

Проектная работа

2D-Игры

Куда загружаем проект

<https://goo.gl/forms/sEXAdLRaRvZtY3hu2>

Требования к работе

- Уровень должен быть "упакован" в архив с расширением .unitypackage
- Этот архив должен называться именем и фамилией студента с указанием города.
Например: PetrovVladimir(SPБ).unitypackage
- После загрузки файлов и отправки формы данные, связанные с Вашим аккаунтом Google (имя и фотография), будут записаны.

Информация о результатах проектной работы

- 15.05.2018 - в каждом сообществе Юниум в Вконтакте будет опубликована запись об окончании приема работ.
- С 2.05.2018 по 17.05.2018 команда жюри оценивает работы студентов.
- 20.05.2018 будут подведены итоги проектной работы - итоги будут размещены в сообществах Юниум, а также в чате компьютерных курсов.
- До 25.05.2018 работы будут опубликованы в приложении, об этом также будет сообщено в сообществах Юниум и в чате.

Критерии оценки

1. Аккуратность и внимание к деталям - объекты в уровне расставлены ровно, не перекрывают друг друга и прочее - от 0 до 10 баллов.
2. Проходимость - уровень возможно пройти, он не слишком сложный, но и не слишком простой - от 0 до 10 баллов.
3. Интерес - субъективная оценка участника команды жюри - от 0 до 10.

Где скачать материалы

<https://drive.google.com/open?id=1UTeUQbsajGu7e9YJjSdtncAtZenkYGYb>

Задача: Создать 2D-игру по типу T-Rex из Google Chrome

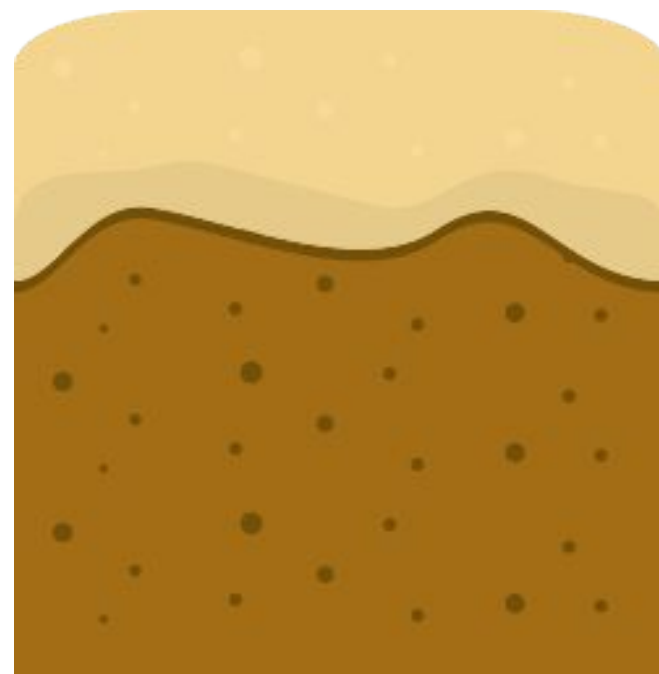
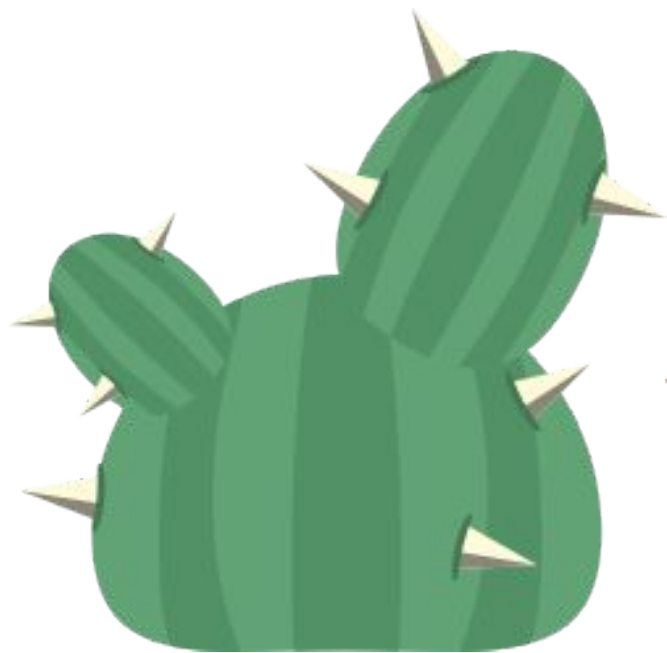


Unable to connect to the Internet

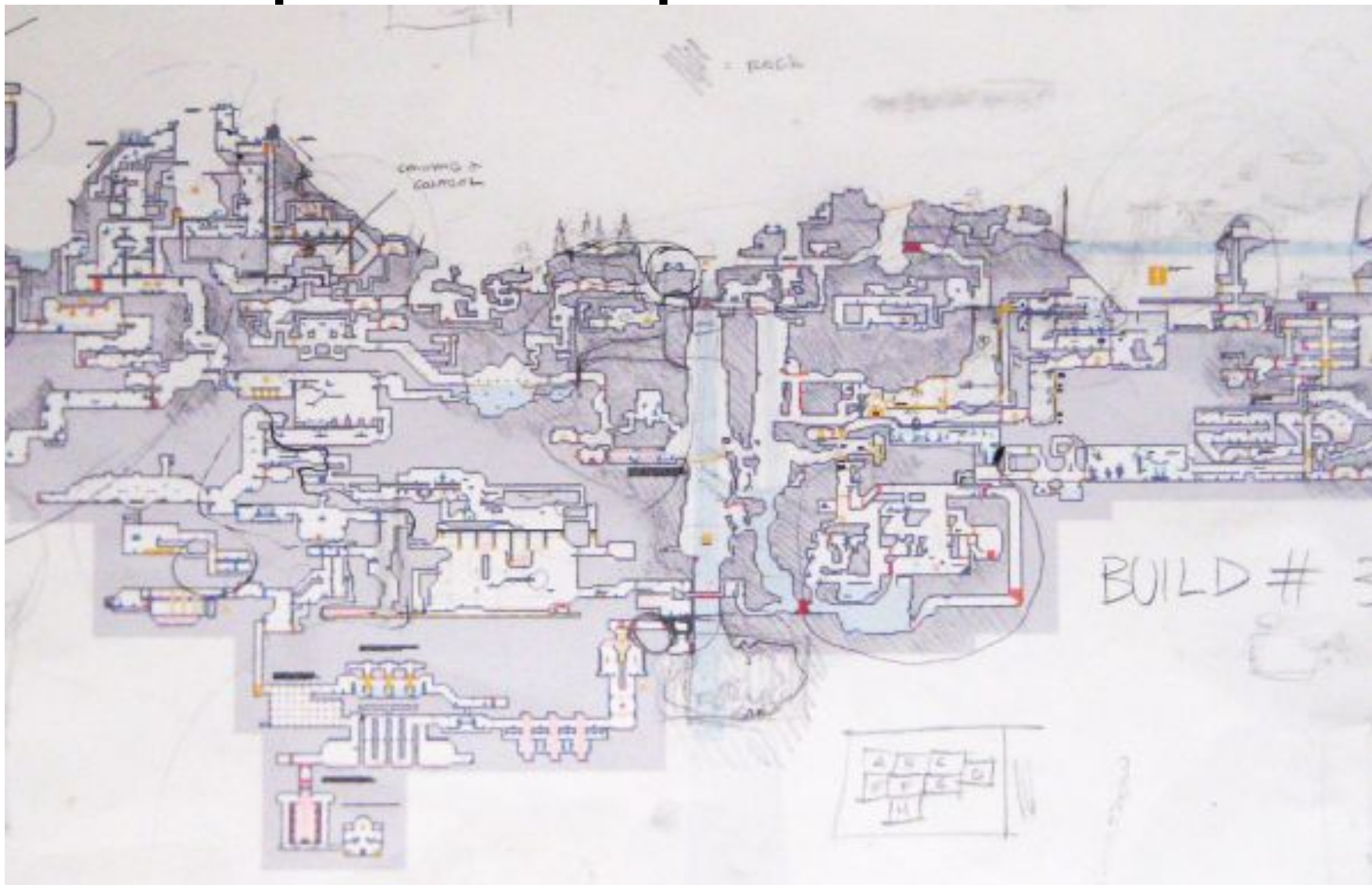
Google Chrome can't display the webpage because your computer isn't connected to the Internet.

ERR_INTERNET_DISCONNECTED

Шахтер-старатель



Часть 0. Прототипирование



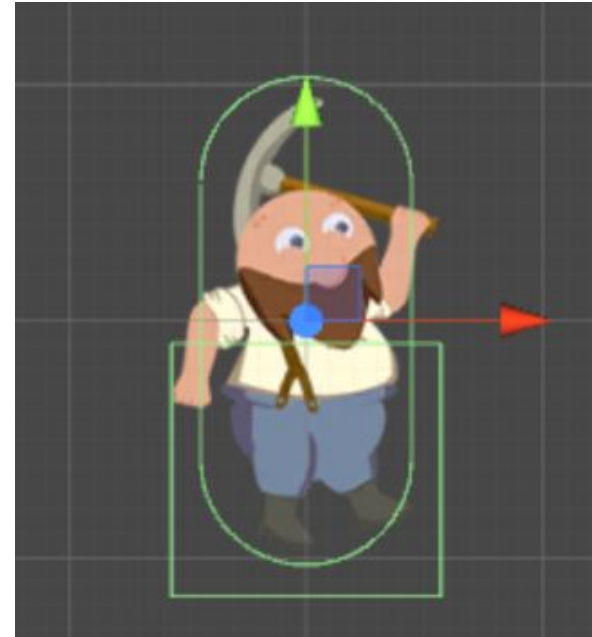
Часть 1. Игрок и платформа

- **Игрок (PlayerControllerAM.cs)**

1. Создаем пустой объект (Create Empty) и переименовываем его в “Player”
2. Добавляем объекту Player объекты: Camera и Sprite
3. Добавляем объекту следующие компоненты:
 1. Box Collider 2D (isTrigger = true);
 2. Capsule Collider 2D;
 3. Rigidbody 2D (Constraints - Freeze Rotation Z = true);
 4. Constant Force 2D;
 5. Player Controller AM

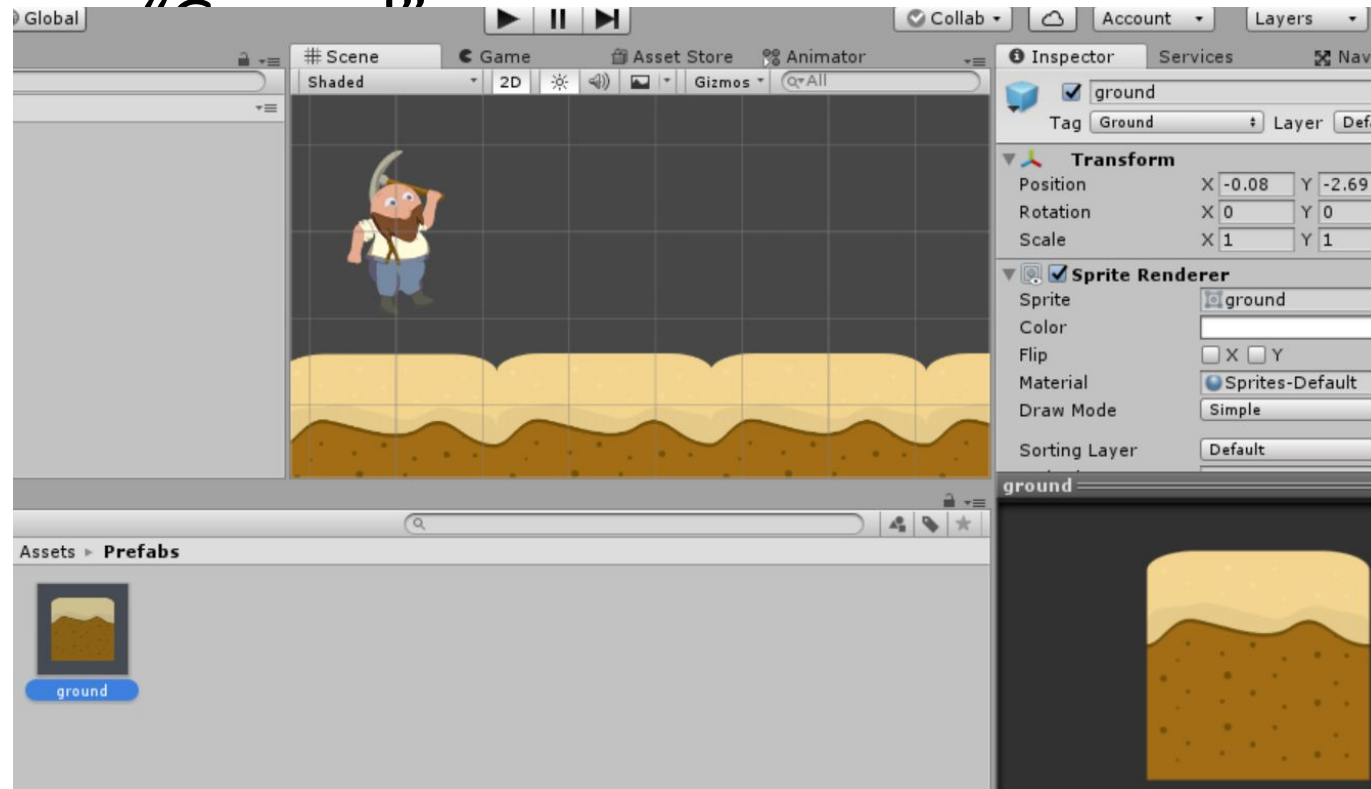
Часть 1. Игрок и платформа

Объекту Sprite нужно добавить Sprite Renderer и анимировать его используя изображения из архива. После настройки Игрок в Сцене будет выглядеть примерно вот так:



Часть 1. Игрок и платформа

- Далее необходимо сделать платформу, сохранить ее в prefab и собрать из платформ уровень по уже нарисованному прототипу. У платформы обязательно должен быть Box Collider 2D и



Часть 1. Игрок и платформа

Параметры игрока

Jump Force - сила, с которой игрок прыгает вверх;

Start Speed - стартовая скорость;

Max Speed - максимальная скорость, с которой может двигаться игрок;

Speed Step - шаг изменения скорости.

Если кликнуть мышкой в левой стороне экрана, то скорость (Current Speed) уменьшается на значение Speed Step, но никогда не будет меньше Start Speed.

Если кликнуть мышкой в правой стороне экрана, то скорость (Current Speed) увеличивается на значение Speed Step, но никогда не будет больше Max Speed.

Тестирование! По критериям

1. Аккуратность и внимание к деталям - объекты в уровне расставлены ровно, не перекрывают друг друга и прочее - от 0 до 10 баллов.
2. Проходимость - уровень возможно пройти, он не слишком сложный, но и не слишком простой - от 0 до 10 баллов.
3. Интерес - субъективная оценка участника команды жюри - от 0 до 10.

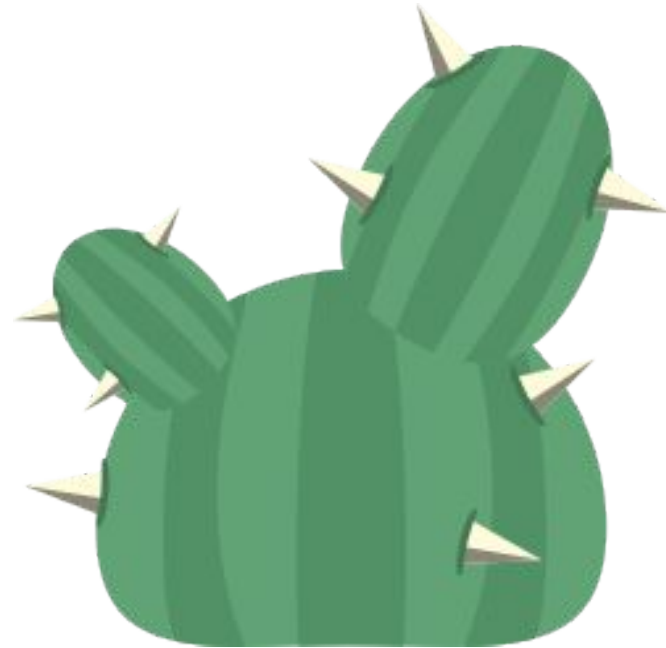
Где скачать материалы

https://drive.google.com/open?id=1C7OBRqU0vgJIjwDBtWkJ_HVbD4WdWAtU

Часть 2. Ловушки и монеты

Настройка ловушки.

Чтобы ловушка работала, необходимо выбрать подходящее изображение и добавить объекту Box Collider 2D (`isTrigger = true`), а также компонент `OnTriggerSceneLoader.cs`, в нем нужно указать название сцены, в которой размещена ловушка.



Часть 2. Ловушки и монеты

Настройка монеты.

Чтобы монета работала, необходимо выбрать изображение и добавить объекту Circle Collider 2D (isTrigger = true), а также компонент Gold.cs. Также Игроку необходимо добавить компонент GoldKeeper.cs - этот компонент считает монетки.



Где скачать материалы

<https://drive.google.com/open?id=12lkJgbX-EYGC9oKEdH6xc790rTOjvIwT>

Часть 3. Интерфейс и подача работы на конкурс

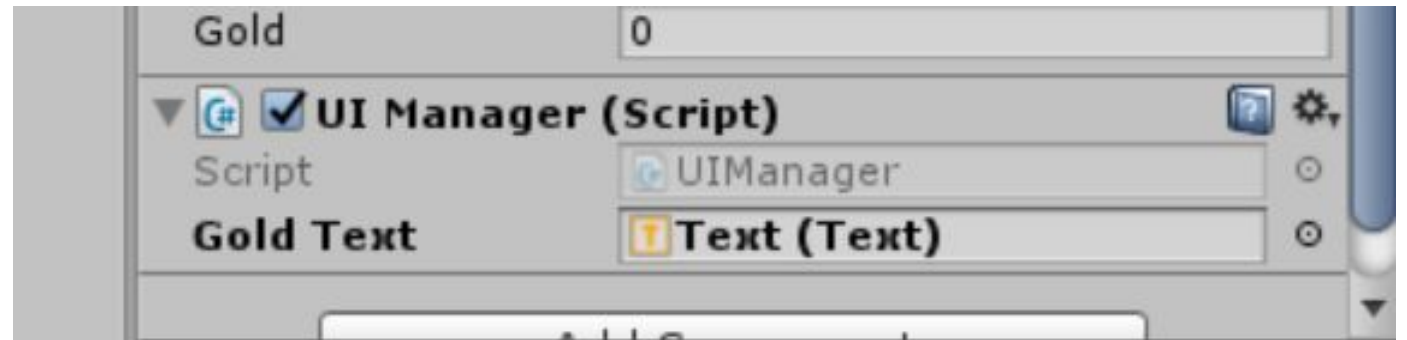
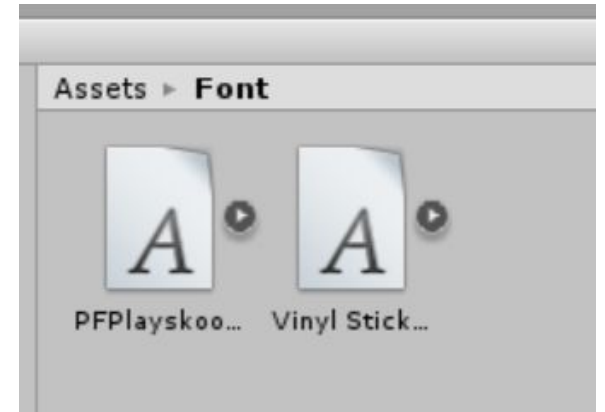
Используя шрифт и изображение необходимо разместить этот элемент в верхнем правом (или левом) углу экрана.

Игроку нужно добавить компонент UIManager.cs

Этому компоненту необходимо указать объект Text (не путать с 3D Text!), который был создан ранее.

Часть 3. Интерфейс и подача работы на конкурс

- Используя шрифт и изображение необходимо разместить этот элемент в верхнем правом (или левом) углу экрана.
- Игроку нужно добавить компонент UIManager.cs
- Этому компоненту необходимо указать объект Text (не путать с 3D Text !), который был создан ранее.



Тестирование! По критериям

1. Аккуратность и внимание к деталям - объекты в уровне расставлены ровно, не перекрывают друг друга и прочее - от 0 до 10 баллов.
2. Проходимость - уровень возможно пройти, он не слишком сложный, но и не слишком простой - от 0 до 10 баллов.
3. Интерес - субъективная оценка участника команды жюри - от 0 до 10.

Упаковываем игру!

Assets > Export Package

Сохраняем с названием:

имя и фамилия студента с указанием города. Например:

PetrovVladimir(SPB).unitypackage

Награды победителям

- 1-3 место - по 3 Юшки;
- 4-9 - по две Юшки;
- 10-15 - дополнительно по 1 Юшке;
- все работы победителей будут опубликованы с их согласия.