Оглавление

ВВЕДЕНИЕ

Становление виртуальной реальности (*VR*) происходит на наших глазах, именно сейчас зарождается огромный спрос на новый и захватывающий опыт в *VR*. В качестве основы растущего многомиллиардного рынка *VR* ставит восхитительно сложные задачи перед разработчиками компьютерных игр, причем одновременно с погружением многих ранее не затронутых отраслей в мир компьютерной графики в реальном времени.

Создание аркадной игры в классическом стиле, визуализация изысканной виллы на травянистых холмах Тосканы — независимо от типа игры *VR* обеспечивает беспрецедентный уровень погружения в любую приглянувшуюся вам сферу. Однако это погружение вызывает множество проблем. При разработке под *VR*, хотите того или нет, вы пишете свои правила.

В этой книге рассмотрены общие парадигмы человеко-машинного взаимодействия, появившихся за последние несколько лет, и лучшие практики в этой области. Все сообщество *VR* — от крупнейших игроков до разработчиков-одиночек — с потрясающей скоростью вносит свой вклад в свод знаний *VR*. Эта книга не только продемонстрирует, как реализовать эти парадигмы средствами *Unreal Engine*, но и поможет выбрать решения, подходящие именно вашему проекту.

В основе книги рецептов лежит практический подход к изучению особенностей *VR*-разработки. Каждый «рецепт» представляет собой вариант сборки достаточно общей системы, подходящей для разнообразных *VR*-игр. Независимо от того, разрабатываете вы игру-стрелялку от первого лица или симулятор для отдыха, содержание каждого примера достаточно абстрактно для использования в игре любого жанра, но дополнительно предлагает конкретные решения, хорошо работающие для определенных типов игр.

Кому будет полезна эта книга?

Эта книга предназначена для людей, которые уже знакомы с навигацией в *Unreal Engine* 4 (*UE*4) и *Blueprints*. Если у вас мало опыта, ознакомьтесь с документацией по *Unreal Engine* перед чтением этой книги в интернете по адресу: *http://docs.unrealengine.com*.

Тем не менее я объясняю большинство вещей, когда дело касается непосредственно кодирования, а большая часть математики раскрывается в приложениях и покрывается основным содержанием книги, поэтому серьезных навыков разработки программного кода от читателя не требуется.

Как организована книга

Книга состоит из трех частей.

1. Часть 1 «Начало работы»: в главах 1–3 содержится введение в некоторые термины, используемые в этой книге и в отрасли *VR*. В этой части также содержатся инструкции по созданию простых проектов для различных *VR*-шлемов.
2. Часть 2 «Рецепты»: главы 4–10 содержат основные «рецепты» книги. Эта часть охватывает все: от взаимодействия с контроллером до схем движения в *VR*.
3. Часть 3 «Приложения»: это вспомогательная информация о редакторе *VR* и ресурсах, которые помогут вам в *VR*-разработке.

Условные обозначения, используемые в этой книге

В этой книге используются следующие соглашения при оформлении текста:

- моноширинным шрифтом набраны блоки программного кода;
- *курсивным начертанием* выделены ключевые слова или фразы.

> **Заметка**
>
> Означает подсказку, предложение или общее примечание.

ДОПОЛНЕНИЕ

Содержит вспомогательную информацию к основному тексту, такую как объяснение используемых математических принципов или работ, связанных с основным содержанием.

> **Предупреждение**
>
> Обозначает предупреждение или предостережение.

Почему *Blueprints*?

Когда вы программируете в *UE4*, есть два способа реализации логики игры: язык визуального программирования *Blueprints* и ставший традиционным язык программирования *C++*.

По сравнению с *Blueprints*, *C++* может быть сложнее, поскольку изучение синтаксиса может занять некоторое время; однако он предлагает различные варианты в использовании скрытых возможностей ядра. Слабое владение *C++* не создаст проблем при работе с этой книгой, потому что большая часть материала представлена на уровне, чтобы возможности *Blueprints* позволяли его освоить.

Использование *Blueprints* также облегчает способы миграции вашей работы из одного проекта в другой, что позволит вам взять любую работу, проделанную по этой книге, и легко применить ее в ваших проектах.

Дополнительные материалы

По адресу *https://eksmo.ru/files/ue4_vr_projects.zip* можно скачать архив с файлами с исходным кодом для каждой части. Это позволит быстро проверить работоспособность рецептов, приведенных в книге.

Об авторе

Митч Маккефри — независимый разработчик игр и создатель многих общественных ресурсов для разработчиков виртуальной реальности на *Unreal Engine*. Он обучает лучшим практикам разработки игр на своем *YouTube*-канале *Micth's VR Lab*, где демонстрируются примеры, предложенные членами сообщества *VR*-разработчиков. Адрес его сайта *https://mitchellmccaffrey.com*.

ЧАСТЬ I

НАЧАЛО РАБОТЫ

ГЛАВА 1

ТЕРМИНОЛОГИЯ И ЛУЧШИЕ ПРАКТИКИ

Мир разработки виртуальной реальности (*VR*) может показаться сложным из-за разнообразия доступного на рынке оборудования и программного обеспечения. Кроме того, поскольку *VR* — новая среда, многие принципы, принятые разработчиками игр, неприменимы и могут не работать в мире *VR*-игр.

Если вы не уверены, что понимаете разницу между *Oculus VR*, *OSRV* и *OpenVR*, или просто ищете общие рекомендации о том, как начать работать с *VR*, эта глава для вас.

1.1. Терминология

Экосистема *VR* непрерывно усложняется и развивается благодаря обилию технологий, программного и аппаратного обеспечения. Чтобы убедиться, что мы верно понимаем друг друга, рассмотрим ключевые части этой экосистемы, которые должен знать каждый *VR*-разработчик, использующий *Unreal Engine* 4 (*UE*4). Если вы знакомы с текущим состоянием отрасли виртуальной реальности и используемыми технологиями, можете пропустить эту главу.

1.1.1. Устройства

Для работы с виртуальной реальностью вам потребуются *шлем виртуальной реальности* [*Head Mounted Display — HMD*] и контроллер. *Unreal Engine* на уровне коробочного решения поддерживает большинство из представленных на рынке решений, что избавляет разработчиков от необходимости однозначного выбора конкретного *VR*-шлема на первых этапах проекта. Кроме того, в *UE*4 для *VR* реализован высокий уровень абстракции, что позволяет легко справиться со сменой устройства, для которого вы разрабатываете проект (или если вы ориентируетесь сразу на несколько устройств).

Таблица 1.1 содержит список и описание *VR*-шлемов, штатно поддерживаемых *UE*4.

Таблица 1.1. Совместимые *VR*-шлемы (*HMD*)

VR-шлем	Описание
Samsung Gear VR	*Gear VR* — аппаратура, совместимая со смартфонами *Samsung*, разработанная в партнерстве *Oculus* и *Samsung* с целью обобщить возможности *Oculus* в области программного обеспечения для мобильной виртуальной реальности. При работе с *Gear VR* вы можете использовать все функции и преимущества, которые дает среда разработки *Oculus Mobile software development kit* (*Oculus Mobile SDK*). *Gear VR* в настоящее время поддерживает только отслеживание поворотов головы виртуальной модели
HTC Vive	*HTC Vive* разработан в результате партнерства, сходного с *Gear VR*, но в котором *HTC* выступили поставщиком аппаратного обеспечения, а *Valve* — программного *SDK* в виде *SteamVR/OpenVR*. *Vive* обеспечивает отслеживание вращений, а также взаимно однозначное отслеживание положения шлема и включенных контроллеров движения. С прицелом на разработку под *Vive*, *Valve* предоставляет *OpenVR SDK* (для получения дополнительной информации о *OpenVR* см. таблицу 1.3), который предоставляет доступ к различным датчикам и данным настройки, необходимым для *VR*
Oculus Rift	Аппаратура *Oculus Rift* прошла через много итераций, представленных как выпуски для разработчиков, не предназначенные для широкой публики. В 2016 году *Oculus* выпустила первую массовую версию (*CV*1), предназначенную для продажи пользователям. *CV*1 — это полностью интегрированная система, состоящая из оборудования *Oculus* и программного обеспечения (см. *SDK Oculus* в таблице 1.3). *Rift* поддерживает отслеживание вращения, подобно *HTC Vive*. В комплект также входит контроллер *Xbox One* и пульт дистанционного управления *Oculus Remote*, являющийся устройством ввода по умолчанию; *VR*-контроллеры *Oculus Touch* (см. таблицу 1.2) продаются отдельно

VR-шлем	Описание
Google Cardboard	*Google Cardboard* — это небольшая и недорогая (350–1500 рублей) картонная коробочка, в которой есть одна кнопка и линзы. Коробка позволяет закрепить смартфон и просматривать стереометрический контент. «Картон» использует встроенные датчики смартфона для обнаружения вращения головы пользователя (разумеется, на неоткалиброванном телефоне измерения будут неточными). *Google Cardboard* использует собственные *SDK Android* для обеспечения совместимости с устройствами (что может приводить к существенным задержкам рендеринга по сравнению с другими решениями). Тем не менее «Картон» на сегодня является самой дешевой *VR*-гарнитурой
Google VR/Daydream VR	*Google VR* или *Daydream VR* — еще одна инициатива компании *Google*. По сравнению с *Google Cardboard* эта программно-аппаратная система контролируется жестче. Для работы с ней ваш смартфон и гарнитура должны быть классифицированы как *Daydream Ready*, что позволяет *Google* использовать более продвинутые функции программного обеспечения. *Daydream* также поддерживает контроллер *Daydream* (см. таблицу 1.2), предоставляющий сенсорную панель и обеспечивающий отслеживание вращения. Таргетинг *Daydream* реализуется через *Google VR SDK* (таблица 1.3)
PlayStation VR	*PlayStation VR* является периферийным устройством для *VR*-игр на игровых приставках (консолях) *Sony PlayStation* 4 и *PlayStation* 4 *Pro*. Отслеживание положения осуществляется при помощи внешней камеры *PlayStation*, отслеживание вращения реализовано гиростабилизаторами (*IMUs*, *inertial measurement units*) примерно так же, как у других шлемов, представленных в этой таблице. Игроки могут использовать контроллеры *PlayStation Move* (таблица 1.2). Также можно использовать контроллер *DualShock* 4, в том числе для отслеживания положения
OSRV	Виртуальная реальность с открытым исходным кодом (*Open Source Virtual Reality, OSRV*) — это инициатива, поддерживаемая несколькими корпоративными партнерами. В настоящее время *OSRV* предлагает *Hacker Development Kit (HDK)* в виде *HDK* 1.3 и *HDK* 2, разработанных компанией *Razer*. Обе гарнитуры обеспечивают отслеживание положения и вращения

В *UE*4 контроллеры перемещения поддерживаются с помощью единого компонента *Motion Controller Component*, который упрощает взаимодействие, в том числе с несколькими контроллерами. Список контроллеров перемещения, поддерживаемых *UE*4, приведен в таблице 1.2.

Таблица 1.2. Совместимые контроллеры

Контроллеры перемещения	Описание
Oculus Touch	Контроллеры *Oculus Touch* позволяют отслеживать как вращение, так и положение при помощи системы *Constellation*. Каждый контроллер имеет пару кнопок (*A* и *B* на правом, *X* и *Y* на левом контроллере) и аналоговый джойстик под большой палец, кнопку под указательный палец («спусковой крючок») и кнопку для остальных пальцев. Поддержка этих контроллеров осуществляется через *Oculus SDK*
Vive	Контроллеры *Vive* позволяют отслеживать вращение и перемещение при помощи системы *Lighthouse*. Каждый контроллер содержит круговой трекпад, который работает как аналоговый джойстик или кнопка, а также кнопку под указательный палец («спусковой крючок»), кнопку меню и кнопку «Захват». Работа с контроллерами *Vive* обеспечивается при помощи *OpenVR SDK*

Контроллеры перемещения	**Описание**
PlayStation Move	Контроллер *Move* позволяет отслеживать вращение и перемещение при помощи камеры *PlayStation* и светящегося шара на самом контроллере. Помимо обычных для контроллеров *PlayStation* кнопок *Cross* (крест), *Circle* (окружность), *Triangle* (треугольник), *Square* (квадрат), *Start* (запуск) и *Select* (выбор), контроллер *Move* имеет кнопку *Move* (перемещение), а также «спусковой крючок» (или кнопку *T*)
Daydream	Контроллер *Daydream* поддерживает только отслеживание вращения. У контроллера также имеется сенсорная панель, которая может выступать в качестве кнопки, и кнопка *App* под ней. Работа с контроллером *Daydream* осуществляется при помощи *Google VR SDK*

1.1.1.1. Программное обеспечение

Существует множество программных библиотек, *SDK** и *API***, помогающих взаимодействовать с оборудованием виртуальной реальности. Идеология *UE*4 построена на повышении уровня абстракции до единых аппаратно-независимых интерфейсов и компонентов, что облегчает совместимость. При необходимости в ваших руках остается возможность прямого взаимодействия с различными *SDK*. Вам как разработчику будет полезно ознакомиться различиями в концепциях проектирования при помощи этих *SDK*, поскольку при разработке игры или реализации другого проекта могут требоваться преимущества конкретных функций выбранного *SDK*. Для работы с пакетами *SDK*, вам не нужно скачивать какие-либо отдельные файлы, поскольку *UE*4 включает их при скачивании модулей.

Список и описание совместимых *SDK* приведены в таблице 1.3.

Таблица 1.3. Совместимые *SDK*

SDK	**Описание**
Oculus PC SDK	*Oculus PC SDK* обеспечивает *UE*4 всей необходимой информацией для рендера подходящего представления. Эта информация включает в себя положение и ориентацию головы пользователя, а также его межзрачковое расстояние (*interpupillary distance*, *IPD*). *UE*4 может сгенерировать кадр и загрузить его в *Oculus compositor*, где будут применены искажения, необходимые для настройки линз *VR*-шлема. К счастью, движок в большинстве случаев справляется с этим и не требует от вас вмешательства. *Oculus SDK* также предоставляет слои, которые дают вам возможность рисовать на экране объекты, имеющие разное разрешение. Это полезно для пользовательского интерфейса (*UI*), где может потребоваться, например, отобразить текст с разрешением выше, чем у фона. Такие функции, как *ATW* (*Asynchronous Timewarp*; см. таблицу 1.4) автоматически обрабатываются во время выполнения и не требуют от вас дополнительных действий

* *Software Development Kit* — комплект для разработки программного обеспечения. — *Прим. ред.*

** *Application Programming Interface* — интерфейс прикладного программирования. — *Прим. ред.*

SDK	Описание
Oculus Mobile SDK	*Oculus Mobile SDK* предоставляет доступ ко многим возможностям, соответствующим *PC SDK*, включая информацию о положении головы и другую пользовательскую информацию, необходимую для рендеринга. Как и среда исполнения *PC*, мобильная среда исполнения использует *ATW*, но она дополнительно поддерживает рендеринг переднего (или кадрового) буфера, также известный как *scanline racing* (см. таблицу 1.4)
Oculus Audio SDK	*Oculus Audio SDK* предоставляет возможности реалистично распределить звук в пространстве и создать 3*D*-звук для *VR*. Этот *SDK* позволяет применять *HRTF** (*head-related transfer function*) к аудиоисточникам для имитации направленного звука (посредством моделирования геометрии человеческого уха). Он также может имитировать расстояние от звука до уха, гася громкость. *HRTF Oculus Audio SDK* реализован в *UE*4, позволяя вам настраивать кривые, ограничивающие распространения звука. Однако в настоящее время реализовать их можно только в проектах на базе *DirectX* (например, *PC*)
OpenVR SDK	*OpenVR SDK* во многом схож с *Oculus PC SDK* и предоставляет доступ к информации о *VR*-шлеме (включая положение и ориентацию в пространстве). *OpenVR SDK* также предоставляет интерфейсы для доступа к системе *Chaperone* (описывающей виртуальные границы, обозначающие игровое пространство для пользователя) и *Overlay* для генерации 2*D*-контента в компоновщике аналогичном слоям *Oculus*. *OpenVR* также поддерживает перепроецирование (аналогично *ATW*) и предсказание
OSRV SDK	*OSVR SDK* дает вам доступ к набору интерфейсов для создания *VR*-сюжетов. Построенный по принципу «поддержка всего», данный *SDK* включает наиболее значимые интерфейсы для получения такой информации *VR*-шлема, как положение и ориентация в пространстве и интерфейсы для отслеживания глаза, всенаправленных беговых дорожек (*omnidirectional treadmills*), жестов и так далее. *Unreal Engine* интегрируется с *OSRV SDK*, начиная с версии 4.12
Google VR SDK	*Google VS SDK* дает вам доступ к данным отслеживания положения головы *VR*-шлема и набор функций для оптимизации вашего *VR*-проекта. Это позволяет включить устойчивый режим производительности, что важно для запусков на смартфоне, потому что смартфон может перегреться от высокой нагрузки и резко снизить производительность, что может плохо повлиять на восприятие пользователем *VR*. *Google VS SDK* также предоставляет доступ к функциям *scanline racing*, которые оптимизируют задержку в линии рендеринга (см. таблицу 1.4)

Помимо *SDK* и библиотек, которые *UE*4 использует для взаимодействия с различной *VR*-аппаратурой, в ядре также реализованы некоторые программные функции, позволяющие значительно улучшить впечатления пользователя от *VR*. Однако существуют и другие программные функции (например, *ATW*), реализованные в различных средах исполнения, как следствие, включенные в ядро по умолчанию и неподконтрольные разработчикам. Таблица 1.4 содержит примеры функциональности обоих видов, доступных в *UE*4.

* *Head-related transfer function* — передаточная функция, описывающая положение источника звука относительно слушателя. — *Прим. ред.*