window.yaContextCb.push(()=>{ Ya.Context.AdvManager.render({ "blockId": "R-A-7038977-3", "renderTo": "yandex_rtb_R-A-7038977-3" }) })

Что за видеокурс.

Авторский обучающий видеокурс по разработке игр на движке Unreal Engine 5.

Автор курса Михаил Русаков.

За 81 день Вы научитесь создавать игры на C++ в Unreal Engine 5, увидите наглядный пример создания игры, создадите свой собственный серьёзный проект для портфолио, и всё это с нуля!

Что вошло в видеокурс.

Содержание курса.

Модуль 1. Введение. 3 урока.

Список уроков:

1. Как проходить курс?

- В уроке будут даны важные рекомендации по эффективному прохождению курса.

2. Установка необходимого ПО.

- В уроке скачаем и установим весь необходимый софт. Уже даже на этом этапе у новичков возникают проблемы, но благодаря этому уроку у Вас их не будет.

3. Установка и настройка clang-format.

- Совершенно необходимый инструмент, нужный для быстрого форматирования/выравнивания кода.

Модуль 2. Основы C++ в Unreal Engine. 17 уроков

Материал этого раздела совершенно необходим, прежде чем пытаться хоть что-то делать в Unreal Engine на C++.

Список уроков:

1. Создание первого проекта.

- Создадим проект из шаблона ThirdPerson и подробно разберём каждую строчку кода. Это уже будет отличным стартом.

2. Как работает движок изнутри?

- Достаточно важный урок, если Вы действительно хотите понять, как работает вся «магия» изнутри. Это, во-первых, может пригодиться на собеседовании, во-вторых, Вы лучше будете понимать, что делаете.

3. Рефакторинг шаблона от третьего лица.

- В уроке мы почистим сгенерированные классы от лишнего мусора, переименуем классы (а это не так просто сделать, как кажется), наведём порядок.

4. Создание C++-класса.

- В уроке создадим C++-класс с нуля.

5. Типы данных.

- В рамках урока Вы познакомитесь с основными типами, которые нам предоставляет фреймворк Unreal Engine 5.

6. Логирование.

- После урока Вы научитесь делать записи в лог, в том числе, выводя переменные самых разных типов. Это необычайно важные знания, особенно, когда речь заходит об отладке.

7. Макрос UPROPERTY.

- Подробно обсудим этот важнейший макрос в Unreal Engine.

8. Construction Script в C++.

- Разберём аналог Construction Script из Blueprint, но в рамках C++.

9. Asserts.

- Разберём эту группу макросов, которые очень важны, если Вы хотите писать качественный код.

10. Макрос UFUNCTION.

- Ещё один фундаментальный макрос, который мы тщательно обговорим.

11. Структуры.

- Обсудим создание структур в C++, которые будут доступны, в том числе, и в Blueprint.

12. Перечисления.

- После этого урока Вы научитесь создавать перечисления в C++ для Unreal Engine.

13. Делегаты.

- Это эквивалент Event Dispatcher из Blueprint. И из урока Вы узнаете, как создавать делегаты с параметрами или без, только для C++ или для C++ и Blueprint.

14. Виджеты.

- Обсудим процедуру создания виджетов с использованием C++. Здесь очень много тонкостей и нюансов, дойти до которых можно лишь, имея большой опыт. Но Вы получите всё и сразу в этом уроке.

15. Таймеры.

- После этого урока Вы научитесь использовать таймеры в C++.

16. Delay.

- Функция Delay является очень часто используемой (гораздо чаще, чем надо, кстати), и как её реализовать в C++ мы и обсудим.

17. Интерфейсы.

- В уроке мы разберём процедуру создания и применения интерфейсов на C++, которая доступна благодаря фреймворку Unreal Engine.

Модуль 3. Создание игры: подготовка проекта. 5 уроков.

Материал этого раздела совершенно необходим, прежде чем пытаться хоть что-то делать в Unreal Engine на C++.

Список уроков:

1. Разбор технического задания.

- Один из ценнейших уроков курса, в котором мы, во-первых, подробно разберём то, что мы будем создавать. Во-вторых, Вы научитесь писать свои технические задания (для себя или для других), узнаете про итеративные методики создания ПО, поймёте, как, в целом, строится pipeline разработки.

2. Создание и настройка проекта.

- Сделаем все необходимые стартовые вещи.

3. Добавление персонажа.

- Добавим персонажа для игрока, импортируем анимации, наведём порядок.

4. Добавление животных.

- Добавим животных и все необходимые ассеты для них.

5. Добавление предметов.

- Поскольку у нас будет создаваться инвентарь в будущем, мы заранее добавим сами предметы в наш проект, чтобы потом на этом не останавливаться.

Модуль 4. Создание игры: основные классы и компоненты. 13 уроков.

С этого раздела начинается активное и серьёзное программирование.

Список уроков:

1. Создание уровней и GameMode.

- Создадим карту для тестового уровня, для главного меню, а так же необходимые GameMode.

2. Создание класса персонажа и игрока.

- После этого урока у нас уже будет готовый персонаж с видом от первого лица, которым можно управлять. Оба класса будут созданы с нуля.

3. Создание AnimInstance и AnimBlueprint.

- В рамках этого урока мы создадим необходимый интерфейс (чтобы не было лишних зависимостей), реализуем его, применим. Так же создадим AnimInstance, который ближе к концу урока ляжет в основу AnimBlueprint.

4. Создание класса животных и их AnimBlueprint.

Создадим класс для животных, которые в будущем будут жить своей жизнью. Также благодаря нашему классу AnimInstance, созданному ранее, Вы увидите, насколько приятно создавать AnimBlueprint для самых разных персонажей.

5. Создание HUD.

- Создадим HUD, основа которого нам пригодится в будущем.

6. Создание «границы».

- Сделаем так, что если игрок попытается уйти за пределы карты, то его возвращало обратно на уровень. Подобная механика реализована в таких играх как «Ведьмак 3: Дикая охота» и «Kingdom Come Deliverance».

7. Создание компонента взаимодействия.

- Создадим компонент, который позволит нам взаимодействовать с внешним миром (например, взаимодействовать с магазином, поднимать предметы).

8. Создание компонента здоровья.

- Думаю, не нужно пояснять, что просто добавить переменную Health в класс персонажа - это плохая идея. Это классический пример нарушения принципа единственной ответственности класса. В общем, в уроке мы всё сделаем правильно.

9. Создание компонента для выносливости.

- Поскольку в будущем быстрый бег и прицеливание из лука будут расходовать выносливость, нам необходимо заранее написать компонент для выносливости.

10. Создание компонента CharacterMovementComponent.

- Поскольку у нас есть режим спринта, а также расходование выносливости в этом режиме, мы реализуем свой компонент для передвижения, попутно заменив в классе персонажа базовый компонент своим.

11. Режим спринта и расход выносливости.

- Реализуем спринт, плюс расход выносливости от него.

12. Создание компонента камеры.

- При расходе выносливости у нас будет искажаться и затемняться экран, поэтому будет разумным создать свой компонент камеры, чем мы и займёмся.

13. Создание компонента CameraShakeSourceComponent.

- Данный компонент будет создавать тряску при прицеливании из лука. Более того, чем ниже выносливость, тем сильнее будет тряска (и тем сложнее будет прицелиться).

Модуль 5. Создание игры: AI. 6 уроков.

В рамках этих уроков мы реализуем на C++ всю логику искусственного интеллекта для животных.

1. Создание AnimalAIController.

- Создадим основной контроллер, который будет своего рода «пультом», отвечающего за включение и выключение тех или иных компонентов, которые будут созданы впоследствии.

2. Создание базового AI-компонента.

- Создадим базовый AI-компонент, от которого будут наследоваться остальные.

3. Создание компонента патрулирования.

- При запуске в контроллере этого компонента животные будут спокойно прогуливаться по территории недалеко от своего спавна.

4. Создание компонента для убегания от противника.

- Если животное увидит противника, то немедленно начнёт от него убегать до тех пор, пока расстояние не будет безопасным. Как реализовать подобную логику, Вы узнаете в этом уроке.

5. Подключение AI-компонентов к контроллеру.

- В уроке мы возьмём созданные компоненты и подключим их к нашему контроллеру, а также настроим логику их включения/выключения.

6. Создание спавнера.

- Создадим спавнер для животных (вообще говоря, он подойдёт для любых Actor), который можно будет в редакторе вытащить на сцену и настроить.

Модуль 6. Создание игры: инвентарь. 6 уроков.

В рамках этого раздела мы будем создавать техническую часть инвентаря для нашей игры.

1. Добавление своих настроек в Project Settings.

- В этом уроке Вы увидите, как можно добавить свои собственные настройки в Project Settings, доступ к которым потом получить из кода на C++.

2. Создание структуры и DataTable.

- В уроке мы создадим структуру для предметов, а также таблицу с ними, которую будет очень легко заполнять, добавляя всё новые и новые предметы в игру.

3. Создание класса предметов.

- В уроке Вы узнаете, как использовать DataTable, создадим необходимые Helper и, само собой, сам класс предмета.

4. Создание компонента инвентаря.

- Я своими глазами видел, как один «учитель» с очень положительными комментариями под его видео создавал инвентарь в виде Actor (видимо, у инвентаря по мнению автора есть местоположение и поворот в мире), а попутно хранил сами предметы в массиве опять же типа Actor. Мы же всё с Вами реализуем, как полагается: с интерфейсом, с хранением только RowName и в виде ActorComponent.

5. Сбор предметов.

- Реализуем подбирание предметов и добавление их в инвентарь.

6. Создание компонента стартовых предметов.

- Чтобы в начале игры у наших персонажей (игрока и животных) были уже какие-то предметы в инвентаре, мы реализуем и добавим необходимый для этого компонент.

Модуль 7. Создание игры: виджеты. 19 уроков.

В рамках этого большого раздела будут созданы все необходимые виджеты. Помимо большого объёма C++-кода, здесь нам потребуется использовать и редактор виджетов для их вёрстки.

Список уроков:

1. Создание базового класса виджета.

- В уроке разберём, зачем нужен наш собственный базовый класс для всех виджетов, и создадим его.

2. Создание виджета кнопки.

- Поскольку кнопки мы будем использовать многократно в других виджетах, разумно создать отдельный виджет для неё, чтобы всегда можно было поменять её дизайн, а, по сути, автоматически изменить дизайн всех остальных виджетов.

3. Создание виджета для взаимодействия.

- Непосредственно создадим логику виджета и сделаем его вёрстку.

4. Создание компонента InteractWidgetComponent.

- Создадим компонент, который будет отвечать за вывод виджета при наведении на объекты мира (например, на предметы, которые можно подобрать) с информацией о них и о том, как с ними взаимодействовать.

5. Вывод виджета при взаимодействии с миром.

- Объединим созданный виджет из позапрошлого урока с компонентом из прошлого и подключим всё это, например, в тот же класс предмета.

6. Создание GameHUDWidget.

- Создадим и выведем виджет, который будет доступен во время игры.

7. Создание виджета для слота инвентаря.

- Создадим виджет слотов инвентаря.

8. Создание виджета инвентаря.

- Полностью создадим весь виджет инвентаря, включая логику и вёрстку.

9. Открытие и закрытие виджета инвентаря.

- Используя делегат, реализуем механизм открытия и закрытия инвентаря.

10. Создание виджета для сообщений инвентаря.

- Реализуем механизм, при котором, подбирая или тратя предметы в инвентаре, у нас будет писаться соответствующее сообщение.

11. Создание стратегий для инвентаря.

- Реализуем паттерн Strategy, благодаря которому мы гибко сможем делать обработку нажатий на предмет в инвентаре.

12. Создание главного меню.

- Создадим виджет для главного меню.

13. Обработка главного меню.

- В уроке обработаем кнопки главного меню.

14. Создание загрузочного виджета.

- Процесс перехода из главного меню на уровень игры занимает некоторое время, на протяжении которого желательно вывести какой-нибудь загрузочный виджет, чем мы и займёмся в этом уроке.

15. Открытие и закрытие загрузочного виджета.

- Реализуем логику появления и плавного исчезновения загрузочного экрана.

16. Создание меню паузы.

- В этом уроке мы создадим меню паузы.

17. Открытие и закрытие виджета паузы.

- Из этого урока Вы узнаете, как правильно реализовать функционал переключения видимости виджета с помощью нажатия одной и той же кнопки.

18. Обработка меню паузы.

- В меню паузы будет несколько кнопок, обработку которых мы реализуем в этом уроке.

19. Добавление курсора.

- Добавим свой собственный курсор в нашу игру.

Модуль 8. Создание игры: лук и экипировка. 10 уроков.

В этом разделе мы займёмся созданием механизма экипировки и стрельбы из лука.

Список уроков:

1. Создание класса оружия.

- В уроке будет создан общий класс для любого оружия.

2. Создание анимации лука.

- Поскольку в игре тетива лука будет натягиваться, нам потребуется AnimInstance и AnimBlueprint, чем мы и займёмся в этом видео.

3. Создание базового компонента для экипировки.

- Данный компонент будет являться общим для любых других компонентов-экипировки.

4. Создание компонента для экипировки оружия.

- Создадим компонент экипировки, который подойдёт для любого оружия. Всю общую логику, характерную именно для экипированного оружия, мы пропишем в этом компоненте.

5. Создание компонента для экипировки лука.

- Реализуем компонент, который будет отвечать за экипировку лука.

6. Создание класса стрелы.

- Реализуем класс для стрелы, которая будет лететь по баллистическому закону.

7. Создание класса лука.

- Создадим класс для лука.

8. Экипировка лука.

- Реализуем нужную стратегию и экипируем уже сам лук, путём клика по нему на слот в инвентаре.

9. Реализация стрельбы из лука.

- Реализуем саму стрельбу из лука, а так же сделаем так, чтобы при попадании стрелы в голову, наносился повышенный урон.

10. Сборка ресурсов с животных.

- Сделаем сбор ресурсов с животных и добавление их в инвентарь.

Модуль 9. Создание игры: магазины. 3 урока.

В рамках этого раздела мы реализуем механизм магазинов. У нас будут магазины, которые скупают товары (например, кожу и мясо), а будут магазины, которые продают товар (например, различные луки и стрелы).

Список уроков:

1. Создание базового класса магазина.

- В уроке мы создадим общую логику для любых магазинов.

2. Создание класса для магазина-покупателя.

- Реализуем магазин, который будет скупать соответствующие предметы у игрока, когда тот взаимодействует с магазином.

3. Создание класса для магазина-продавца.

- Создадим логику магазина, который продаёт различные предметы. И при взаимодействии с ним игрок сможет делать покупки.

Модуль 10. Создание игры: смена времени суток. 4 урока.

В этом разделе мы создадим динамическую смену дня и ночи.

Список уроков:

1. Создание кривой для солнца.

- Поскольку солнце будет двигаться, то мы заранее создадим CurveFloat, в котором опишем движение солнца.

2. Создание класса для изменения времени суток.

- Напишем основной класс, в котором можно будет задавать текущее время, а также в котором будет происходить его течение.

3. Добавление Луны.

- В уроке добавим Луну и подключим к нашей системе., причём не только с её движением по небосводу, но и с фазами луны.

4. Движение Луны и её фазы.

- Настроим движение Луны по небосводу, а дополнительно сделаем и плавное изменение фаз Луны.

Модуль 11. Создание игры: звуки и музыка. 8 уроков.

В рамках этого раздела мы добавим в наш код проигрывание самых разных звуков и музыки.

Список уроков:

1. Добавление звуков лука.

- Натягивание тетивы, выстрел из лука, попадание стрелы - для всего этого мы добавим звуки.

2. Добавление звуков кнопки.

- Достаточно простой урок, в котором мы обойдёмся без программирования.

3. Добавление звуков животных.

- Добавим различные звуки для животных. Они будут проигрываться произвольным образом в случайное время.

4. Добавление звуков леса.

- Поскольку часть нашего мира будет покрыта лесами, то разумно будет сделать правильную атмосферу в нём с помощью звуков, чем мы и займёмся в этом уроке.

5. Добавление звука попытки уйти с карты.

- Добавим фразу персонажа при попытке уйти за границы карты.

6. Создание классов AnimNotify.

- Объекты данных классов будут вызываться во время проигрывания анимации. Таким образом, в определённые моменты анимации мы сможем выполнить определённую логику.

7. Добавление звуков шагов.

- В рамках этого урока мы, используя класс AnimNotify из прошлого урока, реализуем проигрывание звуков шагов в зависимости от поверхности под ногами персонажа.

8. Добавление музыки.

- Добавим музыку в игру.

Модуль 12. Создание игры: игровой уровень. 7 уроков.

Заключительный раздел, в котором мы соберём все наши уже работающие механики в рамках игрового уровня, а также сделаем и протестируем готовую сборку.

Список уроков:

1. Рефакторинг.

- Проведём небольшой рефакторинг для улучшения кода.

2. Замена Input на Enhanced Input.

- Заменим классические Input на Enhanced Input, которые уже на данный момент являются стандартом Unreal Engine 5.

3. Импорт ассетов.

- В уроке мы импортируем необходимые ассеты для дизайна уровня.

4. Создание игрового уровня.

- Используя импортированные ассеты, создадим мир нашей игры.

5. Добавление игровых элементов.

- Добавим все необходимые элементы: спавнеры животных, звуки леса, навигационную сетку, магазины, дом ловчего, стартовые предметы и много чего ещё по мелочи.

6. Сборка проекта.

- Соберём и протестируем shipping-версию.

7. Заключение.

- Важные слова напутствия и советы, в каком направлении стоит двигаться дальше, исходя из моего опыта.

Видеопрезентация.

Бонус.

- Курс "Тестирование Unreal-проектов на Python"

Дополнительную информацию смотрите на сайте видеокурса: **Узнать подробности (перейти на страницу видеокурса)**

**   Карточка курса:** * Название: Программирование на C++ в Unreal Engine
Автор(ы): Русаков Михаил
Формат курса: Видеокурс
Цена: 5970 руб.
Партнерская программа: Есть
*-данные карточки курса могут не соответствовать текущим данным о курсе, подробную информацию смотрите на сайте курса
**   Другие курсы автора(ов), издателя:**Ссылки на страницы автора(ов), издателя: Русаков Михаил
window.yaContextCb.push(()=>{ Ya.Context.AdvManager.render({ "blockId": "R-A-7038977-2", "renderTo": "yandex_rtb_R-A-7038977-2" }) })