**1. Введение**

Тема проекта: Beat Em Up игра

Цель проекта – изучить процесс создания игр на игровом движке unity .

Функциональность:

* реализация паттерна statemachine для оптимизации работы с игровыми существами
* создания игрового персонажа, реализующего данный паттерн с состояниями walk, idle, hitten, attak, и т.д. (в том числе реализовать логику самих состояний)
* создание противника по подобию игрового персонажа
* создание простого интеллекта для противника
* реализация системы здоровья, и обработка получения урона
* “спаун” противников

**2. Выбор технологий**

|  |  |  |
| --- | --- | --- |
| Компонент | Технология | Описание |
| Язык программирования | C# | Современный язык с мощной экосистемой |
| Игровой движок | Unity 6 | Мощное и популярное решения в котором многие члены команды уже имеют опыт работы |
| IDE | Visual Studio / Rider | Для разработки |
| Создание анимаций | Asprite | Ускоспециализированная программа, отлично подходящая для создания pixel-art картинок и анимаций |
| Документация | Markdown, Draw.io, PlantUML | Для технического описания и UML |

**3. Исследование: Как создать игру в жанре BeatEmUp?**

В ходе изучения жанра было очень полезно видео(<https://www.youtube.com/watch?v=IW01-RZv06g>)

Было выяснено, что каноничным для жанра является вид с боку, псевдо объём в плоском пространстве, разбиение действий персонажей на состояния, управление толпами противников, набор умений типа: передвижение, нанесение урона, отталкивание.

Мы узнали как работать с игровым движком Unity (в частности детально изучили внутренние классы, такие как: MonoBehavor, Transform, UnityEvent и т.п.) И поняли как другие люди реализовывали наши задумки

**4. Пошаговое создание игры в жанре Beat em Up**

**Шаг 1: Создание StateMachine**

1. Создайте абстрактный класс StateManager, в котором будет реализована механика смены, активации и работы отдельных состояний

2. Создайте абстрактный класс State, в котором будут методы для активации, выхода и работы конкретного состояния

3. реализуйте эти классы, добавив нужные механики(например для передвижения проигрывание анимации и изменение позиции игрока)

4. Создайте анимации для каждого состояния и логику перехода между ними

**Шаг 2: Создайте игрока**

1. Реализуйте StateMachine для конкретного игрового персонажа
2. Добавьте ему анимации
3. Создайте систему здоровья (я это сделал через отдельный скрипт, а связал состояния и это скрипт через интерфейс IDamagable)
4. Реализуйте нанесение урона через AnimationEvent

**Шаг 3: Создайте противника**

1. По тому же принципу реализуйте StateMachine, добавте анимации, свяжите с системой здоровья и добавьте AnimationEvent
2. Создайте отдельный скрипт, управляющий переходами между состояниями, пусть он отвечает за логику и последовательность действий противника(можно назвать это его мозгами)

**Шаг 4: Настройте всё в редакторе**

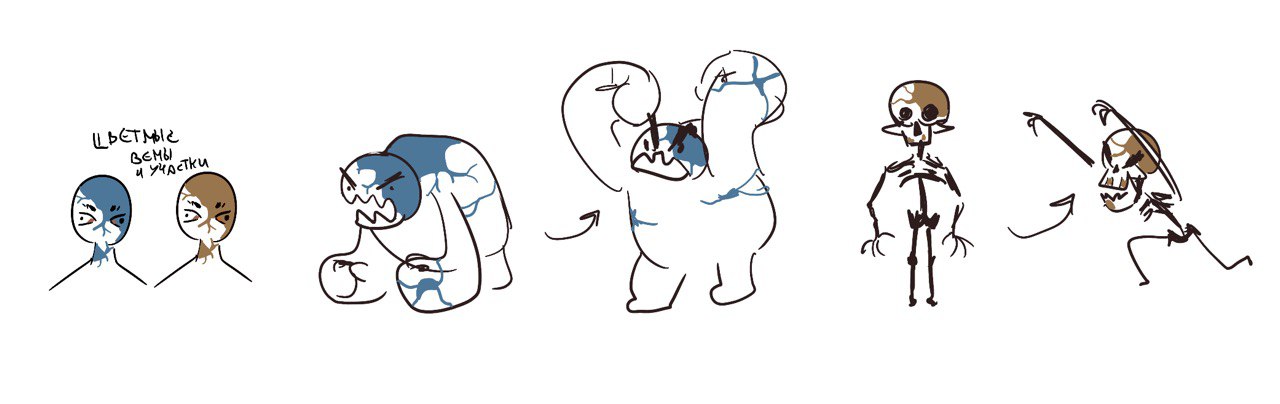
1. В редакторе заполните все нужные поля
2. Создайте несколько ScriptableObject для упрощения работы

**Итог:**

На этом этапе у нас есть базовая игра с основными механиками и одним противником.

Мы упустили большую часть этапов разработки, рассказав лишь обо всём в общем, потому что работа оказалась слишком объёмной, если интересно посмотреть код, можете зайти в папку scr/Assets/sctipts

**5. Иллюстрации**

Концепт арты:

Анимации персонажа

**6. Модификация проекта: Spown противников**

**Цель модификации**

Добавить интереса в игру, появление новых противников, хорошо скажется на реиграбельности

**Техническая реализация**

Создали отдельный скрипт Spowner, который содержит точки появления противников и сам префаб противника, через коррутину раз в некоторое время появляется противник в случайной точке

**Идеи для будущей доработки**

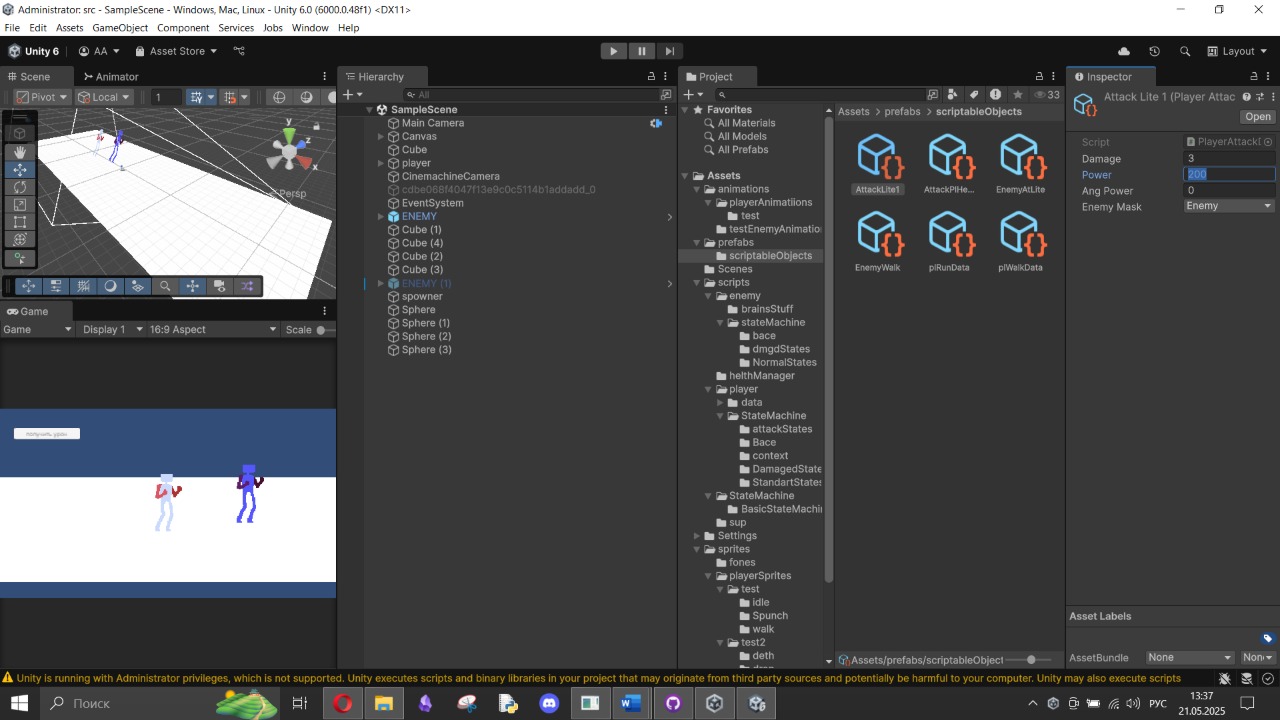
* Добавить новых противников
* Предметы окружение
* Музыку
* Звуки
* Улучшить анимации
* Исправить баги

**Демонстрация проекта:**

**Видео**



**Скриншоты:**

****

