль **Sphere** обрабатывает местоположение спауна, но при этом распределяет частицы в сферическом

объеме. Радиус сферы может быть установлен распределением. Как правило, постоянного распределения вполне достаточно. Как и модуль **Initial Location**, модуль **Sphere** может использоваться для применения масштабирования скорости каждой частицы. Это полезно, поскольку скорость, которую применяет этот модуль, выравнивается по поверхности сферы. Итоговый эффект хорошо работает для вспышек или частиц, испускаемых точечным источником (например, взрывом).

Модули **Initial Rotation** и **Rotation Rate**

Одно из ограничений систем частиц состоит в том, что сложно сделать отдельные частицы отличными от соседних. Использование случайного размера и цвета поможет создать иллюзию разнообразия, но для большего эффекта рандомизации важен также случайный поворот каждой частицы. Модули, связанные с поворотом, помогут создать интересное движение и разнообразие эффектов. Ниже приведены некоторые детали об этих двух модулях.

- ▶ **Initial Rotation.** Это простой модуль, который устанавливает стартовый угол поворота каждой частицы. Вместе с параметром **Distribution Float Uniform** модуль **Initial Rotation** может создать столь необходимое разнообразие практических всех эффектов. Масштаб поворота варьируется от 0,0 до 1,0; значение 1,0 представляет один полный поворот частицы.
- ▶ **Rotation Rate** (частота вращения). Иногда исходного поворота бывает недостаточно для ваших задач. В таких случаях модуль **Rotation Rate** как нельзя кстати. Вы можете использовать этот модуль, чтобы задать всем частицам уникальное значение угловой скорости. Как и в случае модуля **Initial Rotation**, значения модуля **Rotation Rate** варьируются от 0,0 до 1,0; при этом значение 1,0 означает, что частица выполняет полный поворот за одну секунду.

Модули **Initial Rotation** и **Rotation Rate** работают вместе и могут складываться.

Присвоение материала частицам

Понимание взаимодействия между эмиттерами частиц и материалом частиц очень важно для разработки впечатляющих эффектов. Существуют модули, включая те, что связаны с модуляцией цвета, которые не могут работать, если сначала не установить материалы для интерпретации этих модулей.

Модули эмиттеров придают свойства и параметры частицам, затем эти свойства интерпретируются прикрепленными материалами, но только в том случае, если в материалах используются надлежащие ноды.

Цвета частиц

Способность модулировать цвет частиц невероятно важна и требуется почти всегда. Поэтому установка материала для интерпретации этого свойства — столь частый процесс при создании визуальных эффектов.

В своей наиболее каноничной форме материал частиц принимает значения RGB и A из входных текстур и перемножает их с соответствующими данными входного нода цвета частицы. На рис. 10.4 показан простой прозрачный неосвещенный прозрачный материал с такой установкой.

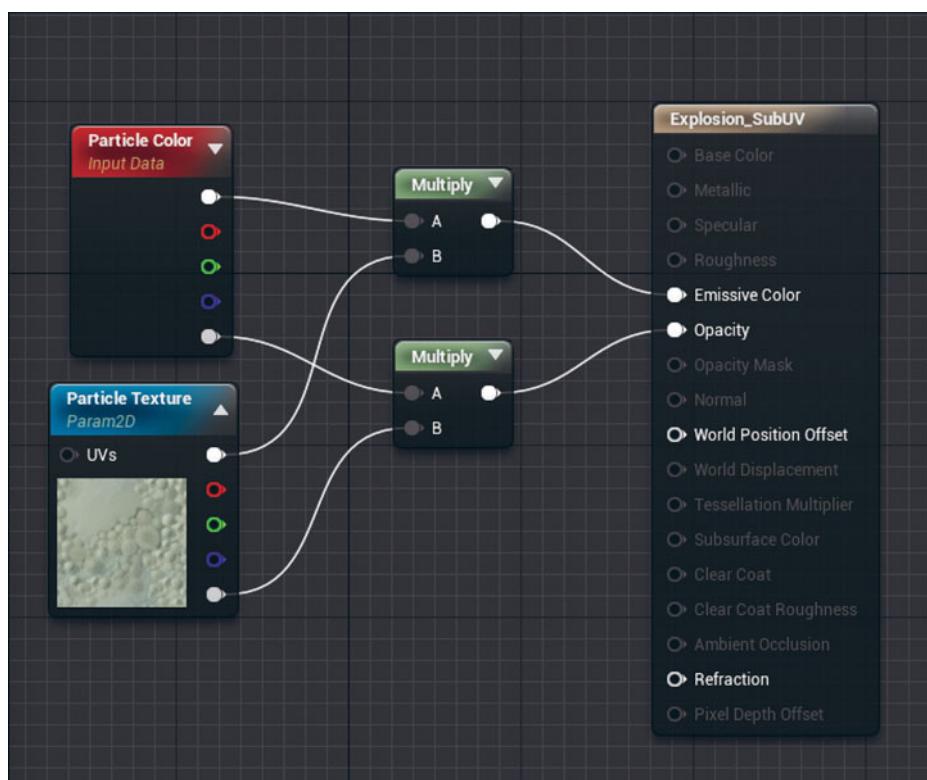


Рис. 10.4. Простой прозрачный неосвещенный материал установлен с возможностью принимать цвет различных цветовых модулей эмиттеров

SubUV-текстуры

Один из самых удобных и универсальных способов создания динамических эффектов состоит в том, чтобы использовать опции **SubUV**, предоставляемые UE4 для создания эффектов SubUV-текстур.

Эффекты *SubUV-текстур* используются для визуальных эффектов, в которых различные кадры анимации просчитываются заранее в отдельную текстуру. Различные кадры накладываются в сеточный шаблон, и во время игры UE4 выбирает и производит интерполяцию между различными кадрами, создавая иллюзию анимации.

На рис. 10.5 показан порядок, в котором отображается пример *SubUV-текстуры*, и очередьность отображения кадров.

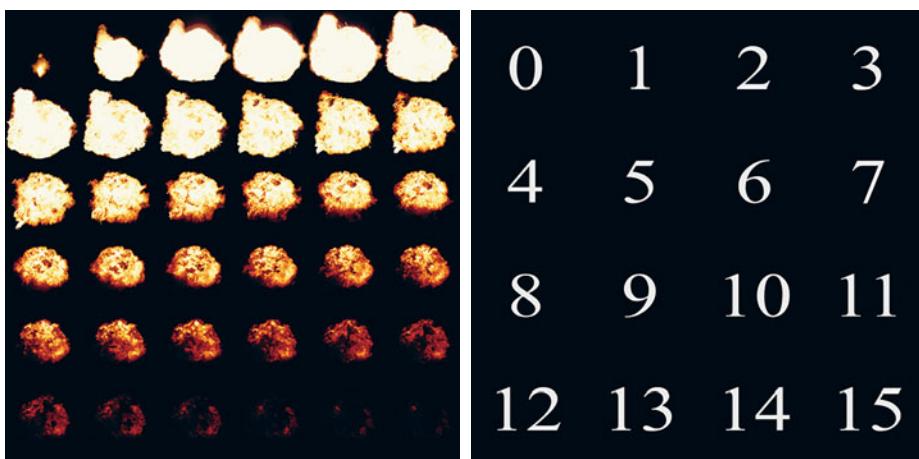


Рис. 10.5. Пример *SubUV-текстуры*. Левое изображение показывает пример заранее просчитанного эффекта взрыва 6×6 , полученного из стороннего пакета. Правое изображение показывает порядок, в котором производится анимация, с номерами кадров для *SubUV-текстуры* размером 4×4

Для использования *SubUV-текстуры* требуется выполнить три шага. В первом шаге необходимо указать в модуле **Required** эмиттера количество столбцов и строк в *SubUV-текстуре*. Во втором шаге создается *SubImage-индекс* в эмиттере частиц, и устанавливается кривая, указывающая в какое время какие кадры должны отображаться. Последним шагом необходимо поместить нод **ParticleSubUV** или **TextureParameterSubUV** в материал.

ПРИМЕЧАНИЕ

SubImage-индекс

SubImage-индекс определяет кривую, в которой горизонтальной оси соответствует значение от 0 до 1, где 1 — это продолжительность жизни частицы. Вертикальная ось представляет целое число, отражающее точное количество отображаемых кадров, начиная с 0. Поэтому если текстура имеет размер 4×4 , она имеет 16 кадров, и значение, которое будет использоваться, в итоге в *SubUV-текстуре* должно быть равно 15.

ПОПРОБУЙТЕ САМИ

Создайте эффект SubUV-текстуры

Использование SubUV-текстур — это один из лучших способов создавать глубокие и сложные эффекты с большим количеством вспомогательной анимации. Выполните следующие шаги, чтобы создать новый простой эффект эмиттера частиц и настроить материал, используя SubUV-текстуру взрыва.

- На панели **Content Browser** в папке проекта *Hour_10* (доступной на сайте книги) создайте новый шаблон частицы и назовите его **SimpleExplosion**.
- Откройте редактор **Cascade** для шаблона **SimpleExplosion**, дважды щелкнув по нему.
- Удалите модуль по умолчанию **Initial Velocity**.
- Откройте детали модуля **Required** и прокрутите их вниз до категории **Sub UV**. В этой категории установите параметр **Interpolation Method** в значение **Linear** и задайте параметрам **Sub Images Horizontal** и **Sub Images Vertical** значение **6**.
- Щелкните правой кнопкой мыши по эмиттеру частиц и выберите вариант **SubUV ⇒ SubImage Index**.
- Установите распределение в значение **Distribution Float Curve Constant**.
- Под полем распределения раскройте параметр **Point #1** и установите параметру **Out Val** значение **36**.
- На панели **Content Browser** создайте новый материал и назовите его **SimpleExplosion_Material**.
- Откройте материал **SimpleExplosion_Material**.
- В общих настройках материала установите параметру **Blend Mode** значение **Translucent**, а параметру **Shading Model** — значение **Unlit**.
- Добавьте новый нод **Particle Color** в материал.
- Создайте нод **TextureSampleParameterSubUV** и затем установите параметр **Parameter Name** в значение **Particle SubUV**.
- Замените текстуру нода **Particle SubUV** текстурой */Game/Textures/T_Explosion_SubUV*.
- Создайте два нода **Multiply**.
- Соедините белые RGB-контакты обоих нодов **Particle Color** и **Particle SubUV** с первым нодом **Multiply**.
- Соедините вывод первого нода **Multiply** с вводом **Emissive Color**.
- Соедините альфа-вывод нода **Particle Color** и красный вывод нода **Particle SubUV** со вторым нодом **Multiply**.
- Соедините вывод второго нода **Multiply** с вводом нода **Opacity**.
- В редакторе **Cascade** шаблона **SimpleExplosion** выберите модуль **Required** и установите свойство **Material** в значение **SimpleExplosion_Material**.
- Уменьшите значение параметра **Spawn Rate** до **0,0** и создайте элемент **burst spawn** со значением **1,0**.

21. В модуле **Required** установите параметру **Duration** значение **1,0**.
22. В модуле **Lifetime** выберите вариант **Distribution Float Constant** распределения свойства **Lifetime** и установите параметру **Constant Value** значение **1,0**.
23. Поместите экземпляр системы частиц **SimpleExplosion** на уровень.

Запуск систем частиц

Некоторые эффекты частиц всегда должны быть активированы. В других случаях требуется больше контроля, чтобы точно регулировать время активации эффектов частиц.

Автоматическая активация

Большинство таких эффектов окружения, как огонь или ветер, как правило, не требуют отключения. В таких случаях параметр **Auto Activate** (активировать автоматически) в актере эмиттера может использоваться для упрощения размещения и активации эффектов.

Чтобы активировать систему частиц сразу после ее помещения в эмиттер частиц, выберите актер и взгляните на категорию **Activation**. Если установлен флагок **Auto Activate** в актере эмиттера, эмиттер автоматически активируется, когда начнется игра.

Активация систем частиц с помощью блюпринтов уровней

В некоторых случаях гейм-дизайнеры и разработчики нуждаются в более полном контроле того, когда и как должна производиться симуляция эффектов частиц. Блюпринты уровней и блюпринт-классы могут использоваться для лучшего контроля поведения активации различных эмиттеров.

В блюпринтах уровней можно получать прямой доступ к ссылке на актер эмиттера и использовать ее для контроля состояния активации системы частиц.

Перетаскивание ссылки на систему частиц дает доступ к двум чрезвычайно полезным нодам: **Activate** (активировать) и **Deactivate** (деактивировать, см. рис. 10.6).

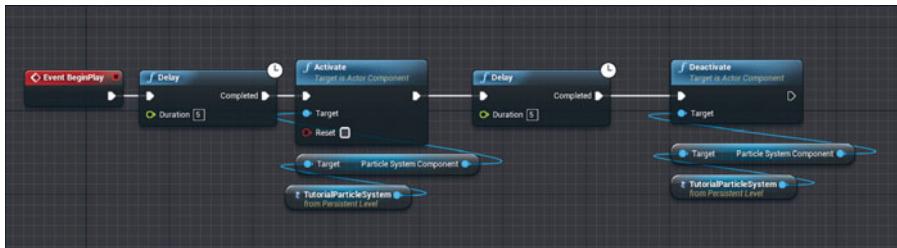


Рис. 10.6. Простая схема события, ожидающая в течение 5 секунд перед активацией системы частиц и еще 5 секунд перед деактивацией этой системы

Резюме

В этом часе вы узнали о модульном подходе UE4 к созданию и управлению эффектами частиц. Вы узнали о тесно взаимосвязанных отношениях между системами частиц и материалами, прикрепленными к ним. Вы увидели мощную технологию SubUV-текстур и научились запускать эффекты с помощью блюпринтов уровней. Создание систем частиц — глубокая тема и может потребовать времени на освоение, но основные принципы, раскрытие которых в этом часе, являются основой большинства возможных эффектов.

Вопросы и ответы

Вопрос: Я создаю разовый эффект, но он продолжает перезапускаться. Как это исправить?

Ответ: В модуле Required эмиттера, количество циклов частиц устанавливается в свойстве Emitter Loops. Если значение равно 0, эмиттер перезапускается бесконечно. Установите значение 1, чтобы эффект проигрывался только один раз.

Вопрос: Я подключил свою систему частиц к блюпринтам уровней, чтобы она активировалась после триггера, но она по-прежнему воспроизводится с самого начала уровня. Как сделать так, чтобы она запускалась в нужное время?

Ответ: Не забывайте убирать флажок Auto Activate в актере эмиттера на уровне. Свойство Auto Activate включено по умолчанию, и пока оно будет включено, эмиттер будет запускаться с самого начала игры.

Вопрос: Я пытаюсь использовать SubUV-текстуру, но отображаемая текстура выглядит неправильно. Мой эффект странно обрезан. Что не так?

Ответ: Наиболее частая причина странного или обрезанного вида текстур в неверных настройках SubUV. В node Required попробуйте изменить

свойства SubUV Horizontal и SubUV Vertical, чтобы они лучше соответствовали исходной текстуре.

Вопрос: Модули настройки цвета не влияют на мои частицы. Что могло произойти?

Ответ: Проверьте материал, который вы используете, чтобы убедиться, что нод ввода цвета частиц используется надлежащим образом. Свойства RGB должны быть подключены к тем каналам, на цвета которых вы хотите повлиять. Также вывод ALPHA должен быть подключен к каналу Opacity или Opacity Mask. Если каналы Opacity или Opacity Mask затемнены, проблема может заключаться в настройках материала. Убедитесь, что параметр Blend Mode установлен в значение Translucent или Masked.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Истинно или ложно высказывание: каждая система частиц может иметь только один эмиттер частиц.
2. Истинно или ложно высказывание: эмиттер не может иметь несколько одинаковых активных модулей в одно и то же время.
3. Истинно или ложно высказывание: значения кривых могут меняться на панели **Details** или в редакторе кривых.
4. Истинно или ложно высказывание: GPU- и CPU-частицы одинаковы, за исключением количества частиц, которое может симулироваться эффективно.

Ответы

1. Ложь. Система частиц может иметь любое количество эмиттеров.
2. Ложь. Некоторые модули могут суммироваться. В таком случае модули применяют свои атрибуты сверху вниз, складывая свои эффекты вместе. Некоторые модули игнорируют установленные ранее значения.
3. Истина. Редактор кривых предоставляет более интуитивный способ изменения распределений кривых, но использование панели **Details** отлично подходит для изменения атрибутов кривых.

4. Ложь. GPU-частицы могут симулироваться в гораздо большем количестве, но некоторые модули не могут использоваться с ними. Когда модуль несовместим с добавленным типом данных, возникает красный значок X и сообщение об ошибке, которое описывает проблему.

Упражнение

Попрактикуйтесь в создании новой системы частиц и используйте различные модули, доступные для управления эмиттерами частиц. Взгляните на систему частиц */Game/Particles/P_Explosion* и измените ее таким образом, чтобы получился взрыв голубой энергии.

1. В проекте *Hour_10* откройте систему частиц */Game/Particles/P_Explosion*.
2. В эмиттере **Shockwave** выберите модуль **Color Over Life**. Измените цвета 0 и 1 на голубой.
3. Измените материал модуля **Required** на предоставленный */Game/Material/M_explosion_subUV_blue*.
4. Измените кривую **Color Over Life** огненного шара на голубой цвет, выбранный в шаге 2.
5. Повторяйте шаги 2–4 для оставшихся модулей.

11-Й ЧАС

Использование актеров скелетных мешей

Что вы узнаете в этом часе

- ▶ Что такое скелетный меш и чем он отличается от статичного
- ▶ Импорт скелетных мешей из 3D-пакетов
- ▶ Использование редактора **Persona**
- ▶ Воспроизведение анимации нового актера скелетного меша

Зачастую появляется необходимость обрабатывать более сложные анимации, нежели простое перемещение трансформируемых компонентов статичных мешей. Один из способов создавать объекты, состоящие из различных частей, движущихся независимо друг от друга, заключается в использовании *скелетных мешей* (Skeletal Meshes). С их помощью вы можете вдохнуть жизнь в персонажей благодаря анимации. Эти анимации часто берутся из сторонних пакетов и импортируются в UE4. Чаще всего, когда вы играете контролируемым персонажем, этот персонаж представляет собой скелетный меш. В этом часе вы изучите мощь скелетных мешей, узнаете, как импортировать персонаж из стороннего пакета и как помещать и анимировать скелетный меш.

ПРИМЕЧАНИЕ

Подготовка к практике

Для работы в этом часе необходимо открыть проект *Hour_11*, находящийся в папке *Hour_11* (доступной в файлах проектов по адресу: http://addons.eksмо.ru/it/UE_24.zip), содержащей примеры анимации Unreal. Вы также можете воспользоваться магазином, чтобы добавить примеры контента самостоятельно.

Определение скелетных мешей

Если статичные меши создают игровой мир, то скелетные меши оживляют его. Главная разница между статичными и скелетными мешами отражена в их названиях: в статичном меше каждая вершина связана с одним местоположением — якорем

объекта; в скелетном меше вершинами управляет подобная скелету иерархия независимых местоположений. Эта иерархия может перемещать и анимировать отдельные части одного меша независимо друг от друга. Эта базовая способность позволяет анимировать сложные персонажи, монстров, животных, транспортные средства, механизмы и многое другое.



Рис. 11.1. Скелетный меш из проекта-примера UE4. Этот меш имеет совокупность костей, представляющую собой скелет (на скриншоте белые линии). Близлежащие вершины заскинены (skinned) к костям в сторонней программе, поэтому они будут следовать за скелетом во время анимации

На рис. 11.1 изображен скелетный меш UE4. В нашем случае это меш с наложенными текстурами (скином) сверху скелетной иерархии. Белые кости (или сочленения) — то место, где будет воспроизводиться анимация, а близлежащие заскиненные вершины меша будут деформироваться, следя за ними.

Термин *скининг*, или *заскиненный меш*, относится к процессу привязки вершин к лежащему под ними скелету. Дизайнер создает заскиненный меш с помощью стороннего инструмента, задавая связи между вершинами и соответствующими костями. Для этого процесса может использоваться множество пакетов и инструментов. Многие такие пакеты программного обеспечения имеют режимы визуализации, показывающие распределение весов скининга. На рис. 11.2 показан пример такой визуализации.

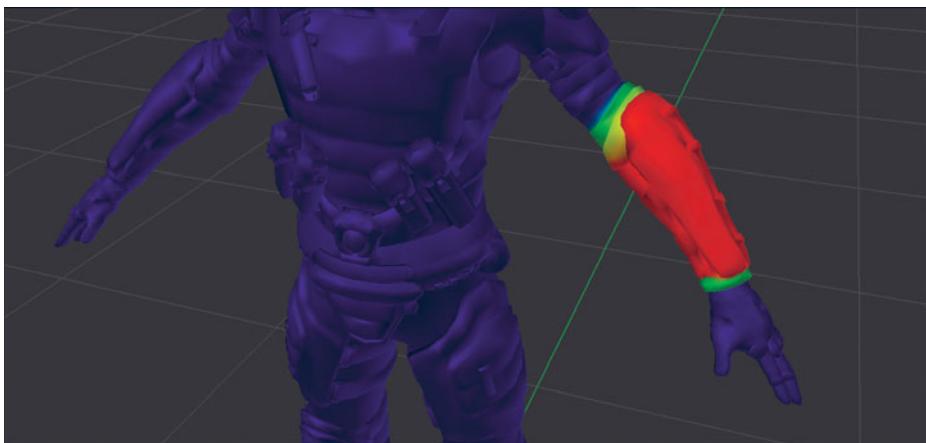


Рис. 11.2. Визуализация весов вершин левого предплечья персонажа в программе Blender. В этой визуализации красный цвет сигнализирует о том, что вершина будет полностью следовать за выбранной костью, желтый или зеленый цвета сообщают о том, что несколько костей влияют на позицию вершины, а синий означает, что на вершину не влияет выбранная кость

Когда вершины будут заскинены к скелету, аниматоры могут поворачивать, перемещать и масштабировать кости персонажа, чтобы создать динамичные анимации. На рис. 11.3 изображен анимированный персонаж.



Рис. 11.3. Простая поза удара персонажа Оуэна, представленная в Unreal Engine 4 в примерах контента

Анимирование скелетных мешей в Unreal Engine 4 требует по меньшей мере трех отдельных компонентов. Понимание того, что представляют собой эти компоненты и за что они по отдельности отвечают, обязательно для работы со скелетными мешами.

- ▶ **Скелетный меш.** Главный компонент — скелетный меш — это заскиненные вершины, идущие в наборе с внутренними костями. Этот компонент определяет отображение меша, включая то, как присвоен материал.
- ▶ **Скелет.** Каждый скелетный меш прикреплен к отдельному ассету, который называется *скелет* (*skeleton*). Один скелет может использоваться множеством скелетных мешей, но один скелетный меш может иметь только один скелет. Разные скелетные меши (с потенциально уникальной иерархией) могут быть анимированы UE4 с использованием одних и тех же ассетов. Такой уровень косвенности позволяет различным персонажам использовать одни и те же анимации.

Существует несколько правил использования скелетов скелетными мешами. Различные иерархии могут использоваться, только если базовая иерархия костей та же. Излишне спешное удаление костей из иерархии ломает связь между дочерними костями и скелетом. Кроме того, соглашения о названиях между скелетом и скелетным мешем должны совпадать. Все кости, контролируемые скелетом, должны иметь одинаковые названия в скелете и скелетном меше.

- ▶ **Последовательность анимации.** Последовательность анимации — это запись движения скелета, включающая местоположение ключевого кадра, угол поворота и масштаб каждой анимированной кости в скелете. Последовательность анимации может быть присвоена только одному скелету, поэтому, если необходимо использовать один и тот же ассет анимации двумя скелетными мешами, они могут просто использовать один и тот же скелет.

Вдобавок к этим трем необходимым компонентам анимации меша существует два других компонента скелетной системы. Эти компоненты не являются необходимыми, но они позволяют расширять поведение, особенно если вы создаете персонажей.

- ▶ **Ассеты физики.** При создании персонажей или скелетных мешей, которые должны взаимодействовать с системой физики, создается четвертый файл, который называется *physics asset*. Он определяет упрощенную геометрию коллизии, прикрепленную к скелету. Этот ассет позволяет персонажам становиться «тряпичными куклами» (*ragdoll*) после смерти или получать урон от лучей.
- ▶ **Блюпринты анимации.** Персонажи и меши, требующие сложную анимацию, часто используют блюпринты анимации. Эти специальные блюпринты отвечают за логику, необходимую для того, чтобы выбирать, какие последовательности анимации использовать в определенный промежуток времени. Блюпринты анимации часто отвечают за обработку анимированного передвижения, а также за смешивание различных анимаций на основе вводимых пользователем данных.

Рис. 11.4 показывает иерархию ссылок скелетных мешей, скелетов и анимаций, что хорошо иллюстрирует суть скелета.

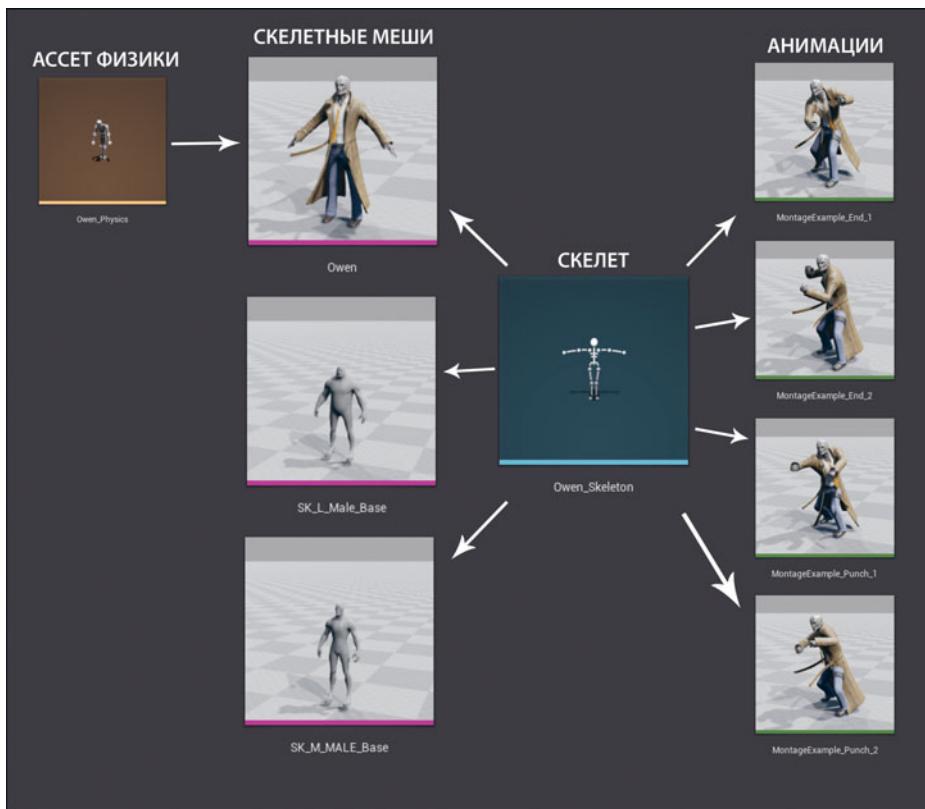


Рис. 11.4. На изображении показана иерархия ссылок скелетных мешей, скелетов, последовательностей анимации и ассетов физики. Каждая стрелка направлена от ссылки к источнику ссылки. Например, ассеты скелетов не знают, какие анимации используются ими, но каждая последовательность анимации знает, какой скелет к ней применен

Импорт скелетных мешей

Unreal Engine 4 — это игровой движок, и по большей части он не является инструментом создания контента. Как и в случае со статичными мешами или текстурами, для создания скелетного меша необходимо использовать стороннюю программу. UE4 взаимодействует со сторонними программами с помощью формата Autodesk Filmbox (.fbx). Файл .fbx содержит меш, скелет и данные анимации, необходимые для использования анимированных персонажей в UE4.

ПРИМЕЧАНИЕ

Пакеты программного обеспечения 3D-графики

Создание собственных персонажей может быть сложным процессом и потребовать программного обеспечения помимо UE4. Обычно используется несколько различных пакетов, каждого из которых вполне достаточно для работы. Maya, 3DS Max компании Autodesk и Houdini компании SideFX — это три профессиональных, платных приложения, каждое из которых имеет свою ценовую политику для независимых разработчиков. Помимо них существует инструмент Blender с открытым исходным кодом, который также имеет все необходимое для создания и анимирования персонажей.

Выполнение скининга и анимирования выходят за пределы этой книги. В этом часе описано, как выполнять импорт скелетных мешей и анимаций в UE4. При том есть несколько моментов, на которые следует обратить внимание. Прежде чем вы импортируете скелетный меш, он сначала должен быть экспортирован из пакета по созданию контента, в котором он был создан. Несмотря на то что каждый пакет программного обеспечения обрабатывает этот шаг немного по-разному, существует несколько хороших приемов экспорта из любого пакета.

- ▶ Триангуляция меша перед экспортом всегда предпочтительна.
- ▶ При экспорте лучше всего выбрать корень скелета меша и использовать такую опцию, как **Export Selected Only**.
- ▶ Необходимо включить опцию **Smoothing groups**.
- ▶ Необходимо включить опцию **Preserver edge orientation**.
- ▶ Необходимо отключить опцию **Tangent and binomials** (или **Tangent space**).

ПРЕДУПРЕЖДЕНИЕ

Масштаб единиц измерения редактора

Большинство пакетов контента принимает 1 единицу за 1 метр. Unreal Engine, в отличие от них, принимает 1 единицу за 1 сантиметр. Поэтому, когда вы работаете с пакетами по созданию контента, важно принимать во внимание эту разницу во время экспорта или, возможно, стоит внести изменения в редакторе программы по созданию контента, чтобы соответствовать масштабу единиц UE4. Если внести изменения в пакет по созданию контента не представляется возможным, вы можете установить опцию **Import Uniform Scale** в значение 0,01 в разделе **Transform Options** в диалоговом окне **FBX import** в UE4, чтобы исправить это несоответствие.

Blender-файл в папке *Hour_11/Raw/BlenderFiles/_UE4_StartupFile.blend* настроен таким образом, чтобы среда масштабирования программы Blender соответствовала масштабам UE4. Если вы будете использовать этот файл или если вы настроите используемый вами пакет так, чтобы 1 единица соответствовала 1 сантиметру, это предупредит необходимость постоянно менять масштабы импорта и экспорта.

ПРИМЕЧАНИЕ

Файлы примеров

Папка *Hour_11/RAW* (доступная на сайте книги) включает несколько тестовых ассетов, которые вы можете использовать, чтобы попрактиковаться в импорте. Некоторые связанные файлы сцены Blender также доступны в этой папке.

Импорт скелетного меша и создание скелета поначалу сводится к простому перетаскиванию файла с расширением *.fbx* на панель **Content Browser** или импорту с помощью щелчка по кнопке **Import**. Когда вы сделаете это, откроется диалоговое окно, показанное на рис. 11.5. Оно позволяет создать новый скелет для импортируемого меша, если его нет.

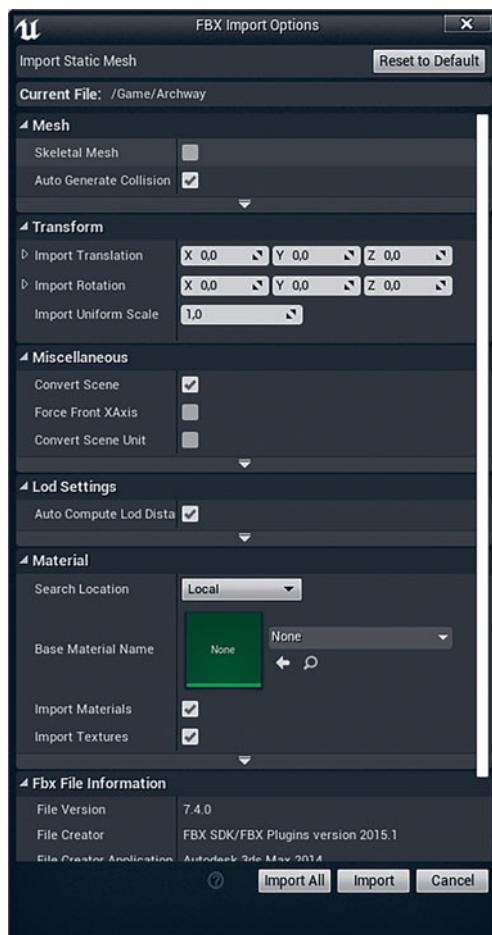


Рис. 11.5. Диалоговое окно **FBX Import Options**, открывающееся при обнаружении скелетного меша UE4

Важно обратить внимание на следующие свойства в диалоговом окне **FBX Import Options**.

- ▶ **Skeletal Mesh.** Если это свойство включено, меш считается скелетным. В противном случае импортируемый меш считается статичным.
- ▶ **Import Mesh.** Сброшенный флажок этой опции приводит к полному игнорированию меша со стороны UE4.
- ▶ **Skeleton.** Эта опция позволяет использовать существующий скелет. Если оставить ее пустой, будет создан новый скелет. При импорте скелетных мешей поначалу стоит оставлять ее незаполненной и использовать только последующие меши.
- ▶ **Import Animations.** Эта опция позволяет импортировать анимации. Вы можете импортировать анимацию одновременно с импортом первого скелетного меша и созданием скелета, но, как правило, лучше всего сначала импортировать меш без прикрепленных анимаций и персонажа в установочной позе. Чтобы импортировать только анимации без скелетных мешей, установите флажок рядом с опцией **Import Animations** и сбросьте флажок рядом с опцией **Import Mesh**.
- ▶ **Transform.** Некоторые пакеты программного обеспечения могут использовать установку сцены, к которой UE4 не может получить доступ. Используйте опции в разделе **Transform**, чтобы скорректировать различия. В общем случае лучше оставить значения по умолчанию и внести изменения в настройки пакета программного обеспечения для лучшего соответствия с UE4.
- ▶ **Import Materials.** В большинстве случаев, когда вы впервые импортируете скелетный меш, будет уместно импортировать материалы. Когда установлен флажок рядом с опцией **Import Materials**, новые материалы по умолчанию создаются вслед за скелетным мешем с теми же названиями, которые использовались в пакете по созданию контента. Поскольку большинство сторонних пакетов разработки не используют принятую в UE4 систему материалов, то, как правило, эти материалы приходится заменять по мере работы над проектом.
- ▶ **Convert Scene Unit.** Система координат расширения *.fbx* существенно отличается от принятой в UE4. В большинстве случаев флажок опции **Convert Scene Unit** должен оставаться установленным, если вы намеренно не изменили свой *.fbx*-файл, чтобы он соответствовал системе координат UE4. Если этот флажок установлен, UE4 преобразует ось Y+ большей части пакетов по созданию контента к оси Z+ движка Unreal Engine.

После того как вы щелкните по кнопке **Import** диалогового окна **FBX Import Options**, будут созданы запрашиваемые файлы, включая новый скелет. На рис. 11.6 показаны файлы, которые создает UE4 с параметрами настройки, указанными на рис. 11.5, установленными при импорте файла *Hour_11/Raw/HeroTPP.fbx*.

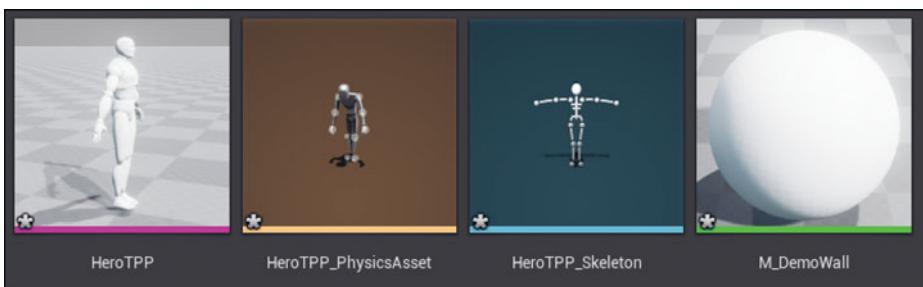


Рис. 11.6. Файлы, созданные диалоговым окном **FBX Import Options** с настройками, определенными на рис. 11.5

СОВЕТ

Кости отсутствуют в установочной позе

В зависимости от пакета 3D-графики, который вы используете для создания контента, *.fbx*-файл будет или не будет содержать информацию об установочной позе. Если он не содержит эту информацию, вы можете получить набор предупреждений во время импорта, которые начинаются с предложения:

«The following bones are missing from the bind pose:»

Если скелетный меш в вашем пакете 3D-графики находится в правильной установочной позе в нулевом кадре, необходимо импортировать скелетный меш вновь с включенной опцией *Use T0 as Ref Pose*. Эту опцию можно найти, щелкнув по стрелке для раскрытия расширенных настроек в разделе *Mesh* диалогового окна **FBX Import Options**.

Прежде чем перейти к обсуждению, как использовать и изменять эти ассеты, выполните упражнение из следующего раздела «Попробуйте сами», чтобы попрактиковаться в импорте скелетных мешей.

ПОПРОБУЙТЕ САМИ

Импортируйте героя

Первый шаг использования ассета всегда заключается в его получении движком Unreal Engine 4. Используйте представленный файл *HeroTPP.fbx* и выполните следующие шаги, чтобы попытаться импортировать новый скелетный меш.

1. В проекте *Hour 11* на панели **Content Browser** найдите папку *TryItYourself* и откройте в ней папку *_1* (доступную на сайте книги).
2. Щелкните по кнопке **Import** в левом верхнем углу панели **Content Browser**.
3. Используя диалоговое окно **Import**, найдите файл */Hour_11/RAW/HeroTPP.fbx* и щелкните по кнопке **Open**.
4. В открывшемся диалоговом окне **FBX Import Options** убедитесь, что для опций **Skeletal Mesh**, **Import Mesh** и **Use T0 as Ref Pose** установлены флажки. (Помните, что найти опцию **Use T0 as Ref Pose** можно, щелкнув по стрелке, раскрывающей список опций, внизу раздела **Mesh**.)
5. Убедитесь, что свойство **Skeleton** установлено в значение **None**, чтобы обеспечить создание нового скелета.
6. Сбросьте флажок свойства **Import Animations**.
7. Убедитесь, что установлен флажок для свойства **Convert Scene Unit** внизу диалогового окна.
8. Щелкните по кнопке **Import**.
9. Когда импорт будет завершен, сравните результат с результатом в папке *Content/TryItYourself/_1_*.

Импорт только анимаций аналогичен импорту скелетных мешей с незначительными отличиями. При импорте анимации для существующего скелета важно установить свойство **Skeleton** в диалоговом окне **FBX Import Options** к совместимому скелету. (Информация об иерархии костей должна быть той же, что и иерархия, которая использовалась для создания анимации.)

Кроме того, когда вы импортируете анимацию из пакета 3D-графики, важно убедиться, что установлен флажок рядом с опцией **Import Animation** и, как правило, лучше сбросить флажок с опции **Import Mesh**. Таким образом, вы избежите импорта и дубликации мешей при каждом импорте новой анимации.

Изучение редактора Persona

Unreal Engine 4 имеет специальный редактор — **Persona Editor** — для работы со скелетными мешами, скелетами, последовательностями анимации и блюпринтами анимации. Редактор **Persona** представляет комплексное решение для любых задач, связанных с анимированием персонажей.

Persona — это комбинированный редактор с отдельными режимами редактирования для различных типов ассетов, которые представляют анимированного персонажа. Такой стиль редактора позволяет использовать тесно взаимосвязанную суть ассетов анимации.

Двойной щелчок по типу аксете, который вы хотите отредактировать, автоматически откроет редактор **Persona** в соответствующем для этого аксете режиме редактирования.

Режим Skeleton

На рис. 11.7 показан режим Skeleton редактора **Persona**. Компоненты редактора включают следующие элементы управления: панель **Preview Scene Settings** (не показана на рисунке); 1) кнопка **Skeleton**; 2) панель **Skeleton tree**; 3) выпадооперт.

Компоненты пронумерованы на изображении и описаны ниже.

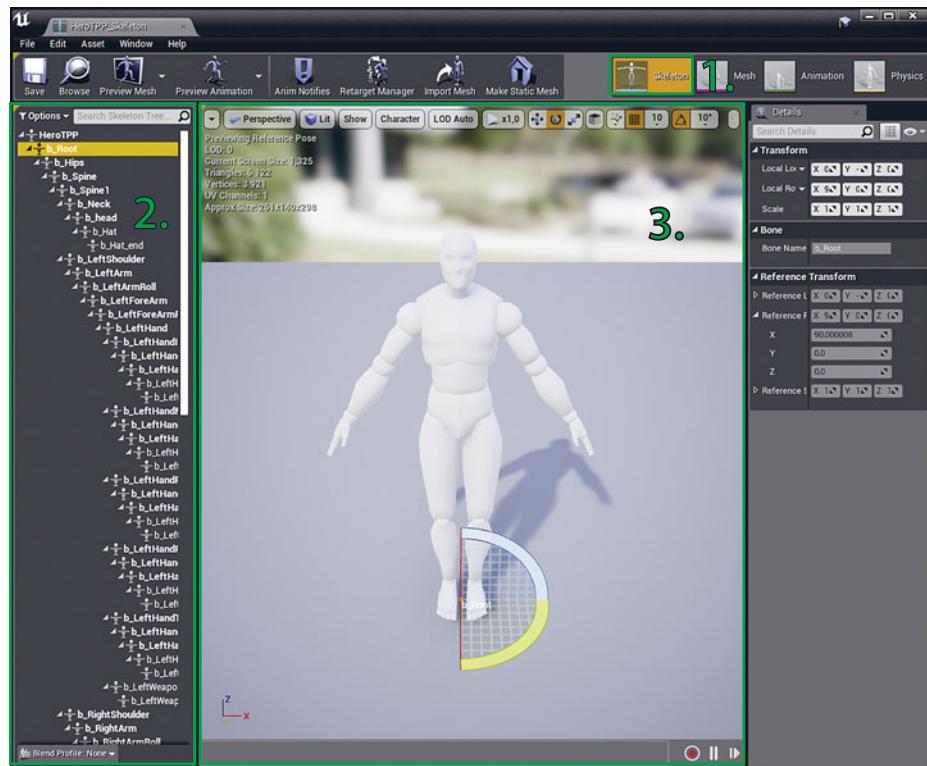


Рис. 11.7. Режим Skeleton редактора **Persona** с подсвеченными точками интереса

- ▶ Панель **Preview Scene Settings**. Эта вкладка отвечает за визуальные настройки сцены предварительного просмотра.
- ▶ Кнопка **Skeleton**. Эта кнопка запускает режим Skeleton редактора **Persona**. Она также содержит ссылку на аксет, что позволяет находить редактируемый скелет на панели **Content Browser**.

- ▶ **Панель Skeleton tree.** Эта область отображает иерархию костей в скелете. Когда вы выбираете кость в этой иерархии, вы можете затем управлять ею во вынопорте. Кроме того, щелкнув правой кнопкой мыши по любой кости, вы можете добавить сокет, который позволяет динамично добавлять дополнительные меши.
- ▶ **Вынопорт.** Вынопорт редактора Persona — это миниатюрная сцена, показывающая выбранный скелет и скелетный меш. В этом вынопорте можно изменять местоположение, поворачивать или масштабировать кости, а также просматривать анимации. Вы можете увидеть дополнительную информацию о меше в левом верхнем углу вынопорта.

Режим Mesh

На рис. 11.8 показан режим Mesh редактора Persona. Компоненты редактора включают следующие элементы управления: 1) кнопка Mesh; 2) категория параметров LOD; 3) категория Physics; 4) визуализатор LOD; 5) статистика вынопорта; 6) вкладка Morph Target Previews. Эти компоненты описаны ниже.

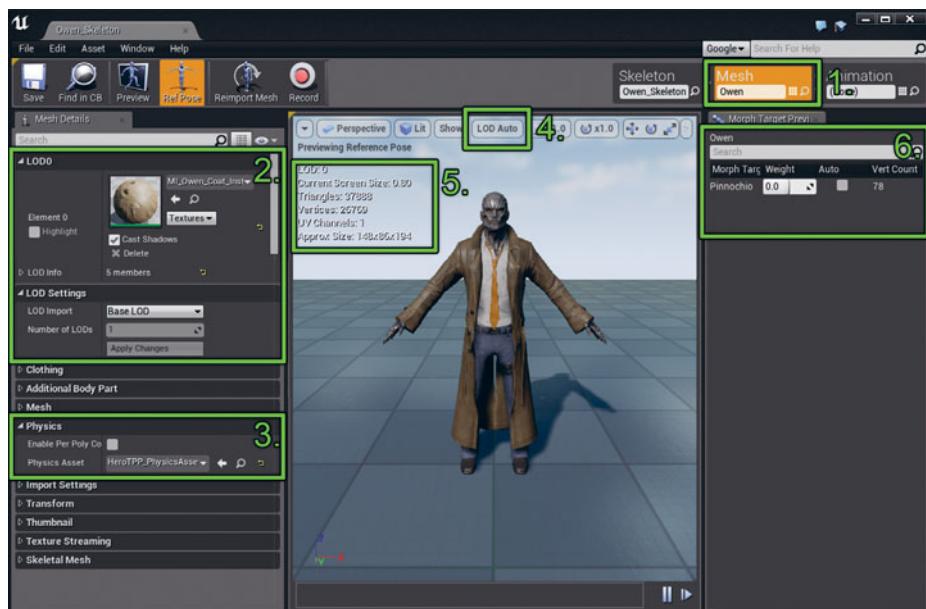


Рис. 11.8. Режим Mesh редактора Persona с подсвеченными точками интереса

- ▶ **Кнопка Mesh.** Эта кнопка запускает режим Mesh редактора **Persona**, она также содержит ссылку на ассет, что позволяет выбирать и редактировать меши в выпадающем меню.
- ▶ **Категория параметров LOD.** В этой части панели **Asset Details** (которая очень похожа на панель **Details** ассета статичных мешей) верхние категории относятся к информации об уровнях детализации и о материалах. В этом разделе можно дать команду UE4 генерировать дополнительные уровни детализации, на которых вы можете устанавливать материалы.
- ▶ **Категория Physics.** В категории **Physics** на панели **Asset Details** можно применять ассеты физики скелетных мешей. В ней также есть опция **Enable per Poly Collision**. По умолчанию флажок с этой опции сброшен, и в большинстве случаев это уместно. Многополигональные коллизии не могут использоваться для симуляции физики, такой как физика «тряпичной куклы», но может использоваться для запросов рейкастинга*. В большинстве случаев эта опция должна быть отключена, и вместо этого следует использовать ассет физики.
- ▶ **Визуализатор LOD.** Эта опция позволяет перезаписать отображаемый уровень детализации, позволяя визуализировать различные уровни детализации на меше. Параметр **LOD**, установленный в значении **Auto**, позволяет виджпорту автоматически отображать уровень детализации персонажа на основе параметров просмотра. Для этой работы должны быть созданы уровни детализации.
- ▶ **Статистика виджпорта.** В левом верхнем углу виджпорта можно увидеть часто используемую статистику об отображаемом меше. Она включает количество полигонов, запрашиваемый уровень детализации и приблизительный размер скелетного меша.
- ▶ **Вкладка Morph Target Previews.** Несмотря на то что этот час не раскрывает целевые объекты морфинга, вы все же можете просматривать их с помощью этой вкладки. В примерах контента анимации, предоставленных Unreal, например, можно найти образец целевого объекта морфинга Pinnochio в скелетном меше Оуэна. Изменение веса целевого объекта морфинга Pinnochio приводит к тому, что нос Оуэна становится очень длинным.

* Рейкастинг (ray casting) — метод рендеринга, при котором сцена строится на основе пересечения лучей с визуализируемой поверхностью. — Прим. ред.

Режим Animation

На рис. 11.9 показан режим Animation редактора **Persona**. Компоненты редактора включают следующие элементы управления: 1) кнопка **Create Asset**; 2) кнопка **Animation**; 3) панель **Details**; 4) панель **Asset Details**; 5) редактор **Anim Sequence**; 6) временная шкала; 7) аксет браузер. Эти компоненты описаны ниже.

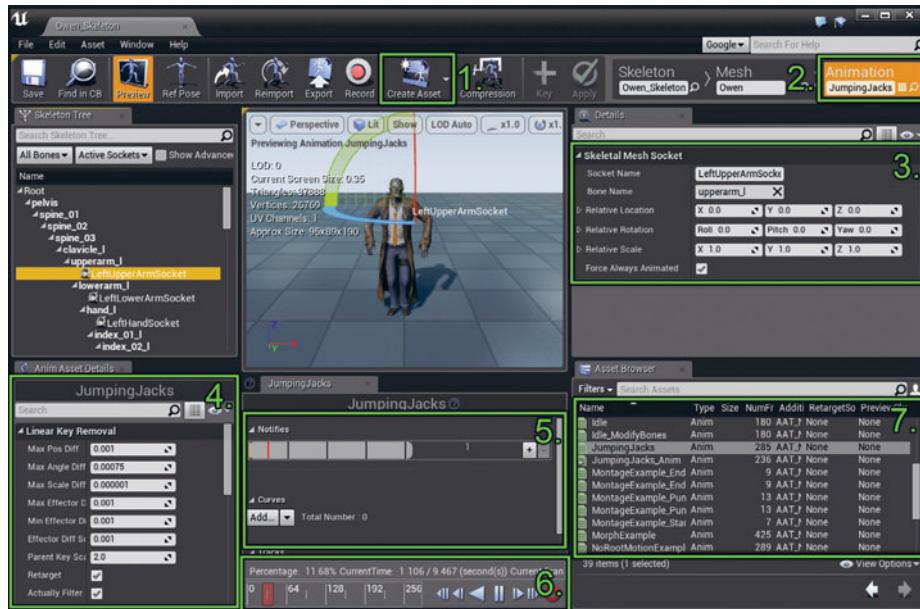


Рис. 11.9. Режим Animation редактора **Persona** с подсвеченными точками интереса

- ▶ **Кнопка Create Asset.** Эта кнопка предоставляет удобный способ создания нового монтажа, последовательностей анимации и другие аксессы анимации.
- ▶ **Кнопка Animation.** Эта кнопка запускает режим Animation редактора **Persona**. Она также содержит ссылку на аксет, что позволяет выбирать и редактировать меши в выпадающем меню.
- ▶ **Панель Details.** Эта панель используется для редактирования сокетов и свойств анимации.
- ▶ **Панель Asset Details.** Этот раздел показывает доступные для редактирования свойства выбранного аксета анимации в Аксет-браузере. Этот раздел можно использовать для изменения параметров, характерных для выбранного аксета.

- ▶ **Редактор Anim Sequence.** Многие ассеты анимации имеют свойства, основанные на временной шкале. В этом разделе вы можете редактировать кривые дополнительной анимации, уведомления и треки, используя временную шкалу и ключевые кадры.
- ▶ **Временная шкала (Timeline).** Вы можете изменять текущее время воспроизведения и команды управления анимацией. Перетаскивая красный указатель по временной шкале, вы можете перемещать точку воспроизведения в выбранной анимации.
- ▶ **Ассет браузер.** Этот быстрый браузер предназначен для ассетов анимации. Двойной щелчок по асsetу в этом браузере сделает асset выбранным для просмотра. Если вы наведете указатель мыши на асset, то во всплывающем окне в реальном времени отобразится предпросмотр анимации.

В режиме Animation вы можете создавать простые анимации только с помощью редактора **Persona**. Этот процесс не так изящен, как в большинстве пакетов по созданию 3D-контента, но вполне выполним, и для простых анимаций может быть весьма полезен. Чтобы его использовать, необходимо установить ключевые кадры анимации для различных костей, которые вы собираетесь анимировать в скелете.

Чтобы установить ключ в существующей последовательности анимации, выберите кость в скелете и затем щелкните по кнопке **+ Key** в панели инструментов (см. рис. 11.10). Затем, поверните или переместите кость (зажав клавишу **E**, чтобы отобразились манипуляторы поворота, или клавишу **W**, чтобы отобразились манипуляторы перемещения) в новое желаемое местоположение и затем нажмите кнопку **Apply** на панели инструментов, чтобы сохранить изменения. Затем переместите точку воспроизведения в новую точку на временной шкале и повторите процесс.

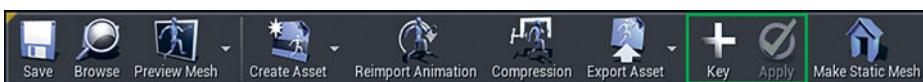


Рис. 11.10. Подсвеченные кнопки **+ Key** и **Apply** позволяют создавать собственные анимации прямо в редакторе **Persona**

После того как вы примените дополнительную анимацию к кости с помощью этого процесса, вы можете вручную изменить ключи с помощью редактора кривых в разделе редактора **Anim Sequence** под вышеперечисленными. Вы можете добавлять новые ключи щелчком правой кнопки мыши во вьюпорте **Track**. На рис. 11.11 показаны различные контекстные меню для редактируемой кости.

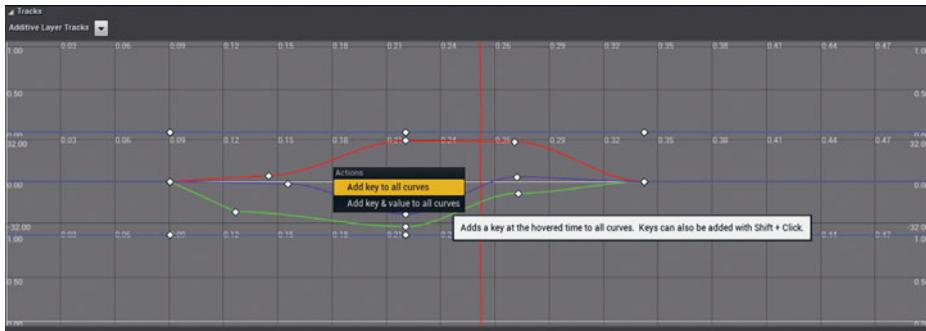


Рис. 11.11. Щелчок правой кнопкой мыши по дополнительному треку кривой позволяет добавлять новые ключи. Вы также можете вручную перемещать ключи в этом редакторе. Кроме того, выпадающее меню, открывающееся по щелчку по стрелке в левом верхнем углу, позволяет удалять или отключать отдельные треки

ПОПРОБУЙТЕ САМИ

Поверните головой героя

Иногда использования аддитивной анимации достаточно для создания простой анимации. Выполните следующие шаги, чтобы создать новую последовательность анимации, которая заставит Оуэна вертеть головой.

- На панели **Content Browser** проекта *Hour 11* найдите папку *TryItYourself/_1*. (Если вы не выполняли практику из первой рубрики «Попробуйте сами» этого часа, найдите папку *Hour_11/TryItYourself/_1_Result*.)
- Дважды щелкните по ассеуту скелетного меша **HeroTPP**, чтобы открыть редактор **Persona** для этого скелетного меша.
- Щелкните по вкладке **Animation** в правом верхнем углу редактора **Persona**.
- Щелкните по кнопке **Create Asset** на панели инструментов редактора **Persona**.
- Выберите вариант **Create Animation ⇒ Reference Pose**.
- Чтобы установить длительность анимации, щелкните правой кнопкой мыши по временной шкале внизу редактора **Persona** и выберите вариант **Append at the End**.
- В появившемся поле введите значение **119**, чтобы длительность анимации составила 120 кадров.
- Щелкните по выпорту и нажмите клавишу **E**, чтобы включить режим редактирования угла поворота. (Или вы можете щелкнуть по инструменту **Rotate** правом верхнем углу выпорта.)
- В поисковой строке скелета введите слово **Head** и затем выберите kostь **b_head**.

10. Перетащите слайдер временной шкалы в кадр 0, если он еще не был туда установлен.
11. Щелкните по кнопке **+ Key** на панели инструментов дважды, чтобы создать нулевой ключ в кадре 0.
12. Перетащите слайдер временной шкалы в последний кадр последовательности (кадр 120).
13. Вновь дважды щелкните по кнопке **+ Key** на панели инструментов, чтобы создать нулевой ключевой кадр в последнем кадре. Это позволит бесшовно зациклить анимацию.
14. Переместите слайдер временной шкалы в первую треть анимации около кадра 40.
15. Щелкните по кнопке **+ Key** и поверните голову персонажа влево приблизительно на 50°.
16. Вновь щелкните по кнопке **+ Key**, чтобы подтвердить изменения.
17. Переместите слайдер временной шкалы во вторую треть анимации около кадра 60.
18. Щелкните по кнопке **+ Key** и поверните голову персонажа вправо приблизительно на 50°. Кадр должен быть зеркально противоположным ключевому кадру 40.
19. Подтвердите изменения, еще раз щелкнув по кнопке **+ Key**.
20. Щелкните по кнопке **Apply** на панели инструментов, чтобы подтвердить внесенные изменения.
21. Щелкните по стрелке **Play** на временной шкале, чтобы просмотреть анимацию.
22. Сравните результат с последовательностью анимации, доступной в папке *TryItYourself/_2_Result*.

Режим Graph

На рис. 11.12 показан режим Graph редактора **Persona**, который безоговорочно является самым сложным режимом редактирования редактора **Persona**. Режим Graph работает в связке с ассетами блюпринтов анимации, чтобы обрабатывать логику смешения различных состояний и поведения анимации.

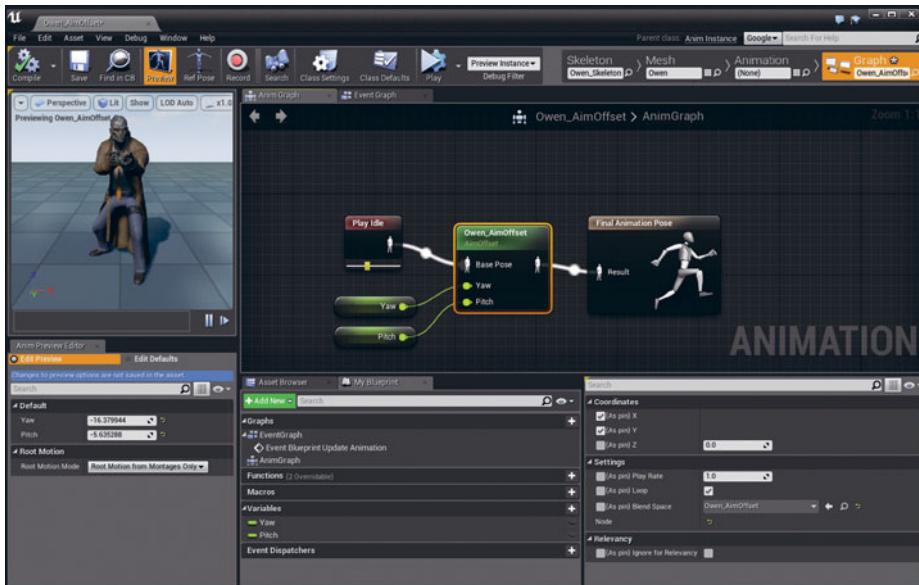


Рис. 11.12. Режим Graph используется для обработки логики сложного смешения различных анимаций и поведения

В отличие от других режимов, вкладка **Graph** появляется, только когда вы открываете блюпринт анимации.

Режим Graph очень важен при работе со сложными анимациями персонажей, особенно если они управляются пользователем. Блюпринты анимации управляют анимацией скелетных мешей с помощью ассетов Blend space или Aim offset, напрямую смешивая две последовательности анимации или даже управляя костями скелета.

СОВЕТ

Мощный набор инструментов, доступный в режиме Graph, поначалу может показаться трудным, а полное его описание выходит за пределы этой книги. Если вы хотите узнать больше о блюпринтах анимации и режиме Graph редактора **Persona**, взгляните на демонстрационную анимацию. Откройте папку *Hour_11/ExampleContent/AnimationDemo/AnimBlueprint* в проекте *Hour_11* (доступном на сайте книги), чтобы увидеть несколько различных примеров того, как блюпринты анимации справляются с различными задачами использования.

Кроме того, великолепная серия видео *3rd Person Game with Blueprints (v4.8)*, выпущенная Epic Games, углубляется в детали того, что представляют собой блюпринты анимации. Эта серия доступна на вики-сайте Unreal Engine по адресу: wiki.unrealengine.com.

Использование актеров скелетных мешей

Один из основных способов использования анимированных скелетных мешей заключается в простом помещении их на сцену в качестве актеров. К счастью, это так же просто, как помещение любых других актеров на сцену. Перетаскивание скелетного меша из панели **Content Browser** во вышупорт — эффективный способ помещения скелетного меша как актера.

Актеры скелетных мешей имеют некоторые уникальные свойства, которые стоит помнить (см. рис. 11.13).

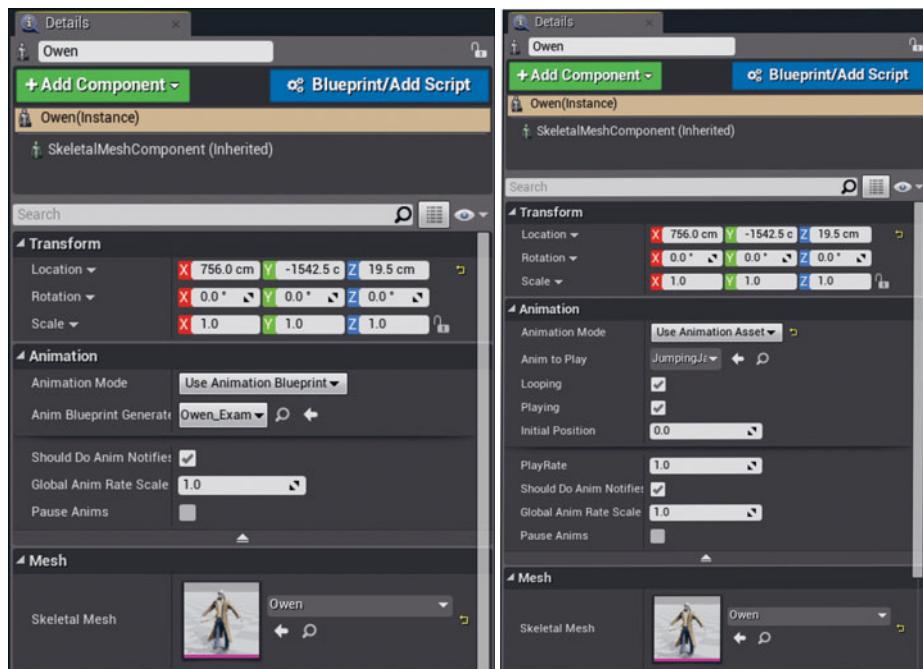


Рис. 11.13. Помещенный актер скелетного меша имеет параметры воспроизведения анимации мгновенно или через блюпринт анимации. На изображении показаны различные панели **Details** режима Animation

В категории свойств **Animation** актера статичного меша доступны команды для того, чтобы установить воспроизведение анимации, зацикливание, а также точку начала и скорость воспроизведения:

- ▶ **Animation Mode:** Это свойство позволяет указывать, использовать ли блюпринт анимации или отдельный ассет анимации. Выбор варианта **Use Animation Asset** делает доступными следующие опции.
- ▶ **Anim to Play.** Эта опция позволяет указывать ссылку из панели **Content Browser** на отдельный ассет анимации.
- ▶ **Looping.** Когда рядом с булевым параметром **Looping** установлен флажок, анимация будет воспроизводиться непрерывно.
- ▶ **Playing.** Когда рядом с этой опцией установлен флажок, булев параметр **Playing** означает, что анимация будет воспроизводиться с начала игры, без принудительного запуска с помощью блюпринтов или других способов. Сбросьте флажок, если хотите контролировать начало воспроизведения.
- ▶ **Initial Position.** Вы можете указать значение (в секундах), которое определяет время начала анимации. Изменение этого значения — хороший способ визуализировать позы, через которые проходит анимация.
- ▶ **PlayRate.** Это значение — коэффициент того, как быстро воспроизводится анимация. Если оно равно 0,5, анимация воспроизводится в половину от исходной скорости; если оно равно 2, анимация воспроизводится в два раза быстрее.

ПОПРОБУЙТЕ САМИ

Поместите героя

Анимации бесполезны, если они никогда не воспроизводятся. Выполните следующие шаги, чтобы поместить HeroTPP на новую сцену и настроить анимацию мотания головой, которую вы создали в предыдущей рубрике «Попробуйте сами».

1. Создайте новый уровень по умолчанию в проекте *Hour 11*.
2. В проекте *Hour 11* на панели **Content Browser** найдите папку *TryItYourself/_1*. (Если вы не выполняли практику из первой рубрики «Попробуйте сами» этого часа, найдите папку *Hour_11/TryItYourself/_1_Result*.)
3. Выберите скелетный меш **HeroTPP**, перетащите его в мир и поместите рядом с центром.
4. Выберите новый актер скелетного меша и на панели **Details** для параметра **Animation Mode** выберите вариант **Use Animation Asset**.
5. На панели **Content Browser** найдите папку *Hour_11/TryItYourself/_2*. (Если вы не выполняли практику из второй рубрики «Попробуйте сами» этого часа, найдите папку *Hour_11/TryItYourself/_2_Result*.) В этой папке выберите последовательность анимации **HeroTPP_Skeleton_Sequence**.

6. Убедитесь, что анимация по-прежнему выбрана, и щелкните по левой стрелке **Use Selected Asset from Content Browser** в свойстве **Anim to Play** или перетащите последовательность анимации в свойство **Anim to Play**.
7. Убедитесь, что рядом с параметрами **Looping** и **Playing** установлены флагки.
8. Щелкните по кнопке **Simulate** или **Play in Editor**, чтобы начать игру и увидеть анимацию персонажа.
9. Сравните получившийся результат с уровнем, доступным в папке *Hour_11/TryItYourself/_3_Result*.

Резюме

Скелетные меши занимают почетное место в наборе инструментов создателя игр. Благодаря этим невероятно многогранным инструментам, вы можете вдохнуть жизнь в разнообразные и интересные 3D-персонажи. В этом часе вы изучили мощь и силу скелетных мешей и узнали, как импортировать новый скелетный меш в редактор. Вы изучили множество опций, которые необходимо настроить, чтобы оживить скелетный меш, и узнали, как они работают вместе. Вы также научились использовать невероятно мощный редактор **Persona**, чтобы вносить изменения в скелетные меши и даже создавать простые анимации внутри Unreal Engine 4. Наконец, вы научились помещать актеры скелетных мешей на сцену и воспроизвести их анимацию.

Вопросы и ответы

Вопрос: Я не владею пакетами 3D-графики, с помощью которых можно создавать скелетные меши. Могу я вместо них использовать UE4?

Ответ: К сожалению, на момент написания книги движок UE4 не содержал каких-либо инструментов скиннинга или моделирования. Вы можете использовать один из пакетов в магазине Unreal, которые предоставляют готовые скелетные меши и анимации для ваших проектов. Также, помимо платных вариантов, компания Mixamo предоставляет 15 великолепных анимированных персонажей в магазине бесплатно.

Вопрос: При импорте анимаций меш выглядит сильно деформированным и неправильно установленным — в исходном пакете анимация воспроизводилась не совсем так. Что произошло?

Ответ: Как правило, если вы сталкиваетесь с чрезмерной деформацией из источника анимации, проблема заключается в несоответствии масштабов. Попробуйте установить параметр **Import Uniform Scale** в значение 0,01 или 100 в качестве теста. Если какой-либо из этих двух вариантов решит

проблему, уделите время тому, чтобы проверить, какой масштаб единиц измерения использует исходный пакет и нужно ли его изменить, чтобы он соответствовал единицам UE4 (1 unit = 1 см).

Вопрос: Я не хочу, чтобы мои анимации воспроизводились с начала игры. Вместо этого я хочу, чтобы воспроизведение начиналось позже. Как это сделать?

Ответ: В настройках актера скелетного меша сбросьте флагок с опции Playing. Вы можете использовать блюпринты уровней, чтобы заставить анимацию воспроизводиться вновь с помощью параметров Play или Play Animation из ссылки на ваш актер скелетного меша. (Чтобы узнать больше о блюпринтах уровней, обратитесь к 15-му часу «Работа с блюпринтами уровней».)

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

- Истинно или ложно высказывание: последовательности анимации и ассе- ты могут воспроизводиться в любом количестве скелетов.
- Истинно или ложно высказывание: каждый скелетный меш требует нали- чия уникального аксета скелета.
- Истинно или ложно высказывание: UE4 принимает 1 единицу за 1 см.
- Истинно или ложно высказывание: в UE4 нельзя создавать анимации, они должны создаваться во внешних пакетах.

Ответы

- Ложь. Каждая последовательность анимации привязана к скелету, с кото- рым она импортирована. Тем не менее вы можете присваивать последо- вательности анимаций к новым скелетам, создавая тем самым уникаль- ные аксеты.
- Ложь. Несколько различных скелетных мешей могут использовать один и тот же скелет, благодаря чему одни и те же анимации могут воспроиз- водиться на различных скелетных мешах. Учитывая этот факт, все ске- летные меши, которые должны использовать один скелет, должны иметь одинаковую базовую иерархию костей и соглашения о названиях костей. Любые несоответствия в цепи требуют использования нового аксета ске- лета.

3. Истина. Это важно, когда вы переносите любые ассеты из одного пакета программного обеспечения в другой.
4. Ложь. Хотя создавать анимации проще в предназначенных для этого программах, вы можете создавать простые анимации скелетных мешей в редакторе **Persona**, используя опцию Additive Animation Tracks.

Упражнение

Попрактикуйтесь в помещении анимаций на сцену с помощью предоставленных примеров контента.

1. В проекте *Hour_11* (доступном на сайте книги) создайте новый уровень.
2. На панели **Content Browser** откройте папку *ExampleContent/AnimationDemo/Animations*.
3. Выберите несколько анимаций скелетных мешей, таких как **Jumping Jacks** или **Run Right**, в этой папке и перетащите их на сцену.
4. Убедитесь, что анимации настроены на воспроизведение и зациклены, затем нажмите на кнопку **Simulate**.

12-Й ЧАС

Matinee и синематика*

Что вы узнаете в этом часе

- ▶ Работа с актерами Matinee
- ▶ Использование редактора **Matinee**
- ▶ Работа с актерами камер
- ▶ Перемещение и поворот актеров с течением времени

Этот час представляет редактор **Matinee** (**Matinee Editor**) движка UE4. Этот редактор позволяет создавать внутриигровые анимационные заставки, последовательности титров и анимации окружающей среды. Редактор **Matinee** — это инструмент, в первую очередь, для аниматоров и дизайнеров уровней. В течение следующего часа вы узнаете об актерах Matinee, редакторе **Matinee**, а также о том, как анимировать короткую последовательность.

ПРИМЕЧАНИЕ

Подготовка к практике

Для этого часа вы можете создать новый проект с шаблоном от первого лица и стартовым контентом или можете использовать готовый проект книги в папке *Hour_12* (доступной в файлах проектов по адресу: http://addons.eksmo.ru/it/UE_24.zip).

Актеры Matinee

Matinee в UE4 — это полный набор инструментов и конвейер для создания внутриигровой синематики. Matinee может использоваться для того, чтобы анимировать различные свойства множества типов актеров. Вы можете использовать

* Синематикой называется часть компьютерной игры, в которой игрок становится зрителем, временно не имея управления. Синематика добавляет в игру визуальное повествование. Хотя синематику иногда вульгарно называют видеороликом, в контексте этой книги мы будем рассматривать синематику на основе движка UE4. — Прим. ред.

этот инструмент для создания видео, и вы можете экспортить и импортировать данные ключевых кадров с помощью файлов с расширением *.fbx*. Вы можете создать столько актеров Matinee на одном уровне, сколько вам нужно, и вы можете контролировать актеры Matinee и запускать события посредством блюпринтов. Этот час расскажет вам о настройке и анимировании камер.

Прежде чем вы начнете использовать редактор **Matinee**, необходимо, чтобы на уровне был актер Matinee. Актер Matinee выполняет две основные функции: во-первых, он позволяет устанавливать предпочтительное поведение Matinee на уровне, и, во-вторых, он указывает на ассет данных Matinee, в котором хранятся группы, треки и ключевые кадры последовательности Matinee. Вы можете добавить актер Matinee на уровень двумя способами: как в случае с любым другим актером, можно перетащить актер Matinee из панели **Modes** или щелкнуть по иконке **Cinematics** на панели инструментов редактора уровней и выбрать вариант **Add Matinee**. После того как вы поместите актер Matinee на уровень, ему необходимо будет дать описательное название на панели **Details** уровня (см. рис. 12.1).

ПРИМЕЧАНИЕ

Добавление актера Matinee

Если вы добавите актер Matinee, щелкнув по иконке **Cinematics** на панели инструментов редактора уровней, редактор **Matinee** откроется автоматически. Чтобы увидеть свойства актера Matinee, необходимо закрыть редактор и выбрать его во выпорте уровня.

Свойства актеров Matinee

Если выбрать актер Matinee и взглянуть на его свойства на панели **Details**, можно увидеть свойство **Play Rate** в разделе **Play** (см. рис. 12.1). Это значение нормализовано, поэтому значение 1 означает 100% скорость воспроизведения, а значение 0,5 составляет 50% от скорости или вдвое меньшую скорость. Также можно увидеть опцию **Play on Level Load**, которую можно выбрать, чтобы начать воспроизведение Matinee сразу после загрузки уровня в память. Кроме того, опция **Looping** позволяет установить повтор воспроизведения. Следующий раздел, который важно рассмотреть, — это **Cinematic**. В нем можно найти возможность временно включать и отключать ввод движения и поворота игрока, вы также можете скрывать игрока и HUD во время воспроизведения Matinee. Эти опции обычно нужны для актеров Matinee, которые используют группы Director и актеры камеры.

ПРИМЕЧАНИЕ

Скорость воспроизведения и количество кадров в секунду

В то время как Matinee использует нормализованное значение для определения скорости воспроизведения, точное количество кадров в секунду (FPS, frames per second) устанавливается в редакторе Matinee в настройках привязки.



Рис. 12.1. Свойство актера Matinee

Редактор Matinee

Редактор Matinee — это интерфейс редактирования последовательностей анимации (см. рис. 12.2). Чтобы его открыть, выберите актер Matinee и на панели Details щелкните по кнопке **Open Matinee**. Ключевые области редактора Matinee выделены на рис. 12.2 и описаны ниже.

- Строка меню.** Стока меню используется для того, чтобы выполнять такие операции, как растягивание ключевого кадра, а также импорт и экспорт данных Matinee в виде файлов с расширением *.fbx*.
- Панель инструментов.** Панель инструментов содержит такие частые операции, как воспроизведение и настройки привязки.
- Редактор кривых.** Редактор кривых позволяет детализировать данные интерполяции с помощью сплайнов.
- Панель Tracks.** Эта панель позволяет управлять группами и треками, а также устанавливать ключевые кадры.
- Панель Details.** Панель Details отображает свойства выбранного трека или группы.
- Временная шкала.** Временная шкала указывает время в последовательности.

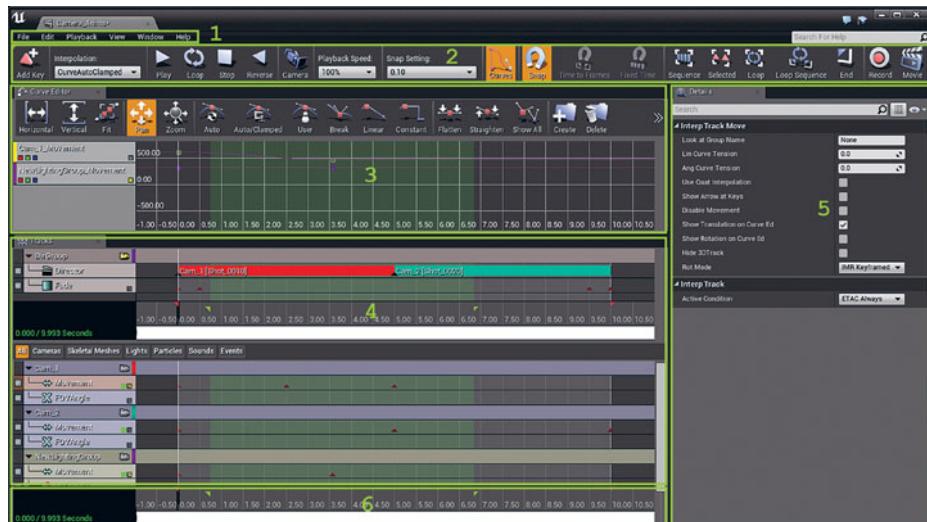


Рис. 12.2. Редактор Matinee

ПРИМЕЧАНИЕ

Работа с редактором Matinee

Значительная часть работы в Matinee требует, чтобы редактор Matinee и вьюпорт уровня были видимы одновременно. Хотя использование двойных мониторов не является обязательным, наличие двух мониторов, а также мыши с колесом прокрутки дает существенное преимущество при работе с Matinee.

ПРИМЕЧАНИЕ

Полезные команды и комбинации клавиш при работе с Matinee

Ниже приведены некоторые полезные комбинации клавиш для использования в Matinee.

- ▶ Прокручивание колеса мыши вверх или вниз приближает или отдаляет временную шкалу.
- ▶ Щелчок по иконке последовательности инструмента Matinee приводит временную шкалу в соответствие с вьюпортом.
- ▶ Нажатие клавиши **Enter** добавляет ключевой кадр в выбранный трек в текущей позиции времени.

Панель Tracks

Панель Tracks — это та область, в которой вы будете тратить большую часть времени при работе с Matinee. Эта панель использует группы и треки для организации контента. В верху панели можно увидеть операции отображения групп на основе типов присвоенных им актеров. Слева находится список групп и треков, а внизу — счетчик кадров и времени, указатель воспроизведения и продолжительность последовательности.

Установка продолжительности последовательности

Внизу панели Tracks можно найти информацию о шкале времени, указатель воспроизведения и текущее время, которое определяется положением указателя. Общая продолжительность последовательности выделена метками в виде красных треугольников, а активная рабочая область — метками в виде зеленых треугольников, как показано на рис. 12.3. Чтобы установить продолжительность последовательности, необходимо разместить метки в виде красных треугольников. Щелкните по красной метке справа и перетащите ее вправо, чтобы добавить время, или влево, чтобы сократить продолжительность. Если вам не хватает свободного места, помните, что вы можете прокрутить колесо мыши, чтобы отдалить временную шкалу, в результате чего на экране будет отображено больше времени.

ПРИМЕЧАНИЕ

Мертвая зона

Многие новички ошибочно полагают, что последовательности заканчиваются на последнем ключевом кадре. Это не так. **Matinee** воспроизводит все до метки в виде красного треугольника. Если вы установили ключи, но между последним ключом и правой красной меткой есть мертвая зона, вы можете перетащить метку к последнему ключевому кадру.

Указатель воспроизведения

Указатель воспроизведения, или метка времени, изображенный на рис. 12.3, показывает, на каком участке временного отрезка вы находитесь при редактировании последовательности, и позволяет помещать ключевые кадры на трек в определенные точки времени. Щелчок по пустому пространству внизу временной шкалы помещает указатель в эту точку. Вы можете щелкнуть по указателю и перетащить его, чтобы вручную перемотать последовательность. На рис. 12.3 выделены следующие области: 1) группа; 2) треки; 3) указатель воспроизведения / строка ручного воспроизведения; 4) ключ; 5) красные метки; 6) зеленые метки.

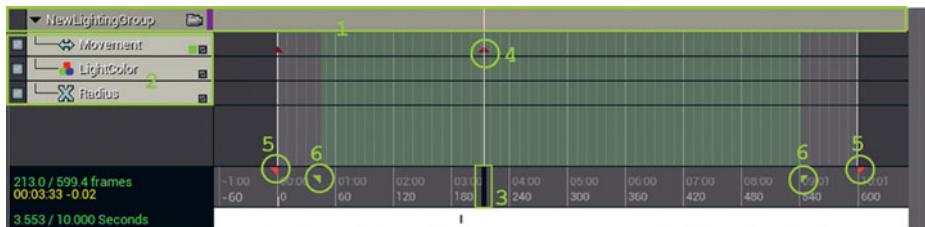


Рис. 12.3. Красные и зеленые метки, указатель воспроизведения, счетчик времени и кадров

Группы

В **Matinee** группы хранят актеров или актеры, которые необходимо контролировать, а также треки, которые используются для того, чтобы влиять на входящие в группу актеры (см. рис. 12.4). Существует пять пресетов групп: **Empty group**, **Camera group**, **Particle group**, **Skeletal group** и **Lighting group**. За исключением пустой группы (**Empty group**) все группы имеют изначально присвоенные им треки для работы над определенными типами актеров, указанными в их названиях. Пустые группы работают с любыми типами актеров, но необходимо вручную присваивать им ваши собственные треки, основываясь на анимируемом актере и его свойствах.

Чтобы добавить группу, необходимо сначала поместить актер, который вы собираетесь контролировать, на уровень. Затем, выделив актер, вы можете добавить группу в Matinee. Matinee автоматически назначает актер в группу. Чтобы добавить новую группу, щелкните правой кнопкой мыши по пустой области списка групп и треков в левой части панели Tracks. В открывшемся диалоговом окне выберите группу, соответствующую помещенному актеру. Если ни одна группа не соответствует вашим типам актеров, выберите вариант **Empty group** и дайте группе название. После этого вы сможете назначать в группу новые треки и устанавливать ключевые кадры.

СОВЕТ

Назначение нового актера в существующие группы

Если необходимо поменять актер, находящийся в группе, выберите новый актер во выпорте уровня и щелкните правой кнопкой мыши по названию группы на панели Tracks. В открывшемся диалоговом окне выберите вариант **Actors ⇒ Replace Group Actors with Selected Actors**. Когда вы добавляете актер статичного меша в группу, Matinee автоматически меняет его мобильность на movable.

Треки

Треки используются для того, чтобы динамично (в течение времени) устанавливать и хранить информацию о ключевых кадрах и их свойствах; треки назначаются в группы (см. рис. 12.4). Например, трек передвижения хранит данные о положении и повороте актера. Существует более 16 предопределенных треков. Для некоторых свойств актеров, которые вы решите анимировать, может существовать трек, не отраженный в списке треков. Тип данных каждого свойства определяет, какой тип трека вам нужен. Например, масштаб актера статичного меша может быть анимирован, но в списке треков нет трека масштабирования. Некоторые свойства определены их типом данных. Масштабирование, например, хранит масштаб актера в виде вектора (X, Y, Z). Поэтому, если вы хотите изменить масштаб актера, вам понадобится добавить трек свойства вектора в группу, в которую входит ассет статичного меша. Matinee показывает список свойств типа данных, на которые можно повлиять. Другой пример: если вы хотите изменить интенсивность актера точечного ИС, который хранится в виде числа с плавающей запятой, вам понадобится трек свойства числа с плавающей запятой. Добавление этого трека в группу, содержащую актер точечного ИС, приведет к тому, что отобразится список всех свойств актера точечного ИС, использующие числовые значения.



Рис. 12.4. Группы и треки

См. 14-й час, «Введение в систему визуального программирования блюпринтов» для получения более полной информации о типах данных переменных.

ПРИМЕЧАНИЕ

Ключевые кадры

Ключ хранит данные о настройках свойств с течением времени. Конкретный тип данных зависит от используемых актера и трека. Практически все свойства актеров могут быть анимированы. При воспроизведении **Matinee** производит интерполяцию между хранящимися данными в каждом ключевом кадре.

Папки

Папки используются для организации групп в **Matinee**. Например, у вас может быть четыре или пять групп, каждая из которых установлена для управления камерой. На панели **Tracks** вы можете создать папку для хранения только групп камер. Использование папок помогает сохранять организацию по мере роста сложности последовательности **Matinee**.

СОВЕТ**Порядок операций при установке ключевых кадров**

Добавить ключевой кадр довольно просто, но при этом важную роль играет порядок операций. Новичкам работы с **Matinee** следует запомнить следующий порядок действий: передвижение во времени, передвижение в пространстве, добавление ключа.

- ▶ **Передвижение во времени.** Передвижение во времени означает передвижение указателя воспроизведения на панели **Tracks** в точку, в которую вы хотите добавить ключ.
- ▶ **Передвижение в пространстве.** Передвижение в пространстве означает передвижение или поворот актера во вышпорте уровня, назначенного в группу, содержащую трек передвижения, в который вы добавляете ключ.
- ▶ **Добавление ключа.** Добавление ключа означает установку ключа в текущее время на соответствующем треке для сохранения изменений в значении.

ПОПРОБУЙТЕ САМИ**Анимируйте актеры статичных мешей и установите ключевые кадры в Matinee**

Выполните следующие шаги, чтобы установить ключевые кадры и создайте базовую повторяющуюся анимацию в **Matinee**.

1. Перетащите статичный меш в форме куба из панели **Content Browser** или со вкладки **Place** на панели **Modes**.
2. Перетащите актер **Matinee** со вкладки **Place** на панели **Modes** и поместите его рядом с кубом на уровень.
3. Выберите актер **Matinee** на уровне и поменяйте его название на панели **Details** на **move cube**. Затем щелкните по кнопке **Open Matinee**.
4. В редакторе **Matinee** щелкните правой кнопкой мыши в темной серой области в левой части окна и выберите вариант **Create New Empty Group**.
5. Дайте группе название **Cube_A**.
6. Чтобы назначить актер статичного меша в группу во вышпорте уровня, выберите куб, который вы поместили в первом шаге, щелкните правой кнопкой мыши по группе **Cube_A** в редакторе **Matinee** и выберите вариант **Actors ⇒ Add Selected Actors**.
7. Включите функцию привязки кадров, щелкнув по иконке с магнитом на панели инструментов **Matinee**, и убедитесь, что параметр настройки привязки установлен в значении **0,5**.
8. Чтобы установить продолжительность **Matinee** в 3 секунды, внизу панели **Tracks** перетащите красный треугольник в правой стороне в значение 3 сек.
9. Щелкните правой кнопкой по только что созданной группе и добавьте новый трек передвижения из списка треков.

10. Чтобы добавить ключ, убедитесь, что указатель воспроизведения находится на участке 0 сек. и что трек передвижения выбран. Затем щелкните по иконке **Add Key** на панели инструментов **Matinee** или нажмите клавишу **Enter**.
11. Переместите указатель воспроизведения в конец временного отрезка Matinee (3 сек., в данном случае) и добавьте второй ключ, щелкнув по иконке **Add Key** или нажав на клавишу **Enter**. Теперь у вас два ключа — один в начале и один в конце, хранящие одни и те же данные о местоположении и повороте в разных участках времени.
12. Переместите указатель воспроизведения в середину временной шкалы, на участок 1,5 сек., и при открытом окне редактора **Matinee** выберите помещенный куб во вьюпорте уровня и переместите его на ось Z на участок приблизительно 500 единиц.
13. Еще раз выберите трек передвижения и добавьте третий ключ, щелкнув по иконке **Add Key** или нажав клавишу **Enter**. Во вьюпорте уровня появится сплайн, показывающий путь куба на протяжении последовательности, который в данном случае представляет собой прямую линию.
14. Перемещая указатель воспроизведения назад и вперед, вы увидите, как куб движется вверх и вниз. Щелкните по кнопке **Play** или **Looping** на панели инструментов **Matinee**, чтобы запустить анимацию (см. рис. 12.5).

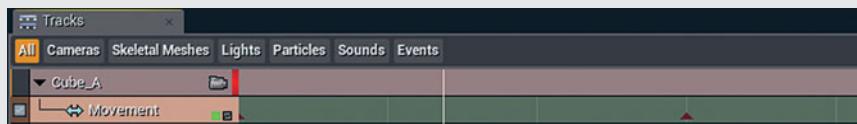


Рис. 12.5. Группы и треки с красными ключевыми кадрами

15. Чтобы установить свойства актера Matinee, закройте редактор **Matinee** и выберите актер Matinee во вьюпорте уровня. Затем на панели **Details** основного редактора установите флажок рядом со свойством **Play on Level Load and Looping**.
16. Запустите уровень.

Редактор кривых

Редактор кривых (**Curve Editor**) в Matinee дает возможность более точного контроля данных интерполяции, представленных в виде сплайна. Сплайн в данном случае — это визуальное представление интерполированных данных с течением времени. Практически все типы ключевых кадров трека могут быть отображены и изменены в редакторе кривых. Редактор кривых имеет собственную панель инструментов для работы с кривыми. Для отображения данных ключевых кадров трека в редакторе кривых необходимо установить флажок рядом

с параметром **Show on Curve Editor** — маленький флагок справа от названия трека (см. рис. 12.6). По умолчанию, когда этот флагок затемнен, параметр выключен, а когда флагок желтый — включен. На рис. 12.6 определены следующие области: 1) переключатель блокировки просмотра; 2) переключатель трека включенный или выключеный; 3) переключатель **Show on Curve Editor**; 4) показать путь сплайна во видпорте уровня.



Рис. 12.6. Установленный флагок **Show on Curve Editor**

Когда флагок **Show on Curve Editor** установлен, чтобы трек в редакторе кривых отображался, название группы и трека отображается в левой части редактора кривых. У вас может быть много треков, отображающих данные о кривых в редакторе кривых в одно и то же время при необходимости. Вам также, скорее всего, понадобится установить отображение в центр. Нажмите клавишу A, чтобы подогнать и отобразить все активные кривые. В зависимости от данных, которые хранятся в треке, у вас может быть одна или несколько кривых, связанных с треком. Например, трек fade в группе Director использует только численные значения, а трек movement имеет шесть кривых: три кривых положения (X, Y и Z) и три кривых поворота (тангаж, рыскание и крен). Вы можете включить или выключить видимость каждой отдельной кривой на панели Details редактора Matinee, когда трек выбран (см. рис. 12.7). Следующие флагки показаны на рис. 12.7: 1) флагок видимости перехода X; 2) флагок видимости перехода Y; 3) флагок видимости перехода Z.

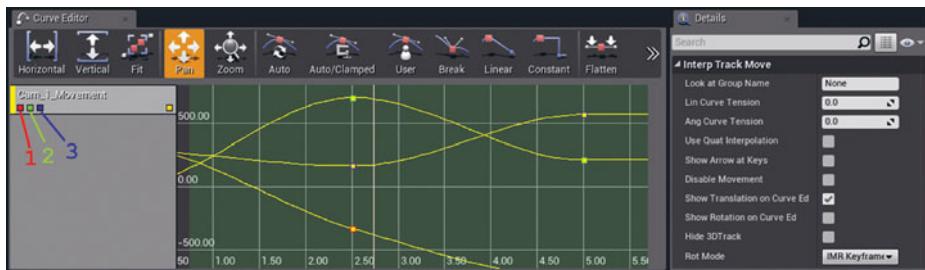


Рис. 12.7. Отображение флагжков кривых в редакторе кривых

ПРИМЕЧАНИЕ

Трек movement

Трек movement отображает только кривые положения по умолчанию. Чтобы отобразить данные поворота, необходимо выбрать трек movement и на панели Details редактора Matinee установить флажок рядом с параметром **Show Rotation on Curve Editor**. Вы также, возможно, заметили, что положение и поворот хранятся в одном и том же ключе. Если необходимо их разделить, вы можете щелкнуть правой кнопкой мыши по треку movement и выбрать вариант **Split Translation and Rotation**, но, как только вы внесете изменения, вы не сможете объединить их обратно, для этого придется удалить трек movement и начать сначала.

Режимы интерполяции

Режимы интерполяции, которые присваиваются отдельным ключам, определяют, как осуществляется переход сплайна от одного ключевого кадра к другому. Существует пять типов режимов интерполяции, но сейчас мы сосредоточимся на трех.

- ▶ **Curve (красный).** Этот тип режима используется для того, чтобы управлять эффектами исчезновения и появления. Этот режим используется, когда вы хотите отредактировать кривую в редакторе кривых.
- ▶ **Linear (зеленый).** Этот тип режима наслаждает изменения значений между ключевыми кадрами равномерно в течение времени.
- ▶ **Constant (черный).** Этот тип режима хранит значение последних ключевых кадров для всех промежуточных кадров до следующего ключевого кадра.

Чтобы изменить интерполяцию ключа, щелкните правой кнопкой мыши по ключевому кадру на панели Tracks и выберите вариант **Interp Mode**, а затем выберите желаемый режим.

ПОПРОБУЙТЕ САМИ

Используйте редактор кривых, чтобы детализировать интерполяции анимации

Выполните следующие шаги, используя редактор кривых для управления поворотом анимированного меша.

1. Откройте редактор **Matinee** для актера Matinee (MoveCube) из предыдущего упражнения «Попробуйте сами».
2. На панели **Tracks** добавьте новую группу и назовите ее **Cube_B**.
3. Поместите другой куб рядом с первым и назначьте его в группу Cube_B.
4. Добавьте трек movement в группу Cube_B.
5. Добавьте ключ в кадр 0 в только что созданный трек movement в группе Cube_B.
6. На панели **Tracks** переместите указатель воспроизведения в конец Matinee (3 сек., в данном случае).
7. Во выпорте поверните новый куб приблизительно на 300 градусов по оси Z.
8. Установите ключ. Теперь, если вы перетащите указатель воспроизведения или щелкните по кнопке **Looping** на панели инструментов редактора **Matinee**, вы увидите, как куб поворачивается по кратчайшей траектории. Вы можете добавить больше ключевых кадров, но это может выглядеть неаккуратно, потребует больше времени и затруднит редактирование в дальнейшем.
9. Включите отображение трека movement группы Cube_B в редакторе кривых и щелкните по серому блоку справа от трека movement. Теперь редактор кривых отображает только данные о положении.
10. Выбрав трек movement, на панели **Details** снимите флажок с параметра **Show Translation on Curve Ed** и установите флажок рядом с параметром **Show Rotation on Curve Ed**.
11. На панели инструментов редактора кривых щелкните по иконке **Fit**, чтобы отцентрировать и подогнать данные кривых в треке movement Cube_B. Поскольку только ось Z имеет изменение угла поворота, вы можете отключить отображение осей X и Y. В окне редактора кривых под треком movement Cube_B щелкните по красному и зеленому блокам, чтобы они стали серыми. Красный блок представляет ось X, зеленый — ось Y, а синий — ось Z.
12. В редакторе кривых щелкните правой кнопкой мыши по первому ключевому кадру и установите значение **0**.
13. Щелкните правой кнопкой мыши по второму ключу и установите для него значение **359**.
14. Запустите уровень и вы увидите, что куб теперь повернут на 360 градусов, и это движение повторяется. Но, поскольку параметр настройки интерполяции кривой по умолчанию установлен в значении **CurveAutoClamp**, куб появляется и исчезает, отчего куб замедляется и ускоряется. Чтобы изменить это, найдите трек movement группы Cube_B на панели **Tracks**, удерживайте клавишу **Ctrl** и щелкните по первому и последнему ключу, чтобы они оба были выбраны.

15. Щелкните правой кнопкой мыши по одному из выбранных ключей и выберите вариант **Interp Mode Linear**, чтобы выпрямить кривую так, чтобы данные о повороте были равномерно интерполированы по продолжительности анимации.
16. Запустите уровень. Вы должны увидеть, что куб поворачивается плавно и непрерывно вокруг оси Z.

Работа с другими треками

До настоящего времени в этом часе вы работали с треком Movement, одним из наиболее часто используемых треков в редакторе **Matinee**. Тем не менее существует гораздо больше треков, с которыми можно работать, такими как, трек Event для событий вызовов в блюпринтах с нодом Matinee Controller или трек Animation, используемый для воспроизведения последовательностей анимации в актерах скелетных мешей. Теперь вы будете использовать трек Sound для воспроизведения звуков в **Matinee**.

Трек Sound

Трек Sound в **Matinee** позволяет воспроизводить ассеты звуковых волн или ассе-ты звуковых сигналов в заданное время в последовательностях **Matinee**. Звуковые ассеты не обязательно должны быть актерами на уровне или входить в какую-либо группу. Вы можете просто добавить трек Sound в существующую группу.

Когда вы устанавливаете ключевой кадр в треке Sound, на панели **Content Browser** должен быть выбран ассет звуковой волны или звукового сигнала. Таким образом, будет установлен ключ, а в трек будет добавлен звуковой ассет. Название выбранного звукового ассета будет отображено на треке вместе с визуальным представлением его продолжительности. Установив ключ на звуковой ассет, вы можете изменить громкость и тон звука, щелкнув правой кнопкой мыши по ключу и выбрав вариант **Set Sound Volume or Sound Pitch**.

ПРИМЕЧАНИЕ

Продолжительность звуковых ассетов

Звуковые ассеты имеют уже установленную продолжительность, определенную импортированным исходным ассетом звуковой волны. Если вам нужен более короткий или долгий звук, вы можете отредактировать ассет в программе по редактированию звуковых файлов, такой как Audacity, или использовать другой звуковой ассет.

ПОПРОБУЙТЕ САМИ**Добавьте трек Sound в существующую группу**

Выполните следующие шаги, чтобы воспроизвести звук в Matinee в определенное время последовательности.

1. Откройте редактор **Matinee** для актера Matinee (MoveCube) из предыдущих упражнений «Попробуйте сами».
2. В списке групп и треков в правой части панели **Tracks** щелкните правой кнопкой мыши по группе **Cube_A** и выберите вариант **Add a New Sound Track**.
3. На панели **Content Browser** или стартовом контенте щелкните по ассете звуковой волны **Explosion01**.
4. Переместите указатель воспроизведения в кадр 0.
5. Выбрав трек Sound, добавьте ключ. После того как ключ будет помещен, вы увидите визуальное представление и название ассета звуковой волны в треке Sound.
6. Запустите Matinee, щелкнув по кнопке **Play** на панели инструментов **Matinee**.

Работа с камерами Matinee

Подлинная мощь Matinee заключается в работе с камерами для создания внутриигровых анимационных заставок. В заключительной части этого часа вы рассмотрите работу с актерами камер и группами камер в Matinee, а также на использование группы Director для переключения между камерами.

Группы и актеры камер

Группа камеры (Camera group) помещает актер камеры (Camera Actor) автоматически. Группа камеры изначально имеет трек **FOVAngle** и **movement** для анимирования актера камеры. В ней также есть другие свойства камеры, которые могут быть анимированы при наличии в группе актера камеры. Например, если вы добавите трек свойства числа с плавающей запятой в группу камеры, вы увидите обширный список свойств, которые могут быть ключевыми кадрами.

ПРИМЕЧАНИЕ**Актеры камер**

Группа камеры автоматически помещает актера камеры, но если у вас уже есть актер камеры, вы можете просто использовать любую пустую группу и назначать трек **movement** и/или **FOVAngle** при необходимости. Если вы добавили трек и затем решили, что он вам не нужен, вы можете просто выбрать трек и нажать на клавишу **Delete**, чтобы удалить его.

ПОПРОБУЙТЕ САМИ

Добавьте и анимируйте две камеры

Выполните следующие шаги, чтобы создать новую 10-секундную последовательность Matinee и анимировать две камеры.

1. Добавьте новый актер Matinee и назовите его **Camera_anims**.
2. На панели **Tracks** добавьте новую группу камеры и назовите ее **Cam_1**. Matinee автоматически добавит актер камеры на уровень, а также трек поля просмотра (FOVAngle) и трек movement в группу камеры. **Matinee** также добавит первый ключевой кадр в кадре 0. (Вы не должны анимировать FOVAngle, если не хотите. Вы можете оставить этот трек как есть или выбрать его и нажать на клавишу **Delete**, чтобы удалить его. Если в дальнейшем вы измените свое решение, вы можете просто добавить новый трек FOVAngle обратно в группу.)
3. На панели **Tracks** установите время Matinee, перетащив красный треугольник на правую сторону на участок 5 сек.
4. Включите иконку **Lock view camera** справа от группы Cam_1, чтобы изменить представление во вьюпорте на просмотр через камеру, которая назначена в эту группу. Затем, если вы переместитесь по уровню с открытым редактором **Matinee**, вы будете перемещать эту камеру. Не беспокойтесь, положение камеры не будет записываться до тех пор, пока вы не установите ключ. Если вы переместите указатель воспроизведения, камера вернется к последнему ключевому кадру.
5. С видом через камеру переместите указатель воспроизведения на участок 2,5 сек. и затем переместите камеру во вьюпорте.
6. Чтобы установить ключ камеры, убедитесь, что камера помещена в нужное место, выберите трек **Cam_1 movement** в редакторе **Matinee** и установите ключевой кадр.
7. Переместите указатель воспроизведения на участок 5 сек. и переместите камеру в новое положение. Установите последний ключевой кадр.
8. Чтобы добавить вторую камеру, переместитесь в новое положение во вьюпорте уровня, добавьте другую группу камеры и назовите группу **Cam_2**. Поскольку первая камера настроена на анимацию первых 5 секунд, вам необходимо анимировать вторую камеру на отрезке последующих 5 секунд. Помните, что первый ключевой кадр автоматически создается при добавлении группы камеры. Переместите указатель воспроизведения на участок 5 сек., не перемещая при этом новую камеру, и добавьте ключевой кадр.
9. Чтобы создать простое движение камеры, убедитесь, что иконка просмотра камеры не выбрана и переместите указатель воспроизведения к последним 10 секундам. (Если иконка просмотра камеры не выбрана, вы сможете перемещать и поворачивать камеру без необходимости смотреть через нее.)
10. Переместите вторую камеру по одной из ее осей примерно на 200 единиц и установите ключ.
11. Переместите указатель воспроизведения или щелкните по кнопке **Looping**, и вы увидите, что камера движется. Первая камера движется в течение первых 5 секунд, а вторая камера — в течение последних 5 секунд.

Группа Director

Группа Director — это уникальная группа, позволяющая вам выступать режиссером монтажа. Когда группа Director добавлена, она отображается вверху панели **Tracks**. В одном актере Matinee может быть только одна группа Director. Эта группа с треком Director, имеющим назначеннную ему группу Camera, осуществляя отображение от лица игрока во время воспроизведения Matinee. Она дает возможность переключаться между камерами во время использования трека Director и добавляет эффекты синематики, такие как появление и затемнение, а также замедленное движение.

Треки группы Director

Некоторые треки уникальны для группы Director, например треки **Fade** и **Slomo**. Треки группы Director влияют на всю последовательность, вне зависимости от используемой камеры. Установка ключевого кадра на трек в группе Director производится так же, как и на любой другой трек: переместите указатель воспроизведения в желаемый кадр, выберите трек и добавьте ключ. Ключевые кадры для большинства треков Director могут отображаться и редактироваться в редакторе кривых для последующей доработки.

Ниже перечислены треки группы Director и их назначение.

- ▶ **Director.** Этот трек переключает текущее отображение между группами Camera на всем протяжении последовательности.
- ▶ **Fade.** Этот трек позволяет устанавливать эффекты появления и затемнения на активной камере последовательности, которая определена треком Director. Значение ключевого кадра 0 означает, что затемнения нет, а значение ключевого кадра 1 означает, что экран черный.
- ▶ **Slomo.** Этот трек использует ключи для временного изменения скорости воспроизведения последовательности.
- ▶ **Audio master.** Этот трек управляет громкостью и тоном всех аудиотреков в последовательности.
- ▶ **Color scale.** Этот трек меняет оттенки отображаемых при воспроизведении последовательности Matinee кадров. Значения RGB устанавливаются в редакторе кривых.

Чтобы добавить трек в группу Director, щелкните правой кнопкой мыши по группе Director в вашем актере Matinee и добавьте желаемый трек из списка, как показано на рис. 12.8.

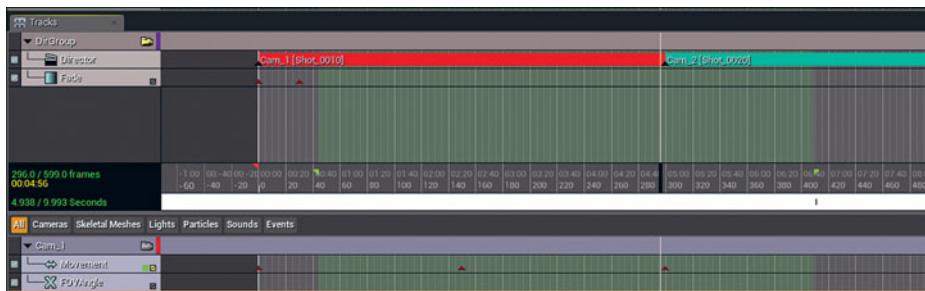


Рис. 12.8. Группа Director с треком Director и треком Fade

ПРИМЕЧАНИЕ

Группы Camera

Если вы собираетесь зациклить Matinee, не используйте трек Director в группе Director. Поскольку трек Director забирает контроль над тем, что видит игрок, то он застрянет на бесконечном просмотре Matinee.

ПОПРОБУЙТЕ САМИ

Используйте группу Director, чтобы переключаться между камерами

Используйте группу Director, чтобы менять камеры, через которые игрок будет смотреть.

- Щелкните правой кнопкой мыши по пустому пространству в левой части панели **Tracks** и выберите вариант **Add New Director Group**. Matinee разделит панель **Tracks** на две секции с группой Director вверху. Группа Director теперь имеет трек Director, являющийся уникальным для этой группы.
- Переместите указатель воспроизведения в кадр 0 и, выбрав трек Director, добавьте ключевой кадр. В появившемся диалоговом окне выберите вариант **Cam_1** и нажмите **OK**.
- Переместите указатель воспроизведения на 5-ю секунду и добавьте еще один ключевой кадр в трек Director, затем выберите вариант **Cam_2** и нажмите **OK**. Теперь во время воспроизведения Matinee переключает отображение между двумя группами камер.
- Щелкните по иконке камеры в группе Director, чтобы включить вид группы Director, и вручную промотайте анимацию или щелкните по кнопке **Looping** на панели инструментов **Matinee**. Вы должны видеть точки обзора обеих камер при воспроизведении Matinee.
- Чтобы подготовить актер Matinee так, чтобы он воспроизводился во время загрузки уровня, закройте редактор **Matinee**, выберите актер Matinee и установите флагок рядом с параметром **Play on Level Load** на панели **Details**.
- Запустите уровень.

Работа с ассетами данных Matinee

По умолчанию каждый раз, когда вы помещаете новый актер Matinee, вы создаете ассет Matinee, прикрепленный к актеру. В большинстве случаев это все, что вам нужно. Если ваш проект все же нуждается в повторном использовании одной и той же синематики и анимации повторно, может быть уместно создать ассет данных Matinee. Ассет данных Matinee хранит данные о группе, треке, папке и ключевом кадре и доступен на панели **Content Browser**, то есть вы также можете использовать его с актерами Matinee и на других уровнях.

Ниже приведены шаги по созданию вашего собственного ассета данных Matinee.

1. Создайте папку и назовите ее *MatineeData* для вашего проекта на панели **Content Browser**.
2. Щелкните правой кнопкой мыши в папке, выберите вариант **Create Advanced Asset** ⇒ **Miscellaneous** и щелкните по пункту **Matinee Data**.
3. Назовите новый ассет данных Matinee и сохраните его.
4. Поместите актер Matinee на уровень. На панели **Details** для этого актера щелкните по выпадающему списку Matinee Data Properties и выберите ассет данных Matinee, который вы создали в шагах 2 и 3.

Теперь вы можете присваивать этот ассет данных стольким актерам Matinee, скольким потребуется. Если вы отредактируете данные, ассет обновится во всех актерах Matinee, относящихся к этому ассету.

Резюме

В этом часе вы узнали, как работать с редактором **Matinee**, чтобы анимировать статичные меши и актеры камер. Рабочий процесс Matinee относится, в основном, к созданию анимационных заставок и управлению анимированными последовательностями окружающей среды. Если вы хотите создать анимированные ассеты, с которыми игрок может взаимодействовать, необходимо использовать блюпринты и времененную шкалу, что описано в 16-м часе «Работа с блюпринт-классами».

Вопросы и ответы

Вопрос: Игрок по-прежнему имеет контроль над аватаром во время воспроизведения синематики. Как можно это отключить?

Ответ: На панели Details для помещенного актера Matinee найдите раздел Cinematic. В нем вы можете отключить возможность передвижения игрока и скрыть аватара или HUD на время воспроизведения Matinee.

Вопрос: Можно ли выбрать несколько ключевых кадров одновременно?

Ответ: Да, на панели Tracks или в редакторе кривых вы можете нажать комбинацию клавиш Ctrl+Alt+ЛКМ+перетаскивание, чтобы создать выбор перетаскиванием, либо вы можете нажать Ctrl+ЛКМ, чтобы добавить текущий выбор.

Вопрос: Как изменить положение отдельного ключевого кадра?

Ответ: Выберите ключ, удерживайте клавишу Ctrl и перетащите его в новое местоположение на временной шкале.

Вопрос: Как удалить группу, трек или ключевой кадр?

Ответ: Просто выберите группу, трек или ключевой кадр и нажмите клавишу Delete.

Вопрос: Я создал отличную анимацию, но она воспроизводится слишком быстро. Могу ли я изменить расстояние между ключевыми кадрами в том же порядке, чтобы изменить продолжительность анимации?

Ответ: Да. Вы можете вручную выбрать и переместить каждый ключевой кадр, что может отнять много времени, в зависимости от количества ключевых кадров. Лучше всего перетащить выбранные кадры: выбрать ключи, которые хотите изменить, а затем выбрать вариант Edit ⇒ Stretch Selected Keyframes. В открывшемся диалогом окне вы можете установить новое время для выбранных ключевых кадров.

Вопрос: Я пытаюсь анимировать открывающуюся и закрывающуюся дверь, но якорная точка меша находится в неправильном положении. Как я могу изменить якорную точку актера статичного меша, который пытаюсь анимировать?

Ответ: Лучше всего отредактировать меш в приложении 3D-графики и повторно импортировать его. Но в качестве обходного пути вы можете прикрепить меш двери к родительскому актеру и затем анимировать родительский актер. Родительский актер становится в таком случае якорной точкой. Вы можете установить для родительского актера параметры отображения на панели Details так, чтобы он был невидим в игре, тогда будет видна только дверь.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Истинно или ложно высказывание: можно изменить актер, который управляет группой в **Matinee**.
2. Истинно или ложно высказывание: камеры должны быть анимированы в отдельных актерах **Matinee**.
3. Истинно или ложно высказывание: масштаб статичного меша не может быть анимирован.
4. Истинно или ложно высказывание: ассет данных **Matinee** может использоваться на разных уровнях.

Ответы

1. Истина. Вы можете назначать и переназначать актеры в существующие группы в **Matinee**.
2. Ложь. Камеры и другие ассеты могут быть анимированы в одном актере **Matinee**.
3. Ложь. Масштаб статичного меша может быть анимирован с помощью трека **Vector Property**.
4. Истина. Ассеты данных **Matinee** хранятся на панели **Content Browser** и могут быть использованы в различных актерах **Matinee** на различных уровнях.

Упражнение

Создайте 15-секундную синематику эпично отрывающейся двери. Синематика должна использовать несколько камер, а также эффект появления на первой камере и эффект исчезновения на последней.

1. Создайте новую карту по умолчанию.
2. Добавьте актер **Matinee** на уровень.
3. На панели **Details** в разделе **Play** установите флажок рядом со свойством **Play on Level Load** для этого актера **Matinee**. Не добавляйте зацикливание.

4. На панели **Details** в разделе Cinematic установите флагки рядом со свойствами **Disable Movement Input**, **Disable Look at Inputs**, **Hide Player** и **Hide Hud**.
5. Создайте черновой набросок уровня, разместив только те актеры, которые хотите анимировать, такие как дверь и дверной проем.
6. Откройте редактор **Matinee** и добавьте необходимые группы и трек movement, чтобы анимировать элементы двери при ее открытии.
7. Используйте треки sound в каждой группе при необходимости.
8. Добавьте три группы камер и анимируйте каждую камеру.
9. Добавьте группу Director с треком Director, чтобы переключаться между камерами.
10. Добавьте трек Fade в группу Director и установите ключи для появления первой камеры и затемнения последней камеры.

13-Й ЧАС

Изучение работы с физикой

Что вы узнаете в этом часе

- ▶ Как заставить актер статичного меша имитировать физику
- ▶ Создание и назначение физических материалов
- ▶ Использование актеров физического ограничения
- ▶ Использование актеров толкателей физики и радиальной силы

Этот час вводит понятие физики в UE4. Вы начнете с изучения того, как установить простое твердое физическое тело из актера статичного меша. Затем вы взглянете на работу с физическими материалами и актерами ограничения. Вы закончите час, создав и использовав актеры силы. Симуляция физики — это объемная тема, поэтому целью этого часа является установка фреймворка и базовое понимание работы с физикой.

ПРИМЕЧАНИЕ

Подготовка к практике

В данном часе вы будете использовать проект *Hour_13* из проектов по адресу: http://addons.eksмо.ru/it/UE_24.zip. Предоставляемый проект имеет режим игры, использующий персонаж от первого лица с физическим ружьем, которое может использоваться для взаимодействия с актерами, симулирующими физику.

Использование физики в UE4

Физическое тело — актер, реагирующий на внешние силы и столкновения. Физика в UE4 обрабатывается физическим движком NVIDIA Phys X. Движок PhysX использует центральный или графический процессор в зависимости от системы для обработки твердых и мягких тел, одежду, разрушаемых объектов и частиц. Редактор UE4 имеет инструменты интерфейса для установки и изменения физических свойств. Этот час сосредоточен на работе с твердыми телами. *Твердое тело* — цельный, недеформируемый объект, например кусок дерева размером 2×4 или надувной мяч.

Основные понятия физики

Начиная работать с физикой, уместно познакомиться со следующими понятиями.

- ▶ *Физическое тело* (physics body) — это универсальный термин, используемый для описания любого объекта, симулирующего физику.
- ▶ *Твердое тело* (rigid body) — цельные, недеформируемые объекты.
- ▶ *Мягкое тело* (soft body) — это деформируемый объект, который подчиняется правилам окружающего мира при столкновении с чем-либо.
- ▶ *Одежда* (cloth) — это разновидность мягкого тела.
- ▶ *Разрушаемый объект* (destructible) — это понятие применяется к твердым телам, которые ломаются и распадаются, если к ним приложено достаточно сил.
- ▶ Термин *линейный* (linear) относится к направленной силе, которая меняет положение актера на уровне.
- ▶ Термин *угловой* (angular) относится к поворачивающим силам, меняющим ориентацию актера.
- ▶ Термин *масса* (mass) относится к количеству материи в данном теле, независимо от количества приложенной гравитации.
- ▶ *Плотность* (density) — это количество массы на единицу объема в заданном физическом теле.
- ▶ *Торможение* (damping) означает, как быстро физическое тело остановится после воздействия силы. Это рассеяние энергии с течением времени.
- ▶ *Трение* (friction) — это величина сопротивления, примененная к скользящему или катящемуся телу.
- ▶ *Восстановление* (restitution) относится к величине удара, который получает физическое тело, и тому, как быстро тело остановится.
- ▶ *Сила* (force) применяется к массе за период времени.
- ▶ *Импульс* (impulse) — мгновенный удар.

Назначение физического режима игры уровню

Чтобы протестировать симуляцию физики, потребуется аватар с физическим ружьем, чтобы вы могли взаимодействовать с физическими телами. Откройте предоставляемый для этого часа проект *Hour_13*. Он содержит установленный режим игры с физическим ружьем и несколько демокарт. Создайте новый уровень и откройте для него панель **World Settings**, щелкнув по иконке **Settings** в панели

инструментов основного редактора над окном виджета и выбрав пункт **Project Settings** (см. рис. 13.1).

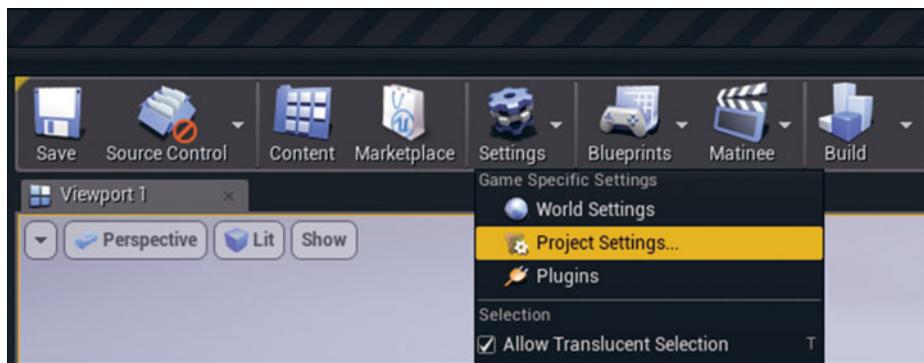


Рис. 13.1. Открытие настроек проекта

В разделе **Game Mode** панели **World Settings** установите параметр **GameMode Override** в значение **SimplePhysicsGameMode**, как показано на рис. 13.2. Протестируйте уровень, и увидите простой указатель в виде красного прицела.

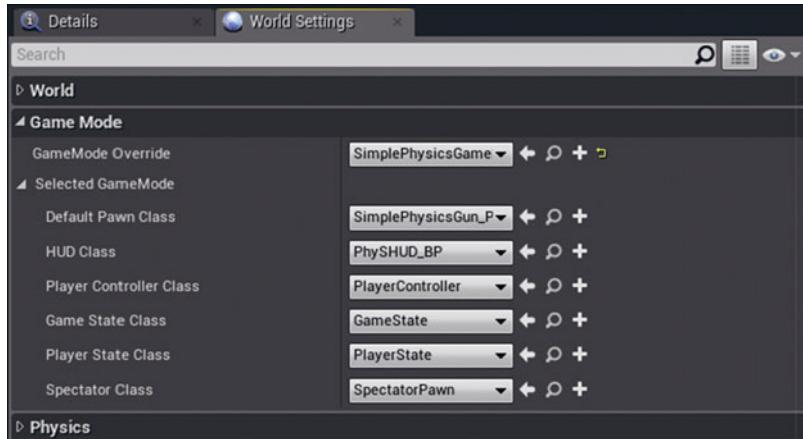


Рис. 13.2. Установка параметра GameMode Override

Настройки проекта и физики мира

Установив режим игры для уровня, вы можете сосредоточиться на настройках физики проекта по умолчанию. Эти настройки находятся в нескольких областях. Во-первых, вы устанавливаете настройки по умолчанию для всего проекта. Чтобы сделать это, выберите пункт **Settings** ⇒ **Project Settings** (относится к рис. 13.1).

Открывшаяся вкладка Project Settings позволяет устанавливать любые атрибуты, относящиеся к физике проекта, но сейчас вам нужны только два из них. Как показано на рис. 13.3, вы можете установить значение гравитации по умолчанию (**Default Gravity**), которое примерно равно значению ускорения, с которым физическое тело падает под воздействием силы тяжести. По умолчанию значение этого свойства равно -980 см по оси Z. Вы также можете установить предельную скорость по умолчанию (**Default Terminal Velocity**), значение которой равно значению максимальной скорости, с которой физическое тело будет способно двигаться. Ее значение по умолчанию равно 4000. Эти настройки применяются ко всем уровням проекта, если вы не измените их на панели **World Settings** какого-либо уровня или в отдельно взятых актерах.

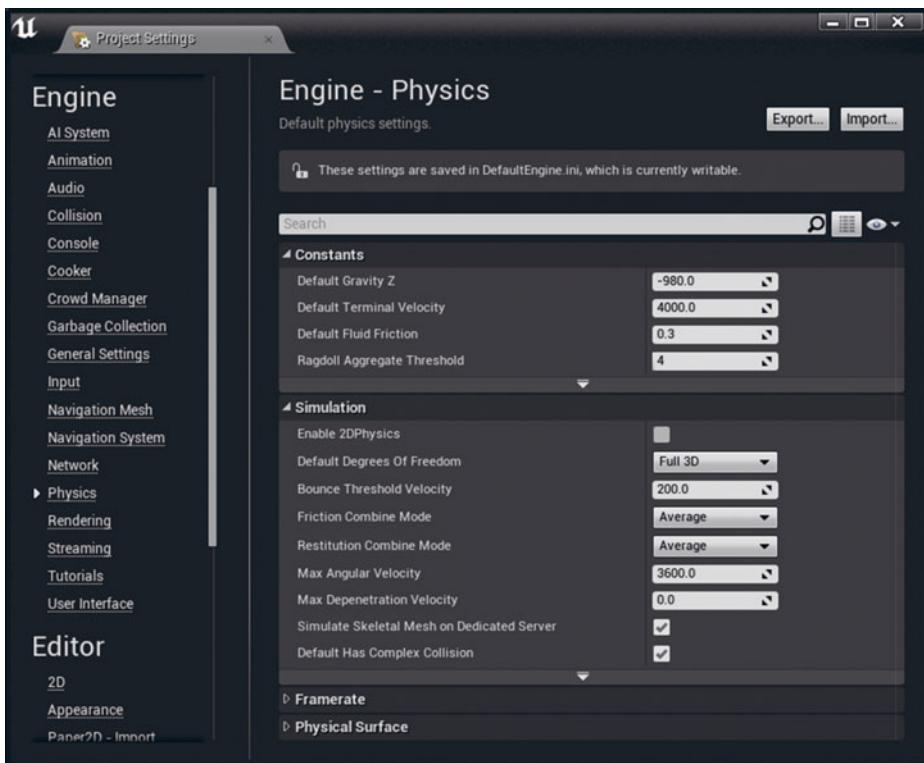


Рис. 13.3. Панель Project Settings

Затем вы можете установить атрибуты физики по умолчанию на определенном уровне. Открыв уровень, выберите вкладку **World Settings** и найдите раздел **Physics**, в котором находятся настройки уровня по умолчанию (см. рис. 13.4). Здесь вы можете переписать настройки гравитации проекта по умолчанию только

для текущего уровня. Вы можете сделать это, например, если работаете над основанной на физике игрой, определенные уровни которой требуют различных настроек гравитации, в отличие от остальной части проекта.

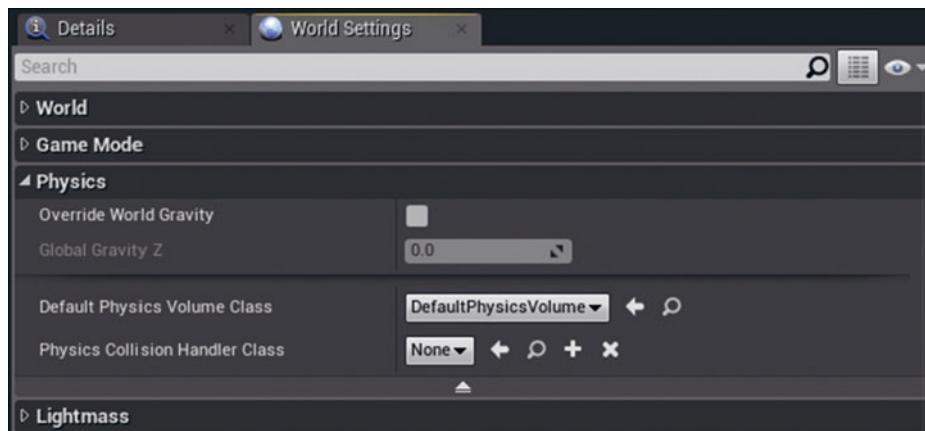


Рис. 13.4. Перезапись настроек гравитации проекта по умолчанию для уровня на панели World Settings

Симуляция физики

Базовое физическое тело в UE4 — это не что иное, как статичный меш, симулирующий физику. Чтобы заставить актер статичного меша симулировать физику, просто поместите его на уровень. Затем на панели **Details** найдите раздел **Physics**, в котором вы сможете установить или снять флажок с параметра **Simulate Physics**, как показано на рис. 13.5. Установленный флажок автоматически меняет мобильность помещенного актера статичного меша на подвижный и меняет пресет коллизии на **Physics Actor**. Если вы запустите уровень, то увидите, что актер статического меша теперь симулирует физику, даже если он не движется, так как уже стоит на земле, и сила к нему не применена. Поместите актер на высоту 500 единиц над землей, что позволит настройкам гравитации по умолчанию повлиять на объект. Также на панели **Details** вы можете увидеть, что актер имеет массу по умолчанию, зависящую от его размеров, задаваемую в килограммах (кг). Если вы отмасштабируете актер, уменьшив или увеличив его, вы увидите, что масса актера меняется в зависимости от его размера. Как можно увидеть, вы можете переписывать эту настройку и вводить любые значения.

Два других свойства, на которые следует обратить внимание, — это **Enable Gravity** и **Start Awake**. Если вы снимете флажок со свойства **Enable Gravity**, вы перепишете настройки гравитации по умолчанию для проекта или уровня для этого

актера, заставляя ее реагировать подобно астероиду в космосе. Снятие флашка **Start Awake** указывает актеру не начинать симулировать физику до тех пор, пока на него не повлияют внешние силы, кроме силы тяжести.



Рис. 13.5. Свойства физики актера статичного меша

СОВЕТ

Нет оболочки коллизии

Если актер статичного меша не позволяет вам заставить его симулировать физику, это, скорее всего, происходит, потому что ассет статичного меша, назначенный актеру, не имеет оболочки коллизии. В таком случае найдите ассет статичного меша на панели **Content Browser**, откройте его в редакторе статичных мешей и присвойте ему оболочку коллизии. Сохраните его после этого.

Заставив актер статичного меша симулировать физику, вы можете запустить уровень и начать взаимодействие с помощью простого физического ружья. Приблизьтесь к физическому телу и наведите на него указатель прицела HUD. Щелкните левой кнопкой мыши, чтобы схватить объект, и подберите его; щелкните правой кнопкой мыши, чтобы отпустить объект. Если вы щелкнете правой кнопкой мыши по объекту, когда не держите ничего в руках, вы ударите по нему.

В табл. 13.1 перечислены ключевые свойства физики, которые вы можете установить для актера статичного меша.

ТАБЛ. 13.1. Свойства физики для актера статичного меша

Свойство	Описание
Simulate Physics	Включает и отключает симуляцию физики для актера
Mass in KG	Масса тела в килограммах, основанная на размере актера в мире. Значение этого свойства можно ввести вручную, включив опцию Override
Linear Damping	Добавление силы сопротивления в целях уменьшения линейного движения
Angular Damping	Добавление силы сопротивления в целях уменьшения углового движения
Enable Gravity	Применять ли к объекту силу тяжести
Constraints	Контролирует, по каким осям актер может двигаться и вращаться при симуляции физики
Modes	Пресеты ограничительных значений
Start Awake	Должен ли объект изначально быть «спящим»
Center of Mass Offset	Указывает компенсацию для центра тяжести этого объекта из расчетного местоположения
Mass Scale	Поэкземплярное масштабирование массы
Max Angular Velocity	Ограничивает величину угловой скорости, которая может применяться

Свойство	Описание
Use Async Scene	Если установлен этот флагок, тело помещается на сцену асинхронной физики. Если флагок сброшен, тело помещается на сцену синхронной физики. Если тело статично, оно помещается на обе сцены, вне зависимости от того, установлен флагок этого свойства или нет
Sleep Family	Набор значений, которые используются для принятия решения, когда помещать тело в «спящее» состояние
Position Solver Iteration Count	Цикл решающего устройства физического тела просчитывает положение. Увеличение значения этого параметра повышает нагрузку на центральный процессор, но приводит к лучшей стабилизации
Velocity Solver Iteration Count	Цикл решающего устройства физического тела просчитывает скорость. Увеличение значения этого параметра повышает нагрузку на центральный процессор, но приводит к лучшей стабилизации

СОВЕТ

Панель Details

Практически каждый подредактор в UE4 имеет свою панель **Details**, и каждая из этих панелей имеет поисковую строку. Если вы не можете найти определенное свойство, введите его название в поисковую строку активной панели **Details**, и оно отобразится на экране.

Использование физических материалов

Физические материалы позволяют изменять поведение физического тела. Термин *физические материалы* (Physical Materials) может быть несколько обманчивым, поскольку эти материалы невидимы, и они, скорее, создают ощущение некоторого вещества. Они могут применяться к отдельным актерам статичных мешей на вашем уровне или быть назначенными обычным материалам. Когда физический материал назначен актеру статичного меша, симулирующему физику, он также влияет на поведение симуляции физики актера.

Создание ассета физического материала

Ассеты физических материалов можно создать на панели Content Browser. Они позволяют устанавливать такие свойства, как Friction, Density и Restitution, для физического тела (см. рис. 13.6).

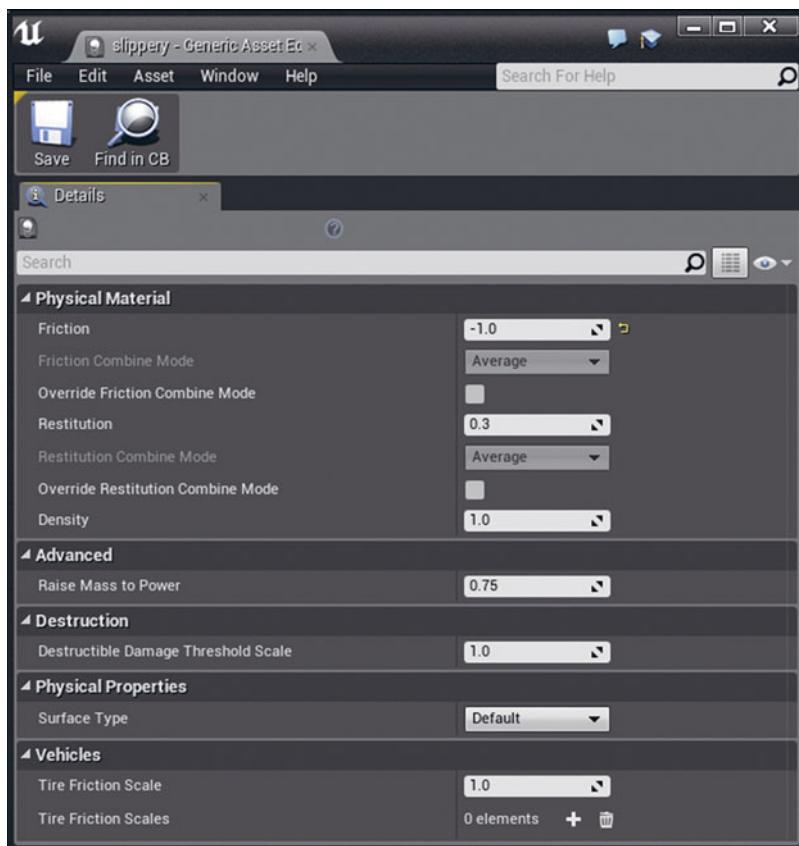


Рис. 13.6. Свойства физического материала

Чтобы создать новый физический материал в редакторе, создайте новую папку на панели **Content Browser**. Щелкните правой кнопкой мыши в области управления ассетами и выберите вариант **Physics** ⇒ **Physical Material** (см. рис. 13.7). Назовите только что созданный ассет и дважды щелкните по нему, чтобы открыть его свойства. Изменив какие-либо значения, не забудьте сохранить изменения, щелкнув правой кнопкой мыши по измененному ассету на панели **Content Browser** и выбрав вариант **Save**.

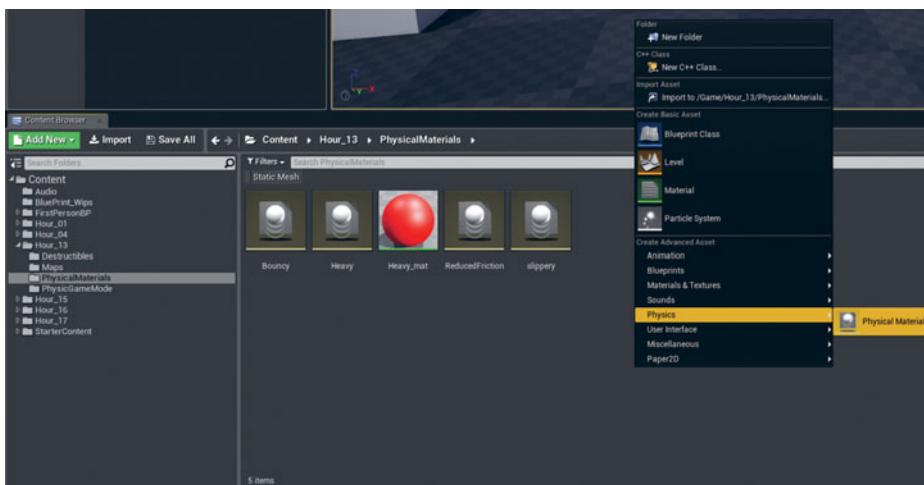


Рис. 13.7. Создание ассета физического материала

Назначение физического материала актеру статичного меша

Чтобы назначить физический материал актеру статичного меша, выберите актер на уровне и найдите раздел **Collision** на панели **Details**. Вы увидите свойство **Phys Material Override** (см. рис. 13.8). Перетащите ассет физического материала из панели **Content Browser** в это свойство, чтобы установить его.

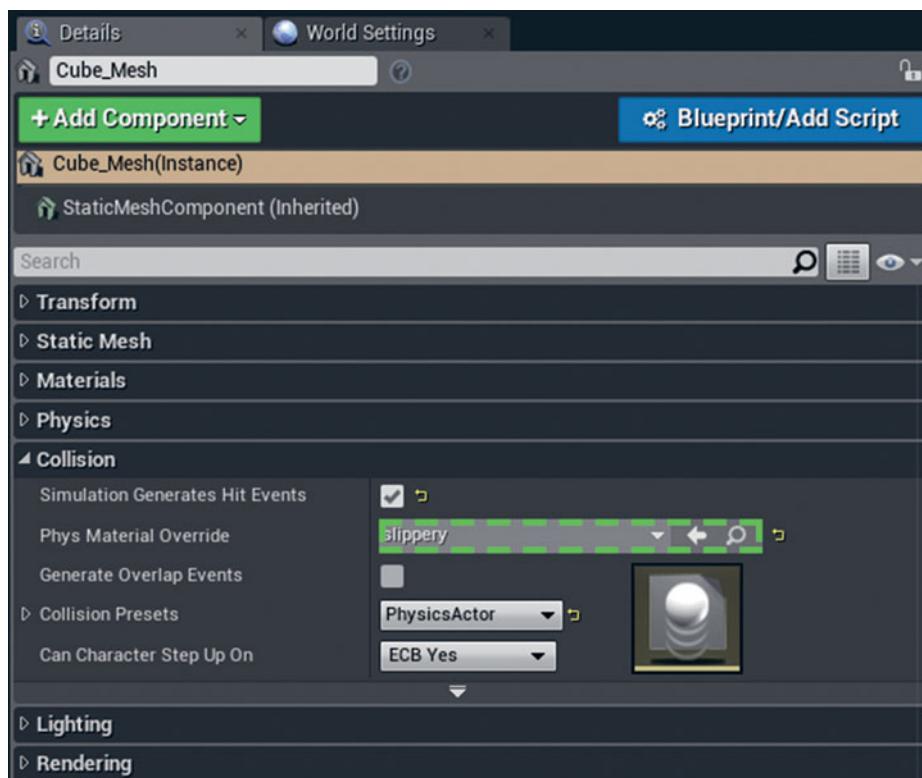


Рис. 13.8. Назначение физического материала ассету статичного меша в разделе **Collision** на панели **Details**

Назначение физического материала другому материалу

Преимущество назначения физического материала обычному материалу заключается в том, что каждый раз, когда вы назначаете материал актеру статического меша, он будет иметь свойства визуальной поверхности обычного материала, а также он будет использовать свойства физического материала, если состояние актера когда-либо менялось на симуляцию физики в процессе игры.

Чтобы назначить физический материал обычному материалу, просто откройте его в редакторе материалов, выберите конечный нод материала и найдите свойство **Phys Material Override** на панели **Details** редактора материалов. Перетащите ассет физического материала в это свойство, сохраните и закройте редактор материалов.

Следующая рубрика «Попробуйте сами» проведет вас через создание ассетов физических материалов и назначение их актерам статичных мешей и обычным материалам.

ПОПРОБУЙТЕ САМИ

Назначение физических материалов актерам статичных мешей

Настало время попрактиковаться в создании и назначении физических материалов актерам статичных мешей, для этого выполните следующие шаги.

- Перетащите три актера статичных мешей — куб и две сферы — из вкладки Place на панели Modes. Установите для них симуляцию физики.
- Создайте папку на панели Content Browser для хранения ваших физических материалов.
- Создайте три физических материала.
- Назовите первый физический материал **Slippery** и задайте его свойству Friction значение **-1**. Назначьте его свойству Phy Material Override куба.
- Назовите второй физический материал **Bouncy** и задайте его свойству Restitution значение **1,6**. Назначьте его свойству Phy Material Override сферы.
- Назовите третий физический материал **Heavy** и задайте его свойству Density значение **10**.
- Создайте новый обычный материал в редакторе материалов и назовите его **Heavy_Mat**. Назначьте ему цвет и соедините выражение Constant3Vector со свойством Base Color.
- В редакторе материалов выберите под материала Primary и на панели Details редактора материалов назначьте ассет физического материала **Heavy**, созданный в шаге 6, свойству Phys Material, как показано на рис. 13.9.
- Сохраните и закройте материал.
- Перетащите материал **Heavy_Mat** в последний актер статичного меша на уровне.
- Запустите уровень и проверьте взаимодействие с каждый физическим телом.

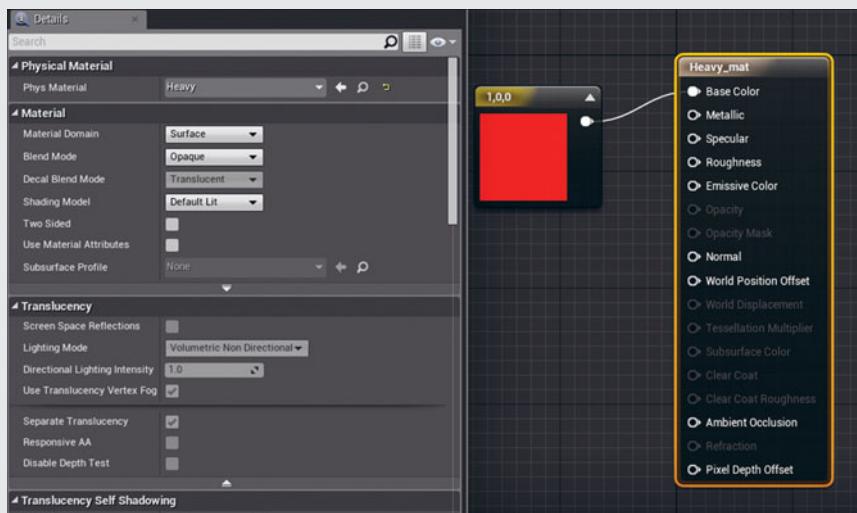


Рис. 13.9. Назначение физического материала другому материалау

Работа с ограничениями

Ограничения позволяют контролировать движение физического тела с помощью блокировки перемещения и поворота по определенным осям. Свойства ограничения устанавливаются в разделе Constraint панели Details уровня для актера статичного меша. В нем вы найдете свойства, позволяющие блокировать физическое тело по определенным осям положения и поворота (см. рис. 13.10). Существует даже свойство режима с пресетами, которое удобно при работе с отдельными физическими телами, имеющими какие-либо требования. Например, вы можете заблокировать движение по осям X и Y, так, чтобы физическое тело могло двигаться только вдоль оси Z; или вы можете заблокировать только вращение, так, чтобы физическое тело могло двигаться, но не могло поворачиваться.

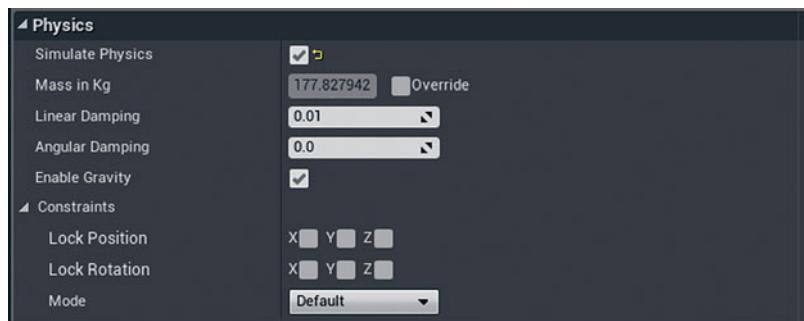


Рис. 13.10. Ограничения актера

Прикрепление актеров физики

В предыдущих часах вы узнали о родительских/дочерних отношениях и прикрепляли актеры друг к другу. К сожалению, в связи с тем, что физические объекты динамично реагируют на окружающий их мир в реальном времени благодаря коллизиям и внешним силам, прикрепление их друг к другу не имеет никакого эффекта. Физическое тело может быть родителем для другого подвижного актера, но прикрепление физического тела в качестве дочернего к родительскому актеру не возымеет эффекта в процессе игры, а их взаимоотношения будут проигнорированы.

ПРИМЕЧАНИЕ

Прикрепление физических тел

Редактор позволяет вам прикреплять физические актеры. Если в настройках статичного меша указана симуляция физики, а дочерний актер является подвижным, то последний будет менять положение и поворот следом за родительским физическим актером.

Актеры ограничения физики

Из-за ограниченности прикрепленных физических тел компания Epic предоставляет актер физического ограничения (Physics Constraint Actor), позволяющий соединять физические актеры с другими актерами, как показано на рис. 13.11. Актеры ограничений используются для создания сочленений или шарниров между двумя физическими телами. Актер ограничения отличается от стандартного метода прикрепления тем, что движение как родителя (актер ограничения 1), так и ребенка (актер ограничения 2) влияет на передвижение и вращение обоих актеров. Поскольку актер ограничения физики работает как сочленение, существуют пресеты типов сочленений. Выбор пресета автоматически настраивает линейные и угловые параметры между значением **Free**, что значит ограничений нет, **Lock**, означающим, что любое движение запрещено, и **Limited**, позволяющим устанавливать определенное движение.

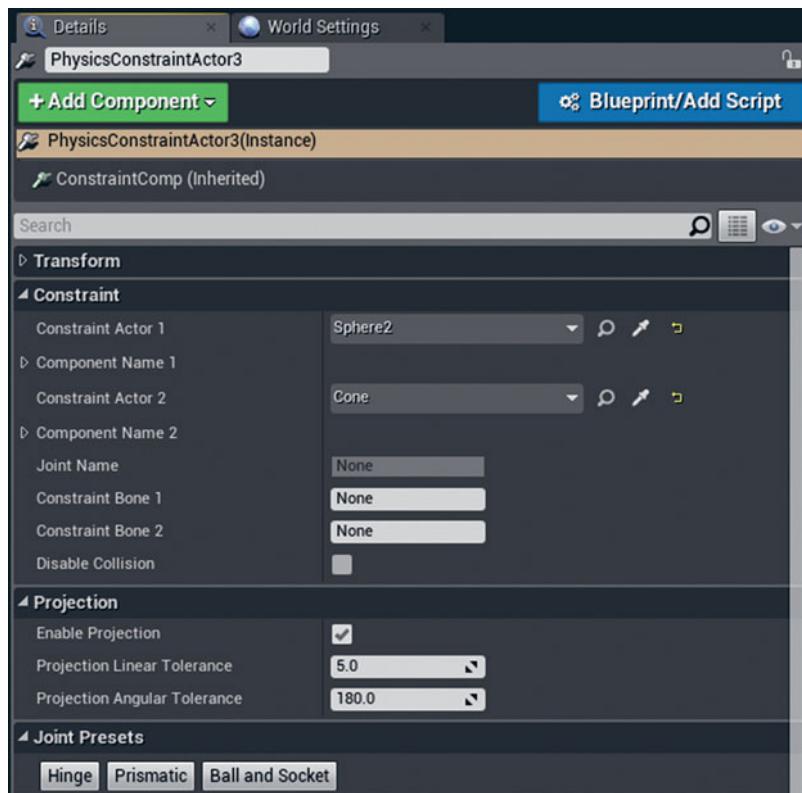


Рис. 13.11. Свойства актеров ограничения физики

ПОПРОБУЙТЕ САМИ**Создайте вращающуюся лампу, используя цепь ограничений**

Выполните следующие шаги, чтобы установить собственную цепь ограничений при создании подвесной лампы.

1. Поместите статичный куб на уровень на высоте 400 единиц над полом. Оставьте его статичным, не симулируйте физику.
2. Поместите статичную сферу прямо под кубом. Установите масштаб по осям X, Y и Z в значении **0,4** и установите флажок для свойства **Simulate Physics**.
3. Поместите вторую статичную сферу прямо под первой. Установите масштаб по осям X, Y и Z в значении **0,4** и установите флажок для свойства **Simulate Physics**.
4. Поместите статичный конус прямо под нижней сферой и установите флажок для свойства **Simulate Physics**.
5. Перетащите актер прожекторного ИС так, чтобы он находился прямо под конусом. Установите для него красный цвет и повысьте его интенсивность до **40000**.
6. Прикрепите прожекторный ИС к актеру конуса с помощью панели **World Outliner**.
7. Перетащите актер ограничения физики и поместите его между кубом и верхней сферой.
8. На панели **Details** для актера ограничения найдите вкладку **Constraint**. Для первого актера ограничения (Constraint Actor 1) щелкните по значку в виде глаза справа и затем щелкните по кубу во выпорте. Так вы назначите статичный куб свойству первого актера ограничения. Если вы сделали это правильно, вы увидите, что куб окружен красным каркасным блоком.
9. Повторите шаг 8 для второго актера ограничения (Constraint Actor 2), но на этот раз выберите верхнюю сферу. Если вы сделаете это правильно, вы увидите, что сфера будет окружена синим каркасным блоком.
10. Повторите шаги с 7 по 9 дважды и каждый раз помещайте новый актер ограничения между следующими двумя актерами статичных мешей и назначайте меш сразу над только что помещенным физическим ограничением для первого актера ограничения и нижний меш для второго актера ограничения. У вас должно быть в общей сложности три актера ограничения физики, когда вы закончите. См. рис. 13.12 для понимания, как правильно разместить актеры.
11. Запустите уровень и используйте физическое ружье, чтобы взаимодействовать с цепью ограничений и заставить источник света вращаться.

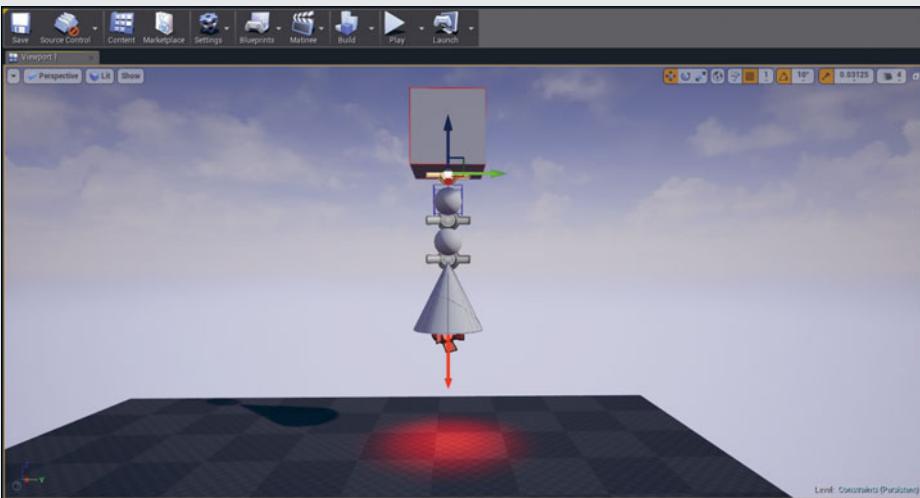


Рис. 13.12. Цепь ограничения лампы

Вместо того чтобы использовать пресеты, вы можете детально контролировать пределы линейного или углового движения в сочленении. Если вы выберете вариант **Limited** для любой из осей в разделе **Angular Limits** на панели **Details**, то увидите больше свойств, позволяющих устанавливать жесткость и торможение, а также пределы углов поворота и изгиба в сочленении, как показано на рис. 13.3. Вы можете даже установить линейные и угловые пределы разрушаемости, которые будут ограничивать разрушаемость сочленения при применении достаточной линейной или угловой силы.

Установив цепь ограничения, позэкспериментируйте с этими свойствами и посмотрите, что произойдет. Вносите небольшие изменения и производите игровое тестирование до тех пор, пока не получите четкое представление о том, для чего предназначены все свойства и как они работают.

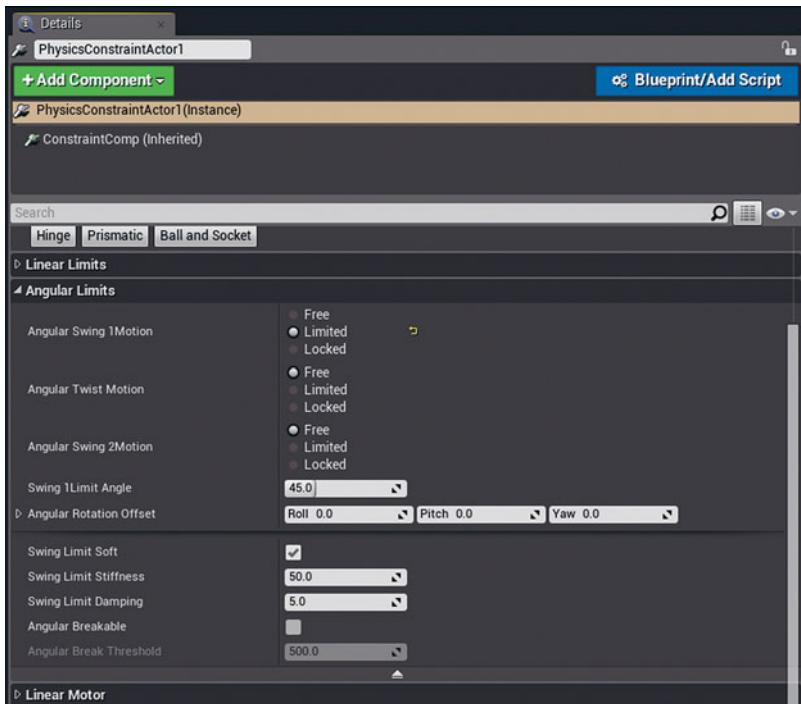


Рис. 13.13. Свойства Angular Limits актера ограничения физики

Установив цепь ограничения, поэкспериментируйте с этими свойствами и посмотрите, что произойдет. Вносите небольшие изменения и производите игровое тестирование до тех пор, пока не получите четкое представление о том, для чего предназначены все свойства и как они работают.

ПРИМЕЧАНИЕ

Симуляция металлических цепей и канатов

Вы можете захотеть использовать описанный метод для создания настоящей цепи, и, несмотря на то что технически это возможно, это не лучший способ создания цепей. Если вы хотите симулировать цепь или даже канат, лучше всего это сделать в редакторе инструмента физических ассетов (PhAT Editor, Physics Asset Tool Editor), используя актер скелетного меша, полагающийся на иерархию костей и сочленений.

Использование актеров силы

При помощи блюпринтов мы можете очень гибко манипулировать физическими телами. Тем не менее компания Epic предоставляет несколько классов для работы, включая актеров физических движителей и радиальной силы.

Актеры физических движителей

Вы можете найти класс Physics Thruster (физический движитель) на вкладке Place панели Modes. Просто введите слово physics в поисковой строке панели Modes и найдите актер Physics Thruster в списке. Чтобы использовать его, поместите его на уровне под актером физики, на который хотите повлиять, и поверните в том направлении, в котором должна воздействовать сила. Затем на панели World Outliner прикрепите его к актеру статичного меша, симулирующему физику. Установите величину силы, которую хотите применить, в разделе Thrust Strength на панели Details уровня. Установите флажок рядом с параметром Auto Activate в разделе Activation панели Details уровня. Величина силы нужна для перемещения физического тела в зависимости от его массы. Поэтому вам может понадобиться высокое значение силы толкания, чтобы появился эффект, в зависимости от того, какое физическое тело выбрано.

СОВЕТ

Контроль массы

Помните, что масштабирование физического тела в сторону увеличения или уменьшения изменяет его массу. Вы также можете указать значение массы статичного меша на панели Details уровня. Выберите актер и в разделе Physics панели Details уровня найдите свойство Mass In Kg. Установите флажок рядом с параметром Override и установите желаемую величину.

ПОПРОБУЙТЕ САМИ

Создайте коническую ракету

Выполните следующие шаги, чтобы создать простую коническую ракету, и используйте прикрепленный актер физического движителя для ее запуска.

1. Перетащите статичный конус из вкладки Place панели Modes. Поместите его на высоту примерно 50–150 единиц над полом.
2. Установите флажок рядом с параметром Simulate Physics для конусовидного актера статичного меша.
3. Перетащите актер движителя из вкладки Place панели Modes.

4. Выделив движитель, установите его местоположение в разделе **Transforms** панели **Details** так, чтобы он имел те же координаты X и Y, что и конус.
5. Поверните актер движителя так, чтобы его желтая стрелка направления указывала прямо вниз.
6. Прикрепите движитель к конусу на панели **World Outliner**.
7. Для актера физического движителя на панели **Details** установите параметр **Thrust Strength** в значение порядка **65000** и установите флажок рядом с параметром **Auto Activate**.
8. Запустите уровень; конус должен взмыть в воздух.
9. Если физическое тело не двигается, убедитесь, что направление движителя правильное. Вы также можете уменьшить массу актера статичного меша или увеличить силу толкания актера физического движителя. Если конус летит неровно, внесите изменения в положение актера движителя или установите ограничения для статичного конуса, заблокировав оси X и Y в свойствах **Physics** актера (см. рис. 13.5).

ПРИМЕЧАНИЕ

Копирование и вставка трансформаций актеров

Вы можете скопировать трансформации актера, щелкнув правой кнопкой мыши по его свойствам **Location**, **Rotation** или **Scale** в разделе **Transforms** на панели **Details** уровня и выбрав вариант **Copy**. Затем вы можете применить их к другому актеру, щелкнув правой кнопкой мыши по соответствующему свойству другого актера и выбрав вариант **Paste**.

Актеры радиальной силы

Актер радиальной силы (Radial Force Actor) применяет силу во всех направлениях из единственной точки воздействия, поэтому его ориентация не имеет значения. Актер радиальной силы влияет только на актеры физики, попадающие в область его воздействия, и вы можете менять размер области воздействия, масштабируя актер. Воздействие имеет значение затухания, поэтому сила, примененная к физическому телу, тем больше, чем ближе оно находится к центру актера радиальной силы. Чтобы поместить актер радиальной силы на сцену, найдите его с помощью поисковой строки на панели **Modes**. Перетащите его на уровень и, выбрав его, установите свойства **Force Strength**.

ПОПРОБУЙТЕ САМИ

Используйте толчок радиальной силы

Выполните следующие шаги, чтобы установить актер радиальной силы так, чтобы он толкал актер, симулирующий физику.

- Перетащите актер статичного меша в виде куба из вкладки **Place** панели **Modes**. Поместите его на высоту 500 единиц над полом.
- Выбрав помещенный статичный меш, установите флашок рядом с параметром **Simulate Physics** на панели **Details** уровня для актера статичного меша и укажите его массу, установив значение **10**.
- Установите значение свойства **Linear Damping** куба в **1**.
- Перетащите актер радиальной силы и поместите его на пол прямо под кубом.
- На панели **Details** для актера радиальной силы установите свойство **Force Strength** в значение порядка **10000**.
- Запустите уровень. Куб должен падать медленно в направлении пола и, ударившись о пол, выталкиваться в сторону.

Резюме

В этом часе была представлена работа с физикой в UE4. Имея только эти базовые знания, вы можете реализовать в проекте множество ваших замыслов. Конечно, изучить можно намного больше. Теперь, когда вы чувствуете себя комфортно при работе с простыми ассетами физики и их свойствами, следующим шагом является изучение работы с разрушаемыми актерами и редактором PhAT (**Physics Asset Tool**), который позволяет назначать физические свойства отдельным костям скелетных мешей. Этот редактор можно использовать для создания множества подвижных объектов от канатов и «тряпичных кукол» до персонажей, по-разному реагирующих на удары. Периодически обращайтесь к демопроекту **Content Examples** от компании Epic. Вы можете загрузить проект на вкладке **Learn** (Обучиться) лаунчера. Просто установите проект, откройте уровень **physics and destructible** и запустите его, чтобы увидеть примеры.

Вопросы и ответы

Вопрос: Я не вижу указатель прицела HUD, когда запускаю уровень. Почему?

Ответ: Убедитесь, что установили значение **SimplePhysicsGameMode** свойства **GameMode Override** для текущего уровня на панели **World Settings**. Вы также можете применить этот параметр настройки ко всему проекту на вкладке **Maps & Modes** панели **Project Settings**.

Вопрос: Почему свойство Simulate Physics затемнено для моего статичного меша на уровне?

Ответ: Убедитесь, что ассет статичного меша имеет оболочку коллизии. Если нет, откройте его в редакторе статичных мешей и назначьте ему оболочку.

Вопрос: Я поместил актер толкателя физики на уровень, но он не воздействует ни на одно созданное мной физическое тело. Почему?

Ответ: Актер физического движителя должен быть прикреплен к статичному мешу, на который вы хотите повлиять. Также для актера толкателя физики должен быть установлен флажок рядом со свойством Auto Activate на панели Details.

Вопрос: Как мне поменять направление силы актера толкателя физики?

Ответ: Просто поверните прикрепленный актер физического движителя, используя виджет трансформации, таким образом, чтобы стрелка была направлена в желаемую сторону воздействия силы.

Вопрос: Когда я использую актер радиальной силы и помещаю физическое тело статичного меша на него, ничего не происходит. Почему?

Ответ: Убедитесь, что установили значение силы, а не импульса, для актера радиальной силы, а также что применили высокое значение силы или низкое значение массы физических тел, на которые хотите повлиять.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Истинно или ложно высказывание: на панели **World Outliner**, если вы прикрепите актер статичного меша, симулирующий физику, к актеру статичного меша, свойство **Mobility** которого установлено в значение **Static**, актер физики будет неподвижным.
2. Истинно или ложно высказывание: твердые тела деформируются при столкновении с другими актерами.
3. Истинно или ложно высказывание: если установить высокое значение для свойства Linear Damping для физического тела, его скорость будет сокращаться с течением времени.
4. Истинно или ложно высказывание: физический материал может не быть материалом, но может быть назначен материалу.

5. Истинно или ложно высказывание: движитель может приводить физическое тело в движение без необходимости прикрепления двух актеров друг к другу.

Ответы

1. Ложь. Поскольку актер статичного меша симулирует физику, его прикрепление к другому актеру не вызывает эффекта. Если вы хотите прикрепить симулирующий физику актер к другому актеру, вам необходимо использовать актер ограничения физики.
2. Ложь. Мягкие тела деформируются при столкновении с другими актерами.
3. Истина. Свойство **Linear Damping** сокращает скорость физического тела с течением времени.
4. Истина. Физические материалы могут назначаться как актерам статичных мешей, так и обычным материалам.
5. Ложь. Чтобы движитель работал, он должен быть прикреплен к родительскому статичному мешу, симулирующему физику.

Упражнение

Множество актеров ограничения могут одновременно воздействовать на один актер статичного меша. Используя ограничение физики и актеры статичных мешей, создайте платформу, подвешенную на четырех раздельных цепях ограничения в каждом углу.

1. Создайте новый уровень и установите значение **SimplePhysicsGameMode** для свойства **GameMode Override** на панели **World Settings**.
2. Создайте такую же цепь ограничения, которую вы создавали для вращающейся лампы в одном из предыдущих упражнений «Попробуйте сами», не добавляя источник света. Добавьте еще один актер ограничения физики вниз цепи и назначьте последний меш свойству **Constraint Actor 1**.
3. Когда цепь будет создана, выберите все актеры, из которых состоит цепь, и продублируйте ее три раза. Расположите каждую копию так, чтобы цепи находились отдельно друг от друга и формировали квадрат. Вы можете выполнить дублирование с помощью виджета трансформации **Move**, удерживая клавишу **Alt** при перемещении выбранных элементов.

4. Добавьте статичный меш в виде блока (box) и отмасштабируйте его так, чтобы он создавал платформу, на которой может стоять игрок. Установите флагок рядом с параметром **Simulate Physics** для этого актера и поместите его под четырьмя цепями ограничений.
5. Назначьте платформу свойству **Constraint Actor 2** для последнего актера ограничения физики в каждой цепи.
6. Запустите уровень и проверьте взаимодействие с платформой при помощи физического ружья или прыгая по ней (см. рис. 13.14).

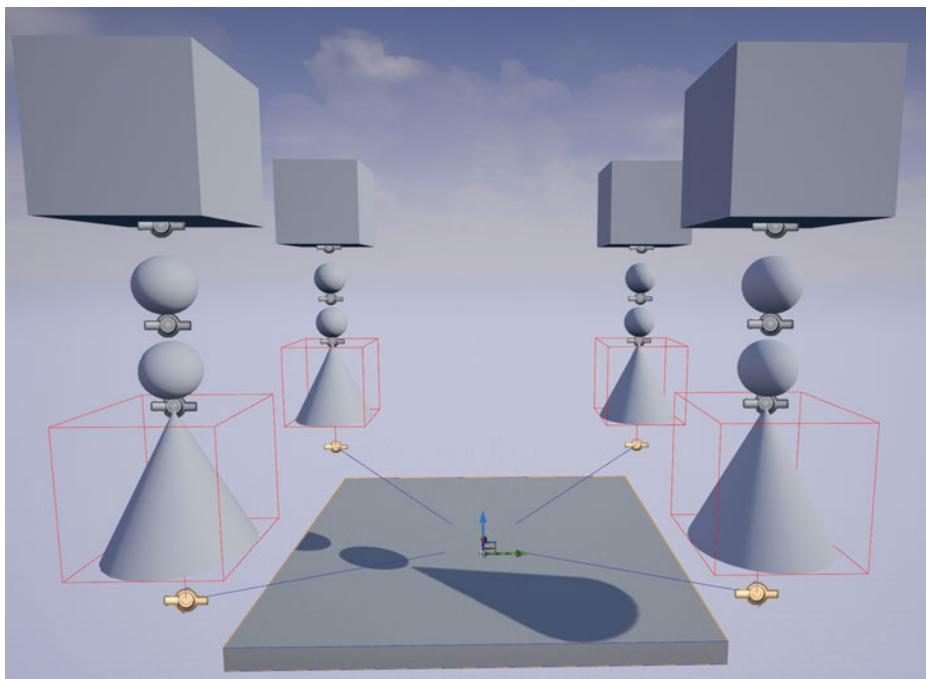


Рис. 13.14. Платформа, подвешенная на четырех отдельных цепях ограничения

14-Й ЧАС

Введение в систему визуального программирования блюпринтов

Что вы узнаете в этом часе

- ▶ Изучение интерфейса редактора блюпринтов
- ▶ Как использовать события, функции и переменные
- ▶ Добавление события
- ▶ Объявление переменной

Практически каждый игровой движок имеет скриптовый язык, позволяющий разработчикам добавлять и изменять функционал их игр. Некоторые движки используют существующую скриптовую среду, например LUA*, другие же создают собственную. UE4 предоставляет два способа создания контента: язык C++ и блюпринт (Blueprint). Система визуального программирования блюпринтов — это мощная и полнофункциональная среда разработки скриптов, работа с которой осуществляется посредством редактора. Она предоставляет художникам и дизайнерам возможность создавать полноценные игры, прототипировать идеи и изменять существующие элементы геймплея. В этом часе представлен редактор блюпринтов и базовые понятия разработки скриптов.

ПРИМЕЧАНИЕ

Подготовка к практике

Для этого часа создайте новый пустой проект без стартового контента.

* Скриптовый язык программирования. — Прим. ред.

Основы визуального программирования

Разработка на языке C++ требует использования интегрированной среды разработки (IDE, Integrated Development Environment), такой как Microsoft Visual Studio, в которой доступна разработка любых программных элементов от новых классов и геймплея до модификаций компонентов ядра движка. Блюпринты, напротив, представляют собой визуальную среду разработки скриптов. Хотя вы не можете использовать блюпринт для написания графического движка, вы можете использовать его и для создания ваших собственных классов и игрового функционала. Визуальная среда программирования, такая как блюпринт, не использует традиционную текстовую среду, но вместо этого предлагает ноды и провода. Ноды (nodes) — это визуальные выражения функций (участков кода, выполняющих определенные операции), переменных (для хранения данных), операторов (выполняющих математические операции) и условий (позволяющих проверять и сравнивать переменные).

ПРИМЕЧАНИЕ

Когда использовать C++

Язык C++ необходимо использовать, когда ваша игра требует 100% эффективности или некоторых модификаций компонентов базового рендеринга, физики, аудио или сетевого движка. Компания Epic предоставляет полный доступ ко всему исходному коду, используемому для создания всех базовых компонентов движка. Некоторые предпочитают использовать текстовую скриптовую среду или среду программирования, такую как C++. Если вы мало знакомы с программированием, работа в визуальной скриптовой среде, такой как блюпринт, — отличный способ изучить базовые концепции программирования, не изучая синтаксис.

Визуальное программирование позволяет художникам и дизайнерам разрабатывать игровой функционал, давая возможность программистам работать над более сложными задачами. Многие игры могут быть созданы полностью с помощью блюпринта, и, поскольку он производит компиляцию на уровне байткода, блюпринт-скрипты весьма эффективны. Вы можете использовать блюпринт для создания полноценных игр для всех платформ, поддерживаемых UE4.

ПРИМЕЧАНИЕ

Компиляция блюпринт-скриптов

Хотя блюпринт — это визуальная среда, блюпринты все же нуждаются в компиляции. Блюпринты компилируются до уровня байткода. Важно понимать следующие термины.

- ▶ **Компилятор.** Программное обеспечение, используемое для компилирования инструкций (исходного кода), написанных на языке программирования.

- ▶ **Компиляция.** Процесс преобразования инструкций в машинный язык (код), который может быть выполнен центральным процессором. Технические требования для выполнения компиляции варьируются в зависимости от аппаратного обеспечения и операционной системы.
 - ▶ **Байткод.** Скомпилированный исходный код, который обрабатывается виртуальной машиной, а не аппаратными средствами. Это означает, что исходный код может быть один раз скомпилирован и запущен на любом аппаратном обеспечении, имеющем виртуальную машину для обработки байткода.
 - ▶ **Виртуальная машина.** Программное обеспечение, транслирующее байткод в инструкции, которое аппаратные средства могут понять и обработать.
-

Изучение редактора блупринтов

Визуальная среда разработки блупринтов — ключевой компонент редактора UE4, и даже в основанных на C++ проектах вы, скорее всего, до некоторой степени будете использовать блупринты. В UE4 существует пять типов блупринтов.

- ▶ **Блюпринт уровня** (Level Blueprint). Этот блупринт используется для управления глобальными событиями на уровне. На одном уровне может быть только один блупринт уровня, и он автоматически сохраняется при сохранении уровня.
- ▶ **Блюпринт-класс** (Blueprint class). Это производный класс от другого существующего класса, созданного с помощью C++ или другого блупринт-класса. Он используется для кодирования функционала актеров, помещенных на уровень.
- ▶ **Data-Only блупринт** (Data-Only Blueprint). Этот блупринт хранит только измененные свойства унаследованного блупринта.
- ▶ **Блюпринт-интерфейс** (Blueprint Interface, BPI). Блюпринт-интерфейсы используются для хранения коллекции определенных пользователем функций, которые могут быть назначены другим блупринтам. Блюпринт-интерфейсы позволяют другим блупринтам обмениваться данными друг с другом.
- ▶ **Блюпринт-макросы** (Blueprint Macros). Блюпринт-макросы — независимые графы часто используемых последовательностей нодов, которые могут многократно использоваться другими блупринтами. Блюпринт-макросы хранятся в библиотеке макро-блупринтов (Blueprint Macro Library).

Блюпринты уровней и блупринт-классы — два наиболее часто используемых типа блупринтов. В этом часе вы сосредоточитесь на знакомстве с редактором блупринтов. В последующих часах вы узнаете больше о работе с блупринт-классами.

ПРИМЕЧАНИЕ

Работа с блюпринтами

Ниже приведены некоторые базовые понятия, необходимые при разговоре о блюпринтах и программировании в целом.

- ▶ **Блюпринт.** Ассет блюпринт-класса, хранящийся на панели **Content Browser**.
- ▶ **Актер блюпринта.** Экземпляр ассета блюпринт-класса, помещенный на уровень.
- ▶ **Объект.** Переменная или коллекция переменных, например структуры данных или функции, хранящиеся в памяти.
- ▶ **Класс.** Шаблон для создания объектов, который хранит исходные значения, присвоенные переменным, а также функции и операции для работы с этими объектами.
- ▶ **Синтаксис.** В традиционном программировании и скриптовых средах синтаксис означает орфографическую и грамматическую структуру, которая ожидается компилятором языка, чтобы он смог корректно перевести код в машинный язык.

Интерфейс редактора блюпринтов

Чтобы открыть блюпринт уровня и увидеть интерфейс редактора (**Blueprint Editor**) выберите на панели инструментов редактора уровней пункт **Blueprints ⇒ Open Level Blueprint** (см. рис. 14.1). Интерфейс редактора блюпринтов и рабочий процесс просты для запоминания, но процессом разработки овладеть достаточно сложно. Несмотря на то что сейчас вам не нужно беспокоиться о синтаксисе в визуальной среде программирования, вам все же необходимо работать с логикой и порядком операций. Это требует практики в любой среде программирования.

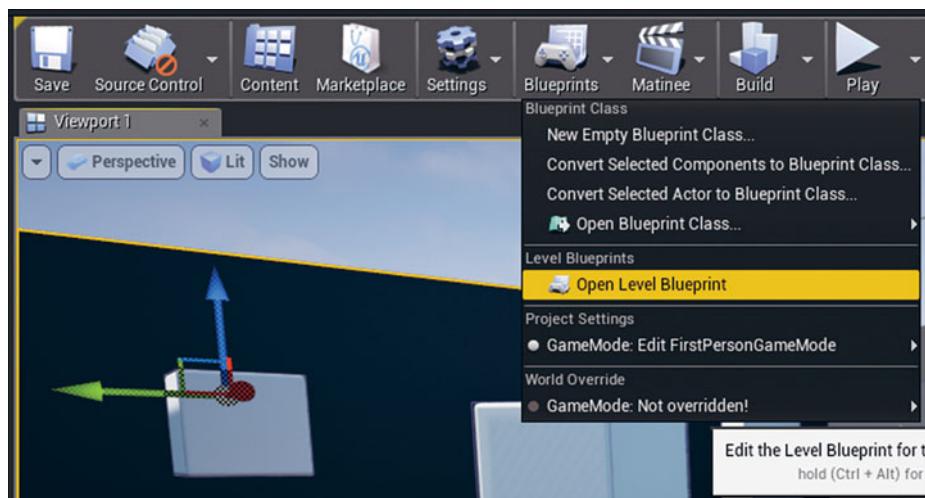


Рис. 14.1. Открытие блюпринта уровня в редакторе блюпринтов для текущего уровня

Интерфейс редактора блупринтов содержит строку меню, панель инструментов для быстрого доступа к основным инструментам и операциям, панель **Event Graph** для проектирования скриптов, панель **Details** для отображения свойств выбранных в редакторе блупримта объектов и панель **My Blueprint**, которая используется для управления и отслеживания графов нодов, функций, макросов и переменных, используемых в выбранном блупримте. Возможности редактора блупринтов выделены на рис. 14.2 и описаны ниже.

- Панель инструментов.** Панель инструментов содержит кнопки (с кратким описанием) для управления редактором блупринтов.
- Панель My Blueprint.** Используется для управления графиками, функциями, макросами и переменными, которые содержатся в вашем блупримте.
- Панель Details.** Вы можете управлять свойствами добавленных в блупримт компонентов, переменных и функций с помощью панели **Details**.
- Панель Event Graph (граф событий).** Эта панель используется для программирования базового функционала блупримта.

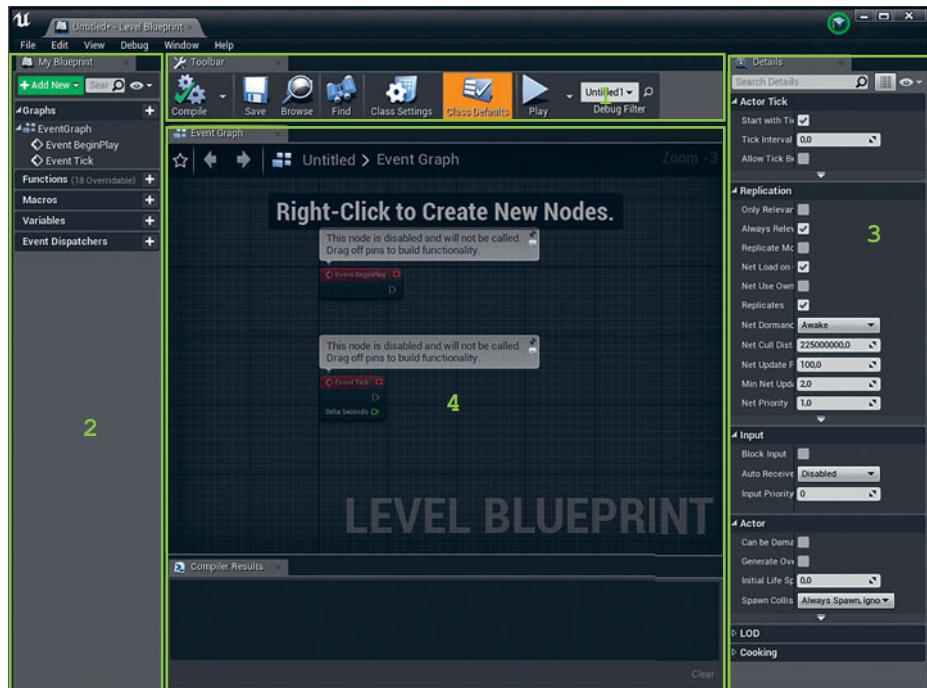


Рис. 14.2. Интерфейс редактора блупринтов

ПРИМЕЧАНИЕ

Панель инструментов редактора блюпринтов

При работе с блюпринтами уровней, панель инструментов редактора блюпринтов не имеет функции сохранения или поиска на панели **Content Browser**, поскольку блюпринт уровня связан с уровнем. Чтобы сохранить блюпринт, просто сохраните уровень.

Панель инструментов редактора блюпринтов

Панель инструментов редактора блюпринтов включает в себя только пять инструментов. Два инструмента сосредоточены на кнопках **Compile** и **Play**. Вы нажимаете кнопку **Compile**, чтобы скомпилировать скрипт и увидеть проблемы в окне результатов компиляции внизу панели **Event Graph**. Кнопка **Play** в этом редакторе аналогична кнопке **Play** в редакторе уровней для запуска уровня. Обратите внимание, что на панели инструментов нет кнопки **Save**. Это связано с тем, что блюпринт уровня связан с уровнем, поэтому, если вы хотите сохранить блюпринт, просто сохраните уровень.

Панель инструментов редактора блюпринтов имеет следующие кнопки для управления блюпринтами.

- ▶ **Compile.** Компилирует блюпринт.
- ▶ **Search.** Открывает панель **Find Results** с поисковой строкой для поиска нодов в блюпринте.
- ▶ **Class Settings.** Показывает опции блюпринта на панели **Details**.
- ▶ **Class Defaults.** Отображает свойства блюпринта на панели **Details**.
- ▶ **Play.** Запускает уровень.

Панель My Blueprint

Панель **My Blueprint** отслеживает все графы нодов, функции, макросы и переменные, которые используются в блюпринте. Каждая категория разделена заголовком, и справа от каждого заголовка есть символ +, по которому можно щелкнуть, чтобы добавить элемент при необходимости. Вы можете использовать панель **My Blueprint**, чтобы добавить, переименовать или удалить все эти элементы.

Панель Event Graph

Панель **Event Graph** — это граф нодов по умолчанию, который используется для кодирования блюпринтов. **Event Graph** — это та область, в которой производится значительная часть работы при использовании редактора блюпринтов. Вы можете

добавлять больше графов нодов в существующий блюпринт при необходимости. Эта панель похожа на листок бумаги в клетку. Вы можете добавлять в блюпринт столько графов, сколько потребуется для сохранения организованности. В табл. 14.1 указаны комбинации клавиш, используемые при работе с нодами в **Event Graph**.

ТАБЛ. 14.1. Комбинации клавиш редактора блюпринтов

Комбинация клавиш	Команда или действие
ПКМ по пустому пространству	Открывает контекстное меню блюпринта
ПКМ + перетаскивание по пустому пространству	Перемещает Event Graph
ПКМ по ноду	Отображает действия с нодом и контактами
ЛКМ по ноду	Выбирает нод
ЛКМ + перетаскивание нода по пустому пространству	Перемещает нод
ЛКМ + перетаскивание по пустому пространству	Выбирает область
Ctrl+ЛКМ	Добавляет или удаляет текущий выбранный нод из группы выбранных нодов
Прокручивание колеса мыши	Приближает и отдаляет Event Graph
Home	Выравнивает Event Graph по центру экрана
Delete	Удаляет выбранные ноды
Ctrl+X	Вырезает выбранные ноды
Ctrl+C	Копирует выбранные ноды
Ctrl+V	Вставляет выбранные ноды
Ctrl+W	Копирует и вставляет выбранные ноды

Контекстное меню блюпринта

Контекстное меню блюпринта (Blueprint Context Menu) — это одно из наиболее часто используемых меню при работе в редакторе блюпринтов. Если вы щелкните правой кнопкой мыши по пустой области или перетащите контакт, откроется контекстное меню блюпринта (см. рис. 14.3), которое позволяет добавлять в граф события, функции, переменные и условия. Это меню по умолчанию чувствительно к контексту, отчего оно отображает только действия, которые соответствуют выбранному элементу или элементу, от которого вы протягиваете контакт.



Рис. 14.3. Контекстное меню блюпринта

Ноды, провода и контакты

Можно представить процесс визуального программирования как работу электрической цепи. Красный нод события отправляет сигнал, идущий по проводам и запускающий выполнение любого нода, через который этот сигнал проходит. Когда нод получает сигнал, он извлекает данные, которые ему требуются в контактах данных в левой части нода. После чего нод выполняет свою операцию, отправляет сигнал дальше и возвращает результаты через контакты данных в правой части нода. Ниже приведена информация, которую нужно знать об этом процессе.

- ▶ Ноды — это визуальные представления событий, функций и переменных. Они имеют цветовой код, выражающий их назначение. Красный нод — это нод события, используемый для инициирования выполнения последовательности нодов. Синие ноды — это функции для выполнения определенных операций. Цветные овальные ноды, каждый из которых имеет только по одному контакту данных, представляют переменные.
- ▶ Контакты выполнения (ехес-контакты) «ввода» и «вывода» представляют собой белые треугольники, один из углов которых направлен вправо, они находятся в верхней части нода и указывают направление последовательности. Красный нод события имеет только один ехес-контакт «вывода»,

поскольку используется для инициирования последовательности, в то время как синие ноды имеют как ехес-контакты «ввода», так и «вывода» (в большинстве случаев), чтобы пропускать сигнал через себя.

- ▶ Контакты данных имеют цветовой код, основывающийся на типе используемых ими данных. Контакты данных в левой части нода извлекают данные, в то время как контакты данных в правой части нода возвращают данные.
- ▶ Провода соединяют ноды. Белые провода соединяют ехес-контакты «ввода» и «вывода», а цветные соединяют контакты данных. Цвет каждого провода отражает тип используемых им данных.

Чтобы установить соединение между контактом выполнения или контактом данных и проводом, щелкните по контакту и перетащите его в другой контакт того же типа. Чтобы оборвать провод, идущий в контакт или из контакта, зажмите клавишу **Alt** и щелкните по контакту. Зажмите клавишу **Ctrl**, затем щелкните по контакту или проводу, чтобы перетащить его в новый контакт.

Фундаментальные понятия разработки скриптов

Все среды программирования используют события, функции, переменные и операторы условий. Следующие страницы познакомят вас с этими базовыми понятиями.

События

Блупринты в UE4 основаны на событиях. *Событие* (event) — это то, что происходит в процессе игры, от нажатия игроком клавиши на клавиатуре или попадания аватара в какое-либо помещение до столкновения актера с другим актером или начала игры. Большинство событий попадает под основные категории, описанные в табл. 14.2. События используются для инициирования последовательности в блупринте. При запуске события из вывода ехес-контакта посыпается сигнал, который проходит по проводам и обрабатывается всеми функциями, встреченными им на пути. Когда сигнал доходит до конца последовательности нодов, он пропадает.

Некоторые события требуют назначения им определенных актеров или компонентов, таких как события коллизий (Collision Events) — как правило, в одно и то же время транслируется более одного события. Например, если у вас на уровне есть актеры Box Trigger и Sphere Trigger и необходимо, чтобы каждый из них реагировал, когда какой-либо актер перекрывает хотя бы один из них, вам необходимо назначить каждому из них событие коллизии OnActorBeginOverlap. Готовность назначать события определенным актерам позволяет вам программировать

реакцию каждого актера. Чтобы назначить актер на уровне события коллизии, выберите актер и затем щелкните правой кнопкой мыши по пустому пространству на панели **Event Graph** блюпринта уровня. Затем в поисковой строке контекстного меню блюпринта введите словосочетание **on Actor begin** и выберите вариант **OnActorBeginOverlap** из списка, чтобы поместить нод событий. Когда нод события коллизии помещен, вы увидите название актера, присвоенное ноду события, так вы узнаете, что актер был назначен ноду. Теперь при наложении оболочки коллизии актера будет запускаться этот нод события коллизии.

ТАБЛ. 14.2. Основные события

Название события	Описание события
BeginOverlap	Срабатывает, когда оболочки коллизий двух актеров перекрываются. (Назначается актеру или компоненту.)
EndOverlap	Срабатывает, когда оболочки коллизий двух актеров перестают перекрываться. (Назначается актеру или компоненту.)
Hit	Срабатывает, когда оболочки коллизий двух актеров соприкасаются, но не перекрываются. (Назначается актеру или компоненту.)
BeginPlay	Срабатывает каждый раз, когда уровень загружается в память и запускается
EndPlay	Срабатывает, когда уровень закрывается
Destroyed	Срабатывает, когда актер удаляется из памяти
Tick	Срабатывает при каждом цикле центрального процессора
Custom	Работает так, как определил пользователь в зависимости от определенных потребностей

ПРИМЕЧАНИЕ

Компоненты

Компоненты в UE4 — это подобъектные элементы, находящиеся в блюпринт-классах. Они раскрываются в 16-м часе «Работа с блюпринт-классами».

Редактор блюпринтов предоставляет предопределенные ноды событий, но вы можете также создать собственные пользовательские события, которые могут быть вызваны в любой точке последовательности блюпринта. Создание пользовательского события позволяет определять название события и любые проходящие при вызове события данные. На рис. 14.4 показаны два существующих события (**EventBeginPlay**

и OnActorBeginOverlap), которые вызывают пользовательское событие и передают переменную строкового типа. Пользовательское событие получает сигнал и использует функцию Print String для отображения строковых данных на экране. Пользовательские события могут помочь вам управлять блюпринтами и их хранением.

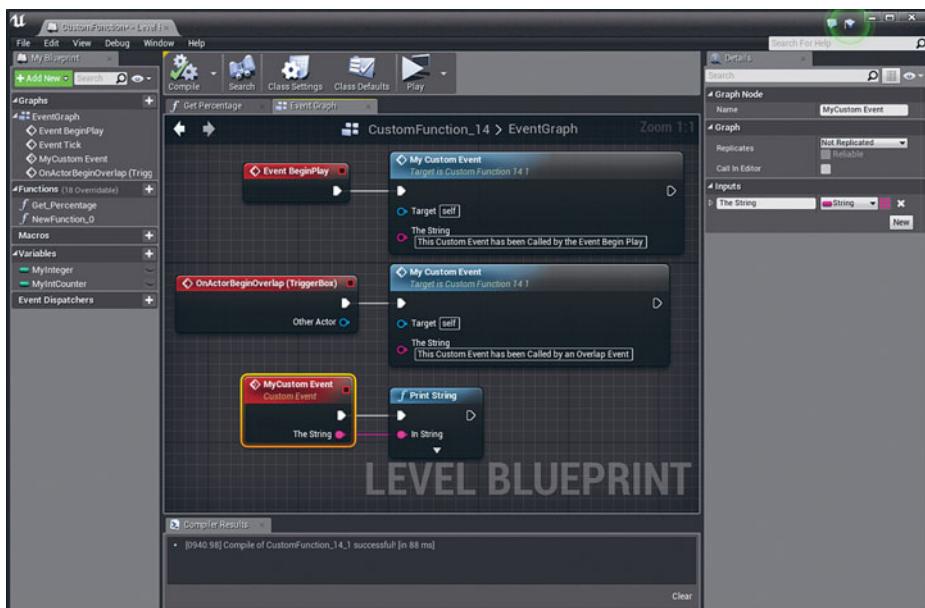


Рис. 14.4. Пользовательские события блюпринта

Чтобы создать пользовательское событие, щелкните правой кнопкой мыши по панели **Event Graph** и в поисковой строке контекстного меню блюпринта напечатайте слово **custom**. Затем выберите вариант **Custom Event** из списка и поместите под пользовательского события. Переименуйте событие, щелкнув по его названию по умолчанию. Чтобы назначить событию переменную, выберите под и на панели **Details** добавьте переменную. Создав пользовательское событие, вы можете вызвать его из другой последовательности, открыв контекстное меню и напечатав в поисковой строке название, присвоенное вами событию. Выберите пользовательское событие из списка, чтобы поместить под, и затем вы можете соединить его проводами с последовательностью.

Функции

Функция — это фрагмент кода, выполняющий определенные операции. Она получает данные, хранящиеся в переменных, обрабатывает информацию и, если это требуется, возвращает результат. Редактор блюпринтов имеет полный набор

предопределенных функций, аналогичных таковым в любой другой среде программирования. Когда функция помещена на панель **Event Graph**, вы, как правило, можете увидеть целевой контакт данных в левой части нода функции. В блюпринте цель обычно представляет собой переменную, хранящую ссылку на актер или компонент актера на уровне, на котором будут выполняться операции этой функции. В примере функции, показанном на рис. 14.5, вы можете увидеть функцию **SetActorLocation**, которая используется для смены местоположения актера на уровне.

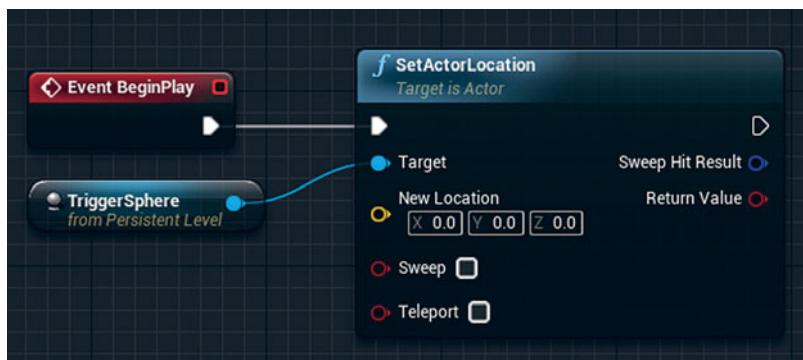


Рис. 14.5. Функция блюпринта

Несмотря на то что блюпринт имеет обширный список готовых к работе функций, вы также можете создавать собственные пользовательские функции в редакторе блюпринтов для отдельных блюпринтов, или даже создать свою библиотеку функций блюпринтов, которая позволит создавать коллекции функций, пригодных для многократного использования в любом блюпринте проекта. На рис. 14.6 показан пример пользовательской функции, созданной в редакторе блюпринтов. Эта пользовательская функция, которая называется **Get Percentage**, получает две переменных с плавающей запятой (A и B), где A — это общее значение, а B — текущее значение. Функция делит текущее значение (B) на общее значение (A) и умножает результат на 100, после чего возвращает итог в процентах в виде числа с плавающей запятой.

Создав пользовательскую функцию, вы можете использовать ее столько раз, сколько захотите, просто перетащив ее из панели **My Blueprint** в граф нодов. Чтобы создать пользовательскую функцию в блюпринте, просто щелкните по символу + в строке Functions на панели **My Blueprint**. Таким образом, вы получите граф нодов, определенный для вашей функции. В графе нодов вы можете увидеть два фиолетовых нода для присвоения переменных ввода и вывода. Вы можете определить переменную ввода/вывода для функции, выбрав ноды ввода или вывода в пользовательском графике функции и на панели **Details**,

при необходимости вы можете создать переменные с различными типами данных. Создав переменные ввода и вывода для функции, вы можете заскриптовать последовательность, как вы сделали бы в любом другом графе нодов, но вы должны будете соединить последовательность проводами с нодами ввода и вывода, когда закончите.

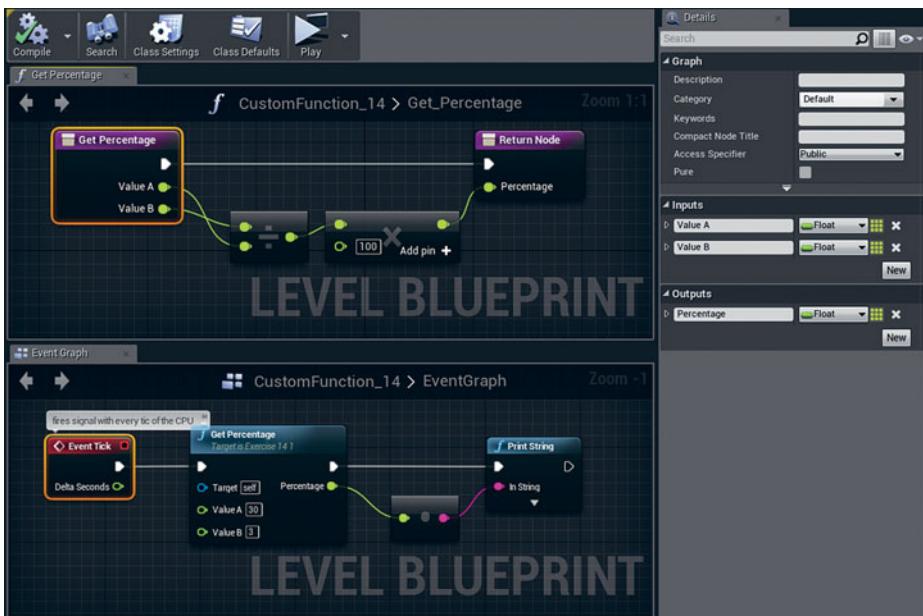


Рис. 14.6. Пользовательская функция блюпринта

ПРИМЕЧАНИЕ

Пользовательские функции

Вы также можете создать пользовательскую функцию, выбрав последовательность уже помещенных нодов и щелкнув правой кнопкой мыши по одному из нодов и выбрав вариант **Collapse to Function** из открывшегося меню. Так у вас появится новая функция, которую можно переименовать. Если вы обычно повторяете последовательность из трех или более предопределенных функций в блюпринте, весьма вероятно, что вы извлечете пользу из «схлопывания» последовательности в пользовательскую функцию.

ПОПРОБУЙТЕ САМИ**Добавьте событие**

Выполните следующие шаги, чтобы добавить событие **Event BeginPlay** и используйте функцию **Print String**, чтобы вывести текст на экран.

1. В меню редактора уровней выберите пункт **File ⇒ New**, чтобы создать новый уровень по умолчанию.
2. На панели инструментов редактора уровней выберите пункт **Blueprints ⇒ Open Level Blueprint**.
3. Выберите события **BeginPlay** и **Event Tick**, которые уже добавлены, и нажмите клавишу **Delete**.
4. Щелкните правой кнопкой мыши по панели **Event Graph** и выберите вариант **Event BeginPlay**. (Используйте поисковую строку, если затрудняетесь найти его.)
5. Щелкните по эхес-контакту в событии **Event BeginPlay**, перетащите его вправо и отпустите.
6. Щелкните правой кнопкой мыши по панели **Event Graph** и в поисковой строке контекстного меню напечатайте словосочетание **print string**.
7. Выберите функцию **Print String**, чтобы поместить для нее нод.
8. Справа от контакта данных **String**, в строке ввода с уже имеющейся надписью **Hello**, напечатайте **Hello level**.
9. Щелкните по клавише **Compile** на панели инструментов редактора блюпринтов и запустите уровень.
10. Каждый раз при запуске уровня вы будете видеть надпись **Hello level** в левом верхнем углу вышпорта уровня в течение нескольких секунд, после чего надпись будет исчезать.

ПРИМЕЧАНИЕ**Функция Print String**

Использование функции **Print String** — не самый подходящий способ коммуникации с игроками. Эта функция обычно используется при разработке в качестве инструмента отладки для коммуникации с происходящим в блюпринте. Если вы хотите отправлять сообщения игрокам, вам необходимо использовать класс **Blueprint HUD** или научиться пользоваться редактором **Unreal Motion Graphics**. См. 22-й час, «Работа с UMG».

Переменные

Переменные хранят данные различных типов. Когда переменная объявлена (создана), компьютер резервирует определенный объем памяти в зависимости от ее типа данных. Эта память далее используется для хранения или извлечения

информации из этой ячейки памяти. Различные типы переменных используют различный объем памяти. Некоторые переменные хранят маленький объем информации, такой как бит, а некоторые хранят значительные объемы, такие как целый актер. В редакторе блупринтов переменные имеют цветовой код, чтобы вы могли быстро определить, какой тип переменной вам потребуется при работе с функциями.

В табл. 14.3 перечислено большинство часто используемых типов переменных, их цветовой код и тип данных, хранящихся в них.

ТАБЛ. 14.3. Основные типы данных

Тип переменной	Цвет	Описание
Boolean или bool (логический или булев)	красный	Хранит значение 0 («выключено» или ложь) или 1 («включено» или истина)
Integer или int (целочисленный)	циановый	Хранит любое круглое число, такое как 1 , 0 , -100 или 376
Float (число с плавающей запятой)	зеленый	Хранит любое значение с десятичной дробью, такое как 1,0 , -64,12 или 3,14159
String (строковый)	маджента	Хранит текст
Vector (векторный)	золотой	Хранит три числа с плавающей запятой — X, Y и Z, например, 100,5 , 32,90 , 100,0
Rotator (ротатор)	пурпурный	Является вектором, хранящим три числа с плавающей запятой, где X — тангаж, Y — рыскание и Z — крен
Transform (трансформация)	оранжевый	Представляет собой структуру, хранящую вектор для определения местоположения, ротатор для определения ориентации и вектор для определения масштаба
Object (объект)	синий	Представляет собой актер на уровне и хранит все свойства в памяти

ПРИМЕЧАНИЕ

Что подразумевается под словом «структура»?

Структура представляет собой коллекцию переменных любого типа, являющуюся единой переменной. Переменные Vector и Rotator технически являются структурами, поскольку хранят в себе три отдельных переменных типа Float. Вы можете создавать собственные структуры в UE4, но вам потребуется изучить эту сложную тему, чтобы уверенно пользоваться редактором блупринтов.

Чтобы объявить переменную, щелкните по символу + в графе **Variables** на панели **My Blueprint** и дайте новой переменной имя. Затем на панели **Details** вы можете установить тип переменной и ее значение по умолчанию. Чтобы установить значение по умолчанию, вам необходимо один раз скомпилировать блюпринт сразу после объявления переменной. После того как вы объявили переменную, дали ей имя и присвоили значение, наиболее часто выполняемой операцией является присвоение и получение данных для переменной. Операция получения (Get) извлекает значение, хранящееся в переменной, а операция присвоения (Set) сохраняет значения. На рис. 14.7 показаны ноды Get и Set для основных типов переменных.



Рис. 14.7. Ноды переменных Get и Set

ПРИМЕЧАНИЕ

Списки переменных

Переменная любого типа в блюпринте может хранить одно значение или массив значений. Переменная, конвертированная в массив, хранит список данных ее типа. Вы можете использовать группу функций для управления массивами переменных — присваивать, получать, удалять или добавлять элементы массива.

ПОПРОБУЙТЕ САМИ

Объявите переменные

Используя блюпринт уровня из предыдущей рубрики «Попробуйте сами», выполните следующие шаги, чтобы объявить целочисленную переменную, задать ей имя и исходное значение.

1. На панели **My Blueprint** щелкните по символу + в графе **Variables**, чтобы добавить новую переменную. Назовите ее **MyInteger**. На панели **Details** установите тип переменной **integer**.
2. Чтобы установить значение по умолчанию новой переменной **MyInteger**, щелкните по кнопке **Compile** на панели инструментов редактора блюпринтов. Затем на панели **Details** в разделе **Default Value** установите исходное значение переменной **100**.
3. Чтобы добавить событие **Event Tick**, щелкните правой кнопкой мыши по панели **Event Graph** и в контекстном меню блюпринта выберите вариант **Event Tick**, после чего появится нод события.
4. Нажмите комбинацию клавиш **Ctrl+W**, чтобы копировать и вставить функцию **Print String** из предыдущего упражнения «Попробуйте сами» и соедините ее с ejecс-контактом вывода **Event Tick**.
5. На панели **My Blueprint** в графе **Variables** щелкните по переменной **MyInteger** и перетащите ее на панель **Event Graph**. Отпустите кнопку мыши, после чего появится диалоговое окно, предлагающее присвоить переменной значение или получить его. Выберите вариант **Get**, чтобы поместить переменную в граф.
6. Щелкните по контакту данных целочисленной переменной и перетащите его в контакт строковых данных функции **Print String**. Редактор автоматически добавит нод преобразования, чтобы конвертировать целочисленную переменную в строковую. По окончании выполнения задания ваш блюпринт уровня должен выглядеть, как показано на рис. 14.8.
7. Скомпилируйте скрипт и запустите уровень. Вы увидите, что целочисленное значение по умолчанию, повторяясь, отображается в левой части выюпорта уровня.

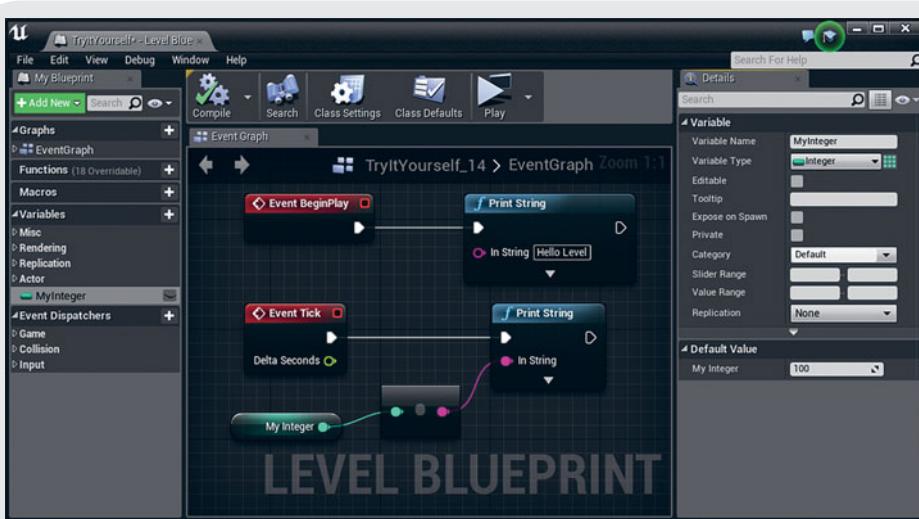


Рис. 14.8. События Event BeginPlay и Event Tick, добавленные на панель Event Graph блюпринта

ПРИМЕЧАНИЕ

Событие Event Tick

По умолчанию событие Event Tick выполняется после каждого рендера кадра (**Tick Interval 0**). Контакт данных Delta Seconds в ноде Event Tick возвращает количество времени, которое отнял рендер каждого кадра в процессе выполнения этого цикла. Если вы ранее работали в среде программирования, использующей игровые циклы, вы можете узнать в событии Event Tick UE-эквивалент этих циклов. Вы можете изменить интервалы обновлений Event Tick, щелкнув по иконке **Class Settings** на панели инструментов блюпринта. После этого вы сможете изменить настройки на панели Details в разделе **Actors Tick/Tick Interval (Sec)**.

Операторы и условия

Операторы и условия можно найти в контекстном меню блюпринта в разделе **Flow Control**. **Операторы** — это математические операции, такие как сложение, вычитание, умножение и деление. Операторы позволяют изменять значение численных переменных, таких как числа с плавающей запятой, целые числа и векторы. **Условные выражения** позволяют проверять или сравнивать выражения переменных и затем реагировать нужным вам образом. Например, вы можете проверить, равна ли одна переменная другой или является ли одна переменная больше другой.

ПОПРОБУЙТЕ САМИ

Используйте условия и операторы и установите переменную

Используя блюпринт уровня из предыдущей рубрики «Попробуйте сами», выполните следующие шаги, чтобы использовать математические операторы и установить переменную для увеличения значения переменной MyInteger.

1. Отсоедините переменную MyInteger от нода преобразования, зажав клавишу **Alt** и щелкнув по контакту данных **MyInteger**.
2. Щелкните по контакту данных **MyInteger**, перетащите его вправо и отпустите. В поисковой строке контекстного меню блюпринта введите символ **+**. В разделе **Math/Integer** выберите вариант **Integer + Integer**, чтобы добавить нод операции сложения целых чисел. Нод операции **+** будет помещен с соединением проводами. Установите нижний контакт целого числа в значение 1.
3. Перетащите переменную **MyInteger** из панели **My Blueprint** в граф и выберите вариант **Set**, чтобы поместить функцию присвоения значения переменной.
4. Соедините проводами ехес-контакт вывода нода **Event Tick** и ехес-контакт ввода функции **Set MyInteger**.
5. Соедините проводами ехес-контакт вывода нода **Set MyInteger** и ехес-контакт ввода функции **Print String**.
6. Соедините проводами контакт данных целого числа нода функции **Set** с нодом **Convert to String**, который уже соединен с функцией **Print String**. Когда вы закончите, ваш блюпринт уровня должен выглядеть, как на рис. 14.9.
7. Скомпилируйте и запустите уровень. Вы увидите, как значение MyInteger в левой части выпорта увеличивается на 1.

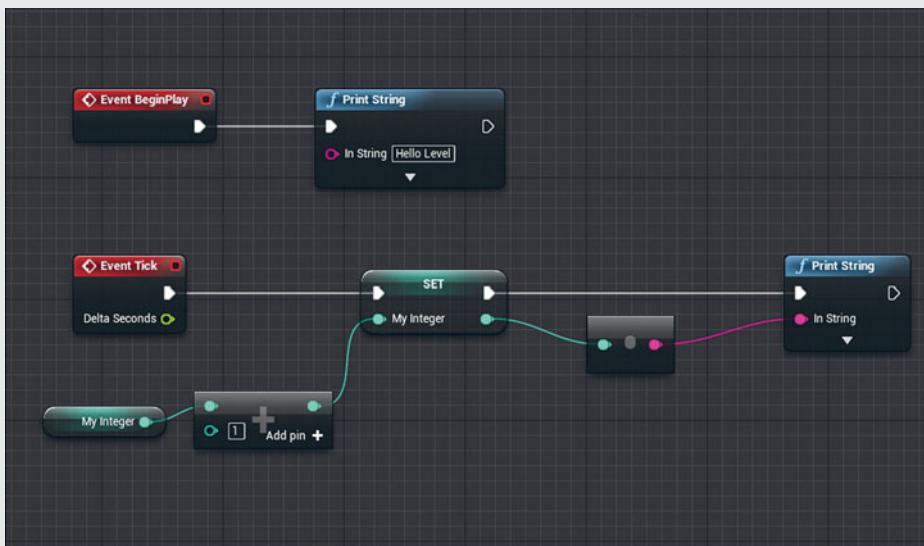


Рис. 14.9. Пример получения в блюпринте уровня целочисленной переменной, добавления значения 1 и сохранения результата в переменной

Организация и комментирование скриптов

В любой скриптовой среде организация и комментирование одинаково важны и при проверке скрипта, написанного вами месяц назад, и при работе над скриптами, написанными вашими коллегами из команды разработки. Хорошо организованные скрипты с комментариями уменьшают время разработки. Как было описано в предыдущих разделах, редактор блупринтов содержит несколько инструментов, чтобы помочь вам сохранить порядок.

Комментирование нодов

Комментирование нодов позволяет оставлять заметки к любому ноду. Просто щелкните правой кнопкой мыши по названию помещенного нода, и вы увидите всплывающий блок комментирования нода (см. рис. 14.10), или наведите курсор на нод, и через некоторое время появится всплывающий блок комментирования.

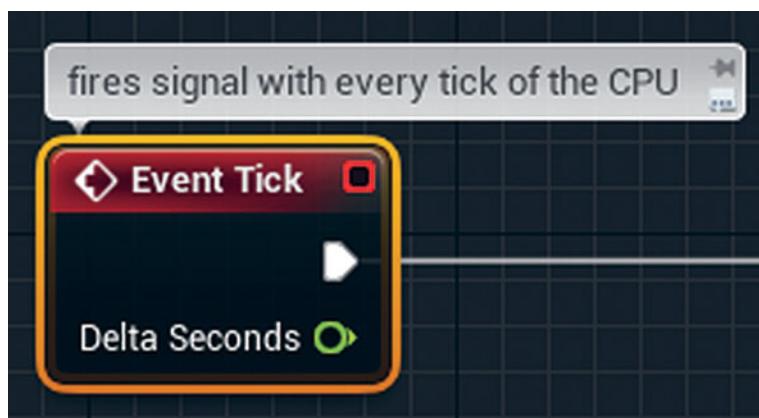


Рис. 14.10. Пример комментирования нода

Блок комментирования

Блоки комментирования (см. рис. 14.11) позволяют помещать выбранные ноды в блок и добавлять текстовые комментарии. Еще одно преимущество блока комментирования заключается в том, что при его перемещении вместе с ним перемещаются все ноды внутри него. Чтобы добавить блок комментирования к выборке нодов, нажмите клавишу **C**.

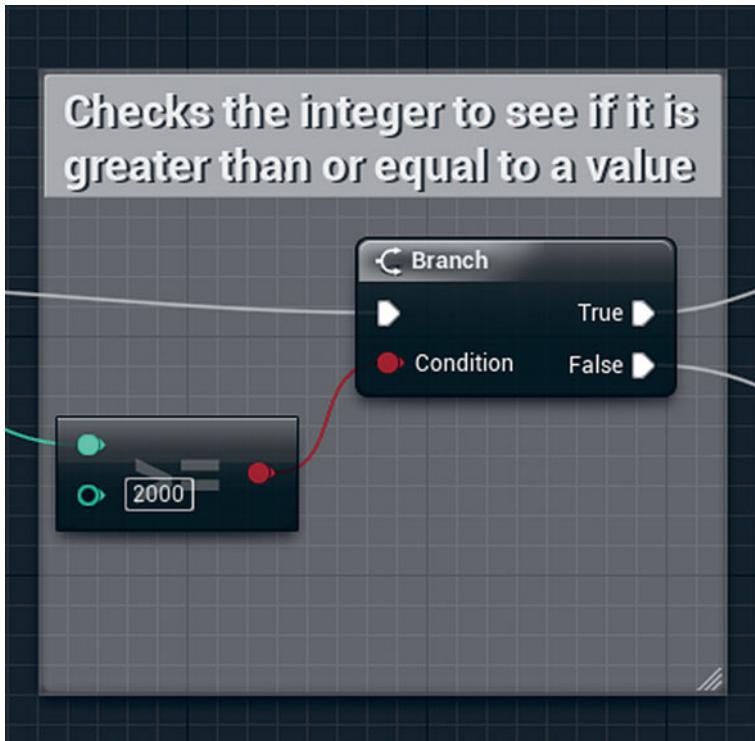


Рис. 14.11. Пример блока комментирования

Ноды перенаправления

По мере усложнения ваших скриптов появляется все больше путаницы в проводах. Нод перенаправления может помочь вам контролировать помещение проводов, как показано на рис. 14.12. Чтобы добавить нод перенаправления, щелкните правой кнопкой мыши по пустому пространству панели **Event Graph**, в поисковой строке контекстного меню напечатайте слово **reroute** и выберите вариант **reroute** из списка, чтобы поместить нод.

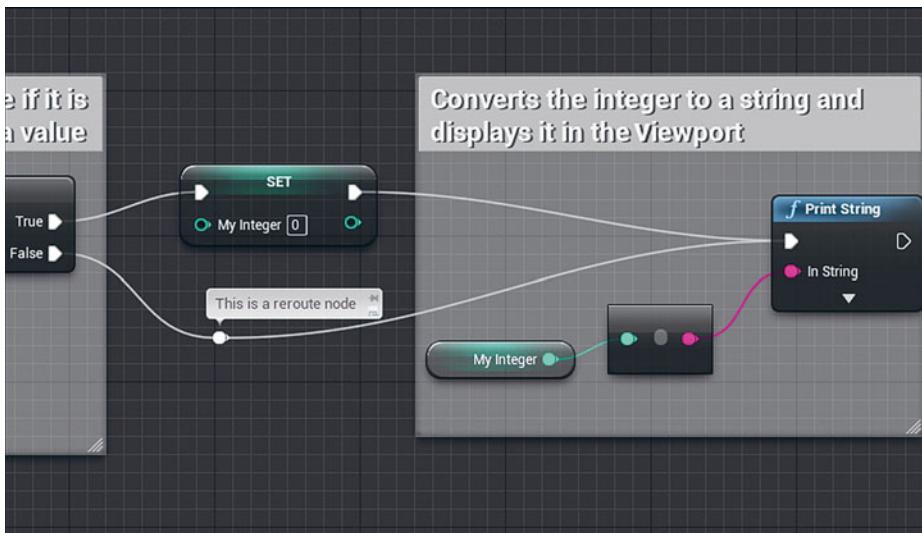


Рис. 14.12. Нод перенаправления для управления проводами

Резюме

В этом часе были представлены два метода программирования в UE4, фундаментальные концепции программирования и интерфейс редактора блупринтов. Вы научились добавлять события и функции, а также узнали, как объявлять переменные, присваивать и получать значения. Эти ключевые навыки нужны при работе с любым блупринтом в UE4.

Вопросы и ответы

Вопрос: Когда я пытаюсь добавить второй нод события (BeginPlay), редактор показывает, что первый нод уже помещен на панель Event Graph. Почему это происходит?

Ответ: Некоторые события, такие как Event Tick и BeginPlay, могут иметь только один экземпляр в блупринте. Использование нескольких панелей Event Graph для одного блупримта могло бы привести к проблемам, но этого не происходит. Если вы хотите направить событие из одного графа в другой, вы можете создать пользовательское событие, которое вызывает Event Tick или BeginPlay каждый раз, когда оно посылает сигнал.

Вопрос: Как выбрать имя для переменной?

Ответ: Вы можете дать переменной любое имя, которое захотите. Постарайтесь выбрать короткое, описательное имя, которое упростит узнавание

переменной. Лучше всего установить соглашение об именах, которое можно применять ко всем блупринтам в проекте в целях последовательности.

Вопрос: Можно ли поменять имя переменной, если у переменной уже есть имя?

Ответ: Да, вы можете изменить имя переменной в окне My Blueprint или можете выбрать переменную и изменить ее имя на панели Details блупринта. Изменение имени обновляет все экземпляры переменной в блупринте.

Вопрос: Можно ли изменить тип переменной после ее создания и использования в графе событий?

Ответ: Да, вы можете изменить тип уже созданной и используемой переменной, но это повлияет на блупринт. Помните, что контакты данных в функциях нуждаются в определенных типах данных, и, если вы измените тип переменной, вы можете оборвать какой-либо провод, соединенный с контактами данных переменных. В таком случае вам понадобится вернуться и настроить скрипт вручную.

Вопрос: Могу ли я использовать одни и те же скрипты блупринта уровня на другом уровне?

Ответ: Нет. Хотя вы можете копировать и вставлять последовательности событий из одного блупринта в другой, все переменные должны быть заново созданы в новом скрипте блупринта. Также многие последовательности событий и действия в блупринте уровня связаны с определенными актерами в этом уровне, которых не существует в новом уровне. В этом заключается преимущество использования актером блупринт-классов, которые вы изучите в следующих часах.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Истинно или ложно высказывание: блупринт можно использовать для того, чтобы переписать базовый движок рендеринга UE4.
2. Истинно или ложно высказывание: использование функции Print String — не лучший способ коммуникации с игроками.
3. Истинно или ложно высказывание: скрипты блупринта компилируются в байткод.

4. Истинно или ложно высказывание: вы можете иметь более одного события BeginPlay в одном скрипте блюпринта.
5. Что такое массив?
6. Истинно или ложно высказывание: комментирование скриптов — пустая трата времени.

Ответы

1. Ложь. Если вам нужно модифицировать компоненты основного движка для вашей игры, вам необходимо программировать на C++.
2. Истина. Функция Print String должна использоваться только для отладки.
3. Истина. Блюпринты компилируются в байткод.
4. Ложь. Вы можете иметь только одно событие BeginPlay в блюпринте, но вы можете использовать другое событие, чтобы передать сигнал ноду Sequence для разделения сигнала.
5. Массив — это переменная, хранящая набор значений одного типа.
6. Ложь. Вы всегда должны комментировать свои скрипты.

Упражнение

Продолжая последнее упражнение «Попробуйте сами», добавьте вторую целочисленную переменную, которая меняет добавляемое к целочисленной переменной MyInteger значение с каждым циклом. Затем используйте условие в последовательности Event Tick, проверяющее значение целочисленной переменной MyInteger и приравнивающее MyInteger к 0, когда оно достигает значения 2000 или выше. Затем закончите последовательность пользовательским событием, которое при вызове воспроизводит вывод строки. Пример показан на рис. 14.13.

1. Откройте блюпринт уровня, над которым вы работали в этом часе.
2. Объявите новую переменную, назовите ее **MyIntCounter**, смените ее тип на **integer** и дайте ей значение по умолчанию **5**.
3. Добавьте переменную **MyIntCounter** в граф события и соедините ее с нодом **+**.
4. После того как вы установите значение **SET** для переменной **MyInteger**, проверьте, является ли значение большим или равным (\geq) 2000.

В поисковой строке контекстного меню блюпринта введите **integer >=** и выберите вариант **integer >= integer**, чтобы поместить нод. Этот нод возвращает значение **0** (ложь) или **1** (истина).

5. Установите контакт данных целого числа **B** в ноде **>=** в значение **2000**.
6. Проверьте, является ли условие истинным или ложным, используя нод **Branch**. Щелкните и перетащите красный контакт булевых данных в нод **>=**, чтобы вызвать контекстное меню блюпринта. В поисковой строке контекстного меню введите **Branch** и выберите этот вариант из списка, чтобы поместить нод.
7. Соедините ехес-контакт вывода нода **SET MyInteger** с ехес-контактом нода **Branch**.
8. Перетащите ехес-контакт вывода **True** из нода **Branch**, чтобы вызвать контекстное меню блюпринта. В поисковой строке введите **set myinteger** и выберите вариант **Set MyInteger**, чтобы поместить нод.
9. В ноде **SET MyInteger** введите **0** в текстовую строку рядом с контактом данных **MyInteger**.
10. Теперь создайте пользовательское событие. Под последовательностью в графе события щелкните правой кнопкой мыши по пустой области, чтобы вызывать контекстное меню. В поисковой строке введите **custom** и выберите вариант **Add Custom Event** из списка, чтобы поместить нод события. Переименуйте событие на **MyCustomEvent**.
11. Перетащите ехес-контакт вывода нода события **MyCustomEvent** и в поисковой строке контекстного меню блюпринта введите **print**. Выберите вариант **Print String** из списка, чтобы поместить нод.
12. Из панели **My Blueprint** перетащите переменную **MyInteger** в контакт данных переменной **In String** в ноде **Print String**. Блюпринт автоматически помещает переменную и добавляет нод конвертации.
13. Соедините ехес-контакт **False** с нодом **Branch** в конце первой последовательности. Перетащите его, чтобы вызвать контекстное меню. В поисковой строке введите **mycustomevent** и выберите вариант **MyCustomEvent** из списка, чтобы поместить функцию.
14. Перетащите провод из ехес-контакта вывода в нод **SET MyInteger**, соединенный с нодом **Branch**, и соедините его с синей функцией

MyCustomEvent. После того как вы закончите, ваш блюпринт уровня должен выглядеть, как на рис. 14.13.

- Сохраните уровень и запустите его.

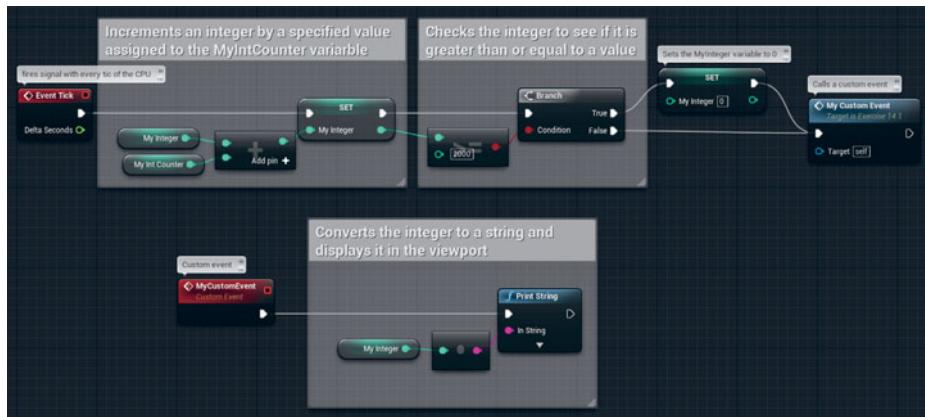


Рис. 14.13. Пример скрипта из упражнения

15-Й ЧАС

Работа с блюпринтами уровней

Что вы узнаете в этом часе

- ▶ Назначение актеров событиям в блюпринте уровня
- ▶ Назначение актера в качестве ссылочной переменной в блюпринте уровня
- ▶ Получение и присвоение значений свойств актера в блюпринте уровня
- ▶ Использование функции Activate
- ▶ Использование функции Play Sound at Location

В каждом уровне есть связанный с ним блюпринт уровня, хотя он может и не использоваться. Несмотря на то что уровень хранит ссылки на все актеры, помещенные на него, блюпринт уровня не владеет информацией об актерах на уровне, если не передать ему эту информацию. В этом часе вы научитесь, как назначать помещенные актеры событиям коллизий и как добавлять их в качестве ссылочных переменных в блюпринт уровня. Затем вы научитесь изменять свойства актеров в редакторе блюпринта уровня, когда событие запущено.

ПРИМЕЧАНИЕ

Подготовка к практике

Прежде чем вы начнете этот час, создайте новый проект из шаблона **First Person** со стартовым контентом и создайте уровень по умолчанию.

Чтобы попрактиковаться в назначении актеров событиям и ссылочным переменным, вы научитесь создавать простую последовательность событий. Когда игрок движется в определенную область на уровне, запускается событие Overlap, изменяется материал, назначенный актеру статичного меша, активируется система частиц и воспроизводится звук.

Чтобы назначить актер событию, вам потребуется актер, помещенный на уровень. В данном случае вы можете использовать актер Trigger.

Существует несколько способов определения области на уровне, с которой может взаимодействовать игрок. Компания Epic предоставляет три основных класса форм триггеров (Box Trigger, Capsule Trigger и Sphere Trigger), а также класс Trigger Volume, работающий с событиями коллизий (см. рис. 15.1). Этот час фокусируетсяся на работе с классами Box Trigger, Capsule Trigger и Sphere Trigger.

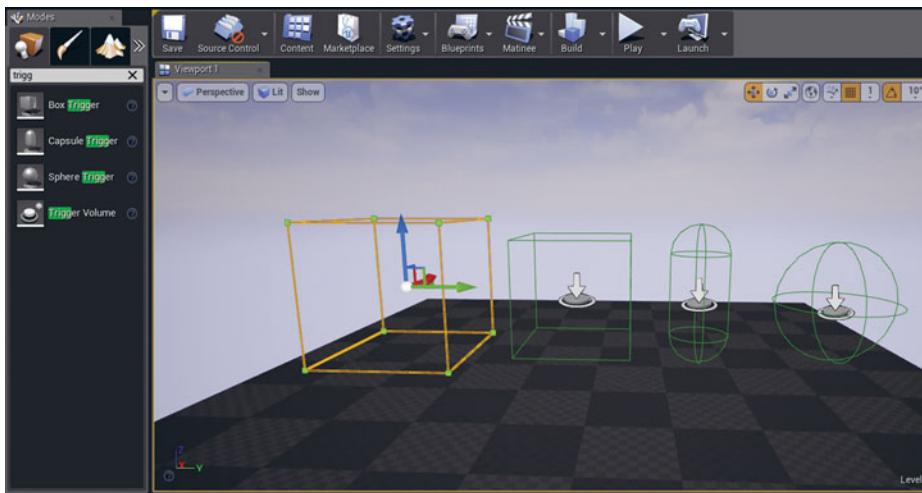


Рис. 15.1. Помещенные триггерные зоны

Все эти классы можно найти на вкладке **Place** панели **Modes**. Найдите класс **Trigger**, воспользовавшись поисковой строкой на вкладке **Place**, или выберите категорию **All Classes**, перетащите **Box Trigger** и поместите его на уровень. Прежде чем назначать помещенный триггер событию, необходимо изменить некоторые свойства этого триггера. Поместив и выбрав актер **Box Trigger**, измените его имя на **MyTriggerBox** на панели **Details** и в разделе **Shape** настройте его размер, установив значение **100** единиц для параметра **Box Extent** по осям X, Y и Z. Вы можете настраивать параметры формы в любое время, но сейчас вам необходимо только убедиться, что попасть в область, определенную актером **Box Trigger**, легко. Затем будет уместно изменить настройки рендеринга актера **Box Trigger**, поэтому в разделе **Rendering** панели **Details** снимите флажок **Actor Hidden in Game**. Так определенная область **Box Trigger** станет видимой при запуске и игровом тестировании уровня.

СОВЕТ

Рендеринг и видимость актеров

Практически каждый помещенный актер имеет свойства рендеринга, которые могут включать и отключать видимость актера как в редакторе, так и на уровне в процессе игры. В большинстве случаев вам не понадобится, чтобы игроки видели триггер,

поскольку это может испортить чувство погружения, в зависимости от визуального стиля игры. Однако включение видимости путем снятия флагка **Actor Hidden in Game** может служить временной мерой, помогающей устанавливать, производить игровое тестирование и отладку блюпринта. Просто не забудьте отключить видимость, когда закончите работу. Настройки видимости актера можно найти в разделе **Rendering** панели **Details** уровня, когда актер выбран.

Настройки коллизий актеров

Когда вы создали уровень по умолчанию, установили режим игры и поместили актер Box Trigger на уровень, вы можете создать событие коллизии в блюпринте уровня. События, основанные на коллизиях, напрямую связаны со свойствами коллизии назначенного актера, в данном случае со свойствами коллизии актера Box Trigger. Когда актер Box Trigger выбран, вы можете увидеть некоторые важные свойства на панели Details редактора блюпринта уровня (см. рис. 15.2): **Simulation Generate Hit Events**, **Generate Overlap Events** и **Collision Responses**.

См. 4-й час «Работа с актерами статичных мешей», чтобы узнать больше о типах реакций на коллизии.

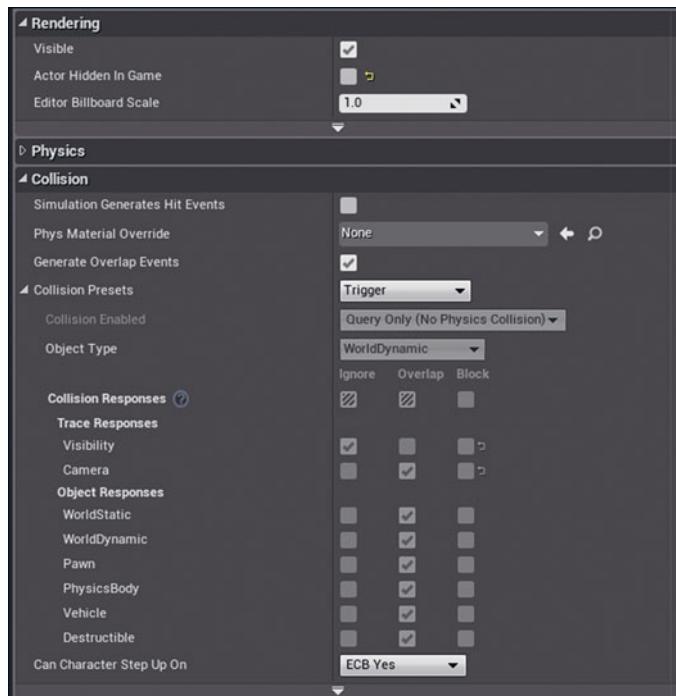


Рис. 15.2. Свойства коллизий актера триггера

Когда флагги **Simulation Generate Hit Events** и **Generate Overlap Events** установлены, они позволяют актеру транслировать события коллизий в блюпринт уровня. *События ударов* (Hit Events) происходят, когда оболочки коллизий актеров соприкасаются, но не пересекаются. *События перекрытия* (Overlap Events) происходят, когда две оболочки коллизий актеров перекрываются или прекращают перекрываться. События ударов и перекрытия напрямую связаны с каждым пресетом коллизий актеров. Если два актера настроены на блокирование друг друга, они никогда не перекрываются.

В блюпринте ноды событий отвечают за сигналы получения ударов и перекрытия от следующих актеров.

- ▶ **Событие OnActorBeginOverlap.** Запускается единожды каждый раз, когда оболочка коллизии актера перекрывается оболочкой коллизии другого актера, тип реакции на коллизию которого соответствует типу реакции первого актера. Если актер покидает область коллизии актера события и затем снова входит в нее, событие запускается вновь.
- ▶ **Событие OnActorEndOverlap.** Работает таким же образом, как событие OnActorBeginOverlap, но только когда актер покидает область.
- ▶ **Событие OnActorHit.** Не требует, чтобы оболочки коллизий актеров перекрывались, но они должны соприкоснуться. Этот тип событий особенно полезен, когда вы работаете с актерами, симулирующими физику. Если актеру Physics назначено событие удара, и он при этом стоит на земле, это событие продолжает срабатывать.

Назначение актеров событиям

Назначение актера событиям в блюпринте уровня производится просто. Необходимо, чтобы актер был выбран на уровне, а событие было назначено в блюпринте уровня через контекстное меню блюпринта. Выберите актер на уровне и щелкните по нему правой кнопкой мыши в графе событий редактора блюпринтов уровней, чтобы вызвать контекстное меню. В контекстном меню убедитесь, что флагок для параметра **Context Sensitive** установлен, разверните категорию **Add Event for MyTriggerBox** и в подкатегории **Collision** выберите нужный вам тип нода события коллизии (см. рис. 15.3).

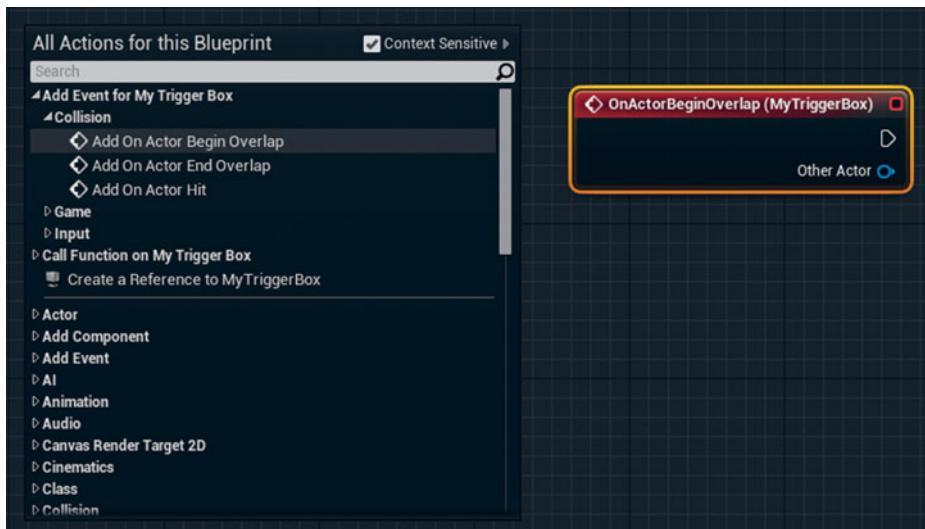


Рис. 15.3. Контекстное меню блюпринта с установленным флагжком **Context Sensitive**

ПРИМЕЧАНИЕ

Параметр **Context Sensitive**

В правом верхнем углу контекстного меню блюпринта есть параметр **Context Sensitive**. Включение этой настройки организует контент в контекстном меню на основе вашего текущего выбора, показывая только события и функции, которые работают с выбранным актером, компонентом или типом переменной.

ПОПРОБУЙТЕ САМИ

Назначьте актер событию **OnActorBeginOverlap**

Выполните следующие шаги, чтобы назначить актер Box Trigger событию **OnActorBeginOverlap**.

1. Откройте редактор блюпринтов уровней.
2. Выберите помещенный на уровень актер Box Trigger.
3. Щелкните правой кнопкой мыши в графе событий блюпринта уровня, чтобы вызвать контекстное меню блюпринта. Убедитесь, что в правом верхнем углу установлен флагжок **Context Sensitive**.
4. В верхней части списка действий в контекстном меню разверните список **MyTriggerBox**, щелкнув по треугольнику слева от названия категории.
5. Разверните подкатегорию **Collision** и выберите вариант **Add OnActorBeginOverlap**, чтобы добавить над событием с назначенным ему выбранным актером, обозначенный именем актера в круглых скобках после имени события.

6. Щелкните и перетащите ехес-контакт вывода события OnActorBeginOverlap. Отпустите кнопку мыши, после чего вновь откроется контекстное меню. Добавьте нод Print String (который вы найдете в разделе Utilities ⇒ String). Ваша последовательность событий теперь должна выглядеть, как показано на рис. 15.4.
7. В функции Print String введите Hello Level.
8. Запустите уровень, пройдитесь по нему и зайдите в область Box Trigger.



Рис. 15.4. Нод события OnActorBeginOverlap с назначенным актером

Нод события OnActorBeginOverlap имеет ехес-контакт, который проводит сигнал события, и контакт вывода данных Other Actor, возвращающий ссылку на актер, инициировавший событие перекрытия. Если вы перетащите контакт вывода данных в контакт данных In String функции Print String, редактор блюпринта уровня автоматически добавит функцию Get DisplayName, которая возвращает имя актера, вызвавшего событие, и пересыпает его в функцию Print String, как показано на рис. 15.5. Произведите эти изменения, запустите уровень и войдите в область еще раз. На этот раз функция Print String возвращает имя объекта Pawn.

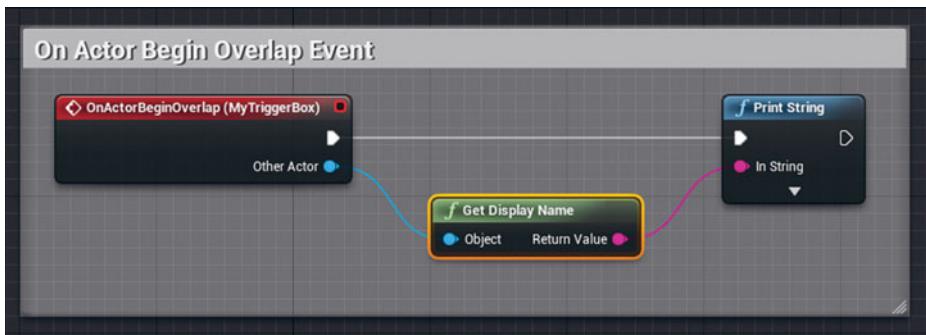


Рис. 15.5. Последовательность событий OnActorBeginOverlap, которая отображает имя Other Actor — актера, инициировавшего событие

Назначение актеров ссылочным переменным

В блюпринте вы можете изменять практически все свойства, которые содержатся на панели **Details** уровня для актера. Ссылочная переменная актера относится к назначенному актеру на уровне и дает блюпринту уровня доступ к свойствам актера.

Процесс назначения актера ссылочной переменной актера аналогичен назначению актера событию. Открыв редактор блюпринта уровня, выберите нужный актер на уровне (см. рис. 15.6). В графе событий щелкните правой кнопкой мыши, чтобы открыть контекстное меню блюпринта, и выберите вариант **Add Actor as Reference Variable**, чтобы поместить в блюпринт переменную, которой назначен актер.



Рис. 15.6. Ссылочная переменная актера

Компоненты актеров

Все актеры, помещенные на уровень, имеют основные параметры настройки, присущие всем актерам, такие как настройки трансформации и рендеринга. Это свойства на уровне актера, в то время как другие свойства влияют на актер на уровне компонента. Компонент — это подобъектный элемент актера, и большинство (если не все) актеров имеют, по меньшей мере, один компонент (см. рис. 15.7). Например, актеры статичных мешей имеют компонент Static Mesh, в то же время актеры эмиттера имеют компонент Particle System, а триггер-актеры — компонент Collision.

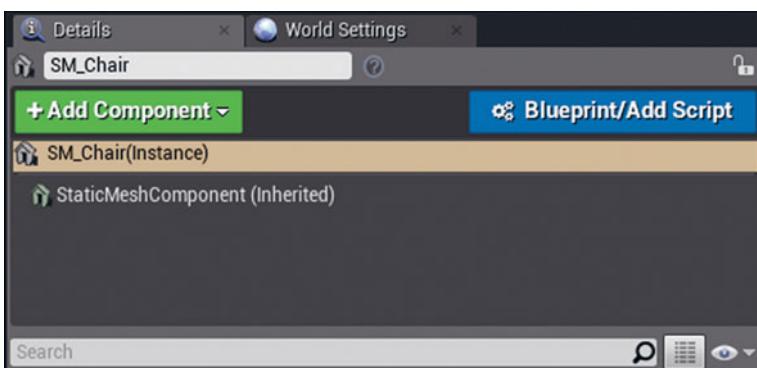


Рис. 15.7. Список компонентов актеров статичных мешей на панели **Details**

ПРИМЕЧАНИЕ

Компоненты

Существует множество типов компонентов. Некоторые актеры имеют только по одному компоненту, в то время как другие могут иметь множество компонентов.

См. 16-й час, «Работа с блюпринт-классами», чтобы узнать больше о работе с компонентами в блюпринте.

Получение и присвоение значений свойств актеров

По аналогии с любыми другими типами переменных вы можете получать и присваивать значения свойств актеров. При получении значения свойства актера создается нод переменной, возвращающий тип данных этого свойства. Например, получение данных о местоположении актера возвращает вектор, как показано на рис. 15.8.

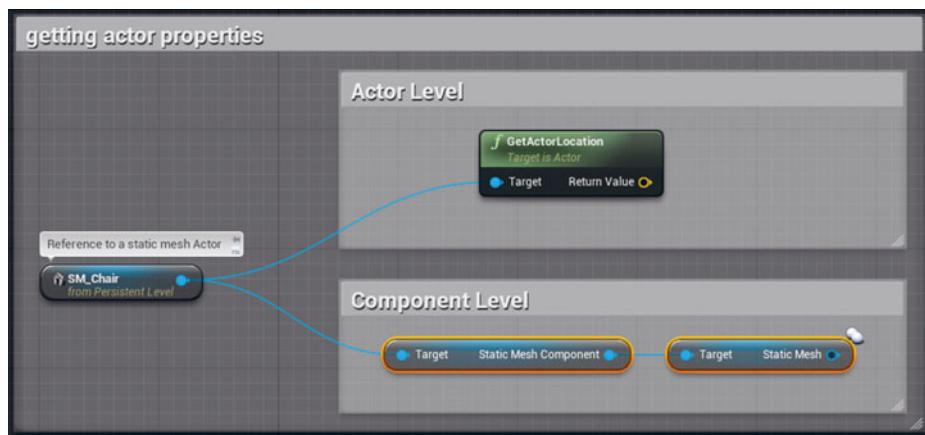


Рис. 15.8. Получение значений свойств актера на уровне актера и на уровне компонента

Присвоение значений свойств актера или его компонентов требует наличия функции, которая имеет своей целью актер или его компоненты, как показано на рис. 15.9.

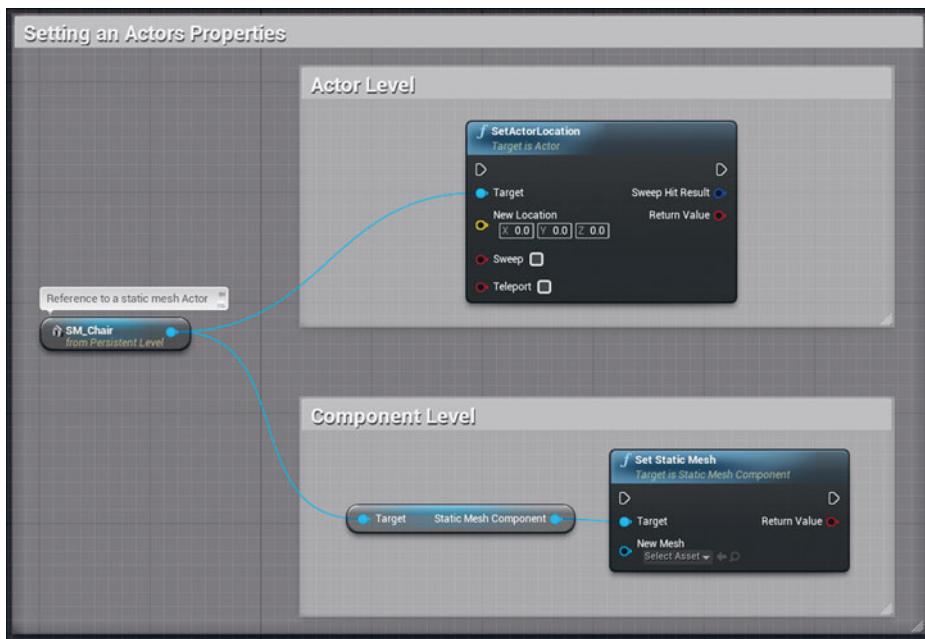


Рис. 15.9. Присвоение значений свойств актера на уровне актера и на уровне компонента

Цели функций

В большинстве случаев, чтобы функция выполнялась надлежащим образом, ей необходимо задать цель, на которую она должна повлиять. Эта цель описывается данными цели в контакте. Синий контакт цели говорит функции, на какой актер или компонент актера функция должна повлиять. Некоторые функции работают на уровне актеров, в то время как другие работают на уровне компонентов. Если вам необходимо изменить свойство актера на уровне актера, вам понадобится функция, целью которой является актер, но если вам необходимо изменить свойство компонента актера, вам потребуется функция, целью которой является компонент. Например, если вы хотите изменить местоположение актера, работайте на уровне актера, но если вы хотите изменить ассет статичного меша, назначенный актеру статичного меша, работайте на уровне компонентов статичного меша.

Если вы изменяете свойство на уровне актера, вам нужна функция, которая работает на уровне актера, и, если вы изменяете свойство на уровне компонента, вам нужна функция, работающая на уровне компонента. Под именем функции есть описание, которое информирует вас о том, что является целью этой функции — актер или компонент.

ПОПРОБУЙТЕ САМИ

Измените свойства материала актера

Выполните следующие шаги, чтобы, используя Box Trigger с событием OnActorBeginOverlap, изменить материал, назначенный актеру статичного меша.

1. Добавьте актер Box Trigger на уровень. На панели **Details** уровня в разделе **Shapes** установите значение **100** параметра **Box Extents** для осей X, Y и Z.
2. В графе событий редактора блюпринта уровня добавьте событие **OnActorBeginOverlap** с назначенным ему актером Box Trigger.
3. Поместите актер статичного меша Cone в центр актера Box Trigger и измените его имя на **MyCone**.
4. Найдите материал на панели **Content Browser** в папке **Starter Content** ⇒ **Materials** и перетащите его на помещенный конус.
5. Выбрав конусный актер статичного меша, щелкните правой кнопкой мыши по графу события в редакторе блюпринта уровня. В контекстном меню блюпринта выберите вариант **Create a Reference to** для актера статичного меша MyCone.
6. Поместив ссылочную переменную актера, щелкните по синему контакту вывода данных. Перетащите его и отпустите кнопку мыши, чтобы вызвать контекстное меню. В поисковой строке введите слова **set Material**.
7. Выберите вариант **Set Material** из списка, чтобы добавить функцию Set Material и ссылочную переменную компонента.
8. Соедините ехес-контакт вывода события OnActorBeginOverlap с ехес-контактом ввода Set Material.
9. В функции Set Material назначьте новый материал, выбрав его в свойствах материалов на панели **Content Browser**. Когда вы закончите, ваша последовательность событий должна выглядеть, как на рис. 15.10.
10. Запустите уровень и переместите аватар вплотную к конусу.

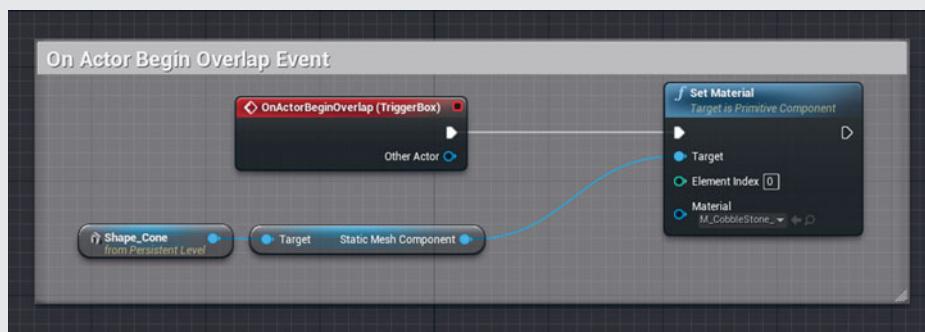


Рис. 15.10. Последовательность событий OnActorBeginOverlap изменяет материал, назначенный актеру статичного меша

Поскольку вы подходите к конусу на уровне, с которым вы работали в предыдущей рубрике «Попробуйте сами», и оболочка коллизии актера перекрывает актер Box Trigger, событие запускается и изменяет материал. Но когда аватар отдаляется, новый материал остается. В следующем упражнении «Попробуйте сами» вы переzapустите материал, заставив его меняться обратно к изначально назначенному, когда аватар покидает область актера-триггера.

ПОПРОБУЙТЕ САМИ

Перезапустите свойства материала актера

Выполните следующие шаги, чтобы, используя событие OnActorBeginOverlap, изменить материал обратно на изначально назначенный.

1. В графе редактора блюпринта уровня добавьте событие OnActorBeginOverlap с назначенным ему актером Box Trigger.
2. В редакторе блюпринта уровня, выбрав конусный актер статичного меша, щелкните правой кнопкой мыши по графу событий. В контекстном меню блюпринта выберите вариант **Create a Reference to MyCone**, чтобы добавить вторую ссылочную переменную для актера MyCone в блюпринт уровня.
3. Поместив новую ссылочную переменную актера, щелкните по синему контакту вывода данных. Перетащите его и отпустите кнопку мыши, чтобы вызвать контекстное меню блюпринта. В поисковой строке введите **set Material**.
4. Выберите вариант **Set Material** из списка, чтобы добавить еще одну функцию Set Material.
5. Соедините ехес-контакт вывода события OnActorBeginOverlap с ехес-контактом ввода **Set Material**.
6. В функции Set Material назначьте свойству материала изначальный материал, который вы поместили в актер статичного меша MyCone в шаге 4 предыдущей рубрики «Попробуйте сами». Когда вы закончите, ваша последовательность событий должна выглядеть, как на рис. 15.11.
7. Запустите уровень и перемещайте аватара внутрь и наружу Box Trigger, чтобы увидеть изменение материала.

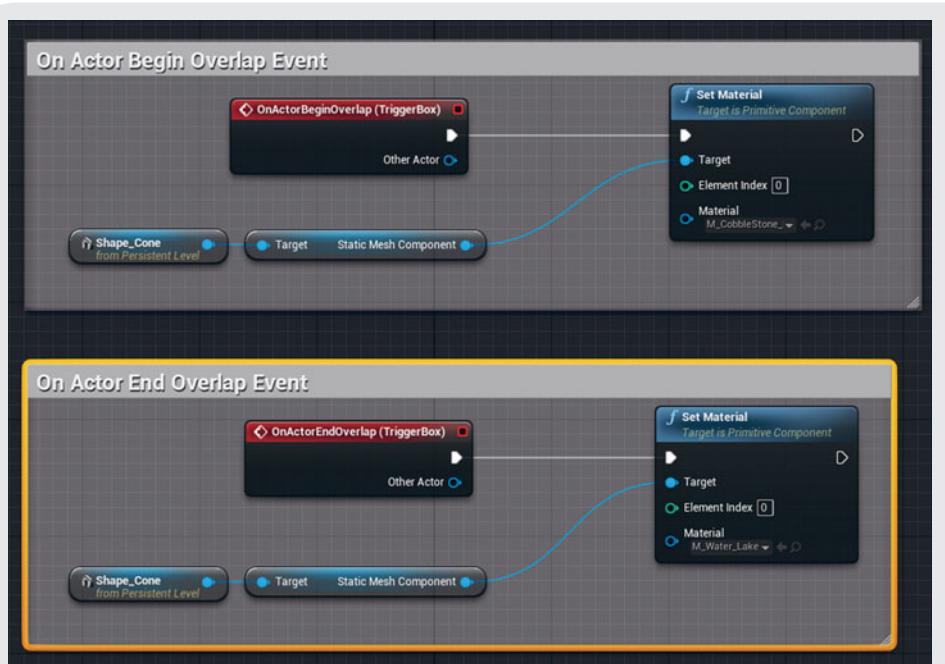


Рис. 15.11. События перекрытия коллизий

Свойство **Activate**

Для некоторых актеров, таких как эмиттеры или звуки окружения, свойство активации указывает начать воспроизведение эмиттера частиц или звука в тот момент загрузки игры. Это свойство включено по умолчанию. Деактивация этого свойства на панели **Details** уровня для актера останавливает воспроизведение до тех пор, пока свойство не включится. Это свойство может быть установлено, изменено и восстановлено в блюпринте некоторыми функциями: **Activate**, **Deactivate**, **IsActive**, **SetActive** и **ToggleActive**. В следующем упражнении «Попробуйте сами» вы поместите актер эмиттера частиц на уровень. У вас также будет последовательность **OnActorBeginOverlap**, активирующую актер эмиттера, и последовательность **OnActorEndOverlap**, деактивирующую его.

ПОПРОБУЙТЕ САМИ

Активация и деактивация актера эмиттера частиц

Выполните следующие шаги, чтобы расширить предыдущие упражнения «Попробуйте сами» и завершить последовательность OnActorBeginOverlap.

1. Найдите ассет системы частиц *P_Explosion* в папке стартового контента *Particles* на панели **Content Browser** и поместите его в центр или рядом с центром актера Box Trigger так, чтобы вы могли его видеть при взаимодействии с триггером.
2. Выбрав помещенный актер эмиттера на уровне, щелкните правой кнопкой мыши по графу событий блюпринта уровня, чтобы открыть контекстное меню, и выберите вариант **Create a Reference to P_Explosion**.
3. Создав ссылку на актер *P_Explosion*, щелкните и перетащите синий контакт вывода данных нода переменной и отпустите кнопку мыши. В поисковой строке контекстного меню введите слово **activate** и выберите вариант **Activate (ParticleSystemComponent)**, после чего будет добавлено два нода: функция *Activate* и ссылочная переменная компонента, которая указывает на систему частиц, назначенную актеру эмиттера.
4. Соедините ехес-контакт вывода Set Material с ехес-контактом ввода *Activate*.
5. Нажмите комбинацию клавиш **Ctrl+W**, чтобы скопировать и вставить ссылочную переменную *P_Explosion*. Перетащите функцию *Deactivate* и соедините ее с концом цепочки событий *OnActorEndOverlap*, протянув провод к ехес-контакту вывода функции *Set Material*. Когда вы закончите, ваша цепочка событий должна выглядеть, как на рис. 15.12.
6. Запустите уровень и войдите и выйдите из зоны. Каждый раз, когда актер павна перекрывает триггерную зону, событие запускается и активирует эмиттер частиц.

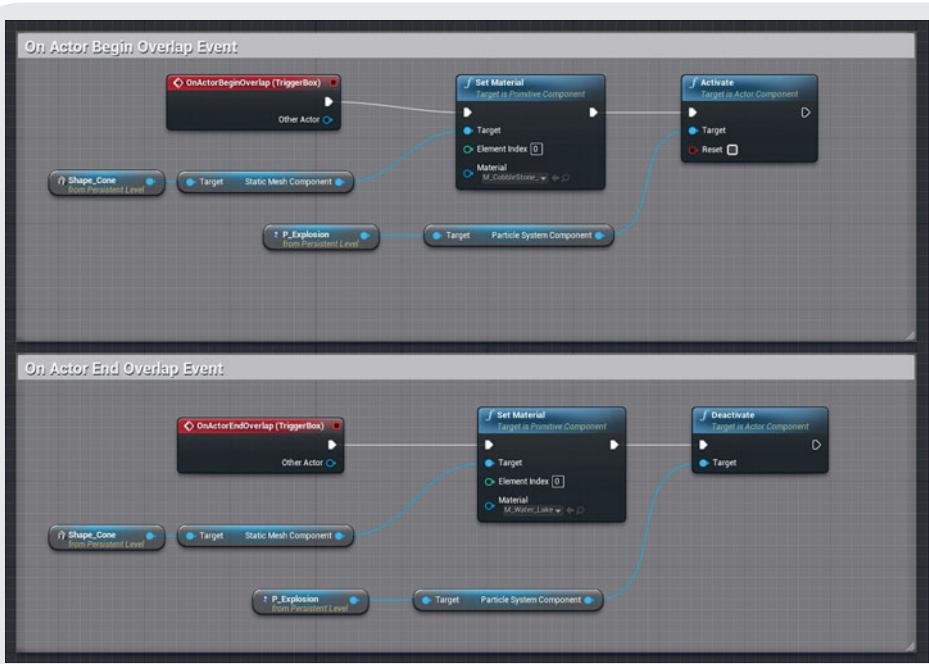


Рис. 15.12. Блюпринт переключения эмиттера частиц

Функция Play Sound at Location

В предыдущих примерах вы использовали редактор блюпринта уровня, чтобы изменять помещенные актеры и управлять ими. В следующем примере вы будете использовать функцию **Play Sound at Location** для воспроизведения ассета звукового сигнала «на лету». Этой функции требуется информация о том, какой звуковой ассет вы хотите воспроизвести и в каком месте на уровне он должен быть воспроизведен. Звуковой ассет, который необходимо воспроизвести, может быть назначен из панели **Content Browser** или выпадающего списка функций. Данные о местоположении вам необходимо получить от одного из уже помещенных актеров.

В следующей рубрике «Попробуйте сами» вы научитесь добавлять функцию **Play Sound at Location** и получать данные о местоположении от помещенного актера.

ПОПРОБУЙТЕ САМИ

Добавьте функцию Play Sound at Location

Продолжая упражнение «Попробуйте сами», выполните следующие шаги, чтобы использовать функцию Play Sound at Location для воспроизведения звукового сигнала или ассета звуковой волны при срабатывании функции OnActorBeginOverlap.

1. Откройте блюпринт уровня, с которым вы работали в предыдущих упражнениях «Попробуйте сами».
2. В графе событий найдите цепочку событий OnActorBeginOverlap из предыдущего упражнения «Попробуйте сами».
3. Щелкните по ejec-контакту вывода функции Activate в конце последовательности. Перетащите его и отпустите кнопку мыши, после чего откроется контекстное меню блюпринта. Введите **Play Sound at Location** в поисковой строке и выберите функцию **Play Sound at Location** из списка.
4. Найдите ассет звукового сигнала Explosion_Cue в папке *Audio* стартового контента на панели **Content Browser** или используйте выпадающее меню нода функции. Щелкните по ассету звукового сигнала Explosion_Cue и перетащите его в свойство Sound функции Play Sound at Location.
5. Выберите актер-триггер и добавьте его в качестве ссылочной переменной в блюпринт уровня.
6. Щелкните по синему контакту данных нода ссылочной переменной, перетащите его и отпустите. В поисковой строке контекстного меню введите **get Actor location** и выберите вариант **Get Actor Location Function** из списка.
7. Щелкните по желтому векторному контакту вывода данных функции Get Actor и перетащите провод в желтый векторный контакт ввода данных функции Play Sound at Location. Когда вы закончите, ваша цепочка событий должна выглядеть, как на рис. 15.13.
8. Запустите уровень и перемещайте аватара внутрь и наружу области актера-тригера, чтобы услышать звук.

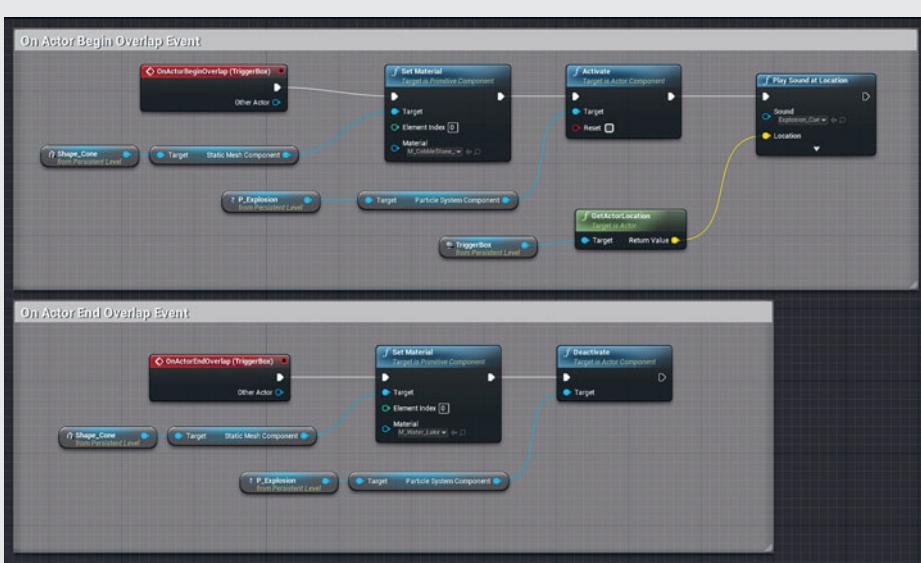


Рис. 15.13. Помещенные триггерные зоны

ПРИМЕЧАНИЕ

Актер звука окружения

При желании вы можете использовать актер звука окружения и функцию `ToggleActivate` для воспроизведения звука при запуске события. Просто поместите незацикленную звуковую волну или звуковой сигнал на уровень и выполните шаги, описанные в предыдущей рубрике «Попробуйте сами». Замените актер эмиттера частиц на актер звука окружения.

Использование актеров физики для активации событий

Ранее вы использовали только аватар для инициализации событий коллизий. Далее вы сделаете так, что событие будет запускаться от актера статичного меша, симулирующего физику, или от пересечения пуль с актерами-триггерами. Мы не будем заниматься программированием, вам лишь понадобится внести несколько незначительных изменений в свойства коллизий на панели **Details** актеров, вовлеченных в процесс.

ПОПРОБУЙТЕ САМИ

Используйте актер физики для запуска событий

Продолжая упражнение «Попробуйте сами», выполните следующие шаги, чтобы заставить событие OnActorBeginOverlap срабатывать, когда актер статичного меша, симулирующий физику, перекрывает зону.

1. В папке *Props* стартового контента на панели **Content Browser** выберите ассет статичного меша и поместите его над Box Trigger из предыдущих упражнений.
2. В свойствах Physics актера статичного меша на панели **Details** установите флагок рядом с параметром **Simulate Physics**.
3. В свойствах Collision актера статичного меша на панели **Details** установите флагок рядом с параметром **Generate Overlap Events**.
4. Запустите уровень. Когда актер статичного меша падает в область, определенную актером-триггером, срабатывает событие перекрытия, и материал изменяется.

ПРИМЕЧАНИЕ

Использование актеров-пуль для активации событий

Если вы используете версию 4.8 или более раннюю, для пресетов коллизий актеров-триггеров по умолчанию установлен параметр **Overlap with Everything but Projectiles**. Чтобы изменить это, поменяйте значение этого параметра на **Custom**. Это действие разблокирует типы реакции объектов, и вы сможете установить категорию **Projectile** в значение **Overlap**. Запустите уровень еще раз и выстрелите в Box Trigger. Категория **Projectile** была удалена в последующих версиях.

Резюме

В этом часе вы научились назначать актеры событиям и добавлять их в блюпринт уровня в качестве ссылочных переменных. Вам была представлена работа с актерами-триггерами, рассказано, как изменять свойства актера в редакторе блюпринта уровня. Вы научились активировать и деактивировать эмиттер частиц, чтобы воспроизводить звуковой ассет в определенном месте, когда запускается цепочка событий. Эти аспекты составляют основные цепочки событий во многих играх. Не забывайте включать свойство **Actor Hidden in Game** для помещенных актеров-триггеров, когда работа окончена.

Вопросы и ответы

Вопрос: Когда я назначаю актер событию коллизии в блюпринте уровня, событие не выполняется. Почему?

Ответ: В таком случае следует проверить несколько вероятных причин. Во-первых, взгляните на свойства коллизий актера, назначенного событию, и убедитесь, что рядом с параметром Generate Overlap Events установлен флашок. Затем проверьте настройки пресетов коллизий и типов реакции объектов, чтобы убедиться, что тип класса актера, который должен вызвать событие, установлен в значении Overlap. Затем проверьте настройки пресетов коллизий и типов реакции объектов для других вызывающих события актеров. Если таковыми актерами являются актеры статичных мешей, проверьте ассеты статичных мешей и убедитесь, что они имеют назначенные им оболочки коллизий.

Вопрос: Я назначил неверный актер событию. Могу ли я поменять его или придется удалить событие и начать заново?

Ответ: Хотя вы можете просто удалить нод события и начать с нуля, вы также можете поменять актер, назначенный уже созданному ноду события. Выберите новый актер на уровне и затем в редакторе блюпринтов уровня щелкните правой кнопкой мыши по заголовку уже помещенного нода события и выберите вариант Assign Selected Actor. Таким образом, вы поменяете назначенный ноду события актер.

Вопрос: Почему эмиттер частиц и звуковые актеры активируются сразу при запуске уровня?

Ответ: Это происходит при включенном свойстве Auto Activate для каждого актера. По умолчанию для этого свойства флашок установлен. Чтобы выключить его, выберите актер и на панели Details уровня в разделе Activate снимите флашок со свойства Auto Activate.

Вопрос: Когда я перекрываю актер-триггер на уровне быстро несколько раз подряд, эмиттер активируется не всегда. Почему?

Ответ: Событие выполняется при каждом перекрытии, но эмиттер частиц имеет предопределенный цикл жизни, который должен закончиться, прежде чем эмиттер активируется вновь.

Вопрос: В чем разница между классами Box Trigger, Sphere Trigger, Capsule Trigger и классом Trigger Volume?

Ответ: Хотя все эти классы могут быть использованы для запуска событий коллизий, основная разница между ними заключается в том, что класс Trigger Volume основан на BSP и может использоваться для определения более сложных зон, нежели простые примитивные формы. Поскольку актеры триггерных зон основаны на BSP, они статичны, что значит, что они не могут двигаться в процессе геймплея.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Истинно или ложно высказывание: классы триггеров можно найти на вкладке Place панели **Modes**.
2. Истинно или ложно высказывание: если актер-триггер не транслирует событие перекрытия назначенному ему ноду события **OnActorBeginOverlap** в блюпринте уровня, это потому что флагок **Simulate Generate Hit Events** не был установлен для актера на уровне.
3. _____ — это объектный элемент актера.
4. Чтобы изменить свойство актера или компонента в функции, вы должны назначить актер или компонент _____ контакту ввода данных функции.
5. Истинно или ложно высказывание: если актер эмиттера начинает производить частицы сразу после перезапуска уровня, это происходит, потому что свойство **Auto Activate** включено для актера.

Ответы

1. Истина. Все классы триггеров находятся на вкладке Place панели **Modes**.
2. Ложь. Настройка **Generate Overlap Events** должна быть включена для работы событий перекрытия, а настройка **Simulate Generate Hit Events** должна быть включена, чтобы работали события ударов.
3. Компоненты — это объектные элементы актеров, и все актеры должны иметь по меньшей мере по одному компоненту.
4. Цель. Свойство **Target** функции указывает ей, на какой актер или компонент функция будет воздействовать при выполнении.
5. Истина. Когда флагок установлен, свойство **Auto Activate** заставляет актер воспроизводиться.

Упражнение

В упражнении к этому часу попрактикуйтесь назначать актеры событиям, добавлять актеры в качестве ссылочных переменных и изменять их свойства. В новом уровне по умолчанию создайте еще три события **OnActorBeginOverlap**, используя

различные типы актеров-триггеров, и измените свойства материалов для каждого из трех различных актеров статичных мешей.

1. В том же проекте, над которым вы работали в этом часе, создайте новый уровень по умолчанию.
2. Поместите три актера-триггера на уровень: Box Trigger, Sphere Trigger и Capsule Trigger.
3. Измените размер каждого актера-триггера и отключите для каждого из них свойство **Actor Hidden in Game** на панели **Details** уровня.
4. Поместите актеры статичных мешей внутрь каждой триггерной зоны на уровне и назначьте уникальный материал каждому из них.
5. Откройте редактор блупринтов уровней и назначьте каждый актер событию **OnActorBeginOverlap**.
6. Добавьте каждый помещенный актер статичного меша как ссылочную переменную в блупринт уровня.
7. Используйте функцию **Set Material** в каждой цепочке событий **OnActorBeginOverlap** для каждого актера статичного меша, чтобы изменить их материалы, когда игрок стреляет в триггерную зону.
8. Используйте функцию **Delay**, чтобы установить ожидание в 1 секунду, и еще одну функцию **Set Material** в каждой цепочке событий **OnActorBeginOverlap**, чтобы материал каждого статичного меша возвращался в изначальный.

16-Й ЧАС

Работа с блюпринт-классами

Что вы узнаете в этом часе

- ▶ Создание простого класса Pickup
- ▶ Добавление и изменение компонентов
- ▶ Работа с нодом Timeline
- ▶ Наследование блюпринт-класса из существующего класса

В этом часе вы познакомитесь с работой с блюпринт-классами. Вы начнете с того, что научитесь наследовать блюпринт-класс из класса Actor и создадите простой класс Pickup, появляющийся и исчезающий, когда персонаж проходит над ним. Затем вы унаследуете блюпринт-класс из класса Point Light и расширите его возможности.

ПРИМЕЧАНИЕ

Подготовка к практике

Прежде чем вы приступите к работе в этом часе, создайте новый проект с шаблоном Third Person и стартовым контентом. Затем создайте две новые папки на панели Content Browser и назовите их *MyBlueprints* и *Maps*. Также создайте новый уровень по умолчанию и сохраните его в папке *Maps*.

Использование блюпринт-классов

Несмотря на то что блюпринты уровней удобны для создания цепочек событий, они связаны с уровнем, над которым вы в данный момент работаете. С другой стороны, блюпринт-классы позволяют скриптовать новые актеры, которые можно многократно использовать на любом уровне. Многократное использование ускоряет разработку, поскольку вам необходимо только один раз разработать функционал блюпринт-класса, после чего вы сможете многократно использовать его на любом уровне. При работе с блюпринт-классами редактор блюпринтов (**Blueprint Editor**)

имеет несколько специальных возможностей для работы с блюпринт-классами. В следующем часе вам будут представлены некоторые из этих отличий.

Добавление блюпринт-класса

На панели Content Browser найдите папку *MyBlueprints*, которую вы создали ранее. Выберите ее и щелкните правой кнопкой мыши по области управления ассетами. В открывшемся контекстном меню выберите вариант **Blueprint Class**, после чего откроется окно **Pick Parent Class**, которое позволяет создать ассет блюпринт-класса. Вверху находится раздел Common Classes, содержащий короткие ссылки на часто используемые типы классов (см. рис. 16.1). Под ним располагается раздел All Classes, перечисляющий все существующие классы, из которых вы можете унаследовать новый блюпринт. Сейчас мы сфокусируемся на создании базового класса Actor, чтобы научиться добавлять и располагать компоненты. Затем вы научитесь создавать новый блюпринт из класса Point Light.

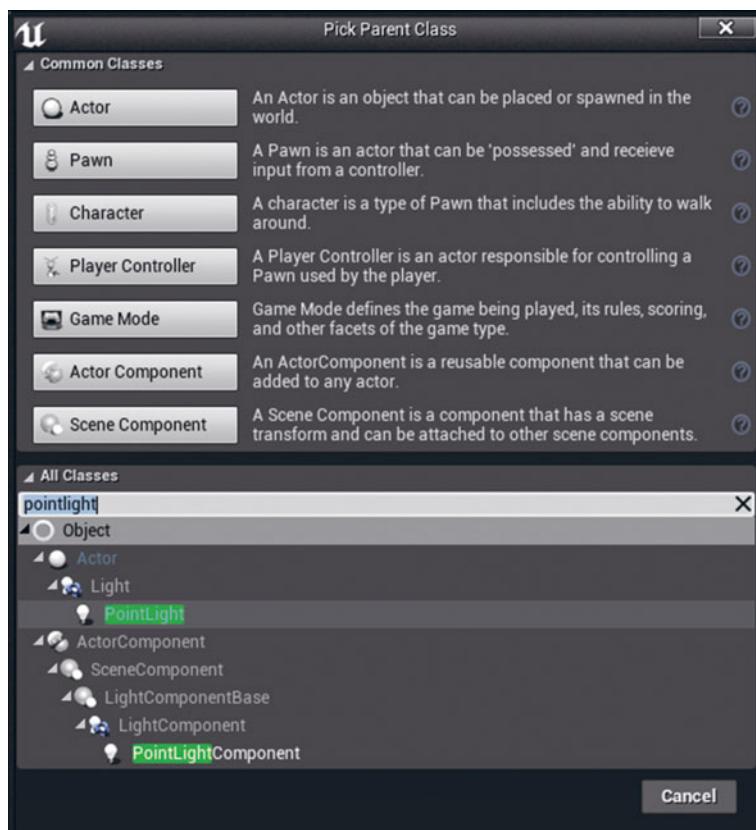


Рис. 16.1. Окно Pick Parent Class

Все блюпринт-классы наследуются из существующих классов, вне зависимости от того, созданы эти классы с помощью C++ или других блюпринт-классов. Новые созданные блюпринт-классы появляются в списке All Classes. Класс Actor, как правило, используется в качестве базового для новых блюпринтов, потому что содержит базовый функционал, необходимый актеру для его помещения и отображения на уровне.

В следующих упражнениях «Попробуйте сами» вы создадите простой класс Pickup, который появляется и исчезает, а также, когда аватар проходит над помещенным актером этого класса, воспроизводит звук, порождает эффект частиц и исчезает. Затем по истечении нескольких секунд актер воспроизводит другой звук, порождающий другой эффект частиц, и появляется меш Pickup, вновь готовый к подбору. Сначала извлечем блюпринт-класс из класса Actor.

ПОПРОБУЙТЕ САМИ

Создайте блюпринт-класс

Выполните следующие шаги, чтобы создать новый блюпринт-класс.

1. В созданной вами ранее папке *MyBlueprints* на панели **Content Browser** щелкните правой кнопкой мыши по пустому пространству в области управления ассетами и выберите вариант **Blueprint Class** в контекстном меню. Откроется окно **Pick Parent Class**.
2. В разделе **Common Classes** окна **Pick Parent Class** щелкните по варианту **Actor** и затем по кнопке **Select** внизу окна, чтобы создать блюпринт-класс. Теперь на панели **Content Browser** есть новый ассет блюпринт-класса.
3. Переименуйте новый ассет в **MyFirstPickup**.
4. Дважды щелкните по **MyFirstPickup**, чтобы открыть редактор блюпринтов.

Интерфейс редактора блюпринтов

Создав блюпринт-класс, взгляните на редактор блюпринтов. Он содержит несколько окон и инструментов, которых нет при работе с блюпринтами уровней. Эти окна обозначены на рис. 16.2 и описаны в следующем списке.

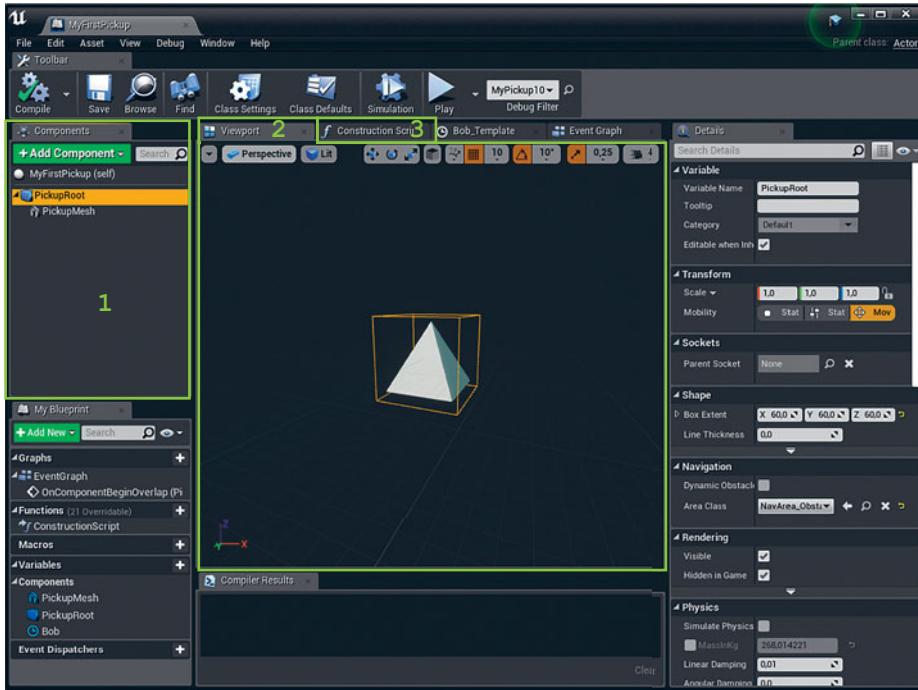


Рис. 16.2. Интерфейс редактора блюпринтов при работе с блюпринт-классом

При работе с блюпринт-классом редактор блюпринтов имеет следующие возможности, недоступные при работе с блюпринтами уровней.

- ▶ **Панель Components.** Панель **Components** содержит список всех компонентов блюпринта и используется для управления ими.
- ▶ **Панель Viewport.** Выюорт отображает компоненты блюпринта и используется для установки пространственных отношений компонента актера.
- ▶ **Construction Script.** Скрипт конструирования — это уникальная функция, которая выполняется, когда экземпляр блюпринта (актер) помещен на уровень. Это граф нодов, который при выполнении изменяет каждый экземпляр независимо от исходного блюпринта.

См. 17-й час «Использование редактируемых переменных и скрипта конструирования», чтобы узнать больше о работе с функцией Construction Script в редакторе блюпринтов.

При работе с блюпринт-классами панель инструментов редактора блюпринтов также имеет дополнительные кнопки для управления блюпринт-классом.

- ▶ **Save.** Сохраняет изменения, произведенные в блюпринте во время работы.
- ▶ **Find in CB.** Находит блюпринт на панели **Content Browser**.
- ▶ **Simulation.** Запускает выполнение блюпринта и отображает результаты во виджете редактора блюпринтов.

См. 14-й час, «Введение в систему визуального программирования блюпринтов» для информации об основных возможностях интерфейса редактора блюпринтов.

Работа с компонентами

Одно из ключевых понятий работы с блюпринт-классами — это компоненты. *Компонент* — это подобъектный элемент блюпринта. Множество различных типов компонентов может быть добавлено в блюпринт, и, в свою очередь, блюпринт может одновременно содержать множество компонентов. Панель **Components** редактора блюпринтов используется для того, чтобы управлять всеми компонентами в блюпринте. Вы можете добавлять, удалять, переименовывать и организовывать компоненты в иерархические отношения, перетаскивая один компонент на другой.

Когда базовый блюпринт-класс создается впервые, он уже имеет компонент **DefaultScene**, назначенный как корневой компонент. Несмотря на то, что блюпринт может содержать много компонентов, корневой компонент может быть только один. Корневой компонент блюпринта — это единственный компонент, имеющий ограничение трансформации. Он является родительским по отношению ко всем остальным компонентам в блюпринте. Это единственный компонент, который нельзя переместить или повернуть, но его можно масштабировать. Местоположение и угол поворота корневого компонента определяются однажды, когда актер помещается на уровень. Все другие трансформации компонента взаимосвязаны с корневым компонентом по умолчанию. Практически любой тип компонентов может быть назначен в качестве корневого компонента.

Добавление компонентов

Панель **Components** позволяет добавлять компоненты в актер двумя различными способами. Если вы щелкнете по зеленой кнопке **+Add Components**, то увидите обширный список организованных в категории компонентов, которые можно добавлять в блюпринт. Чтобы добавить компонент из панели **Content Browser**, щелкните по ассету и перетащите его из панели **Content Browser** на панель **Components** в редакторе блюпринтов. Если для этого ассета существует тип компонента, он автоматически будет добавлен в блюпринт. Вы легко можете выполнить эту операцию для статичных мешей, систем частиц и аудио-ассетов.

При добавлении в блюпринт первого компонента, его родителем станет корневой компонент DefaultScene. Перетаскивая один компонент на другой, вы можете прикреплять компоненты, делая один родительским, а другой дочерним. Оба они по-прежнему будут подобъектными элементами актера, но трансформации дочернего компонента связаны с его родительским компонентом, а трансформации родительского — с корневым, чьи трансформации полностью определены положением актера в мире (на уровне).

Когда компонент добавлен в блюпринт, вы можете редактировать его свойства на панели **Details** редактора блюпринтов.

Многие типы компонентов имеют свойства, которые могут показаться знакомыми, поскольку они аналогичны свойствам уже рассмотренных нами актеров. Например, на уровне вы можете поместить актер статичного меша, но при работе с блюпринт-классом вы используете компонент Static Mesh. Оба имеют одинаковые свойства, используемые для изменения статичного меша, но компонент — это подобъектный элемент в блюпринте.

ПРИМЕЧАНИЕ

Специальные компоненты

Некоторые компоненты, такие как компоненты перемещения, ведут себя иначе, чем стандартные компоненты. Компоненты перемещения влияют на весь актер в целом. Они также не имеют физического воплощения во виджете блюпринта.

Панель **Viewport**

Панель **Viewport** позволяет увидеть пространственные взаимоотношения всех компонентов, добавленных в актер. Как и во виджете уровня, вы можете использовать виджеты трансформации для изменения местоположения, угла поворота и масштаба каждого компонента в блюпринте. Выберите компонент на панели **Components** или во виджете и используйте клавишу пробела для переключения между виджетами трансформации. Также во виджете существуют настройки привязки для всех трансформаций, которые вы можете включать, отключать или настраивать, чтобы упростить расположение каждого компонента.

СОВЕТ

Типы положения: **Relative** и **World**

По умолчанию добавленные трансформации компонента связаны с родительским компонентом и, в конечном счете, с корневым компонентом актера. Вы можете по отдельности изменить местоположение, угол поворота или масштаб. Выберите компонент на панели **Details** редактора блюпринтов и в категории **Transforms** щелкните

по треугольнику справа от параметров **Location**, **Rotation** или **Scale**, чтобы изменить положение на **Relative** или **World**.

ПОПРОБУЙТЕ САМИ

Добавьте компоненты

После создания блупринт-класса, необходимо добавить в него компоненты. Выполните следующие шаги, чтобы добавить компоненты Box Collision и Static Mesh, и сделайте компонент Static Mesh дочерним по отношению к Box Collision.

1. Щелкните по зеленой кнопке **+Add Component** вверху панели **Components** и выберите вариант **Box Collision** в выпадающем списке, чтобы добавить этот компонент в блупринт.
2. Щелкните правой кнопкой мыши по новому компоненту **Box Collision** и переименуйте его в **PickupRoot**.
3. Щелкните по компоненту **PickupRoot** и перетащите его на компонент **DefaultSceneRoot**, чтобы поменять их местами.
4. В папке *Starter Content* на панели **Content Browser** найдите пирамидальный ассет статичного меша **Shap_Quad** и перетащите его на панель **Components** в редакторе блупринтов. Вы добавили компонент Static Mesh в блупринт, который ссылается на ассет статичного меша — пирамиду.
5. Щелкните правой кнопкой мыши по ассету **Shap_Quad** на панели **Components** и переименуйте его в **PickupMesh**.
6. Щелкните по панели **Viewport** в редакторе блупринтов, чтобы увидеть компоненты.
7. Выберите компонент Box Collision на панели **Components** или на панели **Viewport**. Затем на панели **Details** в разделе Shape установите свойство Box Extent в значение **60** для категорий X, Y и Z.
8. Выберите компонент Static Mesh во вьюпорте и используйте виджет трансформации перемещения, чтобы переместить его внутрь прямоугольной коллизии.
9. Скомпилируйте и сохраните блупринт, щелкнув по кнопке **Compile** и затем по кнопке **Save** на панели инструментов редактора блупринтов. Когда вы закончите, ваш блупринт должен выглядеть, как на рис. 16.3.
10. На панели инструментов редактора блупринтов щелкните по кнопке **Find in CB** и найдите ваш блупринт-класс **MyFirstPickup** на панели **Content Browser**.
11. Щелкните по экземпляру блупринт-класса **MyFirstPickup** на панели **Content Browser** и перетащите его на уровень.

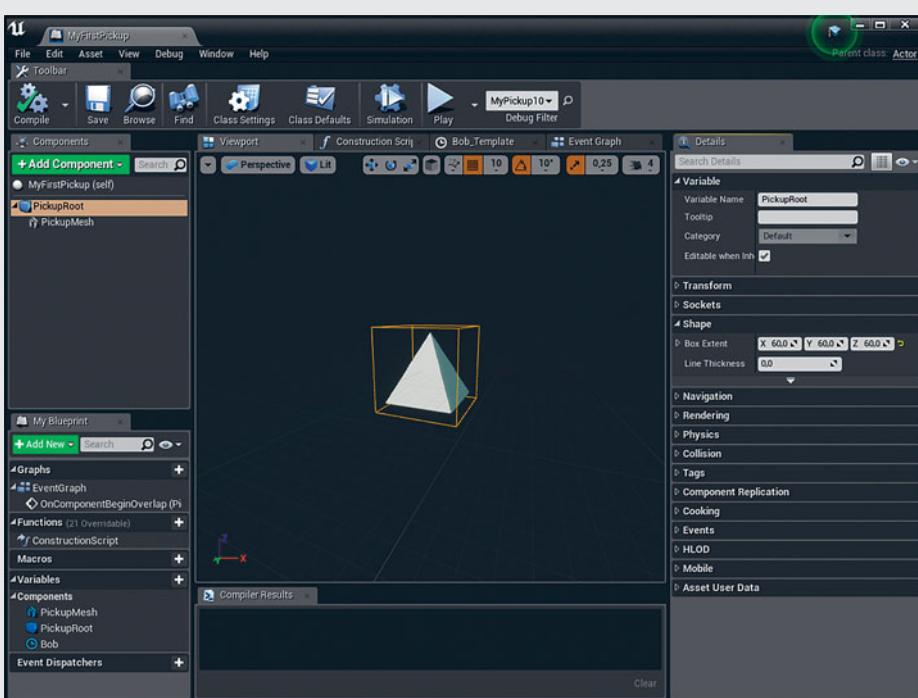


Рис. 16.3. Панели Components и Viewport в редакторе блюпринтов

Программирование блюпринтов с использованием компонентов

Когда вы производите скриптинг блюпринт-класса, у вас есть функции, имеющие своей целью актер, и функции, целью которых являются отдельные компоненты актера, но, поскольку вы работаете внутри актера, вы видите и используете термин *self*, когда ссылаетесь на актер. Поэтому, если вам необходимо использовать функцию, изменяющую весь актер, целью этой функции будет *self*. Это также повлияет на любые события, которые вы можете использовать. Например, у вас есть событие, назначенное актеру, или событие, назначенное отдельному компоненту актера. Предположим, что существует событие коллизии ActorBeginOverlap, назначенное актерам, и событие коллизии OnComponentBeginOverlap, назначенное компонентам актера.

Вы можете добавить любой компонент на панели **Components** в граф событий блюпринт-класса, как ссылочную переменную компонента, щелкнув по компоненту и перетащив его в граф событий. Кроме того, каждый компонент, который

вы добавляете в блюпринт, отображается в разделе Variables панели **My Blueprint**. Когда компонент добавлен как ссылочная переменная компонента, вы можете использовать его в качестве цели для функций, которые могут изменять свойства или поведение компонента.

В следующей рубрике «Попробуйте сами» вы будете скриптовать главный функционал класса Pickup так, чтобы игрок мог пройти над актером и сделать его невидимым.

ПОПРОБУЙТЕ САМИ

Произведите скрипting функционала простого класса Pickup

После добавления компонентов в актер, следующим шагом будет создание события перекрытия компонента, как описано ниже.

1. На панели **Components** выберите вариант **PickupRoot(Box Collision)**. Затем в графе событий щелкните правой кнопкой мыши, чтобы вызвать контекстное меню блюпринта. В поисковой строке контекстного меню введите **on component begin overlap** и выберите событие **OnBeginComponentOverlap**, чтобы добавить его в граф событий.
2. Щелкните по ехес-контакту вывода нода **OnBeginComponentOverlap**, перетащите провод и отпустите кнопку мыши, чтобы вызвать контекстное меню. Найдите потоковый нод **DoOnce** и добавьте его в граф.
3. Щелкните по завершенному ехес-контакту вывода нода **DoOnce**, перетащите провод и отпустите кнопку мыши, чтобы вызвать контекстное меню. Найдите функцию меша **Set Hidden** и добавьте ее. Произойдет добавление нода **Set Hidden in Game** и ссылочной переменной компонента, которая ссылается на компонент меша Pickup. Установите флагок рядом с параметром **New Hidden Data Pin**, чтобы установить значение истины.
4. Щелкните по ехес-контакту вывода нода функции **Set Hidden in Game**, перетащите провод и отпустите кнопку мыши, чтобы добавить функцию **Play Sound at Location**. Назначьте ассет звуковой волны или звукового сигнала (один, чтобы избежать повторов) контакту **Sound Data**, щелкнув по выпадающему списку рядом с ним.
5. Щелкните по ехес-контакту вывода нода функции **Play Sound at Location**, перетащите провод и отпустите кнопку мыши, чтобы добавить нод функции **Spawn Emitter at Location**. Назначьте систему частиц контакту **Emitter Template**, щелкнув по выпадающему списку рядом с контактом.
6. Чтобы установить местоположение для **Play Sound at Location** и **Spawn Emitter at Location** через панель **Components**, перетащите **PickupRoot(Box Collision)** в граф событий, добавив тем самым ссылочную переменную компонента.
7. Щелкните по синему контакту вывода данных на помещенной ссылочной переменной компонента и добавьте **GetWorldLocation (PickupRoot)**. Соедините

контакт вывода данных Return Value Vector с контактами ввода данных Location Vector в нодах функций Play Sound at Location и Spawn Emitter at Location.

8. Чтобы добавить задержку появления класса Pickup, перетащите и отпустите указатель мыши и добавьте под функции Delay. Установите параметр **Duration** в значение **3** (сек.).
9. Скопируйте и вставьте все ноды из нода функции Set Hidden in Game в ноду функции Spawn Emitter at Location. Щелкните и перетащите курсор мыши по пустому пространству в графе событий, чтобы выделить ноды. Затем нажмите комбинацию клавиш **Ctrl+C**, чтобы скопировать, и **Ctrl+V**, чтобы вставить дупликаты выбранных нодов в графе событий.
10. Переместите вставленные ноды так, чтобы они располагались после функции Delay, и соедините ехес-контакт вывода Delay с вставленной функцией New Hidden in Game.
11. Соедините ехес-контакт вывода Pasted Spawn Emitter at Location с ехес-контактом ввода функции DoOnce так же, как в начале последовательности.
12. Когда вы закончите, ваша последовательность блюпринта должна выглядеть, как на рис. 16.4. На панели инструментов блюпринта щелкните по кнопке **Compile** и затем по кнопке **Save**.
13. Поместите несколько копий блюпринта **MyFirstPickup** из панели **Content Browser** на уровень.
14. Запустите уровень и пройдите над актером Pickup.

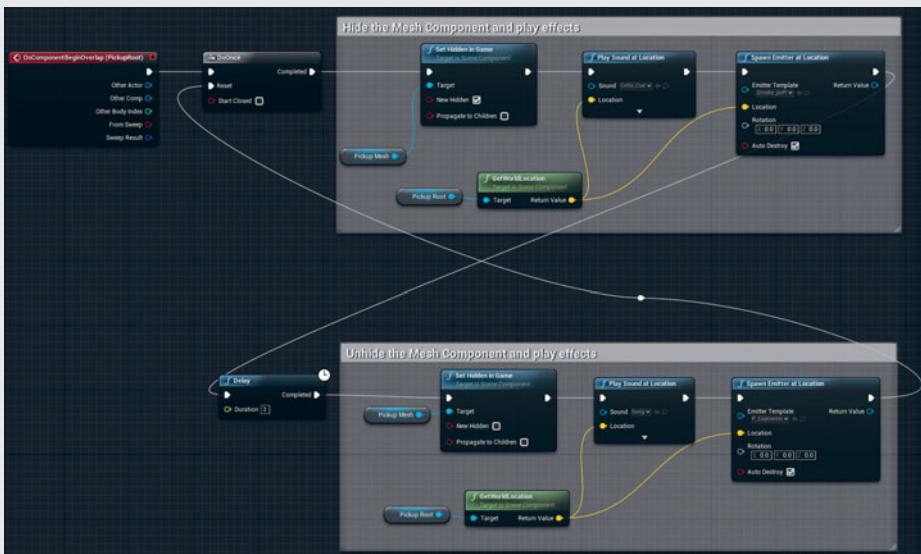


Рис. 16.4. Последовательность события Pickup

СОВЕТ**Коммуникация блюпринтов**

Актеры являются изолированными, то есть они владеют информацией о себе и своих компонентах. Однако они не имеют данных о других актерах на уровне до тех пор, пока не сработает событие, такое как **OnActorBeginOverlap** или **OnComponentBeginOverlap**, которое вернет актер, запустивший событие. В связи с этим существует несколько путей, таких как использование диспетчера событий, блюпринт-интерфейсы и приведение типов. Вы можете более подробно изучить эти понятия, когда ближе познакомитесь со скрипtingом базовых блюпринт-классов, по адресу: docs.unrealengine.com/latest/INT/Engine/Blueprints/UserGuide/BlueprintCommsUsage/index.html.

Работа с нодом Timeline

Нод **Timeline** позволяет создавать данные кривых сплайнов, которые могут использоваться в блюпринте для изменения значений с течением времени. Это можно использовать, чтобы анимировать положение актера и/или его компонентов или чтобы изменять интенсивность компонента источника света. Чтобы добавить нод **Timeline** в граф событий, щелкните правой кнопкой мыши по графу событий, чтобы вызвать контекстное меню блюпринта, и в поисковой строке введите **Timeline**. Выберите вариант **Add Timeline** из списка, чтобы поместить нод **Timeline** в граф. Вы можете помещать в блюпринт столько нодов **Timeline**, сколько потребуется, и хорошим приемом будет давать им описательные имена. Чтобы переименовать нод **Timeline**, щелкните по нему правой кнопкой мыши, выберите вариант **Rename** и введите новое имя. После того как нод **Timeline** был добавлен в граф событий, он также будет отображаться на панели **My Blueprint** в разделе **Variables** (или **Components**), и вы сможете добавлять ссылку переменной в **Timeline** в других последовательностях в вашем блюпринте. Как только у вас появится ссылка переменной на нод **Timeline**, вы сможете изменять ее свойства в блюпринте.



Рис. 16.5. Нод **Timeline**

Нод Timeline имеет множество ехес-контактов ввода для воспроизведения, паузы и перемотки Timeline. Он имеет ехес-контакт вывода Update для запуска последовательности во время воспроизведения Timeline и ехес-контакт вывода Finished, который выполняется по окончании работы Timeline (см. рис. 16.5). Если нод Timeline зациклен, ехес-контакт вывода Finished не запускается.

Ноды Timeline имеют собственное окно редактора, которое автоматически открывается, если дважды щелкнуть по ноду. В редакторе **Timeline (Timeline Editor)** вы увидите панель инструментов для настройки свойств Timeline и изменения их функционирования. Вы можете установить продолжительность Timeline в секундах. Вы можете настроить нод таким образом, что его воспроизведение начнется автоматически в начале игры, а также зациклить его (см. рис. 16.6).

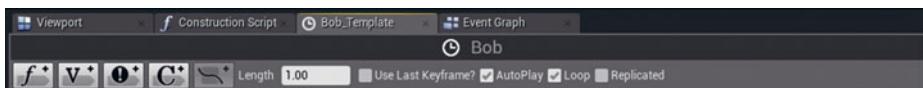


Рис. 16.6. Панель инструментов редактора Timeline

Треки и кривые Timeline

В редакторе **Timeline** вы увидите кнопки для добавления различных видов треков. Существует четыре вида треков для работы в **Timeline**: трек Float, Vector, Color и трек Event. Каждый вид треков используется для редактирования данных кривых с установленными ключами и создает контакт вывода данных в ноде Timeline, который возвращает значение указанного типа переменной на всем временном отрезке воспроизведения трека нодом Timeline. Трек Event, однако, добавляет ехес-контакт вывода в нод Particle System, который запускается в установленное время в зависимости от места расположения ключа.

Нод Timeline может иметь множество назначенных ему треков. Чтобы добавить трек, просто щелкните по виду трека, который хотите добавить, и нод добавит его. Вы можете переименовать трек, щелкнув по его заголовку и введя новое имя. При переименовании трека, соответствующий контакт данных в ноде Timeline обновится, отражая новое имя. По мере добавления новых треков в Timeline создаются новые ехес-контакты вывода и контакты данных в зависимости от видов добавляемых треков (см. рис. 16.7).

Как только новый трек добавлен и переименован, вы можете начать добавлять и редактировать ключи на кривых трека, просто зажимая клавишу **Shift** и щелкая по кривой. Вы можете переместить уже помещенный ключ вручную, щелкнув и перетащив его. Когда помещенный ключ выбран, вы можете ввести точное время и значение вверху трека.

В следующей рубрике «Попробуйте сами» вы будете использовать Timeline, чтобы анимировать компонент меша актера Pickup, заставив его последовательно появляться и исчезать при запуске уровне.

ПОПРОБУЙТЕ САМИ

Установите Timeline

Выполните следующие шаги, чтобы добавить под **Timeline** в граф событий и отредактировать трек **Float**, который будет использоваться для анимирования компонента статичного меша Pickup.

1. Добавьте под Particle System в блюпринт, если у вас его еще нет.
 2. Переименуйте его в **PickupAnim**.
 3. Дважды щелкните по ноду Particle System, чтобы открыть редактор **Timeline**.
 4. Установите продолжительность **Timeline** в значение **1** (сек).
 5. Установите флажок **Auto Play**.
 6. Установите флажок **Loop**.
 7. Добавьте трек **Float** в Particle System, щелкнув по кнопке **f+** в редакторе **Timeline**.
 8. Щелкните правой кнопкой мыши по заголовку **NewTrack_1** в левом верхнем углу трека и переименуйте трек в **bounce**.
 9. Отредактируйте кривую в треке. Чтобы добавить ключ, зажмите клавишу **Shift** и щелкните по кривой в треке. Добавьте ключ на временной точке **0** сек со значением **0**.
 10. Добавьте второй ключ на кривую на временной точке **0,5** сек со значением **1**. Если кривая находится снаружи трека, используйте кнопки со стрелками влево и вправо, а также вверх и вниз, рядом со строкой ввода временной точки вверху трека, чтобы настроить внешний вид трека.
 11. Добавьте последний ключ на временной точке **1** сек со значением **0**.
 12. Чтобы изменить интерполяцию каждого ключа в значение **Auto**, щелкните правой кнопкой мыши по каждому ключу и выберите вариант **Auto** из списка.
- Когда вы закончите, ваша кривая должна выглядеть, как на рис. 16.7.



Рис. 16.7. Нод Timeline с контактом данных с плавающей запятой (слева) и треком кривой Float (справа)

Когда вы настроите Timeline, следующим шагом будет использовать данные кривой Float, чтобы переместить компонент статичного меша Pickup. Вы будете делать это снова в графе событий блюпринта. Во время воспроизведения нода Particle System он возвращает значения от 0 до 1 за 1 секунду, поэтому вы должны умножить числовые значения на расстояние, которое актер Pickup должен преодолеть. Затем вы можете применить результат к оси Z компонента, чтобы переместить его вверх и вниз.

ПОПРОБУЙТЕ САМИ

Анимируйте актер Pickup с помощью Timeline

Выполните следующие шаги, чтобы анимировать компонент меша блюпринта с помощью Timeline, заставляя его быстро прыгать вверх и вниз.

1. Щелкните по компоненту **PickupMesh** и перетащите его из панели **Components** в граф событий, чтобы добавить его в качестве ссылочной переменной.
2. Щелкните по синему контакту данных ссылочной переменной **PickupMesh** и перетащите его, чтобы открыть контекстное меню. В поисковой строке введите **set relative location** и выберите вариант **SetRelativeLocation**, чтобы добавить функцию в граф событий.
3. Соедините ехес-контакт **Update** нода Particle System с ехес-контактом ввода нода SetRelativeLocation.
4. Щелкните по зеленому контакту данных с плавающей запятой нода Particle System и перетащите его, чтобы открыть контекстное меню. В поисковой строке введите **multiply** и выберите вариант **Float * Float**, чтобы добавить его как нод.
5. В текстовой строке рядом со вторым контактом данных с плавающей запятой нода Multiply введите **10**, чтобы обозначить дистанцию, на которую компонент будет перемещаться.
6. В ноде SetRelativeLocation, добавленном в шаге 2, щелкните правой кнопкой мыши по желтому контакту данных **Vector** и выберите вариант **Split Struct Pin** из списка, чтобы расщепить вектор на три зеленых контакта данных с плавающей запятой для указания положения по осям X, Y и Z.
7. Соедините зеленый контакт данных нода Multiply с зеленым контактом ввода для чисел с плавающей запятой **New Location Z** нода SetRelativeLocation.
8. Скомпилируйте и сохраните блюпринт, щелкнув по кнопке **Compile** и затем по кнопке **Save** на панели инструментов редактора блюпринтов. Когда вы закончите, ваша последовательность блюпринта должна выглядеть, как на рис. 16.8.
9. Запустите уровень. Ваши актеры Pickup должны теперь непрерывно двигаться вверх и вниз.

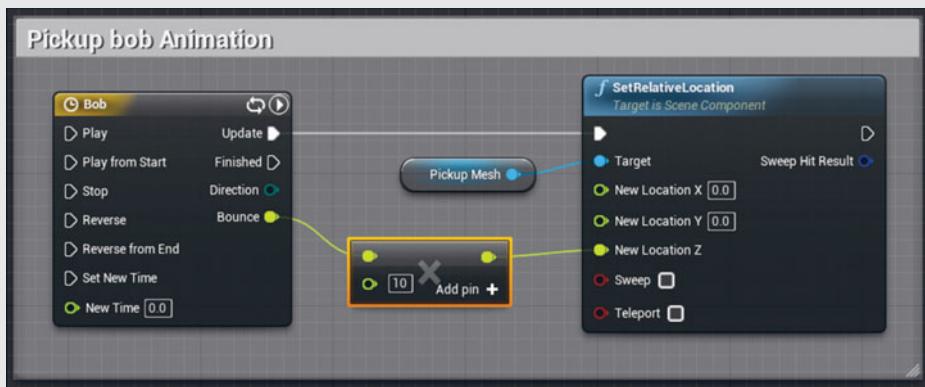


Рис. 16.8. Анимационная последовательность, созданная с помощью нода Particle System

Скрипting пульсирующего источника света

Теперь, когда у вас есть некоторый опыт создания блупринт-класса и работы с компонентами, во второй половине этого часа вы приступите к обучению создания блупринт-класса, который расширяет функционал существующего класса.

В течение последующих нескольких упражнений «Попробуйте сами» вы создадите блупринт пульсирующего источника света, унаследованный из существующего класса, который в случайном порядке генерирует интенсивность свечения в пределах указанного диапазона и затем изменяет интенсивность свечения точечно-го ИС с течением времени, чтобы соответствовать новому значению. Каждый раз, когда интенсивность свечения будет соответствовать новому значению, блупринт в случайном порядке генерирует новое значение интенсивности. Блюпринт будет продолжать этот процесс на всем протяжении игры, поэтому потребуется событие Event Tick, чтобы последовательность работала непрерывно.

Наследование блупринта из существующего класса

В этом разделе вы впервые создадите новый блупринт-класс, унаследованный из класса Point Light (см. рис. 16.9). Этот новый класс будет наследовать свойства и компоненты родительского класса. Затем вы сможете использовать редактор блупринтов, чтобы создать новый функционал.

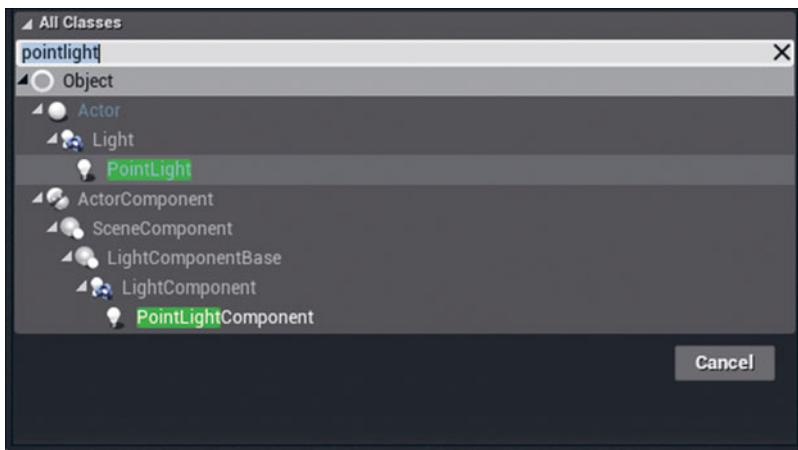


Рис. 16.9. Раздел All Classes окна Pick Parent Class

ПОПРОБУЙТЕ САМИ

Извлеките блюпринт-класс из класса Point Light

Выполните следующие шаги, чтобы создать новый актер блюпринта из класса Point Light.

- На панели **Content Browser** в папке *MyBlueprints*, которую вы создали ранее, щелкните правой кнопкой мыши по пустому пространству области управления ассетами и выберите вариант **Blueprint Class** в диалоговом окне. Откроется окно **Pick Parent Class**.
- В разделе **All Classes** введите **pointlight** и выберите вариант **Point Light**. Щелкните по кнопке **Select** внизу окна **Pick Parent Class**.
- На панели **Content Browser** переименуйте новый ассет в **MyPulseLight_BP**.
- Дважды щелкните по **MyPulseLight_BP**, чтобы открыть редактор блюпринтов.
- Перетащите ассет **MyPulseLight_BP** из панели **Content Browser** на уровень и взгляните на его свойства на панели **Details** уровня. Вы увидите, что актер имеет те же свойства, что и обычный актер точечного ИС.

Теперь вы унаследовали новый блюпринт-класс. Следующим шагом будет создание необходимых типов переменных.

ПОПРОБУЙТЕ САМИ

Установите переменную

Выполните следующие шаги, чтобы создать переменную, необходимую вам для блюпринта.

1. Откройте ассет *MyPulseLight_BP* из предыдущего упражнения «Попробуйте сами», дважды щелкнув по нему на панели **Content Browser**.
2. На панели **My Blueprint** в разделе **Variables** щелкните по символу +, чтобы добавить новую переменную.
3. Выбрав новую переменную на панели **Details**, дайте ей имя **Target_Intensity** и установите тип переменной **Float**.
4. Повторите шаги 2 и 3 еще три раза, чтобы добавить переменные с плавающей запятой **Max_Intensity**, **Min_Intensity** и **Pulse_Rate** в ваш скрипт. Когда вы закончите, щелкните по кнопке **Compile** на панели инструментов редактора блюпринтов.
5. Чтобы установить значения по умолчанию для некоторых новых переменных, установите на панели **Details** переменную с плавающей запятой **Max_Intensity** в значение **10000**, а переменную с плавающей запятой **Pulse_Rate** в значение **100000**.

ПРИМЕЧАНИЕ

Компонент Point Light

Нет необходимости добавлять компонент Point Light в актер, он уже есть в блюпринте, поскольку тот унаследован из класса Point Light.

Переменная с плавающей запятой **Max_Intensity** хранит в себе значение максимальной интенсивности света, а переменная с плавающей запятой **Min_Intensity** хранит наименьшее значение интенсивности, в данном случае, 0, то есть без интенсивности. Переменная с плавающей запятой **Pulse_Rate** используется для установления скорости, с которой интенсивность будет меняться от одного значения к другому. Установив необходимые переменные, вы можете начать скриптинг. Следующим шагом будет случайная генерация нового значения интенсивности и сохранение его в переменной с плавающей запятой **Target_Intensity**, чтобы его можно было использовать для смены интенсивности ИС в компоненте Point Light.

ПОПРОБУЙТЕ САМИ

Сгенерируйте случайное значение интенсивности и установите интенсивность ИС

Выполните следующие шаги, чтобы создать код, генерирующий случайное числовое значение в пределах максимального и минимального значений интенсивности, сохраните его в целевом значении и используйте для того, чтобы установить интенсивность компонента ИС.

1. Перетащите переменную с плавающей запятой Target_Intensity из раздела **Variables** панели **My Blueprint** в граф событий и выберите вариант **Set**.
2. Временно соедините exec-контакт вывода события Event Tick с exec-контактом ввода нода Set Target_Intensity.
3. Щелкните по зеленому контакту ввода данных Set и перетащите его. В поисковой строке появившегося контекстного меню введите **random float in range** и добавьте **Random Float in Range** в граф событий.
4. Перетащите переменную с плавающей запятой Min_Intensity из раздела **Variables** панели **My Blueprint** в граф событий и выберите вариант **Get**. Соедините ее контакт вывода данных с контактом ввода минимальных данных функции Random Float in Range.
5. Перетащите переменную с плавающей запятой Max_Intensity из раздела **Variables** панели **My Blueprint** в граф событий и выберите вариант **Get**. Соедините ее контакт вывода данных с контактом ввода максимальных данных функции Random Float in Range.
6. Щелкните по компоненту **PointLightComponent(Inherited)** и перетащите его из панели **Components** в граф событий, чтобы добавить ссылочную переменную компонента, которая ссылается на компонент Point Light.
7. Щелкните по контакту данных компонента PointLightComponent(Inherited) и перетащите его, в поисковой строке открывшегося контекстного меню введите **set intensity**. Выберите вариант **Set Intensity** из списка, чтобы добавить функцию.
8. Соедините exec-контакт вывода нода Set Target_Intensity с exec-контактом ввода нода Set Intensity.
9. Перетащите переменную с плавающей запятой Target_Intensity из раздела **Variables** панели **My Blueprint** в граф событий. Выберите вариант **Get**, чтобы поместить ссылку переменной с плавающей запятой на целевую интенсивность и соединить ее с новым значением интенсивности функции Set Intensity. Когда вы закончите, ваша последовательность блокпринта должна выглядеть, как на рис. 16.10.
10. Скомпилируйте скрипт и убедитесь, что экземпляр актера блокпринта помещен на уровень. Запустите уровень. Новый актер точечного ИС должен быстро мерцать.

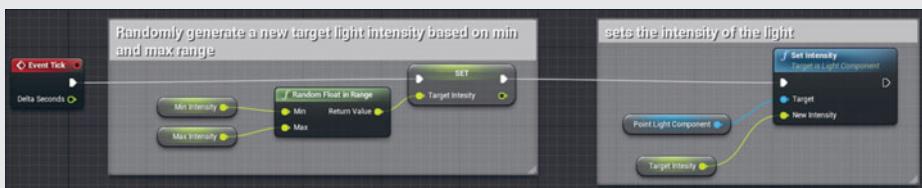


Рис. 16.10. Цепочка событий Event Tick, случайно устанавливающая интенсивность компонента ИС

ПРЕДУПРЕЖДЕНИЕ

Событие Event Tick

Событие Event Tick по умолчанию выполняется с отображением каждого кадра на всей продолжительности игры. Это значит, что оно постоянно запущено. По этой причине оно потенциально может оказать влияние на производительность.

В настоящий момент источник света генерирует целевое значение интенсивности в пределах диапазона, определенного переменными с плавающей запятой Min_Intensity и Max_Intensity, и устанавливает интенсивность в новом целевом значении. Следующим шагом необходимо создать мягкий переход между сменой текущего значения на целевое. Это можно сделать с помощью функции FInterp to Constant. Также необходимо проверить, равняется ли текущее значение интенсивности целевому, генерируется ли новое целевое значение, после чего необходимо повторно интерполировать функцию.

ПОПРОБУЙТЕ САМИ

Сделайте пульсацию света мягкой

Продолжая предыдущее упражнение «Попробуйте сами», выполните следующие шаги, заставьте свет плавно переходить от текущей интенсивности к целевой.

- Щелкните по компоненту **PointLightComponent(Inherited)** и перетащите его из панели **Components** в граф событий, чтобы добавить ссылочную переменную компонента, которая ссылается на компонент Point Light.
- Щелкните по контакту вывода данных ссылочной переменной PointLightComponent(Inherited) и перетащите его. В поисковой строке контекстного меню введите **get intensity**. Выберите вариант **Get Intensity** из списка, чтобы добавить нод в граф событий.
- Щелкните по зеленому контакту вывода данных с плавающей запятой нода переменной **Intensity** и перетащите его. В поисковой строке контекстного меню введите **finterp to constant** и выберите вариант **FInterp to Constant** из списка, чтобы добавить нод в граф событий.
- Заполните контакт данных функции **FInterp to Constant**. Зажмите клавишу **Alt** и щелкните по нему, чтобы разорвать связь между помещенными переменной с плавающей запятой **Target_Intensity** и функцией **Set Intensity**, добавленными в предыдущем упражнении «Попробуйте сами», и соедините **Target_Intensity** с целевым контактом ввода данных **FInterp to Constant**.
- Перетащите контакт вывода данных **Delta Seconds** события Event Tick в контакт ввода данных **Delta Time** нода **FInterp to Constant**.
- Перетащите переменную с плавающей запятой **Pulse_Rate** из раздела **Variables** на панели **My Blueprint** и добавьте ее в граф событий. Соедините ее с контактом данных **Interp Speed** функции **FInterp to Constant**.

7. Соедините контакт данных возврата значений функции FInterp to Constant с контактом ввода данных нового значения интенсивности функции Set Intensity.
8. На панели инструментов редактора блупринтов щелкните по кнопке **Compile**, затем по кнопке **Save**. Когда вы закончите, ваша последовательность блуприфта должна выглядеть, как на рис. 16.11.
9. Запустите уровень. Вы увидите, как источник света пульсирует и мерцает.

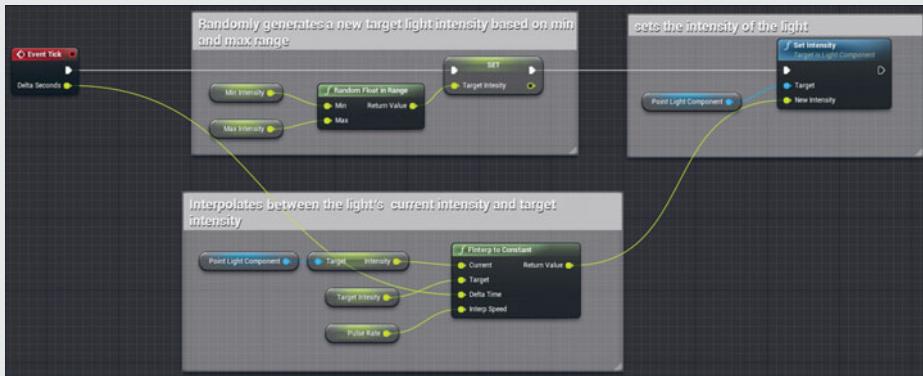


Рис. 16.11. Цепочка событий пульсирующего источника света

Свет пульсирует, но наш скрипт еще не закончен. Хотя свет мигает, событие Event Tick продолжает генерировать новое целевое значение в каждом цикле, и источник света никогда не достигнет целевого значения интенсивности. Вы можете исправить это в следующей рубрике «Попробуйте сами».

ПОПРОБУЙТЕ САМИ

Сравните текущую интенсивность компонента Point Light

Скрипт должен генерировать новое целевое значение интенсивности, когда текущая интенсивность компонента ИС равна целевой интенсивности. В завершающей части этого скрипта вам необходимо сравнить текущее значение интенсивности ИС с новым целевым значением, и если они равны, то продолжить процесс. Выполните следующие шаги.

1. Щелкните по переменной **PointLightComponent(Inherited)** и перетащите ее из панели **Components** в граф событий, чтобы добавить новую ссылочную переменную компонента, которая ссылается на компонент Point Light.
2. Щелкните по контакту вывода данных ссылочной переменной **PointLightComponent(Inherited)** и перетащите его. В поисковой строке

контекстного меню введите **get intensity**. Выберите вариант **Get Intensity** из списка, чтобы добавить нод.

3. Щелкните по зеленому контакту данных с плавающей запятой и перетащите его из нода **PointLightComponent(Inherited)**. В поисковой строке контекстного меню введите **equals**. Выберите вариант **Equals (float)** из списка, чтобы добавить его. Этот нод сравнивает два числовых значения и возвращает **1** (истина) в случае их равенства либо **0** (ложь), если они не равны.
4. Из раздела **Variables** панели **My Blueprint** перетащите переменную **Target_Intensity** и добавьте ее. Соедините его со вторым контактом ввода данных с плавающей запятой нода **Equals**.
5. Щелкните по красному контакту вывода данных нода **Equals** и перетащите его. В поисковой строке контекстного меню введите **branch**, чтобы добавить нод **Branch**, который изменяет поток последовательности в зависимости от состояния булевой переменной. Если булево значение истинно, нод посылает сигнал через **exec**-контакт вывода **True**, если ложно — через **exec**-контакт вывода **False**.
6. Соедините **exec**-контакт вывода события **Event Tick** с **exec**-контактом ввода нода **Branch**.
7. Соедините **exec**-контакт вывода **True** нода **Branch** с **exec**-контактом ввода функции **Set Target Intensity**.
8. Соедините **exec**-контакт вывода **False** нода **Branch** с **exec**-контактом ввода функции **Set Intensity**.
9. Установите значение по умолчанию переменной с плавающей запятой **Pulse_Rate** равное **5000**.
10. На панели инструментов редактора блупринтов щелкните по кнопке **Compile**, затем по кнопке **Save**. Когда вы закончите, ваша последовательность блупримита должна выглядеть, как на рис. 16.12.
11. Запустите уровень. Свет по-прежнему мерцает, но менее хаотично.

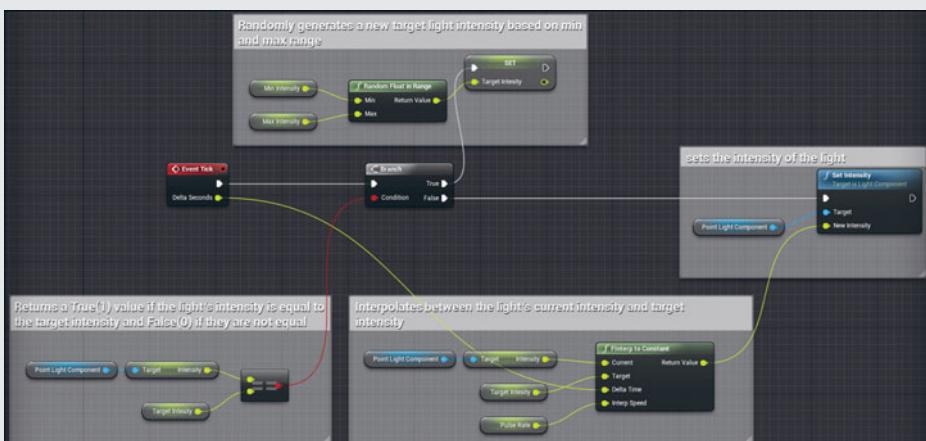


Рис. 16.12. Сравнение интенсивности точечного ИС

Теперь, когда блюпринт запущен, он первым делом сравнивает текущее значение интенсивности компонента Point Light с целевым, и, если они не равны, продолжает устанавливать интенсивность на основе ответа функции FInterp to Constant. Как только два значения будут равны, он случайным образом генерирует новое значение целевой интенсивности в пределах минимального и максимального показателей, в результате чего нод Equals вновь возвращает ложь, и функция Set Intensity запускается еще раз.

Резюме

В этом часе вы научились создавать блюпринт-классы и использовать редактор блюпринтов. Вы создали два блюпринт-класса: один унаследовали из класса Actor, а другой — из класса Point Light. Вы запрограммировали простой актер Pickup и актер пульсирующего точечного ИС. Вы научились использовать нод Timeline, чтобы анимировать компонент Static Mesh. Блюпринт-классы используются для создания практически всех игровых элементов, которые могут вам понадобиться. Хотя скриптинг игрового функционала в блюпринте уровня предоставляет много возможностей, блюпринт-классы являются более мощными, поскольку пригодны для многократного использования во всем проекте. Чем лучше вы усвоите работу с блюпринт-классами, тем более сложные задачи сможете выполнять при работе над собственными играми.

Вопросы и ответы

Вопрос: Блюпринт актера Pickup анимирован, но анимация выглядит механической. Почему?

Ответ: Как и в случае с ключами в кривых сплайнов в редакторе Matinee, вы можете устанавливать типы кривых. Щелкните правой кнопкой мыши по ключу и установите его в значение Auto, чтобы создать плавный переход от ключа.

Вопрос: Когда я стреляю в актер Pickup пулей, пуля отскакивает. Почему?

Ответ: По умолчанию компонент прямоугольной коллизии класса Pickup имеет настройку коллизии на блокирование пуль. В редакторе блюпринтов выберите Box Trigger на панели Components и на панели Details установите пресеты коллизии в значение Custom. Затем установите параметр Collision Response в значение Ignore, чтобы актер игнорировал все, и установите параметр Pawn в значение Overlap, чтобы триггер реагировал только на аватара.

Вопрос: Когда аватар проходит через актер Pickup, он застревает в коллизии статичного меша. Как это исправить?

Ответ: В редакторе блюпринтов для блюпринта Pickup выберите компонент Static Mesh на панели Components. Затем установите пресеты коллизии в значение Custom. Установите параметр Collision Response в значение Ignore, чтобы отключить коллизию компонента Static Mesh и позволить аватару бежать прямо через актер Pickup.

Вопрос: Анимация воспроизводится один раз и останавливается. Как это исправить?

Ответ: Убедитесь, что в редакторе Timeline включен параметр Loop.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Истинно или ложно высказывание: некоторые функции нацелены на актеров, а некоторые — на компоненты.
2. Истинно или ложно высказывание: параметр Auto Play для нода Timeline должен быть установлен постоянно.
3. Истинно или ложно высказывание: нод Timeline может анимировать только значения от 0 до 1.
4. Истинно или ложно высказывание: блюпринт-классы могут иметь множество корневых компонентов.
5. Истинно или ложно высказывание: положение или угол поворота корневого компонента можно отредактировать в блюпринт-классе.

Ответы

1. Истина. В зависимости от того, на что вы хотите повлиять в блюпринте, вы должны использовать корректные типы функций. Под заголовком нода функции может быть надпись Target is Scene Component или Target is Actor.
2. Ложь. Параметр Auto Play нужен, только если вы хотите, чтобы воспроизведение Timeline начиналось при запуске уровня, в противном случае вы можете использовать параметр Event to play the Timeline при необходимости.
3. Ложь. Нод Timeline может использоваться, чтобы анимировать любой диапазон значений.

4. Ложь. Хотя блюпринт-класс может иметь множество компонентов, но корневой компонент в нем может быть только один.
5. Ложь. Вы можете изменять только масштаб корневого компонента.

Упражнение

Вернитесь к актеру Pickup из первой части заданий «Попробуйте сами» в этом часе, добавьте непрерывное вращение и установите переменную времени задержки создания, которая может редактироваться при помещении актера.

1. Добавьте новый трек Float в нод Timeline и переименуйте его в **Rotator**.
2. Добавьте два ключа в новый трек **Rotator** — один в точку времени **0** сек. со значением **0**, и второй в точку времени **1** сек. со значением **1**.
3. Используйте нод функции `setRelativeRotation`, имеющей целью компонент Static Mesh.
4. Щелкните правой кнопкой мыши по контакту данных `New Rotation` нода `setRelativeRotation` и выберите вариант **Split Struct Pin**.
5. Умножьте значение контакта вывода данных с плавающей запятой **Rotator** нода Timeline на **360**.
6. Соедините результат умножения с контактом ввода `New Rotation Z (Yaw)` нода `setRelativeRotation`. Когда вы закончите, блюпринт должен выглядеть, как на рис. 16.13.

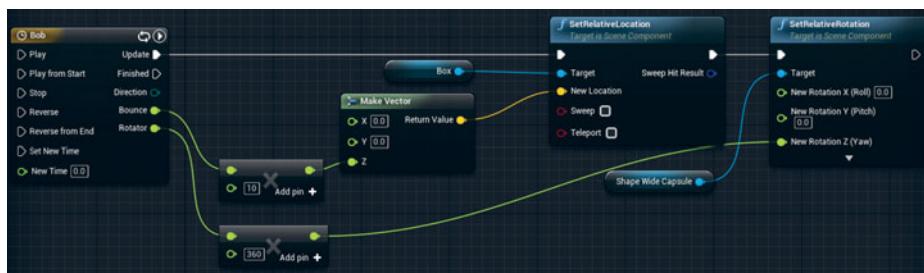


Рис. 16.13. Вращение актера Pickup с нодом Timeline

7. Поместите несколько копий вашего блюпринт-класса Pickup на уровень и запустите его. Они должны непрерывно подскакивать вверх-вниз и вращаться.

17-Й ЧАС

Использование редактируемых переменных и скрипта конструирования

Что вы узнаете в этом часе

- ▶ Как сделать переменные редактируемыми за пределами блюпринта
- ▶ Использование скрипта конструирования
- ▶ Задание диапазона значений переменной

В этом часе вы научитесь создавать редактируемые переменные и использовать скрипт конструирования (Construction Script) в блюпринте. Когда вы используете редактируемые переменные и скрипт конструирования, каждый помещенный экземпляр блюпринта может быть изменен независимо от исходного блюпринт-класса. Базовый функционал класса будет оставаться прежним в процессе геймплея, но исходные настройки актера могут быть уникальными для каждого экземпляра. Этот час проведет вас через процесс создания редактируемых переменных и использования скрипта конструирования.

ПРИМЕЧАНИЕ

Подготовка к практике

Создайте новый проект с шаблоном First Person и стартовым контентом, и затем на панели **Content Browser** создайте новую папку *Hour17Blueprints*.

Подготовка

Предположим, вы создаете блюпринт-класс подбираемого предмета, похожий на тот, что вы создавали в 16-м часе, «Работа с блюпринт-классами». Было бы хорошо иметь возможность изменять статичный меш, скорость вращения или высоту подскакивания подбираемого предмета каждый раз, когда он помещается на уровень. В этом часе вы создадите блюпринт-класс, который, в свою очередь, создает компоненты статичных мешей, число которых определяется пользователем, которые можно переместить или повернуть при необходимости. Сначала вы создадите редактируемую переменную, а затем создадите последовательность блюпринта в скрипте конструирования.

Создание редактируемых переменных

Используя актер блюпринта, который вы создали в 16-м часе, можете поместить на уровень несколько актеров ИС, излучающих пульсирующий свет независимо друг от друга. Тем не менее может понадобиться, чтобы некоторые источники света были ярче остальных, или иметь возможность изменять диапазон интенсивности, значения которой генерируются скриптом. Сейчас эти значения одинаковы для всех экземпляров блюпринта, помещенных на уровень. Вы можете скопировать ассет блюпринта и изменить значения переменных по умолчанию, но это приведет к увеличению количества ассетов, с которыми вы будете работать, что может ухудшить организацию большого проекта.

Редактор блюпринтов позволяет делать переменные в блюпринте редактируемыми. Это означает, что они могут быть изменены вне редактора блюпринтов. Если сделать переменные редактируемыми, они будут отображаться на панели **Details** уровня, когда актер помещен на уровень и выбран. Когда вы делаете переменную редактируемой, то должны дать ей описание и определить, в какой категории она будет храниться. Описание всплывает при наведении курсора на свойство переменной, поэтому, если ваш блюпринт станут использовать коллеги, они будут знать, для чего нужна эта переменная. Категории используются для организации переменных, и это важно, если вам нужно сделать редактируемыми большое количество переменных.

ПОПРОБУЙТЕ САМИ

Сделайте переменные блюпринта редактируемыми

Выполните следующие шаги, чтобы создать блюпринт из класса Actor и сделать переменные редактируемыми, так чтобы каждый помещенный экземпляр актера блюпринта мог быть изменен независимо.

1. В папке *Hour17Blueprints* на панели **Content Browser** щелкните правой кнопкой мыши и выберите вариант **Blueprint Class**.
2. В появившемся окне **Pick Parent Class** выберите категорию **Actor**. На панели **Content Browser** дайте блюпринту любое название по вашему усмотрению и откройте его в редакторе блюпринтов, дважды щелкнув по нему.
3. На панели **My Blueprint** в разделе **Variables** щелкните по символу +, чтобы добавить новую переменную.
4. В текстовом блоке свойства **Variable Name** введите **NumComp** и установите свойство **Variable Type** в значение **Integer**.
5. Установите флагок **Instance Editable**.
6. В текстовом блоке **Tooltip** введите **Set the number of Components to Add**.
7. В текстовом блоке **Category** введите **Actor_Setup**.
8. Щелкните по кнопке **Compile** на панели инструментов редактора блюпринтов.
9. На панели **Details** в разделе **Default Value** установите значение по умолчанию **10**.
10. Создайте еще шесть переменных, основываясь на информации, приведенной в табл. 17.1. Когда вы закончите, ваша категория **Variables** панели **My Blueprint** должна выглядеть, как на рис. 17.1.
11. Щелкните по кнопке **Compile** и затем по кнопке **Save** на панели инструментов редактора блюпринтов.

ТАБЛ. 17.1. Редактируемые переменные, которые нужно добавить в упражнении «Попробуйте сами»

Имя переменной	Тип переменной	Описание	Категория	Значение по умолчанию
PivCompLocation	Vector	Устанавливает местоположение компонента Arrow	Actor_Setup	0,0,20
PivCompRotation	Rotator	Устанавливает угол поворота компонента Arrow	Actor_Setup	0,0,15
MeshCompLocation	Vector	Устанавливает местоположение компонента Static Mesh	Mesh_Setup	100,0,0

Имя переменной	Тип переменной	Описание	Категория	Значение по умолчанию
MeshCompRotation	Rotator	Устанавливает угол поворота компонента Static Mesh	Mesh_Setup	0,0,0
MeshCompScale	Vector	Устанавливает масштаб компонента Static Mesh	Mesh_Setup	1,1,1
SM_MeshAsset	Static Mesh (Object Reference)	Назначает asset меша компоненту меша	Mesh_Setup	SM_CornerFrame

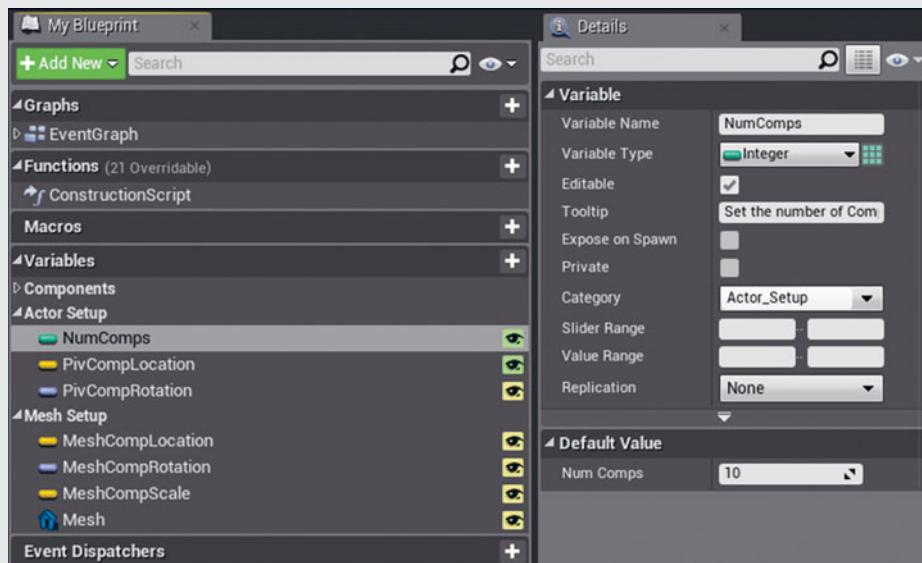


Рис. 17.1. Панели My Blueprint и Details блюпринта, показывающие объявленные редактируемые переменные

Создав все необходимые переменные, перетащите ваш блюпринт из панели Content Browser на уровень. Выбрав его, взгляните на созданные вами категории Actor_Setup и Mesh_Setup на панели Details уровня. В каждой категории вы должны увидеть все созданные переменные, и при наведении курсора на каждую из них должны отображаться описания. Вы можете изменить значения переменных, но ваши изменения не возымеют эффекта, поскольку вы не установили для их использования скрипт конструирования.

Использование скрипта конструирования

Скрипт конструирования (Construction Script) доступен для любого блюпринт-класса. Он обновляется каждый раз, когда изменяются свойства или трансформации актера в редакторе блюпринтов или при компиляции блюпринта. Несмотря на то что редактируемые переменные позволяют изменять каждый помещенный экземпляр актера, вы не увидите изменений до тех пор, пока игра не будет запущена. Скрипт конструирования, тем не менее, обрабатывает изменения в актере в то время, пока вы работаете в редакторе.

ПРИМЕЧАНИЕ

Выполнение скрипта конструирования

По умолчанию скрипт конструирования запускается при каждом изменении переменной на панели **Details** уровня для актера, обновлении трансформации актера, появлении актера и компиляции блюпринта.

В редакторе блюпринтов вы можете увидеть вкладку **Construction Script** рядом с графом событий. Если ее нет, вы можете найти ее на вкладке **Functions** на панели **My Blueprint**, как показано на рис. 17.2. Если дважды щелкнуть по этой ссылке, откроется скрипт конструирования. В скрипте конструирования вы можете увидеть нод события, который называется **Construction Script**. Этот нод выполняет сигнал и обрабатывает соединенные с ним ноды.

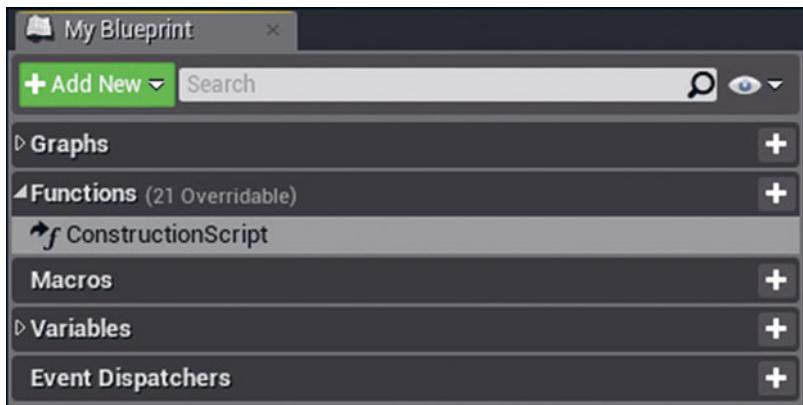


Рис. 17.2. Ссылка Construction Script на панели My Blueprint

Если вы щелкнете по кнопке **Class Settings** на панели инструментов редактора блюпринтов (см. рис. 17.3), вы увидите раздел **Blueprint Options** на панели **Details**

редактора блюпринтов (см. рис. 17.4). Первой опцией в этом разделе является параметр **Run Construction Script on Drag**. Когда рядом с этим параметром установлен флажок, то при изменении положения, угла поворота или масштаба актера, вызывается скрипт конструирования блюпринта.



Рис. 17.3. Панель инструментов редактора блюпринтов

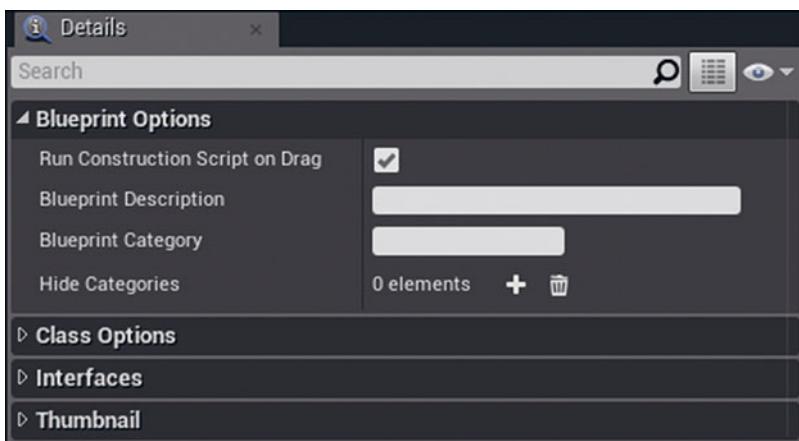


Рис. 17.4. Раздел Blueprint Options панели Details

Поскольку скрипт конструирования запускается в редакторе и часто обновляется, существуют определенные ограничения; некоторые функции недоступны в скрипте конструирования. Например, вы можете добавлять компоненты в блюпринт «на лету», но вы не можете создавать новые актеры. Использование скрипта конструирования — это отличный способ просматривать результаты изменений, редактируемых переменных актера; художники и дизайнеры уровней могут использовать его для получения обратной связи о блюпринтах.

Добавление компонентов Arrow

Теперь, когда вы понимаете, как работает скрипт конструирования, в следующей рубрике «Попробуйте сами» вы будете использовать скрипт конструирования, создадите последовательность, использующую нод ForLoop, и добавите компоненты Arrow в блюпринт.

ПОПРОБУЙТЕ САМИ

Добавьте компонент Arrow

Выполните следующие шаги, чтобы, используя скрипт конструирования и цикл For, добавить несколько компонентов Arrow в блюпринт.

1. Откройте блюпринт, созданный в предыдущем упражнении «Попробуйте сами» (если он не открыт).
2. Выберите вариант **Construction Script** под панелью инструментов.
3. Щелкните по ехес-контакту вывода уже помещенного нода **Construction Script**, перетащите его и отпустите кнопку мыши. В поисковой строке контекстного меню блюпринтов введите **forloop** и выберите вариант **ForLoop** из списка, чтобы добавить нод.
4. В текстовом блоке **First Index** нода ForLoop введите значение **0**.
5. Щелкните по целочисленной переменной **NumComp** и перетащите ее из панели **My Blueprint** в контакт **Last Index** нода ForLoop, чтобы автоматически выбрать вариант **Get** для переменной и добавить ее в граф событий.
6. Щелкните по ехес-контакту вывода **Loop Body** нода **ForLoop** и перетащите его. В текстовом блоке контекстного меню блюпринтов введите **add arrow**. Выберите компонент **Add Arrow**, чтобы добавить его в граф.
7. Щелкните правой кнопкой мыши по оранжевому контакту данных **Relative Transform** и выберите вариант **Split Struct Pin**.
8. Щелкните по контакту целочисленных данных **Index** нода ForLoop, перетащите его и в поисковой строке контекстного меню введите **vector**. Выберите вариант **Vector * Int**, чтобы поместить нод.
9. Щелкните по целочисленной переменной **PivCompLocation** и перетащите ее из панели **My Blueprint** в контакт ввода перемножения векторных данных, чтобы выбрать вариант **Get** для переменной. Соедините контакт вывода перемножения векторных данных с контактом данных **Relative Transform Location** нода **Add Arrow Component**.
10. Повторите шаги 8–9, но в этот раз используйте ноды **ScaleRotator (integer)** и **PivCompRotation**, и соедините их с **Relative Transform Rotation**.
11. Щелкните правой кнопкой мыши по контакту данных **Return Value** нода **Add Arrow Component** и перетащите его. Выберите вариант **Promote to Local Variable**. На панели **My Blueprint** в разделе **Local Variables** переименуйте переменную в **TempArrowComp**. Когда вы закончите, ваш скрипт конструирования блюпринта должен выглядеть, как показано на рис. 17.5.
12. Щелкните по кнопке **Compile** и затем по кнопке **Save** на панели инструментов редактора блюпринтов.

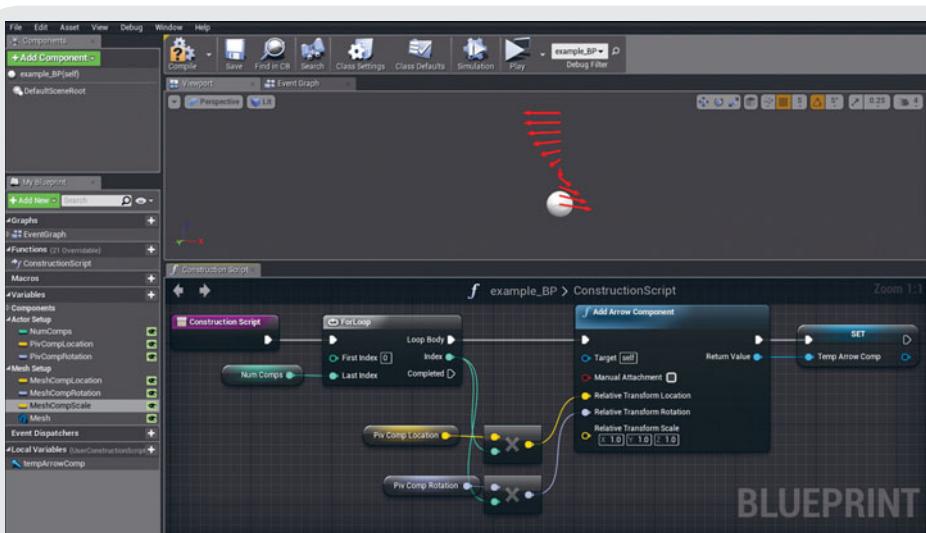


Рис. 17.5. Ваш скрипт конструирования должен выглядеть аналогично изображению

Что вы только что сделали? Вы установили событие ForLoop, которое будет выполняться определенное количество раз в зависимости от значений первого и последнего индекса. Например, если первое значение равно **0**, а последнее — **10**, событие ForLoop выполнится 11 раз. Нод Add Arrow Component просто добавляет компонент Arrow в блюпринт с указанной относительной трансформацией. В данном случае нод добавляет 11 компонентов, и каждый раз он компенсирует положение и угол поворота, умножая переменные PivCompLocation и PivCompRotation на номер индекса события ForLoop. Локальная переменная временно хранит компонент Arrow, добавленный в блюпринт в процессе выполнения (вы будете использовать это далее в этом часе).

ПРИМЕЧАНИЕ

Локальные переменные

Локальные переменные — это временные переменные, созданные в функции. Они доступны только для этой функции. Когда функция прекращает выполнение, переменная больше не используется.

Ранее вы использовали только три созданные вами переменные, но теперь вы установили первую половину скрипта конструирования и можете протестировать его в редакторе блюпринтов. Выберите панель **Viewport** так, чтобы вы могли увидеть компоненты и затем щелкните по кнопке **Class Defaults** на панели

инструментов редактора блупринтов. На панели **Details** блупримта вы увидите ваши категории и все созданные вами переменные. Если изменить значение свойства NumComp, вы увидите, что компоненты Arrow добавляются или удаляются из вышпорта.

ПРЕДУПРЕЖДЕНИЕ

Кнопка Class Defaults

Изменение свойств переменных на панели **Details** блупримта приводит к изменению их значений по умолчанию. Когда вы закончите, убедитесь, что вернули все переменные к их исходным настройкам, как показано в табл. 17.1.

Добавление компонентов Static Mesh

Теперь, когда вы добавили компоненты Arrow в блупримт, необходимо добавить компоненты Static Mesh. После того как вы добавите компонент Static Mesh, вы сможете назначить ассет статичного меша компоненту и увидеть его. Наконец, вы прикрепите его к компоненту Arrow. Компонент Arrow будет служить якорной точкой для компонентов Static Mesh.

ПОПРОБУЙТЕ САМИ

Добавьте компоненты Static Mesh

Выполните следующие шаги, чтобы добавить компоненты Static Mesh, назначить ассет статичного меша каждому компоненту и прикрепить их к созданным ранее компонентам Arrow::

1. В скрипте конструирования щелкните по ехес-контакту вывода TempArrowComp и перетащите его. В поисковой строке контекстного меню введите **add static** и выберите вариант **Add Static Mesh Component** из списка.
2. В node **Add Static Mesh Component** щелкните правой кнопкой мыши по оранжевому контакту данных Relative Transform и выберите вариант **Split Struct Pin**.
3. На вкладке Variables панели **My Blueprint** перетащите векторную переменную **MeshCompLocation** в контакт данных Relative Transform Location нода Add Static Mesh Component, чтобы выбрать вариант **Get** для переменной.
4. Повторите шаг 3 для переменной MeshCompRotation, но назначьте ее контакту данных Relative Transform Rotation. Также повторите шаг 3 для переменной MeshCompScale, но назначьте ее контакту данных Relative Transform Scale.
5. Щелкните правой кнопкой мыши по контакту данных Return Value нода Add Static Mesh Component и выберите вариант **Promote to Local Variable**. На панели **My Blueprint** в разделе Local Variables переименуйте переменную в **TempMeshComp**, чтобы выбрать вариант **Set** для нода.

6. Щелкните по ехес-контакту вывода нода Set, перетащите его и в поисковой строке контекстного меню введите **set static**. Выберите вариант **Set Static Mesh** из списка, чтобы добавить нод.
7. Соедините синий контакт вывода данных нода Set с синим контактом ввода Target нода Set Static Mesh.
8. Перетащите ссылочную переменную статичного меша SM_MeshAsset из раздела Variables панели **My Blueprint** в синий контакт ввода данных New Mesh нода Set Static Mesh, чтобы выбрать вариант **Get** для переменной.
9. Щелкните по ехес-контакту вывода данных нода Set Static Mesh, перетащите его и в поисковой строке контекстного меню введите **attach**. Выберите вариант **AttachToComponent** из списка, чтобы поместить нод.
10. Перетащите переменную TempMeshComp, созданную в шаге 5, из раздела Variables панели **My Blueprint** в синий контакт ввода Target нода AttachToComponent, чтобы выбрать вариант **Get** для переменной.
11. Перетащите переменную TempArrowComp из раздела Variables панели **My Blueprint** в синий контакт ввода данных Parent нода AttachToComponent, чтобы выбрать вариант **Get** для переменной.
12. Щелкните по кнопке **Compile** и затем по кнопке **Save** на панели инструментов редактора блюпринтов.

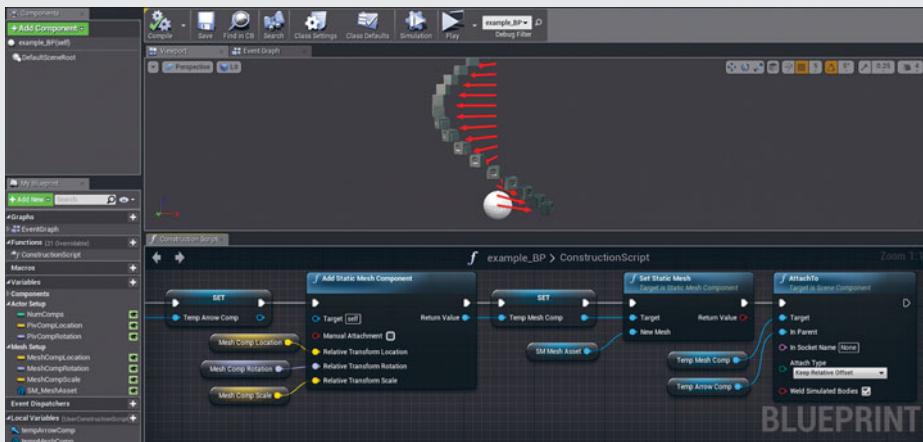


Рис. 17.6. Вторая половина скрипта конструирования

Закончив скрипт конструирования, перетащите ваш блюпринт из панели **Content Browser** на уровень. Выбрав его, взгляните на созданные вами категории **Actor_Setup** и **Mesh_Setup** на панели **Details** уровня. Поэкспериментируйте со значениями переменных, пока не получите желаемый результат. Затем перетащите второй актер в блюпринт, дайте ему другие настройки и назначьте ему другой меш. Оба

актера являются экземплярами одного и того же блюпринт-класса, но могут быть изменены независимо друг от друга.

Ограничение редактируемых переменных

Вы могли заметить, что при повышении значения переменной NumComp, возможны некоторые задержки, связанные с тем, что редактор блюпринтов обновляет скрипт конструирования. При использовании редактируемых переменных уместным является установление ограничений, препятствующих тому, чтобы кто-либо выбирал экстремальные значения. В редакторе блюпринтов выберите переменную NumComp и на панели **Details** блюпринта взгляните на свойства **Slider Range** и **Value Range**. Свойство **Slider Range** позволяет контролировать значения, которые могут выбрать другие разработчики при использовании вашего блюпринта, но пользователи по-прежнему могут вводить любые значения. Свойство **Value Range** блокирует значение таким образом, что пользователи могут лишь выбрать значение в пределах определенного вами диапазона. В первом текстовом блоке для обоих свойств введите 1, а во втором — 100 (см. рис. 17.7). После того как вы установите эти свойства, скомпилируйте и сохраните блюпринт. Затем выберите актер на уровне и на панели **Details** уровня измените свойство NumComp, чтобы увидеть изменения.

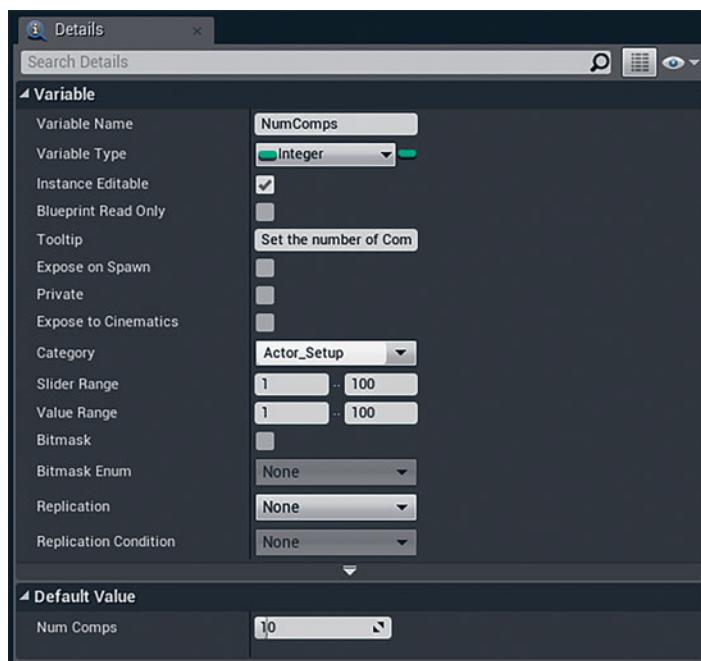


Рис. 17.7. Свойства **Slider Range** и **Value Range** переменной

Свойство Show 3D Widget

Некоторые редактируемые переменные могут быть отображены во виджете уровня, благодаря чему с ними можно взаимодействовать напрямую, используя виджет трансформации. В редакторе блупринтов выберите векторную переменную PivCompLocation и на панели **Details** блупринта установите флажок рядом со свойством **Show 3D Widget**. Скомпилируйте и сохраните ваш блупринт. Затем во виджете уровня найдите виджет в виде каркаса бриллианта рядом с центром вашего актера. Имя переменной также должно быть видимым. Щелкните по виджету в виде бриллианта и переместите его вверх и вниз, и вы увидите обновление положения каждого компонента в реальном времени. Кроме того, значения переменной PivCompLocation на панели **Details** уровня также будут соответствующим образом меняться (см. рис. 17.8). Когда у вас появится возможность, сделайте то же с переменной MeshCompLocation в вашем блупринте.



Рис. 17.8. Выбор свойства Show 3D Widget

Резюме

В этом часе вы узнали, как использовать скрипт конструирования, чтобы обновлять изменения актеров с помощью редактируемых переменных в редакторе блупринта. Как можно увидеть, использование редактируемых переменных и скрипта конструирования имеет мощнейший потенциал. Работая над вашим проектом, ваши коллеги смогут использовать ваш блупринт без всякой необходимости открывать редактор блупринтов, чтобы вносить изменения. Чем комфортнее вы будете себя чувствовать в использовании скрипта конструирования, тем более эффективными в использовании вы сможете делать актеры блупринтов для других членов команды разработчики.

Вопросы и ответы

Вопрос: Что означают значки с зеленым и желтым глазом рядом с переменной на панели My Blueprint?

Ответ: Вы можете использовать значки с зеленым и желтым глазом, чтобы быстро сделать переменную редактируемой. Закрытый глаз означает, что переменная не является редактируемой, желтый глаз означает, что переменная редактируема, но не имеет описания. Зеленый глаз означает, что переменная является редактируемой и имеет описание.

Вопрос: Я не могу редактировать имена моих локальных переменных на панели Details блюпринта. Как мне их переименовать?

Ответ: Переименовывать переменные необходимо на панели My Blueprint. Найдите переменную в разделе Local Variables панели My Blueprint, щелкните по ней правой кнопкой мыши, выберите вариант Rename и введите новое имя.

Вопрос: Почему не отображаются компоненты Static Mesh?

Ответ: Возможны две причины. Первая состоит в том, что вы могли не назначить ассет статичного меша переменной SM_MeshAsset, которую создали в первом упражнении «Попробуйте сами». Вторая заключается в том, что переменная MeshCompScale имеет значения 0,0,0. Измените их на 1,1,1.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Истинно или ложно высказывание: хоть вы и можете добавлять различные компоненты в блюпринт в скрипте конструирования, но с его помощью нельзя создавать новые актеры.
2. Истинно или ложно высказывание: локальная переменная доступна за пределами функции, в которой была создана.
3. Истинно или ложно высказывание: компонент Add Arrow добавляет компонент Static Mesh.
4. Истинно или ложно высказывание: если вы хотите иметь возможность взаимодействовать с векторной переменной во вьюпорте уровня, необходимо установить свойство Show 3D widget переменной.
5. Истинно или ложно высказывание: скрипт конструирования обновляется каждый раз, когда изменяются свойства и трансформации актера.

Ответы

1. Истина. Вы можете создавать новые актеры на уровне в игре из графа событий в блупринте, но вы не можете создавать актеры на уровне из скрипта конструирования.
2. Ложь. Локальные переменные доступны только для функции, внутри которой они были объявлены.
3. Ложь. Компонент Add Arrow добавляет стрелку.
4. Истина. Некоторые типы переменных, такие как векторные переменные, имеют свойство **Show**, отвечающее за отображение визуального представления переменной на уровне, когда экземпляр блупримта помещен на уровень.
5. Истина. В том случае, когда установлен флажок рядом со свойством **Run Construction Script on Drag** в разделе **Class Settings** редактора блупримтов.

Упражнение

В этом упражнении используйте нод Set Material и редактируемую ссылочную переменную интерфейса материала, чтобы изменить материалы всех добавленных компонентов Static Mesh.

1. Откройте блупринт и воспользуйтесь скриптом конструирования.
2. Щелкните по локальной переменной **TempMeshComp**, перетащите ее из панели **My Blueprint** в конец последовательности в скрипте конструирования, выберите вариант **Get** для переменной и добавьте ее в граф событий.
3. Щелкните по контакту переменной **TempMeshComp**, перетащите его в поисковой строке контекстного меню введите **set material**. Выберите вариант **Set Material** из списка, чтобы добавить нод в граф события.
4. Соедините ехес-контакт Set Material с концом нода AttachToComponent.
5. Создайте переменную на панели **My Blueprint**.
6. Установите тип переменной **Material Instance**, переименуйте переменную, сделайте ее редактируемой, дайте ей описание и назначьте ее категории **Mesh_Setup**.
7. Щелкните по кнопке **Compile** и затем по кнопке **Save** на панели инструментов редактора блупримтов.

18-Й ЧАС

Создание актеров и вводимых с клавиатуры событий

Что вы узнаете в этом часе

- ▶ Как сделать переменные открытыми для спауна
- ▶ Спаунинг актера из блюпринт-класса
- ▶ Скриптинг вводимого с клавиатуры события

В предыдущем часе вы научились делать переменные редактируемыми и использовать скрипт конструирования. В этом часе вы научитесь устанавливать вводимые с клавиатуры события в блюпринт, а также как спаунить один актер из другого в процессе игры.

ПРИМЕЧАНИЕ

Подготовка к практике

Создайте новый проект с шаблоном First Person и стартовым контентом, и затем на панели **Content Browser** создайте новую папку *MyBlueprints*.

Почему спаунинг важен

Большинству игр требуется больше, чем просто события коллизий, чтобы реагировать на действия игрока. В данном часе вы создадите блюпринт-класс, который будет спаунить новый актер каждый раз, когда игрок проходит над ним и нажимает клавишу. Возможность спаунить новые актеры на лету открывает дверь для создания более динамичных и интерактивных впечатлений, где подбираемые объекты случайным образом появляются по всему уровню или их число увеличивается во время волн врагов в зависимости от умений игрока. Без спаунинга дизайнеры уровней должны были бы вручную помещать каждый отдельный актер для каждого конкретного сценария, который может открыться во время геймплея,

и это, конечно, невыполнимо. Чтобы произвести спаун актера в процессе геймплея через блюпринт, вы будете использовать функцию **Spawn Actor**, и вам понадобится создать два новых блюпринт-класса: создающий класс и создаваемый класс. Создающий класс спаунит актер другого класса, добавляя его на уровень в процессе геймплея. Создаваемым является порождаемый актер.

Создание блюпринт-класса для спауна

Прежде чем вы сможете заставить один актер спаунить другой из класса, необходимо заставить блюпринт-класс спаунить. В первой части этого часа вы создадите блюпринт-класс, имеющий компонент **Static Mesh**, который симулирует физику. Вы будете использовать скрипт конструирования для изменения свойств каждого создаваемого экземпляра актера физики. Затем вы создадите блюпринт-класс **UseKeySpawner**, который будет спаунить экземпляры блюпринт-класса физики.

ПОПРОБУЙТЕ САМИ

Создайте блюпринт-класс физики

Выполните следующие шаги, чтобы создать блюпринт-класс физики, который вы будете спаунить во время игры.

- Чтобы создать новый блюпринт-класс, щелкните правой кнопкой мыши по пустому пространству в папке **MyBlueprints** на панели **Content Browser** и выберите вариант **Blueprint Class** в контекстном меню. Затем выберите категорию **Actor** на вкладке **Common Classes**.
- Назовите ваш новый блюпринт-класс **PhysicsActor_BP** и затем откройте этот актер в редакторе блюпринтов, дважды щелкнув по нему на панели **Content Browser**.
- На вкладке **Components** щелкните по зеленой кнопке **+Add Component** и выберите вариант **Cube**, чтобы добавить кубический компонент **Static Mesh**.
- Переименуйте новый компонент в **PhysicsMeshComp** и назначьте его корневому компоненту блюпринта, перетащив его в компонент **DefaultSceneRoot** и выбрав вариант **Make Root**.
- Выбрав компонент **PhysicsMeshComp**, на панели **Details** блюпринта в разделе **Physics** установите флашок рядом с параметром **Simulate Physics**.
- Щелкните по кнопке **Compile** и затем по кнопке **Save** на панели инструментов редактора блюпринтов и поместите ваш блюпринт на уровень по умолчанию.
- Запустите уровень и выстрелите в блок; он должен сдвинуться.

Использование скрипта конструирования

Вы создали блюпринт-класс, который будет спауниться, и вам теперь необходимо использовать скрипт конструирования так, чтобы меш и материалы, назначенные компоненту Static Mesh, можно было менять. В дальнейшем это позволит вам изменять внешний вид появляющихся актеров и извлекать пользу из свойства **Expose on Spawn** переменной. Сначала вы создадите скрипт конструирования, а также ссылочные переменные статичного меша и материала. Затем вы сможете сделать каждую переменную открытой для спауна.

СОВЕТ

Нод Sequence

Нод Sequence расщепляет сигнал события на столько сигналов, сколько вам требуется. Поскольку результат сравнения веток неизвестен, вы должны убедиться, что другие сигналы обрабатываются. Вы можете добавить больше ехес-контактов вывода в нод Sequence, щелкнув по кнопке **Add Pin**, а также удалить ненужный ехес-контакт вывода, щелкнув по нему правой кнопкой мыши и выбрав вариант **Remove**.

ПОПРОБУЙТЕ САМИ

Поменяйте статичный меш, назначенный компоненту Static Mesh

Выполните следующие шаги, чтобы использовать скрипт конструирования для смены ассетов статичных мешей и материалов, назначенных компоненту Static Mesh блюпринта PhysicsActor_BP.

1. Откройте блюпринт PhysicsActor_BP.
2. Откройте скрипт конструирования, нажав на соответствующую вкладку или щелкнув по ссылке **Construction Script** в разделе **Functions** панели **My Blueprint**.
3. Щелкните по ехес-контакту вывода помещенного нода Construction Script, перетащите его и отпустите кнопку мыши. В поисковой строке контекстного меню введите **sequence** и выберите вариант **Sequence** из списка, чтобы добавить нод.
4. Щелкните по ехес-контакту вывода **then 0** нода Sequence, перетащите его и отпустите кнопку мыши. В поисковой строке контекстного меню введите **set Static Mesh**. Выберите вариант **Set Static Mesh (PhysicsMeshComp)**, чтобы поместить нод функции Set Static Mesh, целью которой является компонент Static Mesh.
5. В ноде функции Set Static Mesh щелкните правой кнопкой мыши по синему контакту ввода данных слева от свойства **NewMesh** и выберите вариант **Promote to Variable**.

6. Переименуйте новую переменную в **NewMesh** и затем скомпилируйте блюпринт. Поскольку созданной вами переменной Set Static Mesh не назначен меш, меш компонента исчезает из выупорта блюпринта, когда выполняется скрипт конструирования.
7. Чтобы проверить, назначен ли меш переменной прежде, чем будет запущена функция Set Static Mesh, щелкните по контакту вывода данных переменной NewMesh и перетащите его. В поисковой строке контекстного меню введите **isValid** и выберите вариант **?IsValid** из списка, чтобы поместить нод.
8. Щелкните по exec-контакту вывода нода ?IsValid, перетащите его и соедините с exec-контактом ввода нода Set Static Mesh.
9. Затем соедините exec-контакт вывода Then 0 с exec-контактом ввода нода ?IsValid.
10. Повторите шаги 4–9, но теперь используйте exec-контакт вывода Then 1 нода Sequence, чтобы изменить материал нода PhysicsMeshComp с помощью нода функции Set Material и ссылочной переменной материала, которая называется NewMaterial. Когда вы закончите, ваш блюпринт должен выглядеть, как на рис. 18.1.
11. Скомпилируйте и сохраните блюпринт.

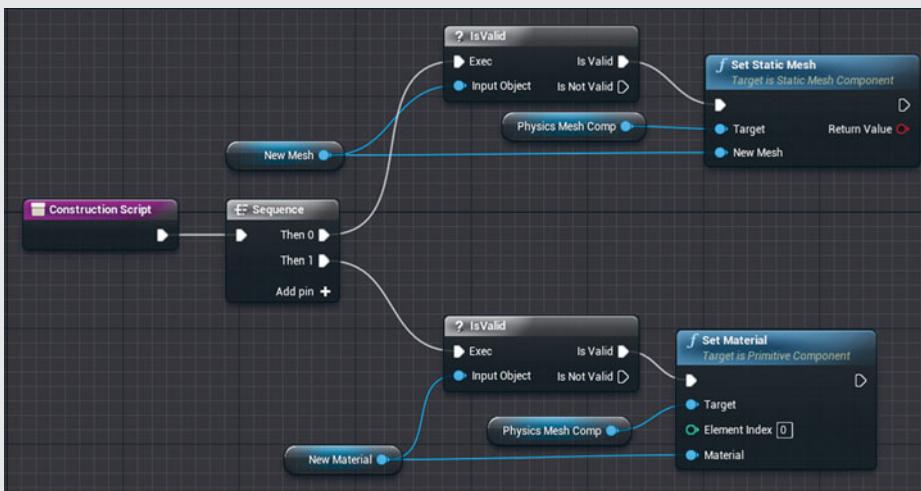


Рис. 18.1. Ваш скрипт конструирования должен выглядеть, как на изображении

Использование свойства переменной Expose on Spawn

Так же как свойство **Instance Editable** делает переменную доступной на панели **Details** уровня для актера, свойство **Expose on Spawn** делает переменную открытой для спауна любым блюпринтом нового актера. Когда класс назначен функции **Spawn Actor from Class**, любая переменная в данном классе с включенным

свойством **Expose on Spawn** отображается в виде контакта ввода данных в функции спауна. Каждая переменная, созданная в блюпринте, может иметь свойство **Expose on Spawn**.

Прежде чем вы перейдете к созданию создающего блюпринта, который будет спаунить актер физики, вам необходимо подготовить две переменные, созданные в предыдущем упражнении «Попробуйте сами».

ПОПРОБУЙТЕ САМИ

Подготовьте переменные, чтобы они были открыты для спауна

Выполните следующие шаги, чтобы отредактировать свойства переменных, которые вы создали ранее в этом часе.

1. Откройте блюпринт PhysicsActor_BP.
2. Найдите и выберите переменную **NewMesh** в разделе **Variables** на вкладке **My Blueprint**.
3. На панели **Details** блюпринта установите флажок **Instance Editable**, как показано на рис. 18.2.
4. В текстовом блоке **Tooltip** дайте переменной информативное описание, например **This will change the mesh of the physics Actor**.
5. Установите флажок **Expose on Spawn**, как показано на рис. 18.2.
6. В текстовом блоке **Category** введите **Mesh_Setup**, чтобы создать новую категорию свойств Mesh Setup.
7. Повторите шаги 3–5 для переменной **NewMaterial** и в графе **Category** выберите уже созданную категорию **Mesh_Setup** из списка.
8. Скомпилируйте и сохраните блюпринт.

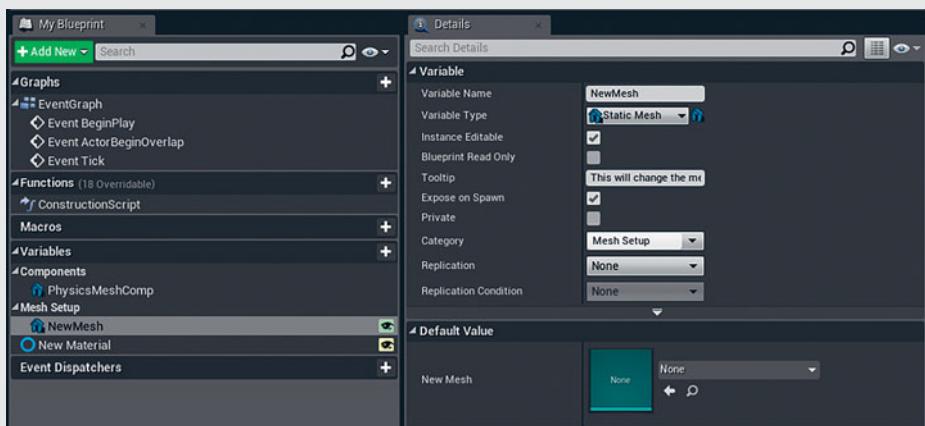


Рис. 18.2. Панели **My Blueprint** и **Details** в редакторе блюпринтов

Создание запускающегося блюпринта

Скриптинг события ввода, требующего, чтобы игрок нажал клавишу для запуска последовательности блюпринта, — довольно простой процесс. Вам необходимо событие ввода, которое назначено определенной клавише, например клавише E. Вам также необходимо задать актеру инструкцию, чтобы он временно включал возможность ввода для определенных контроллеров. Если вы просто включите данную возможность для актера и поместите множество экземпляров актера на свой уровень, все они будут выполняться одновременно при нажатии клавиши ввода. Поэтому возможность ввода нужно включать только для актера, с которым игрок пытается взаимодействовать напрямую. Это можно сделать с помощью события перекрытия, которое включает возможность ввода для актера, когда аватар перекрывает компонент коллизии, и отключает возможность ввода, когда аватар отходит и перекрытие исчезает.

Этот метод хорош для ввода одной клавишей для события, имеющего очень специфическую функцию, такую как спаунинг актера или открытие двери. Для актеров, требующих более мощных систем ввода, таких как аватары, персонажи и транспортные средства, лучше всего устанавливать маппинг клавиш.

См. 20-й час, «Создание аркадного шутера: системы ввода и аватары», чтобы узнать больше о маппинге клавиш.

Чтобы установить этот блюпринт, вам нужен компонент прямоугольной коллизии для события Overlap, статичный меш для визуального представления местоположения актера на уровне и компонент Arrow для определения местоположения спауна актера физики, когда пользователь нажимает клавишу.

ПОПРОБУЙТЕ САМИ

Установите и используйте блюпринт-класс Key Spawner

Теперь, когда актер физики готов к использованию, пора установить актер UseKeySpawner. Для этого выполните следующие шаги.

- Создайте новый блюпринт-класс. Выберите папку *MyBlueprints* на панели **Content Browser**, щелкните правой кнопкой мыши по области управления ассетами и выберите вариант **Blueprint Class** в контекстном меню. Затем выберите категорию **Actor** на вкладке **Common Classes**.
- Назовите ваш блюпринт-класс **UseKeySpawner_BP** и откройте его в редакторе блюпринтов (см. рис. 18.3).
- Добавьте компонент **Box Collision** и установите его местоположение по оси Z в значение **100**. Затем установите размеры блока **100** по оси X, **100** по оси Y и **100** по оси Z.

4. Добавьте компонент Cylinder из раздела **Basic Shapes** и установите его место- положение по оси Z в значение **50**.
5. Скомпилируйте и сохраните блюпринт.

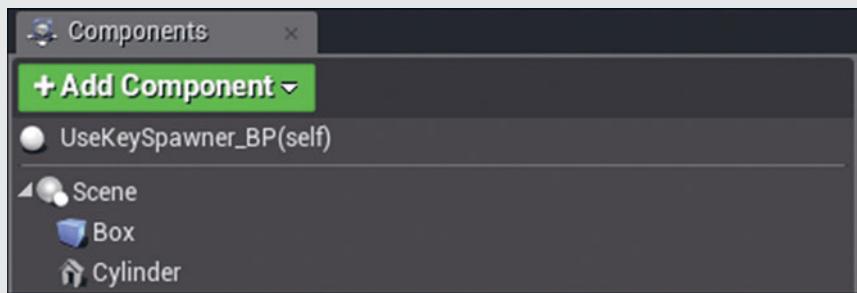


Рис. 18.3. Создание компонента

Когда все необходимые компоненты готовы, требуется заскриптовать последовательность события перекрытия для компонента Box Collision, который включает возможность актера получать ввод от контроллера игрока и отключает ее, когда перекрытие прекращается.

ПОПРОБУЙТЕ САМИ

Заскриптуйте события перекрытия так, чтобы они включали и отключали возможность ввода

Выполните следующие шаги, чтобы использовать скрипт конструирования для изменения свойств актера.

1. Откройте блюпринт UseKeySpawner_BP.
2. Выберите компонент Box Collision на вкладке **Components**. В графе событий добавьте нод события **OnComponentBeginOverlap**.
3. Добавьте нод функции **Enable Input** и соедините ее с контактом вывода **OnComponentBeginOverlap(Box)** с контактом ввода нода **Enable Input**.
4. Добавьте нод **Player Controller** с выбранным вариантом **Get** и соедините его синий контакт вывода данных с синим контактом ввода нода **Enable Input**.
5. Выберите компонент Box Collision на вкладке **Components**. В графе событий добавьте нод события **OnComponentEndOverlap**.
6. Добавьте нод функции **Disable Input** и соедините ее с контактом вывода **OnComponentEndOverlap(Box)** с контактом ввода нода **Disable Input**.

7. Соедините синий контакт вывода данных Player Controller, добавленного в шаге 4, с синим контактом ввода Player Controller нода Disable Input. Когда вы закончите, ваш блюпринт должен выглядеть, как на рис. 18.4.
8. Скомпилируйте и сохраните блюпринт.



Рис. 18.4. Граф событий в блюпринте, отображающий последовательности событий перекрытия

Спаунинг актера из класса

Актеры добавляются в процессе геймплея через спаунинг. Если вы не можете спаунить актеры, то вам необходимо заранее поместить каждый актер, который вам когда-либо потребуется, это сократит количество возможных игровых событий и столкновений, которые можно создать. Спаунинг позволяет скриптовать

динамические впечатления. Существуют функции спауна для добавления определенных основных типов актеров. Например, существует эмиттер спауна эффектов частиц, спаун звука для добавления звуковых актеров на уровень при необходимости. Актеры могут появляться с указанными трансформациями или они могут быть прикреплены к другим актерам. При спаунинге актера вам необходимо определить его местоположение, поскольку вряд ли вы захотите, чтобы актер появился внутри оболочки коллизии другого актера или компонента.

Для этой демонстрации вы будете использовать функцию Spawn Actor from Class, которая может спаунить любой созданный вами блюпринт-класс.

ПОПРОБУЙТЕ САМИ

Добавьте вводимое с клавиатуры событие и создайте актер из класса

Теперь, когда вы добавили события включения и отключения возможности ввода, вам необходимо добавить вводимое с клавиатуры событие, которое запускается, когда игрок нажимает клавишу E. Выполните следующие шаги.

1. Откройте блюпринт UseKeySpawner_BP.
2. Щелкните правой кнопкой мыши по пустому пространству в графе событий, чтобы вызвать контекстное меню. В поисковой строке введите e и выберите вариант E из списка, чтобы добавить его в граф.
3. Добавьте функцию Spawn Actor from Class и соедините ее exec-контакт ввода с exec-контактом вывода Released события E.
4. Справа от фиолетового контакта ввода данных щелкните по полю **Select Class** и используйте поисковую строку, чтобы найти блюпринт PhysicsActor_BP, который вы скриптовали ранее.
5. Добавьте трансформацию **world**, чтобы определить местоположение в мире, куда будет добавлен созданный актер. Выберите компонент Arrow на панели **Components** и перетащите его в граф событий.
6. Щелкните по синему контакту вывода данных ссылки на компонент и перетащите его. В поисковой строке контекстного меню введите **get world transform** и выберите вариант **GetWorldTransform**, чтобы добавить его в граф событий.
7. Соедините оранжевый контакт вывода данных GetWorldTransform с оранжевым контактом ввода данных Spawn Transform нода функции Spawn Actor. Когда вы закончите, ваш блюпринт должен выглядеть, как на рис. 18.5.
8. Скомпилируйте и сохраните блюпринт. Затем поместите его экземпляр на уровень.
9. Запустите уровень, переместите аватар, чтобы он находился над помещенным экземпляром актера UseKeySpawner_BP, и нажмите клавишу E. Актер физики должен будет добавиться на уровень.

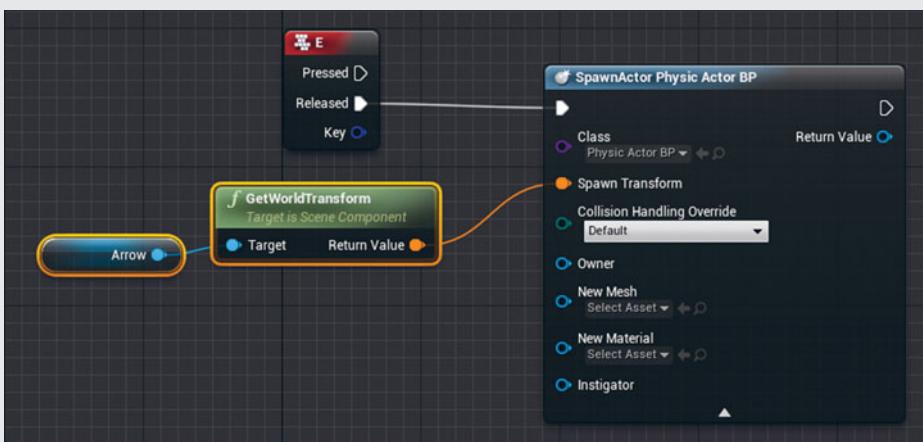


Рис. 18.5. Последовательность вводимого с клавиатуры события

ПРИМЕЧАНИЕ

Ввод с клавиатуры

Вы также можете использовать вводимое с клавиатуры событие для того, чтобы воспроизводить или останавливать временную шкалу, анимирующую компонент меша, например открытие или закрытие двери.

Последнее, что вам необходимо сделать, — это воспользоваться переменными, которые вы открыли для спауна в PhysicsActor_BP. Когда вы добавляете PhysicsActor_BP в свойство класса функции Spawn Actor From Class, открытые переменные также добавляются в функцию благодаря свойству **Expose on Spawn**. Теперь вам необходимо добавить эти переменные в актер UseKeySpawner_BP и сделать их редактируемыми, так чтобы, когда вы помещаете экземпляр на уровень, вы могли выбирать новый меш и материал для появляющегося актера физики.

ПОПРОБУЙТЕ САМИ

Активируйте переменные и сделайте их редактируемыми

Чтобы изменить открытые для спауна переменные, которые отражены в ноде функции Spawn Actor from Class, вам необходимо создать две новые переменные в блюпринте и сделать их редактируемыми. Выполните следующие шаги.

1. Откройте UseKeySpawner_BP в редакторе блюпринтов.
2. В ноде функции Spawn Actor from Class щелкните правой кнопкой мыши по синему контакту ввода данных слева от свойства **NewMesh** и выберите вариант **Promote to Variable**.

3. Теперь выберите переменную и на панели **Details** блупринта установите флагок рядом со свойством **Instance Editable**.
4. Повторите шаги 2–3 для свойства **NewMaterial**.
5. Скомпилируйте и сохраните блупринт.
6. На уровне выберите помещенный экземпляр актера **UseKeySpawner_BP** и на панели **Details** назначьте его новому мешу и материалу.
7. Запустите уровень и провзаимодействуйте с **UseKeySpawner_BP**.
8. Поместите несколько экземпляров актера **UseKeySpawner_BP** и назначьте им различные меши и материалы. Когда вы закончите, ваш блупринт должен выглядеть, как на рис. 18.6.
9. Запустите уровень снова и провзаимодействуйте с каждым помещенным актером.

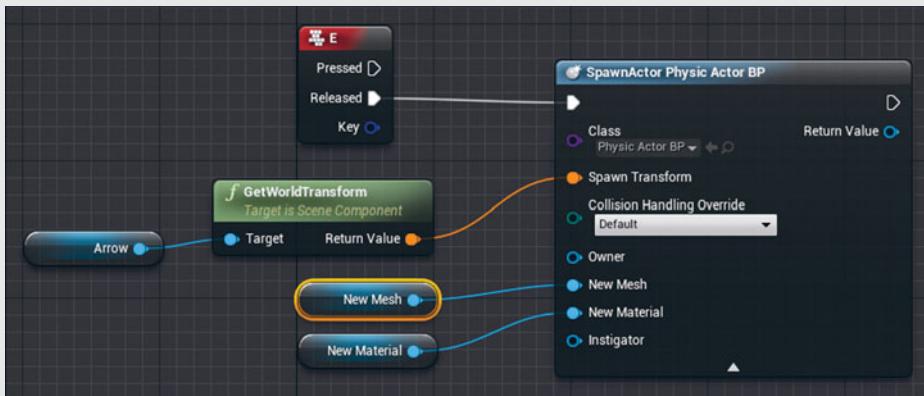


Рис. 18.6. Активированные редактируемые переменные, добавленные в последовательность

Резюме

В этом часе вы узнали, как заставить один актер спаунить второй с другими свойствами. Вы также узнали, как использовать свойство **Expose on Spawn** переменной и как включать и отключать возможность ввода контроллером игрока для актера. Теперь вы можете начать улучшать эти умения, чтобы создавать более динамичные актеры, с которыми игрок может взаимодействовать.

Вопросы и ответы

Вопрос: Будет ли метод ввода с клавиатуры, описанный в данном часе, работать в мультиплеерной игре?

Ответ: Нет, поскольку возможность ввода включается только для контроллера Player Controller 0, который является контроллером по умолчанию в синглплеерной игре.

Вопрос: Как изменить масштаб созданного актера?

Ответ: Вы можете сделать это в трансформации спауна. Отсоедините оранжевый провод от компонента стрелки GetWorldTransform, щелкните правой кнопкой мыши по оранжевому ноду трансформации и выберите вариант Split Struct Pin, чтобы разделить структуру трансформации на отдельные свойства местоположения, поворота и масштаба.

Вопрос: Каждый раз, когда появляется актер физики на уровне, он тут же падает. Почему?

Ответ: Появившийся актер физики может столкнуться с другим актером или компонентом на уровне. Поскольку местоположение спауна определяется компонентом Arrow в блюпринт-классе UseKeySpawner_BP, изменение местоположения компонентов Arrow в блюпринте решит проблему.

Вопрос: После спауна второго актера событие ввода прекращает работать. Почему так происходит?

Ответ: Событие OnComponentEndOverlap компонента Box Collision запускается одним из создаваемых актеров. В UseKeySpawner_BP отредактируйте свойства коллизии компонента Box Collision так, чтобы он реагировал только на аватар.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

- Истинно или ложно высказывание: вводимое с клавиатуры событие работает только для клавиши E.
- Истинно или ложно высказывание: включение свойства **Expose on Spawn** для переменной в блюпринте делает переменную видимой для нода функции **Spawn Actor from Class**.

3. Если вы хотите, чтобы блупринт спаунил актер, какая функция вам нужна?
 - A. GetWorldTransform
 - B. Spawn Actor from Class
 - C. OnComponentBeginOverlap
 - D. Enable Input

4. Истинно или ложно высказывание: если вы включите возможность ввода для блупринт-класса и поместите множество экземпляров актера на уровень, все они будут выполняться одновременно при нажатии клавиши ввода.

Ответы

1. Ложь. Существуют события ввода для любой клавиши.
2. Истина. Свойство **Expose on Spawn** для переменной делает ее доступной для функции `Spawn Actor from Class`.
3. В. Несмотря на то что для спауна существует несколько функций, функция `Spawn Actor from Class` позволяет спаунить ваши собственные блупринт-классы.
4. Истина. Вы должны использовать события, чтобы включать и отключать возможность ввода в блупринт-классе при необходимости.

Упражнение

Спаунинг актера физики при нажатии игроком клавиши E функционирует не очень захватывающе. Также не обеспечивается обратная связь с игроком, когда он или она взаимодействует с объектом, помимо отображения созданного актера. В данном упражнении вы создадите в своем блупринте `UseKeySpawner` анимированный рычаг, который спаунит эффект частиц и воспроизводит звук, когда появляется актер физики.

1. В блупринте `UseKeySpawner_BP` добавьте ассет статичного меша **Shape_Cylinder** из папки *Starter Content*, как компонент, и переименуйте его в **LeverMesh**.
2. Отмасштабируйте компонент **LeverMesh** так, чтобы он был похож на рычаг. Установите размеры **0,1** по оси X и Y и **2,0** по оси Z.
3. Расположите компонент **LeverMesh** на позицию **0,70,0** вправо от цилиндрического компонента меша, добавленного в упражнении «Попробуйте

сами», которое называлось «Установите и используйте блюпринт-класс Key Spawner».

4. В графе событий UseKeySpawner_BP добавьте временную шкалу со снятыми флагками свойств **Auto Play** и **Loop** и параметром **Time** в значении **1** (второй).
5. Добавьте кривую числа с плавающей запятой и назовите ее **LeverRotation**.
6. Добавьте три ключевых кадра к кривой числа с плавающей запятой: Keyframe 1 (со значением параметра **Time 0** и **Value 0**), Keyframe 2 (со значением параметра **Time 0** и **Value 1**) и Keyframe 3 (со значением параметра **Time 1** и **Value 0**).
7. Соедините ехес-контакт вывода Released нода события ввода E с ехес-контактом ввода Play from Start временной шкалы.
8. Щелкните по аксету LeverMesh, перетащите его и выберите вариант **Get**, чтобы добавить его как ссылочную переменную в граф событий.
9. Щелкните по синему контакту данных LeverMesh и в поисковой строке контекстного меню введите **Set relative rotation**. Выберите вариант **SetRelativeRotation** из списка, чтобы поместить нод.
10. Соедините ехес-контакт вывода Timeline Update с ехес-контактом ввода нода SetRelativeRotation.
11. Щелкните по зеленому контакту нода LeverRotation, созданного в шаге 5, перетащите его и в поисковой строке контекстного меню введите **multiply**. Выберите вариант **Float * Float**, чтобы добавить нод. В текстовом блоке введите **60** — это количество градусов, на которое рычаг будет поворачиваться при анимировании.
12. В ноде функции SetRelativeRotation щелкните правой кнопкой мыши по контакту данных о повороте, выберите вариант **Split Pin** и соедините контакт вывода данных с плавающей запятой нода перемножения с контактом New Rotation Y (Pitch) нода SetRelativeRotation.
13. Щелкните по ехес-контакту вывода Finished нода временной шкалы, перетащите его и в поисковой строке контекстного меню введите **spawn emitter**. Выберите вариант **Spawn Emitter at Location**, чтобы добавить нод. В разделе Emitter Template назначьте **P_Explosion**.
14. Щелкните по ехес-контакту вывода Spawn Emitter at Location, перетащите его и в поисковой строке контекстного меню введите **play sound**. Выберите вариант **Play Sound at Location**, чтобы добавить нод. Рядом со свойством Sound назначьте звуковой аксет **Explosion01**.

15. Соедините exec-контакт вывода Timeline Finished с exec-контактом ввода нода Spawn.
16. Используйте трансформацию компонента Arrow, чтобы установить свойства **Location** и **Rotation** нодов Spawn Emitters at Location и Play Sound at Location.
17. Соедините exec-контакт вывода Play Sound at Location с нодом Spawn Actor, который вы поместили в упражнении «Попробуйте сами», которое называлось «Добавьте вводимое с клавиатуры событие и создайте актер из класса», ранее в этом часе.
18. Когда вы закончите, ваш блупринт должен выглядеть, как на рис. 18.7. Скомпилируйте и сохраните блупринт.

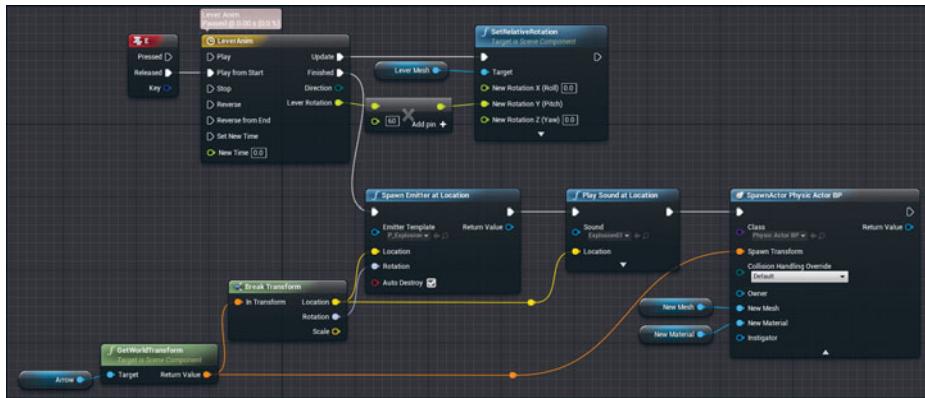


Рис. 18.7. Последовательность блупринта, анимирующая рычаг и создающая актер при нажатии игроком клавиши E

19. Запустите уровень и провзаимодействуйте с созданным актером. Когда вы нажимаете клавишу E, воспроизводится анимация рычага и взрыва, вы слышите звук взрыва и видите появление актера физики.

19-Й ЧАС

Создание экшен-столкновения

Что вы узнаете в этом часе

- ▶ Работа с существующим блюпринт-классом для создания полосы препятствий
- ▶ Изменение свойств передвижения персонажа
- ▶ Назначение режима игры уровню
- ▶ Назначение тега актера

В предыдущих часах вы познакомились с блюпринтами. В этом часе вы будете использовать существующие блюпринт-классы для создания вашего собственного столкновения, основанного на действии. Используя один из предоставленных режимов игры, вы будете помещать и изменять существующие блюпринт-классы, чтобы создавать основанную на времени полосу препятствий.

ПРИМЕЧАНИЕ

Подготовка к практике

Для этого часа вам понадобится открыть проект *Hour_19*, из проектов по адресу: http://addons.eksмо.ru/it/UE_24.zip. В нем вы найдете все, что нужно для работы в этом часе и создания простого столкновения для режима игры от первого лица (шутер) или от третьего лица. На панели **Content Browser** проекта *Hour_19* вы увидите папку *BasicFPSTGame*, в ней вы найдете режим игры под названием *BasicFPSTGameMode*. На панели **Content Browser** проекта вы также увидите папку, которая называется, *Basic3rdPGame*. В этой папке находится режим игры *Basic3rdPGameMode*. Вы также найдете коллекцию блюпринт-классов, организованную в папки в зависимости от их функционала.

Игровые режимы проекта

Для этого часа мы обеспечили два режима игры: шутер от первого лица (FPS, first-person shooter) под названием *BasicFPSTGameMode* и режим игры от третьего лица под названием *Basic3rdPGameMode*. Режим игры FPS использует блюпринт

персонажа, который называется BasicFPSCharacter, а режим игры от третьего лица использует блюпринт персонажа Basic3rdPCharacter.

HUD-интерфейсы

Оба игровых режима для этого часа имеют простую UMG-графику (Unreal Motion Graphics) — HUD-интерфейсы (Heads-Up Displays). HUD-интерфейсы для обоих режимов отображают уровень здоровья персонажа, количество полученных предметов и таймер времени с начала уровня. В обоих игровых режимах персонажи могут умереть от падения с выступа или от получения урона.

См. 22-й час «Работа с UMG», чтобы узнать больше о создании интерфейсов и работе с Unreal Motion Graphics (UMG) UI Designer.

Игровой таймер и система респауна

Оба игровых режима для этого часа имеют системы респауна и таймера, заскриптованные в своих блюпринтах. Таймер начинает отсчет, когда запускается уровень, и отображается в HUD-интерфейсе. Система респауна работает в связке с актерами блюпринтов CheckPoint_BP и KillVolume_BP, которые вы можете найти в папке BP_Respawn на панели Content Browser.

Изучение умений персонажей

При создании столкновения на уровне следует знать все об умениях персонажей. Как быстро они передвигаются? Как высоко и далеко они могут прыгать? Какое оружие у них есть? Чем больше вы знаете об умениях персонажей, тем лучше вы можете спроектировать столкновения на вашем уровне.

Персонаж режима FPS, которым вы будете пользоваться в этом часе, имеет основное оружие, заскриптованное в блюпринт-классе персонажа. В нем содержится «линейное» оружие (стреляющее по линейной траектории), баллистическое оружие и ружье физики. Нажатием клавиш клавиатуры 1, 2 и 3 производится переключение между видами оружия. Вы можете стрелять из линейного и баллистического оружия, нажимая левую кнопку мыши. Когда активно ружье физики, вы можете щелкнуть левой кнопкой мыши, чтобы подобрать актеры физики, и щелкнуть правой кнопкой мыши, чтобы бросить подобранный актер или толкнуть физический предмет, если он стоит на земле.

На панели Content Browser найдите файл *Basic3rdPGame/Blueprints/Basic3rdPCharacter* и откройте блюпринт в редакторе блюпринтов. На панели Components выберите компонент CharacterMovement и взгляните на свойства

на панели **Details** блюпринта для этого компонента. Здесь вы найдете практически всю нужную информацию. По большей части передвижение персонажа базируется на ускорении и скорости. Попробовав протестировать различные значения свойств, вы можете получить лучшее понимание, как они соответствуют реальным единицам измерения.

В следующей рубрике «Попробуйте сами», вы познакомитесь с настройками компонента CharacterMovement.

ПОПРОБУЙТЕ САМИ

Задайте умения игрока в режиме от третьего лица

Выполните следующие шаги, чтобы использовать уровень JumpTest, чтобы задать высоту и расстояние прыжка игрока.

1. На панели **Content Browser** в папке *Hour_19/Maps* откройте уровень **JumpTest**. На данном уровне вы можете увидеть несколько BSP-актеров, имеющих различные размеры. Вы будете использовать их, чтобы получить представление, как быстро, высоко и далеко персонаж может бегать и прыгать.
2. Запустите уровень и попрактикуйте прыжки из одного конца в другой. Попытайтесь прыгать в состоянии покоя и на бегу. Вы должны увидеть некоторые незначительные различия в расстоянии в зависимости от скорости и ускорения игрока. С помощью этих тестов вы можете произвести оценку высоты и расстояние прыжка игрока в режиме от третьего лица. Со значениями по умолчанию игрок может прыгать приблизительно на расстояние в 600 единиц и в высоту на 200 единиц.
3. На панели **Content Browser** найдите блюпринт **Basic3rdPCharacter** и откройте его.
4. В редакторе блюпринтов на панели **Components** выберите компонент **CharacterMovement**.
5. На панели **Details** найдите свойство **Max Walk Speed** в разделе **Character Movement: Walking** (см. рис. 19.1). Измените значение на **300**.
6. Скомпилируйте блюпринт, затем запустите уровень и провзаимодействуйте с BSP-актерами вновь.
7. На панели **Details** найдите свойство **Jump Z Velocity** в разделе **Character Movement: Jumping/Falling** (см. рис. 19.1). Измените его значение на **1000**.
8. Скомпилируйте блюпринт, запустите уровень и провзаимодействуйте с BSP-актерами еще раз.
9. Поиграйте с некоторыми другими свойствами персонажа. Вы всегда можете вернуть значения свойств по умолчанию, просто щелкнув по желтой стрелке справа от значения свойства.

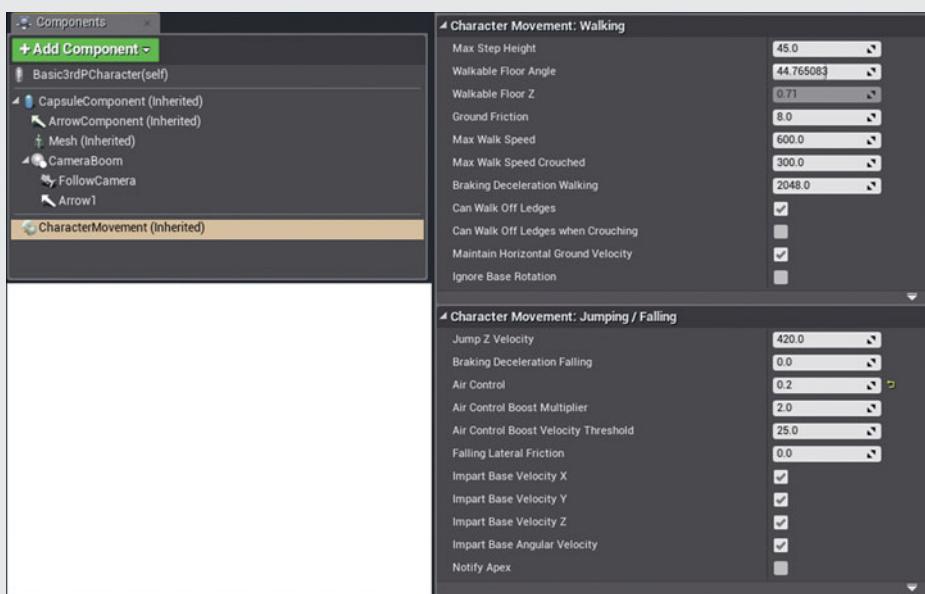


Рис. 19.1. Свойства компонента CharacterMovement

ПРИМЕЧАНИЕ

Базовый режим игры от первого лица

Шаги, которые вы выполнили в предыдущем упражнении «Попробуйте сами», также относятся и к базовому режиму игры от первого лица (BasicFPSCharacter), но вам необходимо изменить значение параметра настройки Game Mode Override для уровня JumpTest на BasicFPSGameMode.

Использование блупринт-классов

Все блупринт-классы, которые вы можете использовать для создания вашей полосы препятствий на уровне, организованы в папки на основе их функционала. В проекте есть папка для движущихся платформ и препятствий, которые могут наносить игроку урон. Другая папка содержит блупринты актеров турелей и баллистического оружия. В другой папке есть рычаги и переключатели, также есть папки с подбираемыми объектами, контрольными точками спауна и килл-зонами.

Все актеры используют скрипт конструирования и редактируемые переменные, поэтому каждый помещенный актер может быть изменен при необходимости. Все

актеры имеют свойства, связанные с их базовым функционалом, которые можно изменить при помощи панели **Details** уровня. Некоторые из них имеют свойства мешей, материалов и частиц, которые можно заменить вашими собственными ассетами.

СОВЕТ

Сетка и привязка

При помещении блюпринтов, используемых в этом часе, может быть полезна привязка к сетке и установка единиц в значение 100, как показано на рис. 19.2.



Рис. 19.2. Сетка и привязка

В следующих разделах описаны предоставляемые в этом часе папки и их содержимое.

Папка BP_Common

Папка *BP_Common* содержит блюпринт-классы, которые можно использовать в обоих предоставляемых игровых режимах. Эта папка содержит демокарту *ActorGallery*, которая демонстрирует базовый функционал всех блюпринт-классов. На панели **Content Browser** в папке *Hour_19/BP_Common* откройте карту *ActorGallery* и запустите уровень.

В этой папке находится шесть актеров, которые вы можете использовать для создания полосы препятствий.

- ▶ **Launcher_BP.** Этот актер запускает персонажа игрока в воздух, используя заданное расстояние и высоту. Чтобы изменить направление, просто поверните помещенный актер.
- ▶ **Mover_BP.** Этот блюпринт анимирует компонент меша между двумя положениями. Вы можете установить параметры скорости движения и временной задержки до смены направления. Вы также можете установить, будет ли анимация запускаться в месте окончания или в месте начала. Конечную точку вы можете установить, выбрав трансформацию *Destination*, переместив и повернув актер в любое желаемое положение.
- ▶ **Pendulum_BP.** Этот актер раскачивается назад и вперед и наносит определенное количество урона персонажу игрока, если ударит по нему. Вы можете

установить скорость раскачивания и начальное направление. Этот актер можно повернуть и равномерно отмасштабировать.

- ▶ **Smasher_BP.** Этот актер анимирует два острых поршня, движущихся назад и вперед и наносящих урон игроку, если тот попадет в центр актера. Вы можете изменить конечное местоположение, скорость возврата и атаки, задержку удара и атаки, а также величину урона. Этот актер можно повернуть и отмасштабировать.
- ▶ **Stomper_BP.** Этот актер анимирует компонент меша между двумя локациями на основе расстояния места его размещения. Он наносит определенный урон персонажу игрока. Этот актер можно повернуть и равномерно отмасштабировать.
- ▶ **SpikeTrap_BP.** Этот актер выпускает шипы из пола, когда персонаж проходит по нему. Вы можете изменить скорость, величину урона и звуковые эффекты этого актера.

В следующей рубрике «Попробуйте сами» вы попрактикуетесь в создании уровня, настройке режима игры для уровня и в работе с одним из предоставленных актеров блюпринта. Чтобы вызвать панель **World Settings** для текущего уровня выберите вариант **Settings** ⇒ **World Settings** на панели инструментов редактора уровней (см. рис. 19.3).

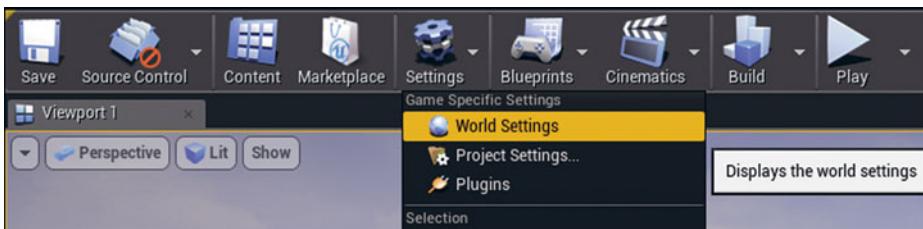


Рис. 19.3. Открытие панели **World Settings**

Панель **World Settings** открывается рядом с панелью **Details** в интерфейсе редактора уровней. Эта панель позволяет устанавливать такие свойства, как Lightmass, Physics и Game Mode для уровня, над которым вы в данный момент работаете. Если вы назначите класс Game Mode, все блюпринт-классы, назначенные классу Game Mode, будут добавлены автоматически. В следующем упражнении «Попробуйте сами» вы пройдете через этот процесс.

ПОПРОБУЙТЕ САМИ

Работа с предоставленными блюпринт-классами

Выполните следующие шаги, чтобы создать уровень по умолчанию и попрактиковаться в использовании класса Mover_BP/

1. Создайте новую карту по умолчанию и сохраните ее в папку *Hour_19/Maps*.
2. На панели **World Settings** задайте параметру **Game Mode Override** значение **Basic 3rdPGameMode** (см. рис. 19.4).
3. Выберите актер Mover_BP на панели **Content Browser** и поместите его на уровень.
4. Выберите помещенный актер Mover_BP, затем выберите синий бриллиант (он называется Destination Transform) и переместите его в новое местоположение.
5. Запустите уровень и обратите внимание, как актер движется. Затем переместите персонаж на платформу, чтобы прокатиться на ней.
6. Остановите игру и, выбрав актер, измените значение свойства **Move Speed** на панели **Details**. Большие значения увеличивают продолжительность и замедлят движение; меньшие значения сократят продолжительность и ускорят движение платформы.
7. Когда вы добьетесь желаемой скорости передвижения, измените значения свойств **Return Delay** и **Destination Delay**, чтобы создать паузу перед началом движения.
8. Создайте копию актера Mover_BP. Выбрав актер, удерживайте клавишу **Alt** и переместите актер или нажмите комбинацию клавиш **Ctrl+W**, чтобы скопировать актер.
9. Переместите скопированный актер в новое место и затем перетащите его виджет Destination Transform, так чтобы он поравнялся с виджетом Destination Transform первого актера.
10. Запустите уровень и сделайте так, чтобы персонаж перемещался с одной платформы на другую.
11. Внесите изменения в трансформации отправной точки и точки назначения, в параметры **Move Speed**, **Destination Delay** и **Return Delay**, чтобы улучшить перемещение.

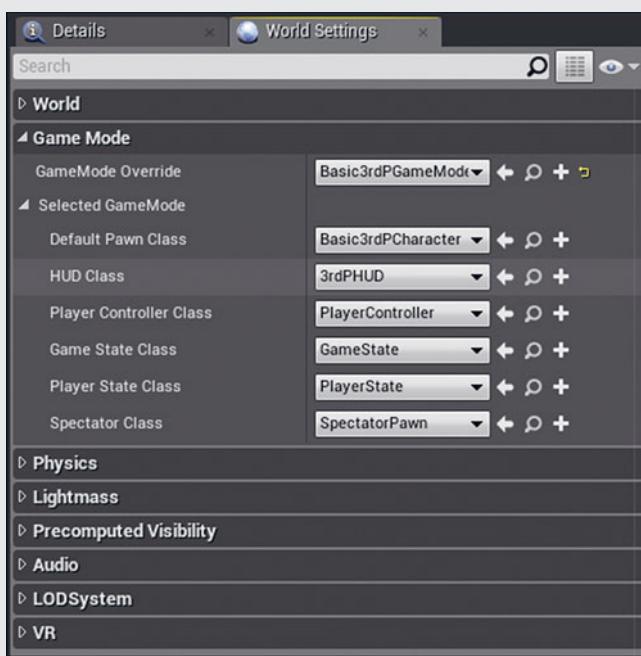


Рис. 19.4. Настройка режима игры

Папка BP_Turrets

Папка *BP_Turrets* содержит три типа блупринтов турелей и блупринты их пуль. В ней есть две отслеживающие турели, которые отслеживают персонажа, когда он находится на определенном расстоянии от их позиции. Также в ней есть основанная на шаблоне турель, которая спаунит пули в определенном направлении указанного шаблона. Обе турели работают с обоими предоставленными режимами игры.

На панели **Content Browser** в папке *Hour_19/BP_Turrets/* откройте карту *TurretGallery* и запустите уровень.

Ниже представлены блупринты, содержащиеся в этой папке/

- ▶ **Pattern_Projectile_BP.** Это блупринт пуль, которые спаунятся блупринтом *PatternTurret_BP*. Они наносят урон игроку при ударе.
- ▶ **PatternTurret_BP.** Этот блупринт спаунит блупринт *Pattern_Projectile_BP* в шаблоне, основываясь на указанных свойствах. Его можно поместить, повернуть и равномерно отмасштабировать при необходимости.

- ▶ **ProjectileTurret_BP.** Эта турель отслеживает персонажа игрока и стреляет пулями (*TurretProjectile_BP*) в персонажа при его движении внутри определенного радиуса. Свойства **Turret Range**, **Track Speed** и **Fire Rate** могут быть изменены при необходимости. Этот блюпринт можно поместить, повернуть и равномерно отмасштабировать, если нужно.
- ▶ **TraceTurret_BP.** Эта турель отслеживает персонажа игрока и стреляет из отслеживающего оружия в персонажа при его движении внутри определенного радиуса. Свойства **Turret Range**, **Track Speed** и **Fire Rate** могут быть изменены при необходимости. Этот блюпринт можно поместить, повернуть и равномерно отмасштабировать, если нужно.
- ▶ **TurretProjectile_BP.** Этот блюпринт спаунится блюпринтом *ProjectileTurret_BP*. Он наносит урон игроку при ударе.

Папка BP_Respawn

В папке *BP_Respawn* находятся два блюпринт-класса, которые можно использовать для респауна на контрольных точках или для уничтожения игрока, когда он падает с уступа.

На панели **Content Browser** в папке *Hour_19/BP_Respawn* откройте карту *Respawn_Gallery* и запустите уровень.

Ниже приведены блюпринты из этой папки.

- ▶ **Checkpoint_BP.** Этот блюпринт-класс работает с системой респауна в предоставленных режимах игры. Когда персонаж проходит над этим актером, последний отправляет данные о своем местоположении в Game Mode. Затем, когда игрок погибает, он респаунится в локации актера. Если на уровне есть несколько актеров этого класса, локацией респауна будет считаться последний актер, взаимодействовавший с персонажем.
- ▶ **KillVolume_BP.** Этот блюпринт-класс уничтожает персонажа игрока при соприкосновении, запуская событие разрушения в предоставленных режимах игры путем удаления и респауна игрока в последней контрольной точке актера.

Папка BP_Pickup

В папке *BP_Pickup* находятся три блюпринт-класса подбираемых объектов. Один из них осуществляет восстановление здоровья, передавая игроку очки здоровья. Другой представляет собой коллекцию подбираемых объектов, которые игрок может собрать при прохождении уровня. Третий является основанным на физике

объектом, который может быть подобран ружьем физики, так, что игрок может подобрать актера и переместить его из одного местоположения в другое. Этот блюпринт работает только с предоставленным базовым режимом игры от первого лица, когда игрок использует ружье физики.

На панели **Content Browser** в папке *Hour_19/BP_Pickup* откройте карту *Pickup_Gallery* и запустите уровень.

Ниже приведены блюпринты из этой папки.

- ▶ **CollectionPickup_BP.** Для этого блюпринта вы можете поменять меш, материал и назначение. Он работает с обоими предоставленными режимами игры от первого и от третьего лица. Блюпринт можно отмасштабировать и поместить в любую точку.
- ▶ **HealthPickup_BP.** Для этого блюпринта вы можете поменять меш, материал и величину здоровья. Он работает с обоими предоставленными режимами игры от первого и от третьего лица. Блюпринт можно отмасштабировать и поместить в любую точку.
- ▶ **PhysicsPickup_BP.** Для этого блюпринта вы можете поменять материал меша. Он работает только с ружьем физики в режиме игры от первого лица. Он имеет тег актера, назначенный ему, поэтому он работает только с ружьем физики.

Папка BP_Levers

Папка *BP_Levers* содержит коллекцию блюпринт-классов, используемых для активации и включения и выключения других блюпринтов. В ней есть блюпринт, требующий, чтобы игрок нажал клавишу E для использования рычага, а также основанный на прикосновении блюпринт, требующий, чтобы персонаж или актер физики с тегом был помещен на него для активации другого актера.

Блюпринты рычагов и переключателей могут использоваться для открытия и закрытия блюпринта *Door_BP* и блюпринт-класса *Stomper_BP* из папки *BP_Common*.

На панели **Content Browser** в папке *Hour_19/ BP_Levers* откройте карту *Lever_Gallery* и запустите уровень.

Ниже приведены блюпринты из этой папки.

- ▶ **UseKeyLever_BP.** Этот актер работает, когда персонаж игрока проходит над ним и нажимает клавишу E. Он анимирует рычаг и посыпает сигнал любому актеру на уровне, который был назначен его списку актеров для активации.

- ▶ **Door_BP.** Этот триггер прикосновения открывает дверь, когда игрок проходит над ним. Он может иметь запертое состояние, требующее еще один актер, например UseKeyLever_BP или TouchActivation_BP, для того чтобы отпереть дверь, прежде чем ею сможет воспользоваться игрок.
- ▶ **PhysicSpawner_BP.** Когда этот актер помещен на уровень, игрок может, проходя над ним и нажимая клавишу E, спаунить физические подбираемые объекты (их можно найти в папке *BP_Pickup*). Есть также свойство, которое позволяет вам назначать теги актерам физики, которые можно спаунить.
- ▶ **TouchActivation_BP.** Этот блупринт-класс работает, когда персонаж или актер физики Pickup_BP с тегом актера настроены на взаимодействие с ним через клавиатурный ввод. Он может отпереть Door_BP или включить ActivateStomper_BP. Вы можете вручную назначить другие актеры на уровне, которые хотите активировать при взаимодействии с TouchActivation_BP.
- ▶ **ActivateStomper_BP.** Этот актер может быть включен, когда получает сигнал от UseKeyLever_BP или TouchActor_BP.

Все эти актеры используют блупринт-интерфейс (BPI, Blueprint Interface) для коммуникации друг с другом. При помещении этих актеров вам необходимо назначить, на какой из помещенных актеров они должны воздействовать.

Теги актеров и компонентов

Чтобы актер физики мог работать с ружьем физики, он должен иметь тег актера. *Teg* — это имя, которое может быть назначено актеру или компоненту актера для того, чтобы различать актеры или компоненты одного типа в блупринте. В этом случае последовательность ружья физики в блупринте персонажа FPS ищет только актеры физики с назначенным тегом актера Pickup. Например, у вас может быть два актера статичного меша, симулирующих физику, но вы хотите, чтобы только один из них мог использоваться с ружьем физики. В таком случае вам понадобится назначить тег Pickup только актеру физики, который вы хотите сделать доступным для подбора игроком.

На панели **Content Browser** в папке *Hour_19/BP_Pickups* откройте карту *ActorTagExample* и запустите уровень. Нажмите клавишу 3, чтобы переключиться на ружье физики.

В этом случае есть два актера статичных меша, и оба симулируют физику. Оба актера можно толкнуть, подойдя к ним, но актер статичного меша справа назначен тегу актера Pickup и его можно подобрать, бросить и толкнуть. Выбрав ружье физики, попытайтесь подобрать актеры.

На рис. 19.5 показаны свойства панели **Details** уровня для актера статичного меша с тегом.

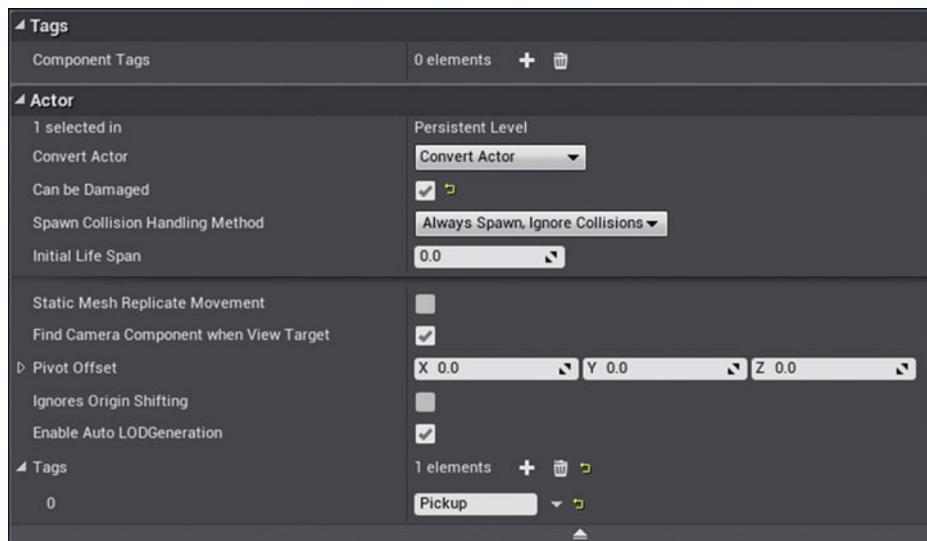


Рис. 19.5. Теги актеров и компонентов

Поскольку актер PhysicsPickup_BP уже имеет назначенный ему тег Pickup, он сразу работает с ружьем физики, когда помещается на уровень или спаунится на нем.

Резюме

В этом часе вы изучили коллекцию блупринт-классов, которые можете поместить и изменить, чтобы создать уровень в виде полосы препятствий для режима игры от первого или от третьего лица. Вы узнали, как изменить исходные умения персонажа, а также познакомились с понятием тегов актеров и компонентов.

Вопросы и ответы

Вопрос: Когда я играю на уровне, который я создал в этом часе, у меня нет персонажа от первого или третьего лица, которым я мог бы управлять. Почему?

Ответ: Не забудьте установить значение BasicFPSGameMode или Basic3rdPGame для свойства Game Mode Override для уровня на панели World Settings.

Вопрос: Когда я назначаю актеры свойству Actor Activate List блупринта TouchActivation_BP, они не включаются. Почему?

Ответ: Хотя актер TouchActivation_BP транслирует сигнал любому актеру из списка через блюпринт-интерфейс (BPI), не каждый актер из списка знает, как получить сигнал. Только блюпринт-классы Door_BP и ActivateStomper_BP настроены на реагирование на сигнал.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

- Чтобы изменить свойства передвижения игрока по умолчанию, какой компонент необходимо отредактировать в блюпринт-классе персонажа, назначенному режиму игры?
- Если вы хотите иметь возможность подобрать актер статичного меша, симулирующий физику, ружьем физики, что вы должны ему назначить?
- Если вы хотите больше контроля и точности при помещении актера на уровень, полезно будет включить _____ для сетки, а также для трансформаций поворота и масштаба.

Ответы

- Компонент Character Movement содержит свойства передвижения персонажа по умолчанию.
- Тег актера. Ружье физики в режиме игры от первого лица взаимодействует только с актерами статичных мешей, симулирующими физику и имеющими тег актера Pickup.
- Привязку. Включение привязки позволяет контролировать количество единиц сетки, градус поворота и процент масштаба.

Упражнение

В этом упражнении вы создадите уровень с полосой препятствий в проекте *Hour_19*, используя предоставленные блюпринт-классы и один из режимов игры. Когда вы закончите создание полосы препятствий, вы можете загрузить один из трех пакетов ассетов среди Infinity Blade и бесплатный пакет Infinity Blade FX из магазина и добавить их в проект, чтобы задекорировать его.

- В проекте *Hour_19* создайте новый уровень по умолчанию, дайте ему имя и сохраните его в папку *Maps*.

2. Откройте панель **World Settings** для уровня и задайте свойству **Game Mode Override** уровня значение **BasicFPSGameMode** для режима игры от первого лица или **Basic3rdPGameMode** для режима игры от третьего лица.
3. Создайте черновой набросок уровня, используя примитивные актеры статичных мешей и/или BSP-актеры.
4. Используя блюпринт-классы из папок *BP_Common*, *BP_Pickups* и *BP_Respawn*, спроектируйте полосу препятствий для игрока.
5. Внесите изменения в свойства актеров, запустите и скорректируйте уровень при необходимости.
6. Когда вы будете довольны своим уровнем, загрузите один из трех бесплатных пакетов ассетов *Infinity Blade* из магазина в лаунчере Unreal и добавьте контент в проект.
7. Загрузите бесплатный пакет *Infinity Blade FX* из Магазина в лаунчере Unreal и добавьте его в проект.
8. Используя ассеты *Infinity Blade*, задекорируйте уровень и поместите источники света и актеры звуков окружения на уровень по мере необходимости.
9. Произведите просчет освещения и запустите уровень.

20-Й ЧАС

Создание аркадного шутера: системы ввода и аватары

Что вы узнаете в этом часе

- ▶ Определение требований и сводки проекта
- ▶ Создание нового проекта
- ▶ Создание пользовательского Game Mode
- ▶ Создание пользовательских аватаров и контроллера игрока
- ▶ Управление движением аватара
- ▶ Настройка фиксированной камеры

При создании новой игры вы почти всегда даете игроку возможность контролировать какую-то часть игрового мира. Это может быть как полноценный персонаж, так и простой объект. Главное здесь то, что игрок должен что-то делать, например нажимать на кнопки или стрелять, а игра будет реагировать на эти действия.

В UE4 можно использовать эти физические действия с помощью контроллеров игрока, а также использовать аватары, способные действовать самостоятельно. В этом часе мы разберем эти понятия и поможем вам создать вашу первую игру: простой аркадный шутер. Вы узнаете, как определить требования исходя из проекта, как создать и настроить новый проект, как спаунить и использовать аватары и как настроить игровую камеру.

ПРИМЕЧАНИЕ

Подготовка к практике

В этом часе мы создадим игру с нуля. Мы создадим пустой проект, включающий только Starter Content. В папке *Hour_20* (файлы проектов можно скачать по адресу: http://addons.eksмо.ru/it/UE_24.zip) вы найдете необходимые для работы ассеты, а также вариант игры под названием *H20_ArcadeShooter*, который вы можете использовать для сравнения со своим результатом.

Определение требований с помощью сводки проекта

Не существует двух идентичных игр. Нам важно сосредоточиться на основных элементах, которые вы хотите добавить в игру. В этом часе мы создадим простой аркадный шутер, похожий на *Space Invaders* или *Asteroids*. Перед тем как создать игру, нам необходимо определить требования к ней и особенности.

Суть игры в нашем случае проста: игрок управляет космическим кораблем, который может двигаться влево и вправо, может маневрировать и уничтожать астероиды, которые попадаются по ходу игры.

Определение требований

Перед началом проекта очень важно уделить некоторое время типам взаимодействия, которые нужно будет реализовать в игре. Понимание требований к игре поможет вам сосредоточиться непосредственно на ее создании. Проект игры, которая создается в этом часе, можно условно разделить на следующие составные части.

- ▶ Игрок управляет космическим кораблем.
- ▶ Космический корабль может двигаться влево или вправо.
- ▶ На пути игрока появляются астероиды, движущиеся вниз.
- ▶ Космический корабль может стрелять в астероиды, чтобы уничтожить их.

В любом подобном кратком описании можно выделить полезную для себя информацию. Проект подсказывает, что в вашей игре потребуется актер, которым игрок сможет управлять; в UE4 они реализованы объектами класса *Pawn*. Проект также подразумевает, что перемещение корабля ограничивается одной осью. Это требование означает, что необходимо настроить привязку элементов управления к этой оси. Поскольку известно, что игрок ограничен в движении, вы можете предположить, что камера неподвижна, и игрок ею не управляет. Вы также видите, с какими препятствиями столкнется игрок, и, соответственно, нужен будет элемент управления для выстрелов.

Создание игрового проекта

Первое, что всегда нужно делать при создании новой игры, — подготовить новый проект в UE4. Unreal Engine содержит множество стартового контента и шаблонов для новых проектов. Но можете создавать фантастические вещи с нуля, используя при создании проекта шаблон *Blank Project*.

СОВЕТ**Настройка стартового уровня**

Вы можете изменить заданный по умолчанию стартовый уровень для игры и редактора, выбрав пункт меню **Project Settings** ⇒ **Maps & Modes**. Изменение карты по умолчанию **Editor Default Map** на карту, с которой вы в настоящий момент работаете, может ускорить процесс работы, а изменение карты игры по умолчанию (**Game Default Map**) меняет карту, которую игра использует при запуске (при игре в одиночном режиме).

В следующем разделе «Попробуйте сами» вы создадите новый пустой проект и пустую карту, на которой, словно на холсте, будете создавать новую игру.

ПОПРОБУЙТЕ САМИ**Создание нового проекта и уровня по умолчанию**

Выполните следующие действия, чтобы создать новый пустой проект и заменить уровень по умолчанию на новый пустой уровень, который ляжет в основу нашего аркадного шутера.

1. Запустите браузер проектов (**Project Browser**) и перейдите на вкладку **New Project**, как показано на рис. 20.1.
2. Выберите шаблон **Blank Project**.
3. Задайте цель проекта **Desktop/Console**.
4. Присвойте параметру **Quality** значение **Maximum Quality**.
5. Задайте папку для сохранения вашего проекта.
6. Присвойте проекту название **ArcadeShooter**.
7. Нажмите кнопку **Create Project**, чтобы создать новый проект.
8. После загрузки проекта выберите команду меню **File** ⇒ **New Level** (или нажмите сочетание клавиш **Ctrl+N**).
9. Выберите в диалоговом окне **New Level** шаблон **Default**.
10. Выберите команду **File** ⇒ **Save As** (или нажмите комбинацию клавиш **Ctrl+Shift+S**).
11. В диалоговом окне **Save Level As** щелкните правой кнопкой мыши по папке **Content** и выберите пункт **New Folder**. Переименуйте новую папку, присвоив ей имя **Maps**.
12. Выбрав каталог **Maps**, введите в поле **Name** название карты **Level_0**.
13. Нажмите кнопку **Сохранить**.
14. На панели **Project Settings** выберите вкладку **Maps & Mods**.
15. Присвойте значение **Level_0** параметрам **Game Default Map** и **Editor Startup Map**.

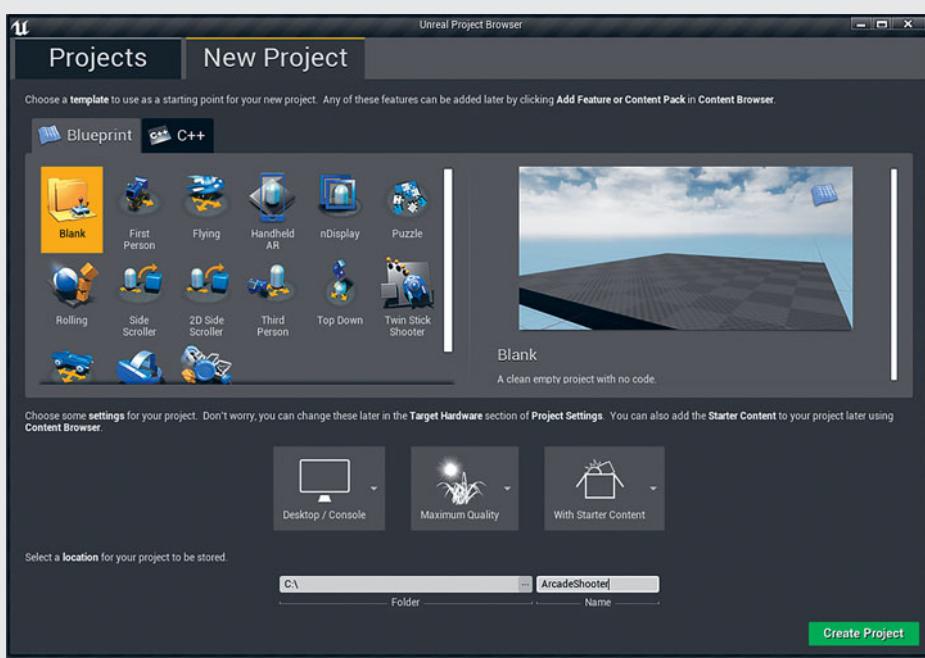


Рис. 20.1. Вкладка New Project в браузере проектов

СОВЕТ

Папка Maps

Разумеется, вы можете хранить UAssets уровня, где угодно внутри папки *Content*, но хорошим тоном является хранить все уровни в каталоге с именем *Maps*. Если уровень находится в папке *Maps*, он будет отображаться в раскрывающихся списках, например в списке выбора карты по умолчанию. Это также позволит создать исполняемый файл *UE4 Front End* для распространения и упростит процесс подготовки, так как ваши уровни будут находиться автоматически.

Теперь, когда вы создали пустой уровень, перейдем к настройке логики и системы игры.

Создание пользовательского режима игры

Вам потребуется место, где будет храниться информация о логике и поведении вашей игры. В UE4 у каждого уровня есть свой блюпринт, в котором можно хранить игровую логику. Но если помещать слишком много скриптов в блюпринт уровня, потребуется много копирований и вставок, чтобы перенести их логику на новые уровни и карты. Вместо этого в UE4 есть понятие *Game Mode* (Режим игры). Как и блюпринт, режимы игры хранят сложные модели поведения, связанные с игрой, но, в отличие от блюпринтов, они позволяют разделить поведение между несколькими уровнями.

Game Mode определяет поведение игры и следит за соблюдением правил. Game Mode содержит информацию о предметах, с которыми игрок начинает игру, и что происходит, когда аватар гибнет или игра заканчивается, включая, например, временные ограничения или подсчет очков.

Режимы игры являются связующим звеном между различными системами в игре. Блюпринт режима игры содержит персонажа или аватары, которых вы используете, эталоны, используемые классом HUD, информацию о том, какой используется класс зрителя, состояние игры и класса игрока, а также контролирует информацию, необходимую для многопользовательской игры.

На самом базовом уровне Game Mode устанавливает правила текущей игры, например, сколько игроков могут присоединиться, как обрабатываются переходы между уровнями, информацию о том, приостановлена или запущена сейчас игра, условия победы и поражения в игре.

Создать новый Game Mode легко. Щелкните правой кнопкой мыши на панели **Content Browser** и выберите пункт **Blueprint Class**, чтобы открыть окно **Pick Parent Class**, в котором вы можете нажать кнопку **Game Mode**, как показано на рис. 20.2.

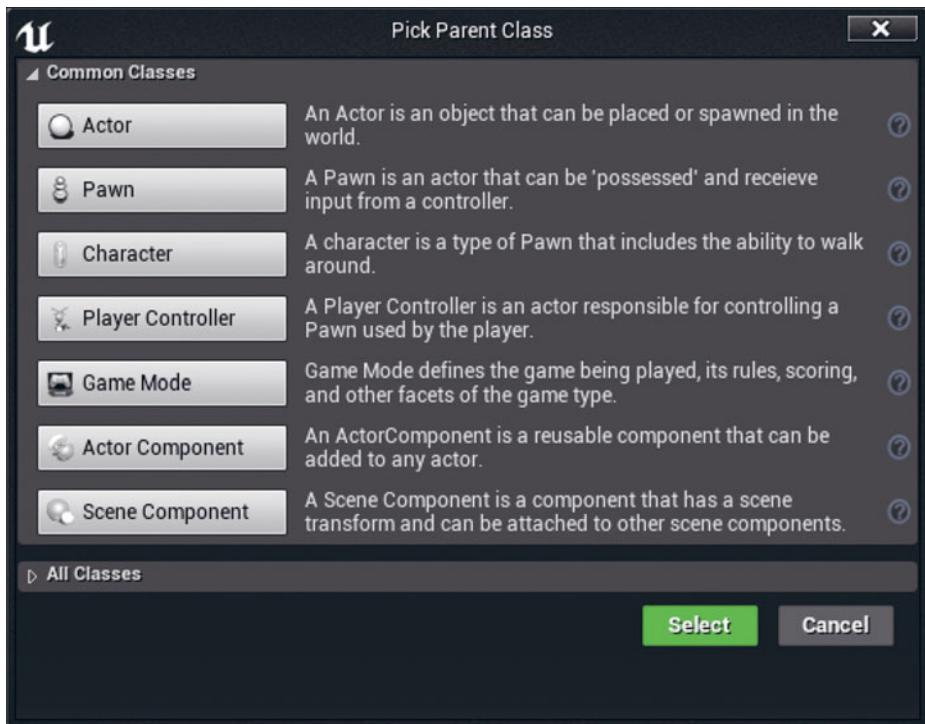


Рис. 20.2. Окно Pick Parent Class. Это окно используется часто, и в нем можно выбрать несколько вариантов класса, в том числе **Game Mode**, который вам нужен

ПОПРОБУЙТЕ САМИ

Создание нового класса Game Mode Blueprint

Выполните предложенные действия, чтобы создать новый класс **Game Mode Blueprint**, в котором мы будем хранить логику игры.

1. Щелкните правой кнопкой мыши на панели **Content Browser** и выберите пункт меню **Folder**.
2. Назовите новую папку **Blueprints**.
3. Дважды щелкните по папке **Blueprints**, чтобы открыть ее.
4. Щелкните правой кнопкой мыши на панели **Content Browser** и выберите пункт меню **Blueprint Class**.
5. В появившемся окне **Pick Parent Class** нажмите кнопку **Game Mode**.
6. Назовите новый режим игры **ArcadeShooter_GameMode**.
7. Выберите команду меню **File ⇒ Save All** (или нажмите комбинацию клавиш **Ctrl+S**).

Создав новый Game Mode, вы должны дать UE4 понять, что нужно использовать именно его вместо режима по умолчанию. Вы можете сделать это на панели **Project Settings**.

СОВЕТ

Переопределение уровня

Иногда для различных частей игры необходимо использовать различные Game Mode. Каждый уровень может также переопределять Game Mode и настройки класса. Чтобы изменить эти параметры для каждого уровня, выберите команду меню **Window** ⇒ **World Settings** и найдите свойство **Game Mode Override** (Переопределение режима игры). Это свойство работает здесь точно так же, как и на панели **Project Settings**. Кроме того, при добавлении параметра **Game Mode Override** вы можете изменить другие свойства, например свойства аватаров или классов HUD, что может быть полезно при создании новых функций.

На уровне присутствует только один Game Mode — либо Game Mode по умолчанию, заданный на панели **Project Settings**, либо Game Mode, установленный для каждого уровня. В многопользовательской игре Game Mode запускается только на сервере, а результаты применения правил и состояния направляются (реплицируются) каждому клиенту.

ПОПРОБУЙТЕ САМИ

Установка нового Game Mode по умолчанию

Выполните следующие действия, чтобы использовать раздел **Maps & Modes** на панели **Project Settings** для задания Game Mode по умолчанию для вашей игры.

1. Выберите пункт меню **Edit** ⇒ **Project Settings**.
2. На панели **Project Settings** выберите раздел **Maps & Modes**.
3. В разделе **Default Modes** щелкните поле **Default GameMode**, чтобы открыть окно поиска для всех режимов игры.
4. Выберите только что созданный режим **ArcadeShooter_GameMode**.

Создание пользовательского аватара и контроллера игрока

В UE4 актеры, которые управляются непосредственно игроками или искусственным интеллектом (ИИ), называются аватарами (*Pawns*). Автоматы могут существовать в виде чего угодно: динозавров, людей, монстров, транспортных средств,

попрыгунчиков, космических кораблей и даже еды. Любой объект, управляемый игроком или ИИ, — это аватар, программно представленный как Pawn.

В некоторых играх нет физического или видимого представления игроков, но аватары, тем не менее, используются, задавая физическое расположение игроков в игровом мире.

Объекты Pawn определяют видимый внешний вид управляемых объектов, а также несут информацию о движении, физике и способностях. Часто бывает полезно считать, что они являются аватарами, то есть физическим воплощением игрока в игровом мире.

Нефизическое воплощение игрока — это контроллер. Контроллеры являются интерфейсом между аватаром и игроком или ИИ, контролирующим его.

Контроллеры — это актеры, которые могут управлять аватарами. Контроллеры не имеют физической формы и, как правило, не определяют физические свойства (например, внешний вид, движение, физику) управляемого аватара. Они используются для передачи команд от игрока.

Связь между контроллерами и объектами Pawn в обе стороны построена по принципу «один контроллер — один Pawn». Это значит, что объекты Pawn могут управляться ИИ через контроллер ИИ или игроком через контроллер игрока.

По умолчанию контроллер игрока обрабатывает большую часть необходимого для игры поведения, но для этого потребуется создать объект Pawn.

Наследование от класса DefaultPawn

Чтобы создать новый тип Pawn, вы можете создать новый блюпринт-класс. На этот раз воспользуемся классом из раздела All Classes окна Pick Parent Class, так как в этом классе уже есть несколько готовых функций. При создании блюпринт-класса раскройте список All Classes, чтобы получить доступ ко всем классам в проекте. Как показано на рис. 20.3, вы можете просматривать этот список и выбрать нужный вам класс. В нашем случае нам требуется класс DefaultPawn, так как в нем автоматически заданы некоторые функции, необходимые в нашей игре.

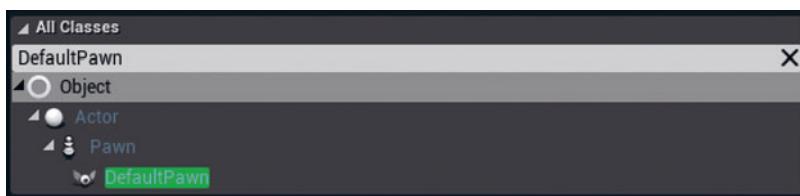


Рис. 20.3. В окне Pick Parent Class раскройте список All Classes и найдите нужный класс Pawn, например DefaultPawn

СОВЕТ**Наследование классов**

Наследование от существующего класса позволяет легко и быстро перенести обобщенные модели поведения на другие объекты. Например, при наследовании от класса DefaultPawn, вы создаете класс, который имеет те же обобщенные модели поведения, но помимо этого позволяет внести изменения. Если в классе DefaultPawn (или любом другом родительском классе) будут сделаны изменения, дочерний тип Pawn также примет их.

Использование наследования по всему проекту поможет избежать повторений и противоречивой работы.

ПОПРОБУЙТЕ САМИ**Создание пользовательских классов Pawn и контроллера игрока**

Выполните предложенные действия, чтобы создать новый блюпринт-класс, который наследует от класса **DefaultPawn**, а также новый блюпринт-класс, который наследует от класса **Player-Controller**.

1. На панели **Content Browser** перейдите в папку *Blueprints*.
2. Щелкните правой кнопкой мыши на панели **Content Browser** и выберите пункт меню **Blueprint Class**.
3. В появившемся окне **Pick Parent Class** разверните список **All Classes**.
4. В поле поиска введите **defaultPawn** и выберите класс **DefaultPawn** в появившихся результатах. Нажмите кнопку **Select** в нижней части окна.
5. Задайте новому Pawn-классу имя **Hero_Spaceship**.
6. Щелкните правой кнопкой мыши в окне **Content Browser** и выберите пункт меню **Blueprint Class**.
7. В появившемся окне **Pick Parent Class** разверните список **Common Classes** и выберите **Player-Controller**.
8. Переименуйте новый класс **Player Controller** в **Hero_PC**.

После создания нового Pawn-класса пора разобраться, из чего состоит каждый класс. Дважды щелкните по новому классу **Hero_Spaceship** на панели **Content Browser**, чтобы открыть его в **Blueprint Class Editor** (Редакторе блюпринт-классов).

Перед вами появится иерархия компонентов. По умолчанию в классе DefaultPawn есть три компонента: CollisionComponent, MeshComponent и MovementComponent (рис. 20.4). Эти три компонента отвечают за основные типы поведения, которые регулируются объектом Pawn.

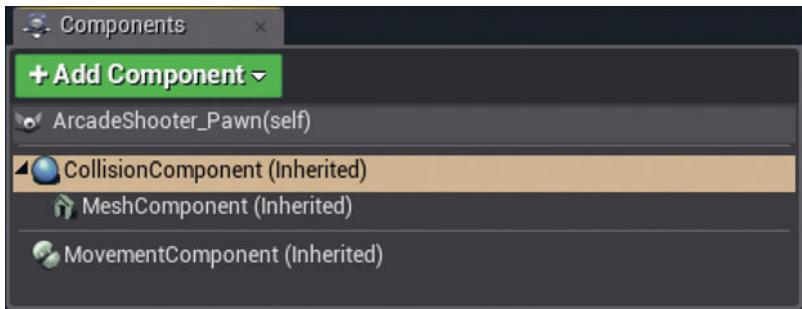


Рис. 20.4. Иерархия компонентов в Blueprint Class Editor для класса DefaultPawn

Компонент **CollisionComponent** обрабатывает физические коллизии аватара и срабатывания триггеров аватара с предметами или актерами на уровне. Компонент коллизии представляет собой физический объем аватара и может принимать форму проще, чем сам аватар. **CollisionComponent** не отображается в игре и не является частью визуального представления аватара.

Компонент **MeshComponent** контролирует визуальные эффекты в игре. В вашей игре класс **MeshComponent** представляет собой сферу, это означает, что визуальное представление вашего аватара является сферой. Вы можете заменить или отредактировать компонент **MeshComponent**, чтобы аватар выглядел так, как вы хотите. Вы можете добавить другие типы компонентов, чтобы изменить визуальные эффекты, например компоненты Particle Emitters, Skeletal Meshes, 2d Sprites и сложные иерархии статичных мешей.

Компонент **MovementComponent** управляет движением аватара. Использование компонента **MovementComponent** весьма удобно для обработки движения игрока. Сложные задачи (например, проверка на коллизии и регулировка скорости) упрощаются именно за счет удобного интерфейса **MovementComponent**.

Пока вы не вносили изменений, поэтому в настоящее время аватар — простая сфера. Вы можете изменить это, полностью заменив компонент **MeshComponent** или при помощи изменения ссылки на статичный меш. В следующей рубрике «Попробуйте сами» мы попробуем импортировать меш НЛО, используемый во многих примерах при работе с UE4, а затем заменим меш текущего аватара на этот меш.

ПОПРОБУЙТЕ САМИ

Поработаем с видом корабля

Ваш новый аватар сейчас — это просто сфера. Выполните следующие действия, чтобы улучшить его внешний вид.

1. В корневой папке на панели **Content Browser** щелкните правой кнопкой мыши и выберите команду **Create Folder**, чтобы создать новую папку.
2. Назовите эту папку **Vehicles**.
3. Откройте папку **Vehicles** и нажмите кнопку **Import**.
4. В диалоговом окне **Import** откройте папку **Hour_20/RawAssets/Models**, предоставленную с книгой.
5. Выберите файл **UFO.FBX** и нажмите кнопку **Open**.
6. В появившемся диалоговом окне **FBX Import Options** оставьте все настройки по умолчанию и нажмите кнопку **Import All**.
7. На панели **Content Browser** выберите команду **Save All** (или нажмите комбинацию клавиш **Ctrl+S**).
8. На панели **Content Browser**, перейдите в папку **Blueprints** и дважды щелкните по блупринт-классу **Hero_Spaceship**, чтобы открыть его в **Blueprint Class Editor**.
9. Если в редакторе отображается только панель **Class Defaults**, то в примечании под заголовком панели, нажмите на ссылку **Open Full Editor Blueprint**.
10. На панели **Components** выберите компонент **MeshComponent**; на панели **Details** выберите раскрывающийся список **Static Mesh**, введите **UFO** в поле поиска и выберите **UFO UAsset** из результатов поиска.
11. На панели **Details** присвойте свойству **Scale** преобразования **0.75, 0.75, 0.75**, чтобы наше НЛО уместилось внутри радиуса **CollisionComponent**.
12. На панели инструментов нажмите кнопку **Compile** и кнопку **Save**.

Управление движением объекта Pawn

В UE4 управление движением объекта Pawn реализуется очень просто. Поскольку Pawn-классы наследуются от класса **DefaultPawn**, вся подготовительная работа уже сделана. Чтобы оценить всю простоту управления движением объекта Pawn, вы можете испытать свою работу.

Во-первых, нужно заставить Game Mode создать игрока, используя новый класс **Hero_Spaceship** по умолчанию. Вы можете сделать это на панели **Defaults** класса в редакторе блупринт-классов Game Mode или в разделе **Maps & Modes** на панели **Project Settings**.

В следующем разделе «Попробуйте сами» мы установим класс Hero_Spaceship как Pawn-класс по умолчанию в ArcadeShooter_GameMode. Мы также установим класс Player Controller в Hero_PC.

ПОПРОБУЙТЕ САМИ

Установка классов DefaultPawn и PlayerController

Game Mode должен знать, какой объект Pawn и контроллер игрока должен порождаться в начале игры. Выполните следующие действия, чтобы сделать необходимые настройки.

1. На панели **Content Browser** перейдите в папку *Blueprints* и дважды щелкните по UAsset класса **ArcadeShooter_GameMode**.
2. На панели **Defaults** класса в категории **Classes** найдите свойство **Default Pawn Class** и нажмите стрелку вниз.
3. Выберите класс блюпринта **Hero_Spaceship**.
4. На панели класса **Defaults** нажмите на стрелку вниз рядом со свойством **Player Controller** и выберите класс блюпринта **Hero_PC**.
5. На панели инструментов нажмите кнопки **Compile** и **Save**.

Когда Hero_Spaceship будет установлен, как Pawn-класс по умолчанию в Game Mode, вы сможете проверить, как движется ваш аватар. На панели инструментов **Level Editor** нажмите кнопку **Play**, как показано на рис. 20.5. Когда игра запустится, используйте клавиши со стрелками или клавишу **W**, **A**, **S** или **D** для передвижения. Вы также можете использовать мышь, чтобы двигать камеру. Когда наиграетесь, нажмите клавишу **Esc**, чтобы остановить игру.

ВНИМАНИЕ

Использование Player Start

Если ваша игра не работает, причина может быть в том, что на сцене нет актера Player Start. Если вы не видите Player Start на панели **World Outliner**, то можете легко добавить его, выбрав пункт меню **Modes** ⇒ **Basic** ⇒ **Player Start** и перетащив его в мир. Не забудьте повернуть его в том направлении, в котором аватар должен смотреть по задумке!



Рис. 20.5. Нажмите кнопку **Play** на панели инструментов, чтобы мгновенно проверить свою игру, не выходя при этом из редактора

Хотя аватар может свободно перемещаться, кое-что в игре явно не соответствует задумке проекта. Во-первых, используется камера от первого лица, а должен быть вид сверху с неподвижной камерой. Во-вторых, аватар движется вперед и назад, а также влево и вправо. Теперь нам придется изменить функции, которые UE4 предоставил в готовом виде, и задать правила, чтобы проект соответствовал идеи.

Отключение движения по умолчанию

В классе DefaultPawn многое задано автоматически, но в нашем случае требуется больше ручной настройки. К счастью, это довольно просто. На панели **Defaults** класса DefaultPawn есть свойство **Add Default Movement Bindings**, которое выбрано по умолчанию. Сбросив флагок с этого свойства, вы можете отключить базовое движение класса DefaultPawn и перенастроить его поведение и привязки под свои нужды (рис 20.6).

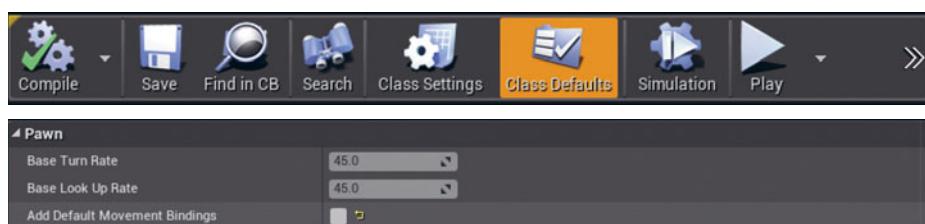


Рис. 20.6. В поле Class Defaults в Pawn отключите свойство Add Default Movement Bindings

ПОПРОБУЙТЕ САМИ

Отключение движения по умолчанию

В игре, которую мы создаем, аватар по умолчанию умеет делать больше, чем нужно. Выполните следующие действия, чтобы отключить лишние возможности с помощью настроек блюпринт-класса Hero_Spaceship.

- На панели **Content Browser** перейдите в папку *Blueprints* и дважды щелкните по классу блюпринта **Hero_Spaceship**.
- На панели **Class Defaults** в категории **Pawn**, убедитесь, что флагок **Add Default Movement Bindings** снят, и эта функция, соответственно, выключена.
- На панели инструментов нажмите кнопку **Compile** и кнопку **Save**.
- Запустите игру и обратите внимание, что вы теперь не можете двигаться. Камера все еще работает от первого лица, но ваш корабль зафиксирован в точке спауна.

Настройка входных воздействий и маппинг осей

Неподвижный космический корабль — это не совсем то, что мы планировали. Мы как-то резко перешли от полной свободы перемещения к полному ограничению, поэтому кое-какие средства управления нужно вернуть. Начнем с привязки нажатия клавиш клавиатуры к различным действиям. Принятие входного воздействия — например, движения джойстика, нажатия клавиши или кнопки геймпада — и связь определенного действия с этим воздействием называется *привязкой*, которая настраивается на уровне проекта.

Чтобы задать входные привязки, выберите команду меню **Settings** ⇒ **Project Settings**, а затем откройте раздел **Input** панели **Project Settings**. В верхней части этого раздела в разделе **Bindings** есть два списка: **Action Mapping** и **Axis Mapping**. Разница между этими двумя разделами тонкая, но важная. **Action Mapping** (маппинг действий) применяется для разовых нажатий определенной кнопки. Этот тип, как правило, используется для прыжков, стрельбы и других дискретных событий. **Axis Mapping** (маппинг осей) применяется для непрерывных вводов, таких как движение, поворот и управление камерой. Оба типа маппинга могут быть использованы одновременно, и выбор правильного типа привязки для ваших действий сделает создание сложных взаимодействий игроков легче.

Маппинг осей может работать по-разному, в зависимости от оборудования, генерирующего входной сигнал. Некоторые устройства (например, мыши, джойстики и геймпады) возвращают входные значения в UE4 в диапазоне от -1 до 1. UE4 позволяет масштабировать это значение в зависимости от того, как сильно тот или иной сигнал должен влиять на игру по задумке пользователя. Но на клавиатурах движения вверх и вниз и влево и вправо контролируются отдельными клавишами, и это не позволяет получить непрерывный диапазон входного сигнала. Клавиша может быть либо нажата, либо нет, поэтому, когда вы используете для маппинга осей клавиши, UE4 должен быть в состоянии интерпретировать значение клавиши в том же диапазоне от -1 до 1.

Для движения используется маппинг осей, а в нашем аркадном шутере нужно ограничить движение игрока одной осью, чтобы и игрок мог двигаться только влево или вправо. В следующей рубрике «Попробуйте сами» мы зададим входные привязки для движения аватара вправо и влево.

ПОПРОБУЙТЕ САМИ

Создание набора маппинга MoveRight

В этом упражнении мы создадим игру, готовую принимать входные сигналы от пользователя. Привяжите все необходимые клавиши и левый джойстик геймпада к движению влево или вправо. Любые привязки, которые позволяют пользователю перемещаться влево, а не вправо, должны иметь значение масштаба -1.0.

1. Выберите пункт меню **Edit** ⇒ **Project Settings**.
2. На панели **Project Settings** выберите категорию **Input**.
3. В категории **Bindings** найдите свойство **Axis Mappings** и нажмите значок «+» рядом с ним.
4. Разверните поле **Axis Mappings**, нажав на стрелку слева от него, и задайте имя **MoveRight**.
5. Нажмите на стрелку слева от привязки **MoveRight**, чтобы развернуть список привязок.
6. Нажмите на значок рядом с полем **MoveRight** четыре раза, чтобы создать пять маппингов с названием **None**.
7. Нажмите на стрелку вниз рядом с каждым полем **None** и отредактируйте каждое поле, как показано на рисунке ниже.
8. Убедитесь в том, что каждое свойство **Scale** установлено так, как показано на рис. 20.7.



Рис. 20.7. Параметры **Axis Mappings** для привязки **MoveRight**, которые разделены на три части: имя маппинга, клавиша или ось, которая в настоящее время привязана, и количество положительного или отрицательного входного сигнала, который накапливается каждую секунду

В верхней части свойств **Axis Mappings** на панели **Project Settings** есть поле, где можно ввести название для действия, которое должно быть выполнено. Щелкните по значку «+» рядом с названием действия, чтобы добавить новую привязку. Каждая привязка имеет две части: входной сигнал, который в настоящее время привязан, и масштаб входного сигнала, влияющий на результат.

Нам нужно, чтобы игра обрабатывала нажатия клавиш **A** и **D** в виде непрерывной оси. Чтобы сделать это, вам нужно сделать так, чтобы одна из них имела отрицательный масштаб. Иными словами, когда вы жмете «влево», значение оси уменьшается, а при нажатии вправо — увеличивается.

Для джойстиков (например, Gamepad Left Thumbstick X-Axis) отрицательные значения уже рассчитаны, поэтому шкала остается без изменений, 1.0.

В этом примере клавиши **A** и **D**, стрелка влево и стрелка вправо, а также левый джойстик геймпада привязаны к действию MoveRight. Это дает нам понять важную вещь: используя маппинг осей или маппинг действий, мы по-разному определяем одно и то же событие. Это означает, что в вашем проекте будет меньше тестирования и дублирования сценариев блупринтов, и это означает, что проект становится более удобным для чтения. Вместо того чтобы сценарий блупримита проверял нажатие клавиши, он может просто обновлять информацию о движении, когда срабатывает событие MoveRight.

Но просто создать входной сигнал не значит оживить объект. Теперь нам нужно использовать созданное действие MoveRight.

Использование событий ввода для перемещения аватара

Теперь вы готовы вернуть объекту движение. У вас есть отличная входная ось под названием MoveRight и аватар, снова готовый двигаться. Сперва вам нужно открыть редактор блупримита-класса Pawn и перейти на панель Event Graph. Здесь вы можете задать поведение объекта для ситуации, когда срабатывает ваше действие MoveRight.

На панели Event Graph, щелкнув правой кнопкой мыши и найдя созданное действие MoveRight, вы увидите на графике событие InputAxis MoveRight, как показано на рис. 20.8.

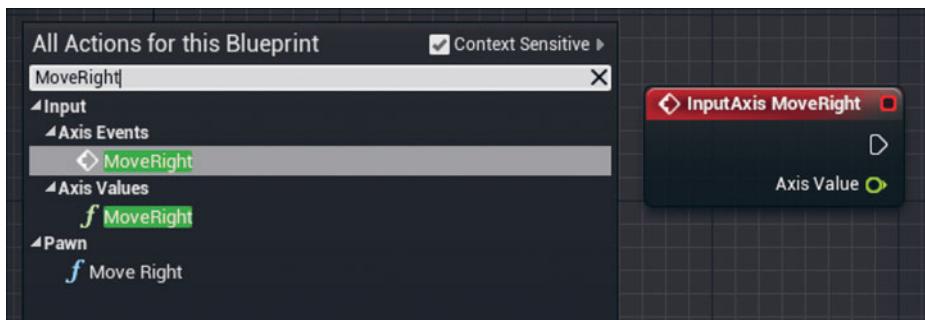


Рис. 20.8. Отображение маппингов оси в разделе Axis Events. Здесь также содержатся функции Axis Values и Pawn, но в данный момент они нам не нужны

Когда у вас есть событие оси, вы можете запросить значение оси и преобразовать его в движение. Для этого нужно еще несколько нодов блюпринтов, например Add Movement Input.

Эта функция работает с **MovementComponent**. Она интерпретирует значение и направление относительно мира для перемещения аватара.

Подключив выполнение события InputAxis MoveRight и возвращаемое в Axis Value значение к Add Movement Input, вы позволите MovementComponent получать входные данные от игрока и перемещать аватар.

Так как вы хотите, чтобы космический корабль по входному сигналу двигался влево или вправо, вам нужно взять вектор правой оси аватара. Вы можете получить этот вектор с помощью нода Get Actor Right Vector и вставить его возвращаемое значение во вход World Direction нода Add Movement Input (рис. 20.9).



Рис. 20.9. Готовый график Add Movement Input

ПОПРОБУЙТЕ САМИ

Привязка маппинга оси MoveRight к движению аватара

При наличии маппинга оси MoveRight, объект Pawn должен знать, как интерпретировать значения, которые приходят из маппинга. Выполните следующие действия, чтобы составить простой график, позволяющий указать ему, как двигаться при получении входных сигналов от игрока.

1. На панели **Content Browser** перейдите в папку *Blueprints* и дважды щелкните по классу блюпринта **Hero_Spaceship**, чтобы открыть редактор класса.
2. Щелкните правой кнопкой мыши по пустому пространству панели **Event Graph** и введите *moveright* в поле поиска.
3. Выберите пункт меню **Axis Events** ⇒ **MoveRight** из результатов поиска.

4. Перетащите контакт вывода нода InputAxis MoveRight и соедините его с контактом ввода Add Movement.
5. Возьмите контакт вывода Axis Value нода события InputAxis MoveRight и соедините его с контактом ввода Add Movement Input.
6. Щелкните по контакту ввода World Direction нода Add Movement, перетащите его и поместите нод Get Actor Right Vector.
7. На панели инструментов нажмите **Compile** и **Save**.
8. Когда граф будет готов, снова протестируйте игру. При нажатии любой из клавиш ввода (**A**, **D**, **←**, **→**) или левого джойстика совместимого геймпада камера будет перемещаться вправо или влево.

СОВЕТ

Прелести умалчиваемых значений объектов Pawn

В нашей игре используется ввод Add Movement, который задает направление в глобальных координатах. Эта мощная функция позволяет перемещать аватара в любом направлении. Однако в классе DefaultPawn есть некоторые удобные функции как раз на этот случай. Попробуйте заменить Add Movement Input и Get Actor Right Vector функцией MoveRight класса DefaultPawn. Вы получите точно такой же результат, но граф будет выглядеть гораздо проще, как показано на рис. 20.10.

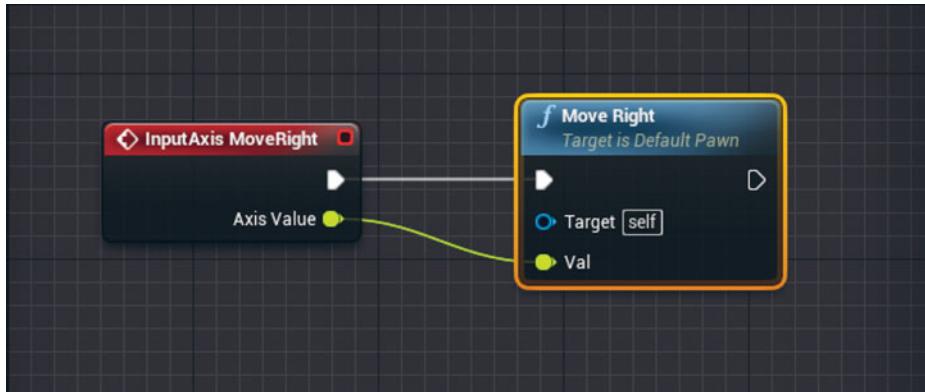


Рис. 20.10. Другой вариант графа с функцией MoveRight класса DefaultPawn вместо нода Add Movement Input

Настройка фиксированной камеры

В настоящий момент камера в вашей игре следует за аватаром. Такая опция задана по умолчанию, но для нашей игры нужно кое-что другое, чтобы камера

оставалась неподвижной и смотрела на космический корабль сверху. И еще она не должна двигаться, когда движется аватар.

Чтобы решить эту задачу, вы можете использовать актеров камеры и точки обзора, а также встроенный класс PlayerController, который задает поле обзора игрока. Эта настройка может быть сделана на уровне блюпринта, но это затруднит перевод логики вашей игры на новый уровень. Вместо этого нужно привязать эту логику камеры к Game Mode. Когда игра начинается, Game Mode создаст камеру для вашей игры, а затем даст классу PlayerController команду использовать эту новую камеру.

В следующем разделе «Попробуйте сами» мы используем событие BeginPlay и нод SSpawn Actor from Class, чтобы создать новую камеру и задать ее положение с помощью нода Make Transform перед установкой его в качестве точки обзора для класса PlayerController.

ПОПРОБУЙТЕ САМИ

Создание и настройка камеры с фиксированным положением

Выполните следующие шаги, чтобы ArcadeShooter_GameMode мог создать новую камеру и установить ее в качестве точки обзора для класса PlayerController.

1. На панели **Content Browser** перейдите в папку *Blueprints* и дважды щелкните по блюпринту-классу **ArcadeShooter_GameMode**, чтобы открыть редактор блюпринт-классов.
2. Если редактор выводит только панель **Class Defaults**, то в примечании под заголовком панели нажмите на ссылку **Full Open Editor Blueprint**.
3. В окне **EventGraph** найдите нод Event BeginPlay. Если такого нода не существует, создайте его, щелкнув правой кнопкой мыши и введя в строку поиска фразу *begin play*.
4. Щелкните по ехес-контакту вывода нода Event BeginPlay, перетащите его и поместите нод **SPawn Actor from Class**.
5. В ноде **Spawn Actor from class** нажмите на стрелку вниз в поле **Select Class** и выберите вариант **CameraActor**.
6. Перетащите свойство **Spawn transform** нода **SPawnActor CameraActor** и поменяйте его в нод **Make Transform**.
7. Присвойте свойству **Location** нода Make Transform значение **0.0, 0.0, 1000.0**.
8. Присвойте свойству **Rotation** нода Make Transform значение **0.0, —90.0, 0.0**.
9. Справа от нода **SPawnActor CameraActor** создайте нод **Get Player Controller**.
10. Щелкните по контакту вывода **Return Value** нода Get Player Controller, перетащите его и поместите нод **Set View Target with Blend**.
11. Соедините контакт вывода **Return Value** нода **SPawnActor CameraActor** с контактом ввода **New View Target** нода **Set View Target with Blend**.

12. Соедините ехес-контакт вывода нода SPawnActor CameraActor с ехес-контактом ввода Set View Target with Blend. На рис. 20.11 показан готовый граф событий для Game Mode.
13. На панели инструментов нажмите кнопку **Compile** и кнопку **Save**.
14. Выполните еще один тестовый запуск игры. В этот момент камера должна смотреть сверху вниз на аватар, который теперь перемещается влево или вправо, когда вы нажимаете соответствующие клавиши.

Резюме

В этом часе мы узнали, как с нуля создать новый проект в UE4 и как настроить его с помощью пользовательского уровня и вновь созданного Game Mode. Вы узнали, что такое объекты Pawn и контроллеры игрока и как использовать их. Вы также узнали, как отключить движение по умолчанию для класса DefaultPawn и как создать собственное движение и ввод на панели **Project Settings**.

В заключение мы изучили один из способов создания фиксированной камеры.

Вопросы и ответы

Вопрос: Зачем нужно описывать всю логику игры в Game Mode, а не на уровне блюпринта?

Ответ: Нет четкого требования того, где именно должна описываться игровая логика. Вместо этого, полезно представлять разделение, как способ избавить себя от повторяющейся работы в дальнейшем. Вся логика, которая относится к нескольким уровням, вероятно, должна описываться в Game Mode или отдельных актерах, а логика конкретного уровня (как триггеры, которые открывают двери или включают освещение) обычно помещаются в блюпринт уровня. Вы можете поместить всю логику в блюпринт, но, если вы захотите создать новый уровень, вам придется потратить немало времени, чтобы убедиться, что все работает. А можно было бы просто сразу использовать Game Mode.

Вопрос: Обязательно ли Pawn-класс должен наследоваться от Pawn по умолчанию?

Ответ: Вовсе нет! DefaultPawn использовать удобно, но при желании все его функции могут быть воспроизведены достаточно быстро. В UE4 также есть и другие удобные встроенные классы Pawn, такие как Character, который содержит компонент Skeletal Mesh и некоторую логику, касающуюся передвижения.

Вопрос: Располагать камеру, используя необработанные числа, довольно трудно. Обязательно ли мне спаунить камеру именно таким образом?

Ответ: Нет. Как вариант, можно самому разместить камеру на уровне, а затем ссылаться ее положения в сценариях при вызове Set View Target with Blend. Это позволяет вынести логику из Game Mode и делает уровень более завершенным, облегчая режиссерскую работу с камерой.

Вопрос: Мне не нравится скорость, с которой движется аватар. Могу ли я изменить ее?

Ответ: Разумеется. Чтобы изменить скорость движения аватара, откройте Blueprint Class Editor и выберите MovementComponent. На панели Details установите три значения с плавающей запятой, которые зададут максимальную скорость, ускорение и торможение аватара.

Вопрос: Я могу использовать всего один MeshComponent в Pawn-классе Hero_Spaceship?

Ответ: Нет, вы можете использовать любое количество компонентов, которые определят внешний вид аватара. При добавлении нескольких компонентов (или даже при работе с одним компонентом), вы можете отключить моделирование физики ваших компонентов Static Mesh. Вы можете изменить настройки коллизии, щелкнув на отдельном компоненте и найдя свойство Collision Presets на панели Details. Выбор свойства No Collision гарантирует, что физика для компонента моделироваться не будет. Убедитесь, что в компоненте CollisionComponent активны свойства Pawn Preset и Generate Overlap Events. Если вы не сделаете этого, то в следующем часе ничего не будет работать. Кроме того, если хотите отключить коллизии каких-либо отдельных статичных мешей или визуальных компонентов, убедитесь, что сфера в CollisionComponent моделирует визуальные эффекты.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Истинно или ложно высказывание: аватары — это актеры, которыми непосредственно управляет игрок или ИИ.
2. Истинно или ложно высказывание: UE4 автоматически определяет, какой Game Mode использовать, находя его на панели Content Browser.
3. Истинно или ложно высказывание: привязки действий и привязки осей работают только с фиксированными названиями, такими как MoveRight.
4. Истинно или ложно высказывание: привязки осей используются для непрерывных входов, например удерживания клавиши или джойстика.

Ответы

1. Истина. Любой актер на сцене, которым непосредственно управляет игрок или ИИ, называется аватаром.
2. Ложь. Game Mode должен быть установлен либо на панели **Project Settings**, либо на панели настроек мира на уровне.
3. Ложь. В поле **Name** привязки может быть введена любая строка — и это будет работать. Например, вы могли бы заменить MoveRight на Strafe.
4. Истина. Каждый раз, когда требуется непрерывный ввод информации, а не простое переключение, используется именно привязка осей.

Упражнение

В этом упражнении вы попрактикуетесь в создании новых привязок ввода для управления аватаром, в его редактировании и настройке уровня. Привяжем движение аватара вправо или влево к движению мыши, а также добавим на уровень стены и другие эффекты. Затем сделаем пол и стены невидимыми. Следующие шаги выполняются в том же проекте, который вы создали в этом часе.

1. Выберите пункт меню **Project Settings** ⇒ **Input** и, найдя привязку **Move Right**, добавьте новую ось.
2. Установите новую ось на **Mouse X** и присвойте свойству **Scale** значение **1.0**.
3. Запустите вашу игру и посмотрите, как движение мыши влияет на расположение аватара.
4. Выберите компонент **MovementComponent** аватара в редакторе блупринт-класса **Pawn**.
5. Скорректируйте параметры **Max Speed** и **Acceleration** компонента **MovementComponent**, чтобы изменить скорость движения аватара.
6. Выберите пол для вашего уровня и несколько раз продублируйте его, нажав сочетание клавиш **Ctrl+W**.
7. Расположите продублированные полы так, чтобы ограничить левую и правую стороны вашего уровня, чтобы аватар не мог покинуть поле зрения камеры.
8. Выберите все продублированные полы и включите свойство **Actor Hidden in Game** в категории **Rendering**, чтобы все поверхности были невидимы, когда игра будет запущена.

21-Й ЧАС

Создание аркадного шутера: препятствия и бонусы

Что вы узнаете в этом часе

- ▶ Создание базового класса препятствий
- ▶ Как заставить препятствия двигаться
- ▶ Урон аватарам
- ▶ Перезапуск игры после разрушения аватара
- ▶ Создание бонусов здоровья
- ▶ Создание блюпринта, который спаунит других актеров
- ▶ Удаление старых препятствий

В предыдущем часе мы создали новый Game Mode и сделали космический корабль, который может двигаться вперед и назад. Пока наша игра не очень-то похожа на игру. Чтобы это изменить, в этом часе мы добавим в игру кое-какие трудности: препятствия, которые могут повредить и уничтожить космический корабль. Мы также дадим игроку возможность чинить повреждения. В этом часе мы узнаем, как создать блюпринт-класс препятствия, от которого препятствия будут наследовать свойства, затем настроим космический корабль на получение повреждений, узнаем, как создать блюпринт-класс для бонуса, позволяющего починить эти повреждения, и как создать блюпринт-класс, который будет автоматически порождать новых актеров.

ПРИМЕЧАНИЕ

Подготовка к практике

В этом часе мы продолжим работать с проектом ArcadeShooter, который начали в часе 20. Если хотите, можете использовать в качестве отправной точки готовый проект H20_ArcadeShooter, который находится в папке *Hour_20* в файлах, прилагаемых к книге (файлы проектов можно скачать по адресу: http://addons.eksmo.ru/it/UE_24.zip).

После того как вы закончите этот урок, сравните свои результаты с готовым проектом H21_ArcadeShooter, который находится в папке *Hour_21* в файлах, поставляемых с книгой.

Создание базового класса препятствия

В игре должно быть что-то, создающее игроку трудности. Препятствия бывают разных форм и видов. Некоторые из них — актеры, которые просто мешают игроку двигаться дальше, а другие могут привести к повреждениям или изменению поведения игрока. Мы создаем игру про космический корабль, так что астероид будет вполне подходящим препятствием. Вам нужно создать в мире нового актера, у которого будет компонент Static Mesh, отображающий внешний вид, компонент коллизии, позволяющий взаимодействовать с аватаром, и способность двигаться вниз по направлению к игроку.

Так как эта задача решается несколькими способами, можно воспользоваться наследованием блюпринт-класса, чтобы избежать многократного переписывания одной и той же логики. Так как одной из основных особенностей препятствий и бонусов является направленное движение, нужно создать базовый класс, который будет описывать способность двигаться. В разделе «Попробуйте сами» мы создадим новый блюпринт-класс, который будет иметь все необходимые компоненты, которые станут наследоваться в классе для препятствий.

ПОПРОБУЙТЕ САМИ

Настройка базового класса для препятствий

Препятствие может представлять собой просто Static Mesh или же быть сложным, как система ракеты-носителя. Есть множество вариантов, в которых требуются примерно одинаковые базовые функции, так что вы можете создать основные функции, необходимые для движения и столкновения, а затем добавить пользовательскую логику. Выполните следующие действия, чтобы настроить препятствие базового класса.

1. На панели **Content Browser** перейдите в папку *Blueprints*.
2. Щелкните правой кнопкой мыши на панели **Content Browser** и выберите команду **Blueprint Class**.
3. В появившемся окне **Pick Parent Class** выберите пункт **Actor** из категории **Common Classes**.
4. Переименуйте новый блюпринт-класс актера в **Obstacle**. Сохраните его и откройте его в редакторе Blueprint Class Editor.
5. На панели **Components** добавьте новый компонент сферической коллизии.
6. Сделайте компонент сферической коллизии корневым для актера, перетащив компонент на **DefaultSceneRoot**.
7. На панели **Details** для компонента коллизии присвойте свойству **Sphere Radius** значение **50.0**, а свойству **Collision Presets** — значение **Overlap All Dynamic**.
8. На панели **Components** добавьте новый сферический компонент Static Mesh.

9. Выберите новый компонент Static Mesh, который в настоящее время называется Sphere1, и назовите его **StaticMesh**.
10. На панели сведений для StaticMesh присвойте свойству **Collision Presets** значение **No Collision**.
11. На панели **Components** добавьте новый компонент вращения.

Давайте вскользь рассмотрим, что мы сделали в этом практикуме. Во-первых, создали несколько компонентов (как показано на рис 21.1), но не у всех из них есть очевидное назначение.

Компонент Static Mesh понятен; вы должны иметь возможность видеть препятствие, поэтому нужен меш, который будет отображать его. Пока что вы можете использовать Sphere Mesh UE4 по умолчанию, но можно также и импортировать и использовать любой статический меш.

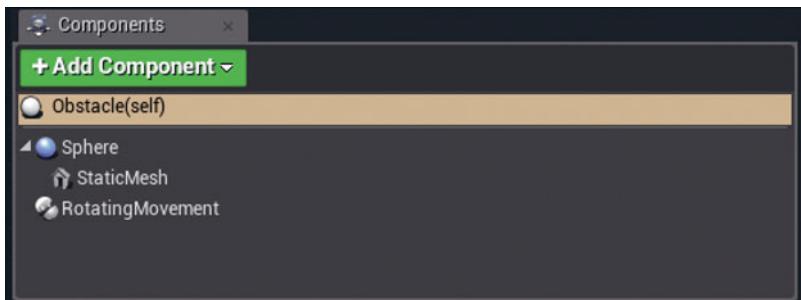


Рис. 21.1. Компоненты блюпринт-класса Obstacle

Компонент сферической коллизии понадобится нам, чтобы обрабатывать все столкновения и пересечения объектов, которые возникнут в игре. При его добавлении вы изменили значение свойства **Collision Presets** на **Overlap All Dynamic**. Вы хотите, чтобы актер знал о столкновении с аватаром, а также вам нужно знать, когда он пересечется с другими препятствиями. Поскольку вам нужно проверять столкновения со сферой, а не со Static Mesh, вы также изменили значение свойства **Collision Presets** компонента **Static Mesh** на **No Collision**.

В 4-м часе «Работа с актерами статичных мешей» более подробно описана работа со свойством Collision Presets.

Наконец, мы добавили компонент вращательного движения, похожий на компонент скользящего движения аватара, который мы добавили в предыдущем часе. Этот компонент содержит поведение, позволяющее актеру вращаться в произвольном направлении. Вы можете использовать его и скрипт конструирования, чтобы препятствие могло вращаться в случайном направлении каждый раз, когда оно порождается.

ПОПРОБУЙТЕ САМИ

Сделаем каждое препятствие уникальным

Если вы прямо сейчас поместите препятствие на уровень, ничего не произойдет. Если вы создадите много его копий на уровне — все они будут выглядеть и вести себя одинаково. Выполните следующие действия, чтобы использовать компонент вращательного движения и мощь скрипта конструирования, чтобы исправить эту ситуацию.

1. На панели **Content Browser** перейдите в папку *Blueprints*.
2. Дважды щелкните по блюпринт-классу препятствия, чтобы открыть его в Blueprint Editor.
3. Перетащите компонент вращательного движения из панели компонентов и поместите его в скрипт конструирования, чтобы создать ссылку на компонент.
4. Щелкните по контакту вывода нода Rotating Movement, перетащите его и поместите под **Set Rotation Node**.
5. Соедините exec-контакт вывода нода Construction Script с exec-контактом ввода нода Set Rotation Rate.
6. Поместите нод **Random Rotator** под нод Set Rotation Rate.
7. Соедините контакт вывода Return Value нода Random Rotator с контактом ввода Rotation Rate нода Set Rotation Rate.
8. На панели **My Blueprint** создайте новую переменную с плавающей запятой и назовите ее **Random Scale Min**.
9. На панели **My Blueprint** создайте новую переменную с плавающей запятой и назовите ее **Random Scale Max**.
10. На панели инструментов нажмите кнопку **Compile**.
11. На панели **Class Defaults** присвойте свойству **Random Scale Min** значение **0.7**.
12. На панели **Class Defaults** присвойте свойству **Random Scale Max** значение **1.5**.
13. Перетащите exec-контакт вывода нода Set Rotation Rate и поместите под **Set Actor Scale 3D**.
14. Поместите нод **Random Float in Range** под нодом Set Actor Scale 3D.
15. С панели **My Blueprint** перетащите переменную **Random Scale Min** на **Random Float** в контакт ввода **Range**.
16. Перетащите переменную **Random Scale Max** в контакт ввода **Max** нода **Random Float in Range**.

17. Соедините контакт вывода нода Random Float in Range с контактом ввода New Scale 3D нода Set Actor Scale 3D. Нод преобразования Float to Vector автоматически появится между двумя нодами.

Теперь у вас есть модифицированный компонент вращательного движения, позволяющий задать случайное вращение при создании каждого актера Obstacle, а размер препятствия может изменяться от 70 до 150% от размера по умолчанию (рис. 21.2).

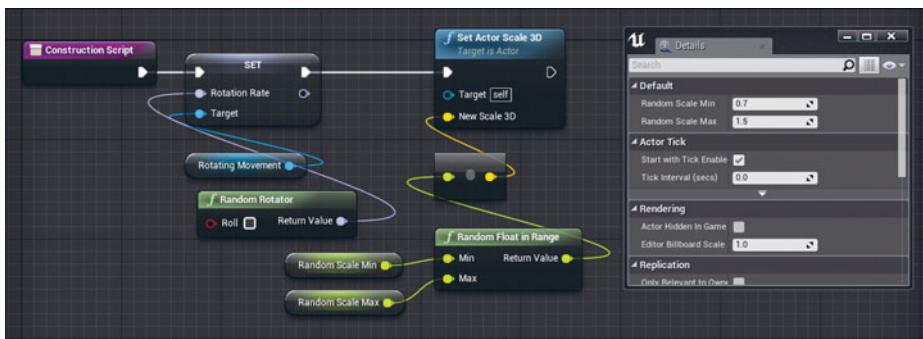


Рис. 21.2. Скрипт конструирования блупринт-класса Obstacle, задающий случайную скорость вращения и случайный масштаб для каждого экземпляра класса Obstacle

Заставляем препятствие двигаться

Мы не можем создать сложную игру из множества вращающихся, но неподвижных астероидов. Вы должны заставить эти препятствия двигаться. Поскольку игра привязана к одной оси, требования достаточно просты, но нам все же придется приложить немного усилий, чтобы все двигалось плавно.

ПРИМЕЧАНИЕ

Перемещение мира

В этом часе мы ограничили движение игрока заданной осью, а перемещать будем мир (или элементы мира) относительно игрока. В готовой версии игры у вас создалась бы иллюзия движения аватара. Это достигается благодаря панорамным эффектам и эффектам частиц. Аналогично можно было бы сделать игру, в которой двигался бы игрок, а не мир.

В этом случае нет правильного или неправильного решения, хотя иногда «подделать» движение игрока с помощью движения мира может упростить обработку поведения и взаимодействия. Или наоборот, реальное перемещение аватара может упростить создание уровня, в зависимости от того, насколько сложен ваш мир.

Для обработки движения аватара вы использовали компонент Flying Pawn Movement. Препятствия и бонусы — это не аватары, и для них достаточно задать простые движения. Вместо добавления компонента можно использовать событие Event Tick, позволяющее перемещать препятствие мимо игрока в каждом фрейме. Кроме того, не хочется, чтобы все препятствия перемещались с одинаковой скоростью, поэтому каждому астероиду задается разная скорость с помощью скрипта конструирования.

В следующем практикуме мы используем событие Event Tick, нод Random Float in Range нод и нод AddActorWorldOffset, чтобы препятствие могло двигаться вперед.

ПОПРОБУЙТЕ САМИ

Препятствие, подвинься!

Ваши актеры-препятствия являются неподвижными, и в этом практикуме мы используем граф событий блупринта и скрипт конструирования, чтобы заставить их двигаться.

- На панели **Content Browser**, перейдите в папку *Blueprints*.
- Дважды щелкните по блупринт-классу препятствия и откройте его в редакторе **Blueprint Class Editor**.
- Щелкните мышью по ссылке **Event Graph**.
- На панели **My Blueprint** создайте новую переменную типа **Float** и назовите ее **Speed Min**.
- На панели **My Blueprint** создайте новую переменную типа **Float** и назовите ее **Speed Max**.
- На панели **My Blueprint** создайте новую переменную типа **Float** и назовите ее **Current Speed**.
- На панели **My Blueprint** создайте новую переменную типа **Vector** и назовите ее **Movement Direction**.
- На панели инструментов нажмите кнопку **Compile**.
- На панели **Class Defaults** присвойте переменной **Speed Min** значение **200**.
- На панели **Class Defaults** присвойте переменной **Speed Max** значение **500**.
- На панели **Class Defaults** присвойте свойству **Movement Direction** значение **-1.0, 0.0, 0.0**.
- Щелкните по контакту вывода **Delta Seconds** нода **Event Tick**, перетащите его и поместите нод **Float * Float**.
- Перетащите переменную **Current Speed** во второй контакт ввода нода **Float * Float**.
- Щелкните по контакту вывода нода **Float * Float**, перетащите его и поместите нод **Vector * Float**.

15. Перетащите переменную Movement Direction во второй контакт ввода нода Vector * Float.
16. Щелкните по ejecs-контакту вывода нода Event Tick, перетащите его и поместите под AddActorWorldOffset.
17. Соедините контакт вывода нода Vector * Float и контакт ввода Delta Location нода AddActorWorldOffset.
18. Щелкните мышью по ссылке **Construction Script**.
19. Щелкните по ноду Set Actor Scale 3D, перетащите его и поместите под **Set Current Speed**.
20. Поместите нод Random Float in Range под нодом Set Current Speed.
21. Перетащите переменную Speed Min в контакт ввода Min нода Random Float in Range.
22. Перетащите переменную Speed Max в контакт ввода Max нода Random Float in Range.
23. Перетащите контакт вывода Return Value нода Random Float in Range на вход Current Speed нода Set Current Speed.

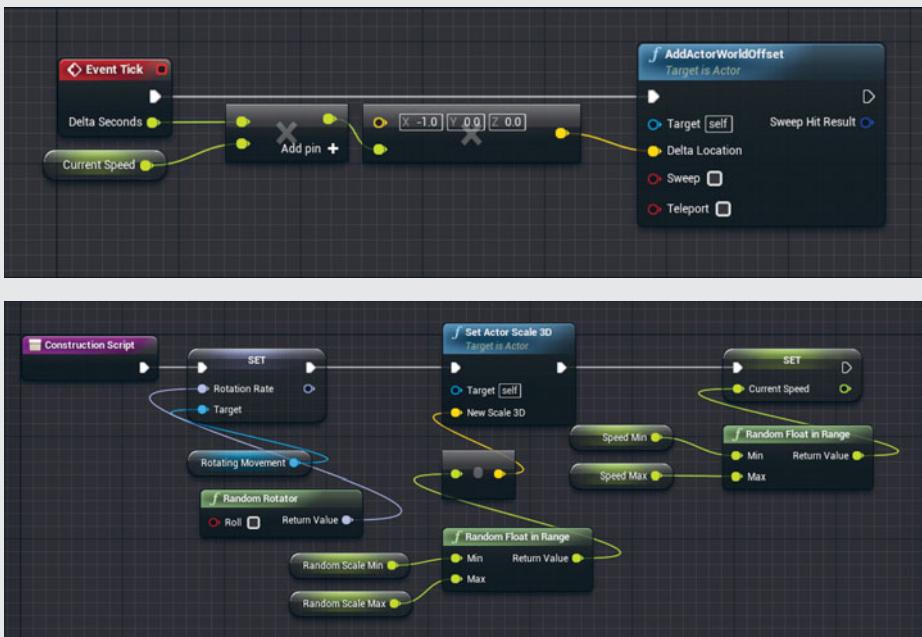


Рис. 21.3. Граф событий и скрипт конструирования совместно позволяют актерам-препятствиям двигаться со случайной скоростью в отрицательном направлении оси X. Эта случайная скорость рассчитывается один раз во время создания каждого актера. Каждый актер будет иметь уникальную скорость, но изменяться для него в процессе игры она не будет

Теперь у каждого препятствия будет своя скорость, но все препятствия будут двигаться в одном и том же направлении, вниз вдоль оси X. Поместите на уровень несколько актеров-препятствий и запустите игру с помощью команды **Play in Viewport**. Все препятствия должны двигаться вниз по экрану.

Вы можете заметить, что можете направить свой космический корабль прямо на препятствия, и ничего вам за это не будет. Это происходит из-за того, что, хотя корабль и препятствия сталкиваются, вы не задали никакого поведения на этот случай.

Теперь, когда вы реализовали большую часть наследуемого поведения в классе **Obstacle**, можете создать класс для астероида. В следующем упражнении «Попробуйте сами» мы создадим новый блюпринт-класс, который будет наследовать от вашего класса **Obstacle Blueprint**.

ПОПРОБУЙТЕ САМИ

Создание дочернего класса для астероида

Большую часть поведения, определенного в базовом классе **Obstacle**, можно заменить, используя наследование. Выполните следующие действия, чтобы создать вариант класса **Asteroid**, дочернего для **Obstacle**, и изменить компонент **Static Mesh** соответствующим образом.

1. На панели **Content Browser**, перейдите в папку *Blueprints*.
2. Щелкните правой кнопкой мыши на панели **Content Browser** и выберите пункт меню **Blueprint Class**.
3. В появившемся окне **Pick Parent Class** разверните категорию **All Classes** и в поле поиска введите **obstacle**.
4. Выберите класс **Obstacle** из списка и нажмите кнопку **Select** в нижней части окна.
5. Переименуйте новый блюпринт-класс **Obstacle** в **Asteriod**.
6. Дважды щелкните по блюпринт-классу **Asteriod**, чтобы открыть его в редакторе **Blueprint Class Editor**.
7. На панели компонентов выберите компонент **Static Mesh**.
8. На панели **Details** щелкните стрелку вниз возле поля **Static Mesh**, введите в поиск строку **sm_rock** и выберите меш **SM_Rock**.
9. Сбросьте свойство **Element 0 Material** на значение по умолчанию, присвоенное компоненту **Static Mesh**, нажав на желтую стрелку **Reset to Base Material** рядом со стрелкой.
10. Присвойте свойству **Location** компонента **Static Mesh** значение **0,0, -30,0**.
11. Присвойте свойству **Scale** компонента **Static Mesh** значение **0,5, 0,5, 0,3**.

Теперь мы создали новый блюпринт, основанный на первоначальном блюпринт-классе `Obstacle`. Мы задали ему статичный меш в виде камня и простую сферу для обработки коллизий. Поскольку мы хотим использовать сферу для обработки всех коллизий, в предыдущем упражнении «Попробуйте сами» было необходимо убедиться, что сфера целиком охватывает ваш камень. Вместо того чтобы изменить размеры сферы коллизии под размер меша, мы масштабируем его, чтобы уместить его внутри сферы. Сделав это, убедитесь, что сфера коллизии астероида и `Static Mesh` имеют примерно одинаковые размеры.

Получение урона аватаром

Получение урона и уменьшение здоровья — частая механика в компьютерных играх. В некоторых играх есть сложные системы для регенерации здоровья, и здоровье персонажа в них отображается подвижными шкалами, эффектами и через пользовательские интерфейсы. В других играх смерть наступает «с одного удара», и у игрока есть несколько жизней — попыток. Некоторые игры сочетают в себе и то, и другое, а урон зависит от состояния.

В нашей игре мы хотим позволить игроку возможность пропустить один удар, но если игрок получит повреждение снова, то игра закончится. Позже мы дадим игроку возможность исцеления, что позволит ему восстановить здоровье. Поскольку мы будем использовать свой класс `Obstacle` в качестве основы для бонусов здоровья и астероидов, нужно внести некоторые различия между ними. Чтобы установить состояние повреждения, требуется свойство объекта `Pawn`, позволяющее контролировать, имеет ли аватар повреждения, а также способ вывести игроку информацию об этом состоянии. В следующих двух упражнениях «Попробуйте сами» мы настроим объект `Pawn` так, чтобы он мог становиться поврежденным, а класс `Asteroid` научим определять попадание в аватара.

ПОПРОБУЙТЕ САМИ

Подготовка функции получения урона и поврежденного состояния

Космос — опасное место. Космический корабль должен знать, поврежден ли он, и должен выводить информацию об этом. Выполните следующие действия, чтобы подготовить граф событий, показанный на рис. 21.4.

1. На панели **Content Browser** перейдите в папку `Blueprints`.
2. Дважды щелкните по блюпринт-классу `Hero_Spaceship`.
3. На панели **My Blueprint** создайте новую логическую переменную и назовите ее `Is Damaged`.

4. На панели инструментов нажмите кнопку **Compile**.
5. Убедитесь в том, что переменная **Is Damaged** по умолчанию имеет значение **False**.
6. На панели **Components** добавьте новый компонент **Particle System** и назовите его **Damage Particle System**, чтобы создать систему частиц, которая будет использоваться, когда ваш космический корабль окажется поврежденным.
7. Присвойте свойству **Template** компонента **Damage Particle System** значение **P_Fire**.
8. Присвойте свойству **Auto Activate** компонента **Damage Particle System** значение **False**.
9. На панели **My Blueprint** создайте новую функцию, нажав кнопку **Add New**, и назовите новую функцию **Take Damage**.
10. На панели **My Blueprint** дважды щелкните по функции **Take Damage**, чтобы открыть граф событий функции.
11. Перетащите ехес-контакт вывода нода **Take Damage** и создайте нод **Branch**.
12. Перетащите переменную **Is Damaged** в контакт ввода **Condition** нода **Branch**.
13. Перетащите контакт вывода **False** нода **Branch** и создайте нод **Set Is Damaged**.
14. Установите контакт вывода нода **Set Is Damaged** на **True**.
15. Перетащите систему частиц **Damage** и поместите его справа от нода **Set Is Damaged**.
16. Перетащите контакт вывода нода **Damage Particle System** и создайте нод **Activate**.
17. Перетащите ехес-контакт вывода нода **Set Is Damaged** в контакт ввода **Activate**.
18. На панели инструментов нажмите кнопку **Compile** и кнопку **Save**.

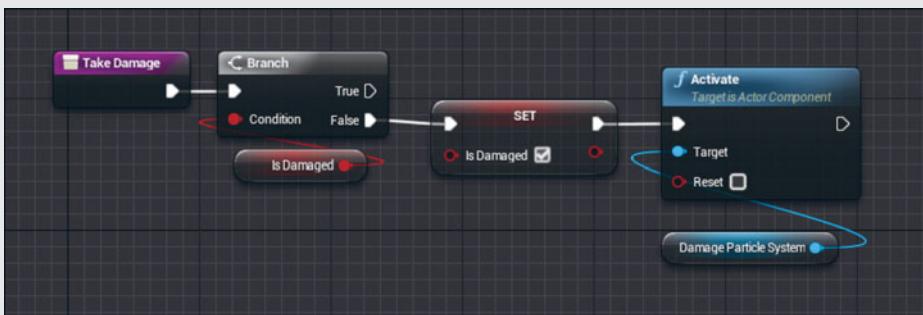


Рис. 21.4. Функция **Take Damage**. Каждый раз, когда объект Pawn получает урон и вызывается эта функция, объект проверяет, был ли он уже поврежден, и, если это не так, присваивает переменной **Is Damaged** значение **True** перед активацией **Damage Particle System**

Мы закончили подготовительную работу по обработке повреждения аватара. Но если вы прямо сейчас нажмете кнопку **Play**, ничего нового не произойдет.

Теперь нам нужно сделать так, чтобы класс актеров-астероидов заставлял аватара получать повреждения. Чтобы сделать это, вы можете использовать два нода: Event ActorBeginOverlap и Cast To. Эти события позволят астероиду определить момент столкновения с игроком. Событие ActorBeginOverlap работает, как событие Event Begin Play или Event Tick, но оно срабатывает каждый раз, когда актер перекрывается другим актером. Оно ссылается на актера, с которым перекрываеться, но вам нужен способ определить, что это именно астероид. В этом случае нам поможет нод Cast To.

Использование нода Cast To

Набор нодов Cast To превращает целевой объект в особый класс, к которому вы пытаетесь получить доступ. При размещении нода Cast To необходимо выбрать конкретный класс, например Cast To Pawn или Cast To Game Mode. Затем нод Cast To пытается привести тип целевого объекта к заданному. Преобразование будет успешно, если объект принадлежит к указанному классу или наследует от него.

Если целевой объект может быть приведен к указанному типу (то есть условие истинно), нод Cast To продолжает выполнение через ехес-контакт вывода Success успеха и возвращает объект в качестве запрашиваемого класса. Здесь можно использовать переменные и функции, которые являются уникальными для запрашиваемого класса.

Если тип целевого объекта не приводим к указанному классу, нод Cast To продолжает выполнение через контакт вывода Failed и возвращает нулевой указатель.

Важно отметить, что никаких дополнительных данных в ноде Cast To не конвертируется. Нод Cast To в некотором роде возвращает вам тот же объект, но интерпретирует его иначе. Целевой объект все еще точно такой же, как и был до приведения типа, но возвращенный ответ несет ту или иную информацию об этом объекте.

Например, класс Hero_Spaceship наследует от класса Actor, но класс Actor не имеет функции под названием Take Damage. Поскольку событие ActorBeginOverlap возвращает перекрываемого актера как ссылку на актера, вы должны использовать нод Cast To Hero_Spaceship, чтобы интерпретировать его как Hero_Spaceship, которым он является на самом деле. Если у вас есть ссылка на Pawn как Hero_Spaceship, вы можете вызвать на нем функцию Take Damage. Любые перекрываемые актеры, которые не являются вашим космическим кораблем, будут проигнорированы.

В следующем разделе «Попробуйте сами» мы используем ноды Event ActorBeginOverlap и Cast To Hero_Spaceship, чтобы вызывать функцию Take Damage для Hero_Spaceship.

ПОПРОБУЙТЕ САМИ

Работаем с перекрытием

Актеры-астероиды должны вызывать функцию Take Damage каждый раз, когда они перекрываются с аватаром, а также они должны уничтожать себя после этого. Выполните следующие шаги, чтобы подготовить граф событий, показанный на рисунке 21.5.

1. На панели **Content Browser** перейдите в папку *Blueprints*.
2. Дважды щелкните по блюпринт-классу *Asteroid*.
3. Щелкните мышью по ссылке **Event Graph**.
4. Перетащите контакт вывода Other Actor нода Event Actor Begin Overlap и создайте нод **Cast To Hero_Spaceship**.
5. Убедитесь в том, что ехес-контакт вывода нода Event ActorBeginOverlap соединен с ехес-контактом ввода нода Cast To Hero_Spaceship.
6. Перетащите контакт вывода As Hero Spaceship нода Cast To Hero_Spaceship и создайте нод **Take Damage**.
7. Убедитесь, что ехес-контакт вывода нода Cast To Hero_Spaceship соединен с ехес-контактом ввода нода Take Damage.
8. Перетащите контакт вывода нода Take Damage и создайте нод **Spawn Emitter at Location**.
9. Установите контакт ввода Emitter Template нода Spawn Emitter at Location на P_Explosion.
10. Поместите нод **GetActorLocation** под нод Spawn Emitter at Location.
11. Соедините контакт вывода Return Value нода GetActorLocation с контактом ввода Spawn Emitter at Location.
12. Перетащите контакт вывода нода Spawn Emitter at Location и создайте нод DestroyActor.
13. На панели инструментов нажмите кнопку **Compile** и кнопку **Save**.
14. Поместите на уровень несколько актеров-астероидов перед местом появления игрока.
15. На панели инструментов уровня нажмите кнопку **Play in Viewport**, чтобы проверить вашу работу. Когда астероиды сталкиваются с нашим кораблем, они должны взрываться и исчезать, а сам корабль — загораться.

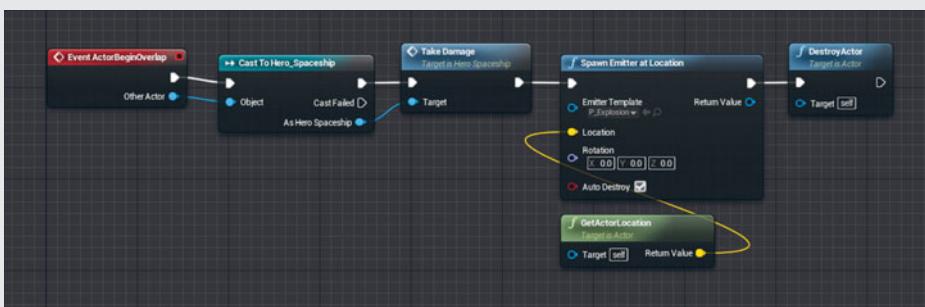


Рис. 21.5. Нод Event ActorBeginOverlap блюпринт-класса Asteroid. Каждый раз, когда другой актер перекрывает астероид, он преобразуется в класс Hero_Spaceship Blueprint. Если он и правда принадлежит классу Hero_Spaceship, Hero_Spaceship получает урон через функцию Take Damage, а астероид спаунит вокруг себя частицы взрыва и уничтожается

ВНИМАНИЕ

Высота размещения препятствий

Если астероиды не сталкиваются с кораблем, причина может быть в том, что они не находятся на одном вертикальном уровне с ним. Самый простой способ исправить это — просто поднять астероиды на ту же высоту, что и аватар.

Перезапуск игры в случае гибели персонажа

Сейчас получение урона в нашей игре носит исключительно косметический характер. Независимо от того, сколько препятствий игрок сшибет на пути, космический корабль не будет уничтожен. Чтобы исправить этот недостаток, вам необходимо изменить функцию Take Damage, чтобы создать состояние смерти персонажа. После этого вы можете перезапустить уровень, используя таймер и функцию Restart Game, которая содержится в Game Mode.

ПРИМЕЧАНИЕ

Таймеры

Иногда необходимо сделать так, чтобы функция или событие срабатывали с некоторой задержкой. Использование таймера весьма удобно для запуска тех или иных действий в определенный момент времени. Например, после того как космический корабль взорвется, нужно будет подождать некоторое время, прежде чем игру можно будет перезапустить. С помощью таймера вы можете гарантировать, что у игрока

будет время рассмотреть эффекты частиц и понять, что произошло, перед тем как игра перезапустится.

В блюпринтах можно установить таймер несколькими способами. Например, вызывать определенные функции по их имени с помощью нода Set Timer by Function. Вы также можете подключить пользовательское событие к ноду Set Timer by Event. Таймер принимает на вход число с плавающей точкой, которое определяет, как долго таймер будет ожидать перед запуском связанного с ним действия.

Таймеры могут работать в цикле через логические переменные в обеих функциях.

В следующем упражнении «Попробуйте сами» мы создадим новую функцию On Death, которая будет вызываться всякий раз, когда космический корабль получит слишком много повреждений. Мы также реализуем таймер, который добавит паузу в несколько секунд перед перезапуском игры с помощью Game Mode.

ПОПРОБУЙТЕ САМИ

Создание состояния смерти

После того как космический корабль столкнется с двумя астероидами, он должен взрываться, а игра — перезагружаться. Выполните следующие действия, чтобы создать граф событий, показанный на рис. 21.6.

1. На панели **Content Browser** перейдите в папку *Blueprints*.
2. Дважды щелкните по блюпринт-классу **Hero_Spaceship**.
3. На панели **My Blueprint** создайте новую функцию, нажав кнопку **Add New**, и назовите новую функцию **On Death**.
4. На панели **My Blueprint** дважды щелкните по функции **On Death**, чтобы открыть график событий функции.
5. Перетащите контакт вывода нода **On Death** и создайте нод **Spawn Emitter at Location**.
6. Поместите нод **GetActorLocation** под нодом **Spawn Emitter at Location**.
7. Соедините контакт вывода **Return Value** нода **GetActorLocation** с контактом входа **Location** нода **Spawn Emitter at Location**.
8. Перетащите контакт вывода нода **Spawn Emitter at Location** и создайте нод **Set Actor Hidden In Game**.
9. Установите контакт входа **New Hidden** нода **Set Actor Hidden In Game** на **True**.
10. Перетащите контакт вывода нода **Set Actor Hidden In Game** и создайте нод **Set Actor Enable Collision**.
11. Перетащите контакт вывода нода **Set Actor Enable Collision** и создайте нод **Set Timer by Function**.
12. Поместите нод **Get Game Mod** под нодом **Set Timer by Function**.

13. Соедините контакт вывода Return Value нода Get Game Mod с контактом ввода Object нода Set Timer by Function.
14. Установите контакт ввода Function Name нода Set Timer by Function на **RestartGame**.
15. Установите контакт ввода Time нода Set Timer by Function на **3.0**.
16. На панели My Blueprint дважды щелкните по функции **Take Damage**, чтобы открыть граф событий функции.
17. Перетащите контакт вывода True нода Branch и поместите под **On Death**.
18. На панели инструментов уровня нажмите кнопку **Play in Viewport**, чтобы проверить вашу работу. Когда астероиды сталкиваются с кораблем, они должны взрываться и исчезать, а сам корабль — загораться. После второго столкновения корабль должен взорваться и исчезнуть, и через три секунды игра перезапустится.

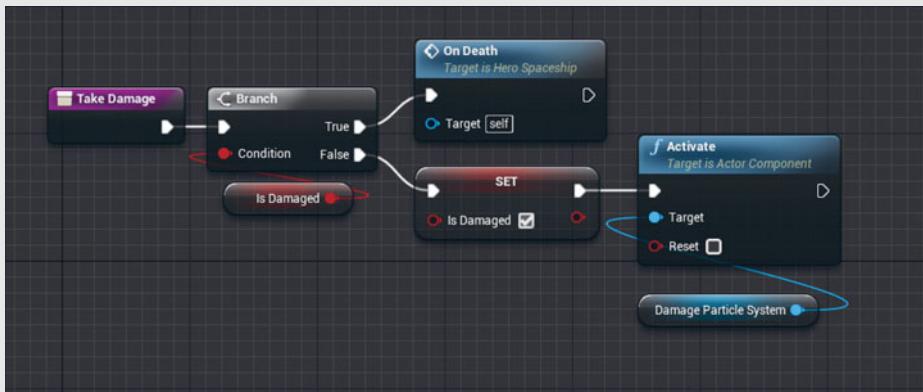


Рис. 21.6. Функции On Death и Take Damage блюпринт-класса Hero_Spaceship. Функция On Death использует таймер для вызова встроенной в Game Mode функции RestartGame. В функции Take Damage нод On Death находится после контакта вывода True нода Branch. Это означает, что функция Take Damage проверяет, был ли аватар поврежден ранее, и, если да, вызывает функцию On Death

Создание бонуса здоровья

Мы создали для игрока возможность получать повреждения, но не дали возможности исцелить их и поправить ситуацию. В этом разделе мы продублируем класс Asteroid и внесем в него изменения, чтобы превратить его в бонус здоровья.

Основное различие между астероидом и бонусом здоровья заключается в том, как он отвечает на столкновение с аватаром. Движение, вращение и коллизию можно оставить без изменений. Поэтому, вместо того чтобы начинать всё с нуля, можно начать с наследования от класса Obstacle, который мы создали ранее.

В ближайших нескольких упражнениях «Попробуйте сами» мы создадим новый класс Health_Pickup, который наследует от класса Obstacle, а затем внесем изменения визуальную составляющую и скрипт конструирования.

ПОПРОБУЙТЕ САМИ

Создаем бонус здоровья

Выжить вашему персонажу довольно трудно. Давайте сделаем ремонтный бонус, который позволит потушить горящий корабль. Выполните следующие действия, чтобы создать новый блюпринт-класс и внешний вид бонуса — крест внутри сферы.

- На панели **Content Browser** перейдите в папку *Blueprints*.
- Щелкните правой кнопкой мыши на панели **Content Browser** и выберите пункт меню **Blueprint Class**.
- В появившемся окне **Pick Parent Class** разверните категорию **All Classes**, а затем введите **obstacle** в поле ввода. Выберите **Obstacle** из списка классов и нажмите кнопку **Select** в нижней части окна.
- Переименуйте новый блюпринт-класс Obstacle в **Health_Pickup**.
- На панели **Class Defaults** установите значение параметра **Random Scale Min** по умолчанию на **1.0**.
- На панели **Class Defaults** установите значение параметра **Random Scale Max** по умолчанию на **1.0**.
- Если в редакторе отображается только панель **Class Defaults**, то в примечании под заголовком панели нажмите на ссылку **Open Full Blueprint Editor**.
- На панели **Components** добавьте два кубических меша, выбрав их в раскрывающемся списке **Add Component**.
- Выберите компонент **Static Mesh** первого куба и присвойте его свойству **Transform Scale** значение **0.2, 0.2, 0.6**.
- Выберите компонент **Static Mesh** второго куба и присвойте его свойству **Transform Scale** значение **0.6, 0.2, 0.2**.

Теперь у нас есть класс Health_Pickup, но пользователю будет непонятно, что это, поскольку он выглядит как обычная сфера. Нам нужно создать специальные материалы, чтобы бонус выглядел так, как мы хотим.

ПОПРОБУЙТЕ САМИ

Создание материала для бонуса

Пользователю пока не очевидно, что именно делает бонус Health_Pickup. Вы можете создать два новых материала, чтобы исправить ситуацию — один для сферы и один для креста внутри нее. Выполните следующие действия для подготовки графа событий материала, как показано на рис. 21.7.

1. На панели **Content Browser** создайте в корневом каталоге новую папку и назовите ее **Pickups**.
2. Щелкните правой кнопкой мыши на панели **Content Browser** и выберите пункт меню **Material**.
3. Переименуйте новый материал в **M_Pickup_Orb**.
4. Дважды щелкните по материалу **M_Pickup_Orb**, чтобы открыть его в редакторе материалов.
5. На панели **Details** присвойте свойству **Blend Mode** значение **Translucent**.
6. На панели **Details**, присвойте свойству **Shading Model** значение **Unlit**.
7. На графе событий материала щелкните правой кнопкой мыши и создайте нод **Constant3Vector**.
8. Соедините контакт вывода нода Constant3Vector с контактом вывода **Emissive Color Pin** нода **M_Pickup_Orb**.
9. На панели **Details** присвойте свойству **Constant** нода **Constant3Vector** значение **0.0, 10.0, 0.0**.
10. На графике событий материала щелкните правой кнопкой мыши и создайте нод **Fresnel**.
11. Соедините контакт вывода нода Fresnel с контактом вывода **Opacity** нода **M_Pickup_Orb**.
12. На панели инструментов нажмите кнопку **Save**.
13. Щелкните правой кнопкой мыши на панели **Content Browser** и выберите пункт меню **Material**.
14. Переименуйте новый материал в **M_Pickup_Cross**.
15. Дважды щелкните по материалу **M_Pickup_Cross**, чтобы открыть его в редакторе материалов.
16. На графике событий материала щелкните правой кнопкой мыши и создайте нод **Constant3Vector**.
17. Соедините контакт вывода нода Constant3Vector с контактом вывода **Base Color** нода **M_Pickup_Cross**.

18. На панели **Details** присвойте свойству **Constant** нода **Constant3Vector** значение **0.0, 1.0, 0.0**.
19. На панели инструментов нажмите кнопку **Save**.

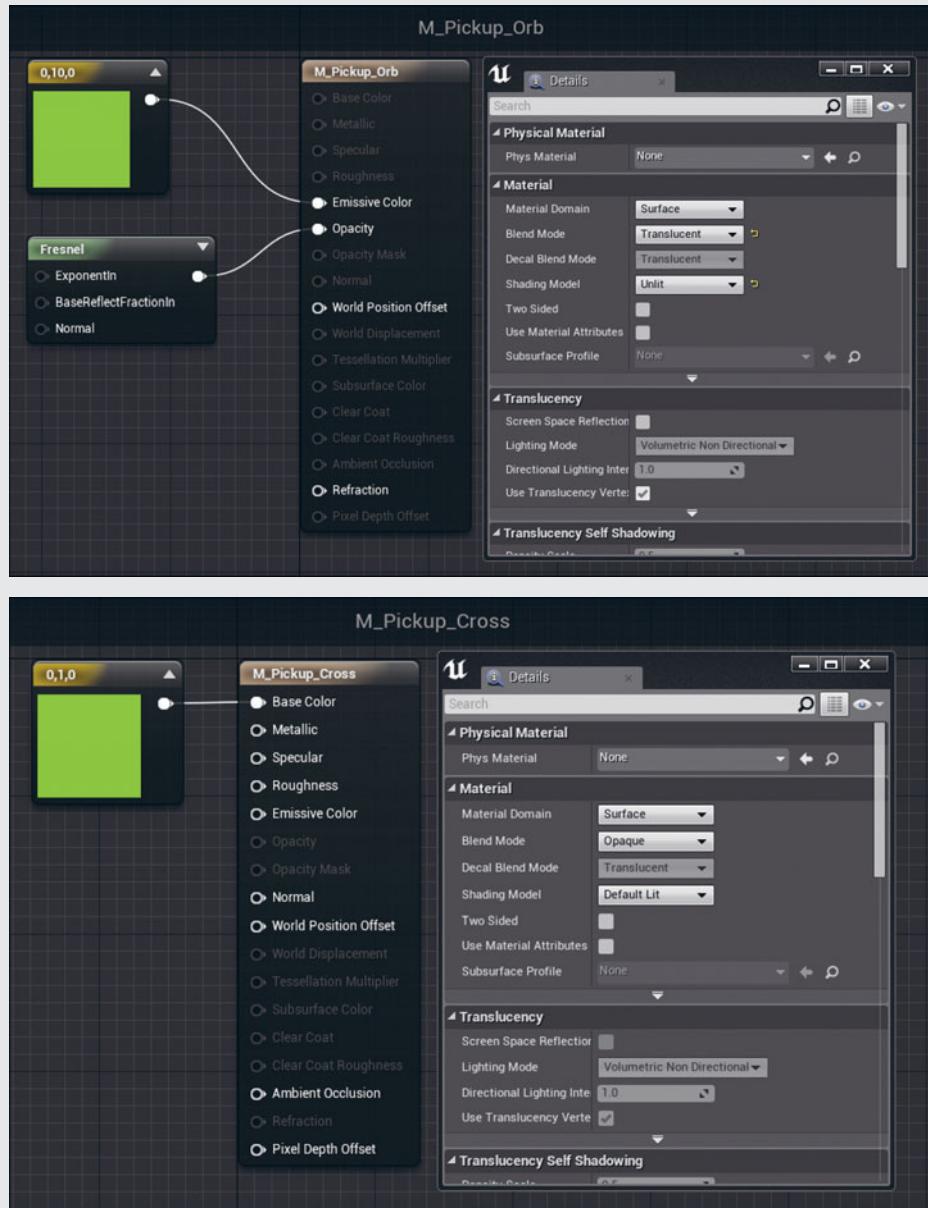


Рис. 21.7. Граф событий и свойства для материалов M_Pickup_Orb и M_Pickup_Cross

Мы создали новые материалы, и теперь нужно настроить их компоненты Static Mesh в блюпринт-классе Health_Pickup.

ПОПРОБУЙТЕ САМИ

Применяем материалы для бонуса

Мы создали материалы, и теперь их нужно применить к различным компонентам Static Mesh в блюпринт-классе Health_Pickup, чтобы бонус выглядел так, как показано на рис. 21.8. Выполните следующие действия.

1. На панели **Content Browser** перейдите в папку *Blueprints*.
2. Дважды щелкните по блюпринт-классу *Health_Pickup*.
3. На панели **Components** выберите компонент **StaticMesh**.
4. Присвойте свойству **Element 0 Material** значение *M_Pickup_Orb*.
5. На панели **Components** выберите первый компонент **Cube**.
6. Присвойте свойству **Element 0 Material** значение *M_Pickup_Cross*.
7. На панели **Components** выберите компонент **Cube1**.
8. Присвойте свойству **Element 0 Material** значение *M_Pickup_Cross*.
9. На панели инструментов нажмите кнопку **Compile** и кнопку **Save**.



Рис. 21.8. Бонус здоровья с соответствующими материалами, присвоенными каждому компоненту Static Mesh

Закончив с визуальными эффектами, давайте добавим логику игры, которая позволит бонусу Health_Pickup влиять на состояние объекта Pawn Is Damaged. Вы можете сделать это по той же схеме, по которой вы настраивали астероид на повреждения,

но в этот раз мы создадим функцию Heal Damage, задающую переменной Is Damaged значение False и отключает компонент Damage Particle System.

В следующих двух упражнениях «Попробуйте сами» мы создадим функцию Heal Damage, благодаря которой аватар сможет излечить себя и отключить частицы огня, а затем используем Event ActorBeginOverlap для блонпринт-класса Health_Pickup, чтобы вызвать функцию Heal Damage.

ПОПРОБУЙТЕ САМИ

Создаем функцию Heal Damage

Когда бонус Health_Pickup создан и движется как задумано, нам нужно настроить функцию, позволяющую починить космический корабль. Выполните следующие действия, чтобы создать график событий, показанный на рис. 21.9.

1. На панели **Content Browser** перейдите в папку *Blueprints*.
2. Дважды щелкните по блонпринту-классу **Hero_Spaceship**.
3. На панели **My Blueprint** создайте новую функцию, нажав кнопку **Add New**, и назовите новую функцию **Heal Damage**.
4. На панели **My Blueprint** дважды щелкните по функции **Heal Damage**, чтобы открыть ее график событий.
5. Перетащите контакт вывода нода Heal Damage и создайте нод **Set Is Damaged**.
6. Убедитесь в том, что контакт ввода Is Damaged нода Set Is Damaged установлен на значение **False**.
7. Перетащите компонент системы повреждения частиц и поместите его справа от нода Set Is Damaged.
8. Перетащите контакт вывода опорного Damage Particle System и создайте нод **Deactivate**.
9. Соедините контакт вывода нода Set Is Damaged с контактом ввода нода Deactivate.
10. На панели инструментов нажмите кнопку **Compile** и кнопку **Save**.



Рис. 21.9. Граф событий функции Heal Damage блонпринта-класса Hero_Spaceship

Когда функция Heal Damage готова, надо добавить бонусу вызов этой функции для исцеления аватара.

ПОПРОБУЙТЕ САМИ

Вызов функции Heal Damage при перекрытии

Космический корабль теперь может починиться, и блюпринт-класс Health_Pickup должен позволять передавать объекту Pawn информацию о том, что он должен излечиться. Выполните следующие действия, чтобы подготовить граф событий, приведенный на рис. 21.10.

1. На панели **Content Browser** перейдите в папку *Blueprints*.
2. Дважды щелкните по блюпринт-классу **Health_Pickup**.
3. Щелкните мышью по ссылке **Event Graph**.
4. Перетащите контакт вывода **Other Actor** нода **Event ActorBeginOverlap** и создайте нод **Cast To Hero_Spaceship**.
5. Соедините контакт вывода нода **Event ActorBeginOverlap** с контактом ввода **Cast To Hero_Spaceship**.
6. Перетащите контакт вывода нода **Cast To Hero_Spaceship** нода **Cast To Hero_Spaceship** и создайте нод **Heal Damage**.
7. Соедините контакт вывода нода **Cast To Hero_Spaceship** с контактом ввода нода **Heal Damage**.
8. Перетащите контакт вывода нода **Heal Damage** и создайте нод **DestroyActor**.
9. На панели инструментов нажмите кнопку **Compile** и кнопку **Save**.
10. Поместите актера **Asteroid** на уровне перед местом появления игрока, а чуть дальше поместите бонус **Health_Pickup**.
11. На панели инструментов уровня нажмите кнопку **Play in Viewport**, чтобы проверить вашу работу. Когда Asteroid столкнется с **Hero_Spaceship**, космический корабль загорится, а затем, когда **Health_Pickup** столкнется с **Hero_Spaceship**, огонь должен погаснуть.

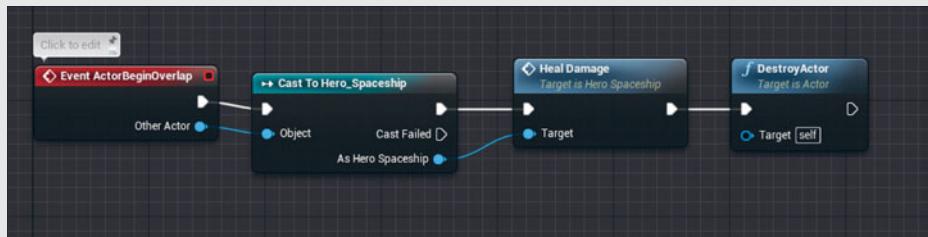


Рис. 21.10. Граф событий функции **Event ActorBeginOverlap** класса **Health_Pickup**, вызывающий функцию **Heal Damage** при перекрытии **Hero_Spaceship**

Спаун актеров

Сейчас вы можете создать игру, только разместив вручную множество астероидов и бонусов на заранее определенном уровне. Вполне допустимо создавать уровни таким образом, и во многих играх используется ручное размещение элементов, что позволяет детально передать замысел создателя. Но нам нужен менее трудоемкий способ создания уровня.

Простой способ сделать это — создать блюпринт-класс, который будет спаунить других актеров. Этот «породитель» или «генератор» будет определять, как часто появляется новый актер, и станет случайным образом выбирать различные типы элементов. В нашей игре нужно случайное появление астероидов и бонусов здоровья.

Создание системы, которая будет постоянно ожидать случайное (но контролируемое) количество времени между порождением новых актеров, может быть довольно затруднительно. Есть несколько способов решить эту задачу, например с помощью таймеров и событий или функции, или с помощью функций с рекурсией*. Один простой, не рекомендуемый метод заключается в том, чтобы ввести таймер обратного отсчета и использовать событие Event Tick для проверки, пришло ли время порождать новый объект. Идея в том, что в вещественную переменную сохраняется случайно заданное количество времени, и время фрейма вычитается из него с каждым фреймом. Когда переменная достигнет нуля, будет порождаться актер, и задаваться новое случайное время отсчета. Это можно продолжать до бесконечности. Проблема этой системы заключается в том, что она требует довольно большого количества ресурсов, и при частом использовании может привести к проблемам производительности в игре. Также такую систему трудно поддерживать. Работать с обратным отсчетом по 1 счету довольно просто, но становится сложно управлять потоком игры, когда все в ней отсчитываются в виде таких событий.

Вместо создания основанной на счете системы мы можем использовать мод Set Timer by Function Name и пользовательские функции, что позволит создать бесконечный поток актеров. В следующих упражнениях «Попробуйте сами» мы создадим блюпринт-класс, который спаунит других актеров и случайным образом решает, какого актера спаунить. Мы создадим новую функцию, которая использует мод Set Timer by Function Name, чтобы спаунить новых актеров.

* Рекурсия — вызов функции из нее же самой. — Прим. ред.

ПОПРОБУЙТЕ САМИ

Подготовка функции порождения

В вашей игре есть актеры. От одних надо уворачиваться, другие надо ловить. Но не стоит размещать их все вручную. Вместо этого следует переложить тяжелую работу на UE4. Выполните следующие действия, чтобы создать граф событий, показанный на рисунке 21.11.

1. На панели **Content Browser** перейдите в папку *Blueprints*.
2. Щелкните правой кнопкой мыши и создайте новый блупринт-класс, родителем которого будет класс **Actor**. Назовите новый класс **Obstacle_Spawner**.
3. Дважды щелкните по блупринт-классу **Obstacle_Spawner**.
4. На панели **My Blueprint** создайте новую переменную с плавающей запятой и назовите ее **Spawn Time Min**.
5. На панели **My Blueprint** создайте новую переменную с плавающей запятой и назовите ее **Spawn Time Max**.
6. На панели **My Blueprint** создайте новую переменную с плавающей запятой и назовите ее **Health Pickup Probability**.
7. На панели инструментов нажмите кнопку **Compile**.
8. На панели **Class Defaults** присвойте переменной **Spawn Time Min** значение **5.0**.
9. На панели **Class Defaults** присвойте переменной **Spawn Time Max** значение **10.0**.
10. На панели **Class Defaults** присвойте переменной **Health Pickup Probability** значение **0.1**.
11. На панели **My Blueprint** создайте новую функцию, нажав кнопку **Add New**, и назовите новую функцию **Spawn**.
12. На графе событий перетащите контакт вывода нода **Event BeginPlay** и создайте нод **Set Timer by Function Name**.
13. Установите контакт ввода **Function Name** нода **Set Timer by Function Name** на **Spawn**.
14. Поместите нод **Random Float in Range** под нодом **Set Time by Function Name**.
15. Перетащите переменную с плавающей запятой **Spawn Time Min** на контакт ввода **Min** нода **Random Float in Range**.
16. Перетащите переменную с плавающей запятой **Spawn Time Max** на контакт ввода **Max** нода **Random Float in Range**.
17. Соедините контакт вывода **Return Value** нода **Random Float in Range** с контактом ввода **Time** нода **Set Timer by Function Name**.
18. На панели инструментов нажмите кнопку **Compile** и кнопку **Save**.

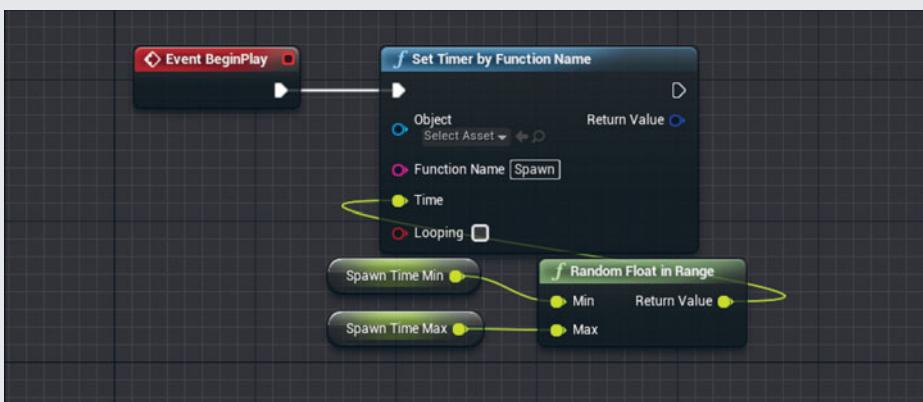


Рис. 21.11. Событие Event BeginPlay блюпринт-класса **Obstacle_Spawner**,зывающее функцию **Spawn** со случайным временем в диапазоне между минимальным и максимальным значениями

В настоящий момент функция **Spawn** не содержит логики, так что, если мы проверим игру, ничего нового не увидим. Теперь необходимо заполнить функцию **Spawn**, чтобы она постоянно и бесконечно порождала новых актеров.

ПОПРОБУЙТЕ САМИ

Выбираем, какой класс спаунить

Функция **Spawn** пока не делает ничего, но вы можете использовать нод **Branch**, чтобы определить, какой блюпринт-класс порождать, а затем установить таймер, чтобы создать рекурсивную схему. Выполните следующие действия, чтобы создать граф событий, показанный на рис. 21.12.

- На панели **Content Browser** перейдите в папку *Blueprints*.
- Дважды щелкните по блюпринт-классу **Obstacle_Spawner**.
- На панели **My Blueprint** дважды щелкните по функции **Spawn**, чтобы открыть ее граф событий.
- Перетащите контакт вывода **Spawn** и создайте нод **Branch**.
- Поместите нод **Random Float** под нодом **Branch**.
- Перетащите контакт вывода **Return Value** нода **Random Float** и создайте нод **Float ⇒ Float**.
- Перетащите переменную **Health Pickup Probability** на второй контакт ввода нода **Float ⇒ Float**.

8. Соедините контакт вывода нода **Float ⇒ Float** с контактом ввода **Condition** нода **Branch**.
9. Перетащите контакт вывода **True** нода **Branch** и создайте нод **Spawn Actor by Class**.
10. Установите контакт ввода **Class** нода **Spawn Actor by Class** на **Health_Pickup**.
11. Поместите нод **GetActorTransform** под нодом **SpawnActor Health Pickup**.
12. Соедините контакт вывода **Return Value** нода **GetActorTransform** с контактом ввода **Spawn Transform** нода **SpawnActor Health Pickup**.
13. Перетащите контакт вывода **False** нода **Branch** и создайте нод **Spawn Actor by Class**.
14. Установите контакт ввода **Class** нода **Spawn Actor by Class** на **Asteroid**.
15. Поместите нод **GetActorTransform** под нодом **SpawnActor Asteroid**.
16. Соедините контакт вывода **Return Value** нода **GetActorTransform** с контактом ввода **Spawn Transform** нода **SpawnActor Asteroid**.
17. Перетащите контакт вывода нода **SpawnActor Health Pickup** и создайте нод **Set Timer by Function Name**.
18. Соедините контакт вывода нода **SpawnActor Asteroid** с контактом ввода **Set Timer by Function Name**.
19. Установите контакт ввода **Function Name** нода **Set Timer by Function Name** на **Spawn**.
20. Поместите нод **Random Float in Range** под нодом **Set Time by Function Name**.
21. Перетащите переменную с плавающей запятой **Spawn Time Min** в контакт ввода **Min** нода **Random Float in Range**.
22. Перетащите переменную с плавающей запятой **Spawn Time Max** в контакт ввода **Max** нода **Random Float in Range**.
23. Соедините контакт вывода **Return Value** нода **Random Float in Range** с контактом ввода **Time** нода **Set Timer by Function Name**.
24. На панели инструментов нажмите кнопку **Compile** и кнопку **Save**.
25. Удалите с уровня все размещенные ранее астероиды и бонусы здоровья.
26. Поместите на сцену несколько актеров **Obstacle_Spawner**.
27. На панели инструментов уровня, нажмите кнопку **Play in Viewport**, чтобы проверить вашу работу. Примерно через пять секунд начнут появляться актеры **Obstacle_Spawner**, которые будут порождать актеров **Asteroid** и **Health_Pickup**.

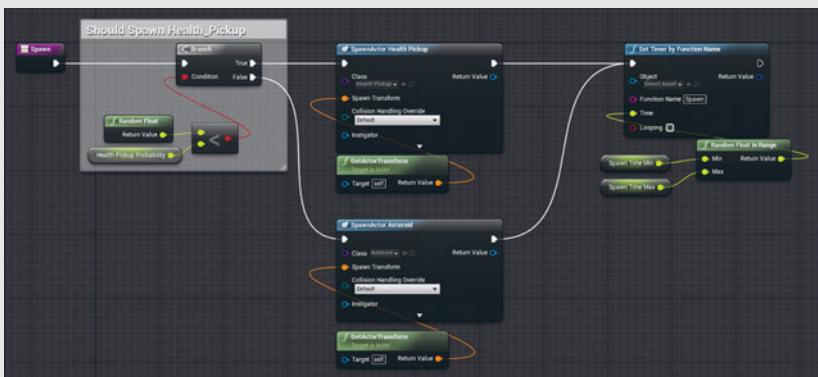


Рис. 21.12. Функция Spawn блюпринт-класса Obstacle_Spawner, которая определяет, какой блюпринт-класс нужно породить, сравнивая случайное значение с плавающей запятой с заданным пороговым значением. Если значение переменной меньше порогового значения, то порождается блюпринт-класс Health_Pickup, в противном случае порождается блюпринт-класс Asteroid. Независимо от того, что именно порождается, функция заново выставляет новое случайное время до следующего порождения

Когда функция Obstacle_Spawner будет готова, больше не придется вручную размещать астероиды и бонусы здоровья. Вместо этого можно разместить за кадром несколько актеров Obstacle_Spawner, чтобы создать множество мест, где может появляться препятствие. На рис. 21.13 и 21.14 показаны примеры возможного размещения точек порождения и результаты.

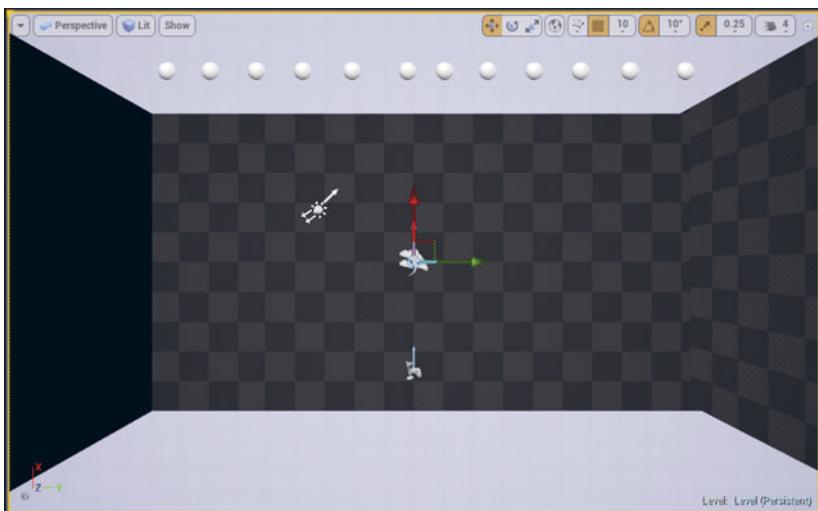


Рис. 21.13. Пример размещения актеров Obstacle_Spawner равномерно по всему игровому пространству



Рис. 21.14. Скриншот игры при размещении актеров Obstacle_Spawner так, как показано на рис. 21.13

Разместите несколько актеров Obstacle_Spawner на ваше усмотрение и запустите игру.

Удаление старых препятствий

Теперь, когда в игре порождается потенциально бесконечное число астероидов и бонусов, может закончиться оперативная память и начаться проблемы с производительностью из-за слишком большого количества астероидов и бонусов на сцене. К счастью, Unreal Engine позволяет легко исправить эту проблему.

Открыв панель **Modes**, поместите новый объект Kill ZVolume в нижней части экрана по всей ширине игровой области. Всякий раз, когда актер попадает в зону Kill ZVolume, актер будет уничтожаться (рис. 21.15).

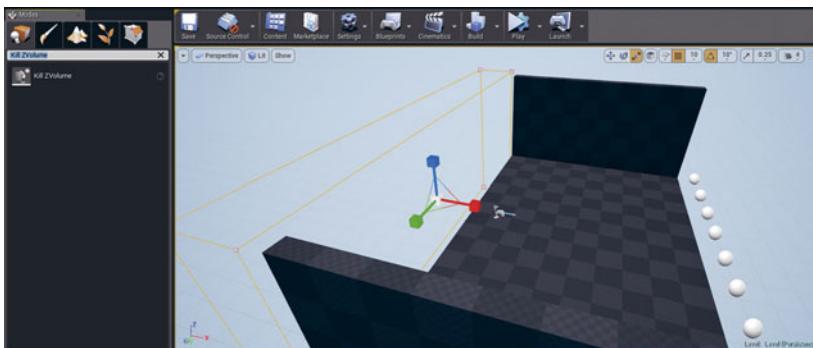


Рис. 21.15. Пример размещения объекта Kill ZVolume, который используется для удаления астероидов, если они вылетят за пределы экрана

После создания Kill ZVolume нажмите кнопку **Simulate**, чтобы убедиться, что ваши астероиды уничтожаются, попадая в зону Kill ZVolume.

Резюме

В этом часе мы узнали, как создать набор блюпринт-классов, которые будут взаимодействовать с аватаром. Мы создали состояние повреждения и механизм для устранения повреждений. Мы узнали, как порождать актеров контролируемым случайнным образом и как уничтожать актеров при взаимодействии с ними. На данный момент в игре есть готовый функционал и основа, которую можно развивать.

Вопросы и ответы

Вопрос: Мои астероиды и бонусы не взаимодействуют с космическим кораблем. Почему?

Ответ: Возможно несколько вариантов. Во-первых, убедитесь, что у всех ваших препятствий и бонусов есть сфера коллизии и что свойству Collision Presets присвоено значение OverlapAll. Затем убедитесь, что у аватара тоже есть коллизия. Наконец, убедитесь, что аватар и препятствие находятся на одной высоте и соприкасаются. При виде сверху это не так очевидно.

Вопрос: Я хочу добавить некоторую логику поведения в скрипт конструирования актера Health_Pickup, но я не хочу, чтобы изменения затрагивали астероиды. Как мне это сделать?

Ответ: Есть два основных способа достижения такого рода разделения между несколькими потомками одного и того же родительского класса. Первый способ — это сделать так же, как вы отключили случайное изменение размера у класса Health_Pickup. Масштабирование было реализовано в скрипте конструирования родительского класса, но мы создали переменные, позволяющие каждому дочернему классу масштабировать себя независимо.

Второй способ: откройте скрипт конструирования класса Health_Pickup, щелкните правой кнопкой мыши на node Construction Script и выберите пункт меню **Add Call to Parent Function**. Это создаст особый нод Parent: Construction Script. Этот нод перенимает все поведение родительского класса (например, класса Obstacle) перед запуском логики, присущей только дочернему классу.

Вопрос: Зачем проверять перекрытие внутри препятствия, а не внутри блюпринта Hero_Spaceship?

Ответ: Можно было бы обработать все перекрытия в блюпринт-классе Hero_Spaceship, но такой подход не рекомендуется. Проблема в том, что, если

вы будете проверять, с чем столкнулся аватар, и это будет трудно обрабатывать. Чтобы обработать два взаимодействия «изнутри» аватара, потребуется создание графа сложнее, с несколькими нодами ветвления для обработки различных типов поведения.

Поскольку игра становится завершеннее и сложнее, работать с таким графом будет трудно. Вместо этого проще будет назначить модели поведения соответствующим объектам. Например, астероид не знает, что значит «нанести повреждение» объекту Hero_Spaceship. Он лишь знает о факте столкновения с Hero_Spaceship и вызывает функцию Take Damage. В то же время Hero_Spaceship не знает, в чем причина получения повреждений, но знает, что нужно делать, когда вызывается функция Take Damage.

Вопрос: Зачем мы использовали дополнительный таймер в функции `Spawn`, вместо того чтобы использовать цикл с проверкой условия в самом таймере событий на `Begin Play`?

Ответ: Вы можете зациклить логический контакт ввода нода `Set Timer by Function Name`, вместо того чтобы вызывать таймер внутри функции. Это приведет к слегка иному поведению, при котором каждый экземпляр точки порождения будет иметь свой интервал порождения нового препятствия, но эти интервалы будут постоянны. Установливая новое значение времени в каждом вызове функции `Spawn`, вы будете уверены, что актеры `Obstacle_Spawner` всегда будут создавать новые препятствия через случайные промежутки времени, что сделает игру менее предсказуемой.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Истинно или ложно высказывание: компонент вращательного движения поворачивает актера, к которому он прикреплен, вокруг оси с заданной скоростью.
2. Истинно или ложно высказывание: класс `DefaultPawn` содержит логику обработки повреждений и информацию, и здоровье аватара.
3. Истинно или ложно высказывание: нод `Cast To` работает только для преобразования актеров в других актеров.
4. Истинно или ложно высказывание: ноду `Set Timer by Function Name` нужно задать точное имя функции, которую вы пытаетесь вызвать.

Ответы

1. Истина. Но этот компонент способен и на большее. Несомненно, вам стоит поэкспериментировать с этим компонентом, чтобы лучше узнать все его возможности.
2. Ложь. Класс DefaultPawn способен на многое, но обработка здоровья и повреждений — это ваша работа.
3. Ложь. Нод Cast To работает со всеми типами. Любой тип (например, текстуры, материалы, эффекты частиц) может быть приведен к другому типу, если у них есть общая иерархия классов.
4. Истина. Нод Set Timer by Function Name не может угадать, какую функцию вы имеете в виду, если она не указана в нем напрямую. Здесь даже пробелы в имени имеют значение.

Упражнение

Вы можете самостоятельно придумать другие типы бонусов и простых актеров, чье поведение основано на коллизии, и ввести их в игру. Затем воспользуйтесь знаниями из часа 20 и этого часа, чтобы создать препятствие, которое может уничтожить астероиды. Этим препятствием игрок выстреливает сам, нажимая на клавиши. Если у вас достаточно энтузиазма, можете также поработать над освещением и окружающей средой, чтобы она лучше соответствовала теме игры.

1. Создайте новый блупринт-класс, который наследует от класса Obstacle, и назовите его **Plasma_Bolt**.
2. Задайте компоненту StaticMesh пользовательский материал. Так как это выстрел из оружия, можно использовать какой-нибудь светящийся красный или синий материал.
3. На панели **Class Defaults** для нового класса Plasma_Bolt измените вектор **Movement Direction** с **-1.0, 0.0, 0.0** на **1.0, 0.0, 0.0**.
4. Присвойте переменным **Random Scale Min** и **Random Scale Max** значение **0.2**.
5. Присвойте переменным **Speed Min** и **Speed Max** значение **1000**.
6. Добавьте новый нод Event ActorBeginOverlap на граф событий препятствия и используйте нод Cast To Asteroid, чтобы обнаружить его пересечение с астероидом.

7. Когда Plasma_Bolt пересекается с астероидом, можно сделать порождение частиц взрыва и использовать функцию DestroyActor, чтобы уничтожить и актера Asteroid, и актера Plasma_Bolt.
8. Создайте новое входное действие для выстрела и привяжите к нему клавишу **Пробел** либо другую клавишу или кнопку геймпада.
9. В графе событий аватара создайте новое событие для вашего действия.
10. С помощью контакта вывода нода Pressed используйте SpawnActor для порождения актера Plasma_Bolt в точке местоположения аватара.
11. Попробуйте вашу игру и с помощью привязанной к выстрелу кнопки из шага 8 постреляйте в надоедливые астероиды!

22-Й ЧАС

Работа с UMG

Что вы узнаете в этом часе

- ▶ Использование редактора пользовательских интерфейсов Unreal Motion Graphics (UMG)
- ▶ Создание виджет-блюпринта
- ▶ Создание Game Mode для стартового меню
- ▶ Создание интерфейса меню

Unreal Motion Graphics UI Designer (UMG) — редактор в UE4, который используется для проектирования, анимации и программирования пользовательских интерфейсов и функциональности HUD. Этот час познакомит вас с UMG и научит разработке стартового меню.

ПРИМЕЧАНИЕ

Подготовка к практике

В этом часе вам потребуется создать новый проект с шаблоном Flying и включить в него Starter Content. Затем, когда проект будет подготовлен, на панели **Content Browser** следует создать папку под названием *StartMenuGame*.

Создание виджет-блюпринта

Существует два наиболее распространенных метода для создания интерфейсов и HUD с помощью блюпринтов. Одним из них является закодировать в Blueprint HUD-класс, привязанный к Game Mode. Второй способ, лучше подходящий для дизайнеров, — использовать конструктор UMG UI. UMG позволяет в интерактивном режиме размещать элементы интерфейса, называемые виджетами, и назначать блюпринту заданную функциональность. Когда вы освоите основы UMG, создание интерфейса станет легким и быстрым.

Перед тем как начать использовать UMG, вам необходимо создать виджет-блюпринт, как описано в следующей рубрике «Попробуйте сами».

ПОПРОБУЙТЕ САМИ**Создание ассета блюпринт-виджета**

Чтобы открыть интерфейс UMG, выполните следующие действия и создайте ассет Widget Blueprint.

1. Щелкните правой кнопкой мыши по папке *StartMenuGame*, созданной для этой главы, и выберите пункт **Widget Blueprint**, чтобы добавить в папку новый виджет-блюпринт.
2. Назовите новый блюпринт **StartMenuWidget_BP**.
3. Дважды щелкните по блюпринту **StartMenuWidget_BP**, чтобы открыть его в UMG.

Навигация по интерфейсу UMG

Интерфейс UMG имеет два режима работы, приведенные на рисунке 22.1: режим конструктора **Designer** для размещения виджетов, например изображений или текста, и графический режим **Graph** для добавления функций в блюпринт. При первом открытии UMG по умолчанию отображает режим конструктора.

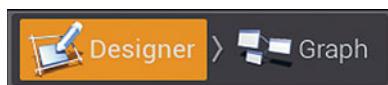


Рис. 22.1. Режимы UMG

Режим конструктора

В режиме конструктора есть панель **Palette**, содержащая список всех виджетов, которые можно использовать, отсортированный по функциональности (см. рис. 22.2). На панели **Hierarchy** (Иерархия) содержатся виджеты, размещенные на вашем интерфейсе. На панели **Hierarchy** можно прикреплять виджеты друг к другу и отделять их друг от друга по мере необходимости. Основной частью блюпринта виджета является панель **Canvas** (холст), и все виджеты, прикрепленные к нему. Также в интерфейсе есть панель **Details** (Информация), на которой отображаются свойства выбранных виджетов, размещенных на интерфейсе. В этом режиме также есть панель **Animations** (Анимации) и **Timeline** (Временная шкала), используемая для создания, управления и редактирования анимацией виджетов. В центре расположена панель **Designer** (Конструктор), которая используется для создания макета интерфейса путем перетаскивания элементов с панели **Palette** (Палитра) и размещения их на панели **Designer**.

На рис. 22.2 номерами обозначены следующие элементы: 1) панель инструментов; 2) палитра **Palette**; 3) панель **Hierarchy**; 4) панель **Designer**; 5) панель **Details**; 6) панель **Animations**.



Рис. 22.2. Интерфейс режима Designer UMG

ПРИМЕЧАНИЕ

Корневой виджет по умолчанию

В новом виджет-блюпринте панель **Canvas** является корневым виджетом по умолчанию, к которому можно прикрепить другие виджеты. Однако вы можете удалить панель **Canvas** и сделать любой другой виджет корневым. Как правило, это делается тогда, когда виджет-блюпринт будет использоваться как часть другого виджет-блюпринта.

Режим Graph

Режим **Graph** — это редактор блюпринтов для виджет-блюпринтов (рис. 22.3). Здесь вы задаете функциональность для размещенных виджетов в виде либо функций, либо последовательности событий. В редакторе блюпринтов режима **Graph** есть панель **My Blueprint**, используемая для управления графиками, функциями, макросами, переменными и диспетчерами событий. На панели **Details** отображаются свойства выбранных нодов. Граф событий используется для создания сценариев. Обычно виджеты, размещенные в режиме **Designer**, отображаются на ней в виде переменных.

ПРИМЕЧАНИЕ

Ссылочные переменные виджетов

Если виджет, размещенный в режиме конструктора, не отображается в виде переменной в режиме **Graph**, но вам нужно, чтобы он отображался, вы можете вернуться в режим конструктора, выбрать виджет, а затем на панели **Details** активировать свойство **IsVariable**.

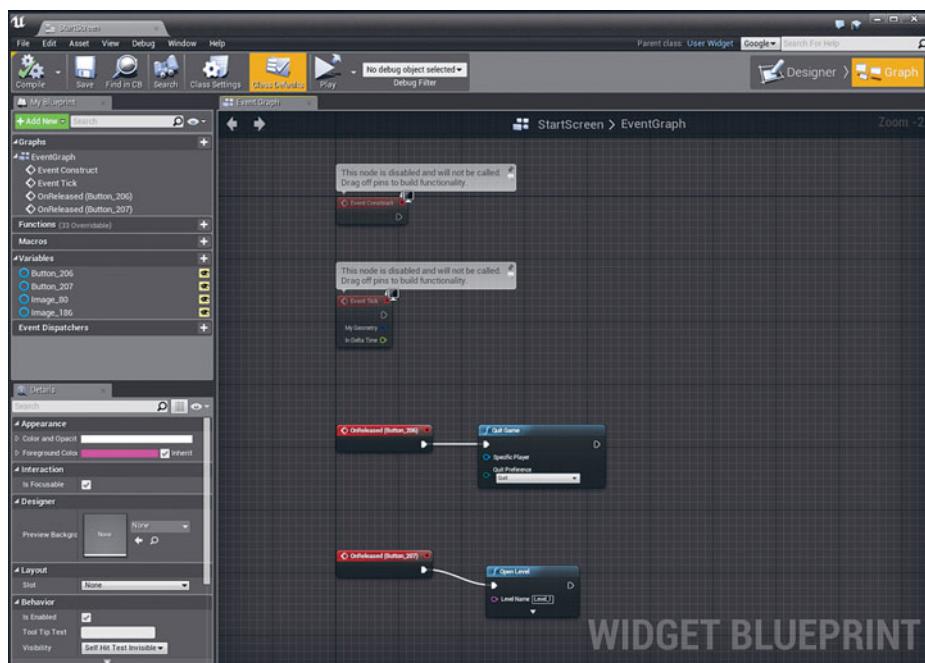


Рис. 22.3. Интерфейс режима Graph UMG

Настройка разрешения

В режиме **Designer** можно установить разрешение, для которого вы создаете интерфейс или HUD. Хотя UE4 масштабирует свою игру под любое разрешение поддерживаемой целевой платформы, интерфейсы следует разрабатывать на основе часто используемых разрешений и соотношений сторон.

Чтобы выбрать разрешение для интерфейса в режиме **Designer**, откройте раскрывающийся список **Screen Size** в верхнем правом углу и выберите разрешение из готового списка часто используемых разрешений (рис. 22.4).



Рис. 22.4. Предустановленные размеры экрана

При разработке игры для конкретной платформы, например ПК или игровой приставки (консоли), вы никогда не можете быть уверены в том, какое разрешение установлено на мониторе конечного пользователя, поэтому ваш интерфейс должен адаптироваться под различные разрешения экрана и соотношения сторон. В таблице 22.1 приведены распространенные разрешения — которые не являются фактическими параметрами для виджет-бллюпринта или проекта, поскольку разрешение все же определяется аппаратными средствами конечного пользователя. Предварительные настройки задают «рабочий масштаб» для работы с вашим интерфейсом.

ПРИМЕЧАНИЕ

Настройка разрешения

Вне зависимости от разрешения и соотношения сторон конкретной платформы, верхний левого угол на любом экране имеет координату (0,0), где X — горизонтальный размер, и Y — вертикальный. HD и UHD* относятся к плотности пикселей, то есть общему их числу, которое получается умножением количества пикселей по горизонтали и вертикали. Например, $1280 * 720 = 921\,600$ пикселей.

ТАБЛ. 22.1. Часто используемые соотношения сторон и разрешения экрана

Соотношение сторон Типовые разрешения

4: 3 (1.33)	320×240, 640×480, 1024×768, 2048×1536 (HD)
16:10 (1.6)	1280×800, 1440×900, 2560×1600 (UHD)
16: 9 (1,77)	1280×720 (HD), 1920×1080 (HD), 3840×2160 (4K UHD)

ПРИМЕЧАНИЕ

Соотношение сторон

Соотношение сторон определяет, работает пользователь на широком экране (16:9) или на стандартном 4:3 (NTSC / PAL**). Соотношение сторон — это отношение числа точек по горизонтали и вертикали. Например, 16: 9 означает, что на каждые 16 пикселей по горизонтали приходится 9 пикселей по вертикали.

Вы можете создать интерфейс для всех разрешений, которые, по вашей задумке, должна поддерживать игра. Но для этого понадобится больше художественных ассетов и виджет-блюпринтов, а это приведет к росту занимаемой проектом памяти и сложности.

Существует хорошее эмпирически выведенное правило — нужно разработать интерфейс для самого высокого разрешения и самого распространенного соотношения сторон, который ваша игра должна поддерживать. Затем подогнать интерфейс под более низкое разрешение и различные соотношения сторон. У компании Epic есть для этого инструменты — якорные точки и масштабирование DPI.

* Высокая четкость (High Definition) и сверхвысокая четкость (Ultra-High Definition). — Прим. ред.

** Системы аналогового цветного телевидения. — Прим. ред.

Якорные точки и масштабирование DPI

Мы используем якорные точки при работе с виджет-бллюпринтами с корневым компонентом Canvas. Каждый размещаемый виджет имеет якорную точку, которая представляет собой нормированную опорную точку для размещения виджета на экране. Положение точки на экране рассчитывается в процентном соотношении, а не по числу пикселей. Это означает, что положение якорных точек не зависит от разрешения экрана и соотношения сторон. Например, якорная точки со значениями X, Y равными (0.5, 0.5) всегда будет находиться в центре экрана, независимо от разрешения и соотношения сторон, а точка в положении 1,1 всегда будет находиться в нижнем правом углу, независимо от разрешения экрана и соотношения сторон. Однако расположение виджета относительно его якорной точки рассчитывается в пикселях, отчего виджет всегда остается на заданном расстоянии от якорной точки, независимо от того, какое у интерфейса разрешение и соотношение сторон.

И тут вступает в дело масштабирование DPI. Масштабирование означает изменение разрешения интерфейса в большую или меньшую сторону в зависимости от разрешения целевой платформы и соотношения сторон таким образом, чтобы все якорные точки установились правильно и, свою очередь, отрегулировали положения виджетов на экране. Для настройки параметров масштабирования DPI выберите пункт меню **Edit** ⇒ **Project Settings** ⇒ **Engine** ⇒ **User Interface** (Редактирование⇒ Настройки проекта ⇒ Движок ⇒ Пользовательский интерфейс).

Вы можете вручную выполнять тонкую настройку кривой DPI, как показано на рисунке 22.5, но настройки по умолчанию и без того хороши. Единственное, что вам может понадобиться изменить, это свойство **DPI Scale Rule** (Правило масштабирования DPI), которое устанавливает, относительно какой оси будет масштабироваться интерфейс. Горизонтальная ось всегда называется X, а вертикальная — Y. Самая короткая и самая длинная стороны изменяются в зависимости от того, работает ли игра в портретном или панорамном режиме.

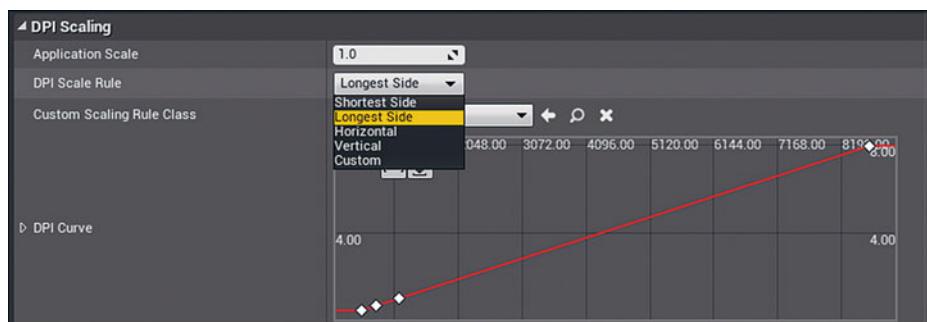


Рис. 22.5. Настройки масштабирования DPI

Создание стартового меню

В оставшейся части этого часа мы создадим стартовое меню игры. Чтобы сделать это, потребуется создать новый Game Mode и блюпринт Player Controller, а также пустой уровень. Player Controller будет настроен на отображение указателя мыши, а Game Mode привязан к уровню так, чтобы каждый раз, когда загружается уровень, автоматически отображалось стартовое меню. Наконец, мы установим этот уровень как уровень игры по умолчанию, так что это будет первый уровень, который станет загружаться при запуске игры.

Импорт ассетов

Чтобы создать интерфейс, вам понадобятся соответствующие изображения и аудиофайлы.

В папке *Hour_22* сопроводительных файлов вы можете найти папку под названием *InterfaceAssets*, в которой находятся все ассеты, необходимые для создания основного интерфейса стартового меню. Изображения будут импортированы в виде текстур, а звуковые файлы как звуковые волны.

При импорте изображений для создания интерфейсов нужно учесть три основные вещи: генерация мипмапа, автоматическое масштабирование и подгрузка (стриминг) текстур и распределение текстур в группы текстур.

При создании и импорте изображений, которые будут использоваться в интерфейсе, правило «степени двойки» не применяется. Следование этому правилу при использовании изображений важно для текстур, используемых в материалах, которые нужно размещать на мешах. У текстур, используемых в интерфейсах, таких ограничений нет.

См. 6-й час «Использование материалов», чтобы узнать больше о работе с текстурами, материалами и о степенях двойки.

При импорте изображений в UE4 нужно отнести их к какой-либо группе текстур, чтобы редактор знал, как их обрабатывать. Чтобы открыть ассет текстуры в Texture Editor, найдите его на панели **Content Browser** и дважды щелкните по ассету текстуры. Для текстур, которые будут использоваться для интерфейса, нужно присвоить свойству **Texture Group** значение **UI** в Texture Editor (см. рис. 22.6).



Рис. 22.6. Texture Editor, в котором показаны настройки Mip Gen текстуры и группа текстуры

Mip в термине «мип мап» означает *multum in parvo* («многое в малом»), а мипмаппинг — процесс генерации последовательности изображений с пониженным разрешением из более крупного изображения. В UE4 мипмапы генерируются автоматически, когда импортируется изображение, соответствующее рекомендации о «степени двойки». Эта технология применяется для повышения производительности в 3D-графике. По мере того как объект удаляется от камеры, его разрешение уменьшается, и, следовательно, можно использовать текстуры с меньшим разрешением. Но изображениям, используемым в интерфейсе, как правило, мипы не нужны, так как интерфейс обычно отображается на переднем плане, поверх всего остального.

Подгрузка (стриминг) текстур — процесс загрузки текстур в память во время выполнения игры. Этот процесс можно заметить, когда уровень игры загружается «на ходу». Изображения низкого разрешения, созданные с помощью мипмаппинга, быстрее загружаются (или уже загружены) в память и отображаются на поверхностях моделей первыми. Затем текстуры с низким разрешением постепенно заменяются на текстуры с более высоким разрешением по мере их подгрузки. Переход от текстур с низким разрешением к текстурам высокого разрешения визуально выглядит как внезапное обновление (popping). Это одна из причин, почему до сих пор игры заставляют нас любоваться экранами загрузки уровней. И уж тем более не хотелось бы демонстрировать обновление текстур на элементах интерфейса. Любой текстуре можно задать свойство **Never Stream**.

ВНИМАНИЕ

Стриминг текстур

Хороший прием — включать свойство **Never Stream** только для текстур, которые используются в интерфейсах. Свойство **Never Stream** текстуры можно найти на панели **Texture Editor Details** в категории **Textures**. Нужно развернуть категорию, чтобы найти эту настройку.

Когда импортируется текстура, размерность которой не является степенью двойки, свойство **Never Stream** активируется, а свойству Mip Gen присваивается значение **NoMipmaps**. Также нужно присвоить свойству **Texture Group** в Texture Editor значение UI.

В прилагаемой папке *InterfaceAsset* есть текстура для фона, текстура заголовка, текстура кнопки и два аудиофайла — *Mouse Hover* и *Mouse Pressed*. Импортируйте все эти ассеты. Когда изображения будут добавлены в **Content Browser**, откройте каждую текстуру в редакторе Texture Editor, присвойте для них значение UI свойству **Texture Group** и включите свойство **Never Stream**.

СОВЕТ

Разрешение текстур

Если вам нужно узнать разрешение изображения после импорта, на панели **Content Browser** наведите курсор на импортируемый ассет, чтобы увидеть соответствующую информацию о нем, либо откройте его в редакторе Texture Editor.

В созданной ранее папке *StartMenuGame* на панели **Content Browser** проекта создайте новую папку и перенесите в нее все импортированные ассеты проекта.

Размещение виджетов на холсте

Когда ассеты импортированы, пора приступить к настройке интерфейса в виджет-блюпринте. Прежде всего, необходимо разместить фоновое изображение и название игры, используя виджеты **Image** в UMG.

ПОПРОБУЙТЕ САМИ

Размещение виджета **Image**

Выполните предложенные действия, чтобы добавить виджет **Image** и назначить ему текстуру.

1. Дважды щелкните по блюпринту **StartMenuItemWidget_BP**, чтобы открыть его в UMG.

2. В режиме **Designer** установите размер экрана равным **1080p (HDTV, Blu-Ray)**.
3. Перетащите виджет **Image** с панели **Palette** на [Canvas Panel] панели **Hierarchy** в окне режима **Designer**.
4. Выбрав виджет **Image**, в разделе **Slot** панели **Details** установите якорную точку в центр экрана с помощью шаблона или вручную, присвоив параметрам **Minimum X**, **Minimum Y** значения **0,5, 0,5** и те же значения параметрам **Maximum X**, **Maximum Y** (рис. 22.7).
5. Перетащите ассет **Background Image Texture** из панели **Content Browser** на свойство **Image** раздела **Appearance** на панели **Details**.
6. В разделе **Slot** на панели **Details** присвойте свойству **Size X** значение **1920**, а свойству **Size Y** значение **1080**.
7. В области просмотра панели **Designer** расположите изображение так, чтобы оно заполнило всю панель **Canvas**.
8. Перетащите еще один виджет **Image** и повторите шаги 3–6, разместив ассет **Title Image Texture**. Якорная точка для этого виджета также должна быть в центре, но свойства **Size X** и **Size Y** должны соответствовать размеру изображения — **641×548**.
9. Сохраните виджет-блюпринт. После этого ваше стартовое меню должен выглядеть так, как показано на рисунке 22.7.

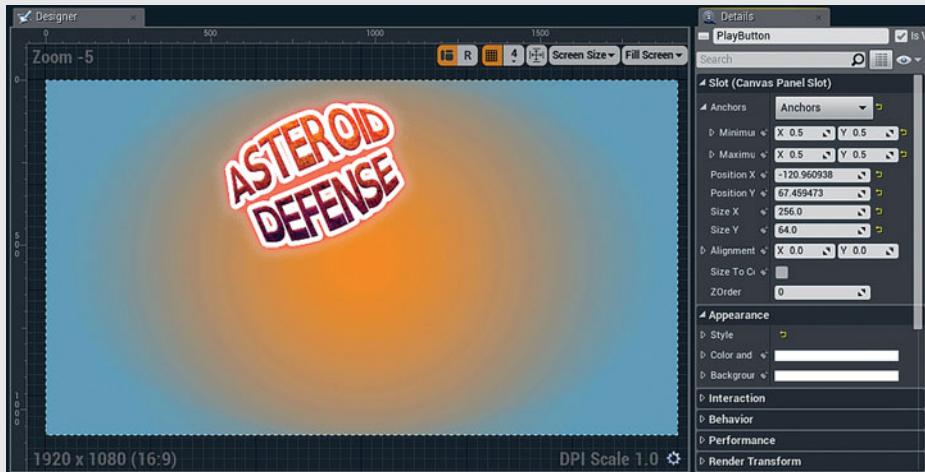


Рис. 22.7. Панель **Designer** UMG, на которой выводится фоновое изображение и название игры

Затем используем виджеты для кнопки и текста, чтобы создать кнопки **Play** и **Quit**. Виджет кнопки уже имеет встроенные функции для взаимодействия с мышью — события **MouseOver** и **MouseDown**. Все, что вам нужно сделать, это разместить

виджет на панели **Canvas** и снарядить его нужными ассетами. Этот процесс описан в рубрике «Попробуйте сами» далее, а в табл. 22.2 показаны состояния кнопки для взаимодействия мыши с виджетом-кнопкой.

ТАБЛ. 22.2. Свойства виджета-кнопки

Состояние кнопки	Описание
Normal	Изображение, которое будет отображаться, когда мышь никак не взаимодействует с кнопкой
Hovered	Изображение, которое будет отображаться при наведении курсора мыши на кнопку
Pressed	Изображение, которое будет отображаться при нажатии на кнопку
Disabled	Изображение, которое будет отображаться, когда кнопка отключена в блюпринте

ПОПРОБУЙТЕ САМИ

Размещение виджетов **Text** и **Button**

Теперь нам нужно создать кнопки Play и Quit для нашего меню. Виджет-кнопка содержит изображение кнопки, а в текстовый виджет — текст для этой кнопки. Текстовый виджет должен быть присоединен к кнопке, чтобы при перемещении кнопки текст перемещался вместе с ней. Выполните следующие действия.

1. Откройте UMG (если он еще не открыт), дважды щелкнув по блюпринту **StartMenuItem_BP**.
2. Перетащите виджет кнопки из панели **Palette** и поместите его в разделе **Canvas Panel** на панели **Hierarchy**.
3. Выбрав виджет, переименуйте его в верхней части панели **Details** на **PlayButton**.
4. Выбрав размешенный виджет, на панели **Details** в разделе **Slot** установите якорную точку в центр экрана с помощью шаблона или вручную, присвоив параметрам **Minimum X**, **Minimum Y** значения **0,5**, **0,5** и те же значения параметрам **Maximum X**, **Maximum Y**.
5. Установите размер виджета таким, чтобы он соответствовал текстуре кнопки. Для этого в разделе **Slot** на панели **Details** присвойте свойству **Size X** значение **256**, а свойству **Size Y** значение **64**.
6. На панели **Details** перейдите в раздел **Appearance** и выберите текстуру **NormalButton** для свойства изображения **Normal**.
7. На панели **Details** перейдите в раздел **Appearance** и выберите текстуру **HoverButton** для свойства изображения **Hovered**.

8. На панели **Details** перейдите в раздел **Appearance** и выберите текстуру **PressedButton** для свойства изображения **Pressed**.
9. Чтобы настроить звук для кнопки, на панели **Details** перейдите в раздел **Appearance** и выберите звуковую волну **MPressed_sw** для свойства **Pressed Sound**.
10. На панели **Details** перейдите в раздел **Appearance** и выберите звуковую волну **MHover_sw** для свойства **Hovered Sound**.
11. Добавьте кнопке текст. Перетащите текстовый виджет с панели **Palette** на виджет **PlayButton** в области просмотра панели **Designer**.
12. Выбрав текстовый виджет, на панели **Details** перейдите в раздел **Content** и задайте значение **PLAY** для свойства **Default Text**.
13. Чтобы создать кнопку **Quit**, повторите шаги 2–11, установив при этом имя виджета **QuitButton** и присвоив значение **QUIT** свойству **Default Text**.
14. Когда все будет готово, ваше стартовое меню должно должен выглядеть так, как показано на рис. 22.8.



Рис. 22.8. Панель **Designer** UMG с отображенном разделом **Appearance** панели **Details**

Создаем функциональность через скрипты

Теперь нужно запрограммировать основные функциональные возможности блю-принта для каждой из созданных вами кнопок. Для кнопки характерны три типа событий: **OnClicked**, **OnPressed** и **OnReleased**.

ПОПРОБУЙТЕ САМИ

Скриптовые события для кнопок

Выбрав виджет PlayButton в режиме **Designer**, в разделе **Events** на панели **Details** вы увидите три типа событий, которые можно добавить к кнопке. Выполните следующие действия, чтобы запрограммировать простые события OnReleased событий для кнопок Play и Quit.

1. В режиме **Designer** в области просмотра панели **Designer** выберите виджет **PlayButton**.
2. На панели **Details** перейдите в раздел **Events** и нажмите символ «+» рядом со свойством **OnReleased**, как показано на рис. 22.9. UMG переключится в режим **Graph** и создаст событие **OnReleased** в графе событий.

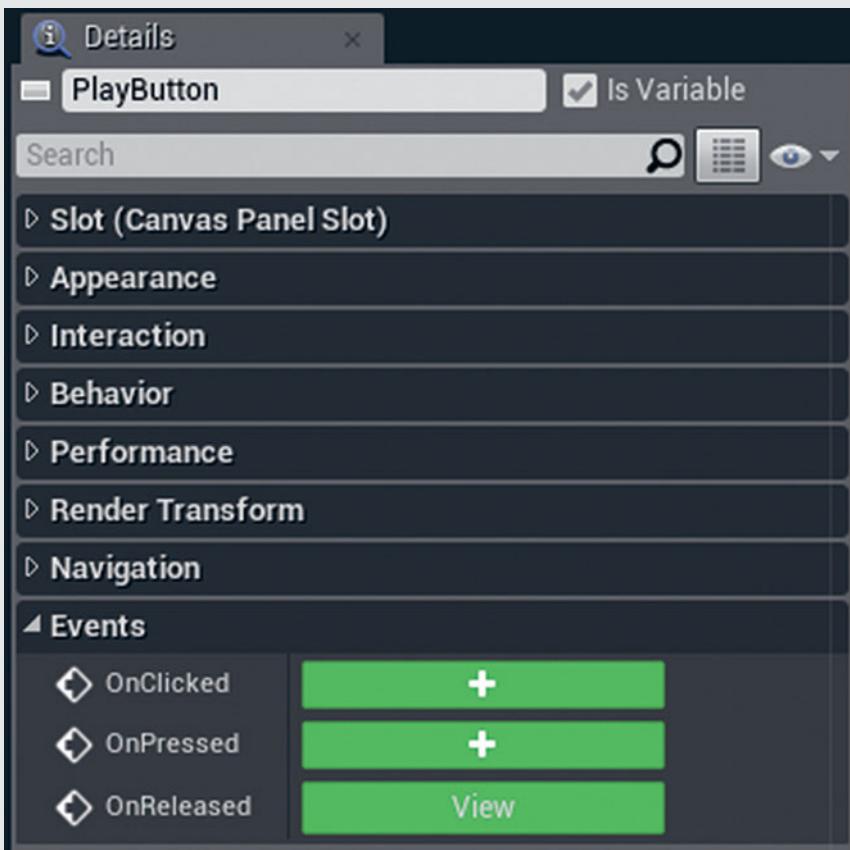


Рис. 22.9. Свойство **Events** кнопки на панели **Details**

3. Перетащите exec-контакт нода **OnReleased**, в контекстном меню введите слово **open** и выберите нод **Open Level**.

4. Разместив нод Open Level, присвойте свойству **Level Name** значение **FlyingExampleMap**.
5. В режиме **Designer** в области просмотра панели **Designer** выберите виджет **QuitButton**, и на панели **Details** перейдите к разделу **Events** и нажмите символ «+» рядом со свойством **OnReleased**, чтобы создать событие OnReleased в графе событий.
6. Перетащите ехес-контакт нода OnReleased, в контекстном меню введите слово **quit** и выберите нод **Quit**. Когда вы закончите, блюпринт должен выглядеть так, как показано на рис. 22.10.
7. Скомпилируйте и сохраните виджет-блюпринт.

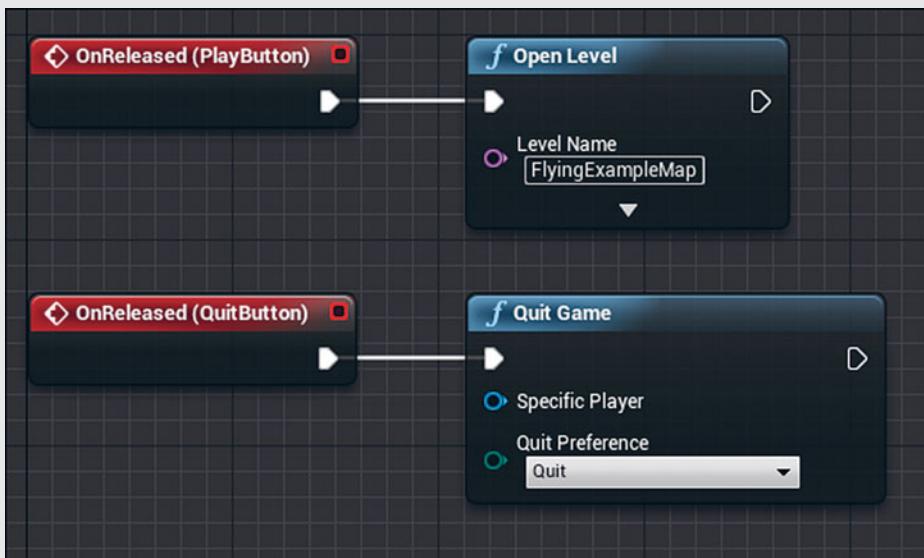


Рис. 22.10. UMG: граф событий с событиями OnReleased

Возможность изменять внешний вид курсора мыши реализуется с помощью блюпринта Player Controller. Теперь, когда виджет Start Menu готов, необходимо добавить виджет-блюпринт на панель **Viewport** во время запуска игры. Для этого необходимо создать новый Game Mode и класс контроллера, который будет назначен уровню, который будет загружаться первым при запуске игры.

Когда виджет Start Menu настроен и готов к работе, нужно создать простой Game Mode, в котором будет использоваться блюпринт Player Controller для отображения курсора мыши.

ПОПРОБУЙТЕ САМИ

Создание простого Game Mode и отображение курсора мыши

Выполните предложенные действия, чтобы создать Game Mode и блюпринт Controller, который будет отображать курсор мыши.

- Чтобы создать новый блюпринт Game Mode, на панели **Content Browser** перейдите в папку *StartMenuGame*.
- Щелкните правой кнопкой мыши в области **Asset Management** на панели **Content Browser** и выберите в диалоговом меню пункт **Blueprint Class**.
- В появившемся окне **Pick Parent Class** выберите класс **Game Mode** в категории **All Classes**.
- Присвойте новому Game Mode название **StartMenuGameMode**.
- Чтобы создать новый контроллер игрока, на панели **Content Browser** перейдите в папку *StartMenuGame*.
- Щелкните правой кнопкой мыши на панели **Content Browser** и выберите пункт меню **Blueprint Class**.
- В появившемся окне **Pick Parent Class** выберите класс **Player Controller** в категории **All Classes**.
- Дайте новому Player Controller название **StartMenuGameMode**.
- Прикрепите Player Controller к Game Mode, откройте блюпринт StartMenuGameMode и выберите пункт **Class Defaults** на панели инструментов Blueprint Editor. На панели **Details** под разделом **Classes** установите блюпринт StartMenuController в качестве значения свойства **Player Controller Class** (рис. 22.11).

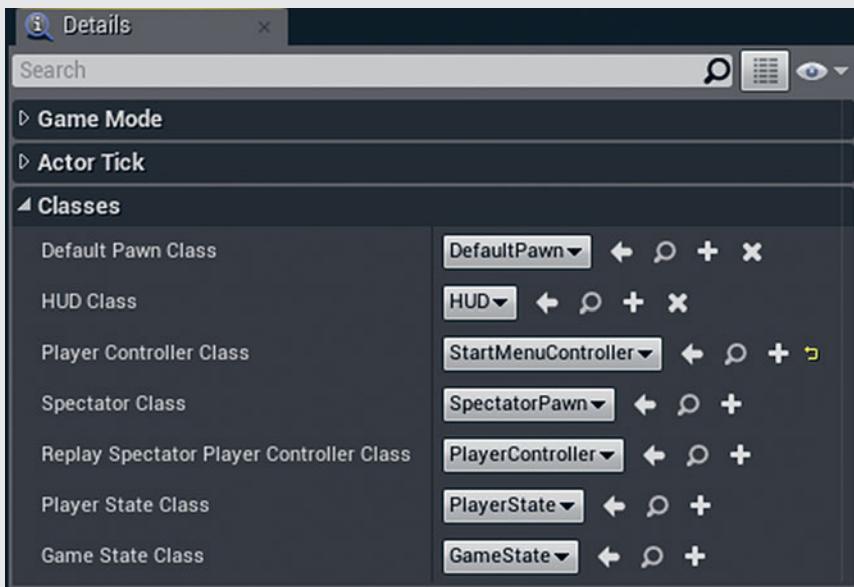


Рис. 22.11. Свойства Class Defaults блюпринта StartMenuGameMode на панели Details

10. Скомпилируйте, сохраните и закройте блюпринт StartMenuGameMode.
11. Теперь вам необходимо указать блюпринту Player Controller отображать курсор мыши. Откройте блюпринт StartMenuController. Выбрав раздел **Class Defaults** на панели инструментов Blueprint Editor, на панели **Details** в разделе **Mouse Interface** включите свойства **Show Mouse Cursor**, **Enable Click Events** и **Enable Mouse Over Events** (рис. 22.12).

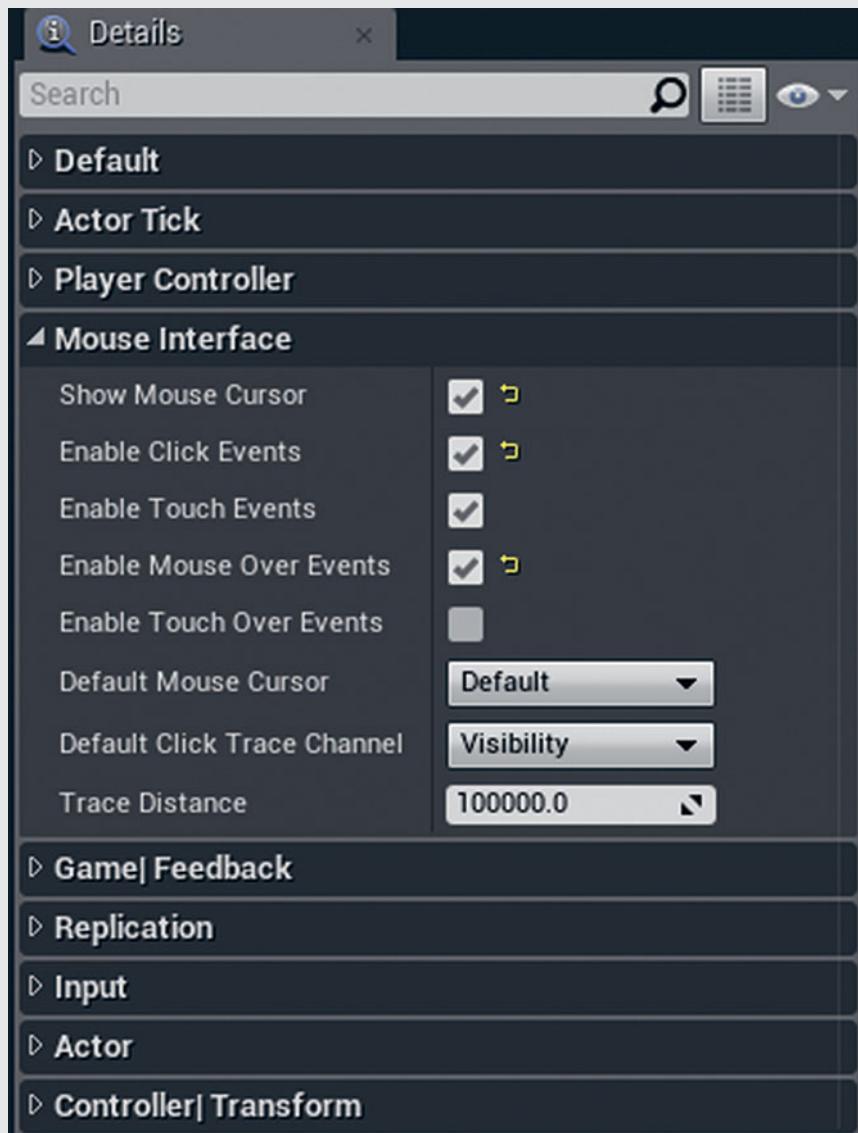


Рис. 22.12. Свойства Class Defaults блюпринта StartMenuController на панели Details

12. Скомпилируйте и сохраните блюпринт StartMenuController.

ПРИМЕЧАНИЕ

Настройка разрешения

Включение и выключение отображения указателя мыши ложится на Player Controller, но задать это свойство можно из любого блюпринта, а не только внутри Player Controller. Для этого перетащите контакт вывода нода GetPlayerController в любом блюпринт-классе, а затем используй нод SetShowMouseCursor для переключения видимости курсора.

Когда Game Mode будет настроен, необходимо создать последовательность блюпринтов, которая добавляет виджет-блюпринт на панель **Viewport** игрока во время игры.

ПОПРОБУЙТЕ САМИ

Добавление виджет-блюпринта на панель Viewport игрока

Выполните предложенные действия, чтобы добавить виджет-блюпринт на панель **Viewport**.

1. Откройте блюпринт StartMenuController в редакторе Blueprint Editor и перейдите на граф событий.
2. В графе событий найдите нод Event BeginPlay.
3. Левой кнопкой мыши перетащите контакт вывода нода Event BeginPlay; в окне поиска контекстного меню введите текст **create widget** и выберите из списка нод **Create Widget**.
4. На вновь созданном ноде функции Create Widget рядом со свойством Class в раскрывающемся списке выберите блюпринт **StartMenuWidget_BP**, который вы создали в начале этого часа.
5. Левой кнопкой мыши перетащите контакт вывода нода Create Widget, в окне поиска контекстного меню введите текст **add to viewport** и выберите из списка нод **Add to Viewport**.
6. Соедините контакт вывода Return Value нода Create Widget с входом Target нода Add to Viewport. Когда все будет готово, блюпринт должен выглядеть так, как показано на рис. 22.13.
7. Скомпилируйте и сохраните блюпринт StartMenuController.



Рис. 22.13. Последовательность функций в блюпринте Player Controller, которая добавляет виджет-блюпринт на панель **Viewport** игрока

Последнее, что нам осталось сделать, это создать новый уровень, который будет загружаться при запуске игры. Вам необходимо сделать этот уровень картой по умолчанию на панели **Project Settings** в разделе **Maps & Modes**. После того как вы сделаете это, во время запуска игры будет загружаться этот уровень и Game Mode, и отобразится стартовое меню.

ПОПРОБУЙТЕ САМИ

Добавление карты по умолчанию

Выполните предложенные действия, чтобы прикрепить Game Mode к уровню, а затем установить уровень в качестве карты по умолчанию.

1. На панели **Content Browser** создайте новую папку и назовите ее **Maps**.
2. Создайте новый пустой уровень и сохраните его в папку **Maps**. Назовите карту **StartLevel**.
3. Откройте панель **World Settings**, нажав кнопку **Setting** на панели инструментов в редакторе Level Editor и выбрав пункт **World Settings**.
4. На панели **World Settings** в разделе **Game Mode** назначьте блюпринт **StartMenuGameMode** свойству **GameMode Override**.
5. Сохраните уровень еще раз.
6. Откройте панель **Project Settings**, нажав кнопку **Settings** на панели инструментов редактора Level Editor и выбрав пункт **Project Settings**.
7. На панели **Project Settings** под разделом **Maps & Modes/Default Maps** присвойте значение **StartLevel** свойству **Game DefaultMap**.
8. Запустите предварительный просмотр уровня. Появится ваше стартовое меню, а курсор должен остаться. Проводите мышью над кнопками, чтобы проверить, как они изменяются и воспроизводится ли звук.
9. Нажмите и отпустите кнопку **Play**. Загрузится уровень **FlyingExampleMap**.

Пример меню

В папке *Hour_22* есть образец проекта с готовым меню, чтобы вы могли изучить и разобраться в нем. Система создана с использованием шаблона Flying. В этом проекте есть стартовое меню, созданное по описанной ранее методике, наряду с двумя другими виджет-блюпринтами, используемыми для создания меню паузы и простого HUD. Виджет-блюпринт меню паузы добавлен на панель Viewport блюпринта FlyingController в шаблоне Flying Game Mode, а виджет-блюпринт HUD добавлен на панель Viewport блюпринта FlyingPawn в шаблоне Flying Game Mode.

Если вы откроете блюпринт FlyingController, то увидите, что виджет-блюпринт меню Pause добавляется на панель **Viewport**, когда игрок нажимает клавишу **Q** или **Esc**. Это связано с тем, что клавиша **Esc** в режиме предварительного просмотра управляется редактором. Когда игра будет упакована и готова к экспорту в исполняемый файл, вы можете отключить событие для клавиши **Q**, а клавиша **Esc** будет по-прежнему работать.

Виджет-блюпринт PawnHud получает данные переменных из блюпринта FlyingPawn путем преобразования в Flying Pawn. Это реализовано в двух функциях, одна из которых отвечает за скорость, а другая — за здоровье аватара. Текстовые виджеты были связаны с функциями, которые выполняют преобразование в Flying Pawn и получают переменные, содержащие значения скорости и здоровья корабля.

Резюме

В этом часе мы узнали, как создать виджет блюпринт, использовать UMG, добавлять виджеты-изображения, кнопки и текстовые виджеты. Мы узнали, как подготовить текстуры для использования в интерфейсе и как настроить Game Mode на взаимодействие с мышью. Но вам есть к чему стремиться. Например, вы можете встроить блюпринт-виджет в другой блюпринт-виджет, чтобы сделать меню более сложным. Но сейчас вы уже знаете, как создать базовое меню, которое есть в любой игре.

Вопросы и ответы

Вопрос: Почему текстуры моего интерфейса отображаются с низким разрешением во время загрузки уровня?

Ответ: У вас не выбран режим UI параметра Texture Group в Texture Editor, поэтому ваша текстура продолжает генерировать мипы.

Вопрос: Зачем UE4 предупреждает меня, когда я импортирую текстуру с размерностью, не равной степени двойки?

Ответ: В зависимости от версии редактора, которую вы используете, вы можете получить предупреждение. Большинство текстур, обычно используемых в материалах, накладываются на Static Mesh и должны иметь размерность степени двойки. Текстуры интерфейса не имеют такого ограничения.

Вопрос: Что такое преобразование блюпринта?

Ответ: В традиционном программировании или средах сценариев вы можете преобразовать один тип переменной в другой. Преобразование

блюпринта также позволяет одному актеру ссылаться на другого актера в игре и вызывать функции или получать и задавать значения переменных в функции Cast Actor.

Вопрос: Когда я импортирую в UE4 собственные текстуры, я хочу иметь возможность видеть сквозь изображение. Что мне нужно для этого сделать?

Ответ: Если вы хотите, чтобы текстура была прозрачной или замаскированной, вам нужно сделать соответствующие преобразования в редакторе изображений. Затем сохраните изображение как 32-битное изображения, чтобы данные о прозрачности и маске импортировались вместе с ним. В файлах типа .tga и .psd эта информация может храниться и импортируется в UE4.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Истинно или ложно высказывание: текстуры интерфейса должны иметь размерность, равную степени двойки.
2. _____ — это процесс создания нескольких вариантов изображения с разным разрешением в процессе импорта.
3. Истинно или ложно высказывание: при импорте текстуры, которая будет использоваться в интерфейсе, нужно определить ее в группу World Texture.
4. Что значит сокращение mip?
5. Какие четыре свойства существует у виджета кнопки?

Ответы

1. Ложь. Текстуры, используемые для интерфейсов, иметь размерность, равную степени двойки.
2. Мипмаппинг — это процесс создания нескольких вариантов изображения с разным разрешением в процессе импорта.
3. Ложь. Нужно определить текстуру в группу UI.
4. «Мип» означает «многое в малом».
5. У виджета кнопки есть четыре свойства: Normal, Hovered, Pressed и Disabled.

Упражнение

В этом упражнении предлагается использовать предоставленные изображения и аудио для создания стартового меню аркадного шутера, который мы создали в часах 20 и 21. Если вы еще не проработали эти часы, можно использовать готовый проект шутера, предоставленный в сопроводительных файлах.

1. Откройте проект аркадного шутера, созданный в часах 20 и 21, либо используйте готовый проект шутера, предоставленный в сопроводительных файлах под названием *ArcadeShooter22*.
2. Создайте Game Mode стартового меню и виджет-блюпринт и создайте интерфейс стартового меню, используя предоставленные ассеты. В качестве основы вы можете использовать результаты упражнений «Попробуйте сами» этого часа.

23-Й ЧАС

Создание исполняемого файла

Что вы узнаете в этом часе

- ▶ Разница между «готовым» и «сырым» контентом
- ▶ Упаковка проекта для Windows
- ▶ Ресурсы для упаковки под Android и iOS
- ▶ Доступ к дополнительным настройкам упаковки

Вы изрядно потрудились и создали игру. Теперь нужно сделать ваше творение доступным для пользователей. Во многих игровых движках этот процесс долгий, сложный и таит в себе множество подводных камней и ловушек. К счастью, в Unreal Engine 4 можно создавать упакованные проекты быстро и легко. В этом часе вы узнаете об упакованном контенте, и как использовать UE4 для создания исполняемого файла с конфигурацией **Shipping**.

ПРИМЕЧАНИЕ

Подготовка к практике

Процедура, описанная в этом часе, может быть выполнена с любым проектом UE4, совместимым с ОС Windows или macOS.

В этом часе мы продолжим работу над проектом ArcadeShooter, который мы начали в часе 20 и продолжили в часах 21 и 22. Если хотите, для простоты можете также использовать готовый проект H22_ArcadeShooter, находящийся в сопроводительных файлах для этого часа в папке *Hour_22* (файлы проектов можно скачать по адресу: http://addons.eksmo.ru/it/UE_24.zip).

Завершив этот урок, сравните полученные результаты с проектом H23_ArcadeShooter, который находится в папке *Hour_23* в сопроводительных файлах.

«Приготовление» контента

Unreal Engine4 хранит контент в форматах, которые могут использоваться только внутри UAsset-файлов. Эти форматы совместимы с редактором UE4, но работают не на всех платформах и не всегда открываются без установленного редактора. Чтобы разработчику не приходилось создавать разные варианты ассетов для различных платформ, в UE4 используется процедура «приготовления», позволяющая убедиться, что контент можно использовать на целевом устройстве.

Этап «приготовления» в процессе разработки, по сути, представляет собой преобразование, в ходе которого выполняется обработка необходимых данных, работающих только в редакторе ресурсов, в ассеты, которые могут использоваться на множестве различных платформ. На этом этапе выполняются и другие задачи, такие как избавление от ненужной или избыточной информации.

«Приготовление» гарантирует, что игровой проект полностью готов, и что в нем нет утерянной информации, которая может вызвать проблемы во время игры в дальнейшем. Процесс «готовки» включает в себя ряд действий, помогающих устраниить баги, например компиляция и проверка блюпринтов на наличие ошибок, проверка правильности составления шейдеров и проверка недостающих ассетов на упакованных уровнях.

Объем времени, необходимый для процесса приготовления, зависит от того, как много контента отправлено на приготовление. По умолчанию UE4 готовит весь контент, необходимый для игры, начиная с уровня по умолчанию. К счастью, UE4 также знает, какая именно информация изменилась после последней упаковки, и обычно этот процесс происходит быстрее, если выполняется уже не в первый раз.

СОВЕТ

«Приготовление» для операционной системы Windows

Даже если вы ориентируетесь на ту же платформу, где выполняется разработка, контент все равно нужно упаковать. В настоящий момент Unreal Engine не поддерживает автономные проекты, работающие с сырьем («неприготовленным») контентом. Поскольку упаковка может занять длительное время, иногда лучше сделать это заранее. Вы можете начать подготовку, выбрав пункт меню **File ⇒ Cook Content for Windows**.

Упаковка проекта для Windows

Подготовка контента — это только первый шаг в процессе передачи проекта пользователям. Следующее, что нужно сделать — взять весь приготовленный контент и упаковать его в пакет с исполняемым файлом, который пользователи смогут запускать. Процедура упаковки зависит от выбранной для проекта платформы. Если вы используете операционную систему Windows и игра разрабатывается для ОС Microsoft, Unreal Engine 4 позволяет легко создавать готовые к распространению пакеты контента.

Для работы редактора UE4 требуется 64-разрядная система, однако UE4 поддерживает упаковку и для 32-разрядных, и 64-разрядных процессоров. Во многих проектах разница будет незаметна. Кроме того, 64-разрядные процессоры могут выполнять 32-разрядные проекты. С современным оборудованием гораздо чаще используются 64-разрядные процессоры, а необходимость в поддержке 32-разрядных компьютеров постепенно сходит на нет. Разработка под 64-разрядные системы позволит вам использовать гораздо больше возможностей современного оборудования.

В общем-то, разрабатывать игру для 32-разрядной системы имеет смысл, только если вы планируете играть на старом оборудовании, или по какой-нибудь другой веской причине.

Упаковка проекта выполняется очень просто с помощью двух команд в меню **File: Shipping** и **Windows (64-bit)** (рис. 23.1).

ПРИМЕЧАНИЕ

Настройки упаковки

На первом изображении на рис. 23.1 показана конфигурация сборки проекта с выбором команды меню **Shipping**. При использовании этой опции будут отключены многие команды отладки. Некоторые из функций отладки, которые активируются с помощью команды **Development**, могут быть использованы неправильно и создавать фатальные ошибки во время игры. Таким образом, любые распределенные пакеты лучше создавать с помощью команды **Shipping**.

Но для тестирования во время разработки опция **Development** вполне подходит.

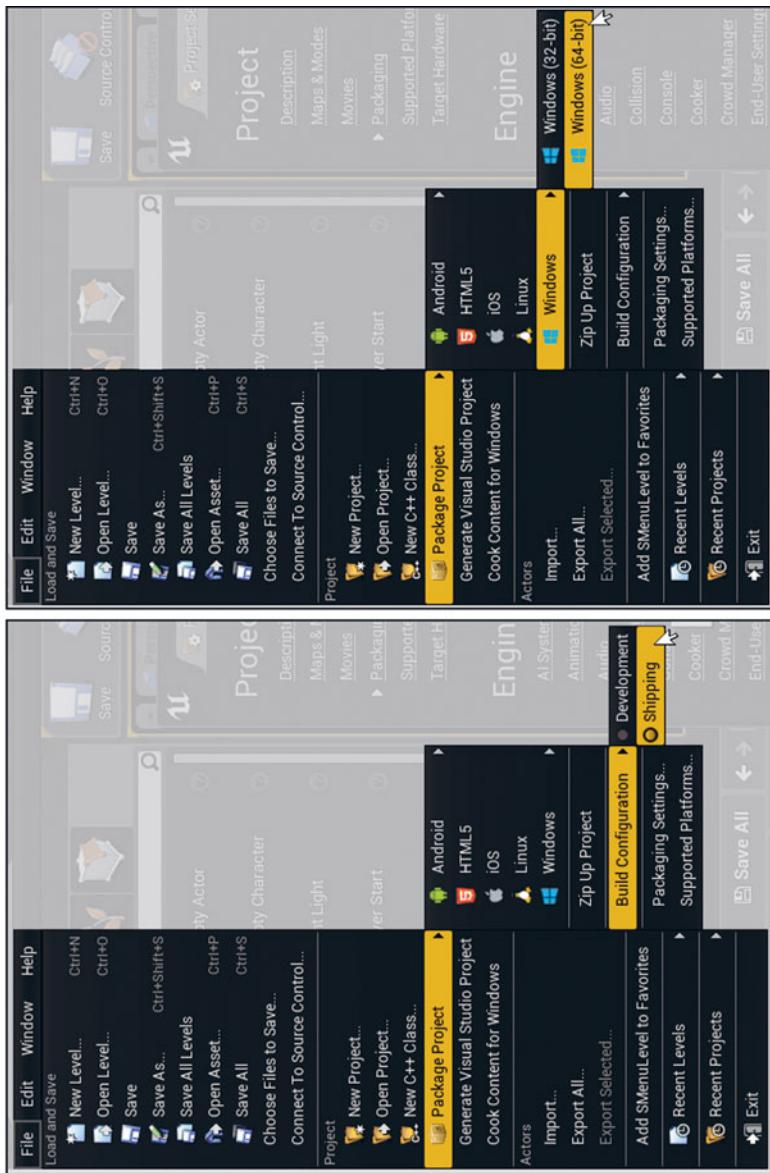


Рис. 23.1. Две команды меню, необходимые для быстрой упаковки проекта под Windows. На левом изображении показан пункт Shipping, а на правом — Windows (64-bit)

Если вы выберете пункт меню **File ⇒ Project Package ⇒ Windows ⇒ Windows (64-bit)**, появится диалоговое окно **Browse For Folder**, показанное на рис. 23.2, в котором можно отметить, где именно нужно поместить пакет. Важно сохранить пакет на жестком диске, где будет достаточно места для хранения проекта, а также дать папке название, соотносящееся с проектом.

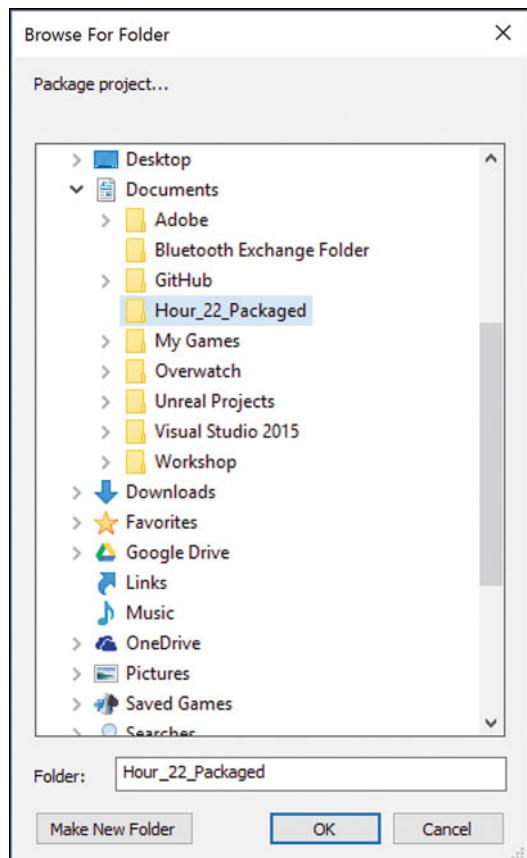


Рис. 23.2. Диалоговое окно **Browse For Folder**, где нужно выбрать расположение для вашего проекта

Когда вы нажмете кнопку **OK**, в правом нижнем углу экрана появится уведомление о начале процедуры упаковки (рис. 23.3).

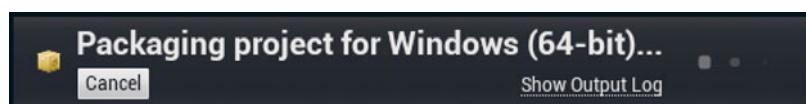


Рис. 23.3. Уведомление о прогрессе процедуры упаковки проекта

Процедура упаковки может занять некоторое время, особенно если перед этим контент нужно подготовить. Подробная информация о процессе упаковки выводится на панель **Output Log** (Журнал), хотя порой бывает трудно визуально воспринять информацию из огромного количества сообщений на экране.

Упрощенная (и с разбиением информации на категории) версия вывода также находится в журнале сообщений, который можно открыть, выбрав пункт меню **Window** ⇒ **Developer Tools** ⇒ **Message Log**. На рис. 23.4 показана панель **Message Log** с выбранной категорией **Packaging Results** (Результаты упаковки) когда возникает ошибка, а на рис. 23.5 та же ошибка, но с подробностями, на панели **Output Log**.

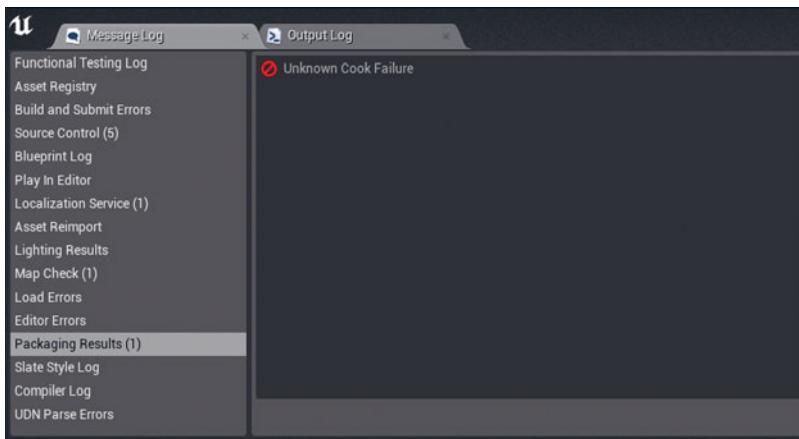


Рис. 23.4. Панель **Message Log** с ошибкой в результатах упаковки из-за неизвестной ошибки приготовления

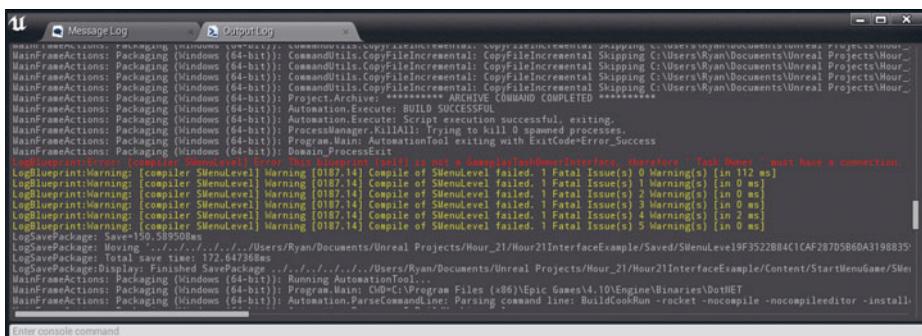


Рис. 23.5. Ошибка приготовления, упомянутая на рисунке 23.4, показана на панели **Output Log**. Эта ошибка связана с повреждением сети блюпринта, которая не может быть скомпилирована

Когда приготовление и упаковка завершены, контент помещается в расположении, указанном вами ранее в диалоговом окне **Browse For Folder**. Если вы теперь откроете эту папку с помощью любого обозревателя, то найдете доступные пакеты. По умолчанию в этой папке появляется новая папка с именем *WindowsNoEditor*, в которой находится весь упакованный проект, необходимый для запуска проекта на Windows.

В папке *WindowsNoEditor* найдите файл *ProjectName.exe*, который запускает проект, как показано на рис. 23.6.

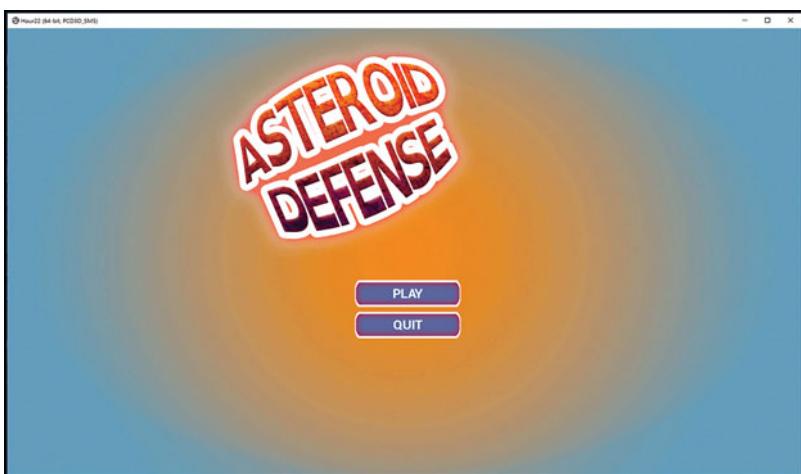


Рис. 23.6. Проект Hour 23, работающий автономно

ПОПРОБУЙТЕ САМИ

Упаковка проекта

Выполните следующие действия на компьютере с ОС Windows, чтобы попрактиковаться в упаковке проекта под Windows.

1. Откройте проект Hour 23 (или любой другой проект).
2. Выберите пункт меню **File** ⇒ **Package Project** ⇒ **Build Configuration** ⇒ **Shipping**.
3. Выберите пункт меню **File** ⇒ **Package Project** ⇒ **Windows** ⇒ **Windows (32-bit)**.
4. В появившемся диалоговом окне **Browse For Folder** создайте новую папку в выбранном месте и дайте ей подходящее название, например, *Hour23_packaged*.
5. Выберите эту новую папку и нажмите кнопку **OK**.
6. Подождите, пока в уведомлении в нижнем правом углу не появится сообщение «**Package Success**» и уведомление не исчезнет.

7. Откройте Проводник и перейдите в папку, созданную на шаге 4.
8. Откройте папку *WindowsNoEditor*.
9. Дважды щелкните по находящемуся в этой папке файлу с расширением .exe, имеющему название типа *НазваниеПроекта.exe* (например, *Hour23.exe*).
10. Наслаждайтесь вашей игрой в автономном режиме!

Ресурсы для упаковки под Android и iOS

На двух самых популярных мобильных платформах, iOS и Android, упаковка и развертывание осуществляется немного иначе. Процесс создания проекта для Android или iOS довольно сложный и требует предварительной работы.

Процедура требует, чтобы вы сначала настроили среду разработки с помощью Android Works SDK или iTunes. Актуальная информация доступна в документации по UE4 по адресам docs.unrealengine.com/latest/INT/Platforms/Android/GettingStarted/ и docs.unrealengine.com/latest/INT/Platforms/iOS/QuickStart/.

Изучить актуальную документацию — лучший способ убедиться, что вы сделаете все правильно при работе с мобильными устройствами в Unreal Engine 4.

Имейте в виду, что загрузка приложений и разработка для Apple App Store для iOS или Google Play Store для Android требует регистрации как разработчика, в обоих этих сервисах. Оба сервиса взимают с разработчика разовый взнос за размещение игры в магазине. Однако вам не придется платить взносы Apple App Store или Google Play Store, пока вы не захотите выставить игру на эти сервисы. Чтобы протестировать приложение на iOS без предварительной регистрации, вы можете использовать Xcode версии 7 или более позднюю версию на компьютере с macOS и развернуть проект напрямую. Если вы разрабатываете на OS Windows, вам придется сначала стать частью программы разработчиков и оплатить регистрационный взнос.

Кроме того, в обоих сервисах нужно пройти процедуру сертификации приложений, обязательную при размещении проекта. Когда вы предоставите всю необходимую информацию, указанную в документации UE4 Quick Start, необходимо выбрать в приложении режим **Distribution** (Распространение). Для переключения в режим **Distribution** выберите пункт меню **File ⇒ Package Project ⇒ Packaging Settings**, затем выберите категорию **Packaging** и включите опцию **For Distribution**. Этим действием вы указываете UE4 упаковать все необходимые сертификаты и подписать пакеты так, как того требуют магазины.

Доступ к профессиональным настройкам упаковки

Процедуры, описанной в предыдущих разделах, обычно достаточно для упаковки большинства проектов, но иногда разработчику требуется больше контроля над процессом. Параметры упаковки UE4 позволяют легко настроить множество дополнительных опций и параметров пакета. Использование этих параметров конфигурации является ключом к достижению успешного процесса упаковки, особенно если вы готовите игру сразу для нескольких платформ.

Чтобы открыть параметры упаковки, выберите пункт меню **File⇒ Package Project ⇒ Packaging Settings**, а затем выберите категорию **Packaging** (рис. 23.7).

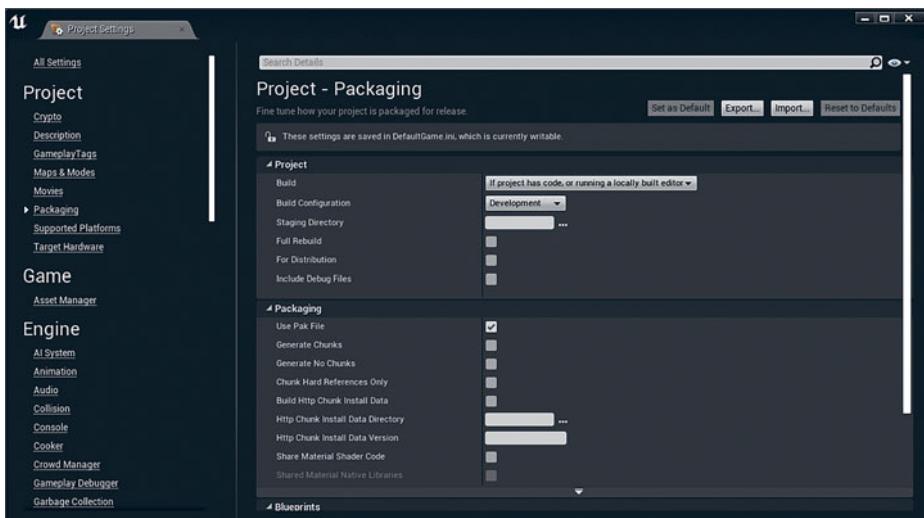


Рис. 23.7. Общие параметры вкладки **Packaging**

На вкладке **Packaging** вы можете настроить следующие параметры.

- ▶ **Build Configuration** (Конфигурация сборки). Вы можете указать конфигурацию сборки, для которой компилируется проект, содержащий C++ код. Если вы используете проекты с блюпринтами, эта опция не сильно отличается от тестирования с использованием конфигурации **Development** и тестирования для финального релиза с использованием **Shipping**.
- ▶ **Staging Directory** (Промежуточный каталог). Вы можете указать каталог для хранения упакованной сборки. Каждый раз, когда вы выбираете новый каталог с помощью меню **File ⇒ Package Project** в диалоговом окне **File Browser**, это поле будет автоматически обновляться с именем новой папки.

- ▶ **Full Rebuild** (Пересобирать целиком). Для проектов с C++ кодом это свойство определяет, будет ли пересобираться весь код, или только измененная часть. Для проектов с блюпринтами опцию лучше отключить.
- ▶ **For Distribution** (Для распространения). Этот параметр определяет, будет ли ваша игра находиться в режиме **Distribution**. Упаковка с этой опцией является обязательным требованием для размещения игры на App Store или Google Play.
- ▶ **Use Pak File** (Использовать файлы .pak). Этот параметр определяет, будут ли ассеты проекта упаковываться в виде отдельных файлов или в один пакет. Когда опция включена, все ассеты собираются вместе в один файл .pak. Если в проекте много файлов, включение этой опции может сильно упростить работу при распространении.
- ▶ **Include Prerequisites** (Включить предпосылки). Когда эта опция включена, все необходимые пререквизиты для упакованной игры будут включены в пакет. Это важно для распространения игры под неизвестные системы, для которых вы не можете сказать наверняка, установлено ли на них все необходимое для вашего проекта.

Резюме

В этом часе вы узнали, как отправить свой проект UE4 в свободный полет и отдать его от редактора. Вы узнали, что значит «приготовить» контент и готовить контент для Windows, а также как можно упаковать контент в отдельную папку для распространения.

Вопросы и ответы

Вопрос: Когда я запускаю игру автономно, загружается не та карта. Как это исправить?

Ответ: Выберите пункт меню Edit ⇒ Project Settings, затем выберите вкладку Maps & Modes и установите в свойство Default Game Level ту карту, которая должна быть стартовой в вашей игре.

Вопрос: Когда я копирую упакованный каталог на другой компьютер, я получаю сообщение об ошибке, но при этом на моем компьютере все работает правильно. В чем дело?

Ответ: В данном случае возможен целый ряд причин, в зависимости от того, какая именно ошибка возникает. Одна из самых распространенных причин заключается в том, что на целевом компьютере могут отсутствовать необходимые предпосылки, которые требуются для работы проекта Unreal

Engine 4. Чтобы устранить эту проблему, нужно упаковать предпосылки вместе с игрой. Для этого в окне Packaging нужно включить свойство Include Prerequisites.

Вопрос: В процессе приготовления в журнале сообщений появляется неизвестная ошибка. В чем дело?

Ответ: На этапе приготовления может возникнуть огромное количество проблем. Если вы видите в журнале сообщений неизвестную ошибку, потребуется некоторое время, чтобы изучить выходной журнал. В этом журнале ошибки окрашены в красный цвет, поэтому вы можете пролистать журнал и найти красный текст. Часто причина проста — например, программа не может найти удаленный объект или файл, поэтому лучше всего просмотреть журнал на наличие очевидных проблем.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Как называется процесс, который заключается в преобразовании контента из специальных форматов редактора в форматы, подходящие для работы на целевых платформах.
2. Истинно или ложно высказывание: 64-разрядные процессоры могут работать с 32-разрядными исполняемыми файлами, созданными в Unreal Engine 4.
3. Истинно или ложно высказывание: при упаковке проекта из редактора загружаться будет тот уровень, который открыт в данный момент.

Ответы

1. «Приготовление». Не путайте с промежуточным хранением, при котором данные хранятся в подготовленном месте на компьютере перед развертыванием на целевом устройстве.
2. Истина. 64-разрядные процессоры могут выполнять 32-разрядные программы, но 32-разрядные процессоры не могут выполнять 64-разрядные программы.
3. Ложь. Уровень, который будет загружаться в автономном проекте первым, указан в параметрах в диалоговом окне **Project Settings** на вкладке **Maps & Modes**.

Упражнение

Создайте и упакуйте новый проект, чтобы отработать навыки упаковки.

1. Открыв лаунчер, создайте новый проект UE4 с любым шаблоном на ваш вкус. Чтобы ускорить процесс, не импортируйте Starter Content.
2. Откройте новый проект.
3. Выберите пункт меню **File** ⇒ **Package Project** ⇒ **Build Configuration** ⇒ **Shipping**.
4. Выберите пункт меню **File** ⇒ **Package Project** ⇒ **Windows** ⇒ **Windows (32-bit)**, чтобы упаковать проект под Windows.
5. Выберите подходящую папку для хранения проекта.
6. Когда проект будет упакован, откройте Проводник, найдите папку с проектом и файл с расширением .exe. Дважды щелкните по файлу .exe, чтобы закончить это упражнение.

24-Й ЧАС

Работа с мобильными устройствами

Что вы узнаете в этом часе

- ▶ Как разрабатывать игры для мобильных устройств
- ▶ Как использовать сенсорные устройства
- ▶ Как использовать данные о движении устройства

Самый большой и самый быстрорастущий рынок компьютерных игр — это не игры на приставке или ПК, а игры для мобильных платформ. Рынок мобильных телефонов в последние годы растет экспоненциально, и количество игр на мобильные платформы (и прибыль от них) измеряется космическими числами. Мобильные устройства не такие мощные, как консоли и ПК, поэтому нужно приложить изрядное количество усилий, чтобы игра вообще заработала на мобильном телефоне. В этом часе вы узнаете об ограничениях большинства мобильных платформ, о том, как в игре работают события касаний, как использовать виртуальные джойстики для управления аватаром и гироскоп, чтобы создать схему управления, присущую только мобильным устройствам.

ПРИМЕЧАНИЕ

Подготовка к практике

В этом часе мы продолжим работу над проектом ArcadeShooter, который начали в часе 20 и продолжили в часах 21, 22 и 23. Если хотите, для простоты можете также использовать готовый проект H23_ArcadeShooter, находящийся в сопроводительных файлах для этого часа в папке *Hour_23* (файлы проектов можно скачать по адресу: http://addons.eksмо.ru/it/UE_24.zip).

Завершив этот урок, сравните полученные результаты с проектом H24_ArcadeShooter, который находится в папке *Hour_24* в сопроводительных файлах.

ПРИМЕЧАНИЕ

Тестирование на мобильной платформе

При разработке мобильной игры приходится выполнять дополнительное тестирование на имеющемся оборудовании. Этот процесс может быть непростым и переменчивым и на различных операционных системах и устройствах может происходить по-разному.

Чтобы убедиться в правильной настройке устройства на ОС Android, ознакомьтесь с документацией: docs.unrealengine.com/latest/INT/Platforms/Android/GettingStarted/.

Чтобы подготовить устройство на iOS к развертыванию, ознакомьтесь с документацией: docs.unrealengine.com/latest/INT/Platforms/iOS/QuickStart/index.html.

Разработка приложений для мобильных устройств

В целом мобильные устройства уступают консолям и ПК во многих отношениях. Они медленнее, у них невысокие графические возможности, меньше оперативной и постоянной памяти, и экран тоже меньше. Однако аппаратная часть с каждым годом становится мощнее, поэтому эти ограничения от года к году становятся менее жесткими.

В некотором смысле ежегодный прогресс в области мобильных устройств обгоняет прогресс игровых приставок и ПК. Как следствие, за этим рынком труднее следить, а разработка с ориентиром на конкретные устройства по-прежнему важна. Самое последнее устройство iPhone или Samsung может быть способно обрабатывать высококачественную графику, но широкое распространение новейших аппаратных средств происходит не сразу, и подавляющее большинство потенциальных пользователей используют устройства двух- или трехгодичной давности. Еще сильнее усложняет ситуацию рост популярности планшетов и сенсорных ноутбуков — то есть некоторые устройства могут быть не слабее ПК, но находиться при этом в категории мобильных устройств.

Поскольку эти устройства развиваются столь быстро, важно определить минимальные требования к проекту. Unreal Engine 4 позволяет гибко регулировать качество и функционал, что позволяет включить одни настройки при работе на Microsoft Surface или iPad и отключить их для мобильных телефонов.

Зная, что требования к мобильным платформам быстро меняются, в этом разделе мы рассмотрим некоторые приемы работы с мобильными устройствами.

Предварительный просмотр для мобильных устройств

При разработке для мобильных устройств следует соблюдать несколько правил. Бывает трудно понять сразу, как возможности вашего устройства повлияют на внешний вид проекта во время работы в редакторе. К счастью, Unreal Engine 4 позволяет отобразить набор функций материалов и уровни рендеринга вашего мобильного устройства.

Чтобы включить эту функцию, на панели инструментов панели **Viewport** выберите пункт меню **Settings** ⇒ **Preview Rendering Level** ⇒ **Mobile/HTML 5** ⇒ **Default Mobile/HTML5 Preview** (рис. 24.1).

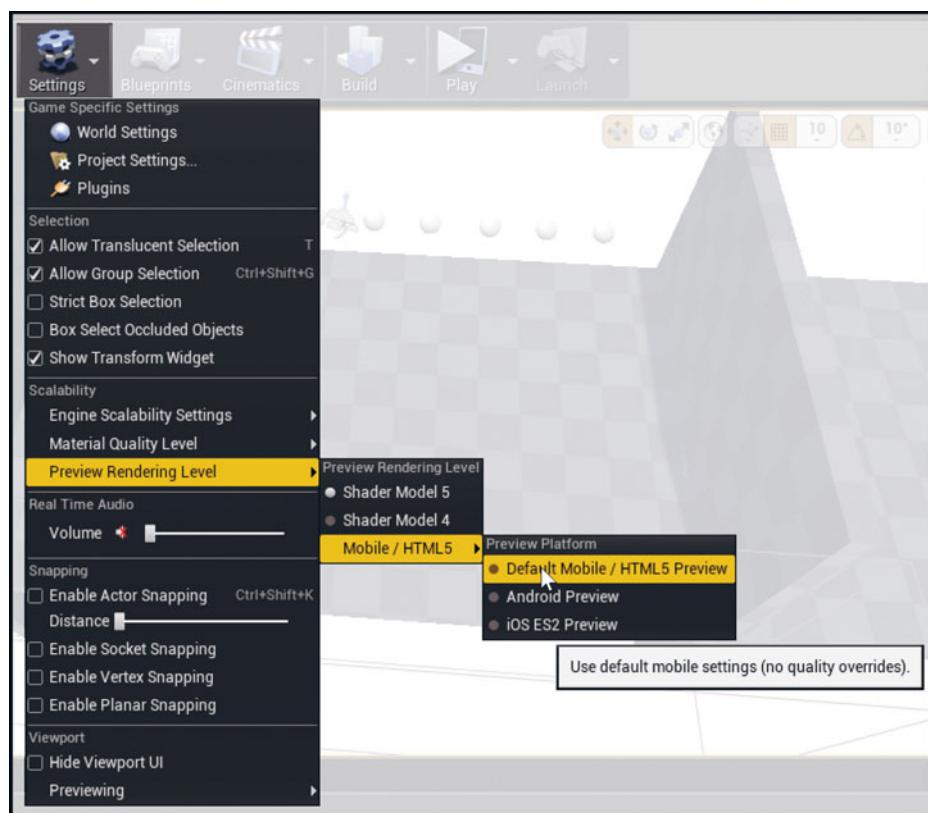


Рис. 24.1. Настройка уровня рендеринга для предварительного просмотра позволяет перекомпилировать все материалы и шейдеры под более жесткие ограничения, аналогичные тем, которые есть у реальных мобильных устройств

Если вы выберете пункт **HTML5 Preview**, то внешний вид уровня, скорее всего, станет ближе к тому, как проект будет выглядеть на мобильном устройстве. Однако уровень рендеринга не может целиком определить конечный результат на вашем устройстве, и лучше проводить «живые» тесты на устройстве.

Так как аппаратная часть мобильных устройств изменяется быстро и хаотично, автоматической оптимизации материалов может быть недостаточно для всех устройств. В этих случаях для отображения ресурсоемких материалов следует использовать нод **Quality Switch**, чтобы убрать из материала «затратные» операции. На рис. 24.2 показан пример удаления ресурсоемкого параметра из всех конфигураций, кроме самой высококачественной.

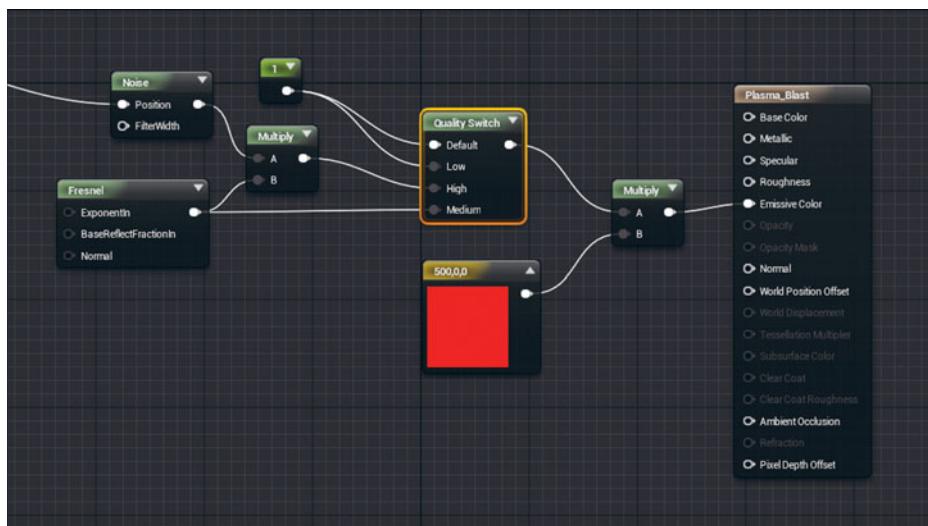


Рис. 24.2. Нод **Quality Switch** используется, чтобы удалить ресурсоемкий нод **Noise** из материала, если качество настроено на среднее или низкое. Также удаляется нод **Fresnel**, если установлен низкий уровень качества

Если вы разместите ноды **Quality Switch** во всех материалах проекта, чтобы удалить «затратные» операции, то сможете установить уровень качества материала для проекта, выбрав пункт меню **Settings** ⇒ **Material Quality Level**, как показано на рис. 24.3. Здесь используется три варианта — **Low**, **Medium** и **High** — разница между которыми проявляется при использовании нода **Quality Switch** для материалов в проекте.

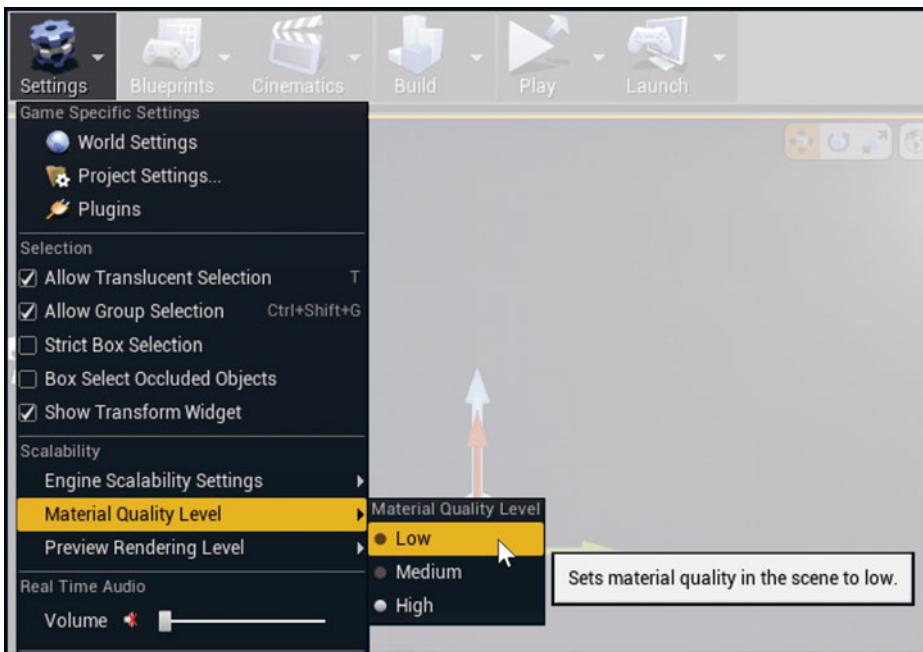


Рис. 24.3. Вы можете установить в настройках уровень качества материала проекта на Low, Medium и High

У многих устройств уникальные соотношения сторон и разрешение экрана. При сборке проекта важно протестировать проект на тех параметрах разрешения, которые, предположительно, будут у устройств пользователей.

На рис. 24.4 показано, как установить разрешение для мобильной платформы в разделе Play панели Editor Preferences, в разделе Play in Standalone Game. Часто используемые разрешения устройств можно выбрать в выпадающем списке Common Window Sizes. Затем вы можете использовать панель инструментов и выбрать пункт меню Play ⇒ Mobile Preview.

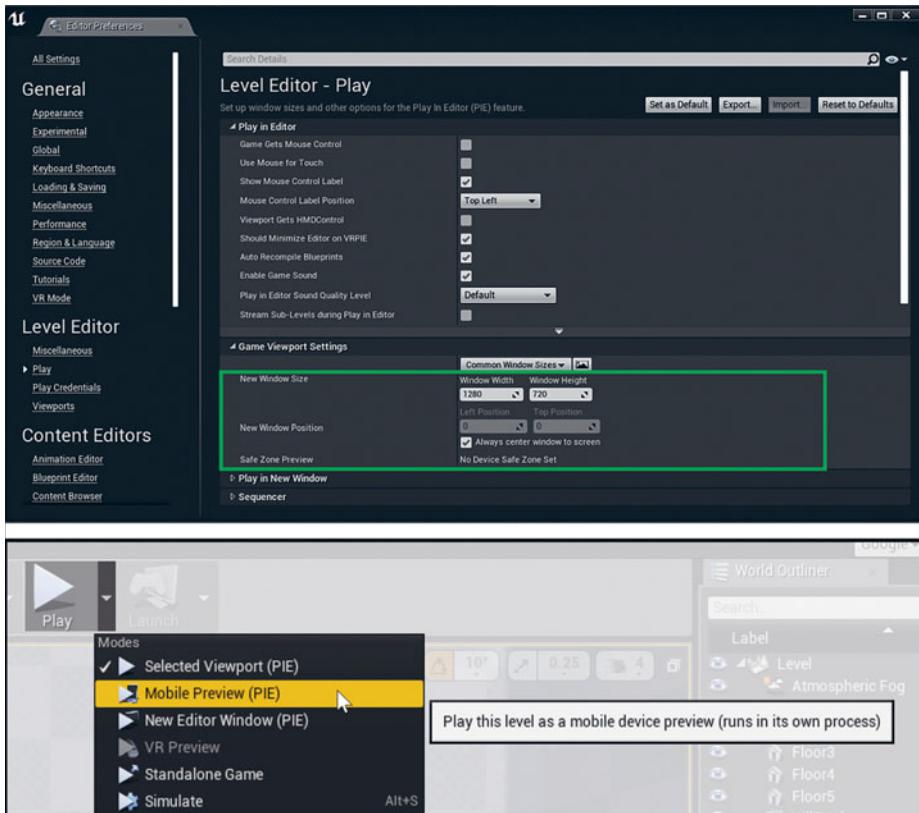


Рис. 24.4. Команда меню **Play** ⇒ **Mobile Preview** создает автономную версию вашей игры в выбранном разрешении мобильного устройства

Оптимизация для мобильных устройств

Существует ряд приемов, которые следует использовать при разработке для мобильных устройств. Иногда в проекте приходится игнорировать одну из этих рекомендаций, но будьте очень осторожны, так как игнорирование методик оптимизации может серьезно снизить производительность вашего проекта.

- ▶ **Всегда запекайте освещение.** Динамическое освещение может значительно повлиять на затратность рендеринга на любой платформе, и большинство мобильных устройств работает особенно медленно, если приходится обрабатывать динамическое освещение. Всегда, когда это возможно, устанавливайте статическое освещение или используйте только один динамический направленный свет, установленный в режим **Stationary**. Однако более мощные мобильные устройства вполне способны обрабатывать динамичные точечные источники света, благодаря функции под названием **Max Dynamic**

Point Lights, которая настраивается в разделе **Rendering** на панели **Project Settings**. Функция **Max Dynamic Point Lights** поможет создать на сцене динамические источники света с меньшими затратами ресурсов за счет ограничения количества точечных источников, которые единовременно освещают каждый отдельно взятый пиксель.

- ▶ **Избегайте использования подвижных источников света при работе на мобильных устройствах.** Даже при наличии функции **Max Dynamic Point Lights** подвижные источники всегда обрабатываются затратнее стационарных или статических источников.
- ▶ **Отключите постобработку, где это возможно.** Можно оставить постобработку Temporal AA, Vignettes, и Film, но даже в этом случае производительность пострадает. Обязательно отключите Bloom, Depth of Field и Ambient Occlusion.
- ▶ **Разумно используйте замаскированные и полупрозрачные материалы.** Перерисовка — процесс, когда устройству приходится затенять один и тот же пиксель более одного раза, что чрезвычайно затратно. При использовании прозрачных или замаскированных материалов убедитесь, что они расположены лишь на небольшой части экрана. Можно использовать режим просмотра **Shader Complexity** (рис. 24.5), чтобы понять, где вы переборщили с количеством слишком сложных материалов. Кроме того, вы можете использовать консольную команду `viewmode shadercomplexity` в мобильном предпросмотре.



Рис. 24.5. Режим просмотра **Shader Complexity** очень полезен для поиска дорогих материалов и перерисовки в проекте

Режим просмотра **Shader Complexity** позволяет отобразить затратность команд на пиксель. Этот режим отображает пиксели цветами от ярко-зеленого (очень дешево) до красного (дорого) и белого (очень дорого). Поскольку в случае с полуопрозрачными и замаскированными материалами каждый пиксель обсчитывается несколько раз, перерисовка создает аппаратные затраты и отображается белым.

- ▶ **Старайтесь, чтобы материалы были как можно проще, с небольшим количеством и текстур.** На большинстве мобильных устройств используется

всего пять сэмплов текстур, но принцип использования как можно меньшего количества сэмплов текстур будет хорошим для любой платформы.

- ▶ Убедитесь, что в светящихся непрозрачных материалах используется только две текстуры. Поскольку в UE4 используется модель затенения на основе физики, легко достичь такой оптимизации путем упаковки текстур. В первой текстуре каналы RGB должны соответствовать контакту Base Color, а контакт Alpha отвечает за параметр **Roughness**. Для первой текстуры следует использовать сжатие TC_Default. Вторая текстура должна иметь Normal Map в RGB и сжатие TC_NormalMap, а параметр **Alpha** должен быть пустым. Это означает, что для контактов Specular и Metallic не используется сэмплов, и вместо них должны использоваться константы. На рис. 24.6 показан пример материала, который имеет именно такой формат.

Поскольку скала в этом примере не является металлической, а затенение в основном зависит от шероховатости, входные контакты Specular и Metallic могут быть заменены константами. В то же время вход Roughness может быть упакован в ту же текстуру, которая содержит базовый цвет RGB. И, наконец, входу Normal требуется собственная текстура RGB, чтобы описать нормальное изменение поверхности скалы.

Этот метод не будет работать для объектов, в которых есть и металлические, и неметаллические детали. В этих случаях может потребоваться еще один Texture Sampler для управления «металличностью» материала.

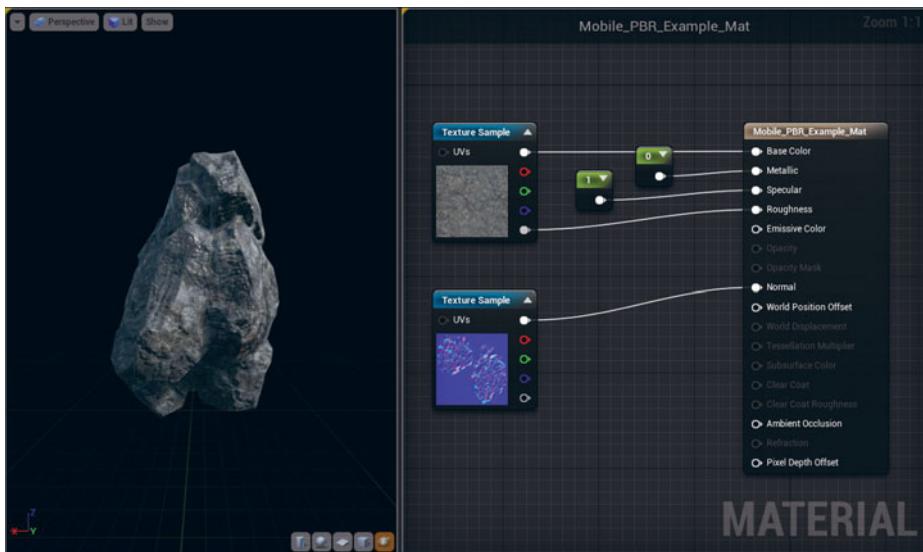


Рис. 24.6. Для большинства материалов достаточно двух текстур, чтобы создать эффективный, но высококачественный материал на основе физики.

- ▶ Не подключайте UV-модификации (например, масштабирование) в сэмплы текстур на графике материала. Вместо этого включите опцию **Num Customized UVs**, чтобы выполнить UV-масштабирование на вершины (рис. 24.7). Вы можете включить UV для отдельных вершин на панели **Details**, когда на графике материала ничего не выбрано. Тогда вычисление UV выполняется для каждой вершины, а не для каждого пикселя, что очень важно для мобильных графических процессоров. Когда масштабированные координаты текстуры подключены ко входу UV, шейдер вершины обрабатывает график и заменяет координаты текстуры на каждой вершине с учетом масштабирования (изменения).

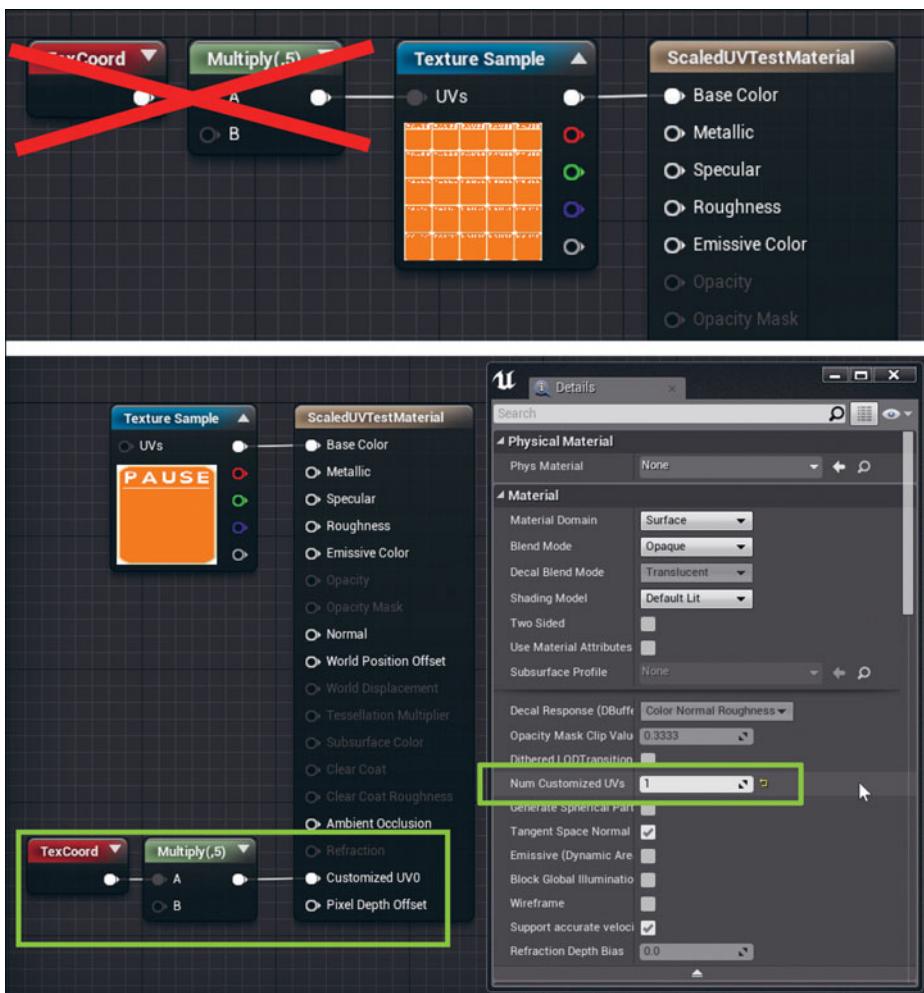


Рис. 24.7. Оба графика, приведенные на этом рисунке, дают одинаковый визуальный результат, но второй график будет намного дешевле на многих устройствах

Этот процесс позволяет масштабировать UV без затрат на масштабирование каждого пикселя во время рендеринга, и без неудобств масштабирования в исходном 3D-пакете (например, Maya, Max, Houdini, Blender).

- ▶ **Сохраняйте количество треугольников минимальным с любой точки обзора.** Использование упрощенных художественных стилей поможет уменьшить количество треугольников в проекте. Если это не представляется возможным и нужны высокополигональные мешы, постарайтесь уменьшить количество актеров и мешей, видимых с каждой точки обзора. Использование консольной команды **Stat RHI** в режиме мобильного предпросмотра может помочь вам определить количество треугольников на сцене. В разделе **Counters** в строке Triangle Drawn указано число треугольников на текущем виде (см. рис. 24.8).

На любом виде должно быть как можно меньше вызовов функции **Draw** (количество объектов, находящихся на экране в одном фрейме). Количество вызовов функции **DrawPrimitive** можно увидеть с помощью команды **Stat RHI** в консоли предварительного просмотра (рис. 24.8).

Максимальное допустимое количество вызовов функции **Draw** для каждого проекта зависит от оборудования, на которое вы ориентируетесь. Но, вне зависимости от устройства, старайтесь держать это количество минимальным, чтобы сберечь производительность.

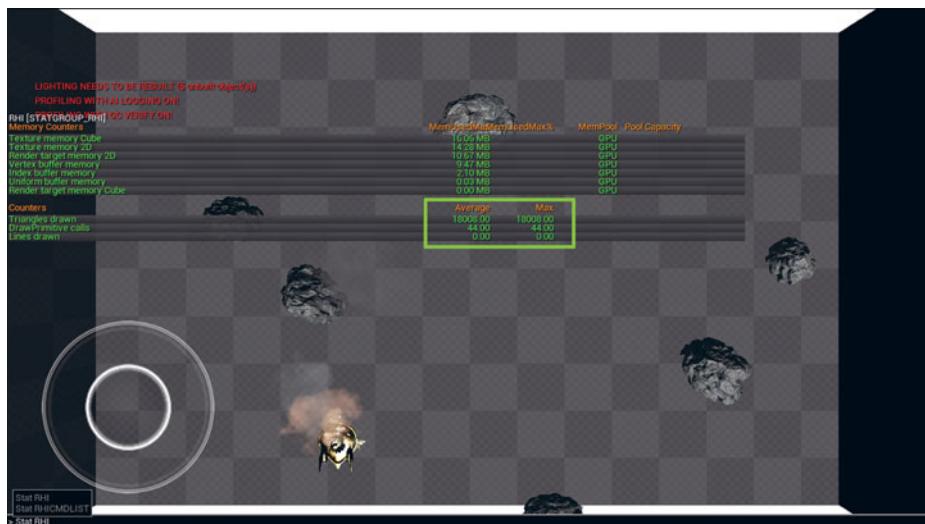


Рис. 24.8. Использование команды **Stat RHI** для вывода текущего количества треугольников и вызовов функции **Draw** в каждом кадре. Консоль можно вызывать с помощью клавиши «~»

- ▶ Всегда используйте квадратные текстуры с размерностью, равной степени двойки (например, 32×32 , 64×64 , 128×128 , 256×256 , 512×512 , 1024×1024). В этом случае вы получите минимальное количество напрасно используемой памяти. Вы можете использовать команду `listtextures` в консоли мобильного предварительного просмотра, чтобы увидеть, как вы используете память для хранения текстур.

Установка платформы развертывания в редакторе

В Unreal Engine 4 есть общие пресеты для проектов, в зависимости от того, ориентирован ли проект на консоли/ПК или мобильные устройства/планшеты. Эти пресеты касаются некоторых функций рендеринга и входов и информируют Unreal Engine о том, под какую платформу вы разрабатываете игру, чтобы снять с вас часть умственной работы.

При создании проекта вы можете выбрать опцию Console/PC. Если вы позже решите, что проект переключится на мобильные устройства/планшеты, то можете задать настройку **Mobile/Target**. Чтобы сделать это, перейдите на панель **Project Settings** и в категории **Target Hardware** присвойте параметру **Optimize Project Settings For** значение **Mobile/Tablet** и **Scalable 3D or 2D**. В разделе **Pending Changes** этого окна возникнут изменения, но вам будет необходимо перезапустить редактор, чтобы увидеть все изменения (рис. 24.9).

В разделе **Pending Changes** отображаются устанавливаемые автоматически настройки проекта. Нажмите кнопку **Restart Editor**, чтобы принять изменения. Внесение изменений приводит к необходимости перекомпилировать каждый освещенный материал в вашем **Content Browser**, на что потребуется некоторое время.

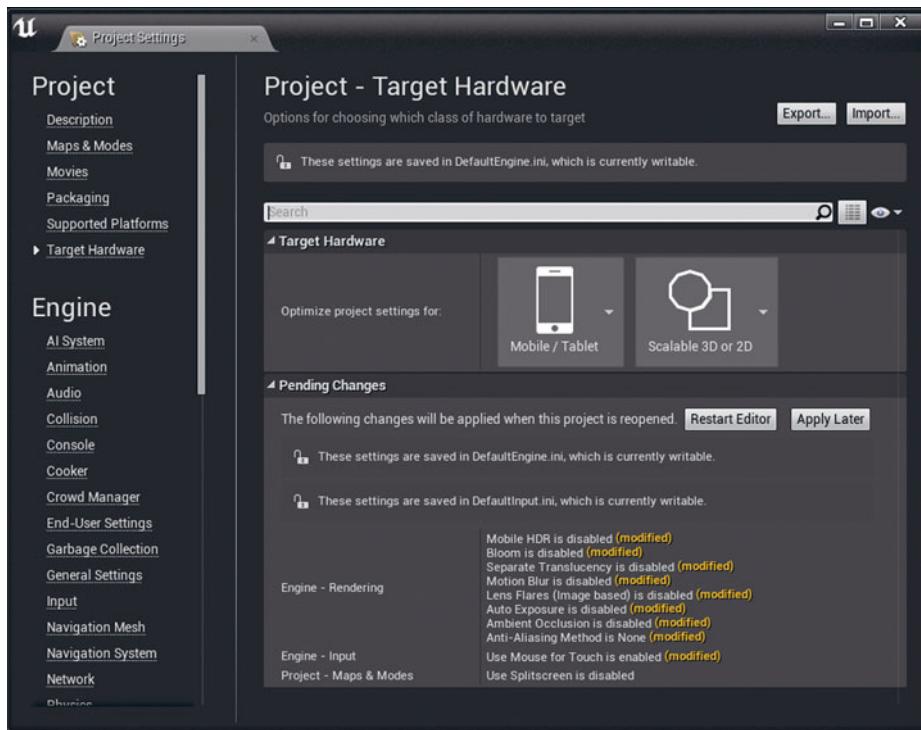


Рис. 24.9. Настройки аппаратных средств на панели Project Settings после присвоения параметра Optimize Project Settings For равным Mobile/Tablet и Scalable 3D or 2D.

Пресет **Mobile/Tablet** отключает некоторые функции рендеринга и эффекты постобработки.

- ▶ **Раздельная прозрачность (Separate Transparency).** Способность рендеринга некоторых полупрозрачных материалов после постобработки. Она обычно используется для рендеринга стекла в сочетании с глубиной резкости и является очень затратной, поэтому не применяется на мобильных устройствах.
- ▶ **Размытие в движении (Motion Blur).** Постобработка, которая размывает экран и актеров, основываясь на относительном движении. Эта функция тратит немало ресурсов, что затратно для мобильных устройств, и поэтому на мобильных устройствах не используется.
- ▶ **Блики объектива (Lens Flares) на основе изображения.** Постобработка, создающая аппроксимацию бликов, рассчитывая высокие значения динамического диапазона света на сцене. Эта оценочная полноэкранная постобработка относительно затратна и не включается на мобильных устройствах.

- ▶ **Автоэкспозиция (Auto Exposure).** Эта функция постобработки оценивает освещение текущей сцены и регулирует экспозицию и видимость. Как и другие эффекты постобработки, недоступна на мобильных устройствах. Также поддерживается свойство Auto Exposure Bias.
- ▶ **Окружающая окклюзия (Ambient Occlusion).** Еще одна затратная функция постобработки. Она отключает процесс окклюзии экранного пространства, который требует, чтобы при рендеринге буфер глубины обрабатывался несколько раз для генерации контактной тени. Этот тип эффекта непозволительно дорог для большинства мобильных устройств и поэтому недоступен.
- ▶ **Сглаживание (Anti-Aliasing).** Постобработка, которая пытается убрать неровные края и субпиксельные артефакты. Сглаживание на мобильном оборудовании не включается. На мобильных устройствах доступно временное сглаживание, но оно может привести к визуальному дрожанию и движению объектов.

Помимо отключения различных высокозатратных функций, пресет **Mobile/Target** задает функцию Mouse for Touch, которая позволяет эмулировать мышь через сенсорный дисплей.

Установка пресета **Scalable 3D or 2D** отключает еще две высокозатратные функции.

- ▶ **Мобильный HDR (Mobile HDR).** Эта базовая функция позволяет мобильному устройству обрабатывать буферы с высоким динамическим диапазоном. Она обеспечивает работу всех световых эффектов. Буферы HDR-рендеринга используются в различных функциях визуализации и эффектах, и отключение этого параметра заметно снижает используемый визуализатором объем памяти. Побочный эффект заключается в том, что все функции рендеринга, которые зависят от этих HDR-буферов, перестают работать.
- ▶ **Свечение (Bloom).** Эта функция постобработки берет размытую форму подсветки на сцене и помещает ее на верхний слой визуализации. Она позволяет светящимся или ярко подсвеченным объектам словно сиять, и сильно зависит от включенного мобильного рендеринга HDR, поскольку во многих случаях свечение применяется только к пикселям со значениями больше, чем 1.0.

ВНИМАНИЕ

Scalable 3D or 2D и освещение на мобильных устройствах

При выборе пресета **Scalable 3D or 2D** функция рендеринга Mobile HDR отключена.

В результате все функции освещения на мобильных устройствах отключаются,

захватывая и статическое запечченное освещение.

В окне мобильного предпросмотра в редакторе это изменение не отображается, так что визуальное отображение вашего проекта на реальном устройстве может быть иным. Для имитации эффекта при работе в редакторе, вы должны удалить или отключить все источники света в сцене.

Если в мобильном проекте вам требуются световые эффекты, не используйте пресет *Scalable 3D or 2D*.

Сенсорные устройства

Одним из главных нововведений мобильных устройств стал сенсорный ввод. Полное соответствие между действиями пальцами и происходящим на экране, безусловно, стало очень важным компонентом мобильных устройств.

Суть работы сенсора в создании множества вариантов взаимодействия с одним и тем же типом ввода. Это может быть, например, имитация аппаратного ввода, виртуальная клавиатура, а также виртуальные джойстики, а могут быть и совершенно новые применения сенсора.

Виртуальные джойстики

Когда вы преобразуете проект в мобильный, UE4 помогает в обработке одного из самых сложных типов ввода, создавая пару виртуальных джойстиков. Эти джойстики (см. рис. 24.10) представляют собой виртуальный вариант отображения входных осей. Внутренний круг имитирует сам джойстик, а внешний круг обозначает зону, внутри которой джойстик можно перемещать.

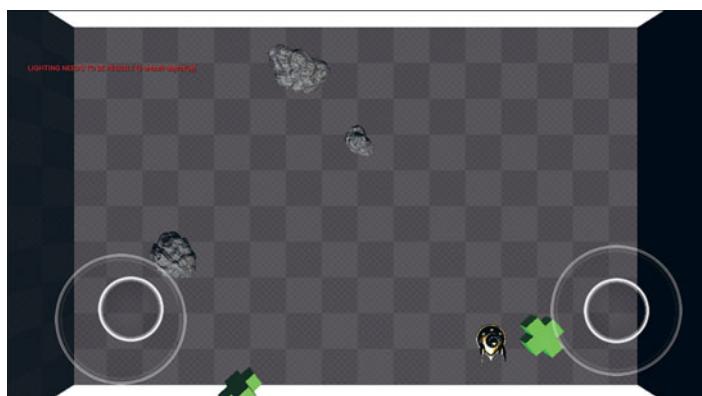


Рис. 24.10. Мобильная версия проекта Hour 23. Две пары белых кругов слева и справа — виртуальные джойстики, созданные UE4. Левый джойстик управляет движением корабля

Такие виртуальные джойстики могут быть особенно удобны, если вы создаете схему управления, требующую наличия двух джойстиков. Вы можете отключить джойстики на панели **Project Settings**, перейдя в категорию **Input** и в раскрывающемся списке **Default Touch Interface** выбрав пункт **Clear** (рис. 24.11).

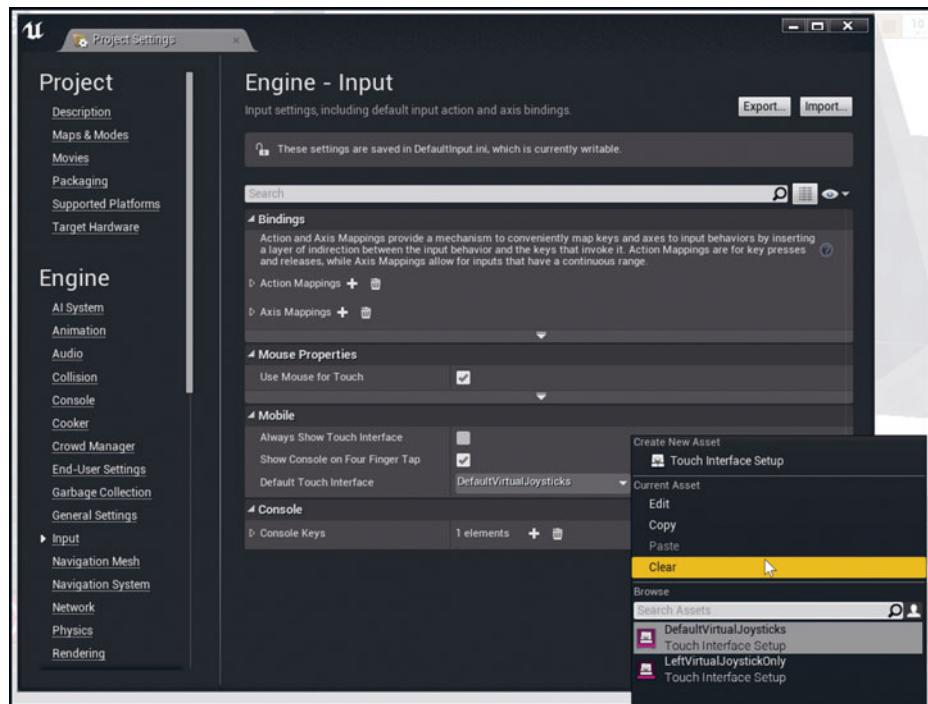


Рис. 24.11. Категория **Input** на панели **Project Settings** с выделенной опцией **Clear** для сенсорного интерфейса

Сенсорный интерфейс — это UAsset, позволяющий создавать и выводить сенсорный пользовательский интерфейс. В таком интерфейсе может выводиться, например, виртуальный джойстик или кнопки.

Скажем, в вашем аркадном шутере вы используете только один направленный вход для управления движением, и правый джойстик вам не нужен. Вы можете легко удалить правый джойстик, присвоив значение **LeftVirtualJoystickOnly** параметру **Default Touch Interface** (Сенсорный интерфейс по умолчанию) в категории **Input** панели **Project Settings**.

ПОПРОБУЙТЕ САМИ**Удаление правого джойстика**

Для вашей игры достаточно всего одного виртуального джойстика. Вы можете использовать настройки по умолчанию, чтобы оставить только один джойстик.

1. В проекте Hour 23 откройте панель **Project Settings**.
2. Перейдите в категорию **Input**.
3. Найдите поле **Default Touch Interface**.
4. Нажмите на стрелку вниз в поле свойств.
5. В нижнем правом углу выпадающего списка выбора нажмите на значок в виде глаза **View Options**.
6. Убедитесь, что выбрана опция **Show Engine Content**. Это позволит найти и выбрать сенсорные интерфейсы, которые есть у движка UE4 по умолчанию.
7. В раскрывающемся списке найдите UAsset интерфейса **LeftVirtualJoystickOnly**.
8. С помощью панели инструментов вы можете оценить изменения и убедиться, что отображается только один джойстик.

События сенсора

Хотя сенсорный интерфейс отлично подходит для создания виртуальных джойстиков, некоторые входы лучше обрабатывать напрямую через блюпринты. В этой книге, добавляя новые входы, вы просто подключали вход с маппингом действий в категории **Input** панели **Project Settings**. Сенсорные события немного отличаются в том смысле, что они обрабатываются непосредственно в блюпринтах.

СОВЕТ**Использование сенсорного интерфейса для нажатия кнопки**

Вы можете использовать сенсорный интерфейс для эмуляции кнопок контроллера, но только для маппинга осей. Если вы создаете собственный сенсорный интерфейс, все входы должны выполнять именно маппинг оси, а не маппинг действий.

Для достижения наиболее четкого управления установите сенсорные события с помощью нода **InputTouch** в графе событий. На рис. 24.12 показан нод **InputTouch** и его свойства.

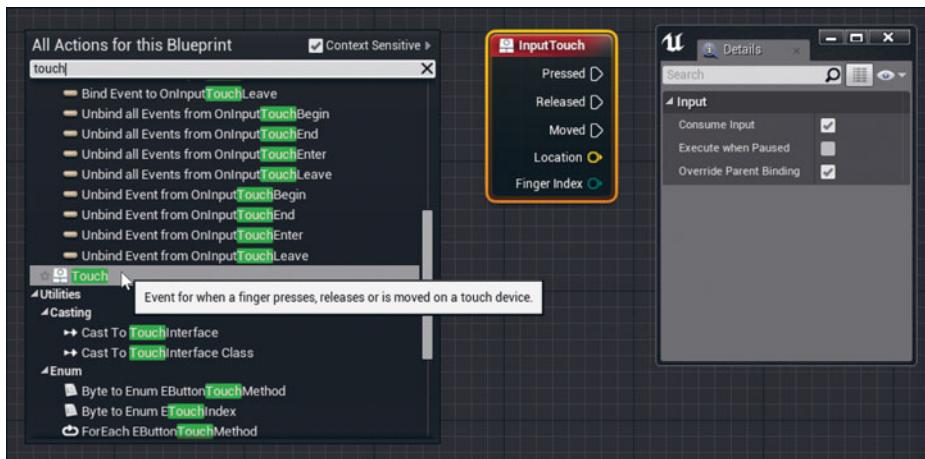


Рис. 24.12. Нод событий `InputTouch` можно найти по слову `touch` в контекстном меню блупринта. Входы и свойства этого нода похожи на входы и свойства любого другого нода событий

Нод событий `InputTouch` имеет три ехес-контакта и два свойства.

- ▶ **Pressed.** Срабатывает один раз для каждого пальца при каждом касании экрана пальцем.
- ▶ **Released.** Срабатывает один раз, когда палец отрывается от сенсора.
- ▶ **Moved.** Срабатывает каждый тик, когда палец движется по сенсору.
- ▶ **Location.** Возвращает текущее местоположение пальца на экране, где $[0,0]$ — координата верхнего левого угла в пикселях. Координата может быть преобразована в мировую координату с помощью нода `Dereference Screen to World`.
- ▶ **Finger Index.** Уникальный индекс, который определяет, вход от какого из пальцев обрабатывается в текущий момент. Порядок обработки зависит от порядка касания и не зависит от реальных пальцев пользователя. Вы можете использовать это свойство в сочетании с нодами условия и сравнения, чтобы обрабатывать несколько входов.

У этого нода также есть свойство, которое стоит отметить отдельно.

- ▶ **Consume Input.** Всякий раз, когда касание задевает несколько актеров, первый актер, у которого установлено это свойство, обрабатывается как единственный актер для этого сенсорного события. Если вы хотите, чтобы событие обрабатывали несколько различных актеров, у всех нодов `InputTouch` у каждого актера флаг **Consume Input** должны быть отключен.

ПОПРОБУЙТЕ САМИ

Настройка сенсорной кнопки для стрельбы

Поскольку у мобильных устройств нет ни мыши, ни кнопок, в аркадном шутере вам нужно настроить аватара на выстрел всякий раз, когда пользователь касается устройства. Выполните следующее.

1. На панели Content Browser проекта Hour 23 дважды щелкните по блюпринту **Hero_Spaceship**.
2. Под нодом InputAction Shoot разместите новый нод события **InputTouch**, найдя его поиском по слову **touch**.
3. Перетащите контакт вывода нода события InputTouch и присоедините его к тому же ноду SpawnActor, который используется нодом InputAction Shoot. Оба события должны быть подключены к одному ноду. На рис. 24.13 показано, как в этом случае должен выглядеть граф событий. Вы можете сравнить свои результаты с готовым проектом Hour 24.

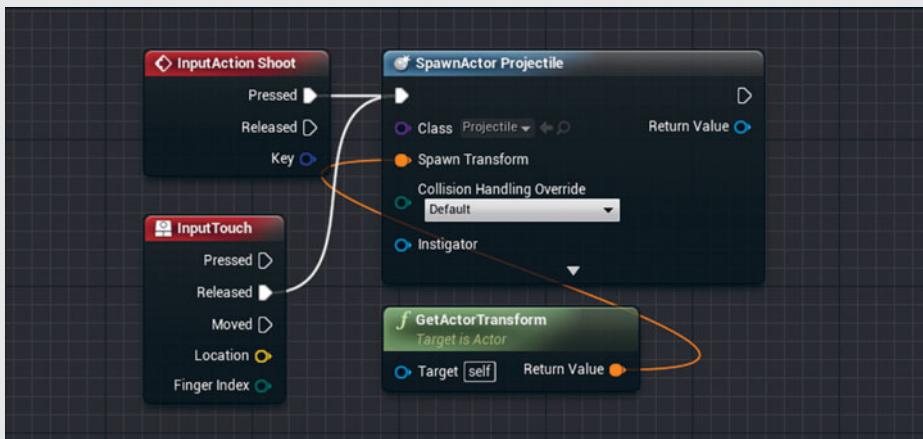


Рис. 24.13. Оба события InputAction Shoot и InputTouch Released привязаны к одному и тому же поведению спаунинга

ПРИМЕЧАНИЕ**Взаимодействие сенсора UMG**

В проекте Hour 22 есть стартовый экран UMG. К счастью, UMG обрабатывает сенсорные события так же, как щелчки мышью, так что ваше стартовое меню сохранит функциональность.

Использование данных о движении устройства

Большинство мобильных устройств оснащено встроенным гироскопом и акселерометром. Эти датчики позволяют определять изменения в ориентации мобильного устройства. Это еще одна замечательная особенность мобильных устройств.

Unreal Engine 4 позволяет легко использовать эти датчики с помощью раздела **Inputs** панели **Project Settings**. Добавляя параметр **Tilt** (Наклон) к существующему маппингу оси, вы можете добавить качественно новый способ управления входом.

На рис. 24.14 показано добавления опции **Tilt** к маппингу оси **MoveRight**. В этом случае нам нужно инвертировать значение, которое поступает из оси **Tilt**, и уменьшить его масштаб, чтобы движение было легче контролировать.

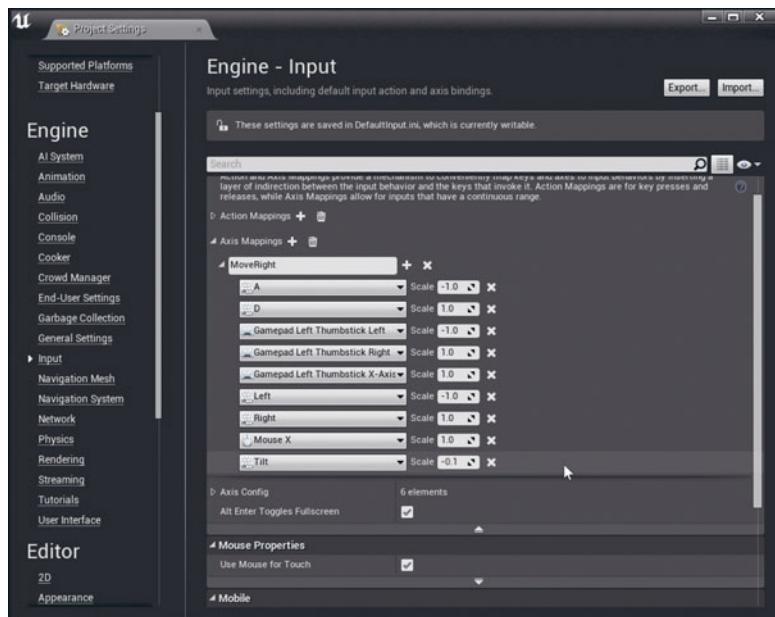


Рис. 24.14. Маппинг оси **Tilt**, добавленный к существующему отображению оси **MoveRight**

Использование осевых входов может быть очень удобным, но иногда требуется получить значения от гироскопа и акселерометра для других целей. Как и в случае с нодом события **InputTouch**, вы можете получить доступ к текущим показаниям гироскопа непосредственно через граф событий.

В отличие от события InputTouch, данные о движении устройства недоступны в виде события. Вместо этого эти данные могут быть доступны из нода Player Controller в виде функции Get Input Motion State. Эти ноды показаны на рис. 24.15.

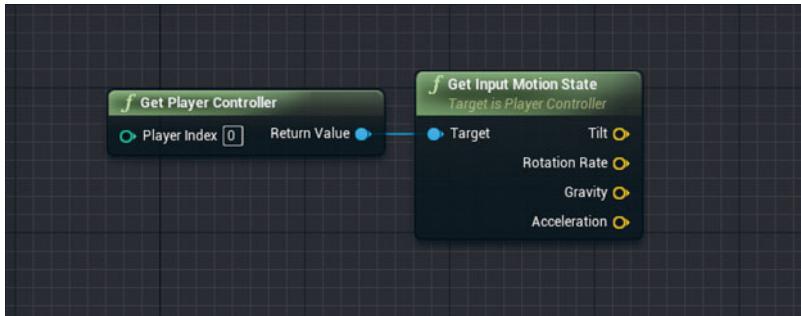


Рис. 24.15. Нод Get Input Motion State, соединенный с нодом Player Controller

Нод Get Input Motion State позволяет получить четыре состояния движения.

- ▶ **Tilt** (Наклон). Вращение устройства относительно осей X и Z.
- ▶ **Rotation Rate** (Скорость вращения). Скорость или изменение координаты вращения в секунду по каждой оси.
- ▶ **Gravity** (Сила тяжести). Ненормированный вектор, направленный в сторону земли, с точки зрения Player Controller A.
- ▶ **Acceleration** (Ускорение). Изменение скорости вращения прибора в секунду по каждой оси. Например, вращающееся с постоянной скоростью устройство будет иметь нулевое ускорение, но ненулевую скорость вращения.

Использование маппинга входов полезно, но не может дать вам полный контроль. Проблема тут в том, что у маппинга входов нет мертвых зон, то есть если мобильное устройство не идеально откалибровано и вы управляете движением вашего персонажа через наклон устройства, он будет двигаться всегда, даже когда устройство просто лежит на горизонтальном столе. Вместо этого с помощью нодов Event Tick и Get Input Motion State вы можете создать собственную схему управления кораблем на основе гравитации.

ПОПРОБУЙТЕ САМИ

Использовании гравитации вашего устройства

Мы используем ноды Event Tick и Get Input Motion State для аватара, чтобы создать схему управления, которая работает, когда игрок наклоняет устройство влево или вправо. Потребуется оснащенное акселерометром мобильное устройство, на котором можно будет проверить работу игры. Выполните следующие действия, чтобы подключить вектор силы тяжести к управлению движением вашего устройства.

1. На панели **Content Browser** проекта Hour23 откройте блупринт **Hero_Spaceship** в папке *Blueprints*.
2. Добавьте новый нод **Event Tick** (или используйте существующий нод Event Tick).
3. Рядом с нодом Event Tick поместите новый нод **Get Actor Right Vector** и нод **Get Player Controller**.
4. Перетащите контакт вывода Return Value нода Get Player Controller и создайте новый нод **Get Input Motion State**.
5. Перетащите контакт вывода Gravity и создайте новый нод Normalize.
6. Создайте новый нод **Vector*Vector** и соедините контакты вывода Return Value нодов Get Input Motion State и Normalize.
7. Создайте новую переменную типа Vector и назовите ее **Internal Gravity Vector**.
8. Установите нод Internal Gravity Vector как результат произведения Vector*Vector и соедините его с контактом вывода нода Event Tick.
9. Создайте новый нод **Branch** и соедините его с контактом вывода нода Internal Gravity Vector's Set.
10. У нода Internal Gravity Vector's Set потяните желтый вектор и создайте новый нод **VectorLength**.
11. Перетащите контакт вывода Return Value нода VectorLength и создайте нод **Float ⇒ Float**. Присвойте переменной **B** значение **0.1**.
12. Соедините контакт вывода нода **Float ⇒ Float** с контактом ввода Condition нода Branch.
13. После контакта вывода нода Branch создайте новый нод **Add Movement Input**.
14. Поместите **Internal Gravity Vector get** на контакт ввода World Direction нода Add Movement Input.
15. Чтобы увеличить ускорение космического корабля при наклоне устройства, установите значение **2.0** для контакта ввода Scale Value нода Add Movement Input. На рис. 24.16 показан готовый граф блупринта Hero_Spaceship.
16. Установите готовое приложение на ваше мобильное устройство, чтобы проверить работу игры. Вы можете сравнить свои результаты с готовым проектом Hour 24.

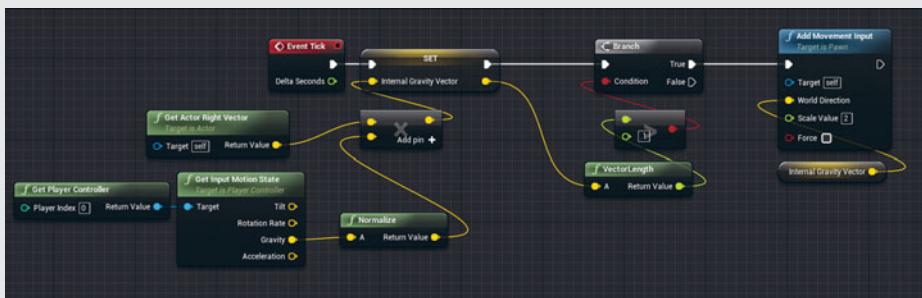


Рис. 24.16. Граф событий, используемый для контроля движения корабля через наклон мобильного устройства

Математика, используемая в этом графе событий, предельно проста. Контакт вывода Get Input Motion State Gravity возвращает ненормированный вектор направления реальной гравитации в мировом пространстве. Вы хотите, чтобы аватар двигался по направлению к земле, где бы она ни находилась. Поскольку двигаться он должен только вправо или влево, мы умножаем вектор силы тяжести на правый вектор объекта Pawn. В результате на него действует только та часть гравитации, которая соответствует правой оси.

Проверка длины результирующего вектора позволяет задать мертвую зону, и каждый раз, когда длина вектора превышает 0.1, в этом направлении добавляется узел Add Movement Input.

Резюме

Мобильная разработка — это быстрорастущая отрасль. Unreal Engine 4 может стать для вас отличным инструментом в этой области. В этом часе мы узнали о существующих аппаратных ограничениях, накладываемых на мобильные устройства; узнали, как подключить сенсор, виртуальные джойстики и гироскоп к управлению игрой. Эти типовые входы незаменимы для создания мобильной игры с сенсорными управлениями.

Вопросы и ответы

Вопрос: Я преобразовал мой предыдущий проект в мобильный, но некоторые из моих материалов при тестировании на мобильном устройстве теперь выглядят как шахматные доски. Что случилось?

Ответ: Возможно множество вариантов потенциальных проблем, но наиболее вероятно, что удаление сэмплов текстур привело к тому, что ваши материалы не скомпилировались. Откройте поврежденные материалы в редакторе и нажмите кнопку Mobile Stats на панели инструментов, чтобы вывести все ошибки компиляции материалов на мобильном устройстве.

Вопрос: Я использую контакт вывода Gravity нода Get Input Motion State, но мой аватар движется слишком быстро, и его становится трудно контролировать. В чем проблема?

Ответ: Вы, скорее всего, не нормировали значение, которое выходит из контакта вывода Gravity. Если вектор силы тяжести не нормируется, его длина может быть больше 1, что приводит к неконтролируемому ускорению.

Вопрос: У меня возникли проблемы с использованием входных привязок к управлению движением с помощью наклона. Как заставить все работать правильно?

Ответ: Каждый раз, когда вы сталкиваетесь с трудностями с привязкой входов к движениям, лучше воспользоваться способом, показанным на рис. 24.14. При перемещении входа в граф событий вы можете более аккуратно и точно настроить поведение так, как вам нужно. Вы можете использовать нод Print String, чтобы вывести выходные значения нода Get Input Motion State и разобраться, что не так с вашим устройством.

Вопрос: Я пытаюсь настроить мультисенсорный ввод с помощью события InputTouch на устройстве Windows, но он не работает. Что случилось?

Ответ: К несчастью, на данный момент UE4 не поддерживает мультитач на устройствах с ОС Windows, и решения для этой проблемы пока нет.

Семинар

Закончив этот час, попытайтесь ответить на следующие вопросы.

Контрольные вопросы

1. Истинно или ложно высказывание: мобильные устройства — это только телефоны.
2. Истинно или ложно высказывание: Unreal Engine 4 работает только на мобильных устройствах с iOS.

3. Истинно или ложно высказывание: вы можете обрабатывать сенсорные входы только непосредственно через граф событий, а не через **Input Binding** на панели **Project Settings**.
4. Истинно или ложно высказывание: свойство **Index Finger** нода событий **InputTouch** означает номер пальца пользователя, где 0 — это большой палец, а 4 — мизинец.

Ответы

1. Ложь. Планшеты и многие новые ноутбуки поддерживают сенсорный ввод, так что они тоже попадают под определение мобильных устройств вместе с телефонами. Однако если мы говорим об ограничениях графики, речь чаще идет о телефонах или слабых планшетах.
2. Ложь. Unreal Engine 4 поддерживает работу с устройствами под управлением операционной системы Android, iOS и Windows 10.
3. Истина. Сенсорные входы (касание или перетаскивание) не могут быть использованы через **Input Bindings**.
4. Ложь. Свойство **Index Finger** не определяет, какой палец используется. Оно определяет порядок работы нескольких пальцев. Первый палец, прикоснувшись к экрану, получает индекс 0, а второй палец, коснувшись экрана одновременно с первым, но позже, получит индекс 1.

Упражнение

Теперь воспользуемся вашими новыми знаниями о привязке действий и работе с мобильными устройствами, чтобы изменить график событий **Hero_Spaceship** и позволить пользователю стрелять непрерывным потоком снарядов, зажав клавишу выстрела, без необходимости многократно нажимать на нее.

1. Откройте график событий блупринт-класса **Hero_Spaceship**.
2. Найдите событие **InputAction Shoot** и переместите все его контакты вывода в новую функцию под названием **Shoot**.
3. Поместите новую функцию **Shoot** рядом с событием **InputAction Shoot** и соедините их контакты вывода.
4. Отключите все контакты вывода нода **Event Touch**.
5. Перетащите контакт вывода **Pressed** нода **Event Touch** и создайте нод **Set Timer by Function**.

6. Присвойте параметру **Function Name** значение **Shoot**.
7. Присвойте параметру **Time** значение **0.1**.
8. Присвойте параметру **Looping** значение **True**.
9. Перетащите контакт вывода **Return Value** нода **Set Timer by Function** и выберите опцию **Promote to Variable**.
10. Назовите новую переменную **ShootTimerHandle**.
11. Перетащите контакт вывода **Released** нода **Event Touch** и создайте ноду **Clear Timer by Handle**.
12. Перетащите переменную **ShootTimerHandle** и поместите ее на контакт ввода **Handle** нода **Clear Timer by Handle**.

Предметный указатель

- .fbx, файловое расширение, 216
- 3D
 - инструменты трансформации, 62
 - системы координат, 60
- ActivateStomper_BP, блюпринт, 389
- AIController, класс, 54, 55
- Android, упаковка, 475
- Animation, режим редактора Persona, 225
- Attenuation Radius, свойство, 107
- Audacity, приложение, 136
- Auto Activate, параметр, 207
- Blueprint Editor, редактор, 282, 328
- BP_Common, папка, 383
- BP_Lever, папка, 388
- BP_Pickup, папка, 387
- BP_Respawn, папка, 387
- BP_Turrets, папка, 386
- BSP-актеры, 173
- C++
 - проекты, 27
 - язык программирования, 281
- Camera, группа, 249
- Cascade, интерфейс, 192
- Cast Shadows, свойство, 107
- Checkpoint_BP, блюпринт, 387
- CollectionPickup_BP, блюпринт, 388
- Collision Enabled, параметр настройки, 95
- Color Over Life, модуль, 201
- Config, папки, 46
- Const Acceleration, модуль, 202
- Construction Script, скрипт конструирования, 354
- Content Browser, панель, 33
 - фильтры, 51
- Content Examples, пример проекта, загрузка, 44
- Content, папки, 46, 47
- Convert Scene Unit, свойство, 219
- Convex Decomposition, панель, 87
- Copy, инструмент, 156
- Create, кнопка, 154
- Curve Editor, редактор, 238, 244
- Data-Only блюпринт, 282
- Default Pawn, класс, 400
- Default Pawn, наследование, 400
- Details, панель, 32, 124, 193, 238, 284
- Door_BP, блюпринт, 389
- DPI масштабирование, 452
- Emitters, панель, 193
- Erosion, инструмент, 156
- Event Graph, панель, 284, 285
- Event Tick, событие, 297, 344
- Exес-контакты, 287

- Falloff, меню, 157
Fill World, кнопка, 154
Flatten, инструмент, 156
FPS, frames per second, 237
FPS, шутер от первого лица, 379
- GameMode, класс, 54
Gameplay Framework, система, 53
GPU-спрайты, 191
Graph, панель, 124, 141
Graph, режим редактора Persona, 229
- Heal Damage, функция, 434
HealthPickup_BP, блюпринт, 388
HUD, класс, 55
HUD-интерфейсы, 380
Hydro Erosion, инструмент, 156
- IDE, интегрированная среда разработки, 281
Import Animations, свойство, 219
Import Materials, свойство, 219
Import Mesh, свойство, 219
Inherit Parent Velocity, модуль, 202
Initial Color, модуль, 201
Initial Location, модуль, 202
Initial Rotation, модуль, 203
Initial Size, модуль, 200
Initial Velocity, модуль, 202
Inside Cone Angle, свойство, 107
Intensity, свойство, 107
InterfaceAssets, папка, 453
Intermediate, папки, 46
- iOS, упаковка, 475
IsVariable, свойство, 449
KillVolume_BP, блюпринт, 387
- Landscape, кнопка, 151
Landscape, панель, 151
Launcher, установка, 24
Launcher_BP, блюпринт, 383
Level Editor, редактор навигации, 29
панель инструментов, 39
Lifetime, модуль, 200
Light Color, свойство, 107
Lightmass Importance Volume, параметр, 183
Lightmass, инструмент, 108
LOD (Level of detailing), уровень детализации, 152
- Manage, вкладка, 151
Material Editor, редактор, 117, 157
Matinee, редактор Sound, трэк, 248
актеры, 235
ассеты, применение, 253
группы, 240
интерполяция, 238
трэки, 241
Matinee, редактор, 238
Mesh, режим редактора Persona, 223
Modes, панель, 30
Modules, панель, 193

- Move, трансформация, 63
Mover_BP, блюпринт, 383
My Blueprint, панель, 284

Noise, инструмент, 156

Object Type, параметр настройки, 95
Obstacle, класс, 430
OnActorBeginOverlap, событие, 309
OnActorEndOverlap, событие, 309
OnActorHit, событие, 309
Outside Cone Angle, свойство, 107

Palette, панель, 125, 142
Paste, инструмент, 156
Pattern_Projectile_BP, блюпринт, 386
PatternTurret_BP, блюпринт, 386
Pawn, классы, 55
Pendulum_BP, блюпринт, 383
Persona Editor, редактор, 221
PhysicSpawner_BP, блюпринт, 389
PhysicsPickup_BP, блюпринт, 388
Play Cue, кнопка, 142
Play Node, кнопка, 142
Play Sound at Location, функция, 320
PlayerController, класс, 54, 55
Print String, функция, 290
ProjectileTurret_BP, блюпринт, 387

Ramp, инструмент, 156
Required, модуль, 198
Retopologize, инструмент, 156
Rotate, трансформация, 64
Rotation Rate, модуль, 203

Saved, папки, 46, 53
Scale Color/Life, модуль, 201
Scale, трансформация, 63
Sculpt, инструмент, 156
Selection, инструмент, 156
Selection, кнопка, 154
Show 3D Widget, свойство, 361
Size By Life, модуль, 200
Skeletal Mesh, свойство, 219
Skeleton, режим редактора Persona, 222
Skeleton, свойство, 219
Smasher_BP, блюпринт, 384
Smooth, инструмент, 156
Sound Cue Editor, редактор, 136
Sound, трэк, 248
Spawn Actor from Class, функция, 372
Spawn, модуль, 199
Sphere, модуль, 202
SpikeTrap_BP, блюпринт, 384
Start Awake, свойство, 261
Static Mesh Editor, редактор, 78
Static Mesh, компонент, 312
Stomper_BP, блюпринт, 384
SubUV-текстуры, 204
Swarm Agent, приложение, 108

Temperature, свойство, 107
Timeline, нод, 336
Tool, меню, 156
TouchActivation_BP, блюпринт, 389
TraceTurret_BP, блюпринт, 387
Tracks, панель, 238
Transform, свойство, 219

TurretProjectile_BP, блюпринт, 387
UMG UI Designer
импорт ассетов, 453
масштабирование DPI, 452
навигация, 447
настройка разрешения, 449
режим Graph, 448
режим конструктора, 447
скрипты, 458
стартовое меню, 453
якорные точки, 452
UseKeyLever_BP, блюпринт, 388
UV-развертки, 78

Vehicle, классы, 55
Viewport, панель, 34, 329
Visibility, инструмент, 156

Windows ОС, требования, 24
World Outliner, панель, 31
World Settings, панель, 258

Автоматически генерируемые оболочки коллизий, 87
Актеры
 Camera, группа, 249
 Director, группа, 251
 Matinee, 235
 актеры звуков окружения, 136, 138
 актеры ИС, 100
 видимость, рендеринг, 307
 выбор, 72
группировка, 71
захвата отражений, 184
звук, применение, 138
инструменты трансформации, 62
классы, спаунинг, 371
коллизии, настройки, 308
компоненты, 312
материалы
 изменение, 315
 перезапуск свойств, 316
объединение, 178
объемов постобработки, 184
перемещение, 70
прикрепление, 73
свойства, 313
скелетных мешей, 211
импорт, 216
определение, 211
применение, 230
редактор Persona, 221
слои, применение, 72
события, 322
 активация, 318
 назначение, 309
ссылочные переменные, назначение, 312
статичный меш, 30, 89
статичных мешей
 анимация, 243
 свойства, 262
тумана, 169, 184
физики
 ограничения, 270
прикрепление, 269

- радиальной силы, 275
толкатели физики, 274
- Актеры ИС, 100
- Актеры радиальной силы, 275
- Актеры скелетных мешей
- Persona, редактор, 221
 - импорт, 216
 - определение, 211
 - применение, 221
- Актеры статичных мешей, 30, 89, 262
- анимация, 243
 - изменение ссылки на меш, 91
 - копирование, 91
 - назначение физического материала, 266
 - настройки мобильности, 91
 - помещение на уровень, 90
 - редактирование коллизий, 93
 - свойства, 263
- Актеры толкателей физики, 274
- Активация
- частиц, 207
- Альбедо, 117
- Анимация
- актеры статичных мешей, 243
 - блюпринты, 215
 - последовательности, 215
 - редактирование, 238
- Аркадные шутеры, 393
- Default Pawn, наследование, 400
 - игровые режимы, 397
 - контроллеры, 399
 - маппинг осей, 406
- павны
- получение урона, 423
- препятствия
- создание, 416
 - удаление, 442
- создание, 394
 - спаун актеров, 436
 - фиксированная камера, 410
- Ассеты
- Matinee, применение, 253
 - затухание звука, 136
 - звуковые волны, 136
 - звуковые сигналы, 136
 - иконки, 50
 - импорт, 453
 - модульные, применение, 177
 - помещение, 176
 - связи, 53
 - статичный меш, 30
 - типы ассетов, 51, 54, 55
 - физики, 215
 - физических материалов, 264
- Ассеты статичных мешей, 30, 77
- импорт, 80
 - присвоение материала, 83
 - просмотр, 80
- Аудио
- ассеты звуковых сигналов, 141
 - затухание, настройка, 139
 - звуковые актеры, применение, 138
 - импорт, 136
 - компоненты, 136
 - обзор, 135

объемы, применение, 145
свойства модуляции, применение, 141
создание мира, 183
трэк Sound, 248

Базовый цвет, 117

Байткод, 282

Блюпринт, 27

- анимации, 215
- проекты, форматирование, 46
- скрипты
 - Event Graph, панель, 285
 - My Blueprint, панель, 285
 - блоки комментирования, 300
 - комментирование нодов, 299
 - компиляция, 281
 - компоненты, 289
 - контекстное меню блюпринта, 286
 - ноды перенаправления, 301
 - обзор, 281
 - операторы, 297
 - переменные, 293
 - понятия, 288
 - события, 288
 - структуры, 294
 - типы, 282
 - управление, 299
 - условия, 297
 - функции, 290
 - уровня, 306
 - частицы, активация, 207
- Блюпринт уровня, 282

Блюпринт-интерфейс, 282

Блюпринт-класс, 282, 326

- Timeline, нод, 336
- актеры, объединение, 177
- добавление, 327
- извлечение из существующих классов, 340
- компоненты, 330
- применение, 326
- пульсирующий ИС, создание, 340
- скриптинг, 333

Блюпринт-макрос, 282

Блюпринты уровней

- Play Sound at Location, функция, 320
- актеры
 - активация событий, 318
 - компоненты, 312
 - назначение событий, 309
 - назначение ссылочным переменным, 312
 - настройки коллизий, 308
 - свойства, 313
 - свойства активации, 310
 - цели функций, 314
- Браузер проектов, навигация, 27

Ввод

- материалы, 125
- сенсор
 - использование, 493
 - события, 496
 - типы, 117
- Векторные распределения, 195

- Вершины, 77
скайнинг, 213
- Виджет-блюпринт, создание, 446
- Виджеты
Show 3D Widget, 361
- Видимость
рендеринг актера, 307
- Визуализаторы
каналов, 197
свойств, 197
- Визуальный скрипting, блюпринты, 281
- Виртуальная машина, 282
- Виртуальные джойстики, 493
- Восстановление, термин физики, 258
- Временные метки, 239
- Выбор актеров, 72
- Вывод
материалы, 125
- Границы, 77
- Граф событий, 285
- Группы, 71
Camera, 249
Director, 251
Matinee, 241
- Данные анимированных следов, 191
- Данные лучей, 191
- Данные мешей, 191
- Данные осколков, 191
- Декартова система координат, 60
- Декорации
- помещение, 176
- Джойстики виртуальные, 493
- Динамическое освещение, 100
- Добавление
GameMode, класс, 54
актеров, 236
блюпринт-классов, 327
виджет-блюпринта, 446
камер, 250
компонентов, 330
компонентов статичных мешей, 318
кривых, 337
множественные оболочки коллизий, 84
направленный ИС, 105
небесный ИС, 104
освещения, 182
 прожекторный ИС, 102
событий, 290
точечный ИС, 101
трэков, 337
- Жесткий диск, требования, 25
- Загрузка
Launcher, 24
Unreal Engine, 25
примера проекта, 44
- Запаковка проекта, 470
- Запуск
частиц, 207
- Затухание, настройка, 139
- Зациклывание, 144

Звук, 135, 138
Зеленые каналы, 120
Игровое тестирование, 183
Игры
аркадный шутер, 393
игровые режимы проекта, 379
таймеры, 380
Извлечение классов, 328
Изменение
ассетов статичных мешей, 78
интерфейсов, 30
ландшафтов, 152
настроек проекта, 29
связей мешей, 91
Импорт
ассетов, 453
ассетов статичных мешей, 80
аудио, 136
контента, 48
текстур, 122
Импульс, термин физики, 258
Инструменты
Copy, 156
Erosion, 156
Flatten, 156
Hydro Erosion, 156
Lightmass, 108
Noise, 156
Paste, 156
Ramp, 156
Retopologize, 156
Sculpt, 156
Selection, 156
Smooth, 156
Visibility, 156
World Outliner, применение, 68
визуализация, 38
ландшафты, 151
настройки привязки, 68
трансформации, 62
Move, 63
Rotate, 64
Scale, 63
Интегрированная среда разработки (IDE), 281
Интерполяция, 242
Интерфейсы
Blueprint Editor, 283, 328
Content Browser, панель, 33
Details, панель, 32
Modes, панель, 30
Viewport, панель, 34
World Outliner, панель, 31
изменение, 30
навигация, 29
строка меню, 30
Использование
эмиттеров частиц, 194
Каналы
визуализаторы, 197
зеленый, 120
Карты высот, 151
Карты освещения, 78
UV-каналы, 82
Кастомизация
игровых режимов, 399

- контроллеров, 400
Классы
 AIController, 54
 Blueprint, 282
 GameMode, 54
 HUD, 55
 Obstacle, 430
 Pawn, 55
 PlayerController, 54
 Vehicle, 55
актеры, спаунинг, 371
контроллеры, 54
Классы препятствий, 416
Ключ, 196
Ключевые кадры, 242
Кнопки
 Create, 154
 Fill World, 154
 Play Cue, 142
 Play Node, 142
 Selection, 154
Количество кадров в секунду (FPS), 237
Коллизии
 Collision Responses, флагги, 95
актеры статичных мешей, редактирование, 93
актеры, настройка, 308
многополигональные, 88
оболочки, 77, 83
 Convex Decomposition, панель, 87
автоматически генерируемые, 84
пресеты, 93
Комментирование
 блоки, 300
 нодов, 299
 скрипты, 299
Компилятор, 281
Компиляция, 282
Компоненты
 актеров, 312
 аудио, 136
 блюпринт-класс, 327
 скриптинг, 289, 333, 340
 теги, 389
Контакты выполнения, 287
Контакты данных, 288
Контекст, единицы измерения, 65
Контекстное меню блюпринта, 286
Контент
 импорт, 48
 пакеты контента, 168
 перенос, 50
Контент браузер, навигация, 33
Контроллеры
 классы, 54
Контроль массы, 274
Копирование, 33
актеров статичных мешей, 91
Корневой виджет по умолчанию, 448
Кривые, добавление, 337
Ландшафты
 Manage, вкладка, 151
 инструменты, 151, 156, 157
 карты высот, 151

- материалы, 157
- объемы, 155
- окрашивание, 157
- применение, 150
- создание, 152
- управление, 154
- формы, 155
- Линейный, термин физики, 258
- Локальная ось, 77
- Локальные трансформации, 65
- Маппинг осей, 406
- Масса, контроль, 274
- Масса, термин физики, 258
- Масштаб, 170
- Масштабирование DPI, 452
- Материалы, 77, 115
 - актеры, 266
 - актеры статичных мешей, 92
 - ассеты статичных мешей, 83
 - базовый цвет, 117
 - ввод, 117, 125
 - вывод, 125
 - зеленый канал, 120
 - ландшафтов, 157
 - металлизированность, 117
 - ноды значений, 126
 - нормаль, 118
 - создание, 117, 123
 - частиц, настройка, 203
 - шероховатость, 118
 - экземпляры, 128
- Меню
- Brush, 156
- Falloff, 157
- Tool, 156
- контекстное меню блупримта, 286
- системы, 464
- Металлизированность, 117
- Метки, временные, 239
- Меши
 - скелетные, 211
 - скиннинг, 213
- Мировые трансформации, 65
- Многополигональные коллизии, 88
- Мобильность, 110
- Мобильность, актеры статичных мешей, 91
- Мобильные устройства
 - введение, 481
 - виртуальные джойстики, 493
 - данные о движении, 499
 - использование сенсора, 493
 - оптимизация, 485
 - разработка для, 480
 - тестирование, 481
 - цели редактора, 490
- Модели, просмотр UV-разверток, 81
- Модули
 - Color Over Life, 201
 - Const Acceleration, 202
 - Inherit Parent Velocity, 202
 - Initial Color, 201
 - Initial Size, 200
 - Initial Velocity, 202
 - Lifetime, 200
 - Required, 198

- Rotation Rate, 203
Scale Color/Life, 201
Size By Life, 200
Spawn, 199
частиц, 193
 необходимые, 194
 основные, 198
 редактор кривых, 196
 свойства, 194
Модуляция, свойства, 141
Мягкое тело, термин физики, 258
- Навигация
 Landscape, панель, 151
 Matinee, редактор, 238
 Project Browser, панель, 27
 Static Mesh Editor, 78
 интерфейсы, 29
 панель инструментов редактора
 уровней, 39
 сцена, 38
Навигация в игровом стиле, 39
Назначение
 актеров
 событиям, 309
 ссылочным переменным, 312
 существующим группам, 240
 физики уровням, 258
Направленный ИС, 105
Настройка
 целей редактора, 490
Настройка
 Collision Enabled, параметр настройки, 95
Object Type, параметр настройки, 95
Timeline, нод, 337
затухания, 139
ключевые кадры, 242
коллизий актеров, 308
мобильности, актеры статичных мешей, 91
переменных, 342
продолжительность последовательности, 239
проекты, изменение, 29
режим игры по умолчанию, 57
спаунинг (блюпринт-класс), 369
частиц, материалов, 203
Небесный ИС, 104
Непрерывность, единицы измерения, 65
Непрямое освещение, 100
Ноды, 287
 комментирование, 299
 перенаправления, 301
Ноды перенаправления, 301
Нормаль, 118
- Оболочки коллизий, 83
Обучиться, раздел, 43, 44
Объявление переменных, 296
Ограничения
 актеров физики, 269
 применение, 269
Одежда, термин физики, 258
Окрашивание ландшафтов, 157
Операторы, блюпринт-скрипты, 297

Операционные системы, требования, 24

Ортогональная проекция, вьюпорт, 35

Освещение

- Swarm Agent, 108
- динамическое, 100
- добавление, 182
- мобильность, 110
- направленные ИС, добавление, 105
- небесные ИС, добавление, 104
- непрямое, 100
- отраженное, 100
- прожекторные ИС, добавление, 102
- просчет, 108
- прямое, 99
- пульсирующие ИС, форматирование, 340
- свойства, 107
- создание мира, 181
- статичное, 100
- тени, 100
- терминология, 99
- типы, 100
- точечные ИС, добавление, 101

Основные модули частиц, 198

Отраженное освещение, 100

Панели

- Content Browser, 33, 51
- Convex Decomposition, 87
- Details, 32, 124, 193, 238
- Emitters, 193
- Event Graph, 285
- Graph, 124, 125, 141
- Modes, 30
- Modules, 193
- My Blueprint, 285
- Palette, 125, 142
- Tracks, 238
- Viewport, 34, 329
- World Outliner, 31, 68
- World Settings, 258

Панели инструментов

- редактор уровней, 29

Папки

- BP_Common, 383
- BP_Levers, 388
- BP_Pickup, 387
- BP_Respawn, 387
- BP_Turrets, 386
- Config, 46
- Content, 46, 47
- InterfaceAssets, 453
- Intermediate, 46
- Saved, 46
- Saved, 53

Павны

- контроллеры, 399
- наследование, 400
- отключение, 405

события ввода, 408

World Outliner, панель, 69

для необработанных ассетов, создание, 52

создание, 49, 70

- Передвижение
павны, 403
отключение, 405
- Передвижение персонажей, 383
- Переменные
блюпринт-скрипты, 293
настройка, 342
объявление, 296
открытие, 368
списки, 295
ссылочные, 312
форматирование, 351
- Перемещение
актеров, 70
ассетов статичных мешей, 80
текстур, 122
- Перенаправление, ноды, 301
- Перетаскивание текстур, 123
- Персонажи
актеры скелетных мешей, 216
передвижение, 383
умения, 380
- Перспективная проекция, вьюпорт, 35
- Плавающие распределения, 195
- Плотность, термин физики, 258
- Повествование через окружение, 168
- Подвижное освещение, 111
- Полигоны, 77
- Пользовательские настройки
инструменты привязки, 68
пресеты коллизий, 93
- Пользовательские функции, 292
- Помещение
актеров звуков окружения, 139
актеров скелетных мешей, 230
ассетов, 176
виджетов павнов, 455
декораций, 176
- Помещение актеров статичных мешей на уровень, 90
- Последовательности
анимации, 215
продолжительность, 239
- Последовательность анимации, 215
- Препятствия
аркадные шутеры, 415
удаление старых препятствий, 442
- Пресеты коллизий, 93
- Привязка, 383
- Приготовление контента, 469
- Прикрепление актеров, 73
- Применение
Static Mesh Editor (редактор статичных мешей), 78
SubUV-текстур, 205
World Outliner, панель, 68
- актеров скелетных мешей, 230
ассетов Matinee, 253
аудио объемы, 145
блюпринт-классов, 326
звуковые актеры, 136
ландшафты, 150
материалы, 115
модульных ассетов, 177
нода Timeline, 336
оболочки коллизий, 86

- ограничений, 269
- растительность, 161
- редактора Matinee, 238
- редактора кривых, 196
- свойства модуляции, 141
- скрипта конструирования, 366
- слои, 72
- трэка Sound, 248
- физики, 257
- Присвоение
 - материалов ассетам статичных мешей, 83
- Провода, 288
- Продвинутые настройки запаковки, 476
- Продолжительность последовательности, настройка, 239
- Проекты
 - C++, 27
 - Content Examples, 44
 - GameMode, класс, 54
 - блюпринты, 46
 - изменение настроек, 29
 - папки, 46
 - перенос контента, 50
 - пустые, форматирование, 46
 - режимы, 379
 - создание, 27
- Прожекторный ИС, 102
- Просмотр
 - кривых, 245
- Просчет
 - освещения, 108
- Прямое освещение, 99
- Пульсирующий ИС, создание, 340
- Пустой проект, 46
- Размеры текстур, 120
- Разрушаемый объект, термин физики, 258
- Рассеивание, 117
- Растительность
 - помещение, 162
 - применение, 161
- Редактирование
 - анимации, 238
 - коллизии
 - актеры статичных мешей, 93
 - ландшафты, 150
 - оболочки коллизий, 84
- Редактируемые переменные
 - ограничение, 360
 - свойство Show 3D Widget, 361
 - форматирование, 351
- Редактор блюпринтов, 282, 328
- Редактор кривых, 238, 244
- Редактор уровней, навигация, 29
- Редакторы
 - Audacity, 136
 - Blueprint Editor, 282, 328
 - Curve Editor, 196, 244
 - Level Editor, 29
 - Material Editor, 117, 157
 - Persona Editor, 221
 - PIE (Play in Editor), 45
 - Sound Cue Editor, 136
 - Static Mesh Editor, 78
 - режимы, 31

- цели, настройка, 490
частицы, интерфейс Cascade, 192
Режим конструктора (UMG UI Designer), 447
Режимы
 добавление класса GameMode, 55
 игры, назначение, 258
 интерполяция, 238
 проект, 379
 просмотра, 37
 редактора, 31
 режим конструктора, 447
Режимы игры, назначение, 258
Рендеринг
 видимость актера, 307
Ресурсы, 43
Свойства
 актеров, 313
 визуализаторов, 197
 звуковых волн, 138
 модулей частиц, 194
 модуля Required, 198
 модуля Spawn, 199
 модуляции, 141
 освещения, 107
Связи
 ассетов, 52
 мешей, 91
Сенсор
 использование, 493
 события, 496
Сетка
 единицы измерения, 66
 придание формы, 67
 Сетка масштабирования, 68
 Сетка перетаскивания, 67
 Сетка поворота, 68
 Сила, термин физики, 258
 Симуляция физики, 261
 Система координат, Декартова, 60
 Системные требования, 24
 Системы респауна, 380
 Скелет, 214
 Скелетный меш, 214
 Скиннинг, 213
 Скрипт конструирования
 использование, 354
 Скрипт конструирования, применение, 366
 Скрипты
 блюпринт
 Event Graph, панель, 285
 My Blueprint, панель, 285
 блоки комментирования, 300
 интерфейс редактора
 блюпринтов, 283
 комментирование, 299
 компиляция, 282
 компоненты, 289, 330
 контекстное меню блюпринта, 286
 обзор, 281
 операторы, 297
 переменные, 293
 перенаправление, 301
 понятия, 288
 события, 288
 структуры, 294

- типы, 282
- управление, 299
- условия, 297
- функции, 290
- Скульптурирование
 - объемов, 155
 - форм, 155
- Слои
 - применение, 72
 - текстуры, соединение, 159
- Смешивание звуковых сигналов, 144
- События
 - OnActorBeginOverlap, 309
 - OnActorEndOverlap, 309
 - OnActorHit, 309
 - ввод сенсора, 496
 - ввода, настройка, 408
 - добавление, 293
 - назначение актеров, 309
 - скрипты, 288
- События ввода, 408
- Соединение текстур со слоями, 159
- Создание мира, 168
 - Lightmass Importance Volume, параметр, 183
 - актеры захвата отражений, 184
 - актеры объемов постобработки, 184
 - актеры тумана, 184
 - актеры, объединение, 177
 - аудио, 183
 - визуальная сложность, 176
 - мир в отдалении, создание, 179
- модульные ассеты, применение, 177
- освещение, 181
- повествование через окружение, 168
- помещение ассетов, 176
- помещение декораций, 176
- создание масштаба, 170
- создание наброска, 174
- создание пределов, 171
- создание слоев, 174
- уровни, форматирование, 169
- фрейминг, 176
- цвета теней, 182
- Создание наброска мира, 174
- Сокеты, 78
- Состояние смерти, создание, 427
- Спаунинг, 373
 - актера из класса, 371
 - блюпринт-класс, настройка, 365
- Сплайн, 244
- Сплайны, 155
- Спрайты, 191
- Ссылочные переменные, назначение, 312
- Старые препятствия, удаление, 442
- Статичное освещение, 100, 110
- Стационарное освещение, 111
- Стока меню, 30
- Структура, блюпринт-скрипты, 294
- Существующие проекты, перенос контента, 50
- Сцены
 - навигация, 38

- организация, 68
- Твердое тело, термин физики, 258
- Теги, 389
- актеров, 389
 - компонентов, 389
- Текстуры, 78
- импорт, 122
 - размеры, 120
 - расширения файлов, 121
 - слои, соединение, 159
 - форматирование, 120
- Тени, 100
- Тестирование
- игровое, 183
- Типы
- ассетов, 51
 - входных данных материалов, 117
 - данных, обзор, 191
 - источников света, 100
 - окон предпросмотра, 35
 - сеток, 67
 - скриптов, 282
 - трансформаций, 62
 - файлов, 48
 - файлов текстур, 121
- Типы данных, обзор, 191
- Толкатели физики, 274
- Торможение, термин физики, 258
- Точечный ИС, 101
- Трансформации
- виды, 64
 - инструменты, 62
 - масштабирования, 63
- перемещения, 63
- поворота, 64
- Требования
- системные, 24
- Требования к разработке аркадных шутеров, 394
- Трение, термин физики, 258
- Трэки
- Director, группа, 251
 - Matinee, редактор, 241
 - Sound, трэк, 248
 - добавление, 337
- Угловой, термин физики, 258
- Удаление старых препятствий, 442
- Указатель воспроизведения, 240
- Умения персонажей, 380
- Упаковка
- Android, 475
 - iOS, 475
 - проекта для Windows, 470
- Управление
- блюпринт-скриптами, 299
 - ландшафтами, 154
- Уровень детализации, LOD (Level of detailing), 152
- Уровни
- Level Editor, редактор, 29
 - блюпринты, активация частиц, 207
 - назначение физики, 258
 - по умолчанию, форматирование, 169
 - помещение актеров статичных мешей, 90

проигрывание, 40
просмотр, 45
создание мира, форматирование, 168
Уровни по умолчанию
форматирование, 169
Условия, блюпринт-скрипты, 297
Установка
Launcher, 24
Unreal Engine, 25
Установочная поза, 220
Файлы
.fbx, 216
аудио, импорт, 136
текстур, 121
типы, 48
Физика, 257
World Settings, панель, 258
актеры
ограничения, 270
прикрепление, 269
радиальной силы, 275
толкатели, 274
ассеты, 215
назначение уровню, 258
ограничения, 269
применение, 257
свойства для актеров статичных мешей, 263
симуляция, 261
терминология, 258
физические материалы, 264
Физические материалы, 264
Физическое тело, термин физики, 258
Фильтры, Контент браузер, 51
Флажки, Collision Responses, 95
Форматирование
аркадные шутеры, 415
ассетов звуковых сигналов, 141
блюпринт-класс, спаунинг, 365
виджет-блюпринтов, 446
ландшафтов, 152
материалов, 117, 123
мира в отдалении, 179
папок, 69
для необработанных ассетов, 52
создание, 49
папок Content, 47
проектов, 27
папки, 46
пустые проекты, 46
редактируемых переменных, 351
состояние смерти, 427
текстур, 120
уровней по умолчанию, 169
уровней, создание мира, 168
цвета частиц, 204
экземпляров материалов, 128
Формы, 155
Фрейминг, 176
Функции
Play Sound at Location, 320
Print String, 293
Spawn Actor from Class, 372
блюпринт-скрипты, 290
цели, 314

Цвета	Шероховатость, 118
базовый цвет, 117	
теней, 182	Экземпляры, 33
частиц, форматирование, 204	материалов, 128
Цели	Экшен-столкновения, 379
редакторов, настройка, 490	BP_Common, папка, 383
Цели функций, 314	BP_Levers, папка, 388
Частицы	BP_Pickup, папка, 387
Auto Activate, параметр, 207	BP_Turrets, папка, 386
Cascade, интерфейс, 192	HUD-интерфейсы, 380
SubUV-текстуры, 204	блюпринт-классы, 382
активация, 207	игровой таймер, 380
запуск, 207	игровые режимы проекта, 379
материалы, настройка, 203	система респауна, 380
модули	теги актеров и компонентов, 389
необходимые, 194	умения персонажей, 380
основные, 198	
редактор кривых, 196	Элементы управления
свойства, 194	ландшафтами, 152
обзор, 190	редактор кривых, 196
цвета, форматирование, 204	Эмиттеры, частицы, 191
эмиттеры, 193	Эффекты
Шейдеры, 115	SubUV-текстуры, 204
	Эффекты реверберации, 145

Все права защищены. Книга или любая ее часть не может быть скопирована, воспроизведена в электронной или механической форме, в виде фотокопии, записи в память ЭВМ, репродукции или каким-либо иным способом, а также использована в любой информационной системе без получения разрешения от издателя. Копирование, воспроизведение иное использование книги или ее части без согласия издателя является незаконным и влечет уголовную, административную и гражданскую ответственность.

Научно-популярное издание

МИРОВОЙ КОМПЬЮТЕРНЫЙ БЕСТSELLER. ГЕЙМДИЗАЙН

Арам Куксон, Райан Даулингсока, Клинтон Крамплер

РАЗРАБОТКА ИГР НА UNREAL ENGINE 4 ЗА 24 ЧАСА

Главный редактор *P. Фасхутдинов*

Ответственный редактор *B. Обручев*

Научные редакторы *H. Веселко, O. Максименкова, A. Незнанов*

Младший редактор *D. Атакишиева*

Художественный редактор *E. Пуговкина*

Компьютерная верстка *Э. Брегис*

Корректор *P. Болдинова*

ООО «Издательство «Эксмо»

123308, Москва, ул. Зорге, д. 1. Тел.: 8 (495) 411-68-86.

Home page: www.eksмо.ru E-mail: info@eksмо.ru

Өндүруші: «ЭКСМО» АҚБ Баспасы, 123308, Мәскеу, Ресей, Зорге көшесі, 1 үй.

Тел.: 8 (495) 411-68-86.

Home page: www.eksмо.ru E-mail: info@eksмо.ru

Тауар белгісі: «Эксмо»

Интернет-магазин : www.book24.ru

Интернет-магазин : www.book24.kz

Интернет-дүкен : www.book24.kz

Импортёр в Республику Казахстан ТОО «РДЦ-Алматы».

Казакстан Республикасында импорташы «РДЦ-Алматы» ЖШС.

Дистрибутор и представитель по приему претензий на продукцию,

в Республике Казахстан: ТОО «РДЦ-Алматы»

Қазақстан Республикасында дистрибутор және енім бойынша арыз-талаптарды

қабылдаушының екілі «РДЦ-Алматы» ЖШС,

Алматы қ., Домбровский көш., 3 «а», литер Б, офис 1.

Тел.: 8 (727) 251-59-90/91/92; E-mail: RDC-Almaty@eksмо.kz

Өнімнің жаралымың мерзімі шектелмеген.

Сертификация туралы акпарат сайты: www.eksмо.ru/certification

Сведения о подтверждении соответствия издания согласно законодательству РФ

о техническом регулировании можно получить на сайте Издательства «Эксмо»

www.eksмо.ru/certification

Өндірген мемлекет: Ресей. Сертификация қарастырылмаған

Подписано в печать 11.09.2019. Формат 70x100¹/16.

Печать офсетная. Усл. печ. л. 42,78.

Тираж экз. Заказ

12+

ISBN 978-5-04-103162-6



9 785041 031626 >

В электронном виде книги издательства вы можете
купить на www.litres.ru

ЛитРес:
один клик до книги



Москва. ООО «Торговый Дом «Эксмо»

Адрес: 123308, г. Москва, ул. Зорге, д.1.

Телефон: +7 (495) 411-50-74. E-mail: reception@eksмо-sale.ru

По вопросам приобретения книг «Эксмо» зарубежными оптовыми покупателями обращаться в отдел зарубежных продаж ТД «Эксмо»

E-mail: international@eksмо-sale.ru

International Sales: International wholesale customers should contact Foreign Sales Department of Trading House «Eksmo» for their orders.
international@eksмо-sale.ru

По вопросам заказа книг корпоративным клиентам, в том числе в специальном оформлении, обращаться по тел.: +7 (495) 411-68-59, доб. 2261.

E-mail: ivanova.ey@eksмо.ru

Оптовая торговля бумажно-беловыми

и канцелярскими товарами для школы и офиса «Канц-Эксмо»:

Компания «Канц-Эксмо»: 142702, Московская обл., Ленинский р-н, г. Видное-2,
Белокаменное ш., д. 1, а/я 5. Тел./факс: +7 (495) 745-28-87 (многоканальный).
e-mail: kanc@eksмо-sale.ru, сайт: www.kanc-eksмо.ru

Филиал «Торгового Дома «Эксмо» в Нижнем Новгороде

Адрес: 603094, г. Нижний Новгород, улица Карпинского, д. 29, бизнес-парк «Грин Плаза»
Телефон: +7 (831) 216-15-91 (92, 93, 94). E-mail: reception@eksmonn.ru

Филиал ООО «Издательство «Эксмо» в г. Санкт-Петербурге

Адрес: 192029, г. Санкт-Петербург, пр. Обуховской обороны, д. 84, лит. «Е»
Телефон: +7 (812) 365-46-03 / 04. E-mail: server@szko.ru

Филиал ООО «Издательство «Эксмо» в г. Екатеринбурге

Адрес: 620024, г. Екатеринбург, ул. Новинская, д. 2щ
Телефон: +7 (343) 272-72-01 (02/03/04/05/06/08)

Филиал ООО «Издательство «Эксмо» в г. Самаре

Адрес: 443052, г. Самара, пр-т Кирова, д. 75/1, лит. «Е»
Телефон: +7 (846) 207-55-50. E-mail: RDC-samara@mail.ru

Филиал ООО «Издательство «Эксмо» в г. Ростове-на-Дону

Адрес: 344023, г. Ростов-на-Дону, ул. Страны Советов, 44А
Телефон: +7(863) 303-62-10. E-mail: info@rnd.eksмо.ru

Филиал ООО «Издательство «Эксмо» в г. Новосибирске

Адрес: 630015, г. Новосибирск, Комбинатский пер., д. 3
Телефон: +7(383) 289-91-42. E-mail: eksмо-nsk@yandex.ru

Обособленное подразделение в г. Хабаровске

Фактический адрес: 680000, г. Хабаровск, ул. Фрунзе, 22, оф. 703
Почтовый адрес: 680020, г. Хабаровск, А/Я 1006

Телефон: (4212) 910-120, 910-211. E-mail: eksмо-khv@mail.ru

Филиал ООО «Издательство «Эксмо» в г. Тюмени

Центр оптово-розничных продаж Cash&Carry в г. Тюмени
Адрес: 625022, г. Тюмень, ул. Пермикова, 1а, 2 этаж, ТЦ «Перестрой-ка»
Ежедневно с 9.00 до 20.00. Телефон: 8 (3452) 21-53-96

Республика Беларусь: ООО «ЭКСМО АСТ Си энд Си»

Центр оптово-розничных продаж Cash&Carry в г. Минске
Адрес: 220014, Республика Беларусь, г. Минск, проспект Жукова, 44, пом. 1-17, ТЦ «Outleto»
Телефон: +375 17 251-40-23; +375 44 581-81-92

Режим работы: с 10.00 до 22.00. E-mail: exmoast@yandex.by

Казахстан: РДЦ Алматы»

Адрес: 050039, г. Алматы, ул. Домбровского, 3А
Телефон: +7 (727) 251-58-12, 251-59-90 (91, 92, 99). E-mail: RDC-Almaty@eksмо.kz

Украина: ООО «Форс Украина»

Адрес: 04073, г. Киев, ул. Вербовая, 17а
Телефон: +38 (044) 290-99-44, (067) 536-33-22. E-mail: sales@forsukraine.com

Полный ассортимент продукции ООО «Издательство «Эксмо» можно приобрести в книжных магазинах «Читай-город» и заказать в интернет-магазине: www.chitali-gorod.ru.
Телефон единой справочной службы: 8 (800) 444-8-444. Звонок по России бесплатный.

Интернет-магазин ООО «Издательство «Эксмо»

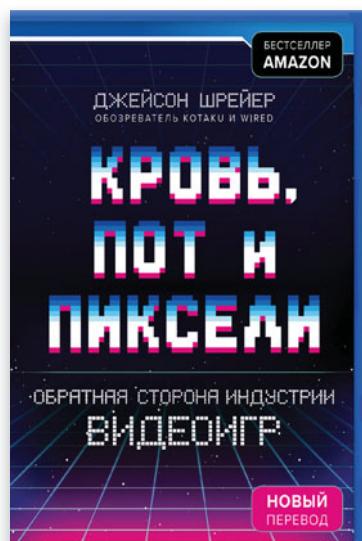
www.book24.ru

Розничная продажа книг с доставкой по всему миру.

Тел.: +7 (495) 745-89-14. E-mail: imarket@eksмо-sale.ru



СЕРИЯ: ЛУЧШИЕ КОМПЬЮТЕРНЫЕ ИГРЫ



ЛЕГЕНДАРНЫЕ КНИГИ О КОМПЬЮТЕРНЫХ ИГРАХ

КОГДА ВЫ ДАРИТЕ КНИГУ, ВЫ ДАРИТЕ ЦЕЛЫЙ МИР

ХОТИТЕ ЗНАТЬ БОЛЬШЕ?

Заходите на сайт:

<https://eksmo.ru/b2b/>

Звоните по телефону:

+7 495 411-68-59, доб. 2261

ВАШ ЛОГОТИП НА КОРЕНЬКЕ



ВАШ ЛОГОТИП
НА ОБЛОЖКЕ

ОБРАЩЕНИЕ
К КЛИЕНТАМ
НА ОБЛОЖКЕ

Разработка игр на Unreal® Engine 4

за **24**
Часа

Всего за 24 урока, каждый продолжительностью 1 час или меньше, вы узнаете, как начать проектировать великолепные игры с помощью движка Unreal Engine 4 под Windows, Mac, PS4, Xbox One, iOS, Android, Интернет, Linux — или для всех сразу!

Пошаговый подход к обучению, представленный в книге, покажет, как работать с интерфейсом Unreal Engine 4, продемонстрирует рабочие процессы и самые мощные редакторы и инструменты движка. За считанные часы вы научитесь создавать эффекты, пользоваться визуальным проектированием, реализовывать физику и даже вести разработку для мобильных устройств и HUD-интерфейсов. Каждый урок дополняет знания, полученные вами в предыдущих, создавая крепкий фундамент для успешной работы с реальными задачами.

Пошаговые инструкции познакомят вас с часто встречающимися задачами разработки в Unreal Engine 4.

Практические, реальные примеры продемонстрируют, как применять навыки.

Контрольные вопросы и упражнения помогут вам протестировать свои знания и расширить навыки.

Примечания и советы подскажут комбинации клавиш и решения.

Арам Куксон — профессор кафедры интерактивного дизайна и разработки игр (ITGM, Interactive Design and Game Development) Колледжа арта и дизайна Саванны (SCAD, Savannah College of Art and Design). Он уже 15 лет преподает игровую графику и гейм-дизайн на основе технологии Unreal Engine. Райан Даунингсок — технический художник франшизы «Gears of War», использует Unreal Engine 4 для конструирования систем разрушений, растительности, визуальных эффектов и многое другое. Клинтон Крамплер работал художником в Battlecry Studios (Bethesda), KIXEYE и Army Game Studio. Он специализируется на артах окружающей среды и разработке шейдеров.

Авторы в сжатой форме самоучителя предложили замечательное и емкое пособие для быстрого старта по разработке компьютерных игр в Unreal Engine 4. Уроки подробны, прекрасно иллюстрированы и спроектированы в лучших традициях практикумов, а тематические эксперименты, вопросы и задачи делают книгу настоящей находкой.

ОЛЬГА МАКСИМЕНКОВА,
к.т.н., Научно-учебная лаборатория интеллектуальных систем и структурного анализа факультета компьютерных наук НИУ ВШЭ

БОМБОРА

Бомбара — это новое название Эксмо Non-fiction, лидера на рынке полезных и вдохновляющих книг. Мы любим книги и создаем их, чтобы вы могли творить, открывать мир, пробовать новое, расти. Быть счастливыми. Быть на волне.

f @ bomborabooks
www.bombara.ru

Дополнительные материалы можно скачать по ссылке:
http://addons.eksмо.ru/it/UE_24.zip

ISBN 978-5-04-103162-6



9 785041 031626 >