

Диздок (дизайн-документ) для игры

Курсова робота: за межами реальності

(рабочее название, может поменяться)

Общие правила диздока:

- **зелёное выделение** - аспект сделан
- **жёлтое** - в процессе
- **ПЕРЕМЕННАЯ** - ссылается на значение из раздела "Переменные"
- **Красный текст** - комментарий
- не удалять и не убирать! Если надо, то просто уменьшаем текст как тут

TODO:

- согласовать общую концепцию игры
- разработать и спланировать сюжет
- определить, сколько мини-игр будет
- создать приватный GitHub репозиторий
- распределить мини-игры и приступить к работе
- протестировать мини-игры
- сделать графику для мини-игр
- сделать кат-сцены
- SFX и музыка
- озвучка
- настройки, меню паузы, главное меню, система сохранений
- **документация (самое ужасное)**
- ГОТОВО

Внимание! Тут и далее Алексей и Борис - абсолютно ВЫМЫШЛЕННЫЕ персонажи. Все совпадения с реальностью - случайны.

Что это за игра?

Квест, состоящий из мини-игр разных жанров - от платформера до головоломок и шутеров. Игра предоставит разнообразный геймплей вместе с увлекательным, смешным и линейным сюжетом.

Синописис

Борис и Алексей делает курсовую работу и побеждают трудности, которые появляются во время разработки

Фишка сюжета

Абсурдный юмор по типу "Рика и Морти". Ситуации по типу "Инопланетяне похитили флешку с курсовой поэтому мы взорвали Марс" - норма для сюжета.

Дополнительно

Нужно использовать визуальные гэги на полную

Правила оформления

Писать надо чётко и понятно, без двойных трактовок. Красота диздока не так важна, как его функциональность.

В разделе “Мини-игры” содержатся описания мини-игр, для каждой мини-игры нужно выделить 2 пункта:

- 1) Роль в сюжете
- 2) Геймплей
- 3) Как это будет работать

Эти пункты дают возможность ответить на вопросы “Уместна ли эта игра в сюжете?”, “Как она будет связана с другими мини-играми?”, “Будет ли в неё интересно играть?” и “Как она будет реализована в движке?”

Переменные

КОЛИЧЕСТВО_ИГР - 7

РАННЕЕ_НАКОПЛЕНИЕ_БОНУСА - 4 секунды

Концепт-арты



Референсы

Геймплей:

<https://youtu.be/xeYlQbZjCH0>

The hex - много жанров в одной игре. Мини-игры, объединённые сюжетом - то, что мы делаем.

https://youtu.be/i8uOFH_IP08

Ералаш-ленд - прикольный артефакт из эпохи квестов.

Визуал:

https://youtu.be/OY_0RzzciO0

Стиль анимации подходит нашей игре.

Сюжет:

Рик и морти - прекрасный пример юмора и абсурда. Именно тот дух, который нам нужен.

Время приключений - отличный юмор.

Финис и ферб - неплохой пример юмора для всех поколений.

Идеи для сюжета:

КОЛИЧЕСТВО_ИГР мини-игр - для каждой будет по 1-2 недели, успеем достаточно проработать

3 игры за одного персонажа, 2 игры за другого, 2 общие

Идеи для мини-игр:

- * раннер
- * Shoot`em up
- * головоломка про программирование - соединить куски кода в правильной последовательности что-бы получить нужный результат
- * квест

Игра начинается со сна - кликера, где Алексей становится миллиардером

Игра начинается со сна “Университет Остроградка: легенда о курсовой работе” (**первая мини-игра**). Кликер, где игрок прокачивает двух героев (Бориса - мага и Алексея - воина). Они зачищают подземелье от монстров и доходят до артефакта - “Священная курсовая работа”.

Внезапно Алексей просыпается (ему звонит староста, напоминает про курсач), заводит свою машину и начинается **вторая мини-игра** - раннер на машине в стиле Subway serfers. В ходе игры пару раз меняется вектор движения - сначала слева-направо, потом снизу-вверх и обратно. А в один момент машина Алексея улетает в космос и начинается бодрый Shoot`em up (**третья мини-игра**).

Игра переключается на Бориса. **Четвёртая мини-игра** - Борис ловит оценки. Потом начинается **пятая мини-игра** - Борис так увлёкся написанием курсовой работы, что заснул за компьютером, а на утро понял, что он написал телепортатор. Из-за неаккуратности Борис активирует его и переносится на

необитаемый остров. Там он должен есть падающие кокосы и отбиваться от крабов. Вид сверху. Потом его спасает Алексей на его машине-космическом корабле и они летят в остроградку.

Будет прикольно, если я сделаю все мини-игры для персонажа Алексея, а ты сделаешь мини-игры для персонажа Бориса.

Алексей и Борис в универе. Начинается **шестая мини-игра** - серия головоломок.

Первый этап - отбор идей.

Второй этап - головоломка с программированием (правильно соединить блоки кода)

Третий этап - оформление документации (ну хз, печатание слов на время? Пропускают вперёд только когда набрал нужный процент точности)

И вот Алексей и Борис несут флешку с курсовой, но её похищают инопланетяне (или пираты?) и начинается платформер или Shoot'em up с видом сверху. Или головоломка с механикой переключения персонажей.

Начинается платформер-головоломка (**седьмая мини-игра**), где Борис и Алексей связаны и игрок должен одновременно управлять обоими персонажами что-бы пройти дальше. Или позвать друга.

В двух последних мини-играх надо показать командный дух, что мы работаем вместе.

В конце они сдают курсач и получают самый высокий балл - Harry end! Другой концовки в игре быть не может.

Мысли по геймплею и дизайну

Сделать главное меню в виде рабочего стола

Анимация выбора персонажа в стиле гта

Усложнённые режимы игр

Общие технические особенности

Использование паттернов проектирования Singltone, Event Bus (через GlobalFlags).

МИНИ-ИГРЫ

Хоррор-point&click-квест (работает Алексей, документацию оформил Борис) scary but silly

Ключевые моменты

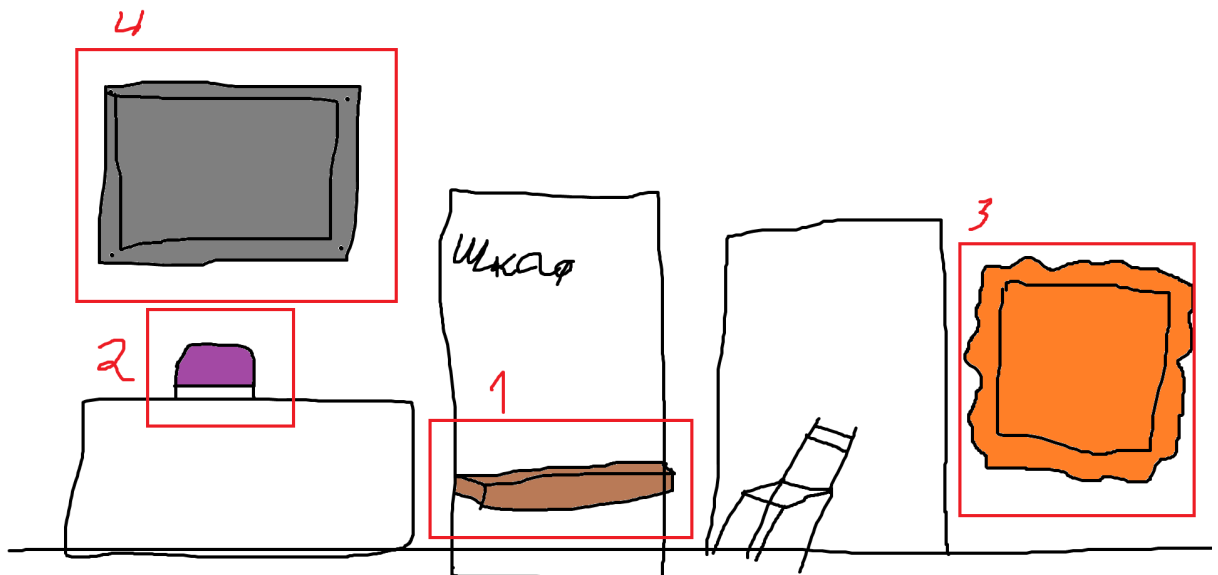
- +- 5 минут геймплея - 4-5 сегментов
- таймер для каждого сегмента

Сюжет

Борис и Костя играют в шахматы в рекреации универа. Борис пытается что-то сказать, НО фигуры вдруг оживают и начинают гоняться з ними. Борис убегает по коридору в аудиторию, баррикадируется там.

ГОЛОВОЛОМКА №1

40 секунд



В приоткрытой полке шкафа взять ключ - ключом открыть шкатулку -> в шкатулке взять другой ключ и открыть им сейф за картинок -> взять отвёртку с сейфа и открутить вентиляцию

ГОЛОВОЛОМКА №2

60 секунд



Борис попадаєт в хімічну лабораторію.

Находить в приоткрытом шкафчике записку:

“Обережно! В ЖОДНОМУ РАЗІ не можна робити таку послідовність дій:

- встановити перший перемикач на 3
- встановити другий перемикач на 2
- встановити третій перемикач на 4
- активувати головний перемикач.

В результаті буде створено небезпечну рідину, яка навіть може ПЛАВИТИ ДЕРЕВИНУ”

Ігрок устанавлюєт это в аппарате справа (управление переключателями - правый щелчок - +1, левый - -1).

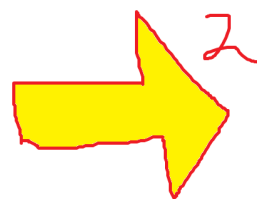
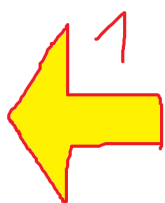
Берёт смесь и использует её на двери

ГОЛОВОЛОМКА №3

80 секунд

Реклама КРНУ со стёбом

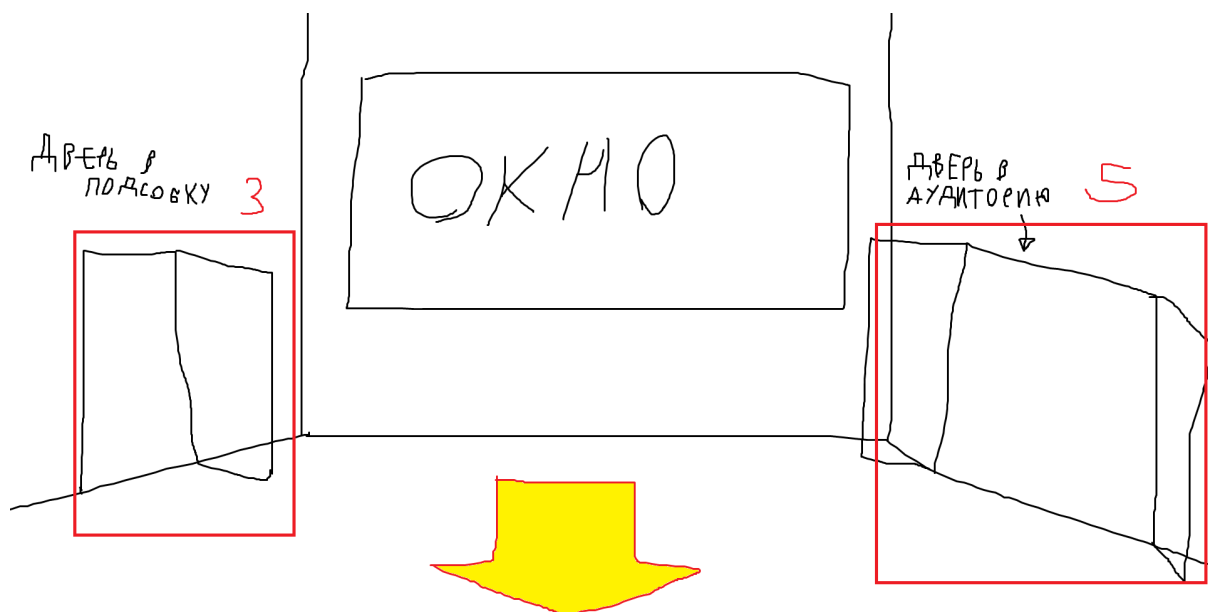
(и с первой подсказной)



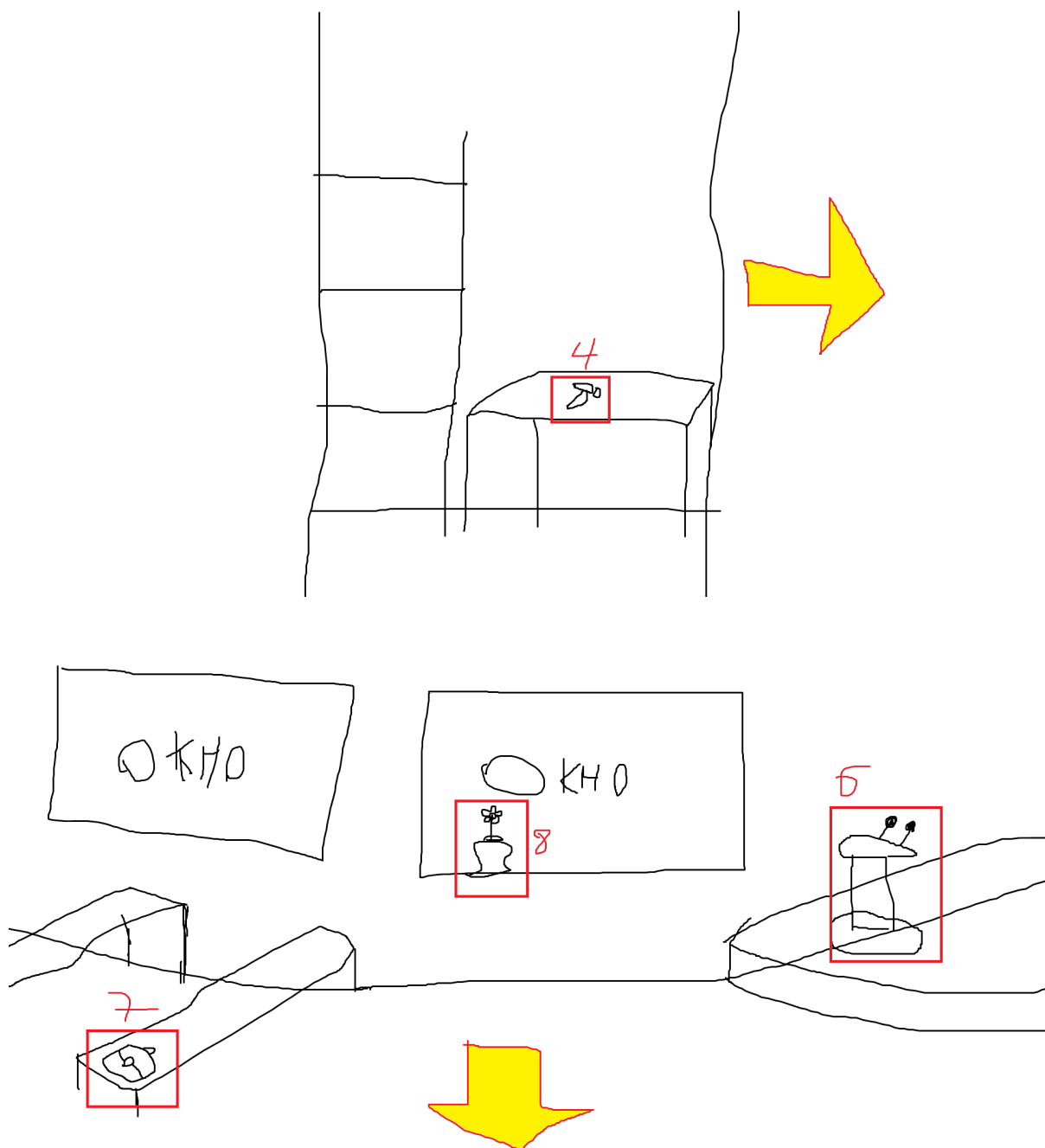
Игрок имеет выбор пойти налево или направо.



Слева - железная дверь с кодовым замком.



Справа - две двери - в подсобку и аудиторию.



Код подбирается так:

Первая цифра - текст на рекламном плакате КРНУ гласит:

“Наша **ПЕРША** ініціатива по виборам в студраду - ми є номер **ОДИН** серед університетів по рівню демократичності!”

Вторая цифра - в той штуке под номером 6 будет отрывок шпаргалки выступающего

“... саме тому **ДРУГА** версія системи проектування CadMat має **ДЕВ'ЯТЬ** панелей керування матрицями...”

Третья цифра - разбиваем молотком из кладовки вазу, находим записку:

“**ТРЕТЯ** ознака подібності трикутників стверджує, що **ДВА** трикутники є подібними за умови попарної пропорційності їхніх сторін”

Четвёртая цифра - в портфеле на парте находим записку:

“**ЧЕТВЕРТА** програма, із якою я познайомився, був фотошоп. Це - номер **ОДИН** у сфері фотообробки.”

Пароль - 1921 (год основания КрНУ).

Подсказки - это не отдельные предметы инвентаря, а просто объекты, некоторые из них при нажатии увеличиваются что-бы было удобней читать.

ГОЛОВОЛОМКА №4

30 секунд

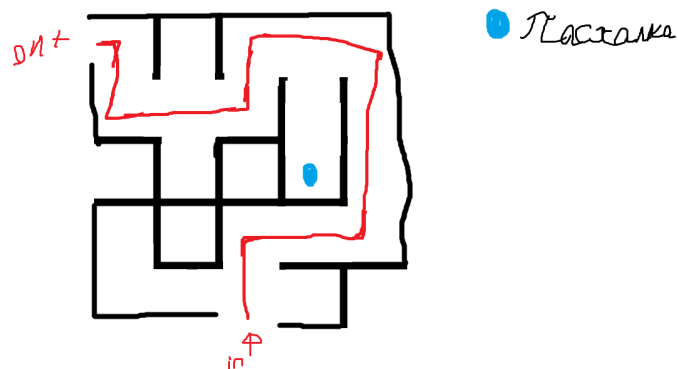
ОТМЕНА

Лабиринт с видом от первого лица.

План лабиринта

Лабиринт

(представь что от первого лица)



РАННЕР (Работает Борис)

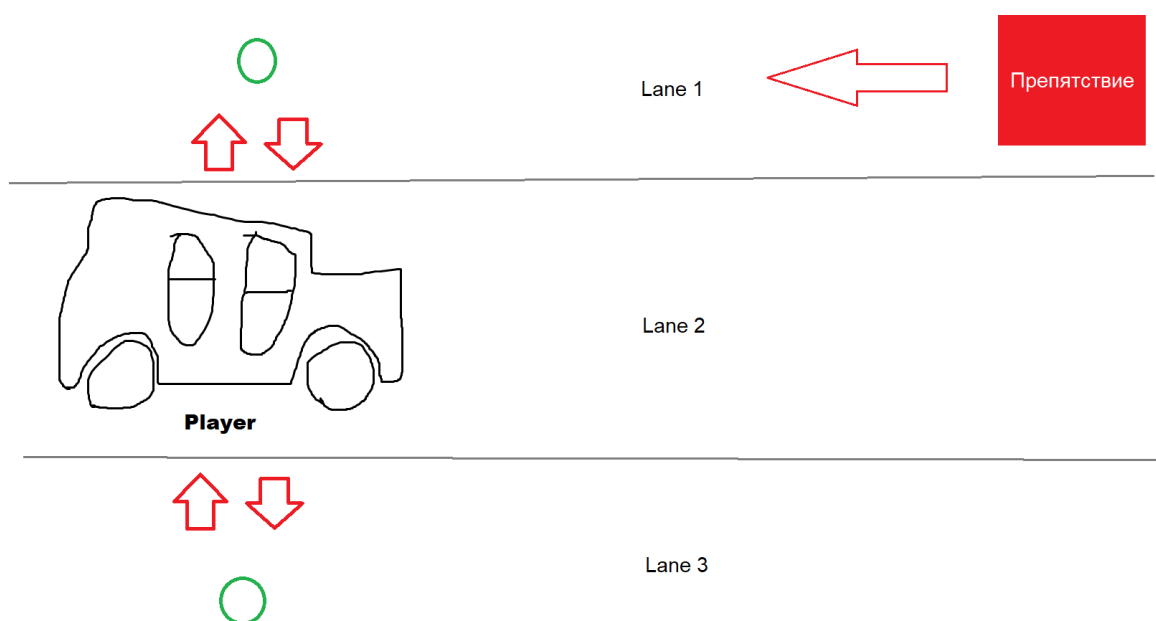
Роль в сюжете

Объясняет ту часть истории, как Алексей попадает в универ.

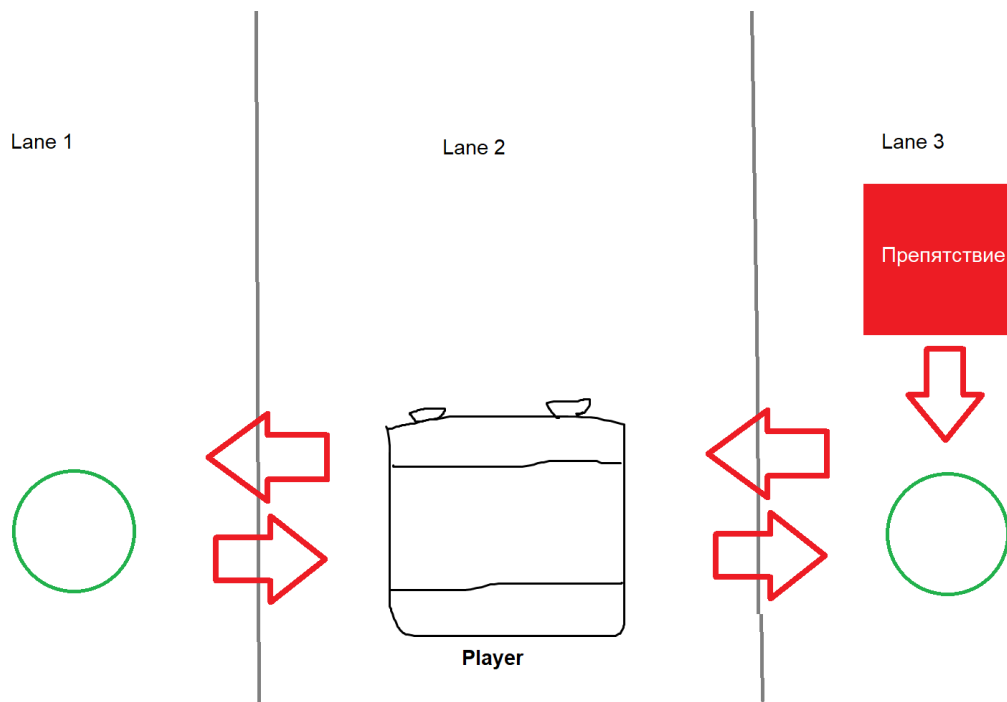
Геймплей

Игрок управляет машиной с ограниченным управлением - между тремя точками, что-бы уворачиваться от разных препятствий (пешеходы, машины, разный мусор). В качестве баланса сложности каждые **РАННЕР_НАКОПЛЕНИЕ_БОНУСА** секунд игрок может использовать супер-способность - прыжок.

Для того, что-бы игрок успевал отреагировать на препятствия, перед их появлением будет показываться индикация знака опасности.

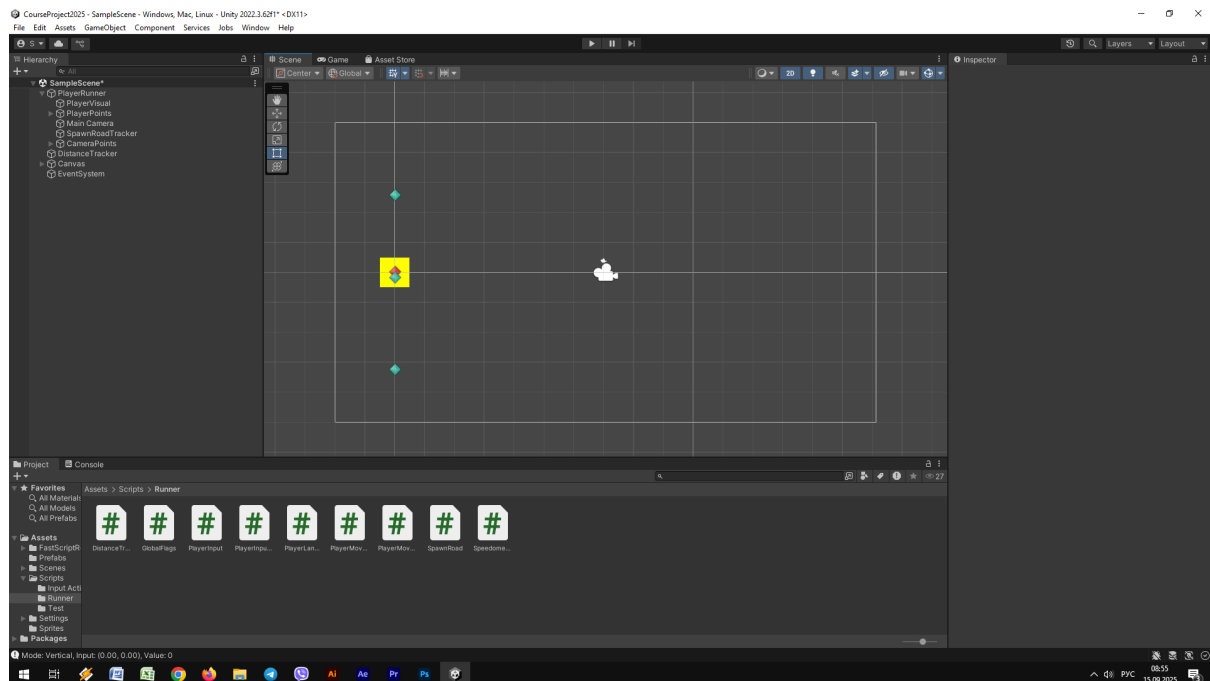


Первый этап - несложный, скорость разгона - до 50 км/час. Вектор движения - слева-направо.



Второй этап - усложненный, разгон до 100 км/час, вектор движения снизу-вверх.

Третий этап - самый сложный, разгон до 150 км/час, вектор движения слева-направо.



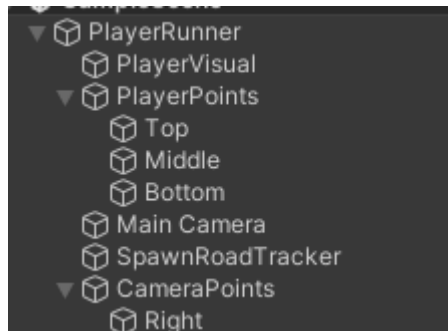
Ранний прототип раннера

Триггер смены этапов - пройденное расстояние.

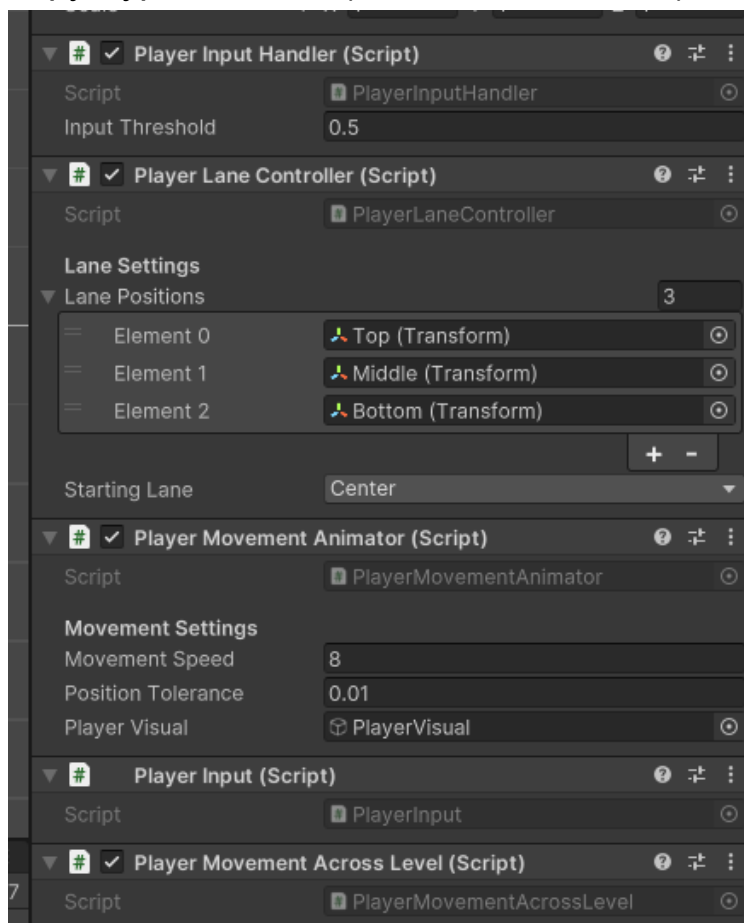
Затем кат-сцена - Алексей видит, что не успевает на пару, и машина превращается в космический корабль. Начинается shoot`em up с видом сверху в космосе.

Как это будет работать

Для первых трёх этапов игрок будет управлять объектом PlayerRunner



Структура объекта (может поменяться)



Скрипты объекта PlayerRunner

В логике скриптов чётко прослеживается разделение визуала, внутренней логики и обработки ввода пользователя.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.InputSystem;

Скрипт Unity (1 ссылка на ресурсы) | Ссылки: 3
public class PlayerInput : MonoBehaviour
{
    // Общая система ввода для игрока
    // Использует Unity Input System

    private PlayerInputAction _playerInputActions;

    Сообщение Unity | Ссылки: 0
    void Awake()
    {
        _playerInputActions = new PlayerInputAction();
        _playerInputActions.Enable();
    }

    Ссылки: 3
    public Vector2 GetMovementVector()
    {
        return _playerInputActions.Player.Move.ReadValue<Vector2>().normalized;
    }
}
```

Player Input - универсальный скрипт, который использует Input System для обработки нажатий. Назначение клавиш можно посмотреть в Scripts/Input Actions/PlayerInputAction. Эту систему желательно использовать и для других мини-игр т.к. это удобно.

Player Input Handler - специфический скрипт для раннера, который позволяет переопределять вектор движения.

```
Ссылки: 2
private float GetInputValueByMode(Vector2 movementVector)
{
    return CurrentInputMode switch
    {
        InputMode.Horizontal => movementVector.x,
        InputMode.Vertical => -movementVector.y,
        InputMode.None => 0f,
        _ => 0f
    };
}
```

Distance Tracker - скрипт для набора скорости, реализует паттерн singleton.

```
Ссылка: 1 | Ссылка на ресурсы | Ссылка: 6
public class DistanceTracker : MonoBehaviour
{
    Ссылка: 5
    public float CurrentSpeed { get; private set; }
    private float currentAcceleration = 12f;
    public float CurrentDistance = 0f;
    private const float maxSpeed = 80f;

    private static DistanceTracker instance;

    Сообщение Unity | Ссылка: 0
    private void Awake()
    {
        if (instance == null)
        {
            instance = this;
        }
    }

    Ссылка: 2
    public static DistanceTracker GetInstance()
    {
        return instance;
    }

    Сообщение Unity | Ссылка: 0
    private void Update()
    {
        if (CurrentSpeed < maxSpeed) CurrentSpeed += currentAcceleration * Time.deltaTime;
        CurrentDistance += CurrentSpeed * Time.deltaTime;
    }
}
```

Global flags - скрипт, который реализует функционал системы глобальных флагов через HashSet. Любой скрипт может обратиться к нему - либо что-бы добавить флаг, либо убрать, либо что-бы проверить состояние флага. Также можно подписаться на изменение состояние определённого флага. Рекомендуется использовать во всех играх.

```

using System;
using System.Collections.Generic;
using UnityEngine;

Ссылка: 0
public static class GlobalFlags
{
    private static HashSet<string> flags = new HashSet<string>();
    public static Action<string, bool> onFlagChanged;

    Ссылка: 1
    public static void SetFlag(string flagName)
    {
        flags.Add(flagName);
        onFlagChanged?.Invoke(flagName, true);
    }

    Ссылка: 1
    public static bool GetFlag(string flagName)
    {
        return flags.Contains(flagName);
    }

    Ссылка: 1
    public static void ClearFlag(string flagName)
    {
        flags.Remove(flagName);
        onFlagChanged?.Invoke(flagName, false);
    }

    Ссылка: 0
    public static void ToggleFlag(string flagName)
    {
        if (GetFlag(flagName)) ClearFlag(flagName);
        else SetFlag(flagName);
    }

    Ссылка: 0
    public static class Flags
    {
        public const string PLAYER_CAN_MOVE = "player_can_move";
        public const string GAME_STARTED = "game_started";
        public const string LEVEL_COMPLETED = "level_completed";
        public const string CAR_TURNING = "car_turning";
        public const string INPUT_LOCKED = "input_locked";
    }
}

```