Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 21.Б10-мм

Реализация видеоигры «Time Reactor» в жанре Sci-Fi с помощью Unity

Береснёв Руслан Анатольевич

Отчёт по учебной практике
в форме «Производственное задание»

Научный руководитель:
ст. преподаватель кафедры системного программирования, к.т.н., Литвинов Ю. В.

Санкт-Петербург
2024

Оглавление

Введение

Современная IT-индустрия подразделяется на множество областей, среди которых есть и разработка видеоигр. Каждый год выпускается множество видеоигр разного масштаба в рамках определённых жанров и бюджета. Цель создания игры может заключаться в приобретении опыта реализации полноценного продукта со всеми стадиями разработки и сопровождения, а может, например, заключаться в получении прибыли от продажи пользователям интересного геймплея.

Для разработки используются библиотеки для работы с графикой и алгебраическими операциями, либо полноценные «движки», в которых можно смоделировать физические законы и удобно задать поведение и взаимодействие объектов. Одним из таких «движков» является Unity Engine.

Дистрибуция видеоигр происходит в пределах торговых площадок. Примером площадки-дистрибьютора может быть Steam, где помимо продуктов от таких известных издателей, как Valve, Electronic Arts, Ubisoft и т.д. [2], есть много небольших игр начинающих разработчиков. Поэтому создание простой видеоигры станет первым полноценным опытом игровой разработки, результат которого вполне может быть опубликован в Steam.

Два года назад была начата разработка проекта «Time Reactor», в котором затем принимала участие группа студентов, а теперь планируется постепенное завершение проекта и его релиз в Steam.

В рамках учебной практики планируется добавление новой функциональности в уже существующий проект на Unity в жанре научной фантастики, а также завершение игрового цикла: это означает, что игрок наконец сможет полноценно пройти хоть и «сырую» в плане механик и дизайна, но технически законченную игру.

1. Постановка задачи

Целью работы является доработка игры «Time Reactor» с помощью Unity Engine. Для её выполнения были поставлены следующие задачи:

1. Выполнить обзор использованных в проекте технологий и ресурсов.
2. Выполнить обзор реализованной ранее функциональности.
3. Реализовать механики, необходимые для возможности прохождения игры.
4. Провести апробацию среди пользователей, которые пройдут игру.

2. Обзор

2.1. Обзор использованных технологий и ресурсов

Целью обзора является выявление преимуществ и недостатков инструментов и ресурсов, использованных при разработке проекта.

Unity Engine

Разработка игры ведётся с использованием «движка» Unity Engine [6] — кросплатформенной среды, предназначенной для создания 2D и 3D десктопных и мобильных игр, а также игр с виртуальной или дополненной реальностью.

Преимуществами Unity перед библиотеками для работы с графикой являются:

- встроенные механизмы обработки графики: не нужно вручную задавать правила и порядок отрисовки объектов, достаточно нужным образом на сцене расположить камеру относительно объектов, которые необходимо отрисовать;
- возможность симуляции физических процессов: в документации [5] представлен спектр классов и методов для работы с различными физическими явлениями;
- наличие окна с текущей сценой, в которой находятся все взаимодействующие объекты: данная функциональность позволяет размещать объекты игрового мира с помощью ручного перемещения в пределах сцены;
- возможность задавать поведение объектов в специальном инспекторе свойств с помощью отдельных скриптов на языке программирования C# и регулирования их полей через этот же инспектор.

Недостатком Unity перед разработкой с помощью графических библиотек является отсутствие полного контроля над событиями рендера

графики и обработки физических взаимодействий. Например, пользователь не может задать воздействие источника света и поведение теней для каждого объекта отдельно, а может лишь выбрать общий для сцены способ рендеринга [3].

Причина выбора Unity Engine, а не альтернативной среды для создания видеоигр — проект уже некоторое время разрабатывался с использованием Unity, поэтому нет смысла переносить всю созданную архитектуру в какую-либо другую среду.

Unity Asset Store

Для создания игровых локаций используются ассеты, которые можно приобрести в Unity Asset Store [4] — так называются готовые модели объектов, различные текстуры или специфичные настройки освещения, которые помогают задать необходимую атмосферу игре. Чтобы не тратить много времени на создание ассетов вручную, можно приобрести их в Asset Store — специальной площадке от Unity, где размещены самые разнообразные как бесплатные, так и платные ассеты. Объекты интерьера комнат, а также многие предметы в разрабатываемой игре как раз являются такими ассетами.

Преимущество Unity Asset Store заключается в относительном удобстве приобретения необходимых ассетов засчёт интеграции со средой Unity: выбранный в Asset Store ассет можно сразу импортировать в проект. Также стоит упомянуть, что почти все ассеты (и платные, и бесплатные) попадают под лицензию “Standard Unity Asset Store EULA”, благодаря которой они могут быть использованы даже в коммерческих проектах без указания авторства. Существенным недостатком Asset Store является небольшой ассортимент вариантов некоторых ассетов, тем более бесплатных: например, при запросе “Tesla coil” появится всего три результата, которые при этом являются платными.

2.2. Обзор проекта

Цель данного обзора — продемонстрировать системы проекта, которые были реализованы под руководством автора до начала работы над данной практикой другими контрибьюторами.

Важной системой является механизм случайной генерации интерьера в комнатах [7]. На сцене создаётся специальный объект, который осуществляет генерацию. В параметрах объекта можно настроить габариты прямоугольной зоны генерации, в которой есть отдельные области (у верхнего края, в углу, в любой точке и т.д.), для каждой из которых задаются определённые правила: список объектов, которые могут появиться в области, их максимальное количество, а также список дочерних объектов, которые могут быть созданы рядом с родительским объектом. Система случайной генерации обеспечивает разнообразие появляющихся комнат, что делает игровой процесс интереснее.

Ещё одной важной системой является специальный инспектор для создания и настройки различных видов оружия [8]: лазерного пистолета, аннигилирующего оружия, гранатомёта, барьерной винтовки и обычного автомата. В инспекторе доступен выбор вида оружия, настройка наносимого урона, радиуса урона, дальность атаки, частота выстрелов и режим стрельбы. Оружейная система построена на механизме наследования от классов лазерного оружия и оружия, выпускающего снаряды, которые в свою очередь наследуются от базового класса оружия. Благодаря инспектору сильно упрощается создание объектов оружия, а также разноображивается взаимодействие с игровой средой и врагами.

3. Реализация

В данном разделе приведены новые игровые механики, представлено описание игрового цикла и показана архитектура добавленных классов.

3.1. Реализованные игровые механики

В ходе работы были реализованы следующие игровые механики:

- Генерация универсальных боеприпасов: при подбирании игроком они увеличивают количество патронов в запасе для любого оружия. Боеприпасы появляются в случайных местах комнат с некоторой вероятностью.
- Генерация аптечек: при подбирании игроком аптечка увеличивает количество очков здоровья. Как и боеприпасы, случайно появляется в комнатах с некоторой вероятностью.
- Создана финальная комната с временным реактором, в которой также есть электро-генераторы реактора и четыре отдельных помещения-мастерские (см. рис. 1) со своей спецификой научно-технической деятельности.
- Добавлен механизм появления финальной комнаты на случайном этаже из заданного диапазона (в пределах [-50; -20]).
- Добавлен враг-босс "Rapid Turret" (см. рис. 2): располагается в комнате с реактором. В режиме поиска пространство сканируется прицельным лучём, а при атаке луч фиксируется на цели, затем производится преследующий огонь с большой частотой.
- Добавлена механика починки электро-генераторов реактора (см.рис. 3).
- Добавлена механика, заканчивающая прохождение игры, а также меню победы.

Рис. 1: Одна из мастерских в финальной комнате

Рис. 2: Враг-босс "Rapid Turret"

- Исправлены две небольшие ошибки и оптимизирован игровой баланс.

Рис. 3: Механика починки электро-генератора

3.2. Описание игрового цикла

После реализации приведённых выше механик игра стала технически завершённой, хоть и является «сырой» в плане оптимизации, отсутствия ошибок и дополнительных механик, которые могли бы разнообразить игровой процесс. Теперь есть цельный игровой цикл, в рамках которого можно пройти игру несколько раз.

На данный момент геймплей устроен следующим образом: игрок спускается в самую нижнюю часть лабораторий, исследуя по пути случайно генерируемые помещения и усиливая возможность замедлять время при сражении с врагами, а в финальном помещении с названием "Time Reactor Center" устраняет стражников в виде скоростных турелей, либо уклоняется от их атак, а затем чинит шесть электро-генераторов, тем самым восстанавливая повреждённый реактор.

3.3. Архитектура добавленных классов

Приведена диаграмма классов на рис. 4, содержащая новые классы и классы, от которых они зависят.

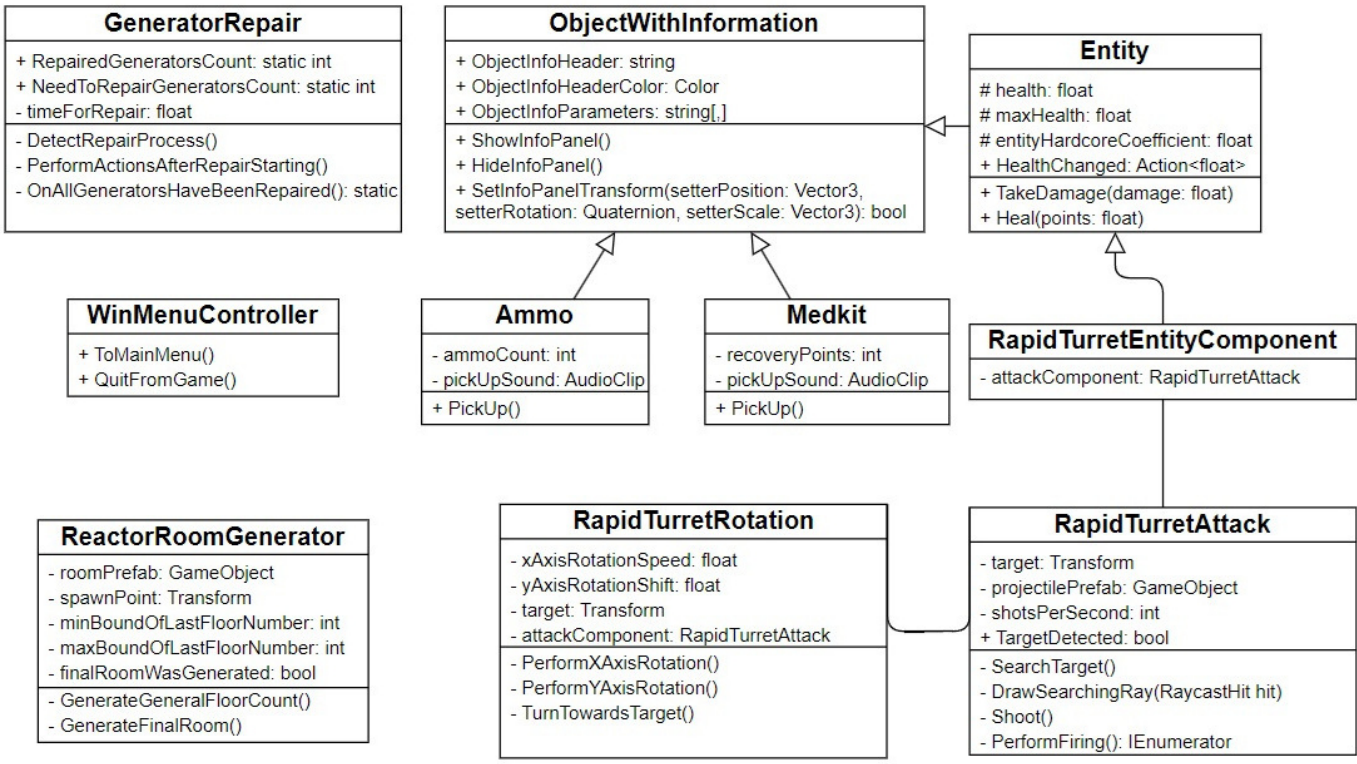

Рис. 4: Архитектура добавленных классов

- Ammo отвечает за поведение боеприпаса;
- Medkit является реализацией объекта аптечки и вместе с Ammo наследуется от ранее реализованного класса ObjectWithInformation;
- ReactorRoomGenerator генерирует номер этажа, на котором будет размещена финальная комната, а также создаёт её и располагает нужным образом;
- RapidTurretRotation отвечает за вращение турели по двум осям в обычном режиме и за преследование цели, если она была обнаружена;
- RapidTurretAttack отвечает за обнаружение цели и атаку выстрелами с большой частотой, если цель была замечена;
- RapidTurretEntityComponent отвечает за здоровье турели, отображение информации о ней и возможность её уничтожения. Класс

наследуется от ранее реализованного Entity, который в свою очередь является наследником ObjectWithInformation;

- GeneratorRepair является реализацией механики починки генераторов реактора и играет ключевую роль в возможности пройти игру, так как следит за условием победы;
- WinMenuController реализует поведение победного меню, из которого можно выйти в главное меню, либо покинуть игру.

4. Апробация

Среди методик оценки SUS, GUESS и ENJOY для проведения апробации была выбрана методика GUESS (Game User Experience Satisfaction Scale) [1], так как она заточена именно под оценку видеоигр и пользовательского опыта в них. GUESS — это комплексная шкала с девятью подшкалами для оценки пользовательского опыта, полученного в видеоиграх. GUESS была разработана и проверена на основе оценок более 450 уникальных игр различных популярных жанров. Таким образом, данную шкалу можно применять ко многим типам видеоигр в отрасли разработки видеоигр в качестве способа оценки аспектов игры, которые способствуют удовлетворению пользователей.

Стандартный опросник GUESS состоит из 55 вопросов, ответ на каждый вопрос соответствует 7-балльной шкале Лайкерта от “Категорически не согласен” до “Полностью согласен”. Однако при разработке видеоигр обычно нужна непрерывная обратная связь, так как часто внедряется новая функциональность, которую нужно тестировать, поэтому постоянно отвечать на 55 вопросов может быть утомительно.

Для упрощения процесса авторами методики была придумана модификация GUESS-18, состоящая лишь из 18 самых важных вопросов. В данной вариации в каждой из девяти подшкал выделено по два вопроса, на которые нужно ответить.

Результат оценки подсчитывается следующим образом: ответ “Категорически не согласен” даёт 1 балл за текущий вопрос, а ответ “Полностью согласен” — 7 баллов. Для каждой подшкалы берётся среднее арифметическое, затем все девять значений складываются. 1 из 18 вопросов должен быть отрицательным по отношению к остальным, например, “Мне скучно во время игры”. Таким образом, можно получить от 9 до 63 баллов включительно, где 63 означает, что видеоигра набрала максимальные оценки по всем параметрам.

Апробацию видеоигры “Time Reactor” прошло 7 человек, в итоге был получен усреднённый балл, равный 45 единицам. Учитывая, что балл, равный 36 единицам, является серединным для данной шкалы, можно

сделать вывод, что итоговая оценка оказалась чуть выше серединной.

В ходе апробации были собраны комментарии о процессе игры. Среди них можно выделить следующие:

- “Ниче не понял но игру прошел. Понравилось строить домики кубострелом. Не понравились лаги и то что ниче не понял.”;
- “Очень жестко лагает при загрузке комнат, особенно в последней (нет видеокарты)”;
- “Лаборатория офис завод, прям как в универе”;
- “Хороший эффект уничтожения стен”;
- “Самое имбовое - гранатомет (РПГ)”
- “Игра простенькая конечно, не спорю, но весело. И вообще, свечение мне нравится кстати.”
- “Мне больше всего нравится атмосфера Sci-Fi и то, как выглядят локации. Пока что в игре много багов и нет оптимизации, но считаю, что у игры есть потенциал”.

Заключение

В рамках данной учебной практики были получены следующие результаты:

- выполнен обзор Unity Engine и Unity Asset Store;
- выполнен обзор реализованной ранее функциональности;
- реализованы механики, необходимые для возможности прохождения игры: генерация боеприпасов и аптечек, механизм появления финальной комнаты на одном из этажей, а также сама комната; добавлены враг-босс, препятствующий починке генераторов реактора, механика завершения игры и победное меню;
- проведена апробация и получены комментарии об игровом опыте от игроков.

Ссылка на исходный код — https://github.com/RuslanBeresnev/Time-Reactor-Game (дата обращения: 08.06.2024)

Список литературы

[1] RUX Lab. The GUESS. — URL: https://www.ruxresearch.com/the-guess (дата обращения: 8 июня 2024 г.).

[2] SteamDB. Popular Game Publishers on Steam. — URL: https://steamdb.info/publishers/ (дата обращения: 7 июня 2024 г.).

[3] Unity Documentation. Rendering paths in the Built-in Render Pipeline. — URL: https://docs.unity3d.com/ru/2021.1/Manual/RenderingPaths.html (дата обращения: 8 июня 2024 г.).

[4] Unity Technologies. Unity Asset Store. — URL: https://assetstore.unity.com/ (дата обращения: 8 июня 2024 г.).

[5] Unity Technologies. Unity User Manual 2022.3 (LTS). — URL: https://docs.unity3d.com/Manual/index.html (дата обращения: 8 июня 2024 г.).

[6] Unity Technologies. Вперед к разработке. — URL: https://unity.com/ru (дата обращения: 7 июня 2024 г.).

[7] Савельева Полина. Реализация процедурной генерации интерьера комнат в 3D-игре на Unity. — 2024. — URL: https://github.com/PolinaSavelyeva/Articles/tree/main/2023/interiors_PCG/interiors_PCG_report (дата обращения: 8 июня 2024 г.).

[8] Яворовский Александр. Реализация системы оружейных механик в игре Time Reactor. — 2024. — URL: https://se.math.spbu.ru/thesis/texts/Javorovskij_Aleksandr_Artemovich_Autumn_practice_2nd_year_2023_text.pdf (дата обращения: 8 июня 2024 г.).